Escuela Politécnica Superior de Linares

Grado en Ingeniería Mecánica

UNIVERSIDAD DE JAÉN
Escuela Politécnica Superior de Linares

ACTUALIZACIÓN DE MÁQUINA DE VIBRACIONES LIBRES Y FORZADAS DE USO DOCENTE

ESCUELA POLITÉCNICA SUPERIOR DE LINARES

Alumno: José Ángel Culpián López

Tutor: Prof. D. Luis Felipe Sesé
Depto.: Ingeniería Mecánica y Minera

Escuela Politécnica Superior de Linares

FELIPE SESE LUIS ANTONIO - 26242833V

Firmado digitalmente por FELIPE SESE LUIS ANTONIO - 26242833V
Nombre de reconocimiento (DN): c=ES, serialNumber=IDCES-26242833V, givenName=LUIS ANTONIO, sn=FELIPE SESE, cn=FELIPE SESE LUIS ANTONIO - 26242833V
Fecha: 2018.07.23 08:56:16 +02'00'

Culpián López, José Ángel

26259775P

Felipe Sesé, Luis Antonio

26242833V

Septiembre, 2018

Alumno: José Ángel Culpián López

AGRADECIMIENTOS.

Quisiera mostrar mis agradecimientos a todas aquellas personas que han estado a mi lado a lo largo de esta etapa universitaria: familiares, amigos, compañeros de clase y profesores, ya que todos ellos han sido un apoyo académico como emocional muy importante en estos últimos años.

También quisiera dar las gracias al profesor y mi tutor Don Luis Felipe Sesé, por la ayuda aportada, su disponibilidad y los conocimientos que me han trasmitido a lo largo de la realización de este proyecto.

Así como el profesor Don Mario Miró Barnés y a mi compañero Sergio Palomares López, los cuales me ayudaron a fabricar algunas de las piezas necesarias para este trabajo así como soportarme en el taller.

Tabla de contenido

1. RESUMEN.

El movimiento vibratorio u oscilatorio tiene cierta importancia en el ámbito industrial, ya que puede inducir esfuerzos al elemento que se encuentra sometido a vibración.

Con el fin de ilustrar un ejemplo de un movimiento oscilatorio, se desea actualizar una máquina de vibraciones libres y forzadas antigua de la Escuela Politécnica Superior de Linares. Esto es necesario, ya que la máquina está obsoleta y limitada. Con dicha actualización, se pretende dar más prestaciones a la máquina para que pueda ser empleada en prácticas.

Para la actualización, ha sido necesario el cambio del sistema de excitación forzada y sensorización , lo que conlleva también la fabricación de diversos elementos. Además se pretende desarrollar un sistema de control y monitorización de dicha máquina que permita controlarla y extraer datos de manera digital .

Una vez realizada la actualización de dicha máquina se realizó una serie de análisis ilustrativos sobre las características de los resortes así como caracterizar el movimiento oscilatorio. Dicho estudio se centrará en determinar la *constante de rigidez*, los parámetros relacionados con la amortiguación y la *frecuencia de resonancia.* Lo que facilitará la comprensión de estos conceptos por parte de los estudiantes.

2. INTRODUCCIÓN.

2.1. Conceptos básicos sobre vibraciones.

Una vibración es el movimiento que describe una partícula o cuerpo cuando oscila sobre una posición de equilibrio [1]. Dichas vibraciones se producen cuando el sistema se desplaza de su posición de equilibrio, y puede permanecer oscilando de forma indefinida o volver a la posición de equilibrio debido a fuerzas restauradoras [2].

Dentro de los movimientos vibratorios periódicos se pueden apreciar los siguientes conceptos, algunos de los cuales se pueden ver en la figura 2.1:

- Ciclo: es la evolución del sistema entre un instante y el siguiente que presente las mismas características del movimiento.
- Periodo: es el tiempo que tarda un sistema en realizar un ciclo. Se mide en segundos.
- Frecuencia: es el número de ciclos que realiza el sistema o partícula en un segundo. Es la inversa del periodo. Sus unidades son los s^{-1} o *hertzios* (*Hz*).
- Amplitud: distancia máxima recorrida por el sistema con respecto a la posición de equilibrio.
- Posición de equilibrio: es la posición que adopta el sistema cuando permanece en reposo. Por tanto, se producirá *desequilibrio* cuando el sistema permanece fuera de dicha posición.

Figura 2.1. Representación de un movimiento armónico en función del tiempo.

Dependiendo a las fuerzas a las que estén sometidas el sistema, las vibraciones se pueden clasificar en libres y forzadas.

Las vibraciones libres son aquellas en las que no interviene ninguna fuerza externa durante la vibración. Este tipo de vibraciones se produce cuando el conjunto se desplaza de su posición de equilibrio y describe oscilaciones hasta que el sistema se estabilice. Un ejemplo de vibración libre, se puede apreciar en la figura 2.2. En dicha imagen se puede apreciar como la amplitud de las oscilaciones disminuye a medida que pasa el tiempo.

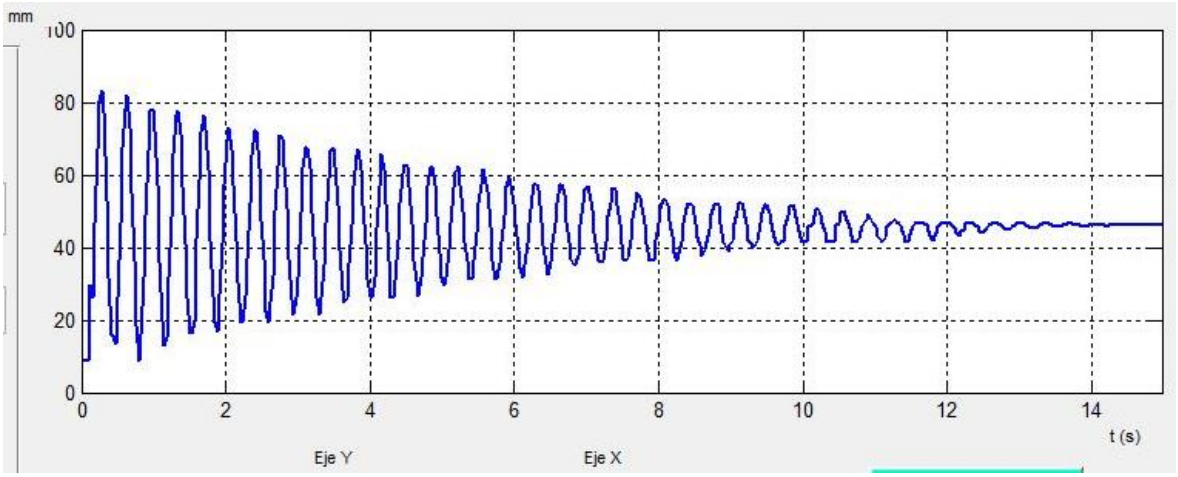


Figura 2.2. Ejemplo de vibración libre,

Las vibraciones forzadas ocurren cuando el sistema es sometido a fuerzas periódicas externas que lo obligan a oscilar. Se presenta un movimiento similar al que presenta la Figura 2.3. En esta imagen se observa como las oscilaciones del sistema se mantienen constantes. Esta situación se mantendrá así mientras las fuerzas de excitación estén actuando.

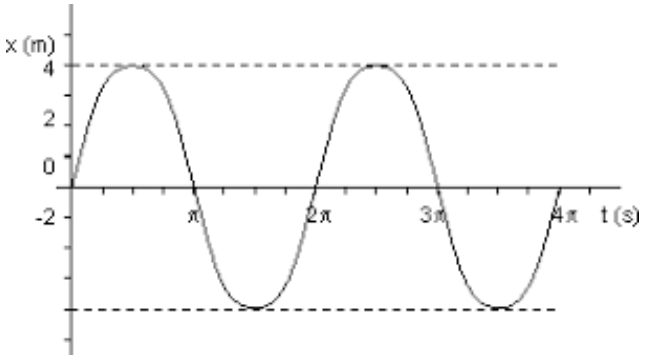


Figura 2.3. [33] Representación de una vibración forzada en función del tiempo.

Dentro de este grupo de vibraciones, se pueden encontrar:

- Vibraciones aleatorias: son aquellas que presentan una amplitud y frecuencia variable [34]. Presentan una oscilación más compleja.
- Vibraciones senoidales: tienen un comportamiento cercano a una función senoidal, como la que aparece representada en la Figura 2.1 o 2.3.

Dichas vibraciones son importantes en ingeniería, ya que se presentan en máquinas debido a desequilibrios, irregularidades en el par motor, flujo de agua... [3].

2.2. VIBRACIONES EN INGENIERÍA.

Desde el punto de vista práctico, es importante tener en cuenta el hecho de que pueden existir vibraciones en máquinas. En la mayor parte de los casos no son deseados y su presencia supone la aparición de esfuerzos, que a la larga pueden ocasionar desperfectos, tales como desgastes de piezas, roturas (Figura 2.4.) o mal funcionamiento de la máquina.

Figura 2.4. Colapso del puente de Tacoma Narrows, por la vibración producida por el viento.

Aparte, hay diversos factores que acentúan la presencia de vibraciones, como desgastes de las piezas de la máquina, imprecisión en la fabricación, fuerzas electromagnéticas periódicas o desequilibrios térmicos [3]. Además, es importante el estudio de las vibraciones, sobre todo los últimos años, ya que hay tendencia en fabricar máquinas de alta velocidad y

estructuras muy ligeras [1]. Otro factor a tener en cuenta, es que las vibraciones pueden afectar a la salud humana, las cuales dependen de la frecuencia de vibración [17].

2.2.1. Patologías producidas por vibraciones.

Dependiendo de la frecuencia de vibración, se pueden presentar las siguientes patologías:

- Vibraciones de muy baja frecuencia: generan sensaciones de mareo.
- Vibraciones de baja frecuencia: afectan a la columna vertebral, sistema digestivo, a la visión, función respiratoria y función cardiovascular. El tipo de patologías que se producen con este tipo de vibraciones son lumbalgias, falta de visión por resonancia, hernias, alteraciones del equilibrio...
- Vibraciones de baja frecuencia: dan lugar a trastornos osteoarticulares. Algunos ejemplos son artrosis del codo, lesiones de muñeca varias, calambres o el aumento de incidencia de enfermedades estomacales.

Por tanto es importante cuantificar las vibraciones.

2.2.2. Cuantificación de vibraciones.

En el análisis de vibraciones, primero se realiza la detección y luego la medición. El punto de la detección se realiza examinando todos los componentes y equipos de la empresa. Si no se realiza bien dicha inspección, existe el hecho de que los operarios que traten con dichos elementos sufran las patologías anteriormente expuestas [17].

El siguiente punto consiste en la medición de las vibraciones. Dicha medición se realiza con un vibrómetro como el de la Figura 2.5, que consta de un acelerómetro que transforma los desplazamientos sufridos por la superficie debido a la vibración, en señales eléctricas. Las mediciones se realizan colocando el acelerómetro en la superficie a estudiar, de forma rígida y sin holguras. También se debe tener en cuenta el punto donde se realiza la medición, ya que tiene que ser el punto representativo, así donde se trasmite la vibración del equipo al operario [17].

Figura 2.5. Vibrómetro utilizado en la medición de vibraciones.

Una vez realizada la medición, se interpretan los resultados mediante el asunto de diversas técnicas, y por tanto conocer el estado de la máquina [18]. Las dos técnicas más utilizadas son las siguientes [18]:

- Análisis de frecuencia: se analiza en base al espectro obtenido en la señal de vibración. La gráfica de dicho espectro presenta en el eje *X* los valores de la frecuencia, mientras que en el eje *Y* es usual que en máquinas rotatorias aparezca la velocidad.
- Análisis de tiempo: Se utilizan cuando hay impactos, holguras y en máquinas de baja velocidad y cajas de cambio.

3. OBJETIVOS.

Tras haber presentado la importancia del fenómeno de las vibración, el objetivo principal del presente trabajo es la creación de herramientas que ayuden a entender el movimiento vibratorio. Dichas herramientas se dividen en *máquinas de estudio,* que en este caso es una máquina de vibraciones libres y forzadas. Por otro lado se encuentran las *herramientas de control y procesado*, constituido por elementos de control (por ejemplo, un sensor de posición), así como un programa de procesado de la señal.

Estos objetivos se pretenden realizar mediante la actualización de una máquina de vibraciones disponible en el laboratorio de Ingeniería Mecánica. Esta actualización busca añadir herramientas de control y procesado a la máquina de vibraciones libres y forzadas del laboratorio. Esto se realiza con el fin de proporcionar a la máquina, varias formas de realizar un estudio del movimiento oscilatorio.

Dicha actualización es necesaria ya que el estado que presentaba la maquina era es muy obsoleto y simple.

Por tanto, ese objetivo constará a su vez de diferentes tareas. En concreto se pretende dotar a la máquina de un sensor de posición, de forma que registre los datos y que puedan ser procesados (realizando gráficas de posición en función del tiempo, así como presentar la señal en dominio de la frecuencia). También se busca cambiar el motor por otro que tenga una velocidad de giro controlable. Estos elementos se desean controlar mediante un programa informático.

Finalmente, para conseguir el objetivo principal, se pretende realizar un estudio del movimiento vibratorio, usando dicha máquina para que, además de comprobar y caracterizar su comportamiento, sirva como ejemplo para la realización de prácticas de las diferentes asignaturas de Ing. Mecánica.

Los aspectos a estudiar con la máquina son principalmente tres: la constante de rigidez del resorte que tiene la máquina (analizado en Diseño de Máquinas), los parámetros de la amortiguación que presenta el movimiento vibratorio en el caso de vibraciones libres y la frecuencia de resonancia del resorte tanto para vibraciones libres como forzadas, siendo estos últimos parámetros estudiados en Cinemática y Dinámica de Máquinas

Como todo análisis técnico, también se realizará comparaciones de estos valores experimentales con valores obtenidos de forma teórica. Adicionalmente, para el caso de la constante de rigidez, se realizará una simulación para obtener otro punto de comparación.

4. FUNDAMENTOS.

En el presente apartado se explicarán cuestiones tales como descripción general sobre resortes, la constante de rigidez de un muelle, parámetros sobre amortiguación, la frecuencia de resonancia, la trasformada de Fourier y se expondrá una presentación general de los objetos utilizados en la actualización.

4.1. RESORTES.

Un resorte es un elemento mecánico que se deforma elásticamente y recupera su forma original cuando la fuerza que lo deforma desaparece. Es un elemento que tiene diversos usos, como por ejemplo ejercer fuerzas, proporcionar flexibilidad o almacenar energía [4].
Hay diversos tipos de resortes, los más utilizados son los siguientes:

- Resortes de compresión (Figura 4.1): este tipo de resortes se utilizan para mantener una fuerza entre dos superficies o ejercer una fuerza de resistencia. Se diseñan de forma que puedan ser insertados en agujeros o en un eje, por ejemplo, el muelle de un bolígrafo [19].

Figura 4.1. [13] Resortes de compresión.

Los resortes de compresión pueden presentar cuatro tipos de extremos: plano, plano y esmerilado, a escuadra y cerrado y a escuadra y esmerilado [4]. Todos estos extremos se

representan en la Figura 4.2. Los extremos más simples son los planos, ya que no tienen ninguna geometría especial. Los extremos cerrados, se caracterizan por tener un espacio reducido entre las espiras finales, de forma que el extremo de la última espira entra en contacto con la espira penúltima. Cuando se indica que están esmerilados, significa que la espira del extremo se encuentra limada, de forma que resulte totalmente plano a la superficie de contacto.

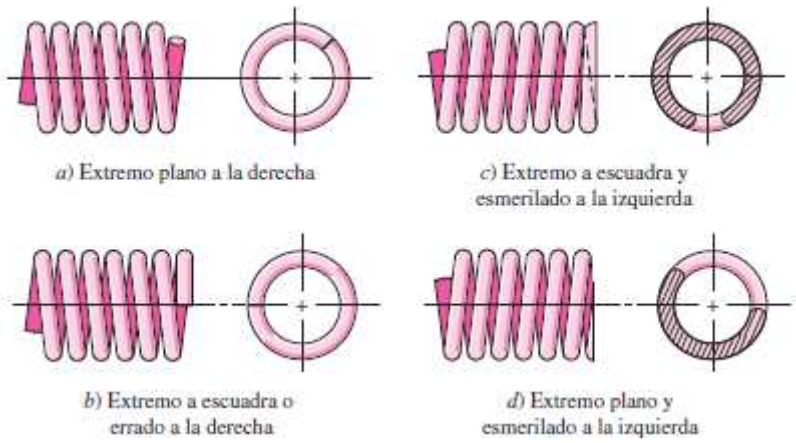


Figura 4.2 [4]. Tipos de extremos.

El alambre de estos resortes, se encuentran sometidos a esfuerzos de torsión, a esfuerzos de tracción y a esfuerzo cortante ya que la fuerza axil de compresión torsiona el alambre, así como traccionarlo . Por tanto, el alambre se encuentra sometido a esfuerzo cortante [4].

- Resortes de extensión (Figura 4.3): este tipo de resortes se utilizan para resistir fuerzas de tracción. Físicamente, difieren de los resortes de compresión en los extremos, ya que estos tienen forma de ganchos. La geometría de los ganchos puede ser muy variada.

Figura 4.3. [20] Resortes de extensión.

Los esfuerzos que aparecen en el cuerpo del resorte son los mismos que aparecen en los resortes de compresión, esfuerzos por torsión, por esfuerzos axil y cortante. La diferencia se encuentra en los ganchos. En un gancho *simple*, se estudia dos partes del mismo: la espira trasversal (espira *A* de la Figura 4.4a) y la espira longitudinal (espira *B* de la Figura 4.4b). La espira trasversal se encuentra sometido a un esfuerzo de tracción (ya que estira la espira) y a esfuerzo de flexión (la fuerza se aplica a una distancia del centro de la espira). La espira longitudinal, se encuentra sometida a esfuerzos de torsión, ya que está junto al cuerpo del resorte [4].

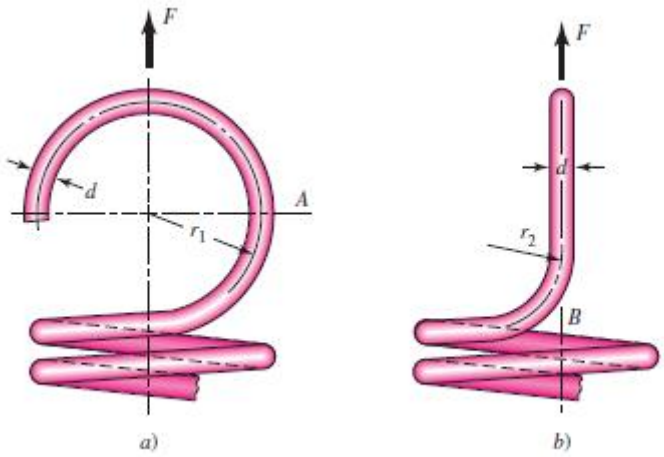


Figura 4.4. [4] Alzado y perfil de un gancho.

Este tipo de resortes, pueden tener su aplicación en mecanismos de ventanas o en la puerta del maletero de un coche.

- Resortes de torsión (Figura 4.5): es un tipo de resorte helicoidal, pero en sus extremos están diseñados de tal forma que se le aplique torsión. Este tipo de resortes se utiliza cuando es necesario mantener dos superficies con una determinada orientación. También se utiliza para que quede alojado en un mandril, ya que cuando se aplica torsión en los extremos, el diámetro interior aumenta y permite ser quitado o desplazado. Cuando se deja de aplicar torsión, el diámetro interior vuelve a su dimensión original, que suele estar diseñado para que coincida con el diámetro del mandril [4].

Figura 4.5.[14]

En un resorte de torsión, las espiras se encuentran sometidas a esfuerzos flexionantes, ya que la fuerza comparte plano con la espira [4].

Este tipo de resortes se pueden encontrar en amortiguadores de puertas, son fácilmente visibles en pinzas de la ropa o se usan para retener las pesas en las barras de los gimnasios.

Es importante tener en cuenta el tipo de esfuerzos a los que se encuentran sometidos estos tipos de resortes. Dependiendo del tipo de esfuerzo al que se encuentren sometido, la *Energía de deformación* (es decir, la energía a aplicar para deformar un cuerpo [7]) tendrá un valor u otro. Dicho valor es necesario para determinar la Constante de rigidez del resorte.

Los esfuerzos a los que se encuentran los distintos tipos de resortes se recogen en la Tabla 4.1.

Tipo de resorte	Parte del resorte	Esfuerzos
Compresión	Cuerpo	Torsión, axil y cortante
Extensión	Cuerpo	Torsión, axil y cortante
	Espira longitudinal	Torsión
	Espira trasversal	Axil y flector
Torsión	Cuerpo	Flector

Tabla 4.1. Resumen de los esfuerzos encontrados en resortes.

4.1.1. Constante de rigidez del resorte.

Según la *Ley de Hooke*, la Constante de rigidez del resorte (k) es la relación entre la fuerza ejercida y la deformación que sufre el resorte:

$$k = \frac{F}{\delta} \quad (4.1)$$

Donde δ es la diferencia entre la longitud inicial del resorte y la longitud del resorte cuando se encuentra deformado bajo carga.

De forma teórica, *k* se calcula dividiendo la carga aplicada entre la deflexión, es decir, se usa la expresión 4.1. Para calcular la deflexión se aplicará el *Teorema de Castigliano.* Dicho teorema indica que "*cuando actúan fuerzas sobre sistemas elásticos sujetos a desplazamientos pequeños, el desplazamiento correspondiente a cualquier fuerza, colineal a dicha fuerza, es igual a la derivada parcial de la energía de deformación con respecto a esa fuerza*" [6].

La expresión matemática genérica del correspondiente teorema es la siguiente:

$$\delta_i = \frac{\partial U}{\partial F_i} \quad (4.2)$$

El valor de la *Energía de deformación* (*U*) dependerá del tipo de esfuerzo al que se encuentra sometido el cuerpo a estudiar.

Para esfuerzo flector, *U* tiene la siguiente expresión:

$$U = \int \frac{M^2 dx}{2EI} \quad (4.3)$$

Para torsión:

$$U = \frac{T^2 L}{2GJ} \quad (4.4)$$

Por esfuerzo cortante:

$$U = \frac{F^2 L}{2AG} \quad (4.5)$$

Y para esfuerzo axil:

$$U = \frac{F^2 L}{2AE} \quad (4.6)$$

Se va a calcular la constante de rigidez de un resorte que se encuentra sometido a esfuerzos de torsión y a esfuerzos axil. Esta parte del resorte se encuentra sometido a esfuerzo de torsión y a cortante. Por lo que la energía de deformación presenta el siguiente valor [4]:

$$U = \frac{T^2 L}{2GJ} + \frac{F^2 L}{2AG} \quad (4.7)$$

$$A = \frac{\pi d^2}{4} \quad (4.8)$$

Por tanto la expresión 4.7 queda como sigue:

$$U = \frac{4F^2 D^3 N_a}{d^4 G} + \frac{2F^2 D N_a}{d^2 G} \quad (4.9)$$

Aplicando el *Teorema de Castigliano* (expresión 4.2):

$$\delta = \frac{\partial U}{\partial F} = \frac{8FD^3N_a}{d^4G} + \frac{4FDN_a}{d^2G} \quad (4.10)$$

Donde D es el diámetro medio del resorte, d el diámetro del alambre y N_a el número de espiras activas. El índice del resorte se define como [4]:

$$C = \frac{D}{d} \quad (4.11)$$

Sustituyendo dicho valor y reordenando la expresión 4.10:

$$\delta = \frac{8FD^3N_a}{d^4G} \quad (4.12)$$

Por lo que se tiene la siguiente rigidez:

$$k = \frac{F}{\delta} = \frac{d^4G}{8D^3N_a} \quad (4.13)$$

4.1.2. Parámetros de amortiguamiento.

Para deducir la ecuación sobre el amortiguamiento, se parte de la ecuación del movimiento un sistema como el de la Figura 4.6 que consta de un resorte, masa colgante y amortiguador:

$$m\ddot{x} + c\dot{x} + kx = 0 \quad (4.14)$$

Donde *m* es la masa que cuelga del resorte, *c* la constante de amortiguación y *k* la constante de rigidez del resorte [2].

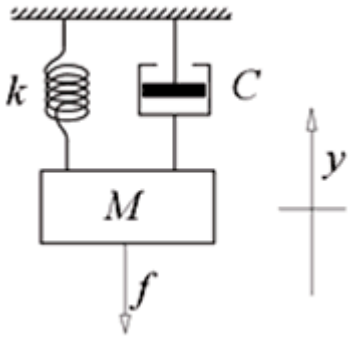


Figura 4.6. [21] Conjunto masa, resorte y amortiguador.

La solución de esta ecuación diferencial es la siguiente:

$$x(t) = Ce^{rt}$$

$$\dot{x} = Cre^{rt} \quad (4.15)$$

$$\ddot{x} = Cr^2e^{rt}$$

Sustituyendo en la ecuación 4.15 en 4.14:

$$(mr^2 + cr + k)Ce^{rt} = 0 \rightarrow mr^2 + cr + k = 0\,;$$

$$r = -\frac{c}{2m} \pm \sqrt{\frac{c^2}{4m^2} - \frac{k}{m}} \quad (4.16)$$

La solución general es de la siguiente forma:

$$x(t) = C_1 e^{r_1 t} + C_2 e^{r_2 t} \quad (4.17)$$

Tomando como factor común r_1, r_2 y $-c/2m$:

$$x(t) = exp\left[-\frac{c}{2m}t\left(C_1 e^{\sqrt{\frac{c^2}{4m^2}-\frac{k}{m}}*t} + C_2 e^{-\sqrt{\frac{c^2}{4m^2}-\frac{k}{m}}*t}\right)\right] \quad (4.18)$$

C_1 y C_2 son dos constantes que dependen de las condiciones iniciales.
La solución se simplificará expresando las ecuaciones de la siguiente forma:

$$r = \pm\sqrt{-\frac{k}{m}} = \pm\, i\omega_n$$

$$x(t) = C_1 e^{i\omega_n t} + C_2 e^{-i\omega_n t} = x(t) = C_1 \cos(\omega_n t) + C_2 \sin(\omega_n t) \quad (4.19)$$

Donde ω_n es la frecuencia natural del sistema.
Dependiendo de las magnitudes de $c^2/4m^2$ y k/m, se pueden presentar tres tipos de movimientos: movimiento subamortiguado, movimiento sobreamortiguado y movimiento críticamente amortiguado. Se pueden apreciar estos tres movimientos en la gráfica comparativa de la Figura 4.7.

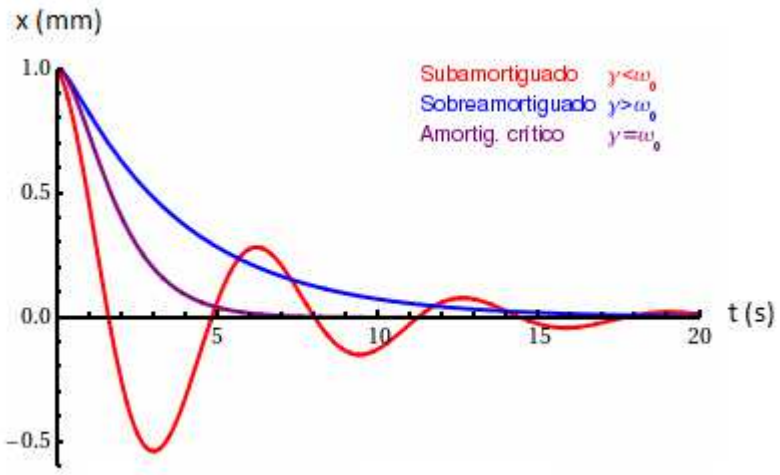


Figura 4.7. [22] Tipos de movimientos oscilatorios.

En el caso del movimiento subamortiguado (es decir presenta oscilaciones que disminuyen su amplitud a medida que pasa el tiempo, ver Figura 4.8), por lo que se tiene que cumplir la siguiente condición:

$$\frac{c^2}{4m^2} < \frac{k}{m} \quad (4.20)$$

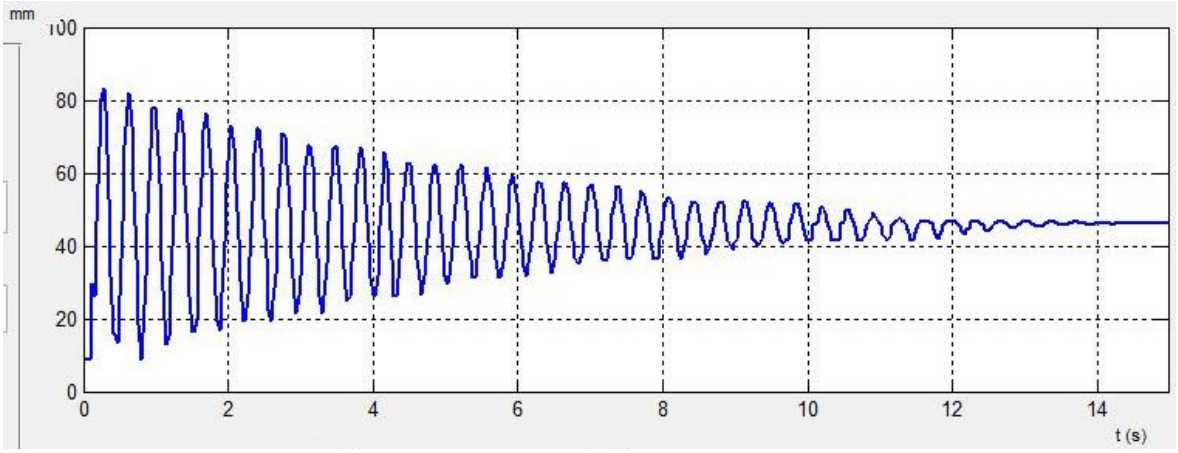


Figura 4.8. Movimiento subamortiguado.

Las soluciones r_1 y r_2 son complejas, por lo que la solución en forma trigonométrica presenta la siguiente forma:

$$x(t) = e^{-\frac{c}{2m}t}[C_1 \cos(\omega_d t) + C_2 \sin(\omega_d t)] \quad (4.21)$$

Donde ω_d es la frecuencia natural del amortiguada, cuyo valor es:

$$\omega_d = \sqrt{\frac{k}{m} - \frac{c^2}{4m^2}} \quad (4.22)$$

Para t = 0 se tiene x (0) = x_0 y $\dot{x}$ (0) = $\dot{x}_0$, sustituyendo en la ecuación de x y para su primera derivada con respecto al tiempo, para t = 0 se tiene:

$$C_1 = x_0$$

$$C_2 = \frac{\dot{x}_0}{\omega_d} + \frac{x_0}{\omega_d}\frac{c}{2m} = \frac{\dot{x}_0}{\omega_d} + \xi x_0 \frac{\omega_n}{\omega_d} \quad (4.23)$$

$$\frac{c}{2m} = \xi \omega_n$$

La ecuación 4.21, en función de las condiciones iniciales quedará de la siguiente forma:

$$x(t) = e^{-\xi \omega_n t}\left[x_0 \cos(\omega_d t) + \left(\frac{x_0 + \xi \omega_n x_0}{\omega_d}\right)\sin(\omega_d t)\right] \quad (4.24)$$

En un movimiento subamortiguado, el movimiento se encuentra limitada por una exponencial que presenta la siguiente forma:

$$e^{-\frac{c}{2m}t} = e^{-\xi\omega_n t} \quad (4.25)$$

Por tanto, la expresión 4.22 queda como sigue:

$$\omega_d = \sqrt{\frac{k}{m}}\sqrt{1-\frac{c^2}{4mk}} = \omega_n\sqrt{1-\left(\frac{c}{c_{cr}}\right)^2} = \omega_n\sqrt{1-\xi^2} \quad (4.26)$$

Dicha expresión relaciona la frecuencia natural amortiguada con la frecuencia natural del sistema:

$$\left(\frac{\omega_d}{\omega_n}\right)^2 + \xi^2 = 1 \quad (4.27)$$

El decremento logarítmico, representa la disminución de amplitud de dos máximos consecutivos [24].

Para calcular el *decremento logarítmico*, se sigue el siguiente procedimiento:

La amplitud inicial (t_0) tiene el valor x_0. Después de un periodo de oscilación, presenta la amplitud x_1, para $t_1 = t_0 + T$ y para dos ciclos se tiene x_2 en $t_2 = t_0 + 2T$. Si suponemos la constante C que depende de x_0 para t_0, se puede expresar de la siguiente forma:

$$x_0 = Ce^{-\frac{c}{2m}t_0}$$

$$x_1 = Ce^{-\frac{c}{2m}(t_0+T)} = x_0e^{-\frac{c}{2m}T} \quad (4.28)$$

$$x_2 = Ce^{-\frac{c}{2m}(t_0+2T)} = x_0e^{-\frac{c}{2m}2T}$$

Para *n* ciclos:

$$x_n = Ce^{-\frac{c}{2m}(t_0+nT)} = x_0 e^{-\frac{c}{2m}nT} \quad (4.29)$$

El valor $\frac{c}{2m}T = \delta$ es el decremento logarítmico. Por tanto, la expresión 4.29 queda de la siguiente forma:

$$x_n = x_0 e^{-n\delta} \quad (4.30)$$

Despejando δ:

$$\delta = \frac{\ln\left(\frac{x_0}{x_n}\right)}{n} \quad (4.31)$$

La relación de amortiguamiento (ξ), es la razón entre el amortiguamiento y el amortiguamiento crítico [24]:

$$\xi = \frac{c}{c_{cri}} = \frac{c}{2m\omega_n} \quad (4.32)$$

Para relacionarlo con la relación de amortiguamiento, se hace lo siguiente:

$$T = \frac{2\pi}{\omega_d} \quad (4.33)$$

$$\delta = \frac{c}{2m}T = \frac{c}{2m}\frac{2\pi}{\omega_d} = \frac{c}{2m}\frac{2\pi}{\omega_n\sqrt{1-\xi^2}};$$

$$\delta = \frac{2\pi}{\omega_n\sqrt{1-\xi^2}}\frac{c}{2m}\sqrt{\frac{m}{k}};$$

$$\delta = \frac{2\pi\xi}{\sqrt{1-\xi^2}} \quad (4.34)$$

El coeficiente de amortiguamiento, se calcula a partir de una de las definiciones de la relación de amortiguamiento (expresión 4.32). El coeficiente de amortiguamiento se obtiene despejándola de dicha expresión:

$$C = 2\xi\sqrt{km} \quad (4.35)$$

Para el movimiento sobreamortiguado, se produce cuando se tiene la siguiente condición [22]:

$$\frac{c^2}{4m^2} > \frac{k}{m} \quad (4.36)$$

En este caso el término ω_d presenta la siguiente forma:

$$\omega_d = \sqrt{\frac{c^2}{4m^2} - \frac{k}{m}} \quad (4.37)$$

Por tanto, la solución a la ecuación 4.14 es la siguiente:

$$x(t) = Ce^{\left(-\frac{c}{2m}+r\right)t} + Ce^{\left(-\frac{c}{2m}-r\right)t} \quad (4.38)$$

Siendo *r*:

$$r = \frac{c^2}{4m^2} \pm \sqrt{\frac{c^2}{4m^2} - \frac{k}{m}} \quad (4.39)$$

Este tipo de movimiento se caracteriza en que las oscilaciones decaen con el tiempo. Esto queda demostrado con la ecuación 4.38, ya que tiene exponentes negativos por tanto se obtiene una función decreciente [22].
Y para un movimiento críticamente amortiguado, se tiene la siguiente condición [24]:

$$\frac{c^2}{4m^2} = \frac{k}{m} \quad (4.40)$$

Por tanto se tiene que la frecuencia natural amortiguada es cero. Y la solución a la ecuación 4.14 es la siguiente:

$$x(t) = e^{-\frac{c}{2m}t}(C_1 + C_2 t) \quad (4.41)$$

Todos los movimientos anteriormente mencionados, aparecen en la Figura 4.8. Se puede apreciar que el movimiento que más tarda en amortiguarse, es el subamortiguado ya que en este caso, hay menos rozamiento. Mientras que el que menos tarda en estabilizarse es el

críticamente amortiguado. Esto es así debido a que el amortiguamiento es proporcional a la velocidad, y el caso del movimiento críticamente amortiguado es donde mayor velocidad hay [22].

4.1.3. Frecuencia de resonancia.

Si un sistema es sometido a una fuerza que lo hace oscilar y la frecuencia de oscilación es cercana o igual a la frecuencia natural del conjunto, la amplitud de la oscilación será máxima. Bajo estas condiciones, el sistema se encuentra bajo el fenómeno de Resonancia [23].
Se entiende por frecuencia natural (mencionada en el párrafo anterior), como la frecuencia a la que seguiría vibrando el sistema mecánico cuando no tiene fuente de excitación. Se diferencia de la frecuencia de resonancia en que esta última no hay amortiguación [35].
Para el caso de una masa colgada en el extremo de un resorte , la frecuencia de resonancia se obtiene a partir de la ecuación del sistema:

$$\sum F = ma = m\ddot{x} \quad (4.42)$$

En ese caso, la única fuerza existente es la que soporta el resorte:

$$F = kx \quad (4.43)$$

Por tanto, queda de la siguiente forma:

$$m\ddot{x} + kx = 0 \quad (4.44)$$

Dividiendo ambos términos por la masa:

$$\ddot{x} + \frac{k}{m}x = 0;\ \ddot{x} + {\omega_n}^2 x = 0 \quad (4.46)$$

Donde el término ω_n es la *frecuencia natural del sistema* en rad/s:

$$\omega_n = \sqrt{\frac{k}{m}} \quad (4.47)$$

Para obtenerla en hertzios (Hz) a la ecuación 4.47 se divide entre 2π, ya que un ciclo equivale a 2π radianes:

$$\omega_n = \frac{1}{2\pi}\sqrt{\frac{k}{m}} \quad (4.48)$$

Ya que un ciclo equivale a 2π radianes. Dicho valor es la frecuencia de resonancia del sistema constituido por una masa colgada de un resorte.

4.1.4. Diseño mecánico de un resorte.

Antes de utilizar un resorte cualquiera, es importante realizar un diseño del mismo. Es necesario determinar algunos parámetros importantes de los mismos, como por ejemplo la resistencia última (S_{ut}), la resistencia a fluencia de torsión (S_{Sy}) o la resistencia a fluencia (S_y).

La resistencia última, se puede definir como la carga máxima que puede soportar un resorte sin que rompa. Para calcular dicho valor se aplica la siguiente aproximación:

$$S_{ut} = \frac{A}{d^m} \quad (4.49)$$

Donde *d* es el diámetro del alambre del resorte. Los parámetros *A* y *m* están tabulados y dependen del tipo de material con que se fabrica el resorte. Algunos valores de *A* y *m* se ilustran en la Figura 4.9:

Material	ASTM núm.	Exponente *m*	Diámetro, pulg	*A*, kpsi · pulg^m	Diámetro, mm	*A*, MPa · mm^m	Costo relativo del alambre
Alambre de piano*	A228	0.145	0.004-0.256	201	0.10-6.5	2 211	2.6
Alambre T y R en aceite†	A229	0.187	0.020-0.500	147	0.5-12.7	1 855	1.3
Alambre estirado duro‡	A227	0.190	0.028-0.500	140	0.7-12.7	1 783	1.0
Alambre al cromo vanadio§	A232	0.168	0.032-0.437	169	0.8-11.1	2 005	3.1
Alambre al cromo silicio‖	A401	0.108	0.063-0.375	202	1.6-9.5	1 974	4.0
Alambre inoxidable 302#	A313	0.146	0.013-0.10	169	0.3-2.5	1 867	7.6-11
		0.263	0.10-0.20	128	2.5-5	2 065	
		0.478	0.20-0.40	90	5-10	2 911	
Alambre de bronce fosforado**	B159	0	0.004-0.022	145	0.1-0.6	1 000	8.0
		0.028	0.022-0.075	121	0.6-2	913	
		0.064	0.075-0.30	110	2-7.5	932	

Figura 4.9. [25] Valores de *A* y *m*.

La resistencia a fluencia de torsión, se puede definir como la carga máxima que puede soportar un resorte hasta que se deforme el alambre torsionalmente de forma permanente. Dicha resistencia se calcula como un porcentaje de la tensión última. El porcentaje a aplicar depende del tipo del material con que se fabrica el resorte. Los porcentajes para resortes de compresión, se recogen en la Figura 4.10. Normalmente se utilizan los porcentajes de la primera columna.

Material	*Porcentaje máximo de la resistencia a la tensión*	
	Antes de la remoción de la deformación (incluye K_w o K_B)	Después de la remoción de la deformación (incluye K_s)
Alambre de piano y acero al carbono estirado en frío	45	60-70
Acero al carbono templado y revenido y acero de baja aleación	50	65-75
Aceros inoxidables austeníticos	35	55-65
Aleaciones no ferrosas	35	55-65

Figura 4.10. [26] Porcentajes para calcular S_{Sy} en resortes de compresión.

Para resortes de extensión, los porcentajes son los que aparecen en la Figura 4.11:

Materiales	Porcentaje de la resistencia a la tensión		
	En torsión		En flexión
	Cuerpo	Extremo	Extremo
Aceros al carbono y acero de baja aleación pavonado, estirado en frío o templado y revenido	45-50	40	75
Acero inoxidable austenítico y aleaciones no ferrosas	35	30	55

Figura 4.11. [27] Porcentajes para calcular S_{Sy} en resortes de extensión.

La resistencia a fluencia se define como la carga máxima que puede soportar el resorte antes de que se deforme permanentemente. Esta resistencia se considera más en resortes de torsión. Para calcular dicho valor de tensión se aplican los porcentajes que se muestran en la Figura 4.12:

$$S_y = \begin{cases} 0.78 S_{ut} & \text{Alambre de piano y aceros al carbono estirados en frío} \\ 0.87 S_{ut} & \text{Aceros al carbono y de baja aleación T y R en aceite} \\ 0.61 S_{ut} & \text{Acero inoxidable austenítico y aleaciones no ferrosas} \end{cases}$$

Figura 4.12. [28] Valores de S_y.

Una vez calculadas dichas resistencias, resulta interesante calcular las tensiones que sufre el resorte cuando está cargado. Si alguna de estas tensiones supera la resistencia a fluencia, el resorte se deformará. Si superan la resistencia última, el resorte se romperá.

Dependiendo del tipo de resorte, las tensiones se calculan siguiendo distintos procedimientos.

En resortes de compresión, solo hay tensiones en el cuerpo, ya que es la única parte que tiene el resorte. Como se ha indicado antes, este resorte se encuentra sometido a esfuerzos axil de compresión (ya que la carga comprime el resorte) y a esfuerzos de torsión (ya que dicha fuerza de compresión retuerce el alambre). Por tanto, la tensión total del resorte se calcula como sigue [4]:

$$\tau_{máx} = \frac{F}{A} + \frac{Tr}{J} \quad (4.50)$$

Donde F es la carga aplicada en Newtons (N), A es el área transversal del alambre del resorte en mm^2, r es el valor del radio del alambre del resorte en mm, T es el momento torsor que se obtiene multiplicando la carga aplicada por la mínima distancia con respecto al punto al que se quiere estudiar:

$$T = F\frac{D}{2} \quad (4.51)$$

Y J es el momento polar de inercia:

$$J = \frac{\pi}{32} d^4 \quad (4.52)$$

Los términos D y d hacen referencia al diámetro medio del resorte y al diámetro del alambre respectivamente. Para calcular el diámetro medio se suma el diámetro del alambre al diámetro interno del resorte, o se resta el diámetro del alambre al diámetro externo del resorte:

$$D = D_{in} + d = D_{ext} - d \quad (4.53)$$

Sustituyendo estos valores en la ecuación 4.50 se tiene:

$$\tau_{máx} = \frac{4F}{\pi d^2} + \frac{8FD}{\pi d^3} \quad (4.54)$$

Definiendo la constante de resorte (Ecuación 4.11) y sustituyendo en 4.54 se tiene la tensión sufrida por el resorte:

$$\tau = K_B \frac{8FD}{\pi d^3} \quad (4.55)\ [4]$$

El término KB, es el factor de Bergsträsser. Con dicho factor se corrige la curvatura del alambre. Se calcula de la siguiente forma:

$$K_B = \frac{4C+2}{4C-3} \quad (4.56)\ [4]$$

Para un resorte de extensión se deben estudiar tres tensiones: la tensión sufrida en el cuerpo, la tensión sufrida en la espira longitudinal y la tensión en la espira transversal.
En el cuerpo del resorte se tienen los mismos esfuerzos que en un resorte de compresión [4]. Por tanto, la tensión soportada por el cuerpo se calcula con la expresión 4.55.
Para la espira longitudinal (ver Figura 4.4b) los esfuerzos son también de torsión y tracción. Entonces, su valor es el siguiente:

$$\tau = (K)_B \frac{8FD}{\pi d^3} \quad (4.57)\ [4]$$

Donde el término (K)B es otro factor de concentración de esfuerzo por curvatura [4]:

$$(K)_B = \frac{4C_1-1}{4C_1-4} \quad (4.58)$$

$$C_1 = \frac{2r_2}{d} \quad (4.59)$$

Donde d es el diámetro del resorte y r2 es el radio de la espira longitudinal (Figura 4.4b). Para la espira trasversal (Figura 4.4a) que se encuentra sometida a flexión y a tracción, la tensión se calcula con la siguiente fórmula:

$$\sigma_A = F\left[\frac{4}{\pi d^2} + (K)_A \frac{16D}{\pi d^3}\right] \quad (4.60)\ [4]$$

$$(K)_A = \frac{4C_2^2 - C_2 - 1}{4C_2(C_2 - 1)} \quad (4.61)$$

$$C_2 = \frac{2r_1}{d} \quad (4.62)$$

Y para el caso de los resortes de torsión, se estudian la tensión que hay en el cuerpo, que está sometido a esfuerzos de flexión. Por tanto se tiene:

$$\sigma = K_i \frac{32Fr}{\pi d^3} \quad (4.63)\ [4]$$

$$K_i = \frac{4C^2 - C - 1}{4C(C - 1)} \quad (4.64)$$

4.2. TRASFORMADA DE FOURIER.

El análisis de Fourier se utiliza para determinar la amplitud y la fase de cada una de las componentes de frecuencia de una señal. Para el caso de señales periódicas se usan las Series de Fourier, mientras para señales no periódicas se utilizan las Trasformadas de Fourier [36]. Uno de los usos más importantes de la Trasformada de Fourier, es que realiza la trasformación de una señal en dominio del tiempo al dominio de frecuencia [36].

Otro uso es que se puede realizar el estudio de como cambia la amplitud de una señal sinusoidal, cuando pasa a través de un sistema lineal invariante en el tiempo [36].

Fourier defiende que una señal compleja se puede obtener como la suma de señales más simples [15]. Un ejemplo de esta suma aparece en la Figura 4.13, donde la señal roja es el resultado de sumar la función sen(x) con 0,5*sen(2x).

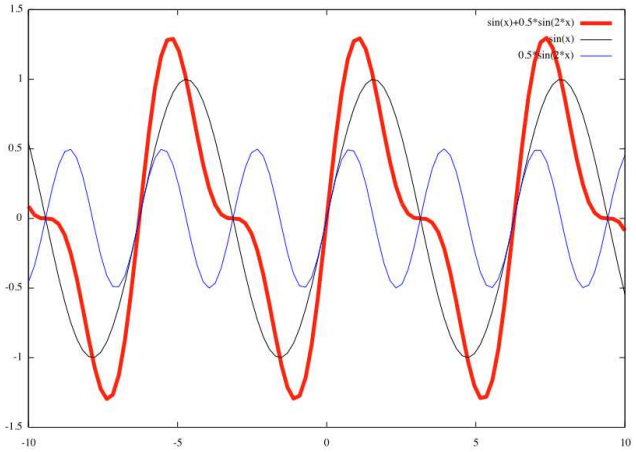


Figura 4.13. [16] Suma de señales simples.

Es importante indicar, que cuando se analiza una señal en frecuencias, se considera la información más importante la de la señal simple con la amplitud mayor [15].

La expresión básica de la trasformada de Fourier es la siguiente [15]:

$$F\left(\frac{n}{NT}\right) = \frac{1}{N}\sum_{k=0}^{N-1} m(kT)e^{-j\frac{2\pi nk}{N}} \quad (4.65)$$

Donde n = 0,1...N-1. N es el número de muestras, T es el periodo de muestreo, n es el valor de frecuencia a obtener y m(kT) es la muestra tomada en el instante kT, siendo k el número de muestra.

El valor de n determina la frecuencia que se va a descomponer la señal de partida. Por tanto, para estudiar todas las frecuencias, se estudia todo el rango de n [15].

Si se deja el término exponencial de la expresión 4.65 en términos seno y coseno, se queda de la siguiente forma:

$$e^{-j\frac{2\pi nk}{N}} = cos\left(-2\pi n\frac{k}{N}\right) + j * sin\left(-2\pi n\frac{k}{N}\right) \quad (4.66)$$

De forma que el sumatorio de la expresión 4.65 queda como sigue:

$$F\left(\frac{n}{NT}\right) = \frac{1}{N}\sum_{k=0}^{N-1} m(kT)\left[cos\left(-2\pi n\frac{k}{N}\right) + j * sin\left(-2\pi n\frac{k}{N}\right)\right] \quad (4.67)$$

La expresión 4.81 es la trasformada de Fourier expresada en términos de seno y coseno.

4.3. EQUILIBRADO.

Muchas de las piezas de máquinas tienen movimientos alternativos, giratorios ... Si dichos elementos no se encuentran debidamente equilibrados, se presentan fuerzas de inercia (que son aquellas fuerzas que se oponen al movimiento) que tienden a producir vibraciones en las máquinas [37].

El objetivo del equilibrado consiste en disminuir o evitar los efectos perjudiciales producidos por la vibraciones [37].

Hay dos tipos de equilibrios: estático y dinámico.

- Equilibrio estático: es el equilibrio que se presenta cuando el sistema permanece en reposo, es decir parado [37]. La expresión que define dicho equilibrio es la siguiente:

$$\sum F = 0 \quad (4.68)$$

Para momentos ejercidos por fuerzas, se tiene una expresión similar:

$$\sum M = 0 \quad (4.69)$$

Dicha expresión, indica que al aplicar diversas fuerzas en un punto concreto no se produce ningún efecto, es decir sigue parado.

- Equilibrio dinámico: se produce cuando las fuerzas de inercia y los momentos ejercidos por las masas del sistema están en equilibrio. Este equilibrio se expresa a través de la Segunda Ley de Newton:

$$\sum F = m * a \quad (4.70)$$

Para el momento de una fuerza:

$$\sum M = I * \alpha \quad (4.71)$$

Donde I es el momento de inercia y α la aceleración angular.

Si no se cumplen estas condiciones, el sistema se encuentra en desequilibrio.

A continuación, se estudiará el caso de un sistema compuesto por un árbol y una masa excéntrica giratoria.

El efecto de las masas giratorias es producir fuerzas centrífugas sobre al árbol al que se encuentran unidas. El equilibrado del sistema consiste en la reorganización de las masas de forma que las fuerzas que originan se equilibren entre sí, dando lugar al equilibrado del sistema [37].

Si un árbol gira a una velocidad angular constante y tiene una masa excéntrica, cuyo centro de gravedad se encuentra a una distancia r_1 del árbol, esta generará una fuerza centrífuga de valor:

$$F_1 = m_1 r_1 \omega^2 \quad [37] \ (4.72)$$

Dicha fuerza da lugar a la aparición de vibraciones. Para contrarrestar el efecto de dicha carga, se debe colocar otra masa excéntrica diametralmente opuesta:

$$m_1 r_1 \omega^2 = m_2 r_2 \omega^2 \quad [37] \ (4.73)$$

Este conjunto de dos masas excéntricas opuestas, da lugar al equilibrio estático y dinámico. Esto es debido a que el centro de gravedad del conjunto se encuentra sobre el eje del árbol, por tanto dicho centro permanece fijo y con respecto a la máquina no supone ningún cambio [37].

4.4. ELEMENTOS DE CONTROL.

En la rehabilitación de la máquina se han utilizado diversos elementos:

- Tarjetas de programación.
- Sensores.
- Motor eléctrico.

4.4.1. Tarjetas de programación.

Estas tarjetas, se utilizan para poder crear sistemas embebidos. Los sistemas embebidos sirven para cubrir necesidades específicas.

Hay diversos tipos de tarjetas, las cuales se organizan en dos grupos: placas con microcontroladores y placas minicomputador [38].

Las tarjetas con microcontrolador se caracterizan por tener un microcontrolador y por no tener un sistema operativo. Su aspecto es como se ilustra en la Figura 4.14. Sólo tienen un lenguaje de programación [38].

Figura 4.14. Ejemplo de una tarjeta con microcontrolador.

Dentro de este grupo se encuentran los siguientes ejemplos:

- Arduino: se puede relacionar con el entorno mediante el uso de sensores u otros sistemas de control conectados a través de pines de entradas. En función de lo que se programe, la señal que tenga de las entradas las tratará y mandará señales eléctricas a los pines de salida, donde se encuentran conectados los actuadores[5]. Dicha placa presenta las ventajas tales como la simplificación en la programación y trabajo con microprocesadores. Resultan más baratas que otras placas con microcontroladores. Se puede utilizar para diversos sistemas operativos, ya que la mayoría solo se pueden usar en Windows. Por otro lado, un inconveniente que presenta es que al usar librerías, las instrucciones que contienen pueden tardar un poco más (cuestión de microsegundos), que puede afectar a la toma de datos. Otro inconveniente es, que al venir la plataforma ensamblada, se tienen que adaptar los espacios en función de las dimensiones de la placa [39].

- Launchpad MPS 430: pertenece a *Texas Instrument*. Para controlar motores, leds, conectar a Matlab... presenta las mismas ventajas que Arduino. Además es posible modificar los pines y aumentar los puertos USB. Se programa por medio de Code Composer Studio, lo que puede suponer un problema ya que se trata de un lenguaje ensamblador (lenguaje muy similar al que utiliza el microprocesador). Este aspecto hace difícil tratar con esta placa al principio. Para solucionar este problema, se usa la herramienta *Energia*, convirtiendo el panorama de trabajo igual que el de Arduino. Otro inconveniente es que no se encuentra fácilmente información sobre esta placa [40].
- Wiring: su entorno de trabajo está escrito en *Java*. En dicho entorno de trabajo se encuentra un compilador GCC para C/C++. La principal razón para incorporar el lenguaje a C/C++ es para facilitar las operaciones de entrada y salida. Además, dicho entorno de trabajo es igual al de Arduino. La principal desventaja, es la dificultad de encontrar información al respecto de esta placa.

Las placas minicomputadoras, se caracterizan por tener un procesador que son capaces de soportar un sistema operativo [38]. Para hacerse una idea sobre el tipo de placa, se ilustra la Figura 4.15.

Figura 4.15. Tarjeta microcomputadora.

Algunos ejemplos de este tipo de placas son:

- Raspberry Pi: es un computador de placa única o computador de placa simple (SBC) de bajo coste, que tiene un procesador ARM. El Rashberry cuenta con un sistema operativo basado en GNU/LINUX [38]. Una de las ventajas que presenta es su

consumo, ya que este tiene un valor muy bajo (en reposo 0,5 W) [41]. Otra ventaja que presenta, es que puede ejecutar varias versiones de Linux. Además con la poca memoria que presenta es capaz de ejecutar muchas tareas. También resulta muy económico [42]. Como inconvenientes se pueden mencionar que presenta menor procesador y memoria RAM, por lo que limita las tareas. Con el uso se puede llegar a calentar bastante [41].

- ODroid C2: Se encuentra a la par de Raspberry Pi, ya que presenta ventajas similares, pero es más potente y tiene mayor memoria RAM. Además, puede trabajar con Ubuntu o con Android lo que puede levantar gran interés [43]. Su inconveniente, radica en que no tiene conexión Wifi de serie, pero es solucionado mediante un adaptador [44].

4.4.2. Sensores.

Un sensor es un dispositivo que convierte magnitudes físicas (temperatura, humedad, velocidad...) en señales eléctricas.

Para realizar las mediciones, un sensor puede utilizar diversos principios físicos. Según el principio físico que utilice, los sensores se pueden clasificar en:

- Sensores resistivos: utilizan la variación de la resistencia eléctrica (oposición de un material para conducir corriente eléctrica) de un conductor al variar la magnitud física a la que son sensibles. Estos tipos de sensores son los más utilizados, ya que son muchas las magnitudes físicas que producen cambios en el valor de la resistencia eléctrica [45]. Se encuentran en galgas extensiométricas, potenciómetros... Los cuales se encuentran en la Figura 4.16. Un potenciómetro (Figura 4.16a) es un dispositivo con un contacto deslizante. El valor de la resistencia eléctrica es proporcional al valor de la distancia recorrida por el cursor. La galga extensiométrica (Figura 4.16b) se utiliza para medir deformaciones. Se basa en el efecto del cambio de resistencia eléctrica debido a la deformación sufrida.

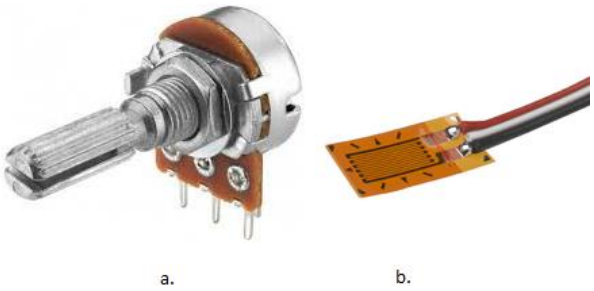


Figura 4.16. Ejemplos de sensores resistivos: 4.16a potenciómetro y 4.16b galga extensiométrica.

- Capacitivos: utilizan la variación de la capacitancia eléctrica (capacidad de almacenar energía eléctrica) al variar la magnitud a la que son sensibles. Dichos sensores están formados por dos electrodos concéntricos de metal formando un condensador eléctrico como el de la Figura 4.17.

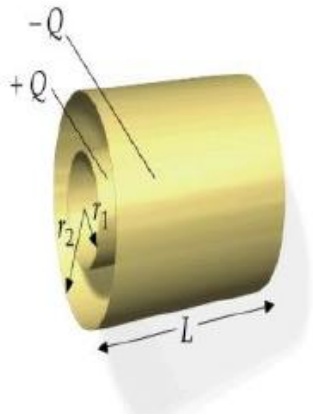


Figura 4.17. Condensador cilíndrico de electrodos concéntricos.

Estos tipos de sensores funcionan de la siguiente forma: el condensador genera un campo eléctrico. Dicho condensador forma parte de un circuito resonador, de forma que cuando un objeto se acerca al sensor e interactúa con el campo eléctrico, el circuito empieza a resonar. Por tanto, la capacitancia aumenta [46].

- Sensores inductivos: en este caso lo que varía es la magnitud de un campo magnético. Generalmente tienen las partes que aparecen en la Figura 4.18. Estos sensores se caracterizan por tener en su interior una bobina que genera un campo magnético.

Cuando pasa un metal conductor cerca del sensor, el campo magnético lo "detecta" [47]. Cuando el campo magnético detecta el conductor, en este debido a la tensión inducida se genera un campo magnético en dicho conductor. El campo magnético producido en el conductor, es comparado por el sensor y si detecta cambios realizará el proceso de detección [47].

Este tipo de sensores suelen ser utilizados para medir posición o desplazamientos.

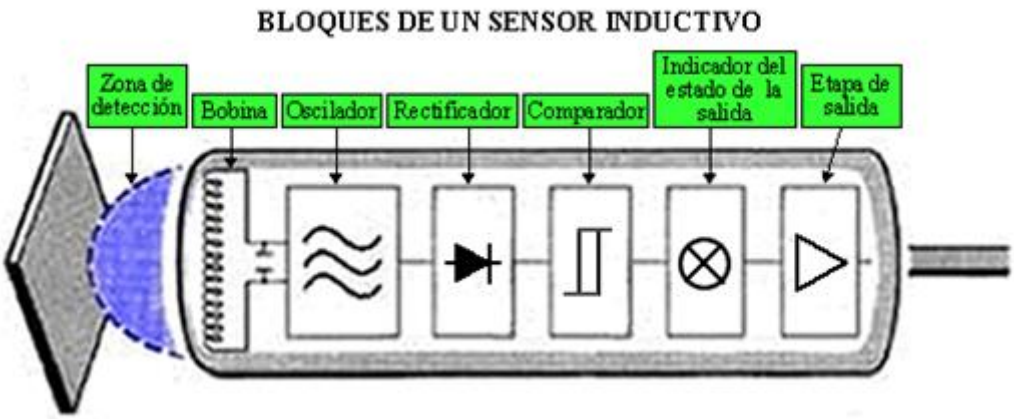


Figura 4.18. [47] Partes de un sensor inductivo.

- Sensores electromagnéticos: utilizan la Ley de Faraday. Dicha Ley indica que se produce corriente eléctrica en una espira cuando hay un campo magnético de magnitud variable que pasa por la espira. Se puede deducir, que mientras el campo magnético permanezca constante o inmóvil, no se producirá tensión eléctrica en la espira.
- Sensores de Efecto Hall: se basan en las corrientes eléctricas que se producen cuando un campo magnético actúa sobre un material conductor, por el cual circula corriente. El efecto que se produce es que en el conductor a detectar sufre una caída de tensión cuando aparece un campo magnético perpendicular al flujo de corriente del conductor [48].
- Sensores piezoeléctricos: aprovechan las *propiedades piezoeléctricas* de los materiales piezoeléctricos, como el cuarzo. Las propiedades piezoeléctricas son dos: cuando a un material piezoeléctrico es sometido a una deformación o a una vibración, en su superficie se genera una diferencia de potencial eléctrica, y por tanto se genera tensión. La segunda propiedad piezoeléctrica, es la inversa de la primera, es decir

sometiendo al material a una tensión, este se deforma o vibra. Suelen ser utilizados para medir fuerza.

Un sensor de posición es aquel que detecta la distancia de un objeto con respecto a un punto de referencia o detectar la presencia de un objeto a cierta distancia.

Se pueden clasificar en los siguientes grupos:

- Sensores de presencia: son sensores que emiten salidas binarias (todo o nada) y mediante esta forma indica la presencia de un objeto ante el detector. Un ejemplo es el detector final de carrera, como el que se muestra en la Figura 4.19. Dicho sensor se activa mediante contacto mecánico.

Figura 4.19. Sensor final de carrera.

- Sensores de distancia o posición: entregan una señal analógica o digital que indica la distancia respecto a un punto de referencia.
- Sensores de pequeños desplazamientos y deformaciones: como su nombre indica, se utilizan para medir pequeños desplazamientos o deformaciones. Se emplean adosados a piezas elásticas o con palpadores. Se utilizan para medir deformaciones, rugosidad, planitud de superficies...

4.4.3. Motores eléctricos.

Un motor eléctrico, es una máquina que convierte la energía eléctrica en energía mecánica. Dependiendo del tipo de corriente, los motores se clasifican en dos grupos:

- Motores de corriente continua: como su nombre indica, se alimentan de corriente continua. Son los más idóneos para el accionamiento a velocidad variable o para tratar con cargas pequeñas. Además, son controladores sencillos y económicos. Como inconvenientes, se pueden mencionar que solo pueden trabajar con potencias reducidas o vida limitada debido a la conmutación de las escobillas.

  El funcionamiento de un motor de corriente continua se basa en la repulsión de los polos magnéticos entre los imanes que se encuentran en el estátor del motor, es decir la carcasa y los electroimanes que hay en el rotor, que es la parte del motor que gira. Ambas partes se encuentran representadas en la Figura 4.20.

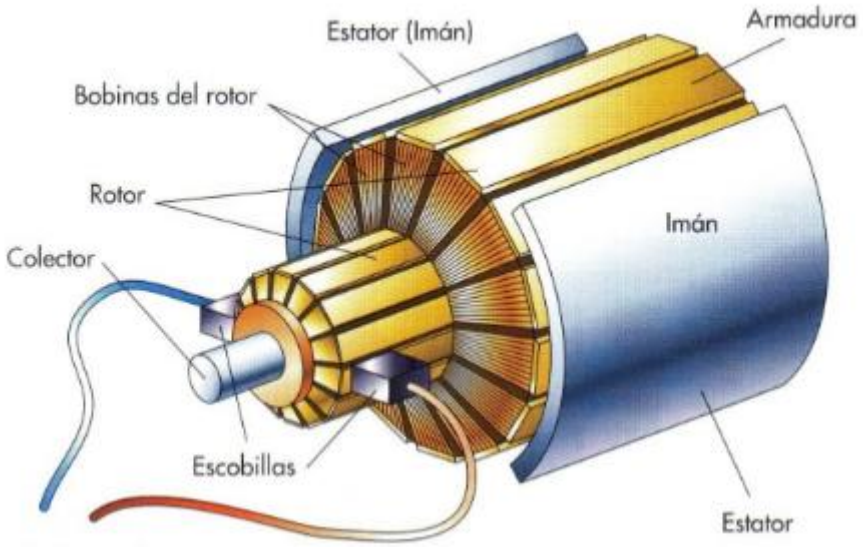


Figura 4.20. [29] Partes de un motor de corriente continua.

Cuando circula corriente eléctrica por los electroimanes del rotor, interactúan con los imanes permanentes. Si los campos magnéticos son del mismo signo, se repelen y dan lugar a un movimiento de giro en el rotor [30].

En el rotor se encuentra un colector que sirve para cambiar el sentido de la corriente, cambiando la polaridad de los electroimanes. Dicho cambio se produce cada media vuelta del rotor. De esta forma, siempre el polo norte del electroimán coincide con el

polo norte del imán permanente y lo mismo ocurre con el polo sur. Por tanto siempre se están repeliendo y generando movimiento [30].

- Motores de corriente alterna: se alimentan de corriente alterna. Las ventajas que presentan son su sencillez y fiabilidad, resultan económicos (ya que no hace falta un gran mantenimiento) y capacidad para trabajar con cargas mayores. El principal inconveniente que presenta estos tipos de motores, es la dificultad que presentan para regular la velocidad de giro.

  Su funcionamiento es similar al de un motor de corriente continua. La diferencia es que tanto en el estátor, como en el rotor hay bobinas y en ambas partes se producen campos magnéticos variables.

Aparte, existen los motores denominados *Paso a paso*. Se denominan así, porque una revolución completa del rotor se traduce en un número de pasos. Como ventajas, presentan su sencillez, fiabilidad y su bajo mantenimiento. Como inconvenientes, solo trabajan a bajas velocidades y siempre necesitan un controlador electrónico.

Existen tres tipos:

- De imán permanente: el rotor está formado por un cilindro magnético con un número determinado de polos magnéticos. En el estator se encuentran las bobinas que crean los polos positivos o negativos de forma secuencial, dando lugar a un par de giro. Un esquema del mismo aparece en la Figura 4.21:

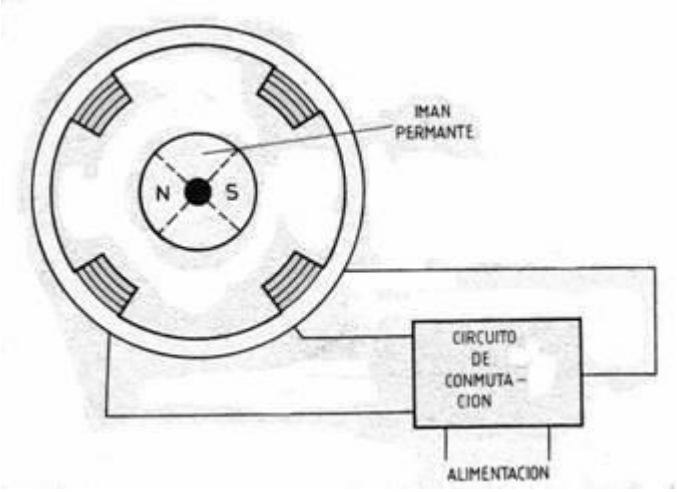


Figura 4.21. [49] Esquema de motor paso a paso de imán permanente.

- Reluctancia variable: el rotor se construye con hierro dulce creando varios salientes. Dichos salientes tratan de alinearse con los del estator para crear el camino de menor reluctancia (resistencia de un circuito al flujo magnético). Como en el caso de los

motores paso a paso de imanes permanentes, la activación secuencial de los motores causa el giro. Una representación del mismo se encuentra en la Figura 4.22:

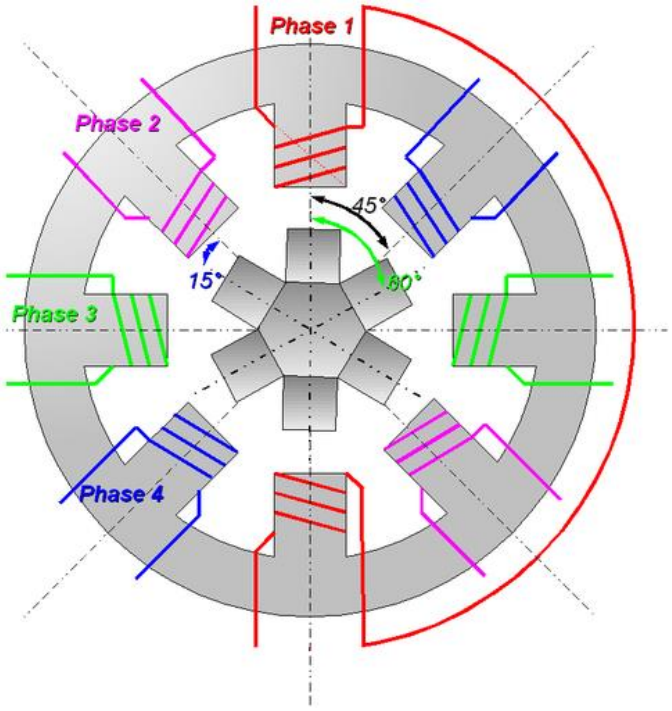


Figura 4.22. [50] Motor de reluctancia variable.

- Híbridos: el rotor se construye con dos imanes permanentes ranurados y magnetizados N-S con un desfase entre ambos. Un esquema del mismo se encuentra en la Figura 4.23:

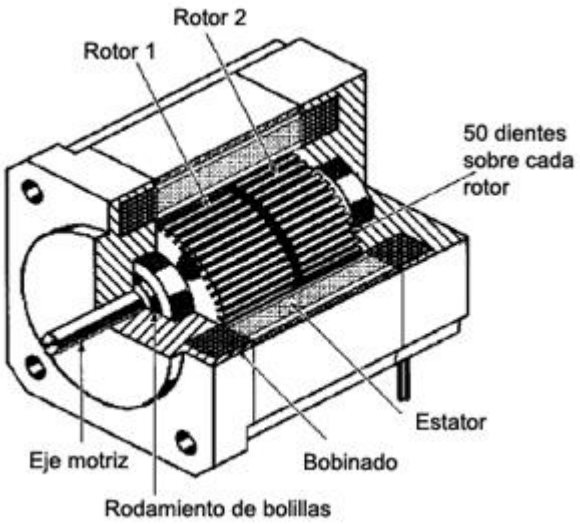


Figura 4.23. [51] Motor paso a paso híbrido.

5. MATERIALES Y MÉTODOS.

Este capítulo consta de dos partes principales. La primera parte explica todos los materiales empleados, tanto para rehabilitar la máquina así como lo que se ha utilizado para realizar los ensayos.
La segunda parte indica la metodología seguida en la rehabilitación de la máquina, además de describir el proceso seguido en los ensayos.

5.1. EQUIPAMIENTO

5.1.1. Máquina de ensayo.

Para poder realizar el estudio del resorte y del movimiento oscilatorio, es necesario rehabilitar la máquina de vibraciones libres y forzadas. El estado inicial aparece representado en la Figura 5.1. La máquina tenía un sistema de registro de oscilaciones muy rudimentario, ya que constaba de una tira de papel enrollado en un cilindro y un lápiz pintaba la oscilación. Además, el motor giraba a una velocidad constante, por lo que hacía muy difícil el estudio de la frecuencia de resonancia del resorte.

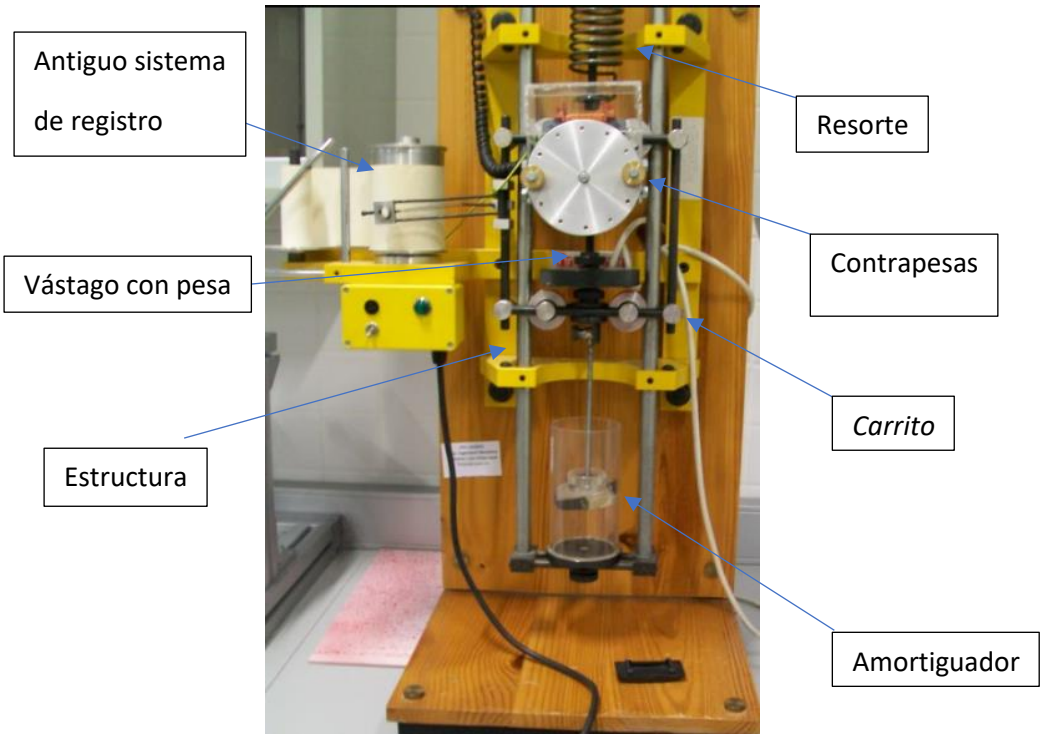


Figura 5.1. Estado inicial de la máquina.

Para facilitar el estudio, se pretende modernizar el sistema de registro de la oscilación con un sensor. De esta forma, es fácil registrar la oscilación en función del tiempo. Otro aspecto que mejoraría dicha modernización sería el hecho de poder controlar la velocidad de giro. Con ello se podría examinar la frecuencia de resonancia del conjunto. Todos estos elementos estarían controlados mediante un software propio y una interfaz de usuario (GUIDE en *Matlab*), que controlaría la placa *Arduino DUE*. Dicha placa controla todos los elementos de la máquina.

5.1.2. Muelle a ensayar.

Una vez rehabilitada la máquina, se ha procedido a estudiar un muelle de *extensión* de acero inoxidable, el cual tiene la geometría que aparece en la Figura 5.2. Por tanto tiene un *Módulo de Young* (*E*) de 193 GPa y un *Coeficiente de elasticidad transversal* (*G*) de 69 GPa.

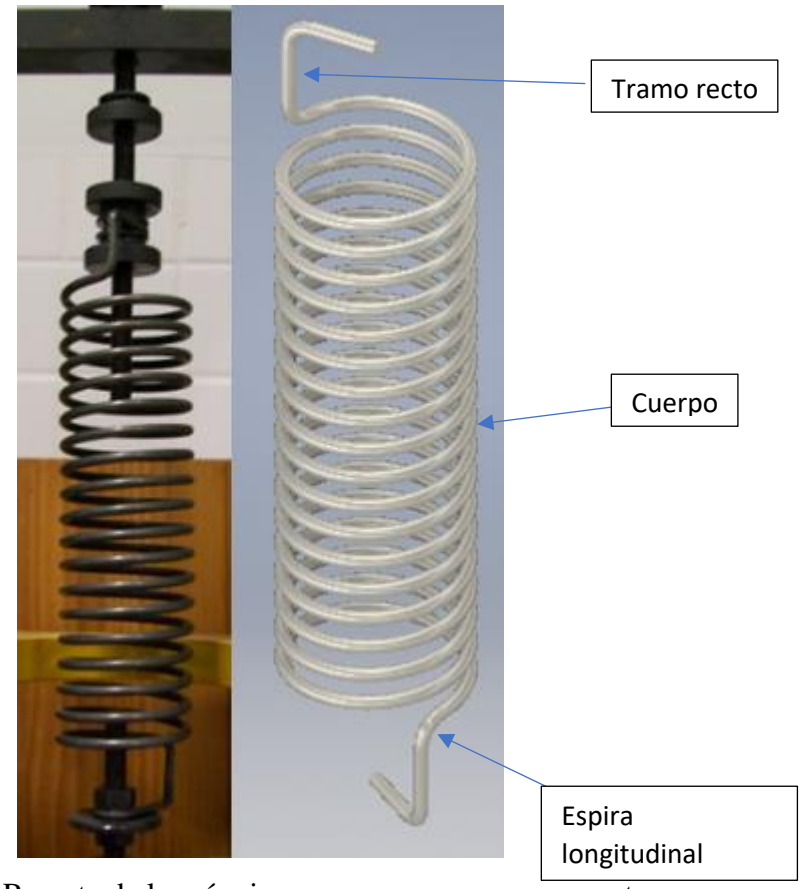


Figura 5.2. Resorte de la máquina y un esquema con sus partes.

Aparte de las propiedades antes mencionadas, el resorte del laboratorio tiene un número total (N_T) de 18 espiras, un diámetro exterior de 54,45 mm, un diámetro de alambre de 4,2 mm y una longitud inicial (L_0) de 141,5 mm.

Las dimensiones de los extremos son las siguientes: el radio de la espira longitudinal, es decir, la curvatura que hay en los extremos del cuerpo del muelle, tiene un valor de 7,4 mm y el tramo recto que hay después de esa curvatura tiene una longitud de 19,2 mm.

Para calcular de forma teórica la rigidez del resorte, es importante tener en cuenta que dicho resorte tiene cinco partes, el cuerpo del resorte, dos espiras longitudinales y los dos tramos rectos, como se ve en la Figura 5.2. En consecuencia, es como si se tiene cinco resortes en serie.

5.1.2.1. Constante de rigidez teórica.

Definiendo la rigidez del cuerpo como k_1, las de las espiras longitudinales como k_2 y la de los tramos rectos como k_3, la rigidez total se calcula aplicando el siguiente procedimiento:

$$F_{eq} = k_1\delta_1 = 2k_2\delta_2 = 2k_3\delta_3 \quad (5.1)$$

$$\delta_{eq} = \delta_1 + 2\delta_2 + 2\delta_3 \quad (5.2)$$

$$\frac{F_{eq}}{k_{eq}} = \frac{F_{eq}}{k_1} + 2\frac{F_{eq}}{k_2} + 2\frac{F_{eq}}{k_3} \quad (5.3)$$

$$\frac{1}{k_{eq}} = \frac{1}{k_1} + 2\frac{1}{k_2} + 2\frac{1}{k_3};$$

$$k_{eq} = k = \frac{k_1k_2k_3}{k_2k_3+2k_1k_3+2k_2k_1} \quad (5.4)$$

Donde k_1 se determina utilizando la expresión 4.13:

$$N_a = N_b + \frac{G}{E} = 18 \text{ espiras} + \frac{69 \text{ GPa}}{193 \text{ GPa}} = 18{,}358 \text{ espiras} \quad (5.5) \text{ [4]}$$

$$k_1 = \frac{d^4G}{\pi D^3 N_a} = \frac{\left(4{,}2*10^{-3}\right)^4 \text{m}^4 * 69*10^9 \frac{\text{N}}{\text{m}^2}}{8(50{,}25*10^{-3})^3 \text{m}^3 18{,}358 \text{ espiras}} = 1152{,}186 \ \frac{\text{N}}{\text{m}} \quad (5.6)$$

Para la espira longitudinal se tiene esfuerzos de torsión. Por tanto su Energía de deformación (U), se determina a partir de la expresión 4.4 donde el torsor (T) se determina con la expresión 4.51 y la longitud a estudiar de la siguiente forma:

$$L = r_2 \frac{\pi}{2} \quad (5.7)$$

Consecuentemente, la expresión 4.4 queda de la siguiente forma:

$$U = \frac{r_2 F^2 D^2 \pi}{16GJ} \quad (5.8)$$

Por tanto, la constante de rigidez de esta espira se obtiene aplicando primero la expresión 4.2 (Castigliano) y luego la fórmula 4.1:

$$\delta = \frac{r_2 F D^2 \pi}{8GJ} = \frac{2 r_2 F D^2}{G d^4} \quad (5.9)$$

$$k_2 = \frac{G d^4}{2 r_2 D^2} \quad (5.10)$$

Sustituyendo valores:

$$k_2 = \frac{69*10^9 \frac{\mathrm{N}}{\mathrm{m}^2} * \left(4{,}2*10^{-3}\right)^4 \mathrm{m}^4}{2*7{,}4*10^{-3}\mathrm{m}(50{,}25*10^{-3})^2 \mathrm{m}^2} = 574{,}530 \ \frac{\mathrm{kN}}{\mathrm{m}} \quad (5.11)$$

Y para el tramo recto hay esfuerzo axil, su Energía de deformación se obtiene mediante la fórmula 4.6. Dicha expresión se sustituye en 4.2, obteniéndose así la elongación del tramo y una vez hecho esto, se aplica en 4.1:

$$\delta = \frac{Fl}{EA} = \frac{4Fl}{\pi d^2 E} \quad (5.12)$$

$$k_3 = \frac{\pi d^2 E}{4l} = \frac{\pi\left(4{,}2*10^{-3}\right)^2 \mathrm{m}^2 193*10^9 \frac{\mathrm{N}}{\mathrm{m}^2}}{4*19{,}2*10^{-3}\mathrm{m}} = 139{,}266\,\frac{\mathrm{MN}}{\mathrm{m}} \quad (5.13)$$

Sustituyendo valores en 5.4 se tiene el valor teórico de la Constante de rigidez:

$$k = 1147{,}564\ \frac{\mathrm{N}}{\mathrm{m}} \quad (5.14)$$

5.1.2.2. Frecuencia de resonancia teórica.

A continuación, se evaluarán los resultados teóricos de la frecuencia de resonancia. Para ello se usará la expresión 4.48. En la mencionada expresión, aparece involucrado el término de la masa del sistema que describe el movimiento oscilatorio. Por tanto esta se obtiene mediante la suma de la masa del *carrito* (1,930 Kg), más la masa del pistón del amortiguador (0,1 Kg), más la masa del conjunto del soporte, motor y volante (0,4555 Kg) y los valores de los contrapesos y pesas que se coloquen. El valor de las pesas que se colocan en el vástago es de 1,039 Kg. Los contrapesos de latón son de 62 g y los contrapesos de aluminio son de 31 g.

Por ejemplo, para el caso del volante con un contrapeso de latón con una pesa en el vástago, la masa del conjunto se determina de la siguiente forma:

$$m = 1{,}930\ \mathrm{Kg} + 0{,}1\ \mathrm{Kg} + 0{,}4555\ \mathrm{Kg} + 1{,}039\ \mathrm{Kg} + 0{,}062\ \mathrm{Kg} = 3{,}587\ \mathrm{Kg} \quad (5.15)$$

Por tanto, se han obtenido los valores teóricos obtenidos se recogen en la Tabla 5.1.:

	Pesas en el vástago	Masa (Kg)	f (Hz)
1 contrapeso de Latón	Sin pesas	2,548	3,378
	1 pesa	3,587	2,847
	2 pesas	4,626	2,507
2 contrapesos de Latón	Sin pesas	2,610	3,338
	1 pesa	3,649	2,823
	2 pesas	4,688	2,490
1 contrapeso de Aluminio	Sin pesas	2,517	3,399
	1 pesa	3,556	2,859
	2 pesas	4,595	2,515
2 contrapesos de Aluminio	Sin pesas	2,548	3,378
	1 pesa	3,587	2,847
	2 pesas	4,626	2,507

Tabla 5.1. Valores teóricos de frecuencia de resonancia y valores de la masa colgada del resorte.

5.1.3. Placa Arduino DUE.

Para la máquina de vibraciones libres y forzadas, se utilizará una placa de Arduino DUE, que presenta la forma que aparece en la Figura 5.3. Se ha elegido esta placa, porque como se comenta en el apartado 4.4.1., es fácil de realizar la programación (algo que en otras placas no ocurre), se puede utilizar en casi cualquier sistema operativo y es la más económica.

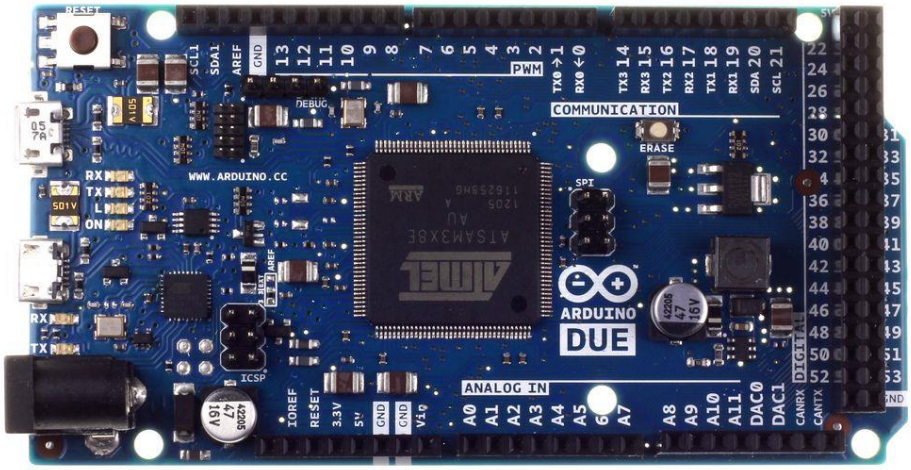


Figura 5.3. Placa Arduino DUE utilizada en la actualización .

La placa de la Figura 5.3. se encargará de recibir la señal emitida por el sensor, luego dicha señal se la manda al ordenador (para que sea representada). La razón por la que se haya escogido el modelo DUE, es porque este modelo tiene una velocidad de reloj (tiempo que tarda en hacer una operación)de 84 MHz, por tanto es la placa más rápida. Este último aspecto lo hace idóneo para recoger datos del sensor.

Otra función que desempeña el Arduino será controlar la velocidad de giro del motor. El ordenador emite la orden al Arduino y este emite una señal eléctrica, cuya tensión irá bajando cuando se ordena que baje la velocidad (a menor tensión, menor velocidad) y la tensión aumentará si se desea aumentar la velocidad (a mayor tensión, mayor velocidad). Esta señal de tensión variable será comunicada al motor eléctrico.

El sensor se conectará al pin *A0* de la parte ANALOG IN. Los *pines* de salida que van conectados a la placa reguladora será el pin *10* de la parte PWM. Para consultar con exactitud la conexión, se recomienda observar el Anexo 9.4.8.

Para ver las características de esta placa Arduino, consultar Anexos 9.2.3.

5.1.4. Sensor de posición.

El sensor utilizado es el modelo MPS-128TSTU0 de SICK. Es un sensor electromagnético de tres hilos. Se alimenta con una tensión de 24 V y emite una señal de salida de 0-10 V. Si se desea consultar la hoja de características consultar Anexos 9.2.2.
Un sensor electromagnético resulta idóneo para medir la posición del *carrito*. Este tiene una vara en voladizo donde en el extremo hay un imán de botón. Cuando el *carrito* se mueve, el imán también y de esta forma se obtiene un campo magnético variable. Estos son los motivos por lo que se ha elegido este tipo de sensor.
Cuando se encuentra en la posición inferior, el sensor proporciona 0 V. Mientras cuando el imán se encuentra en el punto superior del sensor, este emite 10 V. Pero la placa de Arduino DUE no soporta más de 3,3 V (consultar Anexos). Por lo que es necesario diseñar un *divisor de tensión* (una resistencia conectada en serie al sensor y otra conectada en paralelo). La configuración del divisor es como se muestra en la Figura 5.4. Para calcular los valores de R_1 y R_2, se aplican las leyes de *Kirchhoff*, en concreto la *Ley de Kirchhoff* de malla

(entendiendo por malla como el conjunto de conductores y elementos conectados que forman un polígono cerrado). Dicha ley dice lo siguiente:

$$\sum V = 0 \quad (5.16)$$

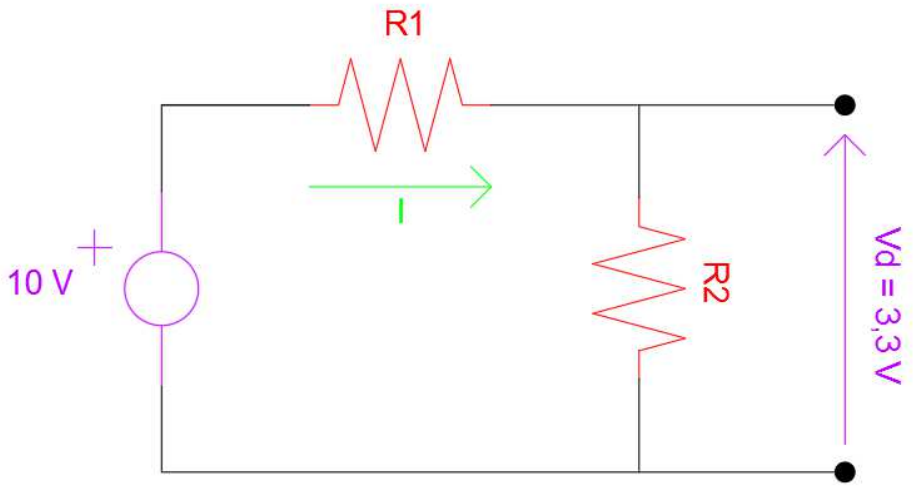


Figura 5.4. Esquema del divisor de tensión.

Primero se estudiará la malla compuesta por la tensión V_d y R_2, que es la que menos incógnitas tiene. Aplicando la *Ley de Ohm*, la tensión en R_2 es la siguiente:

$$V = R_2 I \quad (5.17)$$

El valor de la intensidad es de 20 mA (consultar en la hoja de características del sensor en Anexo 9.2.2.). Por tanto, la *Ley de Kirchhoff* en esta malla queda de la siguiente forma:

$$0{,}02\ \text{A}\ R_2 - 3{,}3\ \text{V} = 0 \quad (5.18)$$

La expresión 5.20 da un valor de R_2 de 165 Ω. Como dicho valor no está normalizado, se proporciona el valor normalizado más cercano, en este caso 150 Ω.

Aplicando la expresión 5.18 en la malla compuesta por la tensión de 10 V, R_1 y R_2:

$$-10\ \text{V} + 0{,}02\ \text{A} * R_1 + 0{,}02\ \text{A} * 150\ \Omega = 0 \quad (5.19)$$

Despejando R_1 de la ecuación 5.21 se obtiene un valor de 350 Ω, el valor normalizado más cercano es de 390 Ω.

Como se ha mencionado antes, el sensor es de tres hilos. El código de conexión es el siguiente: el cable marrón va conectado al positivo de la fuente de alimentación, el cable azul va conectado al negativo de la fuente y el cable negro es el positivo de la señal de salida del sensor.

Para ver la conexión del sensor con el Arduino consultar Anexo 9.4.6.

5.1.5. Placa reguladora de velocidad y motor.

Si se conecta directamente un motor eléctrico a una placa de Arduino, este no se moverá. Esto es debido a que los pines de salida del Arduino emiten una señal muy baja, por tanto no pueden mover un motor. Por tanto, es necesario un elemento que permita adecuar la señal para poder mover un motor con el Arduino.

La solución a este problema es muy simple, ya que en el mercado existen diversos tipos de placas electrónicas que permiten adecuar la señal de salida del Arduino.

Existe un tipo placa electrónica que permiten regular la velocidad del motor con un potenciómetro como la que se ilustra en la figura 4.16a. Dicha placa tiene dos bornes para la alimentación de la placa, dos bornes de salida y una clavija donde se conecta el potenciómetro que es el que regula la velocidad de giro.

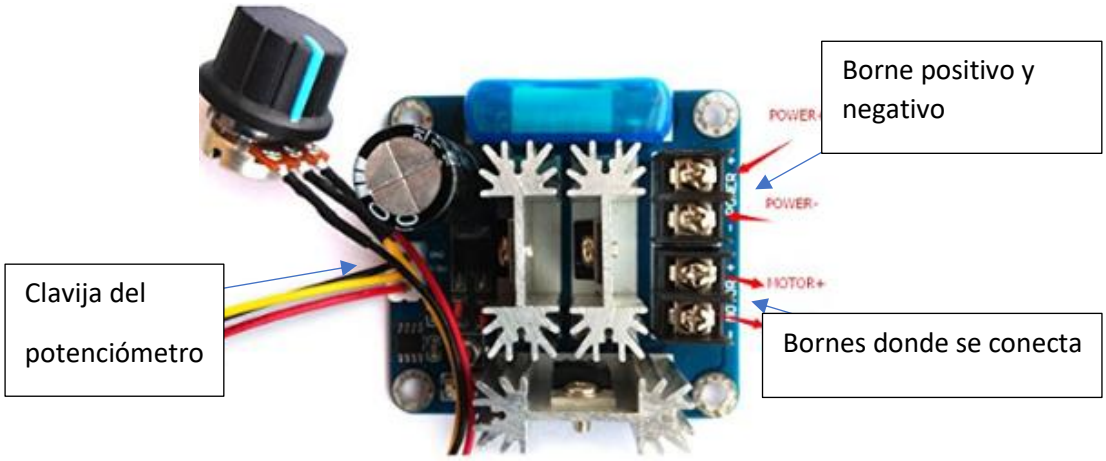


Figura 5.5. Regulador de velocidad utilizado.

Este tipo de regulador recibe la tensión de alimentación de una fuente externa. A través del potenciómetro, se comunica al regulador la variación de tensión que se desea. La placa aumenta el valor de la tensión comandada por el potenciómetro, teniendo un valor de tensión aceptable para mover el motor. Dicha tensión sale por los bornes de salida de la placa para el motor.

Pero esa solución no es la óptima, ya que se pretende controlar esta placa por el ordenador. Por tanto, se ha conectado la placa al Arduino. La conexión es como sigue: el cable por donde se debe transmitir la señal de salida del Arduino, se corresponde con el cable amarillo de la Figura 5.5. El negativo, con el negro de la figura antes mencionada. Realizando la conexión de esta forma, ambas placas quedan comunicadas. Para consultar las características de esta placa, véase el Anexo 9.2.4.

Al regulador, se encuentra conectado un motor de corriente continua de la marca RS Components, *modelo* DC RS Pro 970 D61. Se ha elegido un motor de corriente continua, porque como se ha indicado en el apartado 4.4.3., se va a controlar una carga ligera, además de que se necesita variar la velocidad de giro. Consultando el Anexo 9.2.1. se puede observar que el motor tiene una alimentación de 12 V. Además, el motor lleva incorporado una reductora de engranajes, es decir una transmisión de ruedas dentadas que reduce la velocidad de giro (se puede ver un ejemplo de reductora en la Figura 5.6, siendo la rueda motriz el piñón negro) , que sirve para obtener una velocidad de giro constante. Además también sirve para reducir la velocidad de giro del motor, disminuyendo así la velocidad máxima de giro.

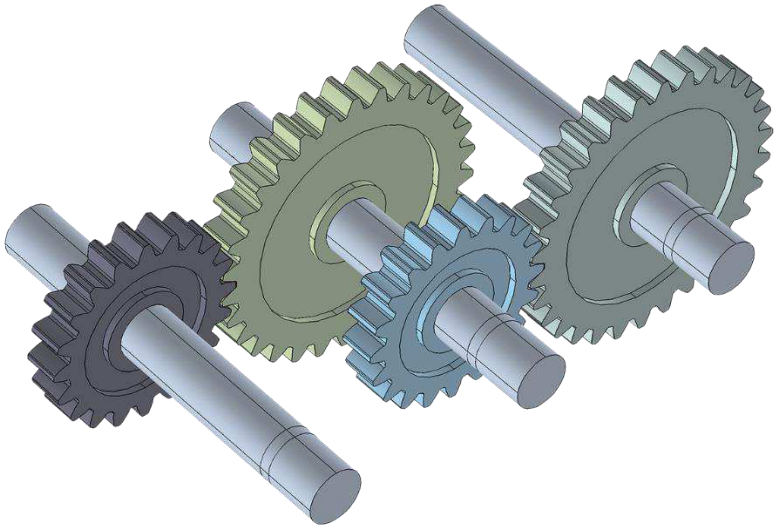


Figura 5.6. [31] Reductora de engranajes.

La conexión entre el Arduino y el motor aparece en el Anexo 9.4.7.

5.1.6. Juego de contrapesos y pesas.

Este juego consta de los siguientes elementos:

- Contrapesos que se colocan en el volante: se atornillan en el volante de forma excéntrica y sin equilibrar, para forzar las oscilaciones en el muelle cuando gira el eje del motor. Hay dos pesas de aluminio de 31 g y dos pesas de latón de 62 g.
- Pesas para el vástago: se colocan en el vástago del *carrito* para poder analizar la rigidez y las oscilaciones bajo varios estados de carga. Cada pesa tiene una masa de 1,039 Kg.

5.1.7. Interfaz gráfica.

La interfaz gráfica es la ventana que aparece en el ordenador para controlar la máquina. Dicha interfaz se realiza con la función *GUIDE* de *Matlab*, debido a que es bastante fácil de programar y que es una de las pocas formas que hay de programar una "*ventana*" que mande

sobre el Arduino. Una vez realizada la programación de dicha interfaz, se tiene la ventana que aparece en la Figura 5.7.

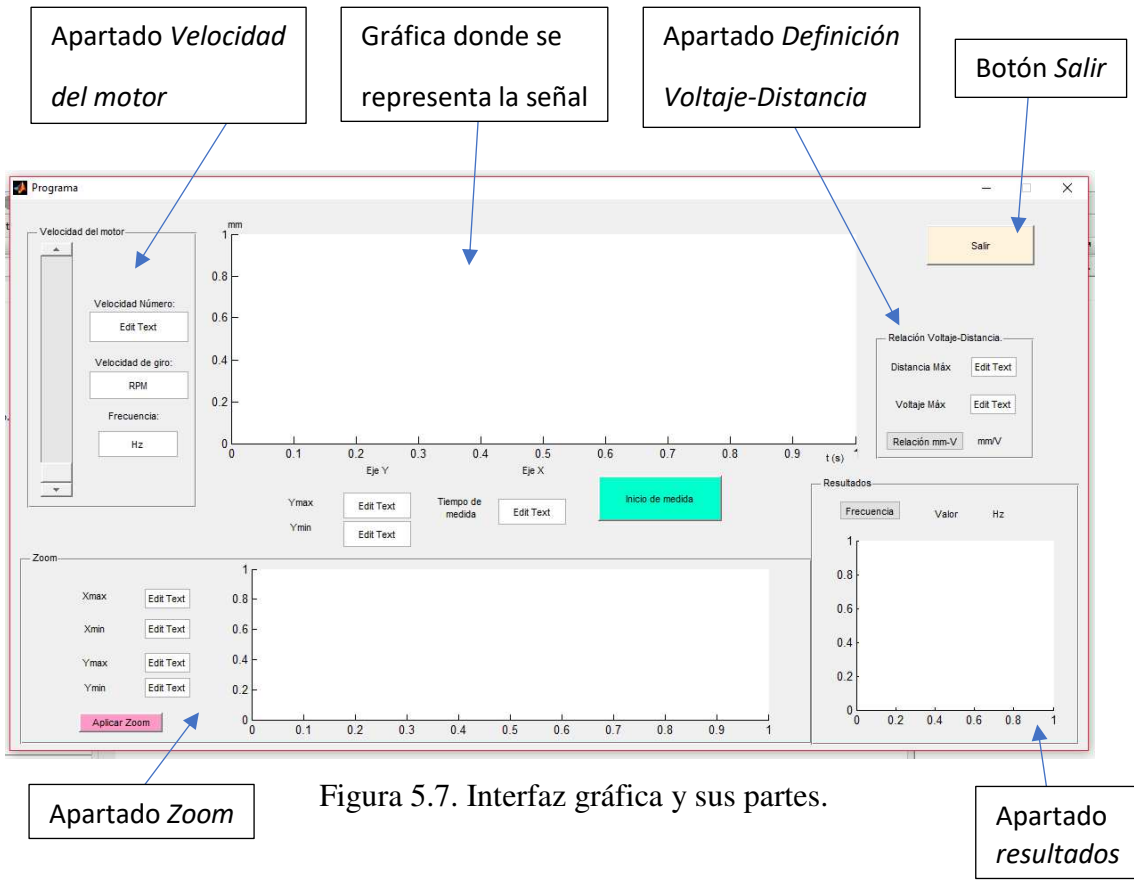


Figura 5.7. Interfaz gráfica y sus partes.

La ventana que aparece en la Figura 5.7 presenta además las siguientes funciones:

En el apartado *relación Voltaje-Distancia*, para que el programa calcule dicha relación y pueda convertir la señal del sensor de voltios a milímetros. Esto es necesario, ya que no tiene sentido expresar la posición de *carrito* en voltios.

En los ejes del centro, se coloca el tiempo de medición en segundos en *Tiempo de medida*, y los valores máximos y mínimos de la distancia que se quiere estudiar en *Ymax* y en *Ymin*. El programa empieza a representar la señal recibida del sensor cuando se seleccione *Inicio de medida*, y seguirá así el tiempo que se haya indicado.

En el recuadro de *Zoom*, se indica los valores máximos y mínimos de los ejes *X* e *Y* deseados, para poder obtener valores más exactos de la señal.

Con el recuadro *Resultados*, se obtiene la señal en dominio de la frecuencia. Por tanto, se representa la señal en dominio de la frecuencia y el valor de la frecuencia de resonancia.

Y con el recuadro *Velocidad del motor*, se indica el número de la velocidad del motor (posición del *slider* o el deslizador con que se indica la velocidad), además de indicar la

velocidad de giro del mismo (en revoluciones por minuto) así como la frecuencia de giro. Estos dos últimos valores son aproximados. Ya que se ha utilizado una expresión de tercer grado para aproximar valores intermedios. Dicha expresión se obtuvo midiendo la velocidad de giro en determinadas posiciones del *slider*, después se graficaron dichos puntos y luego se interpoló la expresión.

Otra función es la de salir, que cuando se desea cerrar el programa es recomendable usar esta opción, ya que borra todas las variables, así como de borra todo aquello que aparece en la pantalla de comandos.

5.1.8. Máquina de vibraciones libres y forzadas rehabilitada.

La máquina en cuestión, presenta las siguientes partes:

- Estructura: Sirve de apoyo a todos los elementos. En la parte superior se encuentra sujeto el resorte.
- Resorte a estudiar: como se ha indicado en el apartado anterior, es un resorte de extensión de acero inoxidable (E = 193 GPa y G = 69 GPa). Tiene un total de 18 espiras, diámetro exterior de 54,45 mm, un diámetro de alambre de 4,2 mm y una longitud inicial de 141,5 mm.
- *Carrito*: Se encuentra sujeto en el extremo inferior del resorte. Consta de un vástago donde se pueden colocar pesas de 1,039 Kg. Además en el *carrito* se apoya el soporte del motor (para conocer las dimensiones de dicho soporte, consultar Anexo 9.4.1.). También el *carrito* presenta una vara en voladizo, el extremo de la misma hay un imán de botón para ser detectado por el sensor.
- Volante: dicho elemento se coloca en el eje de salida de la caja reductora de velocidad del motor. Es un disco en el que se le han practicado doce taladros. Gracias a estos taladros, se pueden atornillar pesas de forma que el volante se encuentre desequilibrado. Esto es necesario para cuando el volante sea girado por el motor, esto haga que el carrito suba y baje.
- Amortiguador: Se encuentra en la parte inferior de la máquina. Dicho amortiguador consta de un pistón y un cilindro. Dicho cilindro se puede rellenar de líquido amortiguante y por tanto, ofrecer resistencia al movimiento oscilatorio del sistema por

rozamiento viscoso. En los estudios a realizar, este elemento no se tendrá en cuenta, ya que el cilindro no tiene líquido amortiguante.

- Sensor: que como se ha mencionado, medirá el desplazamiento del *carrito*. Los desplazamientos serán detectados por el imán de la vara del *carrito*.
- Circuito eléctrico: consta de la tarjeta Arduino DUE (donde se encuentra almacenado el programa que controla las funciones de los elementos eléctricos del sistema), el ventilador de refrigeración del Arduino, el regulador de velocidad (lo que controla la velocidad de giro del motor), el motor eléctrico y las dos fuentes de alimentación (una de 24 V para alimentar el sensor y otra de 12 V, que alimenta al resto del sistema).

  En la Figura 5.8 se presenta la distribución de la máquina, mientras en la Figura 5.9, aparece representada la distribución del sistema eléctrico sin las cubiertas. El sistema eléctrico con cubiertas aparece en la Figura 5.10.

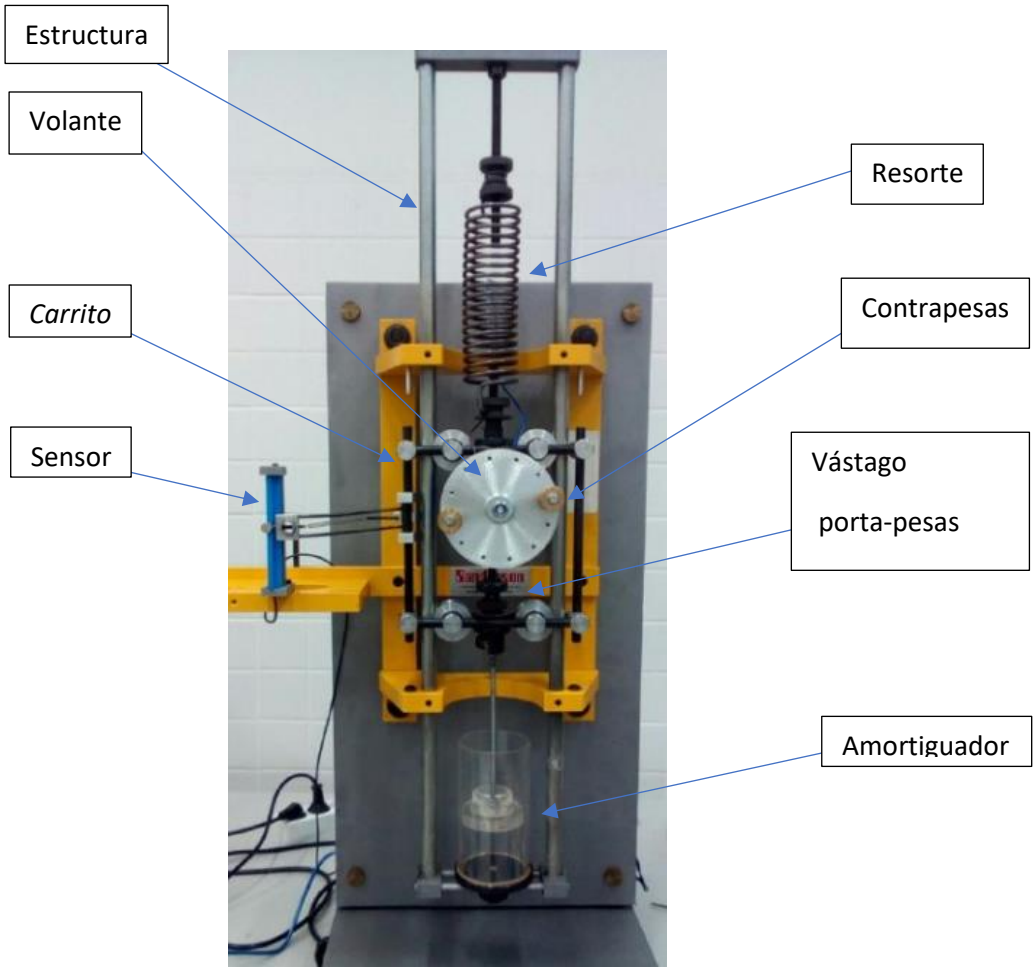


Figura 5.8. Máquina actualizada.

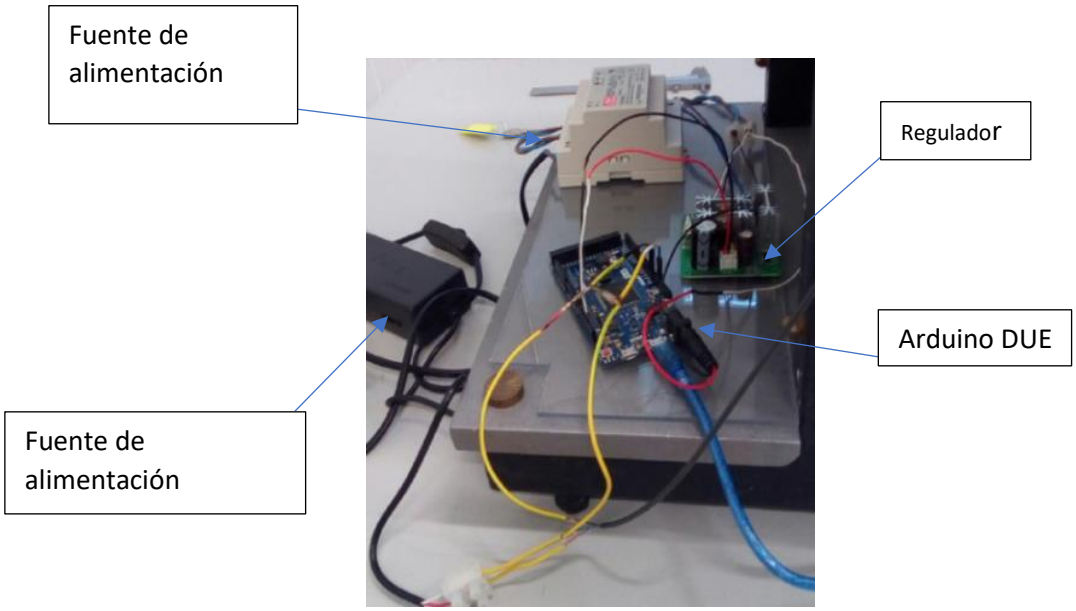


Figura 5.9. Sistema eléctrico sin cubiertas.

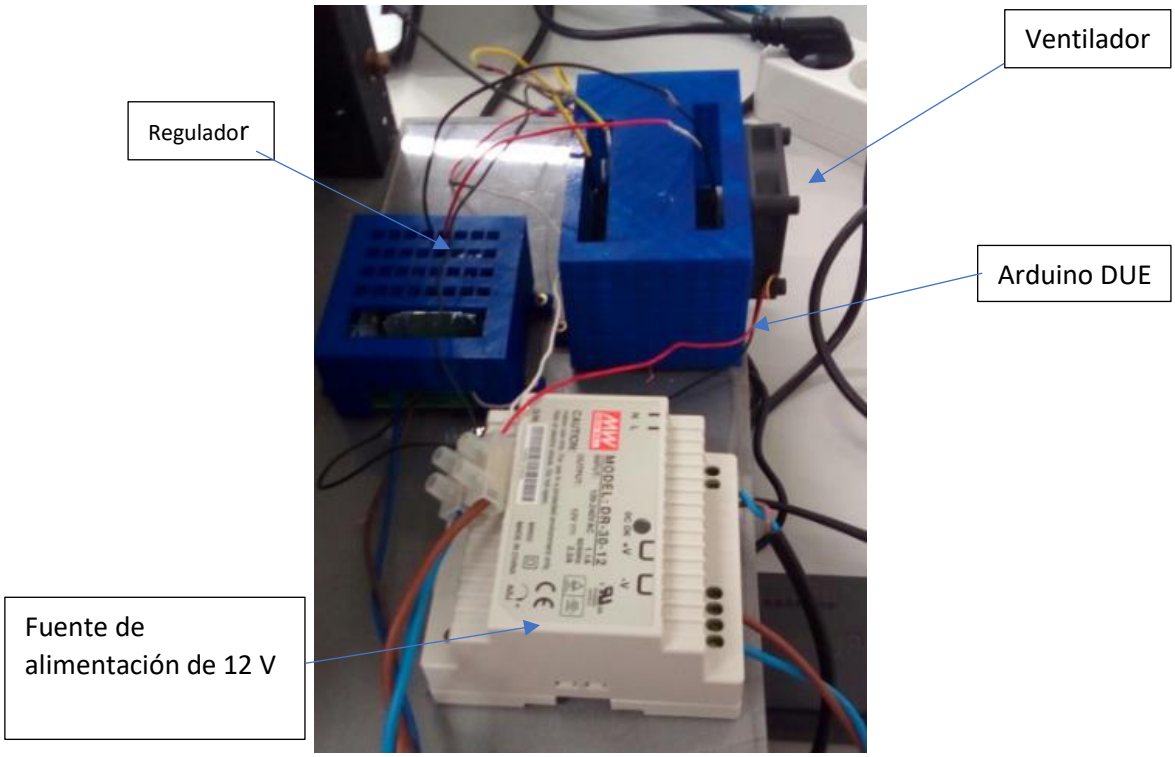


Figura 5.10. Aspecto final del sistema eléctrico.

Los esquemas eléctricos aparecen en los Anexos 9.4.5-9.4.8, así como en el apartado 5.2.1. La máquina se comunica con el ordenador mediante una placa *Arduino DUE* (la placa está conectada mediante el puerto *USB*). La señal leída por el sensor llega al Arduino y de esta

placa va al ordenador. Para regular la velocidad de giro del motor, llega la orden (en forma de una señal que tiene una tensión eléctrica variable, es decir, cuando se indica que gire más rápido, la tensión aumenta) del ordenador al Arduino y luego manda la señal a la placa reguladora de velocidad.

5.2. PROCESO DE ACTUALIZACIÓN.

Dicho proceso consta de la instalación del software en el ordenador, realización del programa y montaje.

5.2.1. Instalación del Software.

Antes de programar en Matlab, como se ha mencionado en el apartado 4.7.5. es necesario ejecutar en el entorno de trabajo de Arduino un ejecutable llamado *Adioes*, para que la placa pueda entender la órdenes de Matlab.

Para instalar el programa Arduino, se dirige a la siguiente dirección web: *https://www.arduino.cc/en/Main/Software*. A continuación, aparece la ventana de la Figura 5.11 donde se debe indicar el sistema operativo del ordenador donde se pretende instalar. En el ordenador donde se pretende instalar es un Windows XP. Por tanto, se selecciona Windows Installer for Windows XP and up.

Figura 5.11. [32] Selección de sistemas operativos.

A continuación, se selecciona just download.

Dentro de la carpeta que se genera, se ejecuta el archivo arduino-1.8.5-windows.exe y se selecciona I agree, luego a siguiente y luego se pone la dirección donde esté guardada la carpeta. Por último se selecciona Install y se espera. Consecuentemente aparece el icono de Arduino en el escritorio.

Luego se conecta la placa al puerto USB y se selecciona el Inicio del ordenador, después se selecciona con el botón derecho del ratón a equipo y por último a administrar. En la parte izquierda de la pantalla del administrador de equipo, se selecciona administrador de dispositivos y aparece dispositivo desconocido. Dicho dispositivo es el Arduino. Se selecciona con el botón derecho al dispositivo desconocido y luego se selecciona actualizar software de controlador... Después de ello se indica Buscar software de controlador en el equipo. A continuación, se selecciona Examinar y se busca la dirección donde se encuentra guardada la carpeta descargada de la página web del Arduino. Dentro de la carpeta, se escoge Drivers. Por último se selecciona aceptar y luego instalar.

Una vez que haya terminado la instalación, el Arduino sí es reconocido por el ordenador (dentro de administrador de dispositivos aparece en el apartado de puertos COM).

Una vez instalado el programa o IDE de Arduino, se procede a ejecutar el programa Adioes. Obviamente, antes hay que obtenerlo, ya que dicho programa no viene en la carpeta descargada antes.

Para obtener el paquete se va a la siguiente dirección http://www.mathworks.com/matlabcentral/fileexchange/32374-matlab-support-package-for-arduino-aka-arduinoio-package?download=true. Dicho enlace conduce a un documento *ZIP* donde se encuentran los elementos necesarios para comunicar el IDE de Arduino con Matlab. Es necesario descomprimir el documento como una carpeta.

Una vez descargado dicho documento, con la placa Arduino conectada, se abre el IDE de Arduino, y aparece la ventana de la Figura 5.12. En dicha imagen se puede observar que en la parte superior se encuentran los menús de Archivos, Editar, Programa, Herramientas y Ayuda. Pues en el menú Herramientas se selecciona Placa y se busca el modelo en el que se está trabajando, en este caso es el Arduino DUE. Y dentro de Herramientas se selecciona Puerto y se indica el puerto COM en el que está conectado (se consulta en administrador de dispositivos).

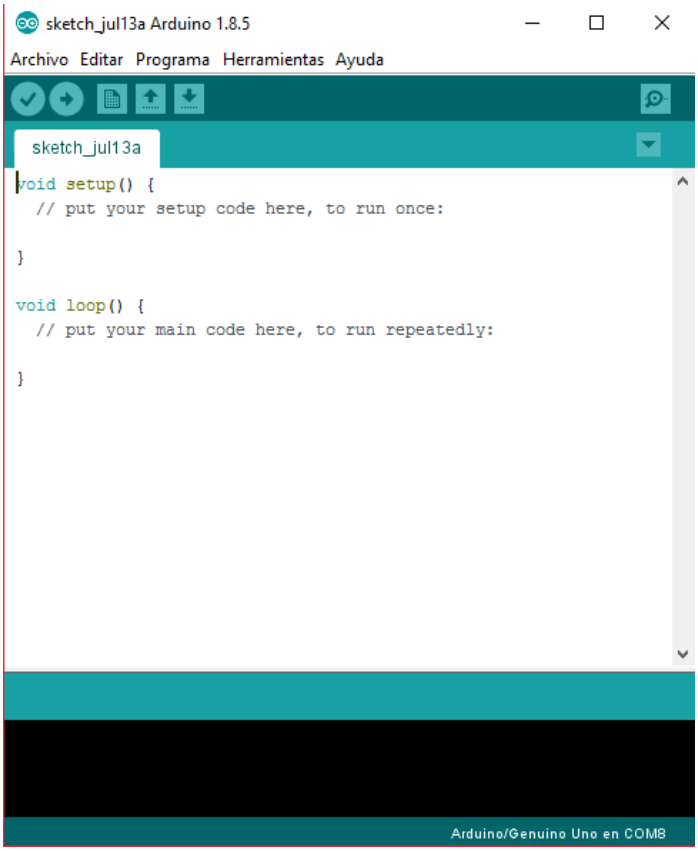


Figura 5.12. IDE Arduino.

Una vez asegurado de que el IDE de Arduino está trabajando con la placa y el puerto correspondiente, se selecciona el menú de Archivo y se busca el programa de la carpeta que se ha descargado. Dentro de dicha carpeta la dirección es ArduinoIO > pde > adioes. Dentro de *adioes* hay un documento denominado *adioes.pde*. Ese es el programa que se debe ejecutar, pero si no se encuentra también vale el documento *adiosrv*. Para ejecutarlo se selecciona subir que está representado por el símbolo de la Figura 5.13, se encuentra debajo del menú de Archivo. A la izquierda se encuentra la opción de verificar, que sirve para comprobar si los programas están bien realizados.

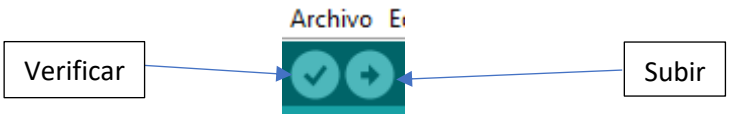


Figura 5.13. Símbolo subir.

Es muy importante que una vez ejecutado el programa *adioes*, no se programe nada más en el IDE de Arduino. Si se ejecuta otro programa, este será almacenado en la placa y por tanto no se queda el programa adioes y la placa no entenderá Matlab.

Una vez que se ha almacenado el programa, se cierra el IDE de Arduino y se abre Matlab como administrador. Una vez abierto Matlab, se ubica el directorio dentro de la carpeta ArduinoIO, que es donde se encuentran las funciones de instalación. En la pantalla de comandos de Matlab se escribe install_arduino.

En principio, ya se podría programar en Matlab para controlar el Arduino, pero para comprobar si se ha realizado bien la instalación, en la pantalla de comandos se escribe a = arduino(COMnúmero de puerto). Si está bien, debe aparecer lo que se muestra en la Figura 5.14, que básicamente es una lista de todos los pin que tiene la placa entre otros elementos (no se ha puesto entero, ya que la lista es muy larga). Si algo se ha realizado mal aparecerá un mensaje de error.

```
>> a=arduino('COM8')
Attempting connection .............
Analog & Digital I/O + Encoders + Servos (adioes.pde) sketch detected !
Arduino successfully connected !

a =

arduino object connected to COM8 port
Analog & Digital I/O + Encoders + Servos (adioes.pde) sketch running on the board

Digital Pin 02 is currently UNASSIGNED
Digital Pin 03 is currently UNASSIGNED
Digital Pin 04 is currently UNASSIGNED
Digital Pin 05 is currently UNASSIGNED
Digital Pin 06 is currently UNASSIGNED
Digital Pin 07 is currently UNASSIGNED
Digital Pin 08 is currently UNASSIGNED
Digital Pin 09 is currently UNASSIGNED
Digital Pin 10 is currently UNASSIGNED
Digital Pin 11 is currently UNASSIGNED
Digital Pin 12 is currently UNASSIGNED
Digital Pin 13 is currently UNASSIGNED
Digital Pin 14 is currently UNASSIGNED
Digital Pin 15 is currently UNASSIGNED
Digital Pin 16 is currently UNASSIGNED
Digital Pin 17 is currently UNASSIGNED
Digital Pin 18 is currently UNASSIGNED
Digital Pin 19 is currently UNASSIGNED
Digital Pin 20 is currently UNASSIGNED
Digital Pin 21 is currently UNASSIGNED
Digital Pin 22 is currently UNASSIGNED
Digital Pin 23 is currently UNASSIGNED
```

Figura 5.14. Confirmación de instalación de Arduino en Matlab.

5.2.2. Realización del programa.

Antes de programar en Matlab, es necesario saber que una placa Arduino no “entiende” las órdenes y funciones que usa Matlab. Por tanto, para sortear este inconveniente, en la ventana de programación del programa Arduino se debe ejecutar el programa *adioes* (dicho programa

“traduce” las órdenes y funciones de Matlab como funciones de Arduino). Una vez ejecutado este programa, se cierra Arduino y se puede empezar a programar en Matlab.

Para realizar la programación de la “ventana” se usa la función GUIDE en Matlab. Dicha función devuelve la pantalla que muestra la Figura 5.15.

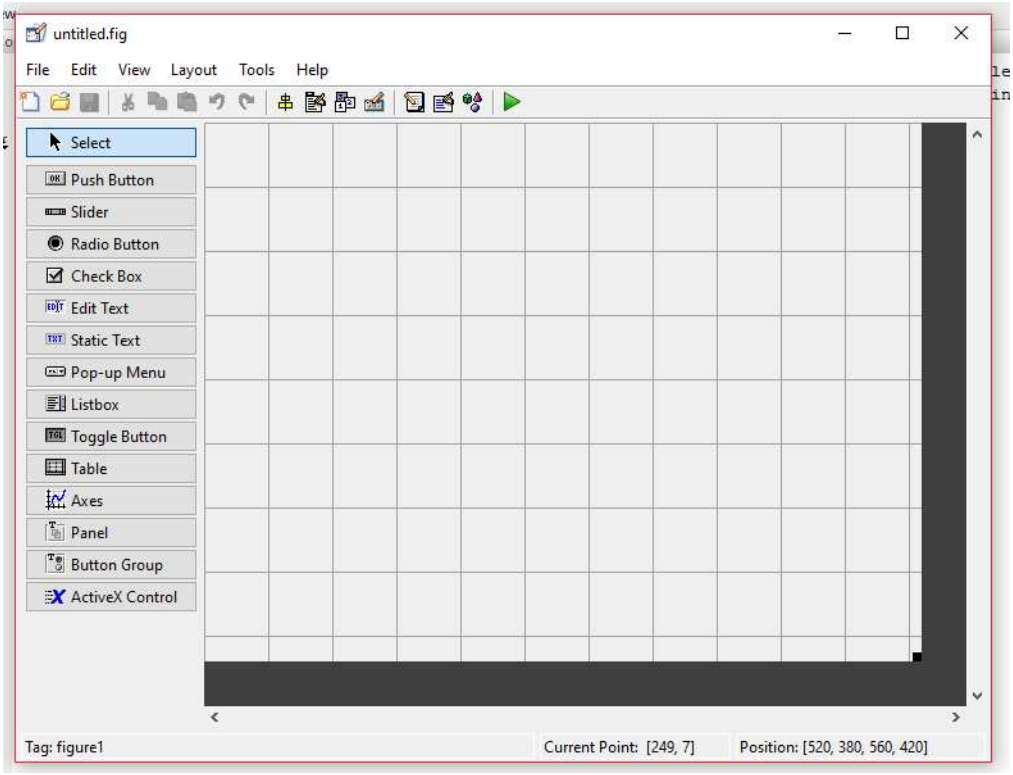


Figura 5.15. GUIDE Matlab.

Los elementos que aparecen a la izquierda de la Figura 5.15 que se suelen utilizar son los siguientes:

Push Button: son los “botones” o “teclas” del programa que se desea realizar.

Slider: este elemento es como un “deslizador”, como la barra que permite subir y bajar por un documento Word o PDF.

Edit text: aparece como un cuadro en blanco que lleva escrito Edit Text o bien otra palabra que se defina en el editor del elemento (haciendo doble click y en la línea de Tag se escribe lo que se desea).

Static text: para indicar algo del programa o bien para mostrar el resultado numérico de alguna operación.

Axes: se usa para representar la señales, gráficas...

Para realizar la programación se utilizará la función GUIDE de Matlab, que permite programar una ventana. Con dicha ventana, como se ha mencionado en el apartado 3, se pretende controlar y obtener los siguientes puntos:

- Recoger la señal obtenida por el sensor y representar esta en función del tiempo. Para lograr ello, primero se establecen unos ejes con el objeto Axes. Luego se colocan tres Edit text, con estos objetos se definen los valores máximos y mínimos del eje Y (valores de desplazamiento) y en principio el tercer Edit text se programa indicar el número de muestras (eje X). A continuación se sitúa un Push buttom que se denominará Inicio de medida y se programará para que una vez que se seleccione dicho botón se empiece a dibujar la gráfica de la señal. Una vez realizada dicha programación, se selecciona guardar y ejecutar. Para dejar el eje X en función del tiempo, se mide el tiempo que tarda el Arduino en recibir, por ejemplo, 1000 muestras. Para obtener y corregir dicho tiempo, se vuelve a realizar una medición de 500 muestras.

  Con la placa Arduino DUE, se obtiene 0,032 s. Una vez obtenido dicho valor, el valor del Edit text para el eje X se divide entre 0,032. Con esto se consigue que el número introducido, que ahora será el tiempo en segundos que se desea que dure la medición, sea convertido en número de muestras. Para representar el eje X con valores de tiempo, dentro de la programación de Inicio de medida, se define un vector de ceros de tamaño el número de muestras. Cuando el sistema recoja un valor del sensor, ese vector de ceros recoge el tiempo en que se ha captado esa muestra.
- Ampliar la señal del sensor obtenida: en este caso, se coloca un nuevo conjunto de ejes, con Axes y se colocan cuatro Edit text. Dos de esos Edit text recogen el valor máximo y mínimo del eje Y, mientras los dos restantes recogen los valores máximos y mínimos del eje X que se quieren examinar. Por último, se añade un Push Buttom que se llamará Zoom. Se programará de forma que cuando se presione, aparezca la parte de la señal aumentada que se desea. Esta parte es necesaria, por si en los ejes principales hay valores que no se aprecian bien.
- Determinar la frecuencia de oscilación: para obtener la frecuencia se programará un Push Buttom, que cuando se selecciona represente la trasformada de Fourier de la señal (esto en otro juego de ejes), además de indicar el valor de X del pico de la señal, que será la frecuencia buscada.

  Para obtener la trasformada de Fourier, se usa la función de Matlab fft, la cual devuelve la trasformada de la señal. Para obtener el valor del pico de la trasformada,

se definen cuatro variables denominadas máximo 1, máximo 2, máximo 3 y máximo 4, los cuales tienen un valor muy bajo. A continuación se inicia un bucle (orden en Matlab que indica que se repita un conjunto de órdenes durante un número determinado de veces) que recorra el vector y compare los valores con esas variables, dos de esas variables son para el eje Y , en concreto el valor más alto y el segundo más alto y los otros dos son para los valores correspondientes en el eje X. Como el valor más alto de Y se tiene en x = 0, esto carece de sentido. Por ello se queda con el segundo valor más alto. Cuando el programa recorra el vector y compare los valores con las cuatro variables máximo, como van a ser más altos que el valor original, el programa borra dicho valor y se va quedando con los valores más altos. Al final se quedan con los valores más altos de los ejes X e Y junto con los segundos más altos.

- Regular la velocidad de giro del motor: donde se colocan un Slider y tres Edit text. El primer Edit text se programa de forma que muestre el valor de la posición del slider. El segundo Edit text tiene una fórmula que aproxima el valor del primer Edit text como la velocidad de giro del motor. Esto se ha realizado midiendo la velocidad de giro del motor para seis posiciones del Slider y luego interpolando una función en esos seis puntos. El tercer Edit text muestra la frecuencia de giro del motor.

Para aclarar todo esto, se muestra el código de programación aparece en el Anexo 9.1.
Una vez realizado la programación, la ventana queda como se muestra en la Figura 5.7.

5.2.3. Montaje.

Una vez listo el programa, se realizaron las conexiones del Arduino con el sensor y del Arduino con la placa reguladora y el motor. En la Figura 5.16 aparece un esquema de las conexiones entre el Arduino y el sensor, mientras en la Figura 5.17 se muestra la conexión del Arduino con el motor. También aparecen dichos esquemas en los Anexos 9.4.6 y 9.4.7.

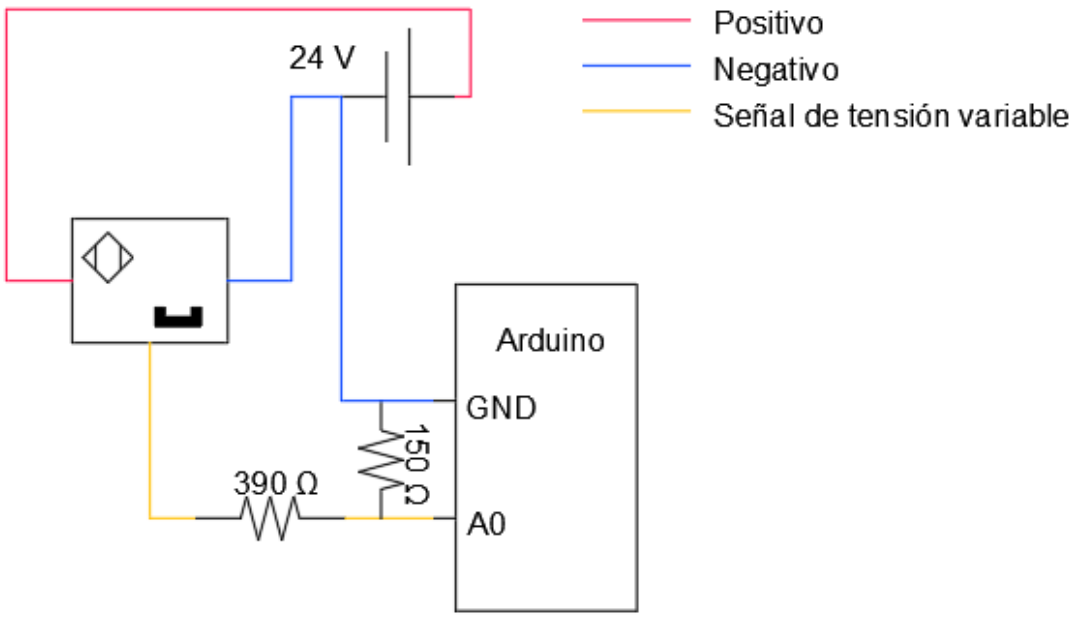


Figura 5.16. Conexión del sensor y el Arduino.

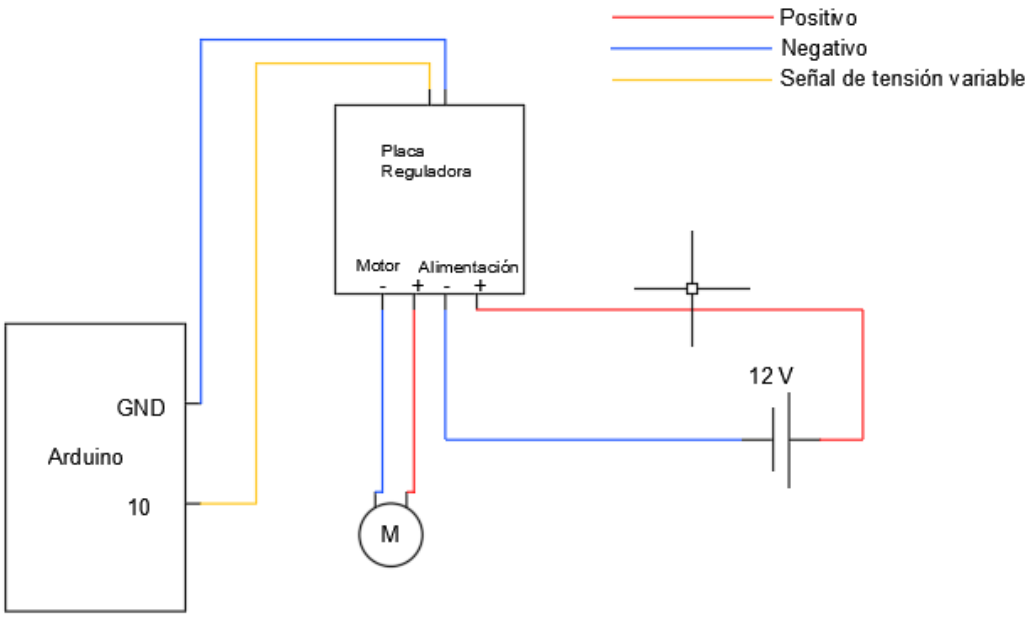


Figura 5.17. Conexión del motor con el Arduino.

Antes de colocar el montaje eléctrico en la máquina, se conecta el Arduino al ordenador y se prueba a ver si el conjunto funciona correctamente.

Una vez asegurado de que el programa funciona correctamente, se desmonta la máquina y las planchas de madera. Esto último se hace para pintar las maderas con la pintura gris metálica.

Una vez secada la pintura, se colocan las planchas en la estructura de la máquina. Antes de colocar la estructura amarilla con los raíles, se limpia el óxido y la suciedad de los raíles, se colocan topes metálicos en la parte superior, para que el brazo del carrito no golpee la parte superior del soporte del sensor. Luego se lubrica los ruedines del carrito con aceite, para disminuir el rozamiento del ruedín con su eje.

A continuación se monta la estructura amarilla, con el carrito, el resorte, el soporte del motor con el motor, el sensor con su soporte y el volante en el eje del motor (para colocar el volante en el eje es necesario fabricar un casquillo, su plano aparece en el Anexo 9.4.5.).

Por último, se coloca el montaje eléctrico en la máquina. Las placas electrónicas y las fuentes se anclan en la parte trasera de la máquina con tornillos. Además, para proteger las placas se han instalado cajas que también van atornilladas. Los planos de las cajas aparecen en los Anexos 9.4.3. y 9.4.4.

5.3. SIMULACIÓN NUMÉRICA.

En el presente apartado se va a realizar la simulación del resorte sometido a cargas de tracción para obtener valores de la Constante de rigidez. Con los resultados obtenidos en esta simulación, se pretende realizar una comparación con los resultados experimentales de dicha constante.

Primero se dibuja el resorte en un programa de dibujo en 3D. Para la realización de este trabajo, se ha utilizado el programa *Autodesck Inventor Professional*.

Para dibujar el resorte se utiliza la función *hélice* (aparece la ventana de la Figura 5.18) para el cuerpo (nos pedirá que definamos el radio medio del muelle el paso, que tiene un valor de 9,2 mm) y para los extremos usamos las funciones extruir, ventana de la Figura 5.19 (se dibuja el perfil a extruir y se define la altura de extrusión) y revolución, Figura 5.20 (se define un eje de revolución, el perfil que se quiere revolucionar y el ángulo de giro).

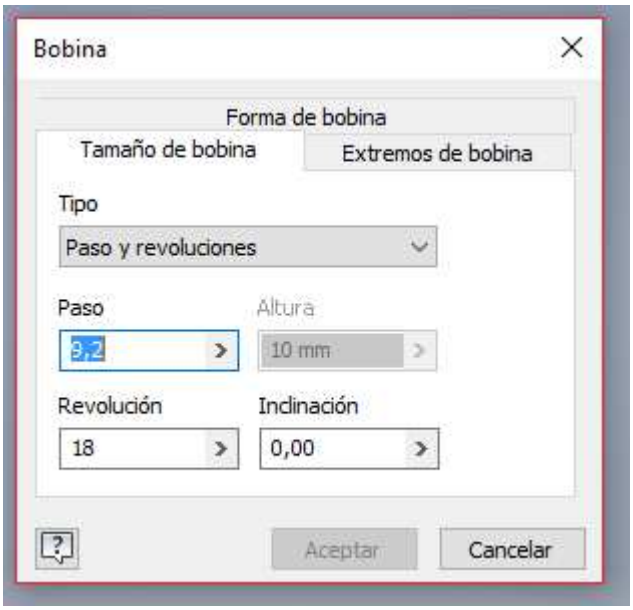


Figura 5.18. Función bobina de *Autodesck Inventor*.

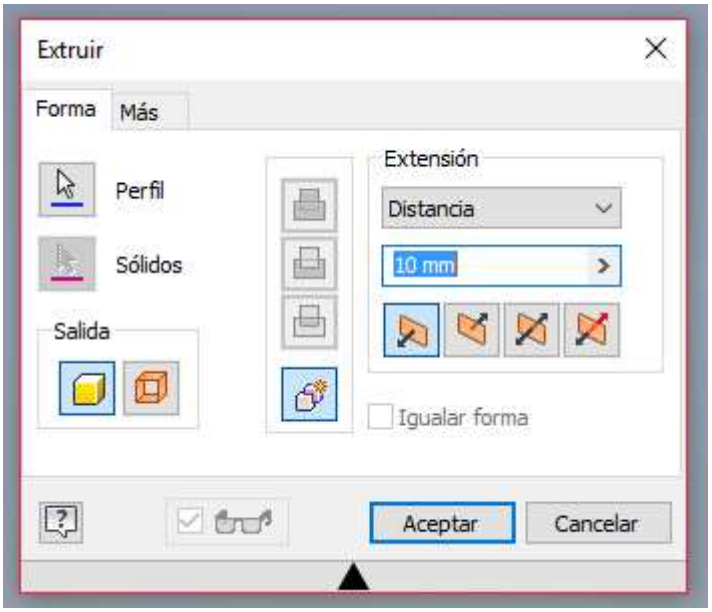


Figura 5.19. *Función Extruir*.

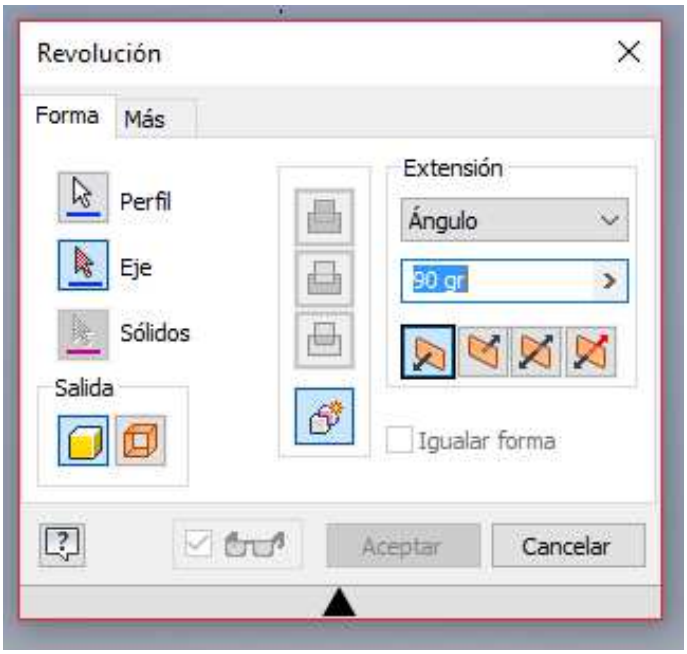


Figura 5.20. Función *Revolución*.

A continuación, se guarda el dibujo como archivo *STEP* (.stp). Luego se abre *Abaqus* y se selecciona que importe el dibujo. Esto se hace seleccionando *File* > *Import* > *Part* y se busca el resorte.

Después se selecciona *Materials* > *Create Material*. Como el muelle está hecho de acero inoxidable se definen las propiedades de ese material. Como ilustra la Figura 5.21, dentro de *Create Material* se selecciona el apartado *General* y luego *Elastic* y se escribe el módulo de Young ($193*10^3$ MPa) y el coeficiente de Poisson (0,399).

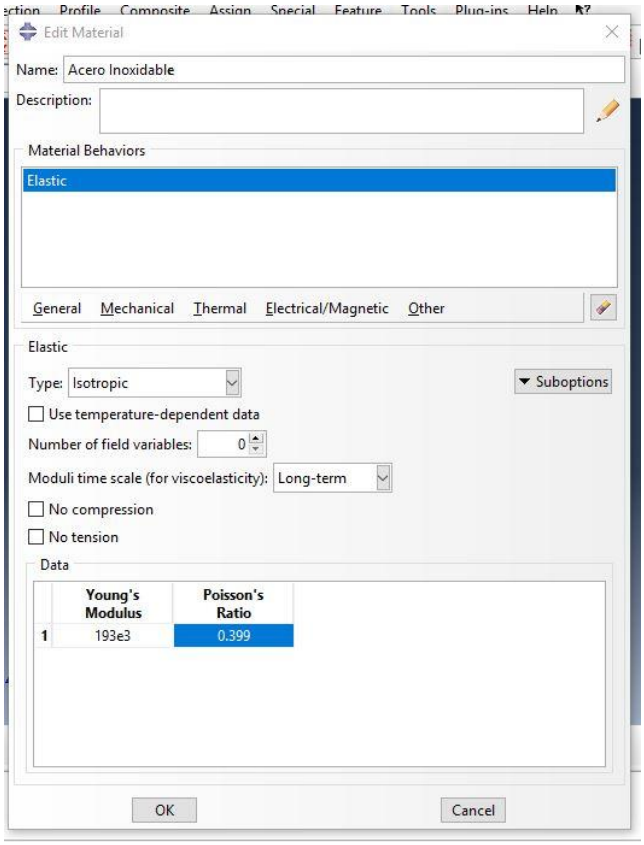


Figura 5.21. Definición de la elasticidad del material.

Dentro de *General*, se selecciona el apartado *Density* y se indica su valor, $8*10^{-9}$ Tn/mm^3, como se puede observar en la Figura 5.22. Es necesario colocar la densidad en estas unidades para obtener los valores de tensión y de deformación en las unidades correspondientes, megapascales y milímetros correspondientemente.

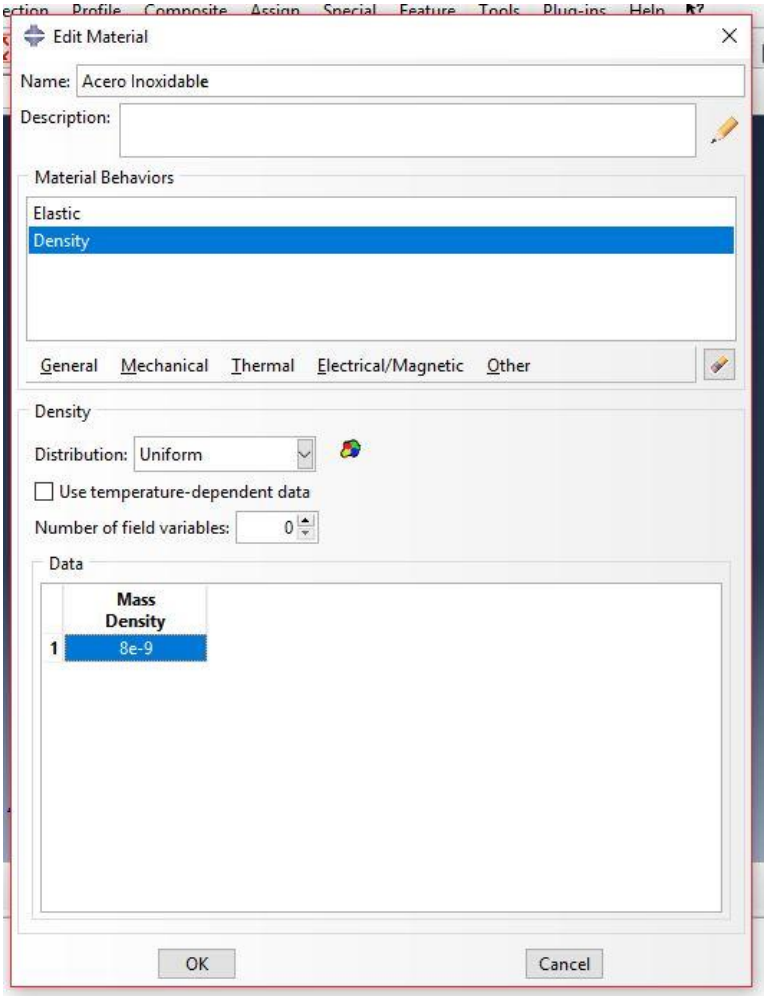


Figura 5.22. Definición de densidad.

Luego se crea una sección (con la función *create section*) con ese material y se los asignamos al muelle. Si está bien asignado el muelle se coloreará de color verde (Figura 5.23).

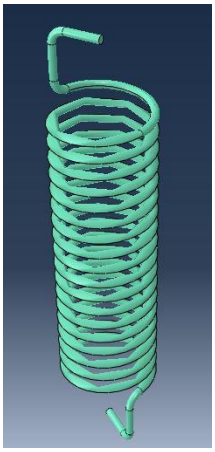


Figura 5.23. Resorte con material definido.

A continuación, se selecciona el apartado *Assembly* y se crea una instancia con la función *Create Instance* (para indicar al programa qué parte se desea estudiar), tal y como aparece en la Figura 5.24. El resorte se debe poner de color azul después de realizar este paso.

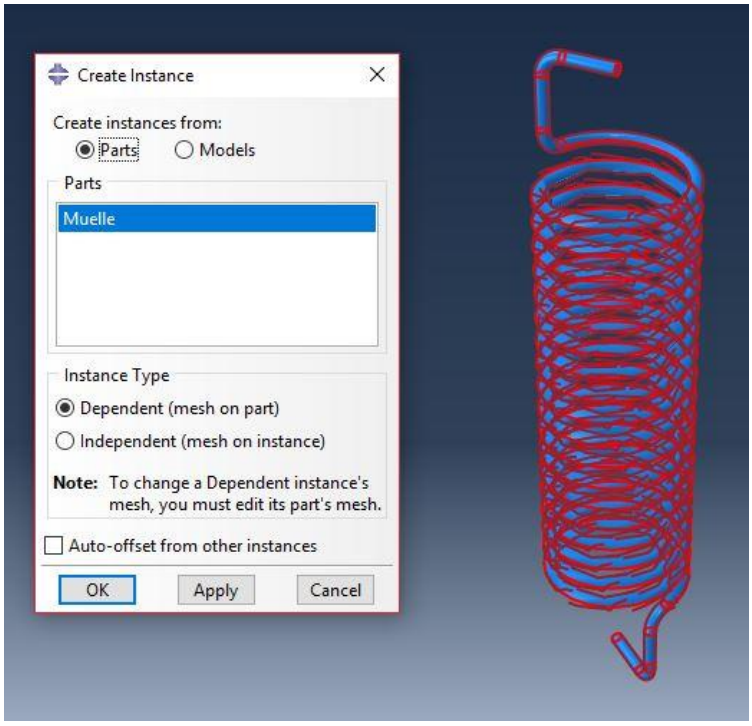


Figura 5.24. Creación de la instancia.

Ahora se definen las condiciones de contorno, en el apartado *BC* y se elige la superficie donde está sujeto el resorte y a continuación se selecciona *ENCASTRE* (empotramiento de la parte superior) y la carga (Figura 5.25), en el apartado *Load* (carga puntual en el extremo inferior). Para tener carga puntual, se debe seleccionar el punto *Mechanical*, *Concentrated Force*. Se indica el punto de aplicación de la carga y el valor de la fuerza aplicada en el resorte, es decir la masa por la gravedad, como aparece en la Figura 5.26.

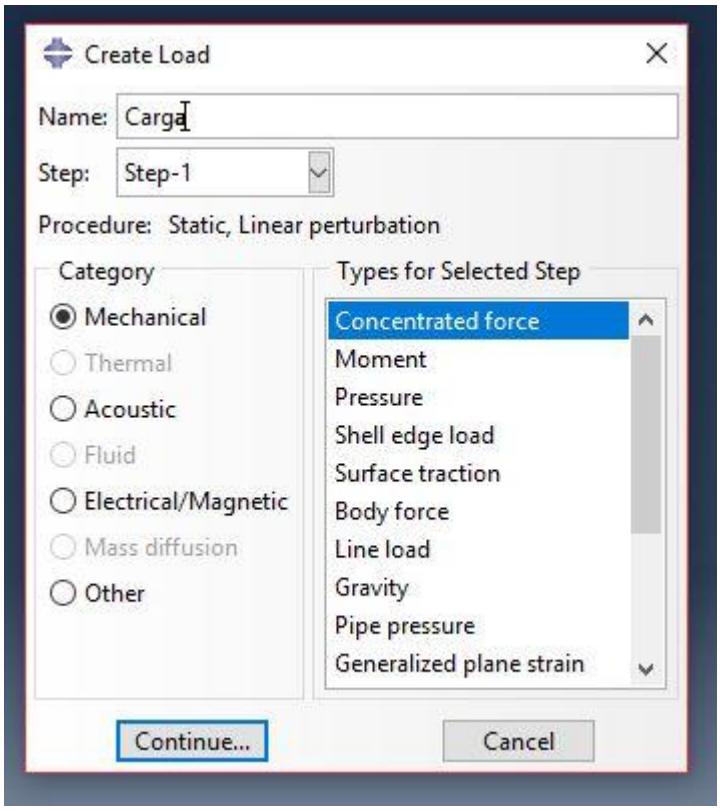


Figura 5.25. Cuadro de carga.

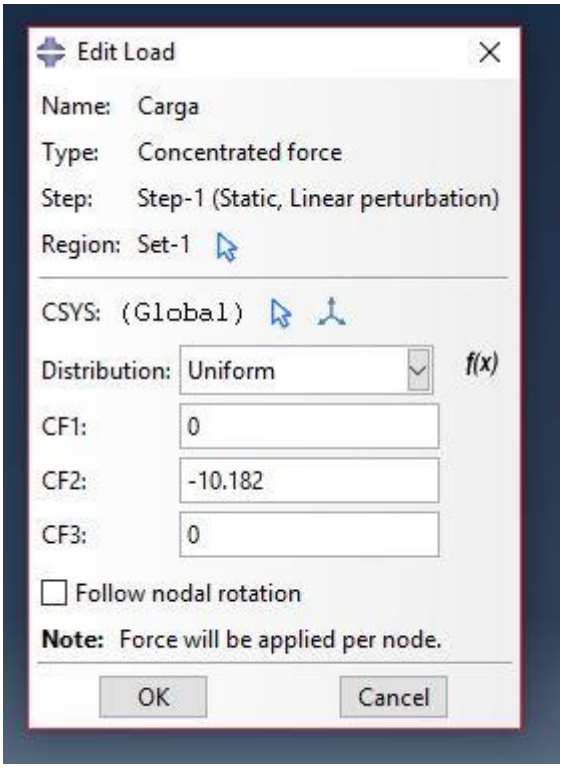


Figura 5.26. Definición de carga.

Para terminar, se mallará el resorte como en la Figura 5.27. se selecciona la pestaña de *Mesh*. Y dentro de ella se escoge *Seed Parts*. En esta ventana nos pedirá que se indique el tamaño de elemento de malla. Nosotros definimos un tamaño de 5. Le damos a Ok y luego se selecciona *Assing Element Type* y luego Ok y a Done. Como el resorte tiene una geometría compleja para el programa, se indica la parte de control de malla y se señala la opción *tet*. Para terminar se indica la función *Mesh Part* y el muelle nos aparecerá mallado.

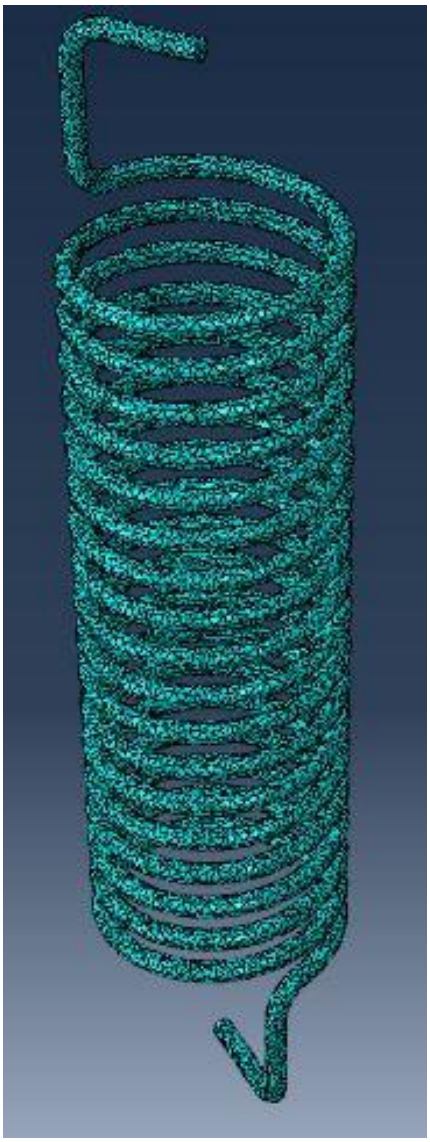


Figura 5.27.

5.4. PROCESO DE MEDIDA.

En este apartado, se explicará los pasos a seguir para medir de forma experimental la constante de rigidez del muelle, la amortiguación y la frecuencia de resonancia.

5.4.1. Constante de rigidez del resorte.

El primer paso es indicar la relación *Voltaje-distancia*. Este paso se realiza en el recuadro que aparece en la Figura 5.28. Es necesario definir dicha relación, ya que la señal de salida que emite el sensor de posición está expresada en voltios y si no definimos ninguna relación de conversión, la posición del *carrito* aparecerá definida en voltios. Obviamente, esto carece de sentido por esa razón en el recuadro *Relación Voltaje-Distancia* ponemos el valor del desplazamiento máximo, que son 100 mm. El valor de voltaje máximo es 3,3 V.

Figura 5.28. Cuadro *Relación Voltaje-Distancia*.

Una vez que se selecciona el botón *Relación mm-V*, al lado se muestra el valor de dicha relación, como aparece en la figura 5.29.

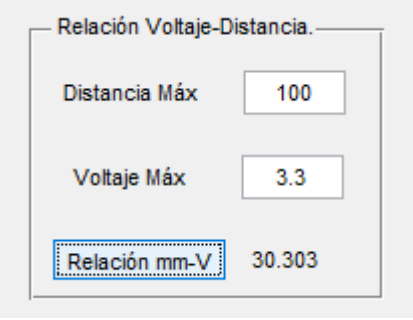


Figura 5.29. *Relación Voltaje-Distancia* definida.

Ahora estudiará la deformación del resorte bajo varios estados de carga. Son tres estados de carga principales, en bloques de dos estados secundarios. Los estados principales son: resorte con carrito, resorte con carrito, motor, volante y contrapesos de latón y resorte con carrito, motor, volante y contrapesos de aluminio. En los tres casos se estudiarán los estados secundarios, es decir, cuando en el vástago hay una pesa y cuando hay dos pesas.
A continuación, como se ilustra en la Figura 5.30, en *Ymáx* se escribe 100, en *Ymin* 0 y en tiempo de medida se coloca un valor de tiempo (en segundos) relativamente corto.

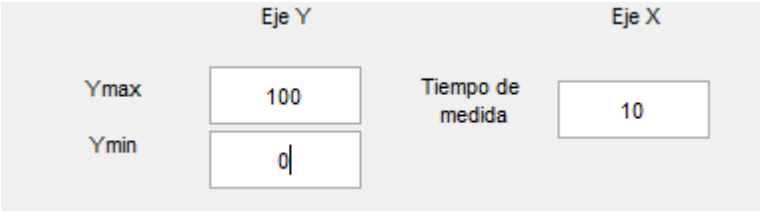


Figura 5.30. Definición de parámetros.

Se prepara la máquina para que tenga cada estado de carga deseado y se selecciona *iniciar medida*. Como no está describiendo ninguna oscilación, en este caso se obtiene una señal constante en mm. Para poder determinar un valor más exacto de la posición adquirida, usa el recuadro *Zoom*, donde se define el valor máximo y mínimo de tiempo (eje *X*) deseado, así como el valor máximo y mínimo de la distancia (eje *Y*) que se quiera.
El punto de corte de la señal con el eje *Y*, es la posición bajo ese estado de carga.
Para determinar *k* se usa la Ley de *Hooke*.

5.4.2. Parámetros de amortiguamiento.

Si no se ha cerrado el programa antes, no es necesario definir la relación mm/V, ya que se encuentra definida de antes. Pero si el programa está recién abierto, habrá que definir la relación.
Luego se establece que *Ymáx* vale 100, e *Ymin* 0 y se define un tiempo de medida (12, 15 o 17 segundos dependiendo de lo que tarde en estabilizarse el sistema).
En este caso, se estudiará 4 situaciones principales: muelle con *carrito*, motor, volante con una contrapesa de aluminio, el segundo con *carrito*, motor, volante con dos contrapesas de aluminio, el tercero *carrito*, motor, volante con una contrapesa de latón y el cuarto caso

carrito, motor, volante con una contrapesas de latón. En cada situación, se estudiará a su vez los casos cuando no hay pesas en el vástago, una pesa y dos pesas en el vástago.
Manualmente, se lleva el carrito a la posición más baja del recorrido, consecuentemente el muelle se estira.
Se selecciona *Inicio de medida* y se suelta el *carrito*. El muelle realizará oscilaciones hasta que se estabilice, como aparece en la Figura 5.31.

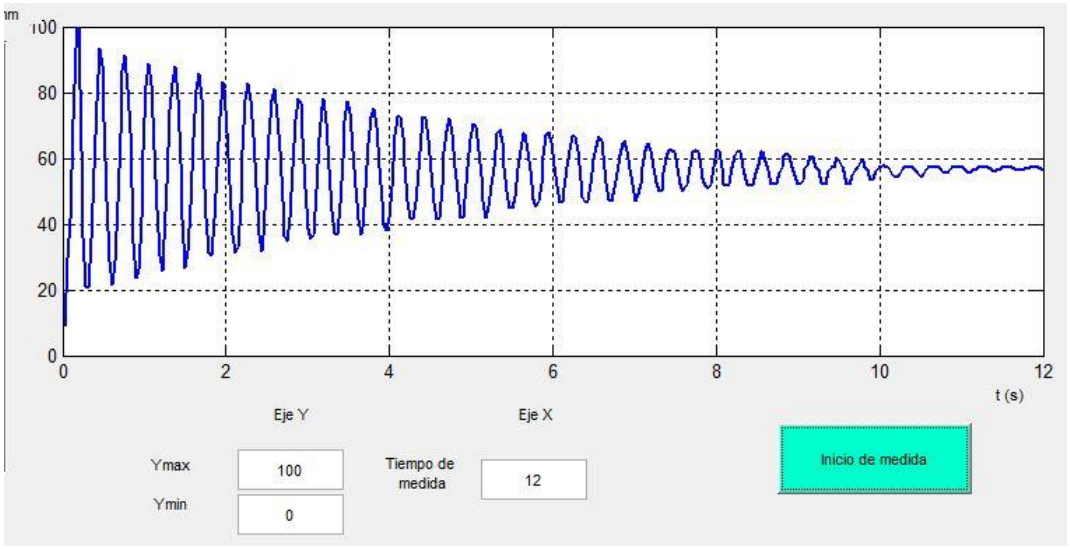


Figura 5.31. Registro de la oscilación

Por último, se mide la *posición media* (que será cuando el carrito se pare cuando esté estabilizado) y luego se mide la amplitud de tres oscilaciones distintas, por ejemplo, la oscilación 1, 10 o 15 y 20. Para obtener mayor exactitud en la mediada usamos el recuadro *Zoom*. A partir de estos datos se podrán determinar los valores del *Decremento Logarítmico*, la *Relación de amortiguamiento* y el *Coeficiente de Amortiguamiento*.
Se deberá realizar los mismos pasos para los distintos estados de carga.

5.4.3. Frecuencia de resonancia.

Como en el caso de la amortiguación, se estudiarán 4 situaciones: una contrapesa de aluminio en el volante, dos contrapesas de aluminio, una contrapesa de latón y dos contrapesas de latón. En cada situación, se estudiará a su vez los casos de sin pesas en el vástago, una pesa y dos pesas.

A continuación, se definen los valores de los ejes, como en el caso anterior se lleva el carrito a la posición más baja, y se selecciona *Inicio de medida* y se suelta el carro. Se vuelven a obtener el mismo tipo de oscilaciones que en el apartado de amortiguación.

Después de que se haya estabilizado, se mide el tiempo que ha tardado en realizar la primera oscilación (en el eje *X* se registran los segundos).

Luego se calcula la frecuencia haciendo la inversa de dicho tiempo. Dicha frecuencia será la frecuencia de resonancia. También podemos determinar la frecuencia de la oscilación gracias al recuadro de *frecuencia*. Se podrá comprobar que se obtiene resultados similares.

Por último, se comprueba si los valores de resonancia se cumplen haciendo girar el motor. Cuando el muelle oscile de forma que la amplitud de la oscilación sea máxima, el sistema *carrito-resorte* estará en resonancia y debe tener la misma frecuencia que la medida anteriormente.

También se deberá realizar los mismos pasos para los distintos estados de carga.

6. RESULTADOS.

Después de seguir los procedimientos explicados en los apartados anteriores, ahora se expondrán los resultados que se pretenden obtener.

Para el caso de la constante de rigidez del resorte también se indicarán los resultados obtenidos por la simulación numérica.

6.1. CONSTANTE DE RIGIDEZ.

6.1.1. Resultados experimentales.

Siguiendo los pasos mencionados en el apartado 5.4.1. se obtienen los resultados de la Tabla 6.1.:

	Masa (Kg)	Elongación (mm)	k (N/m)
1 pesa en el vástago	1,039	10,669	955,346
1 pesa en el vástago con motor, volante y contrapesos de aluminio	1,101	11,890	908,394
1 pesa en el vástago con motor, volante y contrapesos de latón	1,164	11,554	988,472
2 pesas en el vástago	2,078	21,350	954,809
2 pesas en el vástago con motor, volante y contrapesos de aluminio	2,140	22,555	930,765
2 pesas en el vástago con motor, volante y contrapesos de latón	2,203	23,254	929,364

Tabla 6.1. Valores experimentales de la constante de rigidez.

La fuerza considerada para obtener la Constante de rigidez, es la masa añadida multiplicada por la aceleración de la gravedad (9,81 m/s^2) para obtener el peso. La elongación se obtiene restando 100 mm (distancia máxima) la distancia medida por el sensor. Esto es debido a que la parte inferior del sensor, se corresponde con la distancia 0, mientras que el borde superior del sensor se corresponde a los 100 mm y tendría que ser al revés.

Aplicando la media de los datos de la Tabla 6.1. se obtiene el valor experimental medio del resorte.

$$k = 943{,}568\ \frac{\mathrm{N}}{\mathrm{m}}$$

Si se grafican los valores obtenidos y se define una línea de tendencia (ver Figura 6.1), se puede apreciar que la relación *F/δ* se aproxima bastante a un comportamiento lineal. Por tanto, se puede verificar lo que afirma la Ley de *Hooke*, la fuerza aplicada al resorte es directamente proporcional a la elongación que sufre. Dicho parámetro proporcional es la Constante de rigidez. Se reúnen todos los resultados experimentales en la Tabla 6.1.

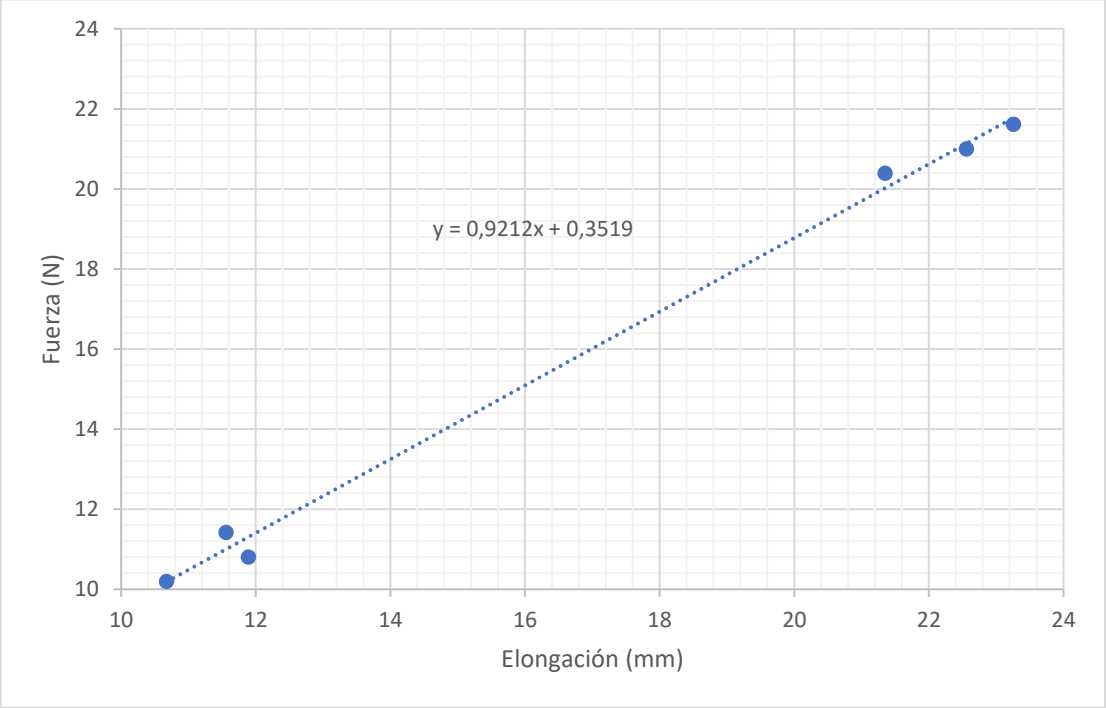


Figura 6.1. Gráfica de la fuerza en función de la elongación.

6.1.2. Resultados obtenidos en la simulación.

Después de seguir los pasos que se indican en el apartado 5.2.3, dentro del apartado *Job* y se selecciona *Create Job*. Por último se escoge la opción *Submit*.

Estos últimos pasos permiten obtener los resultados en el programa. Del programa se han sacado los valores de los desplazamientos, pero se pueden sacar otros datos como la tensión a los que está sometido el muelle, así como las tensiones máximas de Treska y Von Mises.

Los valores de desplazamiento vertical obtenidos son los siguientes:

Para una pesa de 1,039 Kg, el muelle se encuentra sometido a la siguiente fuerza:

$$F = m * g = 1{,}039\ \text{Kg} * 9{,}8\frac{\text{m}}{\text{s}^2} = 10{,}182\ \text{N} \quad (6.1)$$

El programa representa la distribución de desplazamientos que aparece en la Figura 6.2. Indica que el menor desplazamiento se corresponde con el color rojo, con un valor de 0.015 mm, mientras que el valor máximo de desplazamiento es el color azul oscuro con 9 mm. En concordancia con la mencionada imagen, aparece una distribución de desplazamientos coherente con el estado de carga a la que está sometido el resorte, es decir, presenta poco desplazamiento en la parte superior y el mayor desplazamiento ocurre en el extremo inferior, donde está sujeto el *carrito* con masa.

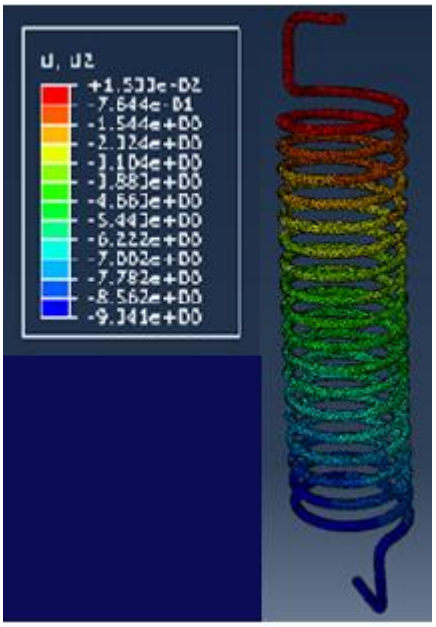


Figura 6.2. Distribución de deformaciones en el resorte.

Según el programa, δ vale 9,141 mm. Por tanto, el valor de la rigidez del muelle se obtiene con la expresión 4.1:

$$k = \frac{10{,}182\ \text{N}}{9{,}141 * 10^{-3}\text{m}} = 1113{,}883\ \frac{\text{N}}{\text{m}}$$

Aplicando este procedimiento con el resto de los estados tensionales, se tienen los datos de la Tabla 6.2.:

	Masa (Kg)	k simulado (N/m)
1 pesa en el vástago	1,039	1113,883
1 pesa en el vástago con motor, volante y contrapesos de aluminio	1,101	1087,921
1 pesa en el vástago con motor, volante y contrapesos de latón	1,164	1085,538
2 pesas en el vástago	2,078	1085,501
2 pesas en el vástago con motor, volante y contrapesos de aluminio	2,140	1096,862
2 pesas en el vástago con motor, volante y contrapesos de latón	2,203	1085,520

Tabla 6.2. Valores de la Constante de rigidez obtenidos en la simulación.

El valor medio obtenido es el siguiente:

$$k = 1092{,}538\ \frac{\mathrm{N}}{\mathrm{m}}$$

6.1.3. Comparativa de resultados.

En la Tabla 6.3 se muestra los resultados obtenidos de forma experimental así como los obtenidos en la simulación.

Masa (Kg)	k (N/m)	k simulado (N/m)	Error (%)
1,039	955,346	1113,883	14,233
1,101	908,394	1087,921	16,502
1,164	988,472	1085,538	8,942
2,078	954,809	1085,501	12,040
2,140	930,765	1096,862	15,143
2,203	929,364	1085,520	14,385

Tabla 6.3. Resultados experimentales-numéricos.

Se pueden apreciar que existen diferencias, ya que el valor experimental obtenido es 943,568 N/m y el obtenido en la simulación es 1092,538 N/m. Hay una diferencia del 13,635 % Esta variación es debido a que en la simulación, ya que el programa tiene una serie de condiciones que no ocurre en la práctica.

También a la hora de realizar el cálculo, el programa usa el método de elementos finitos. Básicamente dicho proceso consiste en dividir elemento a estudiar en partes más pequeñas (elemento de malla), y luego se aplican ecuaciones que relacionan estas partes con los desplazamientos y las fuerzas.

Ese proceso en cuestión es una aproximación, no proporciona el valor exacto.

6.2. PARÁMETROS DE AMORTIGUACIÓN DEL RESORTE.

Para calcular el decremento logarítmico, se utilizará la expresión 4.46 y para calcular la relación de amortiguamiento, se despejará esta de la expresión 4.49. Queda de la siguiente forma:

$$\xi = \frac{\delta}{2\pi+\delta} \quad (6.2)$$

Los datos obtenidos se han resumido en la Tabla 6.4:

		δ	ξ	C
1 contrapeso de Latón	Sin pesas	0,07	0,011	0,0004
	1 pesa	0,06	0,009	0,0003
	2 pesas	0,05	0,008	0,0002
2 contrapesos de Latón	Sin pesas	0,06	0,009	0,0003
	1 pesa	0,06	0,009	0,0003
	2 pesas	0,06	0,009	0,0003
1 contrapeso de Aluminio	Sin pesas	0,07	0,011	0,0004

	1 pesa	0,04	0,006	0,0002
	2 pesas	0,07	0,011	0,0003
2 contrapesos de Aluminio	Sin pesas	0,07	0,011	0,0004
	1 pesa	0,05	0,008	0,0002
	2 pesas	0,06	0,009	0,0003

Tabla 6.4. Valores de los parámetros de amortiguación.

Se puede apreciar que el coeficiente de amortiguación disminuye ligeramente a medida que se carga el sistema. Es el comportamiento más lógico, ya que estos parámetros definen la curva que limita las oscilaciones amortiguadas, que tienen forma de curva exponencial, como se indica en la expresión 4.45. Si hay más masa colgando del resorte, el sistema describirá oscilaciones con la amplitud más reducida que cuando no hay masa.

6.3. FRECUENCIA DE RESONANCIA.

En primer lugar, la frecuencia de resonancia se obtendrá llevando el *carrito* a la parte inferior del recorrido y soltando, se estudiará el periodo y la frecuencia de la primera oscilación. Después se realizará el mismo proceso de medición pero usando el motor.
Se ha tenido los valores de la Tabla 6.5.

		Valores experimentales	Valor teórico	Con motor
	Pesas en el vástago	f (Hz)	f (Hz)	f (Hz)
1 contrapeso de Latón	Sin pesas	3,571	3,378	3,509
	1 pesa	2,857	2,847	3,135
	2 pesas	2,597	2,507	2,597
2 contrapesos de Latón	Sin pesas	3,226	3,338	3,125
	1 pesa	2,857	2,823	2,841
	2 pesas	2,500	2,490	2,591
1 contrapeso de Aluminio	Sin pesas	4,255	3,399	3,54
	1 pesa	2,786	2,859	2,857
	2 pesas	2,564	2,515	2,632
2 contrapesos de Aluminio	Sin pesas	3,846	3,378	3,333
	1 pesa	2,817	2,847	2,915
	2 pesas	2,597	2,507	2,564

Tabla 6.5. Valores de la frecuencia de resonancia.

Las diferencias entre los resultados aparecen en la Tabla 6.6:

	Pesas en el vástago	Error Experimentales-Teóricos (%)	Error Cambio de velocidad-Teóricos (%)	Error Manual-variación (%)
1 contrapeso de Latón	Sin pesas	1,096	2,825	1,779
	1 pesa	6,108	3,023	8,863
	2 pesas	3,082	3,097	0,016
2 contrapesos de Latón	Sin pesas	9,565	12,391	3,226
	1 pesa	5,299	5,834	0,568
	2 pesas	6,086	2,667	3,512
1 contrapeso de Aluminio	Sin pesas	17,130	2,560	20,207
	1 pesa	8,881	6,542	2,502
	2 pesas	4,645	2,120	2,580
2 contrapesos de Aluminio	Sin pesas	6,512	7,699	15,396
	1 pesa	7,430	4,206	3,365
	2 pesas	3,082	4,328	1,303

Tabla 6.6. Errores entre los valores de la frecuencia.

Por último, se realizará la comparación de los datos con los valores proporcionados por el recuadro de *Resultados*, donde se muestra la señal en dominio de la frecuencia.

Estos datos se tomaron cuando las mediciones se realizaron llevando el *carrito* al inferior y soltando.

Después de haber realizado las mediciones, si seleccionamos la opción de *frecuencia* dentro del recuadro *Resultados* se obtiene una gráfica como la que se muestra en la Figura 6.3. En dicha imagen se muestra la representación gráfica de la *Trasformada de Fourier* de la señal de las oscilaciones, que al ser una señal aproximada a una curva senoidal proporciona una trasformada con un pico. El valor del eje *X* donde el valor de la trasformada (valor de *Y*) es la máxima, corresponde con el valor de la frecuencia de resonancia del sistema.

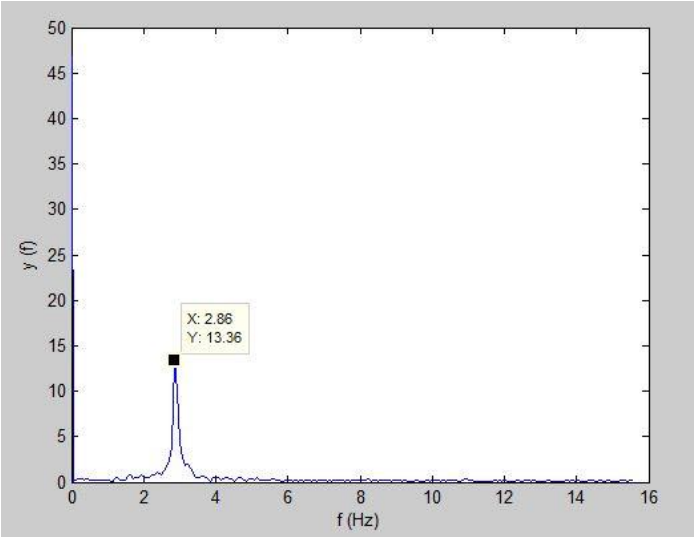


Figura 6.3. Trasformada de Fourier de la señal.

La frecuencia obtenida es de 2,86 Hz.

Los valores restantes se tienen en la Tabla 6.7:

	Pesas en el vástago	f (Hz)
1 contrapeso de Latón	Sin pesas	3,361
	1 pesa	2,86
	2 pesas	2,574
2 contrapesos de Latón	Sin pesas	3,289
	1 pesa	2,86
	2 pesas	2,574
1 contrapeso de Aluminio	Sin pesas	3,361
	1 pesa	2,86
	2 pesas	2,574
2 contrapesos de Aluminio	Sin pesas	3,432
	1 pesa	2,86
	2 pesas	2,574

Tabla 6.7. Frecuencias obtenidos por la Trasformada de Fourier.

Las diferencias expresadas en tantos por ciento entre los valores obtenidos por Fourier y los valores experimentales aparecen en la Tabla 6.8:

	Pesas en el vástago	Error Fourier-Experimental (%)
1 contrapeso de Latón	Sin pesas	5,892
	1 pesa	0,100
	2 pesas	0,901
2 contrapesos de Latón	Sin pesas	1,959
	1 pesa	0,100
	2 pesas	2,960
1 contrapeso de Aluminio	Sin pesas	21,017
	1 pesa	2,674
	2 pesas	0,386
2 contrapesos de Aluminio	Sin pesas	10,768
	1 pesa	1,530
	2 pesas	0,901

Tabla 6.8. Comparación entre los valores de Fourier y los valores experimentales.

Todos los errores se recogen en la Tabla 6.9. En dicha tabla se puede comprobar que el error es muy reducido, salvo en el caso de un contrapeso de aluminio sin pesas en el vástago. Esto puede estar debido a que este estado de carga, es el más ligero y por tanto más difícil de medir.

	Pesas en el vástago	Error Experimentales -Teóricos (%)	Error Cambio de velocidad-Teóricos (%)	Error Fourier-Experimental (%)	Error Manual-variación (%)
1 contrapeso de Latón	Sin pesas	1,096	2,825	5,892	1,779
	1 pesa	6,108	3,023	0,100	8,863
	2 pesas	3,082	3,097	0,901	0,016
2 contrapesos de Latón	Sin pesas	9,565	12,391	1,959	3,226
	1 pesa	5,299	5,834	0,100	0,568
	2 pesas	6,086	2,667	2,960	3,512
	Sin pesas	17,130	2,560	21,017	20,207

1 contrapeso de Aluminio	1 pesa	8,881	6,542	2,674	2,502
	2 pesas	4,645	2,120	0,386	2,580
2 contrapesos de Aluminio	Sin pesas	6,512	7,699	10,768	15,396
	1 pesa	7,430	4,206	1,530	3,365
	2 pesas	3,082	4,328	0,901	1,303

Tabla 6.9. Errores cometidos.

Realizando la media de los valores de la tabla de arriba, para los valores experimentales-teóricos se obtiene un error de 6,576 %. Para los valores de la frecuencia obtenidos variando la velocidad de giro-teóricos da un error de 4,774 %, para Fourier- experimental 4,099 % y para los errores obtenidos de forma manual-variando la velocidad del motor 5,276 %. Se pueden observar que son diferencias muy pequeñas, lo que hace pensar que las mediciones se han realizado relativamente bien. También estos errores hacen que los valores que se han obtenido con la ayuda de la máquina (ya sea soltando el *carrito*, variando la velocidad del motor o con el apartado *resultados*) son bastante fiables y por tanto, la máquina funciona bien.

Los valores que recoge la Tabla 6.5, refleja un aspecto que concuerda con la expresión 4.48. A modo de ejemplo, se muestran la Figura 6.4, la Figura 6.5, Figura 6.6 y la Figura 6.7. Dichos valores demuestran que cuando se le añade al *carrito* más masa, la frecuencia es menor. Otro aspecto que se puede deducir a partir de la expresión teórica, cuando aumenta la rigidez del resorte (aumenta el valor de k), la frecuencia aumenta. Por tanto, la constante de rigidez es un factor muy importante en el movimiento vibratorio.

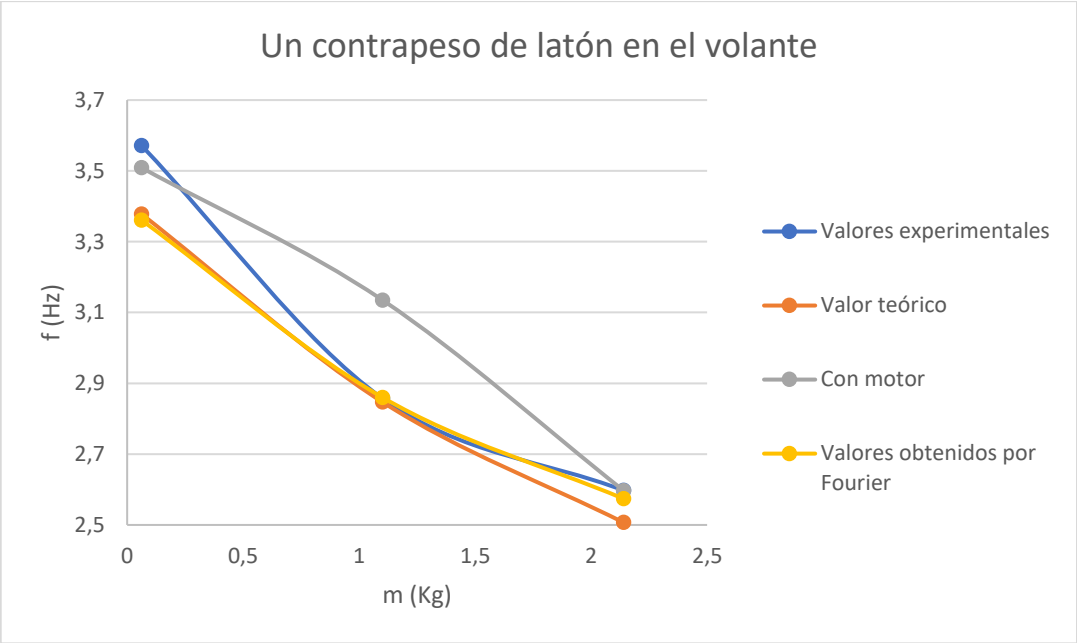


Figura 6.4. Representación de la frecuencia en función de la masa para una contrapesa de latón.

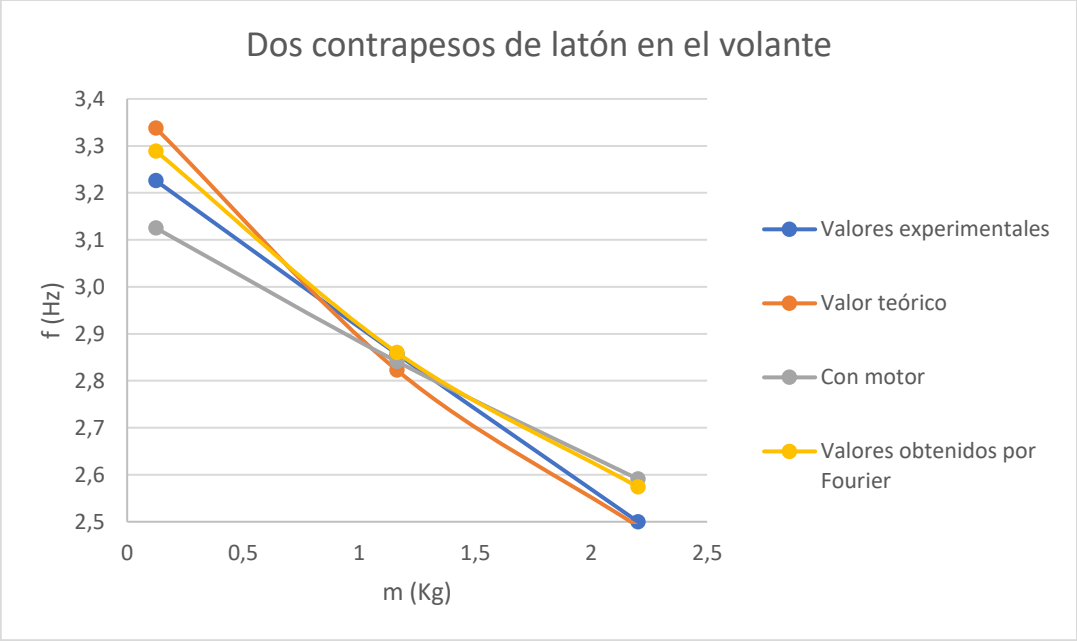


Figura 6.5. Representación de la frecuencia en función de la masa para dos contrapesas de latón.

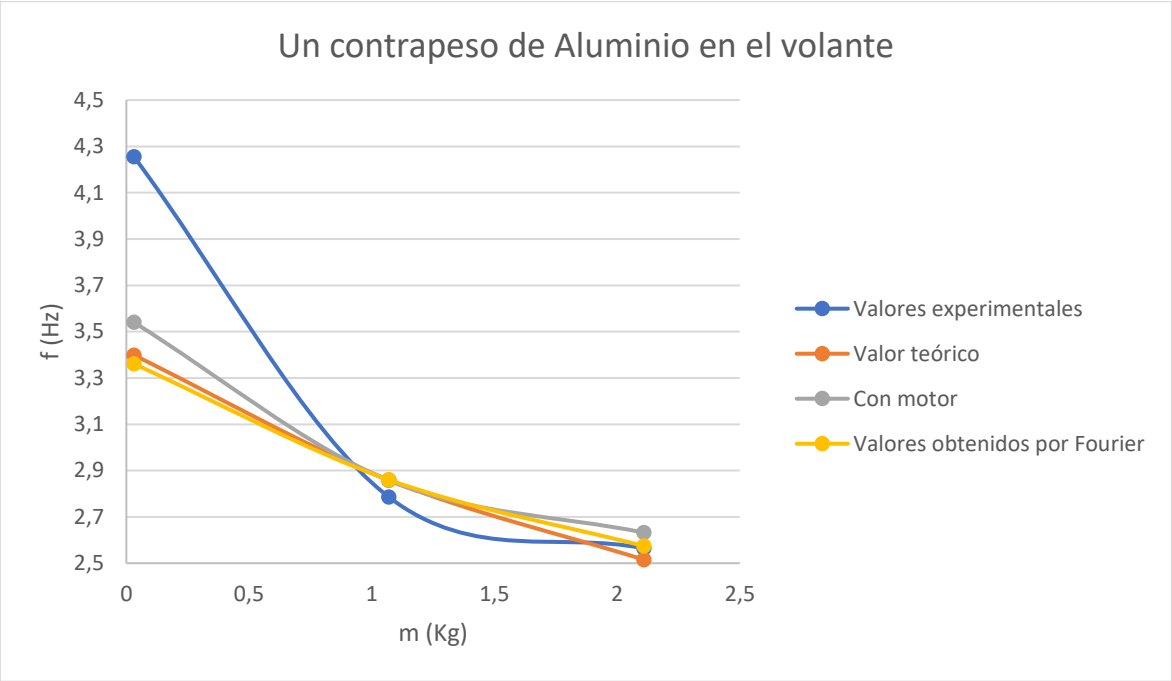


Figura 6.6. Representación de la frecuencia en función de la masa para una contrapesa de aluminio.

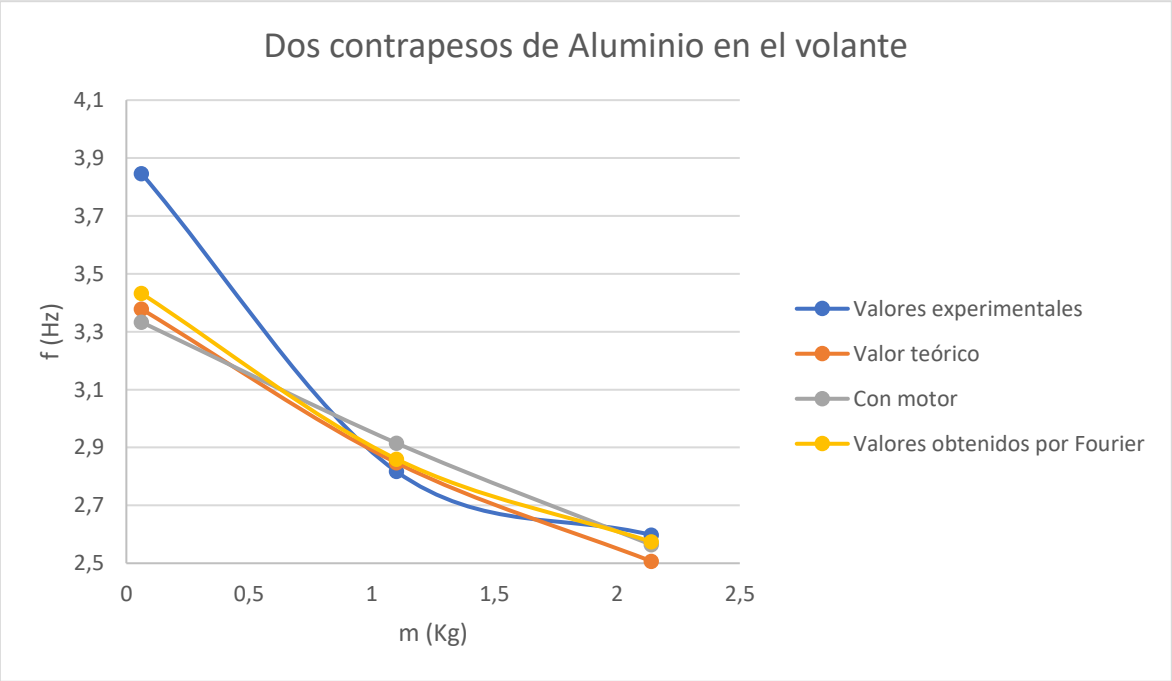


Figura 6.7. Representación de la frecuencia en función de la masa para dos contrapesas de aluminio.

7. CONCLUSIONES.

Se ha rehabilitado de forma satisfactoria la máquina de vibraciones libres y forzadas del laboratorio, ya que se ha dotado a la máquina de algunas prestaciones que no tenía antes o bien, se han modernizado alguna de las prestaciones anteriores. Por ejemplo, se detecta la posición del *carrito* con ayuda de un sensor, para después tratar mejor los datos o bien convertir la señal en dominio de la frecuencia. Con esta función, se calcula de una forma más directa la frecuencia de resonancia del sistema. Una prestación obtenida es que permite determinar el valor más exacto de distancia recorrida, gracias al apartado *Zoom*.

Otra mejora es el cambio del motor de la máquina. El motor antiguo, giraba a una velocidad de 3000 rpm. Mientras el motor nuevo, si se consulta la hoja de características que se encuentra en el Anexo 9.2.1., gira a 700 rpm. Cuando el motor se conecta a la placa reguladora, presenta una velocidad menor, 340 rpm. Esto es debido a que la placa reguladora de velocidad da una tensión máxima de salida de 5 V, mientras que el motor es de 12 V. Como recibe una tensión menor, el motor gira a una velocidad menor. También, se ha conseguido la posibilidad de variar la velocidad de giro del motor, lo que hace posible obtener otra forma de estudiar la resonancia del resorte.

Con respecto a la constante de rigidez, desde el punto de vista experimental, la rigidez del resorte es de 943,568 N/m, mientras la rigidez teórica es de 1147,564 N/m. Se puede apreciar que la diferencia entre ambos valores es considerable (un 18 %). Si se compara el valor obtenido en la simulación (1092,538 N/m) con el valor teórico, se tiene una diferencia del 5 %, por lo que el error es muy reducido. Esto puede ser porque para calcular el valor teórico de la rigidez se han utilizado aproximaciones para el valor del *módulo de elasticidad transversal* (*G*) y el *módulo de Young* (*E*), ya que cuando se trabaja con muelles dichos valores varían sensiblemente. También se ha utilizado una aproximación para determinar el valor de las espiras activas. Dichos valores pueden influir en el valor del resultado final.

Si se examina la tabla 6.4, se puede comprobar que el decremento logarítmico presenta para cada situación principal (cada situación de carga distinta en el volante), un mismo valor de forma aproximada, pues de tres casos solo hay uno que no coincide, por ejemplo (salvo el caso de una pesa de aluminio en el volante, ya que presenta mayor dispersión). Lo mismo ocurre con el coeficiente de amortiguamiento. Esto indica que la curva límite de las oscilaciones se mantiene conforme lo que dicta la teoría, exactamente, la expresión 4.30.

Finalmente, en lo referente a las frecuencias de resonancia, se pueden observar que los valores medidos experimentalmente, los obtenidos variando la velocidad de giro del motor a modo de comprobación y los valores teóricos son bastante similares. Esto indica que se ha cometido poco error en la medición. También se puede observar la tendencia que presentan los datos experimentales, de que cuando aumenta la masa del sistema, la frecuencia disminuye. Dicha tendencia es buena señal, ya que se obtienen resultados acordes con lo que la expresión teórica 4.48. Al aumentar la masa, el término *k/m* es menor. Por tanto esa tendencia es concordante con lo obtenido experimentalmente.

7.1. TRABAJOS FUTUROS.

La máquina restaurada en este trabajo, llevaba bastante tiempo sin utilizarse. Con la restauración realizada permite la posibilidad de realizar prácticas en asignaturas donde se tratan el tema de vibraciones u oscilaciones, así como asignaturas donde se realizan estudios de resortes. Esto puede resultar interesante, ya que a día de hoy no se realizan ninguna práctica de este tipo y sería una forma práctica de ilustrar este ámbito.

También se pretende adquirir un juego de dos o tres resortes, que presenten distintas rigideces. Es deseable debido a que este punto influye en el movimiento oscilatorio, por tanto se puede realizar otro estudio donde se tenga en cuenta los cambios de rigidez.

8. BIBLIOGRAFÍA.

[1] Beer, Johnston y Cornwell *Mecánica vectorial para ingenieros Dinámica*, novena edición, Mc Graw Hill. Capítulo 19, *Vibraciones mecánicas*.

[2] Domínguez Abascal, *Vibraciones Mecánicas*, Universidad de Sevilla, Escuela Superior de Ingenieros Industriales. Capítulo 1, *Vibraciones libres*.

[3] Santamarina Pol y Santamarina Siurana, *Vibraciones Mecánica en Ingeniería*, Universidad Politécnica de Valencia. *Generalidades*.

[4] Richard G. Budynas y J. Keith Nisbett, *Diseño en ingeniería mecánica de Shigley*, octava edición, Mc Graw Hill. Capítulo 10, *Resortes mecánicos*.

[5] Arduino, http://arduino.cl/que-es-arduino/ [Consultado el 27/06/2018].

[6] Richard G. Budynas y J. Keith Nisbett, *Diseño en ingeniería mecánica de Shigley*, octava edición, Mc Graw Hill. Capítulo 4, *Deflexión y rigidez*, apartado 4-8 *Teorema de Castigliano*.

[7] Richard G. Budynas y J. Keith Nisbett, *Diseño en ingeniería mecánica de Shigley*, octava edición, Mc Graw Hill. Capítulo 4, *Deflexión y rigidez*, apartado 4-7 *Energía de deformación*.

[8] RS Components, modelo *DC RS Pro 970 D161*, hoja de características.

[9] SICK, *MPS-128TSTU0*, hoja de características.

[10] Arduino, *Arduino DUE*, hoja de características.

[11] TOOGOO, modelo *011981*, hoja de características.

[12] Matlab, función str2double https://www.mathworks.com/help/matlab/ref/str2double.html [Consultado el 30/06/2018].

[13] Imagen de resortes de compresión, https://articulo.mercadolibre.com.mx/MLM-604881108-forney-72649-wire-resorte-de-compresion-916-pulgadas-por-3-_JM [Consultado el 9/07/2018].

[14] DirectIndustry, imagen de resorte de torsión, http://www.directindustry.es/fabricante-industrial/muelle-hilo-106377.html [Consultado el 9/07/2018].

[15] Transformada de Fourier, conceptos básicos, http://stel.ub.edu/labfon/sites/default/files/EFE-X-JBobadilla_PGomez_JBernal-FFT_una_vision_pedagogica.pdf [Consultado el 9/07/2018].

[16] Gandreoliva, imagen de suma de señales periódicas, http://gandreoliva.org/cursos/fg2/sonido.php [Consultado el 9/07/2018].

[17] Revista Dyna, Vibraciones mecánicas que son y como se evalua el riesgo https://www.revistadyna.com/busqueda/vibraciones-mecanicas-que-son-y-como-se-evalua-riesgo [Consultado el 11/07/2018].

[18] Energiza, medición de las vibraciones e interpretación de los resultados, http://www.energiza.org/mantenimiento-de-plantas/19-mantenimiento-de-plantas/516-analisis-de-vibraciones-una-tecnologia-clave-del-man-tenimiento-predictivo [Consultado el 11/07/2018].

[19] Acxess Spring, aplicaciones de los resortes de compresión. https://www.acxesspring.com/espanol/articulo-tecnico-de-resorte-de-compresion.html [Consultado el 11/07/2018].

[20] guiadelaindustria.com, imagen de resortes de extensión, http://www.guiadelaindustria.com/empresa/resortes-lacas-fca-de-resortes-nuestro-trabajo-esta-dedicado-a-solucionar-las-necesidades-de-nuestros-clientes/37156 [Consultado el 12/07/2018].

[21] iim, imagen del sistema resorte, masa y amortiguador, http://www.iim.unsj.edu.ar/control/curso/B3_est/ecdif31_3.htm [Consultado el 12/07/2018].

[22] Universidad de Sevilla, ecuaciones del movimiento sobreamortiguado y movimiento críticamente amortiguado, http://laplace.us.es/wiki/index.php/Movimiento_oscilatorio [Consultado el 12/07/2018].

[23] Schneider, definición de la frecuencia de resonancia, https://www.schneider-electric.es/es/work/insights/understanding-of-resonance-essential-for-solving-vibration-problems.jsp [Consultado el 12/07/2018].

[24] Definiciones de decremento logarítmico y de la relación de amortiguamiento, http://w3.mecanica.upm.es/~goico/mecanica/ICT/cbd/cbd-4.pdf [Consultado el 12/07/2018].

[25] Richard G. Budynas y J. Keith Nisbett, *Diseño en ingeniería mecánica de Shigley*, octava edición, Mc Graw Hill. Capítulo 10, *Resortes mecánicos*, *Tabla 10-4*.

[26] Richard G. Budynas y J. Keith Nisbett, *Diseño en ingeniería mecánica de Shigley*, octava edición, Mc Graw Hill. Capítulo 10, *Resortes mecánicos*, *Tabla 10-6*.

[27] Richard G. Budynas y J. Keith Nisbett, *Diseño en ingeniería mecánica de Shigley*, octava edición, Mc Graw Hill. Capítulo 10, *Resortes mecánicos*, *Tabla 10-7*.

[28] Richard G. Budynas y J. Keith Nisbett, *Diseño en ingeniería mecánica de Shigley*, octava edición, Mc Graw Hill. Capítulo 10, *Resortes mecánicos*, *Ecuación 10-57*.

[29]Esquema de un motor de corriente continua, https://es.slideshare.net/vcgj/motores-cc-48949345 [Consultado el 12/07/2018].

[30]Asifunciona, funcionamiento del motor de corriente continua, http://www.asifunciona.com/electrotecnia/af_motor_cd/af_motor_cd_6.htm [Consultado el 13/07/2018].

[31] Dibujo de reductor de velocidad de engranajes cilíndricos rectos, https://www.comsol.it/release/5.2a/multibody-dynamics-module [Consultado el 13/07/2018].

[32] Arduino, Enlace de descarga del IDE Arduino, https://www.arduino.cc/en/Main/Software [Consultado el 13/07/2018].

[33] Recoge de ejercicios, esquema de una vibración senoidal, http://www.xtec.cat/~ocasella/exercici/mvh2.htm [Consultado el 17/07/2018].

[34]IBMKnowledgeCentre, definición de vibración aleatoria, https://www.ibm.com/support/knowledgecenter/es/POWER7/p7ebe/p7ebevibrationandshock.htm [Consultado el 17/07/2018].

[35] Definición de frecuencia natural, http://azimadli.com/vibman-spanish/frecuencianatural1.htm [Consultado el 17/07/2018].

[36] issuu, aplicaciones de la Trasformada de Fourier, https://issuu.com/jenniffer2012/docs/aplicaciones_de_la_transformada_de_fourier. [Consultado el 17/07/2018].

[37] Díaz Carrillo, *Cinemática y Dinámica de Máquinas*, Universidad de Jaén, Escuela Politécnica Superior de Linares. Capítulo 20, *Equilibrado de mecanismos*.

[38] lightpath, Tarjetas de desarrollo, http://www.lightpath.io/tarjetas-de-desarrollo/ [Consultado el 18/07/2018].

[39] Acamica, Ventajas e inconvenientes de Arduino, https://edgardosilvi.wordpress.com/2016/02/29/acamica-ventajas-y-desventajas-de-arduino/ [Consultado el 18/07/2018].

[40] Características de Launchpad MP430, https://www.taringa.net/posts/hazlo-tu-mismo/17477034/Arduino-Vs-Launchpad-MP430-Vs-Launchpad-Tiva.html [Consultado el 18/07/2018].

[41] DomePiDomotics, Ventajas y desventajas de RaspberryPi, http://domepidomotics.blogspot.com/2015/05/ventajas-y-desventajas-del-uso-de-la.html [Consultado el 18/07/2018].

[42] Fiebredigital, características de RaspberryPi, https://fiebredigital.com/aparatos/todo-sobre-raspberry-pi-parte-2-836 [Consultado el 18/07/2018].

[43] Lignux, características de ODroidC2, https://lignux.com/raspberry-pi-tiene-un-nuevo-rival-llega-odroid-c2-un-mini-ordenador-superior-por-5-dolares-mas/ [Consultado el 18/07/2018].

[44] Sabien, características ODroidC2, http://www.sabien.upv.es/presente-los-mini-ordenadores-raspberry-pi-3-otras-alternativas/ [Consultado el 18/07/2018].

[45] Sensores resistivos, http://www.investigacion.frc.utn.edu.ar/sensores/Tutorial/TECNO2.pdf [Consultado el 18/07/2018].

[46] Ingeniería Mecafenix, sensores capacitivos, http://www.ingmecafenix.com/automatizacion/sensor-proximidad-capacitivo/ [Consultado el 18/07/2018].

[47] SEAS, sensores inductivos, https://www.seas.es/blog/automatizacion/sensores-inductivos/ [Consultado el 18/07/2018].

[48] Universidad Nacional de San Luis, sensores de efecto Hall, http://www0.unsl.edu.ar/~interfases/labs/lab09.pdf [Consultado el 18/07/2018].

[49] Monografías, esquema de un motor paso a paso de imán permanente, https://www.monografias.com/trabajos93/motor-paso-paso/motor-paso-paso.shtml [Consultado el 18/07/2018].

[50] Esquema de un motor paso a paso de reluctancia variable, http://www.wikiwand.com/es/Motor_de_reluctancia_variable [Consultado el 18/07/2018].

[51] Sapiensman, dibujo de un motor paso a paso híbrido, http://www.sapiensman.com/tecnoficio/electricidad/velocidad_de_motores_electricos4.php [Consultado el 18/07/2018].

9. ANEXOS.

En este apartado se recogen las líneas de código de la GUIDE en Matlab, hojas de características, un ejemplo de práctica que se puede realizar y los planos y esquemas necesarios para la realización de este trabajo.

9.1. PROGRAMA EN MATLAB.

En este apartado se muestra el script del programa que se trasmite al Arduino:

```
function varargout = Programa(varargin) %Llamada de Matlab al programa.
% PROGRAMA MATLAB code for Programa.fig
%       PROGRAMA, by itself, creates a new PROGRAMA or raises the existing
%      singleton*.
%
%       H = PROGRAMA returns the handle to a new PROGRAMA or the handle to
%      the existing singleton*.
%
%      PROGRAMA('CALLBACK',hObject,eventData,handles,...) calls the local
%       function named CALLBACK in PROGRAMA.M with the given input arguments.
%
%       PROGRAMA('Property','Value',...) creates a new PROGRAMA or raises the
%      existing singleton*.  Starting from the left, property value pairs are
%       applied to the GUI before Programa_OpeningFcn gets called. An
%       unrecognized property name or invalid value makes property application
%        stop.  All inputs are passed to Programa_OpeningFcn via varargin.
```

```
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows
only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help Programa

% Last Modified by GUIDE v2.5 13-Jun-2018 22:47:24

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @Programa_OpeningFcn, ...
                   'gui_OutputFcn',  @Programa_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before Programa is made visible.
function Programa_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
```

```
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to Programa (see VARARGIN)

% Choose default command line output for Programa
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes Programa wait for user response (see UIRESUME)
% uiwait(handles.figure1);



% --- Outputs from this function are returned to the command line.
function varargout = Programa_OutputFcn(hObject, eventdata,
handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;
clear all;
global k y1 DUE;  %Definición de variables importantes que se usarán
en el programa.
k=0;              %Número de datos medidos.
DUE=arduino('COM7');     %Definición del puerto USB al que está
conectado Arduino.



% --- Executes on slider movement.
```

```
function slider1_Callback(hObject, eventdata, handles) %Definición del slider que indica la velocidad de giro del motor.

% hObject    handle to slider1 (see GCBO)

% eventdata  reserved - to be defined in a future version of MATLAB

% handles    structure with handles and user data (see GUIDATA)


global DUE;  %Con esta llamada se indica el Arduino al que se está mandando la señal.

slider=get(hObject,'Value'); %Con la función get se adquiere el valor desde la ventana.

slider1=slider*20; %Se multiplica por 20, para obtener un slider de 100 pasos.

slider2=abs(-0.0008*slider1^3+0.1351*slider1^2-
2.6772*slider1+6.5663); %Con slider2 se obtiene la aproximación de la velocidad de giro.

slider3=slider2*(2*3.14159)*60^(-1); %Obtención de la frecuencia de giro en Hz.

set(handles.edit1,'String',num2str(slider1)); %Representación del valor del número de velocidad en el programa.

set(handles.rpm,'String',num2str(slider2)); %Representación de la velocidad en revoluciones por minuto en el programa.

set(handles.hz,'String',num2str(slider3)); %Representación de la frecuencia de giro en el programa.

analogWrite(DUE,10,slider1); %En esta línea, el programa indica a que Arduino debe ir la orden de aumento o descenso de la tensión eléctrica y el pin por donde debe salir esa orden.

guidata(hObject,handles); %La función guidata recoge y almacena la información de esta parte del programa.


% Hints: get(hObject,'Value') returns position of slider

%          get(hObject,'Min') and get(hObject,'Max') to determine range of slider



% --- Executes during object creation, after setting all properties.

function slider1_CreateFcn(hObject, eventdata, handles)

% hObject    handle to slider1 (see GCBO)

% eventdata  reserved - to be defined in a future version of MATLAB

% handles    empty - handles not created until after all CreateFcns called
```

```
% Hint: slider controls usually have a light gray background.
if                          isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end



function edit1_Callback(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit1 as text
%        str2double(get(hObject,'String')) returns contents of edit1
as a double



% --- Executes during object creation, after setting all properties.
function edit1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if       ispc       &&       isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function edit2_Callback(hObject, eventdata, handles) %En este apartado se recoge el valor de Ymax para la medición de las oscilaciones.
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit2 as text
%        str2double(get(hObject,'String')) returns contents of edit2 as a double

handles.dato1=get(hObject,'String');
handles.Ymax=str2double(handles.dato1);  %La función str2double convierte los caracteres en números [12].
guidata(hObject,handles);



% --- Executes during object creation, after setting all properties.
function edit2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit3_Callback(hObject, eventdata, handles) %En este apartado se recoge el valor de Xmax para la medición de las oscilaciones, en segundos.
% hObject    handle to edit3 (see GCBO)
```

```
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% Hints: get(hObject,'String') returns contents of edit3 as text
%        str2double(get(hObject,'String')) returns contents of edit3
as a double


handles.dato2=get(hObject,'String');
handles.Xmax=(str2double(handles.dato2))/0.032;  %Se divide por
0,032 porque el Arduino tarda en tomar una medida 0,032 s. Si no se
hace esta división, se recogería la señal sin estar en función del
tiempo.
guidata(hObject,handles);



% --- Executes during object creation, after setting all properties.
function edit3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called


% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if       ispc       &&       isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function  edit4_Callback(hObject,  eventdata,  handles)  %En  este
apartado se recoge el valor de Ymin para la medición de las
oscilaciones.
% hObject    handle to edit4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of edit4 as text
%        str2double(get(hObject,'String')) returns contents of edit4
as a double


handles.dato3=get(hObject,'String');
handles.Ymin=str2double(handles.dato3);
guidata(hObject,handles);




% --- Executes during object creation, after setting all properties.
function edit4_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if       ispc       &&       isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end




% --- Executes on button press in salir.
function salir_Callback(hObject, eventdata, handles) %Este apartado
programa el botón salir. Cuando se desea cerrar el programa es
recomendable utilizar este botón, para borrar todas las variables,
así como cerrar el puerto USB donde está conectado el Arduino.
% hObject    handle to salir (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
salida=questdlg('¿Seguro      que      quiere      salir      del
programa?','SALIR','Sí','No','No'); %La función questdlg hace que
aparezca una ventana donde aparece una pregunta que se desea, así
```

```
como la respuesta. La respuesta repetida en la línea de programación
aparece resaltada, simbolizando la respuesta preferible.

if strcmp(salida,'No') %Esta línea indica al programa la respuesta
elegida.

    return;

end

clear,clc,close all



% --- Executes on button press in pb1.

function pb1_Callback(hObject, eventdata, handles)%Este apartado
define lo que hace el programa cuando se selecciona Iniciar medida.

% hObject    handle to pb1 (see GCBO)

% eventdata  reserved - to be defined in a future version of MATLAB

% handles    structure with handles and user data (see GUIDATA)


global k y1 c relacion DUE %Variables a utilizar.


axes(handles.axes1); %Ejes donde se va a representar la señal.

y1=[]; %Definición del vector donde se almacenarán los valores
recogidos.

c=zeros(1,handles.Xmax); %Vector lleno de ceros de longitud Xmax.

for k=1:1:handles.Xmax

    c(k)=c(k)+0.032*k; %Este vector "avanza" 0,032 s por cada dato
tomado, es como si fuera un "cronómetro".

    handles.ejex(k,1)=c(k); %Definición del eje X. Como el vector c
está aumentando su longitud constantemente, el eje X aumentará a
medida que pase el tiempo.

    y1(k,1)=relacion*((5/1023)*DUE.analogRead(0)); %En esta línea
se obtiene la señal leída por el sensor. El termino relación (se
define en líneas posteriores)convierte la señal de voltios a
milímetros. El término 5/1023 son factores de conversión para
obtener una señal, dimensionalmente coherente. El término
analogRead(0)indica el pin por donde el programa lee la señal.

    plot(handles.ejex,y1,'LineWidth',2); grid on; %Indicación de la
representación de la señal.

    axis([0 c(k) handles.Ymin handles.Ymax]); %Definición de los
ejes.

    pause(0.01)%Tiempo entre medidas.
```

```
end



function xmax_Callback(hObject, eventdata, handles)%Con este apartado se define el valor máximo de tiempo que se quiere representar en el recuadro de Zoom.
% hObject    handle to xmax (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% Hints: get(hObject,'String') returns contents of xmax as text
%        str2double(get(hObject,'String')) returns contents of xmax as a double
handles.dato4=get(hObject,'String');
handles.Xmaxz=(str2double(handles.dato4)/0.032);
guidata(hObject,handles);



% --- Executes during object creation, after setting all properties.
function xmax_CreateFcn(hObject, eventdata, handles)
% hObject    handle to xmax (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called


% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function xmin_Callback(hObject, eventdata, handles) %Con este apartado se define el valor mínimo de tiempo que se quiere representar en el recuadro de Zoom.
% hObject    handle to xmin (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of xmin as text
%        str2double(get(hObject,'String')) returns contents of xmin as a double
handles.dato5=get(hObject,'String');
handles.Xminz=((str2double(handles.dato5))/0.032);
guidata(hObject,handles);



% --- Executes during object creation, after setting all properties.
function xmin_CreateFcn(hObject, eventdata, handles)
% hObject    handle to xmin (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if       ispc       &&       isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



% --- Executes on button press in zoom.
function   zoom_Callback(hObject,   eventdata,   handles)%En   este apartado, se programa la función del botón Aplicar Zoom.
% hObject    handle to zoom (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
global k y1 c relacion DUE


axes(handles.axes3)%Definición de los ejes donde se representa la
ampliación.
handles.ejex=c;
plot(handles.ejex,y1)
axis([(handles.Xminz*0.032)   (handles.Xmaxz*0.032)   handles.Yminz
handles.Ymaxz]);




function  ymaxz_Callback(hObject,  eventdata,  handles)  %Con  este
apartado se define el valor máximo de la distancia que se quiere
representar en el recuadro de Zoom.
% hObject    handle to ymaxz (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% Hints: get(hObject,'String') returns contents of ymaxz as text
%        str2double(get(hObject,'String')) returns contents of ymaxz
as a double


handles.dato7=get(hObject,'String');
handles.Ymaxz=str2double(handles.dato7);
guidata(hObject,handles);



% --- Executes during object creation, after setting all properties.
function ymaxz_CreateFcn(hObject, eventdata, handles)
% hObject    handle to ymaxz (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called


% Hint: edit controls usually have a white background on Windows.
```

```
%       See ISPC and COMPUTER.
if       ispc       &&       isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function  yminz_Callback(hObject,  eventdata,  handles)  %Con  este
apartado se define el valor mínimo de la distancia que se quiere
representar en el recuadro de Zoom.
% hObject    handle to yminz (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of yminz as text
%       str2double(get(hObject,'String')) returns contents of yminz
as a double

handles.dato8=get(hObject,'String');
handles.Yminz=str2double(handles.dato8);
guidata(hObject,handles);
```

```
% --- Executes during object creation, after setting all properties.
function yminz_CreateFcn(hObject, eventdata, handles)
% hObject    handle to yminz (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if       ispc       &&       isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
```

```
end



function dmax_Callback(hObject, eventdata, handles) %Con este apartado se define el valor máximo de la distancia que puede recorrer el carrito. Dicho valor es necesario para definir el término relacion.
% hObject    handle to dmax (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% Hints: get(hObject,'String') returns contents of dmax as text
%        str2double(get(hObject,'String')) returns contents of dmax as a double


distancia=get(hObject,'String');
ndistancia=str2double(distancia);
handles.dmax=ndistancia;
guidata(hObject,handles);


% --- Executes during object creation, after setting all properties.
function dmax_CreateFcn(hObject, eventdata, handles)
% hObject    handle to dmax (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called


% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if       ispc       &&       isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

```
function vmax_Callback(hObject, eventdata, handles) %Con este apartado se define el valor máximo del voltaje que soporta el Arduino con el que se trabaja. Dicho valor es necesario para definir el término relacion.
% hObject    handle to vmax (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of vmax as text
%        str2double(get(hObject,'String')) returns contents of vmax as a double

voltaje=get(hObject,'String');
nvoltaje=str2double(voltaje);
handles.vmax=nvoltaje;
guidata(hObject,handles);



% --- Executes during object creation, after setting all properties.
function vmax_CreateFcn(hObject, eventdata, handles)
% hObject    handle to vmax (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if       ispc       &&       isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function mmv_Callback(hObject, eventdata, handles)
```

```
% hObject    handle to mmv (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of mmv as text
%        str2double(get(hObject,'String')) returns contents of mmv
as a double




% --- Executes during object creation, after setting all properties.
function mmv_CreateFcn(hObject, eventdata, handles)
% hObject    handle to mmv (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if       ispc       &&       isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



% --- Executes on button press in rmmv.
function  rmmv_Callback(hObject,  eventdata,  handles)  %Con  este
apartado se define el termino relacion. Dicho termino convierte la
señal en voltios a milímetros.
% hObject    handle to rmmv (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

global relacion
d=handles.dmax;
v=handles.vmax;
```

```
relacion=d/v; %Definición de la relación.
set(handles.mmv,'String',relacion); %Con esta línea se representa el valor de la relación en el programa.



function rpm_Callback(hObject, eventdata, handles)
% hObject    handle to rpm (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of rpm as text
%        str2double(get(hObject,'String')) returns contents of rpm as a double



% --- Executes during object creation, after setting all properties.
function rpm_CreateFcn(hObject, eventdata, handles)
% hObject    handle to rpm (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if       ispc       &&       isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function hz_Callback(hObject, eventdata, handles)
% hObject    handle to hz (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
```

```
% handles    structure with handles and user data (see GUIDATA)


% Hints: get(hObject,'String') returns contents of hz as text
%        str2double(get(hObject,'String')) returns contents of hz
as a double



% --- Executes during object creation, after setting all properties.
function hz_CreateFcn(hObject, eventdata, handles)
% hObject    handle to hz (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if       ispc       &&       isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end




% --- Executes on button press in frecuencia.
function frecuencia_Callback(hObject, eventdata, handles) %En estas
líneas, se programa las órdenes para obtener la frecuencia según
Fourier.
% hObject    handle to frecuencia (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

global k y1 c relacion DUE

maximo1=-200000;
maximo2=-100000;
maximo3=-50000;
```

```
maximo4=-25000; %Estos valores se utilizan para buscar la frecuencia más alta. Como son unos valores muy bajos, cualquier término será mayor que estos parámetros.


axes(handles.axes5)

F=31.25; %Frecuencia de muestreo. Inversa de 0,032 s.

T=1/F; %Periodo de muestreo. Son los 0,032 s.

t=(0:k-1)*T; %Vector de tiempo.

f=F*(0:(k/2))/k;

y2=fft(y1); %La función fft determina la trasformada de Fourier de y1.

y3=abs(y2/k);

y4= y3(1:k/2+1);

y4(2:end-1)=2*y4(2:end-1); %Este criterio se usa para cortar la representación de la trasformada y ahorrar espacio.

b=length(y4);

plot(f,y4);

xlabel('f (Hz)');

ylabel('y (f)');

for x=2:1:b;

    if y4(x)>=maximo1;

        maximo1=y4(x);

        maximo3=f(x);

    end

%Esta primera condición de if se dedica a buscar el primer valor máximo. Este primer valor se encuentra al principio, cerca de cuando x vale 0, por tanto no es un valor lógico y carece de sentido. Por esa razón se busca el segundo valor más alto.

    if y4(x)>=maximo2 & y4(x)<=maximo1

        maximo2=y4(x);

        maximo4=f(x);

    end

%Con esta segunda condición, se obtiene el valor lógico de la frecuencia, que se corresponde con el valor de maximo4.

end

set(handles.valor,'String',maximo4);    %Representación de la frecuencia.
```

9.2. HOJAS DE CARACTERÍSTICAS.

En este capítulo se presentan las hojas de características de los elementos eléctricos incorporados en la máquina:

9.2.1. Motor.

9.2.2. Sensor.

9.2.3. Arduino.

9.2.4. Regulador.

9.2.1. Motor.

El motor eléctrico utilizado, es de *RS Components* modelo DC RS Pro 970 D161. Dicho motor presenta una caja reductora de velocidad con engranajes cilíndricos rectos.
Sus características se reúnen en la tabla 8.1 [8]:

DC RS Pro 970 D61	
Velocidad de salida	753 rpm
Tensión de alimentación	12 V
Par de salida	2000 g*cm
Tipo de motor DC	Con escobillas
Diámetro del eje	6 mm
Potencia nominal	19,8 W
Corriente nominal	2,81 A
Relación de reducción	16:1
Peso	252 g

Tabla 8.1. Características del motor.

9.2.2. Sensor.

El sensor incorporado en la máquina, es un sensor magnético de posición de la marca *SICK*. El modelo es el *MPS-128TSTU0.* Las características se recogen en la tabla 8.2 [9]:

Sensor MPS-128TSTU0	
Margen de medida	128 mm
Tipo de salida	Analógica
Tensión de alimentación	15 V a 30 V
Tensión de salida	0 V a 10 V
Consumo de corriente	≤ 22 mA
Demora antes de disponibilidad	1,5 s
Intensidad magnética requerida	3mT
Resolución	0,05 mm
Velocidad máxima	3 m/s

Tabla 8.2. Características del sensor.

9.2.3. Arduino.

El modelo de Arduino utilizado, es el *Arduino DUE*. Se ha utilizado dicho modelo porque tiene un microprocesador más rápido, por tanto es ideal para realizar mediciones. Sus características aparecen en la tabla 8.3 [10]:

Arduino DUE	
Microcontrolador	AT91SAM3X8E
Voltaje de operación	3,3 V
Voltaje de entrada	De 7 V a 12 V
Pines de entrada y salida digitales	54 pines, de ellos 12 proveen salida PWM
Pines de entrada analógicos	12
Corriente de salida en los pines 1/0	130 mA

Corriente de salida en el pin 3,3 V	800 mA
Corriente de salida en el pin 5 V	800 mA
Memoria Flash	512 KB
SRAM	96 KB
Velocidad de reloj	84 MHz

Tabla 8.3. Características de la placa *Arduino DUE.*

9.2.4. Regulador.

El regulador es de la marca TOOGOO y el modelo que se ha utilizado, es el 011981. Sus características se representan en la tabla 8.4 [11]:

TOOGOO 011981	
Voltaje de funcionamiento	De 6 V a 90 V
Tensión de control (de salida)	De 0 V a 5 V
Potencia controlada	De 0,01 W a 1000 W
Corriente de funcionamiento	5 mA

Tabla 8.4. Características del regulador.

9.3. EJEMPLO DE PRÁCTICA CON LA MÁQUINA.

En este anexo, se presenta un ejemplo de una práctica que se puede realizar con esta máquina para uso docente. Dicha práctica está orientada a estudiar el comportamiento del resorte.
La práctica en cuestión, tendría los siguientes apartados: Una introducción, los objetivos y material, la metodología, resultados y conclusiones.

9.3.1. Introducción.

Un muelle o resorte es un elemento mecánico capaz de deformarse sin sufrir deformación. Tienen muchas aplicaciones, algunos ejemplos son ejercer fuerzas (como los muelles de los amortiguadores de las puertas), proporcionar flexibilidad y capacidad de almacenar o absorber energía (por ejemplo, el *muelle real* de un reloj de cuerda) [4].
Hay diversos tipos de resortes: de compresión, de extensión, de torsión, ballestas, cónicos...

9.3.2. Objetivos y material.

En la siguiente práctica se pretende obtener tres objetivos:

- Determinar la *constante de rigidez* (*k*) del muelle.
- Estudiar el amortiguamiento.
- Determinar la *frecuencia de resonancia* del resorte para varios estados de carga.

Todos los ensayos se realizarán con los siguientes materiales:
Máquina de vibraciones libres y forzadas: consta de un carrito que se encuentra sujeto al muelle que se quiere estudiar. Acoplado al carrito hay un motor con un volante colocado en su eje.
Pesas: hay dos tipos de pesas, unas se colocan en el vástago del carrito (de 1,039 Kg cada una) y otras de latón (de 62 g) o de aluminio (de 31 g), que se atornillan en el volante. Estas últimas serán las masas excéntricas que al girar harán que el carrito suba y baje y por tanto el muelle se estira y se comprime.
Resorte a estudiar: es un muelle de extensión. Se encuentra sujeto de forma fija en la parte superior de la máquina y la parte inferior está sujeta al carrito.

Programa en *Matlab*: a partir de este programa se puede observar la posición y desplazamiento del carrito (y consecuentemente del extremo móvil del resorte) y variar la velocidad de giro del motor. La máquina y el programa se comunica mediante una placa *Arduino DUE*.

9.3.3. Fundamentos.

El muelle de la máquina presenta las siguientes partes: el cuerpo (donde están las espiras), la espira longitudinal (hay dos, una en cada extremo del cuerpo) y dos tramos rectos. Por tanto, es como si hubiera cinco resortes en serie. Para calcular la rigidez se aplica esta fórmula:

$$\frac{1}{k} = \sum_{i=1}^{n} \frac{1}{k_i} \quad (9.1)$$

Desarrollando la expresión 9.1 para despejar *k* se tiene:

$$k = \frac{k_1 k_2 k_3}{k_2 k_3 + 2k_1 k_3 + 2k_2 k_1} \quad (9.2)$$

Donde k_1 es la rigidez del cuerpo, k_2 la de las espiras longitudinales y k_3 la de los tramos rectos. Sus expresiones son las siguientes:

$$k_1 = \frac{d^4 G}{8D^3 N_a} \quad (9.3)$$

$$k_2 = \frac{G d^4}{2 r_2 D^2} \quad (9.4)$$

$$k_3 = \frac{4l}{\pi d^2 E} \quad (9.5)$$

Para calcular los parámetros de la amortiguación, se parte de la ecuación del sistema:

$$m\ddot{x} + c\dot{x} + kx = 0 \quad (9.6)$$

Despejando esa ecuación diferencial se tiene para *n* ciclos:

$$x_n = Ce^{-\frac{c}{2m}(t_0+nT)} = x_0 e^{-\frac{c}{2m}nT} \quad (9.7)$$

El valor $\frac{c}{2m}T = \delta$ es el decremento logarítmico y es uno de los parámetros a obtener. Por tanto, la expresión 9.7 queda de la siguiente forma:

$$x_n = x_0 e^{-n\delta} \quad (9.8)$$

Despejando δ:

$$\delta = \frac{\ln\left(\frac{x_0}{x_n}\right)}{n} \quad (9.9)$$

Para relacionarlo con la relación de amortiguamiento, se hace lo siguiente:

$$T = \frac{2\pi}{\omega_d} \quad (9.10)$$

$$\delta = \frac{c}{2m}T = \frac{c}{2m}\frac{2\pi}{\omega_d} = \frac{c}{2m}\frac{2\pi}{\omega_n\sqrt{1-\xi^2}};$$

$$\delta = \frac{2\pi\xi}{\sqrt{1-\xi^2}} \quad (9.11)$$

Para el coeficiente de amortiguación se aplica la siguiente expresión:

$$C = 2\xi\sqrt{km} \quad (9.12)$$

Para el caso de una masa colgada en el extremo de un resorte , la frecuencia de resonancia se obtiene a partir de la ecuación del sistema:

$$\sum F = ma = m\ddot{x} \quad (9.13)$$

En ese caso, la única fuerza existente es la que soporta el resorte:

$$F = kx \quad (9.14)$$

Por tanto, queda de la siguiente forma:

$$m\ddot{x} + kx = 0 \quad (9.15)$$

Dividiendo ambos términos por la masa:

$$\ddot{x} + \frac{k}{m}x = 0;\ \ddot{x} + {\omega_n}^2 x = 0 \quad (9.16)$$

Donde el término ω_n es la *frecuencia natural del sistema* en rad/s:

$$\omega_n = \sqrt{\frac{k}{m}} \quad (9.17)$$

Para obtenerla en hertzios (*Hz*) a la ecuación 9.17 se divide entre 2π, ya que un ciclo equivale a 2π radianes:

$$\omega_n = \frac{1}{2\pi}\sqrt{\frac{k}{m}} \quad (9.18)$$

Ya que un ciclo equivale a 2π radianes. Dicho valor es la frecuencia de resonancia del sistema constituido por una masa colgada de un resorte.

9.3.4. Metodología.

Primero se estudiará la *constante de rigidez* (*k*) del resorte. Para ello se deben realizar los siguientes pasos:

- En el programa, dentro del recuadro *Relación Voltaje-Distancia*, se define el valor máximo que es capaz de recorrer el carrito, que son 100 mm y el voltaje máximo. Como la máquina trabaja con una placa de *Arduino DUE*, la tensión máxima será la máxima que puede soportar la placa, es decir 3,3 V. Después se selecciona el botón *Relación mm-V*, para que el programa calcule dicha relación y pueda convertir la señal del sensor de voltios a milímetros.
- A continuación, en *Ymáx* se escribe 100, en *Ymin* 0 y en tiempo de medida se coloca un valor de tiempo (en segundos) cualquiera, pero que sea corto.
- Se estudia dos situaciones: *carrito* con el conjunto motor, volante y las dos pesas de latón y con el conjunto con las dos pesas de aluminio. A su vez se estudiarán los casos de sin pesas en el vástago, con una pesa y con dos pesas. La *posición cero* será la que no tiene pesas. Una vez preparada la *posición cero*, se selecciona *Inicio de medida* y en la gráfica aparecerá la señal del sensor (que en este caso será una recta horizontal). Para ver la posición, dicho punto es donde corta la recta con la escala. Luego se hace lo mismo con una pesa de 1,039 Kg y luego con dos pesas de 1,039 Kg. Para obtener valores más exactos sobre la posición, se usa el recuadro *Zoom*.

  Para estudiar el amortiguamiento, seguimos los siguientes pasos:

- Se define la *Relación mm-V* (si no se ha cerrado el programa no hace falta, porque está definido de antes). Luego, como en el caso anterior, definimos *Ymáx* como 100 e *Ymin* 0 y se define un tiempo de medida (12, 15 o 17 segundos).
- En este caso, se estudiarán 4 situaciones: una pesa de aluminio en el volante, dos pesas de aluminio, una pesa de latón y dos pesas de latón. En cada situación, se estudiará a su vez los casos de sin pesas en el vástago, una pesa y dos pesas.
- Manualmente, se lleva el carrito en la posición más baja del recorrido, consecuentemente el muelle se estira.
- Después de seleccionar *Inicio de medida*, se suelta el *carrito*. El muelle realizará oscilaciones hasta que se estabilice.
- Por último, se mide la *posición media* (que será cuando el carrito se pare cuando esté estabilizado), luego se evalúa la amplitud de tres oscilaciones distintas, por ejemplo, la

oscilación 1, 10 o 15 y 20. Para obtener mayor exactitud en la mediada usamos el recuadro *Zoom*. A partir de estos datos se podrán determinar los valores del *Decremento Logarítmico*, la *Relación de amortiguamiento* y el *Coeficiente de Amortiguamiento*.

- Se deberá realizar los mismos pasos para los distintos estados de carga.

  Y para la *frecuencia de resonancia* seguimos los siguientes pasos:
- Como en el caso de la amortiguación, estudiaremos 4 situaciones: una pesa de aluminio en el volante, dos pesas de aluminio, una pesa de latón y dos pesas de latón. En cada situación, se estudiará a su vez los casos de sin pesas en el vástago, una pesa y dos pesas.
- Definimos los valores de los ejes, llevamos el carrito a la posición más baja, le damos a *Inicio de medida* y soltamos el carro.
- Después de que se haya estabilizado, medimos el tiempo que ha tardado en realizar la primera oscilación (en el eje *X* se registran los segundos).
- Luego calculamos la frecuencia haciendo la inversa de dicho tiempo. Dicha frecuencia será la frecuencia de resonancia.
- Por último, comprobamos si los valores de resonancia se cumplen haciendo girar el motor. Cuando el muelle oscile de forma descontrolada, estará en resonancia y debe tener la misma frecuencia que la medida anteriormente.

  También se deberá realizar los mismos pasos para los distintos estados de carga.

9.3.5. Resultados.

En este apartado, se deberán obtener los resultados experimentales y teóricos de la Constante de rigidez, los parámetros de la amortiguación y la frecuencia de resonancia.
También se pueden realizar gráficas de la fuerza aplicada en el resorte en función del desplazamiento, para la rigidez así como la frecuencia de resonancia en función de la masa.

9.3.6. Conclusiones.

Y en este apartado se definen las conclusiones que se pueden deducir de los resultados anteriores.

9.4. PLANOS Y ESQUEMAS.

En este anexo se recogen los siguientes planos:

9.4.1. Plano del soporte del motor.

9.4.2. Plano del soporte del sensor.

9.4.3. Plano caja del Arduino.

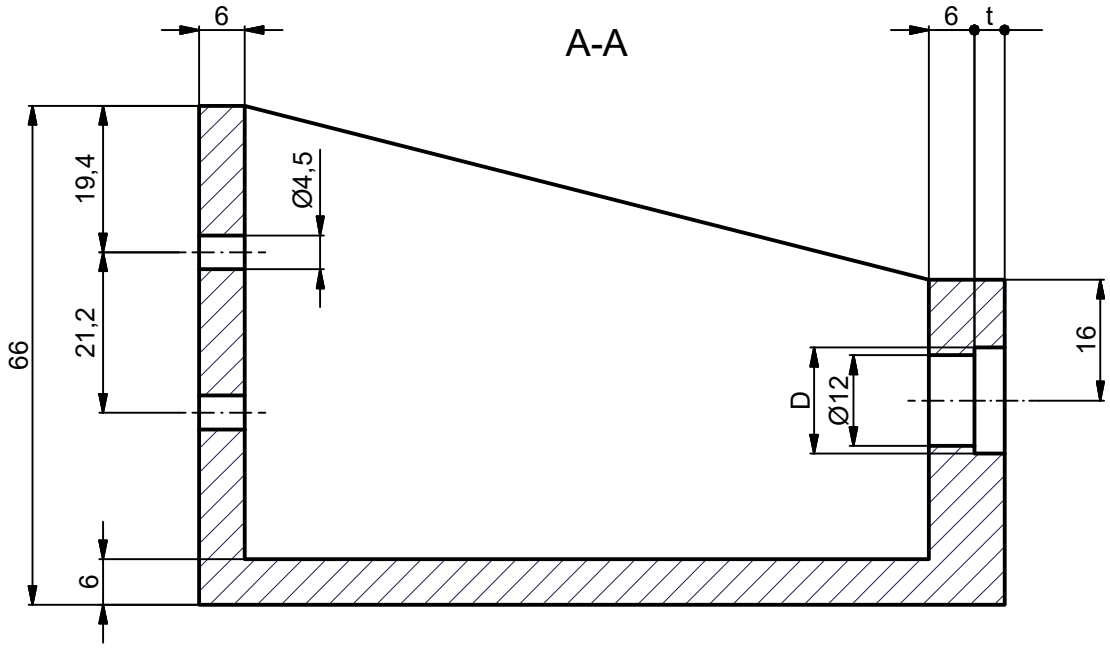
9.4.4. Plano caja del regulador.

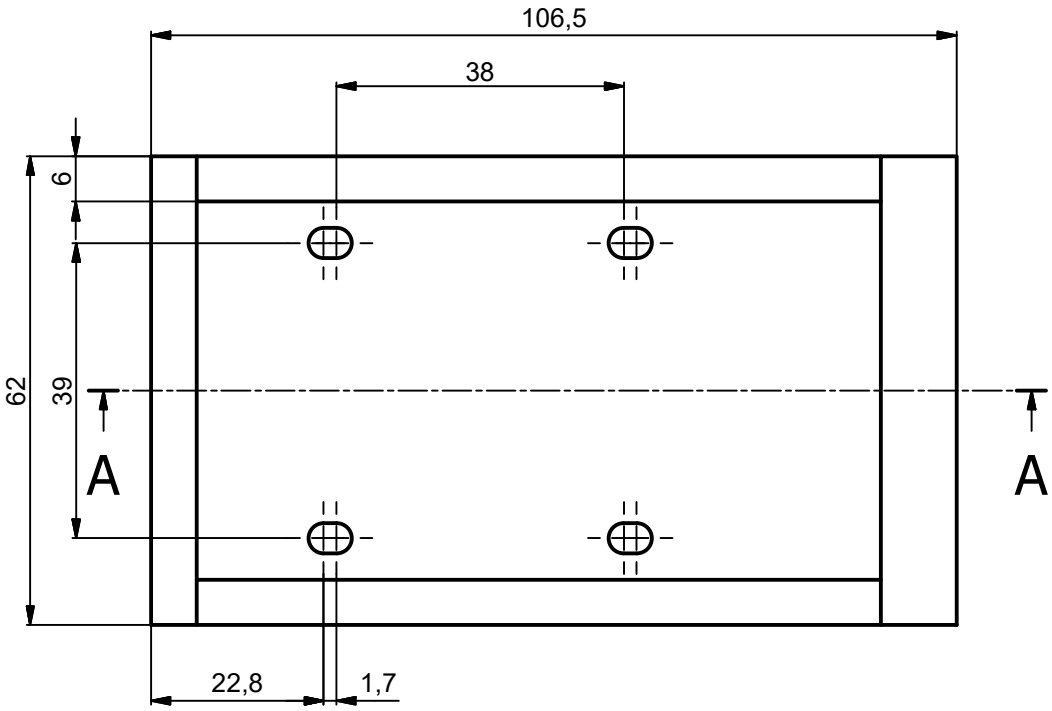
9.4.5. Plano del casquillo.

9.4.6. Esquema de la conexión del sensor con el Arduino.

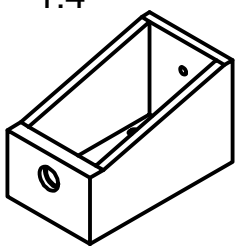
9.4.7. Esquema de la conexión del motor con el Arduino.

9.4.8. Esquema de la conexión del conjunto.





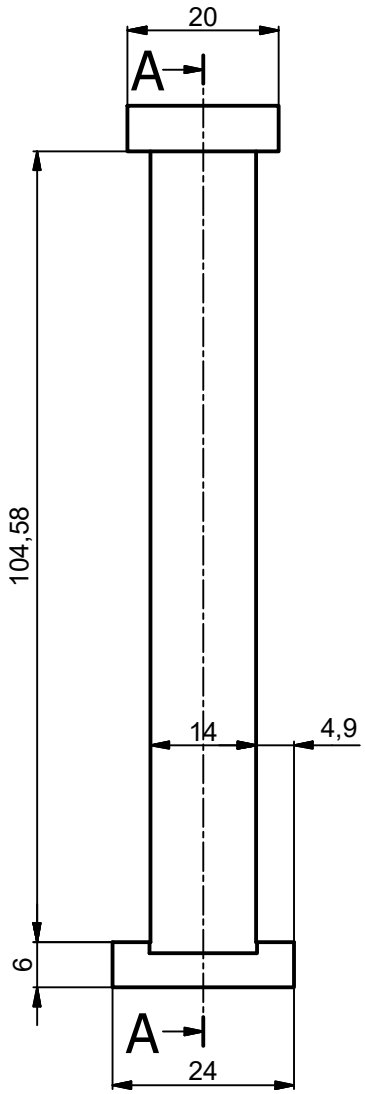
Escala
1:4

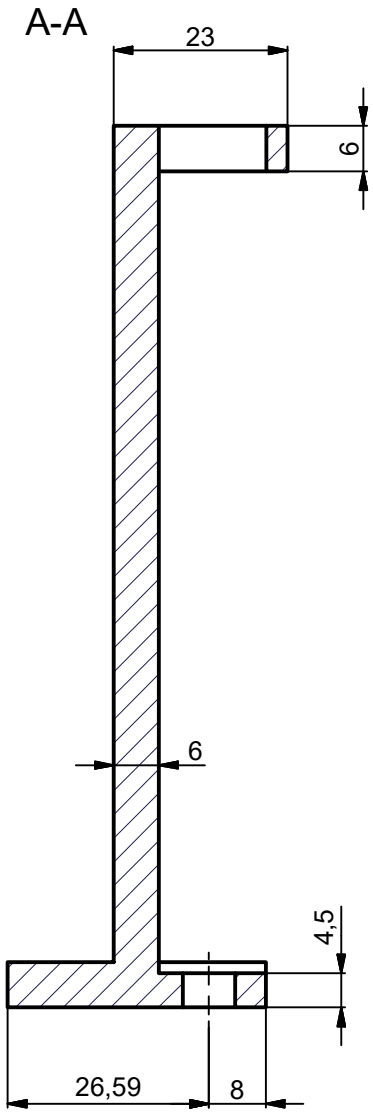


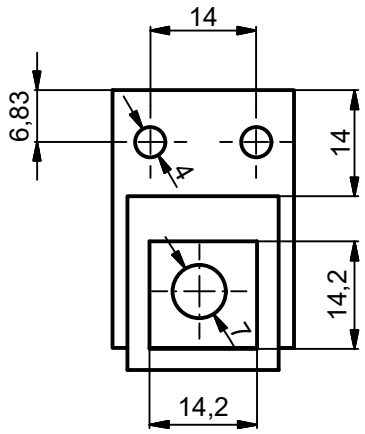
	FECHA	NOMBRE	FIRMAS
DIBUJADO	21/04/2017	Culpián	
COMPROBADO			
C. CALIDAD			

UNIVERSIDAD DE JAÉN
ESCUELA POLITÉCNICA SUPERIOR
LINARES

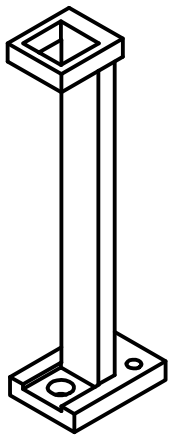
ESCALA	DESIGNACIÓN	Nº DE PLANO
	Plano Soporte del motor	1 / 1
		SUSTITUYE A
		SUSTITUIDO POR







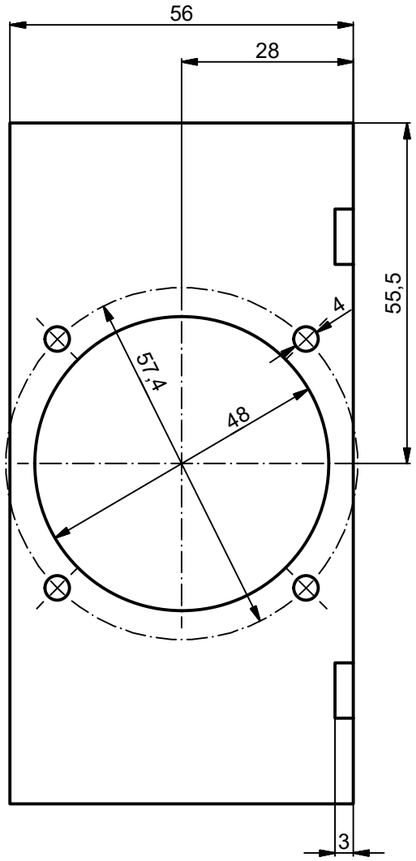
Escala
1:2

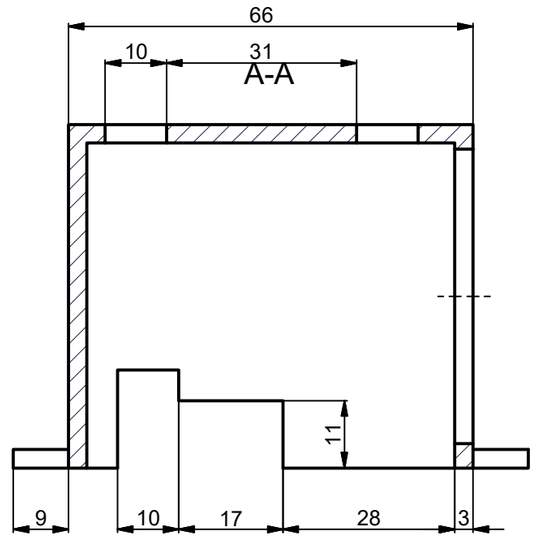


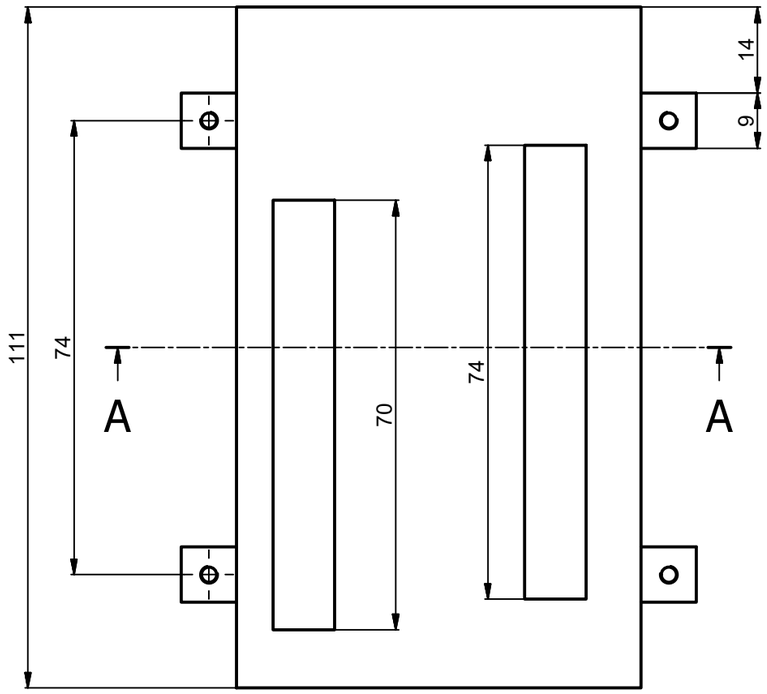
	FECHA	NOMBRE	FIRMAS
DIBUJADO	21/06/2018	Culpián	
COMPROBADO			
C. CALIDAD			

UNIVERSIDAD DE JAÉN
ESCUELA POLITÉCNICA SUPERIOR
LINARES

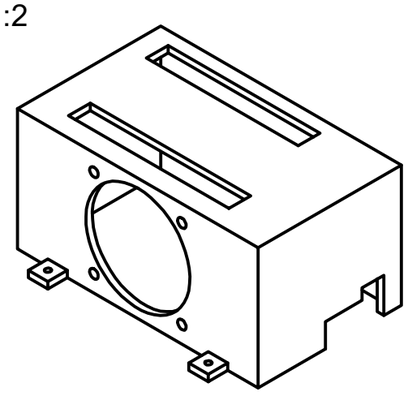
ESCALA	DESIGNACIÓN	Nº DE PLANO
1:1	Soporte Sensor	1 / 1
		SUSTITUYE A
		SUSTITUIDO POR







Escala
1:2

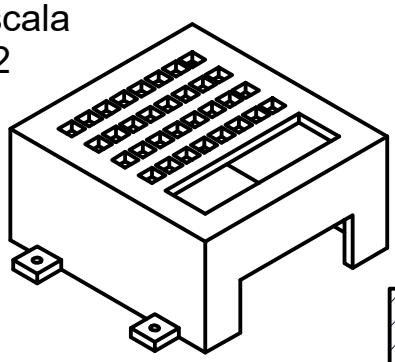


	FECHA	NOMBRE	FIRMAS
DIBUJADO	29/06/2018	Culpián	
COMPROBADO			
C. CALIDAD			

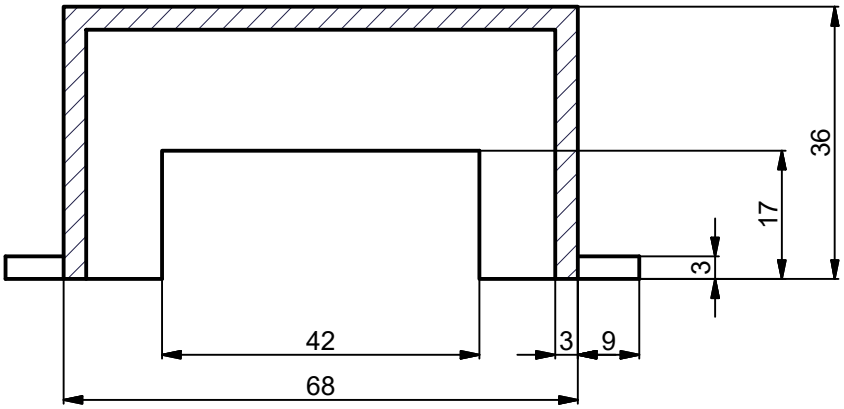
ESCUELA POLITÉCNICA SUPERIOR
LINARES
UNIVERSIDAD DE JAÉN

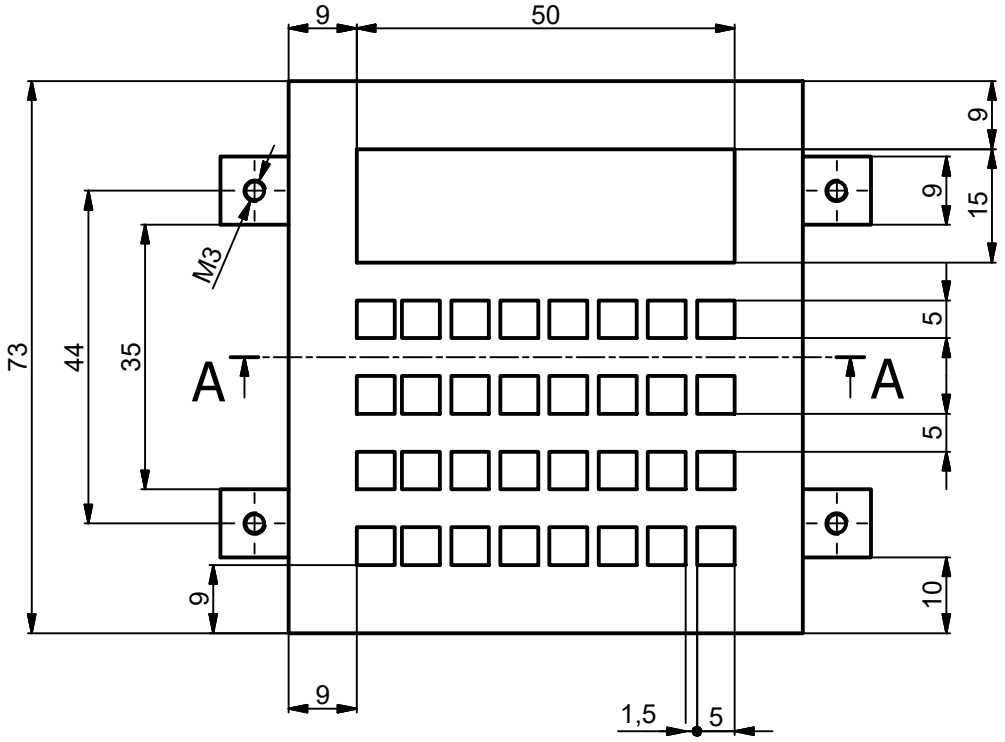
ESCALA	DESIGNACIÓN	Nº DE PLANO
1:1	Caja Arduino	1 / 1
		SUSTITUYE A
		SUSTITUIDO POR

Escala
1:2



A-A

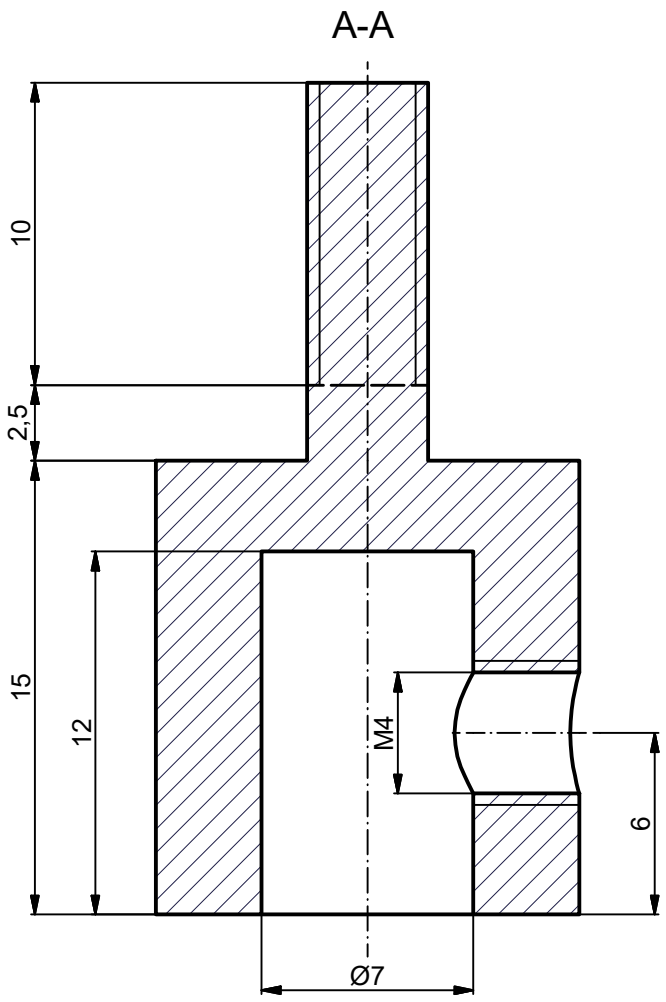


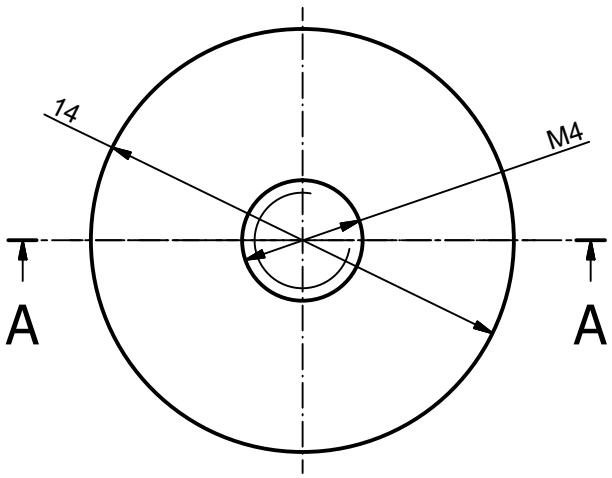


	FECHA	NOMBRE	FIRMAS
DIBUJADO	29/06/2018	Culpián	
COMPROBADO			
C. CALIDAD			

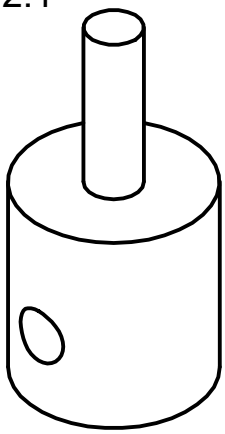
ESCUELA POLITÉCNICA SUPERIOR
LINARES
UNIVERSIDAD DE JAÉN

ESCALA	DESIGNACIÓN	Nº DE PLANO
1:1	Plano Caja del Regulador	1 / 1
		SUSTITUYE A
		SUSTITUIDO POR





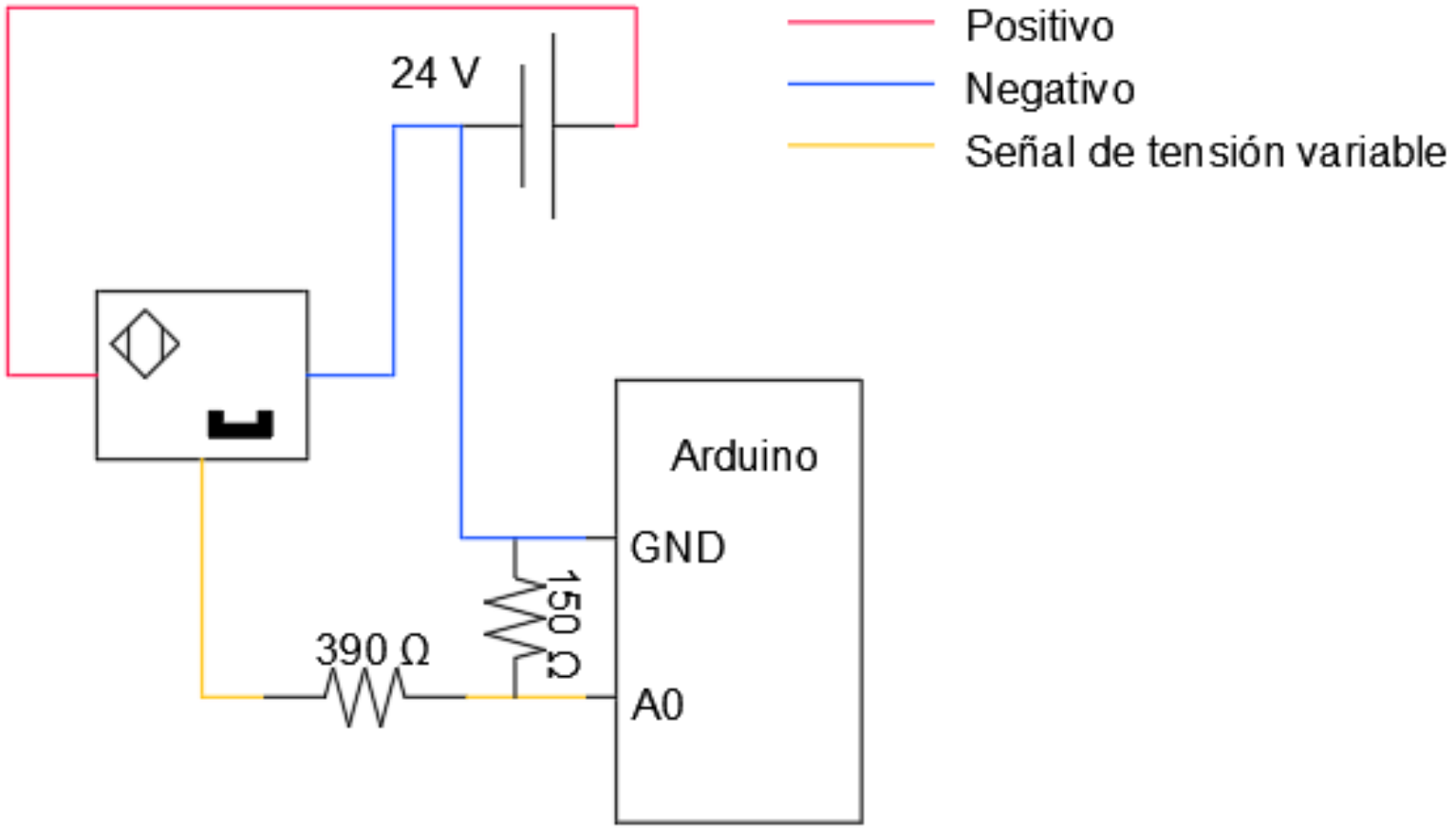
Escala
2:1

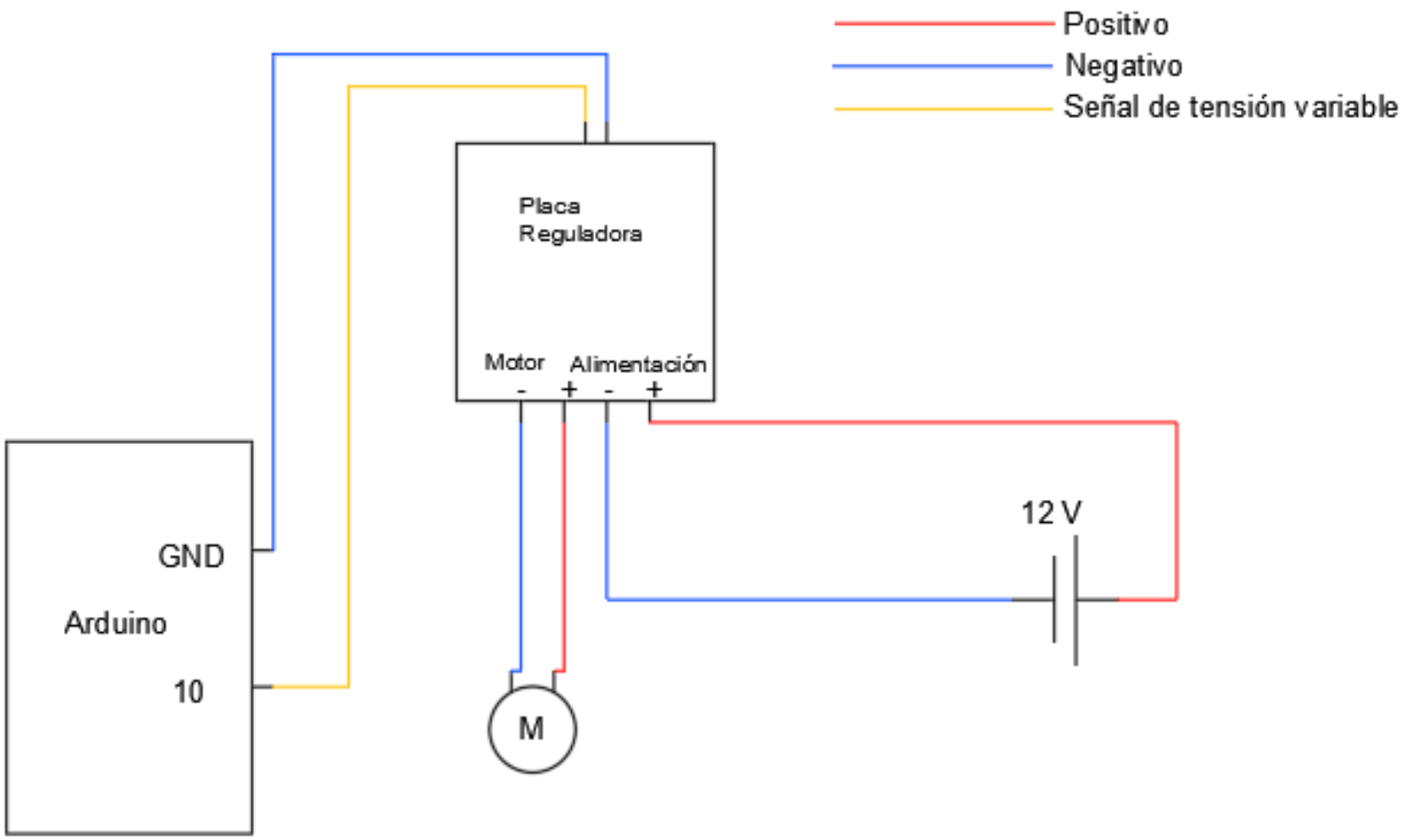


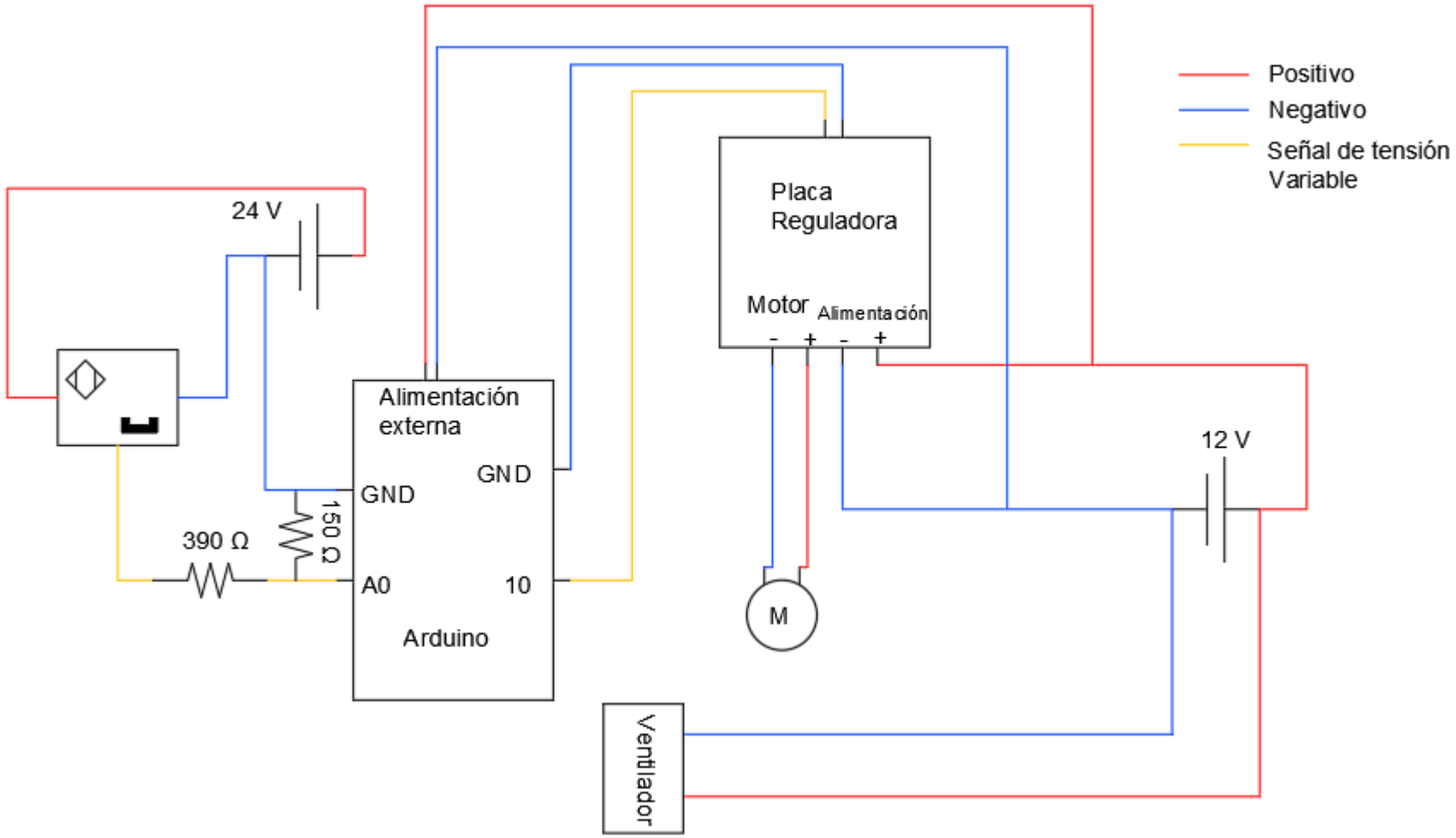
	FECHA	NOMBRE	FIRMAS
DIBUJADO	29/06/2018	Culpián	
COMPROBADO			
C. CALIDAD			

UNIVERSIDAD DE JAÉN
ESCUELA POLITÉCNICA SUPERIOR
LINARES

ESCALA	DESIGNACIÓN	Nº DE PLANO
4:1	Plano del Casquillo	1 / 1
		SUSTITUYE A
		SUSTITUIDO POR


Positivo
Negativo
Señal de tensión variable
24 V
Arduino
GND
A0
150 Ω
390 Ω


Positivo
Negativo
Señal de tensión variable
Placa
Reguladora
Motor
Alimentación
-
+
-
+
GND
Arduino
10
M
12 V


Positivo
Negativo
Señal de tensión Variable
24 V
Placa Reguladora
Motor
Alimentación
- + - +
Alimentación externa
GND
GND
150 Ω
390 Ω
A0
10
Arduino
12 V
M
Ventilador