Escola de Camins
Escola Tècnica Superior d'Enginyeria de Camins, Canals i Ports
UPC BARCELONATECH

Desarrollo de interfaces gráficas en tiempo real para ensayos de vigas de acero inoxidable

TREBALL FINAL DE GRAU

Treball realitzat per:

Karen Cecilie Fornés Christensen

Dirigit per:

Rolando Antonio Chacón Flores

Grau en:

Enginyeria d'Obres Públiques

Barcelona, Juny de 2018

Departament d'Enginyeria Civil i Ambiental

AGRADECIMIENTOS

En primer lugar quiero agradecer a mi tutor Rolando Chacón por su tiempo y paciencia, así como por darme la oportunidad de participar en este proyecto, que me ha aportado muchos nuevos conocimientos y del que estoy muy orgullosa.

Este trabajo ha sido desarrollado en el marco del Proyecto BIA2016-75678-R, AEI / FEDER, UE "Comportamiento estructural de pórticos de acero inoxidable. Seguridad frente a acciones accidentales de sismo y fuego ", financiado por el MINECO (España).

Gracias a Iván por creer en mí y siempre estar ahí para apoyarme y ayudarme en los momentos difíciles.

Gracias a mi familia y amigos, en especial a Adrià e Irene por su confianza y respaldo, recordándome siempre que puedo lograr lo que me proponga.

Gracias a Jordi, que aunque estando lejos nunca pierde la fe en mí, siempre está a mi lado y tiene las palabras indicadas de apoyo para cualquier momento difícil.

Barcelona, Junio 2018

DESARROLLO DE INTERFACES GRÁFICAS A TIEMPO REAL PARA ENSAYOS DE VIGAS DE ACERO INOXIDABLE

Palabras clave: Acero inoxidable, Digital Twin, Gemelo Digital, Arduino, Processing, Sensores.

RESUMEN

Autor:

Karen Fornés Christensen

Tutores:

Rolando Chacón
Itsaso Arrayago

Cada vez más aparecen nuevas ideas y conceptos tecnológicos relacionados con la conectividad de sistemas o personas. Conceptos como el internet de las cosas o Smart City están a la orden del día, con el objetivo de, a través de la tecnología, hacer la vida de las personas y sus interacciones más sencilla, cómoda y eficiente.

En 2002 aparece un nuevo concepto con un objetivo parecido: el Digital Twin o Gemelo Digital. Esta nueva idea consiste en la creación de un modelo digital de un objeto o proceso real, que equipado con sensores muestre y dé información a tiempo real de la producción, calidad o uso del producto de estudio con el fin de optimizarlo.

Con el objetivo de mostrar la aplicabilidad y ventajas de este concepto en el campo de la ingeniería civil, este trabajo pretende aplicar esta nueva idea a través de un desarrollo computacional propio de una interfaz gráfica a tiempo real aplicado, en concreto, a la tecnología estructural.

Para lograrlo se han realizado dos casos de estudio; en primer lugar, un modelo reducido aplicado al ensayo de una pletina de acero al carbono y, en segundo, un prototipo aplicado a un caso a gran escala donde el objeto de estudio será una viga de acero inoxidable, también con el objetivo de profundizar en las ventajas que supone el uso de este material en aplicaciones estructurales.

Los dos casos se desarrollaran mediante plataformas de código abierto y sensores de bajo coste, con tal de generar la visualización a tiempo real de los datos obtenidos en el ensayo, creando así el gemelo digital correspondiente.

Con tal de desarrollar dichas plataformas, en primer lugar, se presentará la situación actual del acero inoxidable como material estructural, así como sus numerosas aplicaciones y ventajas. Seguidamente, se expondrá la definición de la tecnología Digital Twin o Gemelo Digital y sus ventajas y aplicaciones a las estructuras, infraestructuras y a la construcción en general.

Entendidos los conceptos a tratar, se diseñará en primer lugar un sistema para el caso lineal correspondiente a la pletina de acero al carbono, determinando el sistema de adquisición de datos y la interfaz gráfica a mostrar. Este sistema se calibrará mediante un ensayo reducido en laboratorio. El objetivo final será crear un prototipo de interfaz gráfica para la aplicación a gran escala en una viga de acero inoxidable con la particularidad de la no linealidad que caracteriza el comportamiento de este material. El ensayo será llevado a cabo en el Laboratorio de Estructuras de *l'Escola de Camins, Canals i Ports de Barcelona* (UPC).

El fin último de este trabajo, pues, es mostrar la aplicabilidad del concepto Digital Twin en diferentes campos de la ingeniería civil, más concretamente al de la tecnología estructural a través de un diseño y desarrollo propio a modo de ejemplo a pequeña escala y la creación de un prototipo de gemelo digital a escala de una viga de acero inoxidable o bien, de pórticos de mayor entidad.

DEVELOPMENT OF REAL TIME GRAPHICAL INTERFACES FOR STAINLESS STEEL BEAM TESTS

Keywords: Stainless Steel, Digital Twin, Digital Twin in construction, Arduino, Processing, Sensors.

ABSTRACT

Author:

Karen Fornés Christensen

Tutor:

Rolando Chacón
Itsaso Arrayago

New ideas and technological concepts related to the connectivity of systems or people are appearing more and more. Concepts such as the Internet of things or Smart City are examples heard daily that want, through technology, make people's lives and their interactions more simple, comfortable and efficient.

In 2002 a new concept appeared with a similar objective: the Digital Twin. This new idea consists in the creation of a digital model of a real object or process equipped with sensors, to show and give information of the production, quality or use of the product in real time, in order to optimize it.

With the aim of showing the applicability and advantages of this concept in the field of civil engineering, the work wants to apply this new idea through a computational development of a real-time graphical interface, specifically applied to structural technology.

To achieve this, two case studies have been developed; a reduced model tested by a carbon steel plate, and a prototype with a real-scale case studying a stainless steel beam.

The two cases will be developed through open source platforms and low cost sensors, in order to generate the real-time visualization of the data obtained in the test, thus creating the corresponding digital twin.

In order to develop such platforms first, there will be defined the present situation of stainless steel as a structural material with its numerous applications and advantages. Then the definition of Digital Twin technology will also be presented in order to

understand the concept with the aim of analyze its applications to structures, infrastructures and construction in general.

Then, the two systems will be designed; choosing and designing the data acquisition system and the graphic interface to be displayed. The final purpose will be creating a prototype of a graphical interface for the large scale application in a stainless steel beam, with the particularity of the non-linear behavior that characterizes this material.

The tests will be completed in the Laboratory of Structures of the Civil Engineering School *Escola de Camins, Canals i Ports de Barcelona* (UPC).

The final goal of this work, then, is to show the applicability of the Digital Twin concept in different fields of civil engineering, more specifically structural technology application through an own design and development as an example.

ÍNDICE

ÍNDICE DE FIGURAS

CAPÍTULO 1: INTRODUCCIÓN

1.1. MOTIVACIÓN

Es una realidad que la tecnología avanza muy deprisa: en los últimos años se han visto avances que no se podían ni haber imaginado. La misión principal de estos avances reside en hacer la vida de las personas más cómoda, fácil y accesible.

De la misma forma, la obra civil es la forma material y más antigua de hacer la vida de las personas más conectada y segura, dotando al territorio de infraestructuras hechas para la comodidad de la sociedad.

El campo de la obra civil, sin embargo, es de las industrias que menos ha adoptado las nuevas tecnologías en sus procedimientos. Es curioso que dos ámbitos cuyos objetivos se asemejan tanto, sigan estando tan lejos.

Poco a poco, se empieza a querer unir las ciudades e infraestructuras con las nuevas tecnologías, con la misión de un mundo más conectado, efectivo y sostenible. Es el concepto, por ejemplo, de Smart City[1], que engloba iniciativas tecnológicas aplicadas al mundo real para la comodidad, entendimiento y funcionalidad de las ciudades a servicio de los ciudadanos.

Hace poco parecía una utopía pero cada vez es más una realidad, y es por eso que la motivación principal para realizar este proyecto, es la de adquirir una visión sobre cómo unir el mundo de la ingeniería civil con la tecnología e informática de la que, ya hoy en día disponemos.

Tener nociones en programación, o informática en general, así como estar al corriente de las nuevas aplicaciones que aparecen es algo indispensable para el futuro, y por esta razón se quiere, mediante un ejemplo real, comprobar que la unión del campo de la obra civil y la informática es algo factible, útil y no tan futurista como a veces se ve.

---

[1]El concepto Smart City se basa en tres objetivos: gestión eficiente de los recursos, fomento del desarrollo económico y calidad de vida de los ciudadanos. Para lograr estos objetivos, las Smart Cities apuestan por un uso transversal de las tecnologías de la información y la comunicación para cuidar el medio ambiente, potenciar la eficiencia energética, el urbanismo sostenible y para desarrollar una economía del conocimiento[1].

1.2. OBJETIVOS

Una de las motivaciones principales de este trabajo es la de ampliar los conocimientos en programación y combinarlos con los conocimientos en estructuras adquiridos durante los estudios, aprovechando también la oportunidad de participar en el análisis de un material estructuralmente poco convencional como es el acero inoxidable.

Para unir esos objetivos, se pretende entonces, analizar la aplicabilidad de la tecnología Digital Twin o Gemelo Digital en el campo de las estructuras y la construcción, intentando probar su factibilidad así como crear un estudio precedente para ver los campos de aplicación que este concepto puede llegar a abarcar en el campo de la obra civil.

Para conseguirlo, se desarrollará una interfaz gráfica, estudiando previamente el concepto de Digital Twin, y a través de, primero un ensayo reducido con una pletina de acero al carbono y después, la realización de un prototipo para un ensayo real de una viga de acero inoxidable se buscará determinar la viabilidad, las ventajas y las dificultades que supone la aplicación de esta tecnología.

1.3. METODOLOGIA

La metodología que se llevará a cabo, será en primer lugar un acercamiento a la electrónica y programación básica, con tal de entender y determinar un circuito electrónico y código que procese la información captada por el sensor utilizado.

En segundo lugar, se establecerá la configuración visual de la interfaz gráfica, para así estructurar el código que lo materializará y el formato visual que generará. La programación y los ensayos se harán en principio de forma paralela, pudiendo así calibrar y entender el comportamiento tanto del sistema como de la estructura a ensayar.

Se empezará con un ensayo a escala reducida de una pletina de acero al carbono convencional biapoyada, correspondiente a un comportamiento lineal del material. Las cargas serán pesos fijos conocidos y se calibrará el cálculo de la carga a partir de la flecha medida por el sensor, así como la correcta visualización de los datos.

Una vez consolidado el primer programa prueba, se ampliará dicho código para aplicarlo a un ensayo con una viga real de acero inoxidable –cuyo comportamiento es no lineal–.

Con esto, se pretende familiarizarse con el lenguaje y la programación en general utilizando un modelo lineal con una base de tecnología de estructuras y resistencia de materiales más sencilla. Una vez logrado eso, poder aplicar los conocimientos de programación adquiridos para enfocar los esfuerzos en el análisis del comportamiento estructural del acero inoxidable.

El objetivo final será crear, a través de un ejemplo real, un precedente de la aplicación de un modelo a tiempo real Digital Twin en el campo de la tecnología estructural, así como estudiar la viabilidad de este concepto en el campo de la obra civil en general.

El esquema que se seguirá, pues, será el siguiente:

- Acercamiento a la programación básica y a los dispositivos de bajo coste proporcionados.
- Determinación y programación de la interfaz gráfica del modelo reducido.
- Determinación y programación de la interfaz gráfica para el prototipo a gran escala.
- Ensayo del modelo reducido. Calibración del programa y resultados
- Calibración y comprobación de la interfaz gráfica para el prototipo a gran escala.
- Conclusiones

CAPÍTULO 2: ESTADO DEL CONOCIMIENTO

2.1. ACERO INOXIDABLE

- *Introducción*

El acero inoxidable es una aleación de hierro y carbono que contiene un mínimo de 10.5 % de cromo y un porcentaje de carbono menor al 1.2%, pudiendo también contener otros elementos aleantes dependiendo del tipo de acero inoxidable. Tal y como sugiere su nombre, es resistente a la corrosión, y eso se debe al buen comportamiento del cromo frente al oxígeno, creando una capa pasivadora que protege al hierro.

Los aceros inoxidables también se corroen, pero de forma distinta al convencional, formándose una capa de óxido de cromo en su superficie resultado de la combinación del oxígeno del ambiente con el cromo de la aleación, creando una película protectora. La elección del tipo de acero inoxidable dependerá de la clase de ambiente donde estará expuesto, pudiendo además aumentar dicha resistencia con presencia de níquel y molibdeno en la aleación[2].

El cromo presente en el acero inoxidable –donde además en muchas ocasiones también se incorpora níquel para aumentar su comportamiento inoxidable – se añade en estado de fusión para crear una masa homogénea y no necesitar tratamientos posteriores de ningún tipo para mejorar su resistencia a la oxidación.

El procedimiento de recocido brillante, adoptado por los fabricantes, consigue además que el cromo contenido no afluya a la superficie de la pieza para así retardar aún más la oxidación, consiguiendo a la vez un acabado de alto valor estético.

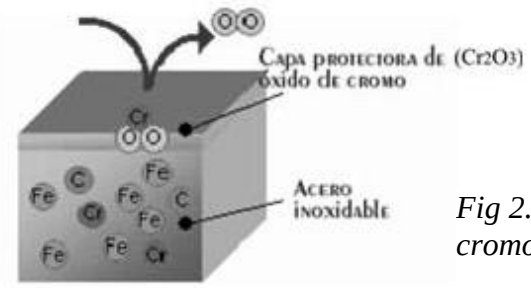


Fig 2.1.1 Capa pasivadora producida por el cromo en contacto con el oxígeno.[2]

Las características que hacen que el acero inoxidable sea interesante desde un punto de vista estructural, a parte de su elevada resistencia a la corrosión, son: su buena capacidad para absorber impactos sin rotura, debido a una muy buena ductilidad, y el endurecimiento por deformación que finalmente se traduce a un mejor comportamiento frente al pandeo.

- *Historia*

El primer descubrimiento de la aleación resistente a la corrosión formada por hierro y cromo, fue en 1821 por el metalúrgico Pierre Berthier, que observó la resistencia de dicha aleación a ciertos ácidos, proponiendo su uso para cuberterías. A causa del desconocimiento del aporte de fragilidad que suponen los altos contenidos de carbono y la incapacidad de la industria metalúrgica en crear aleaciones con bajo contenido de carbono, resultaba un material muy frágil y poco práctico.

En 1875, el científico francés Brustlein denotó la gran importancia del bajo contenido de carbono para la fabricación del acero inoxidable.

A partir de ciertas investigaciones a principios del 1900 hechas por Leon Guillet, que publicó un extenso estudio sobre la aleación hierro-cromo, se consiguieron las primeras composiciones clasificadas como acero inoxidable. Poco después, el descubrimiento del considerable aumento de la resistencia a la corrosión cuando la aleación contenía al menos un 10.5% de cromo, hizo que finalmente en 1912 dos ingenieros de la compañía Krupp patentaran el primer acero inoxidable austenítico [3]. Paralelamente el mismo año, un metalúrgico inglés que investigaba un material con mayor resistencia a la corrosión para cañones de armas de fuego, descubre el acero inoxidable martensítico.

Finalmente, su uso en la construcción empieza en los años 1930 para obras tan importantes como el Edificio Crysler o el Empire State Building de Nueva York, ampliándose más adelante para otras múltiples aplicaciones estructurales.

- *Tipos de acero inoxidable*

Hay diferentes tipos de acero inoxidable que se clasifican según su estructura metalúrgica: austeníticos, ferríticos, martensíticos, dúplex y los endurecidos por precipitación [4].

Los *austeníticos* son aquellos que contienen más de un 7% de níquel, con 17-18 % de cromo y menos del 0.1% de carbono. Representa el tipo de acero inoxidable más producido mundialmente. La pieza a analizar durante el presente trabajo es una viga de acero inoxidable de tipo austenítico.

Los *ferríticos* son aquellos que solamente contienen cromo. Su estructura está formada básicamente por ferrita y se distinguen por ser magnéticos. Tienen una gran resistencia a la corrosión bajo tensión y son mayormente utilizados para interiores.

Los *martensíticos* son aquellos que contienen hasta un 17 % de cromo, 0.1 % de carbono y sin presencia de níquel. También son magnéticos y pueden endurecer por tratamientos térmicos.

Los tipo *dúplex* tienen muy buen comportamiento mecánico, inoxidables bajo tensiones y resistentes al desgaste, conteniendo un alto porcentaje de cromo (22-23%).

Los *endurecidos por precipitación* son aceros inoxidables austeníticos endurecidos por deformación.

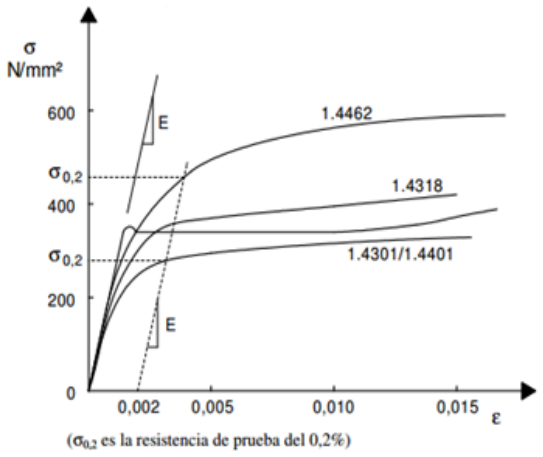
Generalmente, los aceros inoxidables tipo *austeníticos* y *dúplex* son los más utilizados en aplicaciones estructurales por su buena combinación de resistencia a la oxidación y al desgaste.

- *Propiedades mecánicas. Comportamiento tensión-deformación*

Una de las principales diferencias del acero inoxidable y el acero al carbono es su comportamiento tensión-deformación.

El acero al carbono presenta un comportamiento claramente lineal hasta su límite elástico y una zona de fluencia horizontal, donde a tensión constante sigue deformándose para finalmente endurecer por deformación y romper al llegar a su límite último.

El acero inoxidable, en cambio, no presenta un límite elástico definido siendo su curva tensión-deformación continua y redondeada. Al no tener un límite elástico definido, se suele estipular en base a una resistencia prueba a un 0.2% de la deformación plástica.[5]



Acero inoxidable

Acero al carbono (grado S355)

Fig 2.1.2 Curvas tensión-deformación para acero inoxidable y acero al carbono [2]

- *Acero inoxidable estructural*

Gracias a sus numerosas ventajas como material, se ha incrementado su investigación y desarrollo en la producción, sobre todo durante los últimos 20 años, hecho que ha permitido una gran ampliación en su campo de aplicación en la construcción.

Gracias al interés estructural de este material, se han hecho grandes avances en su manufacturación, bajando los costes de su producción y también haciendo énfasis en el desarrollo de sus acabados superficiales y uniones.

Además de esta reducción de costes de producción, también se han hecho notables mejoras en la soldadura de piezas, que antes era más costosa. Actualmente, gracias a equipos compactos y potentes de láser se consiguen muy buenas uniones hasta en perfiles gruesos y, con una distorsión mínima de los componentes. Esta cadena de mejoras ha hecho que se haya incluido este material en especificaciones, códigos y normativas internacionales para regular su uso y características en la construcción aunque, aún en algunos casos, las normativas aún se remiten a especificaciones del acero al carbono para la determinación de algunos parámetros[6].

A parte de su buen comportamiento frente la corrosión ya comentado, al igual que su tenacidad, también permite perfiles de menor grosor y más durables permitiendo diseños con formas y colores diversos resultando estructuras estéticas, adaptables y reciclables.

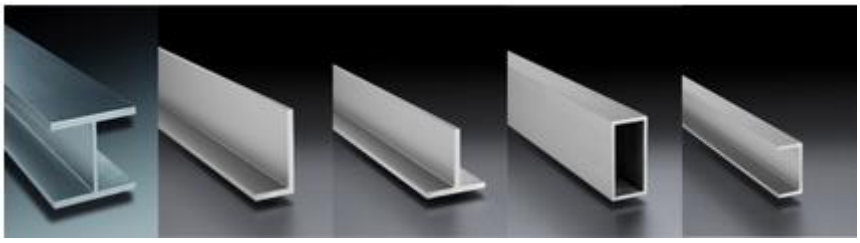


Fig 2.1. 3 Perfiles estructurales de acero inoxidable [6]

Otro aspecto remarcable como material estructural, es su comportamiento frente al fuego, ante el que también presenta mejores características que el acero al carbono, manteniendo su capacidad resistente y rigidez a temperaturas superiores a 500 ºC.

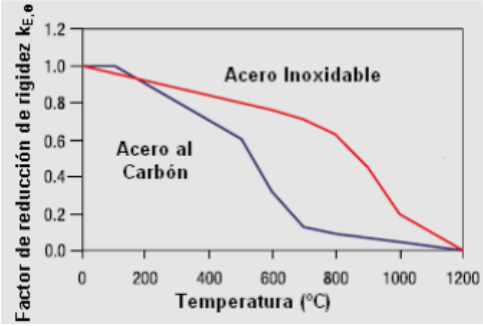


Fig 2.1.4 Reducción de rigidez con la temperatura, comparación entre acero inoxidable y al carbono. [7]

Su mejor comportamiento frente a la sismicidad también es destacable, dado su incremento de resistencia con la tensión, proporcionando un factor de seguridad adicional [7].

Por los numerosos motivos expuestos, en los últimos años ha habido una notable expansión del uso del acero inoxidable en la aplicaciones estructurales, transfiriendo las tecnologías desarrolladas de otras industrias como por ejemplo la aeroespacial, al de la construcción.

Es por eso que recientemente el consumo de acero inoxidable en la construcción ha incrementado representando hasta un 14% del consumo total de este material. Países como China han aumentado su producción notablemente en los últimos 20 años y tienen un destacable uso del acero inoxidable en este sector.

Con tal de dar a conocer los avances obtenidos, diferentes asociaciones han construido edificios prueba, con tal de ampliar y demostrar las posibles aplicaciones y su buen comportamiento. Un ejemplo es la construcción del laboratorio de investigación de acero inoxidable Nisshin Steel en Osaka con vigas de acero inoxidable, que además quedó intacto tras el sismo en Kobe en 1995. Otro ejemplo es el uso de varillas de acero inoxidable de alta resistencia en el muro cortina de vidrio de la pirámide del museo del Louvre, que creó un precedente para otros proyectos de muros cortina y fachadas de vidrio con el mínimo impacto visual.

Su uso como barras de armar también ha sido probado, en el muelle el Progreso de México [8], donde en los años 40 se armó el hormigón con 200 toneladas de barras de acero inoxidable y que hoy en día aún no han presentado deterioro. Las barras embebidas en el hormigón extienden la vida útil de la estructura en ambientes corrosivos, bajando la demanda de inspecciones y mantenimiento.

El puente Millenium en York (Reino Unido) también es un ejemplo de aplicación en la obra civil, dado que se incorporó acero inoxidable en los elementos estructurales principales, así como también se hizo en el puente Tsing Ma (Hong Kong) para controlar el flujo de aire a través de la plataforma dado que, en este caso, el condicionante crítico eran las cargas horizontales de viento.

En definitiva, es un material que está demostrando su potencial en el uso estructural, aunque su coste inicial sea más elevado, pero que por su comportamiento mecánico, alto valor estético, durabilidad, ligereza y baja necesidad de mantenimiento, crea un coste a largo plazo muy competitivo que lo está haciendo cada vez más presente en todo tipo de estructuras.

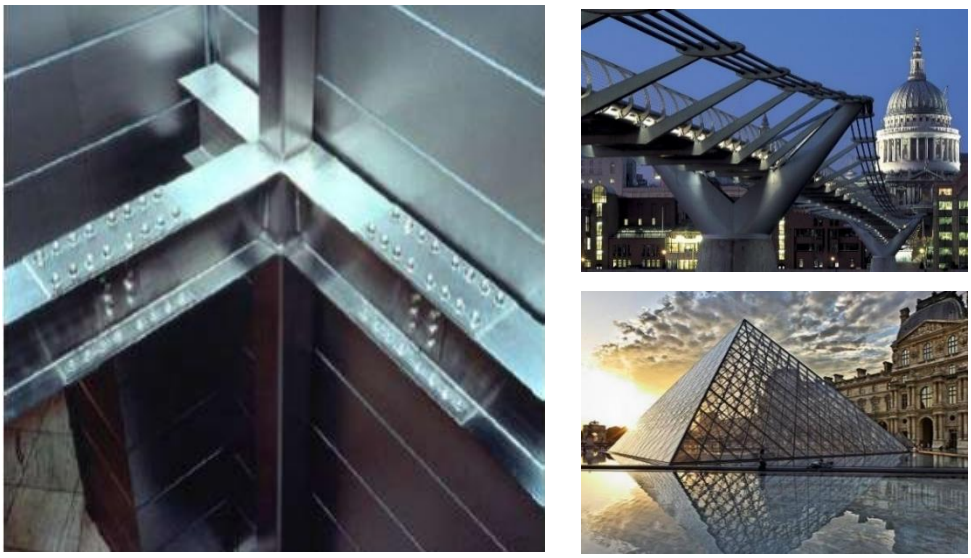


Fig 2.1. 5 Ejemplo de aplicaciones del acero inoxidable.[7]
Detalle de vigas de acero inoxidable en el laboratorio Nisshin Steel (Osaka). (Izquierda)
Millenium Bridge (York). (Arriba)
Muro cortina del museo Louvre (Paris). (Abajo)

- *Normativa aplicable*

La primera normativa referente al uso estructural del acero inoxidable fue publicada por el American Iron and Steel Institute en 1968.

Actualmente, la normativa aplicable en España y Europa es el Eurocódigo 3, "Proyectos de estructuras de acero" Parte 1-4. Reglas generales – Reglas adicionales para los Aceros Inoxidables"[9].

Esta normativa ya tiene en cuenta las diferencias de comportamiento físicas y mecánicas que existen entre el acero al carbono y el acero inoxidable. Así como hace relativamente poco la normativa remitía el diseño de estructuras de acero inoxidable a normativas referentes al acero al carbono, el Eurocódigo 3 ya cuenta con la inclusión de los parámetros propios que diferencian al acero inoxidable, como es el límite elástico y la resistencia última, el módulo elástico, curvas tensión-deformación, cálculo de la deformabilidad –teniendo en cuenta la no linealidad del material– así como los coeficientes de seguridad aplicables, entre otros.

Existe también el "Manual de Diseño de Acero Inoxidable Estructural" [10] preparada por el Steel Construction Institute que cuenta con tres ediciones; la primera publicada en 1994 por Euro Inox, que incluía recomendaciones para la elección del tipo de acero inoxidable, informaciones sobre propiedades, entre otras. La segunda versión fue publicada en 2002 e incluía sobre todo, información referente a secciones huecas circulares y resistencia frente al fuego. Finalmente, en 2006 y financiado por la RFCS (*Research Fund for Coal and Steel*) se publicó la tercera edición, que extiende el campo de aplicación de aceros inoxidables austeníticos trabajados en frio, actualizando las referencias de los Eurocódigos e incluyendo una extensión de los proyectos de estructuras frente a incendio, nuevas secciones de durabilidad, costes de ciclo de vida y nuevos ejemplos de dimensionamiento. Esta tercera revisión fue realizada con la colaboración de la Universitat Politécnica de Catalunya (UPC), The Swedish Institute of Steel Construction (SBI) y Technical Research Centre of Finland (VTT)

El Manual de Diseño es una guía –no una normativa– cuyo objetivo es guiar a técnicos en la labor de diseño y dimensionamiento de elementos estructurales de acero inoxidable, componentes secundarios de edificios, instalaciones offshore y similares. No incluye la aplicación en estructuras especiales que tengan normativa específica, como centrales nucleares o depósitos. El manual está dividido en dos partes; recomendaciones y ejemplos de dimensionamiento Dichas recomendaciones hacen referencia al comportamiento del material, al dimensionamiento de elementos conformados en frío, elementos soldados y a sus uniones, siendo su aplicación a los tipos austenítico y dúplex que son los utilizados habitualmente para aplicaciones estructurales.

2.2. DIGITAL TWINS

- *¿Qué es un Digital Twin?*

El concepto de Digital Twin o Gemelo Digital, es un concepto relativamente reciente en el ámbito de la tecnología y se define como un modelo a tiempo real de un objeto físico o un proceso con el fin de optimizarlo.

Hasta hace relativamente poco, aun pudiendo imaginar el concepto, era difícil llevarlo a cabo dadas las limitaciones de la tecnología respecto, sobre todo, a la gran demanda de espacio que requería el almacenamiento de los datos obtenidos. Sin embargo, estos obstáculos se han reducido mucho en los últimos tiempos, gracias al menor coste de los procesos y al gran avance tecnológico en este ámbito, que ha permitido la combinación de la tecnología de la información (Information Technology IT) junto con la tecnología de las operaciones (Operation Technology OT) para crear y utilizar el Digital Twin.

La concepción y ejecución de esta idea permite una visión completa y global de un producto o proceso desde el diseño y desarrollo, pasando por su vida útil y hasta el final de su ciclo de vida.

El Digital Twin permite analizar y entender, no solo el producto o proceso analizado en sí, sino todo el sistema que lo produce y elabora y cómo este producto es utilizado. De esta forma, se pueden plantear cuestiones como: cómo mejorar el producto o proceso, qué defectos tiene o cómo prevenirlos, todo con el objetivo de crear nuevos y mejorados modelos de producción con un óptimo producto final.

Existen varias definiciones del concepto Digital Twin: una es el de un modelo digital que a través de sensores representa y simula un objeto físico a tiempo real, pero fundamentalmente, se puede concluir que es **un perfil digital del comportamiento, tanto histórico como actual, de un producto o proceso para su control y optimización**[11].

El Digital Twin está basado, entonces, en la **recepción**, **acumulación**, y **procesado** masivo de medidas de datos reales. Esto permite gracias a los avances en el campo de la tecnología y el internet de las cosas[2] (IoT por sus siglas en inglés) simular también, la conexión entre diferentes productos o procedimientos complejos para predecir toda su vida útil, así como sus interacciones.

Así pues, los elementos que hacen posible la integración del mundo físico y digital para materializar el concepto de Digital Twin son los sensores, que captan los estímulos del entorno real creando datos que serán procesados a través de simulaciones algorítmicas que, a su vez, acabarán representando el objeto físico.

Los primeros ejemplos de aplicabilidad de esta nueva tecnología, surgen en rotores eólicos y motores de aviones para monitorizar y evaluar posibles inefectividades o comportamientos frente a determinados estados tensionales, entre otros.

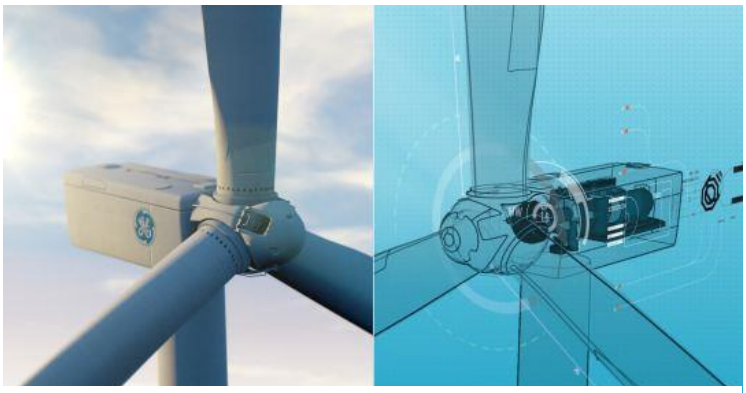


Fig. 2.2.1 Ejemplo de Digital Twin aplicado al rotor de una torre eólica.

[2]El internet de las cosas es un concepto que se refiere a la conexión digital de objetos cotidianos con internet, codificándolos para saber en todo momento su ubicación, consumo o disponibilidad. Es por lo tanto, una conexión avanzada de dispositivos, sistemas y servicios, formando así, la base de un Digital Twin, que pretende crear el modelo digital resultante de dicha información.

- *Origen*

El concepto de Digital Twin – llamado en su inicio Information Mirroring Model- fue por primera vez definido en 2002 en un curso de gestión del producto de la Universidad de Michigan [12], con el objetivo de presentar una idea de la óptima gestión del ciclo de vida de un producto -Product Lifecycle Managment- e incluía todos los elementos que definen un Digital Twin: el espacio real y el virtual y el enlace del flujo de datos e información entre ambos

La idea era la creación de un modelo que consistiera en la presencia de dos sistemas: el físico y el virtual, donde este último contendría toda la información relativa al primero; es decir, un espejo o gemelo bidireccional entre los dos sistemas.

La idea quería ir más allá de la representación estática y vincular a lo largo de todo el ciclo de vida del producto los dos sistemas, conectándose a medida que el producto pasara por sus cuatro fases; concepción, fabricación, utilización y eliminación.

El concepto de Digital Twin fue adoptado muy rápidamente por el campo aeronáutico y aeroespacial. La precedencia de esta tecnología fue utilizada a la práctica por primera vez por la NASA en la misión del Apolo 13 – aun siendo ésta muy anterior a la aparición del concepto de Digital Twin– consiguiendo que los ingenieros en tierra fueran capaces de determinar qué estaba fallando en la nave y solucionarlo de forma remota.

Hoy en día es ampliamente usado por esta organización para reparar, actualizar y monitorizar máquinas en el espacio, evitando la necesidad de estar físicamente presente. También se ha implementado esta tecnología para varias propuestas, entre otras, la exploración espacial sostenible o la construcción de vehículos espaciales.

Más recientemente y ya bajo el concepto de Digital Twin, la fábrica Siemens Amberg ha sido de las empresas pioneras en integrar el gemelo digital en sus sistemas de producción [13]. Esta empresa produce sistemas industriales de control de ordenadores e hicieron una réplica virtual exacta de la fábrica en Amberg (Alemania), esto se tradujo a un aumento de la producción a 15 millones de unidades al año sin expandir el edificio ni contratar más personal. Además del destacable aumento en la producción, se recortó la tasa de defectos de fábrica a casi cero, suponiendo los productos sin necesidad de ajuste un 99.9988 % del total.

De la misma forma, la compañía GE ha implementado ya 500.000 Digital Twins usados para informar de la configuración de cada aerogenerador antes de su construcción, teniendo un parque eólico digital que les ha permitido aumentar un 20% la eficiencia a través del análisis de los datos de cada turbina que alimenta a su equivalente virtual.

- *Funcionamiento*

De forma general, el funcionamiento de un Digital Twin consiste en la recogida de datos a tiempo real, a través de todo tipo de sensores, que almacenados y tratados adecuadamente permiten crear el modelo virtual correspondiente.

Según el objetivo que se pretenda creando dicho modelo y según la información que contiene se pueden encontrar dos tipos de Gemelo Digital [12];

Digital Twin Prototype (DTP): Describe un prototipo físico mediante un modelo virtual y contiene los conjuntos necesarios para describir y producir la versión física a partir del prototipo virtual. Incluye los requisitos para su fabricación, un modelo tridimensional completo, una lista de materiales con sus especificaciones correspondientes, una lista de procesos, servicios y un protocolo de eliminación.

Digital Twin Instance (DTI): Describe un producto físico específico y tiene vinculado a lo largo de su vida útil a su Digital Twin correspondiente. Dependiendo del caso, tiene diferentes conjuntos de información, pero contiene al menos los siguientes: un modelo 3D con la totalidad de su dimensionamiento general describiendo geometría, componentes y tolerancias; una lista de los materiales, especificando los componentes actuales e históricos y una lista de procesos que incluye las operaciones que se han realizado en el producto físico junto con los resultados, mediciones, pruebas, componentes reemplazados, así como los datos reales de los sensores vinculados actuales, pasados y la predicción o pronóstico futuro resultante de todos los datos.

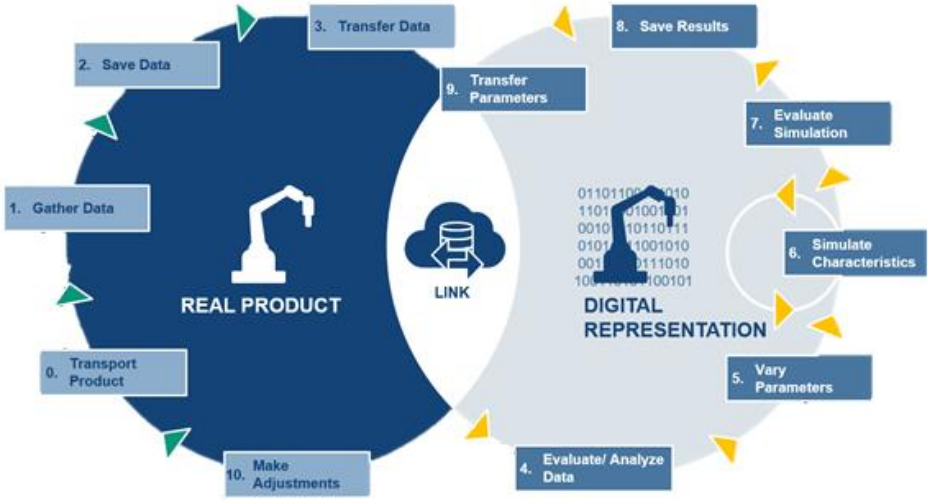


Fig. 2.2.2. Ciclo de funcionamiento de un Digital Twin

De esta forma, y agrupando DTI's con una estructura informática completa que tiene acceso a todos ellos y es capaz de consultar, examinar y correlacionar todos los datos de forma continua, se pueden generar los pronósticos y estadísticas específicas que se requieran, como podría ser cuánto tarda un componente a fallar de media, o cuándo será necesario reemplazar un determinado elemento.

2.3. DIGITAL TWINS APLICADO A LAS ESTRUCTURAS, INFRAESTRUCTURAS Y LA CONTRUCCIÓN

- *Introducción*

Vistas las posibilidades que ofrece la aplicación del Digital Twin en general y sobre todo, en varios tipos de industria distintos al de la construcción, se pretende mostrar en este apartado las posibilidades que esta tecnología puede ofrecer en un campo tan importante y relativamente poco digitalizado como es el de la obra civil.

En primer lugar, la aplicabilidad de esta tecnología y concepto a la construcción, supone el tener acceso siempre a la obra tal y como se ha diseñado y construido –As-designed y As-built–, permitiendo monitorizar, organizar, seguir y corregir el plan de obra hecho, actualizándolo continuamente [14]. Esto significa que se puede basar la toma de decisiones en un modelo predictivo, haciendo uso de simulaciones para analizar diferentes escenarios estimando sus probabilidades de suceso y éxito con los costes asociados, con tal de escoger la más óptima y factible.

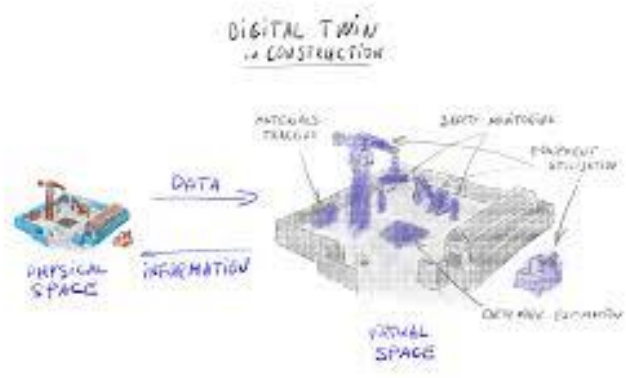


Fig. 2.3.1 Idea de Digital Twin en la construcción [14]

- *Obtención y procesado de los datos*

Hasta el momento, se ha expuesto de forma general la presencia de sensores para la obtención de los datos necesarios para la creación de un Digital Twin. Estos sensores pueden parecer dispositivos muy sofisticados, pero en realidad pueden ser elementos tan cotidianos como cámaras digitales o de smartphones que, mediante la detección de objetos específicos, den información del avance de la construcción u operación que se está realizando. El time-lapse también es un recurso útil con tal de crear un modelo digital a partir del real.

En los últimos años, los drones también se están incluyendo en el mundo de la construcción tanto para la planificación, promoción y comercialización de nuevos proyectos y obras como para, ya durante su ejecución, el seguimiento, inspección, levantamiento topográfico, y creaciones de As-built en 3D.

Así pues, la recogida de datos puede provenir de un sinfín de dispositivos; desde los que nos rodean, como cámaras o sensores ya utilizados como sensores de temperatura o movimiento, hasta algunos de más sofisticados como los drones, permitiendo combinarlos todos creando una base de datos provenientes de diferentes índoles, completa y con el grado de complejidad que se desee.

Relativo al procesado de datos, hace pocos años era impensable que la tecnología y la informática estuvieran en el punto en el que están actualmente; entonces, conseguir un procesado óptimo de los datos suponía todo un reto.

La realidad es que los recientes avances en la potencia de los ordenadores, el procesamiento GPU –Graphics Processing Unit–, la capacidad de almacenaje y los algoritmos cada vez más sofisticados ya han hecho posible un procesamiento automático de los datos.

- *Aplicaciones en la construcción*

A continuación se exponen algunas situaciones en las que la tecnología Digital Twin puede aplicarse a la construcción, con el objetivo de mostrar la factibilidad y avance que esta supone.

Supervisión automática:

El intercambio a tiempo real de los datos ayuda a verificar que el trabajo ejecutado es coherente con el plan y las especificaciones correspondientes, aun siendo necesaria la observación física a pie de obra para verificar el trabajo realizado y determinar la etapa actual del proyecto, permite una corrección inmediata de cualquier desviación respecto a la planificación.

Cabe destacar que la efectividad o facilidad para controlar dicha comparación plan-realidad se ve aumentada con la existencia de un modelo BIM previo que permite automatizar, casi totalmente, la comparación entre lo ejecutado y lo proyectado sin la problemática de una comparación entre un modelo digital a tiempo real y un proyecto en papel.

Gestión de recursos y logística

Según varios estudios, sobre un 25% del tiempo de producción se malgasta en movimientos y deficiente gestión de recursos, materiales y residuos [14]. Con un Digital Twin se permitiría un enfoque organizativo predictivo para la gestión de transporte, materiales y residuos para reducir los tiempos que supone dicha logística.

Por ejemplo, controlando los tiempos de llegada de material, relacionándolo directamente con el planeamiento temporal y técnico de la obra, así como el control de acopio, gasto de material, y organización general de los flujos de transporte, dentro y fuera de la obra.

Seguridad y salud en la construcción

La industria de la construcción, como bien se sabe, es uno de los sectores con más accidentalidad, aunque en los últimos años se está reduciendo gracias a leyes, normativas y controles más exhaustivos en materia de prevención y riesgos laborales.

La tecnología de la monitorización y predicción a tiempo real, puede ser una gran herramienta para detectar malas praxis, tajos o puntos peligrosos de la obra analizada

La implementación de esto – ya utilizado en algunas industrias como la química– puede ser tan sencilla como, a través de cámaras ya instaladas en las obras y los mismos teléfonos móviles de los operarios y técnicos, detectar herramientas, o situaciones de riesgo – por ejemplo, un vertido de residuo peligroso– para avisar a los operarios y solventarlo de forma inmediata, eficiente y con el mínimo riesgo.

Fig. 2.3.2. *Ejemplo de aviso de vertido peligroso* vía detección de cámaras y aviso a teléfono móvil [14]

Otro ejemplo podría ser el control de la correcta utilización de maquinaria o herramientas –asegurando por ejemplo la utilización de EPI's por parte de los operarios– estando todas las personas implicadas, conectadas y avisadas de los riesgos a tiempo real.

Control de calidad

Hoy en día, ya se usan sensores para el control de asientos durante construcciones en terreno inestable, o para controlar los posibles asientos diferenciales en propiedades o estructuras colindantes a la obra. Yendo más lejos, el uso de algoritmos de procesamiento de imágenes permite verificar, por ejemplo, las condiciones del hormigón a través de imágenes fotográficas o de vídeo, verificando la presencia de fisuras o la correcta colocación y disposición de armaduras, permitiendo así, la reducción de inspecciones adicionales y posibles problemas de calidad.

Las investigaciones y progresos, por ejemplo, en el campo de los coches autónomos han hecho del reto del procesado de la gran cantidad de datos adquiridos algo realista. Ya es posible construir modelos 3D a tiempo real a base de fotografías o nubes de puntos –como los procedentes de los dispositivos LIDAR– o la detección de objetos a partir de vídeos, definiendo su forma, posición y velocidad.

- *Iniciativas Europeas. Carreteras conectadas e inteligentes.*

Viendo el potencial que supone un concepto como el de Digital Twin o el internet de las cosas, ya se están tomando iniciativas a nivel europeo para integrar estas ideas a las infraestructuras con tal de crear sistemas inteligentes y conectados.

Un ejemplo de ello son las C-ITS (Cooperative Intelligent Transport System) un proyecto de sistemas inteligentes de transporte cooperativos impulsados por la UE.

Este proyecto pretende conectar digitalmente los vehículos entre ellos y con las infraestructuras del transporte, con el objetivo de mejorar la seguridad vial, la eficiencia de la gestión del tráfico y el confort de la conducción, ayudando al conductor a tomar las decisiones adecuadas y adaptarse a la situación del tráfico a tiempo real.

Esta iniciativa se aprobó a finales de 2016 por la Comisión Europea [15] con tal de adoptar una estrategia para conseguir la cooperatividad, conectividad y automatismo en la movilidad. Para conseguirlo, ya se están llevando a cabo pruebas piloto con tal de facilitar la inversión y creación de un marco legal a nivel europeo.

La misión de los C-Roads –el proyecto principal de C-ITS– es crear una conectividad entre los vehículos y las infraestructuras del transporte a tiempo real y a través de los teléfonos móviles de los usuarios, que servirán tanto de emisor de información como de receptor, avisando a los conductores de eventuales accidentes en la carretera, estado del tráfico, meteorología, entre otras situaciones.

El pasado noviembre del 2017, España se incorporó en este proyecto europeo llamado C-Roads Spain, donde se implementaran 5 pruebas piloto por el territorio español; el piloto de la M-30 (Madrid), Cantábrico, Siscoga extendido (Galicia), piloto Mediterráneo y DGT 3.0, que se llevarán a cabo bajo la coordinación de la Dirección General de Tráfico (Ministerio del Interior) y Carreteras (Ministerio de Fomento) [16].

Lo desarrollará un consorcio con socios pertenecientes al sector público (diferentes autoridades y universidades) y privados (operadores de carreteras, proveedores de servicios y tecnología y otros asociados al sector). Actualmente están en marcha los trabajos de despliegue de estos pilotos, que se desarrollarán por fases entre el 2018 y el 2019, con el 2020 como año fijado para la finalización del proyecto.

El objetivo es ensayar un sistema formado por tecnologías desplegadas en la carretera (sensores y antenas RSU -Road Side Units-) y dispositivos instalados en los vehículos (On Board Units) para la recogida de datos y comunicaciones que serán recibidos, seleccionados y procesados para su correcta y segura difusión a los usuarios en un centro de gestión conectado a la nube.

Aunque esta iniciativa no trabaja bajo el nombre de Digital Twin, sí que comparte muchas características comunes, como la recogida de datos a tiempo real para su modelaje interno con tal de gestionar las infraestructuras y su comunicación exterior a los usuarios para su correcta utilización y optimización. Muestra además, que no es una iniciativa futurista sino real y puesta ya en marcha, demostrando así su total aplicabilidad.

- *El puerto de Barcelona. Digital Port pionero.*

El puerto de Barcelona también es otro ejemplo de la aplicación real del Digital Twin en grandes infraestructuras. Este puerto ha aplicado diferentes tecnologías en su gestión creando el primer Digital Port nacional, generando servicios eficientes en el ámbito comercial, social y logístico [17]. La aplicación de dicho concepto reside en el uso de las tecnologías para transformar los diferentes servicios en servicios interactivos; compromiso medioambiental y adaptación de los servicios portuarios a las necesidades de los clientes.

El recinto del puerto cuenta con cámaras y sensores de diferentes tipos con tal recoger datos de forma masiva y a tiempo real, que van des de cámaras, sensores de control de entradas y salidas, atmosféricos, sensores de clima y medio marino, radares de navegación marítima y sistemas internos de telecomunicaciones; datos que son todos analizados y tratados con tal de contribuir a crear un modelo digital del Puerto.

La gestión automática del alumbrado, la automatización de los controles de entrada y salida o la monitorización de las colas de camiones en las entradas con tal de planificar el tráfico interno del Puerto, son algunos ejemplos de las mejoras operativas actuales que proporciona el Digital Port.

Otras iniciativas destacables son la plataforma telemática del Puerto o El Port Management System que coordinada la gestión de todos los servicios que se prestan en él, remolcadores, amarradores, avituallamiento, sistema de previsión de temporales, etc.

El recinto también dispone de sensores atmosféricos repartidos por el Puerto para determinar la calidad del aire en las diferentes zonas, a tiempo real, compartiendo los datos con el Ayuntamiento de Barcelona y la Generalitat de Catalunya para crear mapas interactivos de emisiones.

Está también en desarrollo una herramienta de cálculo de emisiones CO_2 online, que controla las emisiones de CO_2 de los barcos, para que de esta forma las empresas sepan la huella de carbono que generan, con el objetivo de escoger el modo de transporte más sostenible según cada caso.

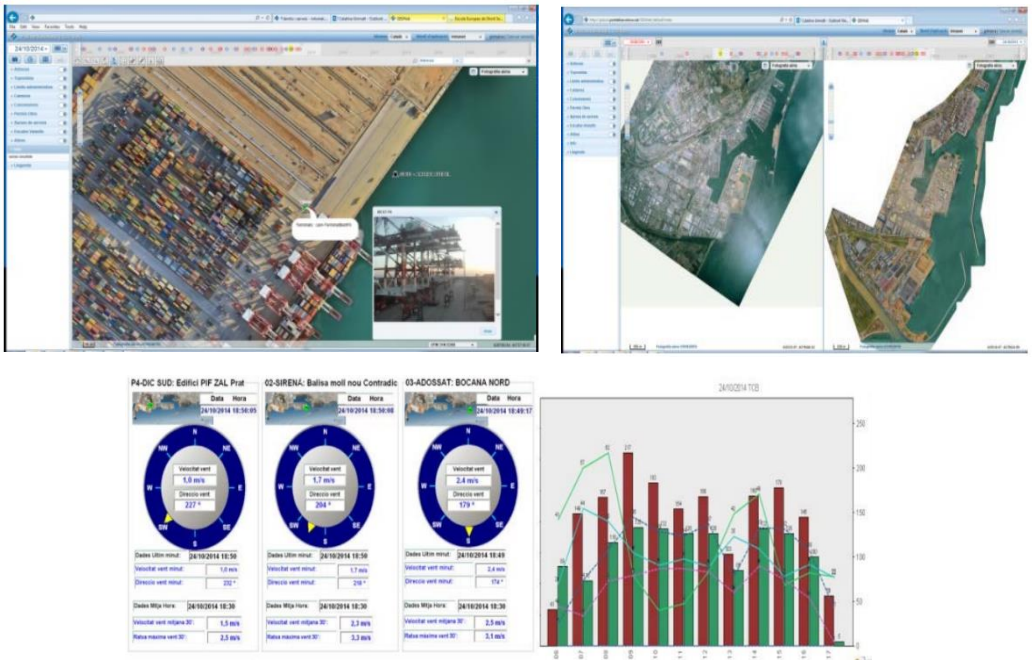


Fig. 2.3. 3. Muestra del Digital Twin del Puerto de Barcelona.[17]

- *Futuros retos*

En definitiva, se ha visto a lo largo de los ejemplos los beneficios que esta tecnología puede ofrecer y cómo ya está siendo usada por diferentes empresas y organizaciones para aumentar la efectividad y sostenibilidad de sus procesos. Falta ahora que, dados los beneficios de la integración digital, las personas y las empresas "evolucionen" en esa dirección para que la totalidad del ciclo sea posible.

Para lograrlo se necesita más investigación e información en los numerosos procesos para poder comprender tanto los problemas como las oportunidades que pueden ofrecer.

La cuarta revolución industrial o Industria 4.0 ya es una realidad y está establecida en muchas industrias como las manufactureras -automóvil, automóviles inteligentes, aeronáutico, energética…-y parece imparable, en un futuro cercano, su influencia en la industria de la construcción de una forma u otra.

CAPÍTULO 3: DISEÑO DEL SISTEMA

Una vez analizados y vistos tanto las ventajas como los componentes necesarios para crear un Digital Twin, se quiere aplicar este concepto de una forma práctica y a través de un ejemplo real y propio.

Será un ejemplo simplificado, concentrándose en un elemento estructural individual, pero con el objetivo de enseñar la aplicabilidad real que reside en la idea del Digital Twin para finalmente crear un prototipo aplicable a vigas de acero inoxidable o pórticos de mayor entidad.

Para ello, se procede a diseñar un sistema adecuado para su aplicación en este trabajo, con las herramientas disponibles, que se describen a continuación.

3.1. HERRAMIENTAS UTILIZADAS

- *Procedimiento*

Las herramientas proporcionadas, son en primer lugar, un **sensor de distancia** por ultrasonido que, colocado debajo de la viga a ensayar, medirá la distancia de la flecha que se produce debido a la carga aplicada. En segundo lugar, se cuenta con un microcontrolador **Arduino** que, conectado al sensor de distancia, procesará las medidas recibidas y que a su vez, enviará dicha información a la plataforma de código abierto **Processing**, que contendrá el código que presentará gráficamente los parámetros deseados.

El objetivo pues, es crear mediante dichas herramientas, un Digital Twin de un ensayo de laboratorio, cuyo dato de entrada será la flecha de la viga a ensayar, medido mediante el sensor de distancia por ultrasonido, y calculando a partir de esta, la carga correspondiente a la que está sometida la viga. Con dichos datos se pretende presentar al usuario una plataforma visual donde ver, en tiempo real, gráficas que muestren y den información sobre la evolución del ensayo.

Para elaborar la interfaz gráfica, se utilizará las plataformas de código abierto Arduino y Processing. De forma general, el primero será el encargado de captar y procesar los parámetros físicos, y el segundo de mostrar visualmente la información.

La base del proceso cuenta con 3 pasos bien definidos:

1) **Sensor y plataforma Arduino:** Medición de la flecha δ en un punto determinado de la viga, x^* (x^*=L/2 en el caso lineal, x^*< L/3 en el caso no lineal).
2) **Processing:** Calcular la carga P^* asociada a esa flecha en ese punto (relación lineal entre carga y flecha en el caso lineal y método numérico a partir de un cero de funciones en el caso de la viga de acero inoxidable.)
3) **Processing:** Calcular y visualizar la flecha en cada punto de la viga y el resto de variables que se requieran para ese estado de carga P^*

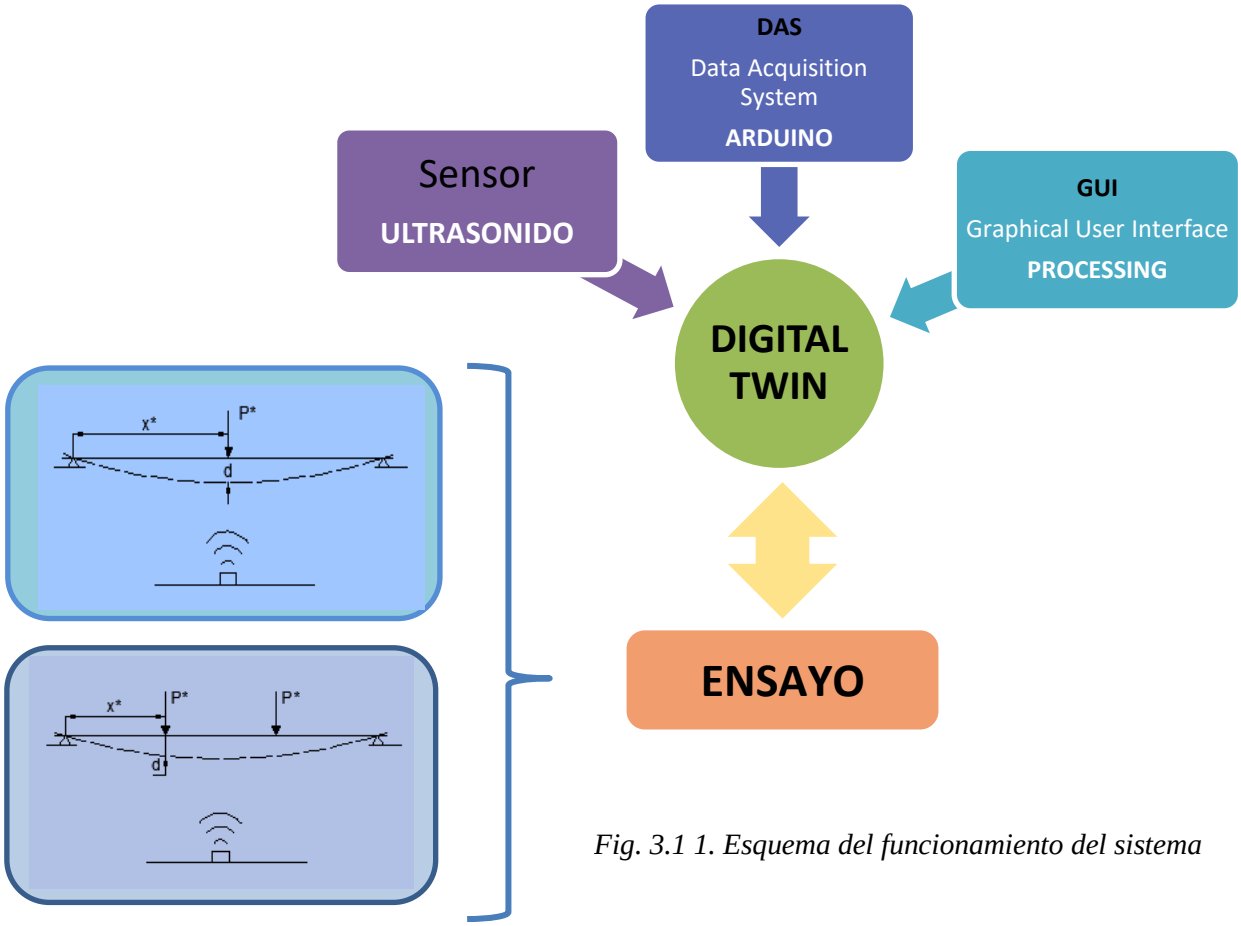


Fig. 3.1 1. Esquema del funcionamiento del sistema

Una vez diseñada y creada la recibida de datos, la comunicación de ésta y el diseño de la interfaz gráfica que constituirá el Digital Twin, se diseñará las características del ensayo que será la aplicación real de éste, sirviendo además, como calibración comprobando su fiabilidad, medida, cálculo y visualización.

Para ello, se explicará en primera instancia en qué consisten, y cómo funcionan las herramientas y plataformas utilizadas, para finalmente definir correctamente el sistema que se pretende.

3.1.1. SENSOR DE ULTRASONIDO

Los sensores son los elementos responsables de transformar la energía que reciben –mecánica, calorífica…– en energía eléctrica.

El sensor utilizado en este trabajo es el sensor de distancia por ultrasonido; este tipo específico de sensor envía un pulso ultrasónico que reflejados en un objeto, recibe el eco producido, convirtiéndolo en señales eléctricas y midiendo el tiempo que tarda la señal en regresar.

Es capaz de detectar la distancia de objetos en movimiento, de cualquier tipo de material o color –siempre que sea dentro de su rango óptimo–, pero también cuenta con algunas limitaciones o fuentes de error que hay que considerar; como por ejemplo, la influencia de la temperatura, que altera la velocidad de propagación de la onda del sonido o la posible dispersión del eco por una mala posición del sensor o el elemento a medir.

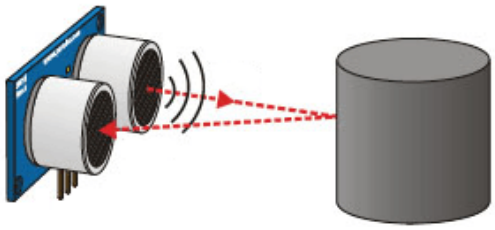


Fig.3.1.1.1 Esquema funcionamiento del sensor

El sensor de ultrasonido utilizado será el modelo **HC-SR04**, cuyo rango óptimo va de unos 15 a 70 cm, aunque, en principio abarca una distancia entre 2 y 450 cm según sus especificaciones. Las especificaciones completas del sensor se adjuntan en el Anexo 1 del presente documento

La alimentación es de 5V y trabaja a una frecuencia de 40 KHz, los pines de conexión, son, tal y como se indica a la siguiente figura:

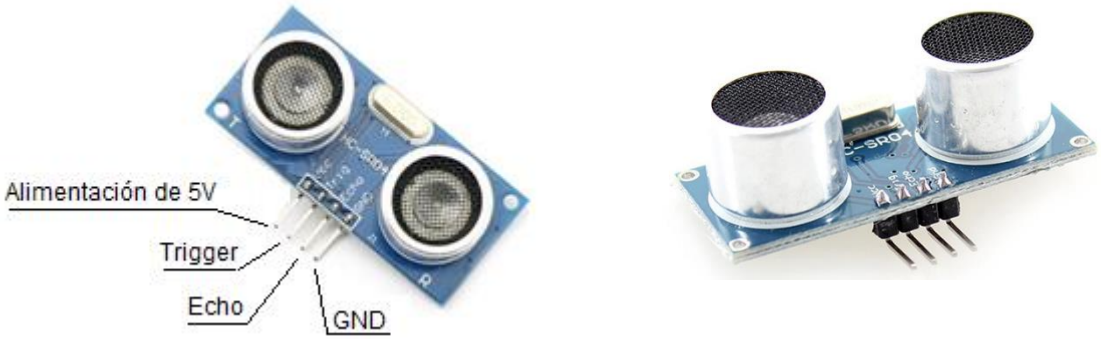


Fig.3.1.1.2 Sensor de ultrasonido HC-SR04

- VCC: Encargado de la alimentación del sensor a través del pin de conexión 5V de la placa *Arduino.*
- Trig: Emisor de la señal ultrasónica.
- Echo: Eeceptor de la señal de retorno de la onda ultrasónica.
- GND: Ground o tierra. Referencia de tensiones en el circuito – voltaje nulo–.

Dichos pines de conexión se conectarán a la placa Arduino, mediante cables, tal y como se especifica en el siguiente apartado, para crear el circuito encargado de recibir los estímulos físicos que serán procesados posteriormente.

Finalmente, y dado que el parámetro calculado por el sensor es el tiempo, será necesaria la conversión de tiempo (µs) a distancia ([3]mm); ésta se hace mediante una sencilla relación entre ambos parámetros y la velocidad del sonido (Ec.3.1.1) y, seguidamente la conversión a las unidades deseadas (Ec.3.1.2). La ecuación resultante (Ec.3.1.3) será la introducida en el programa.

$$t = \frac{2d}{c} \qquad c \cong 340\ m/s \qquad \text{(Ec.3.1.1)}$$

$$\frac{340\ m}{s} * \frac{1\ s}{1000000\mu s} * \frac{1000\ mm}{1m} = \frac{2d}{c} \qquad \text{(Ec.3.1.2)}$$

$$d = \frac{17t}{100} \qquad \text{(Ec.3.1.3)}$$

[3] Dado que la medida requerida es la flecha de la viga ensayada, se ha considerado el milímetro como la unidad adecuada para definición de la distancia.

3.1.2. SISTEMA DE ADQUISICIÓN DE DATOS. ARDUINO

Un sistema de adquisición de datos (DAS por sus siglas en inglés), es el proceso de medir a través de un ordenador, un fenómeno eléctrico o físico, proveniente de sensores y un hardware de medidas.

Para el desarrollo de este trabajo se ha escogido el sistema *Arduino*, creado por una compañía italiana con su mismo nombre, que diseña y manufactura placas de desarrollo de hardware de bajo coste y software libre. Proporciona además de un ordenador de placa reducida –microcontrolador–, el código abierto para poder programarlo, con el objetivo de acercar el mundo de la programación y la electrónica básica, a todo tipo de usuarios, creando proyectos interactivos a través de sensores y dispositivos de bajo coste.

- *Microcontrolador:*

El microcontrolador es la placa electrónica que procesa la información que recibe del sensor o dispositivos conectados en forma de circuito. De esta forma, la placa es capaz de leer los datos que recibe (input) y convertirlo en un dato de envío (output).

La forma de procesar dicha información proviene del código escrito en la plataforma Arduino (IDE) que estará previamente cargado en el microprocesador.

La placa utilizada en este caso, es el modelo **Arduino MEGA 2560,** que dispone de 54 pines digitales, 16 analógicos y cuenta con unas prestaciones un poco superiores al modelo Arduino básico (Arduino ONE).

En la siguiente figura se especifican los elementos de los que se compone el microprocesador:

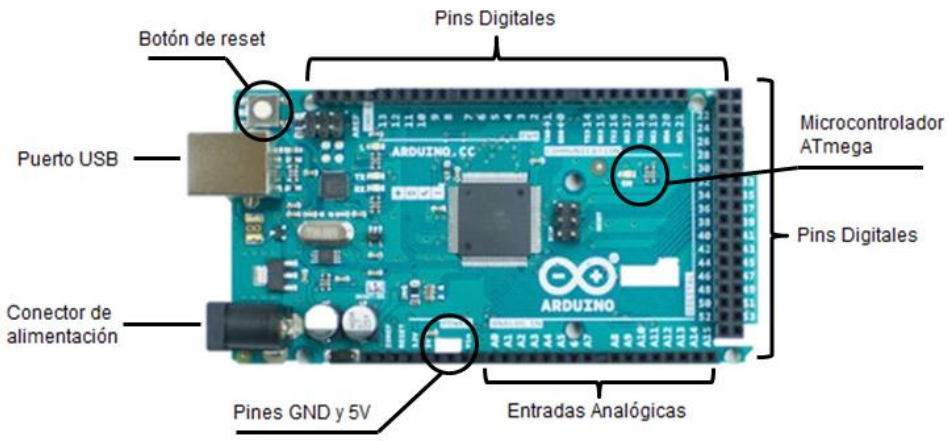


Fig. 3.1.2.1. Microcontrolador Arduino Mega 2560

- **Componentes del microcontrlador:**

- **Puerto USB:** Conector encargado de la comunicación, alimentación y carga de los programas provenientes de Arduino.
- **Botón de reset:** Puesta a cero del microcontrolador ATmega.
- **Pines Digitales.** Entradas/salida digitales de la placa.
- **Microcontrolador ATmega:** Cerebro de la placa.
- **Entradas Analógicas.** Entradas/salida analógicas.
- **Pin GND:** Pin para proporcionar la toma de tierra (0 voltios) para circuitos externos a la placa.
- **Pin 5 V:** Pin para proporcionar una tensión de 5 V.
- **Conector de Alimentación:** Alimentación de la placa en caso de no estar conectado a un puerto USB.

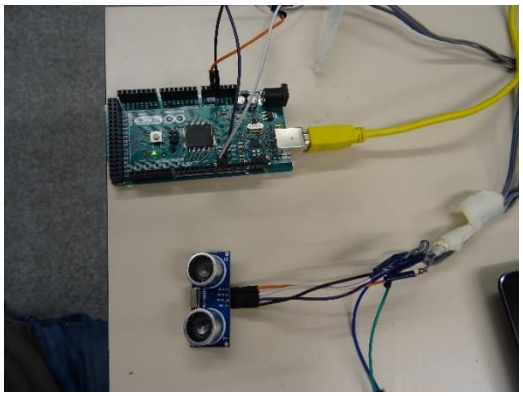
- *Circuito*

Mediante una toma de corriente –en este caso una conexión USB– se creará un circuito permitiendo que la electricidad circule a través de los diferentes elementos, con conexiones cerradas mediante cables.

En un circuito, la electricidad fluye de un punto con un potencial de energía más alto (+) –proveniente de la carga– hacia un punto de potencial más bajo a través de conductores, y diferentes componentes que posibilitan que fluya la carga. La toma de tierra (-/GND) es normalmente el punto con menor potencial de energía en un circuito.

El circuito que se construirá para obtener los datos necesarios, se compone de la unión de la placa Arduino –microcontrolador– y el sensor de ultrasonido mediante cables y utilizando el ordenador como fuente de energía, a través de la conexión USB.

El código cargado en la placa se encargará de hacer el cálculo de la conversión del tiempo medido, a distancia. Este parámetro será enviado a Processing, que lo utilizará como dato inicial, a partir del cual se creará el Digital Twin.



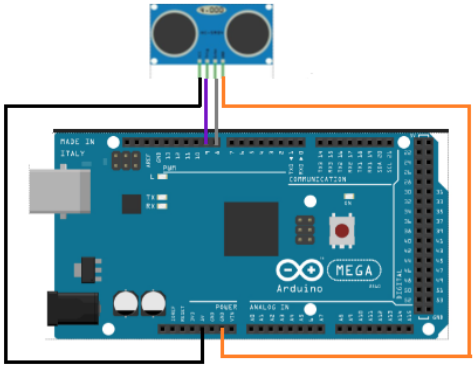


Fig. 3.1.2.2. Esquema de conexión entre el sensor de ultrasonido y la placa Arduino

- *Software*

Las placas o microcontroladores *Arduino* se alimentan a través de un puerto USB –conectado directamente al ordenador del usuario– se programan a través del software facilitado por la compañía y se componen de dos tipos de elementos; el entorno de desarrollo IDE –basado en Processing y la estructura del lenguaje de programación Wiring– y un cargador de arranque que se ejecuta automáticamente dentro del microcontrolador al ser encendido.

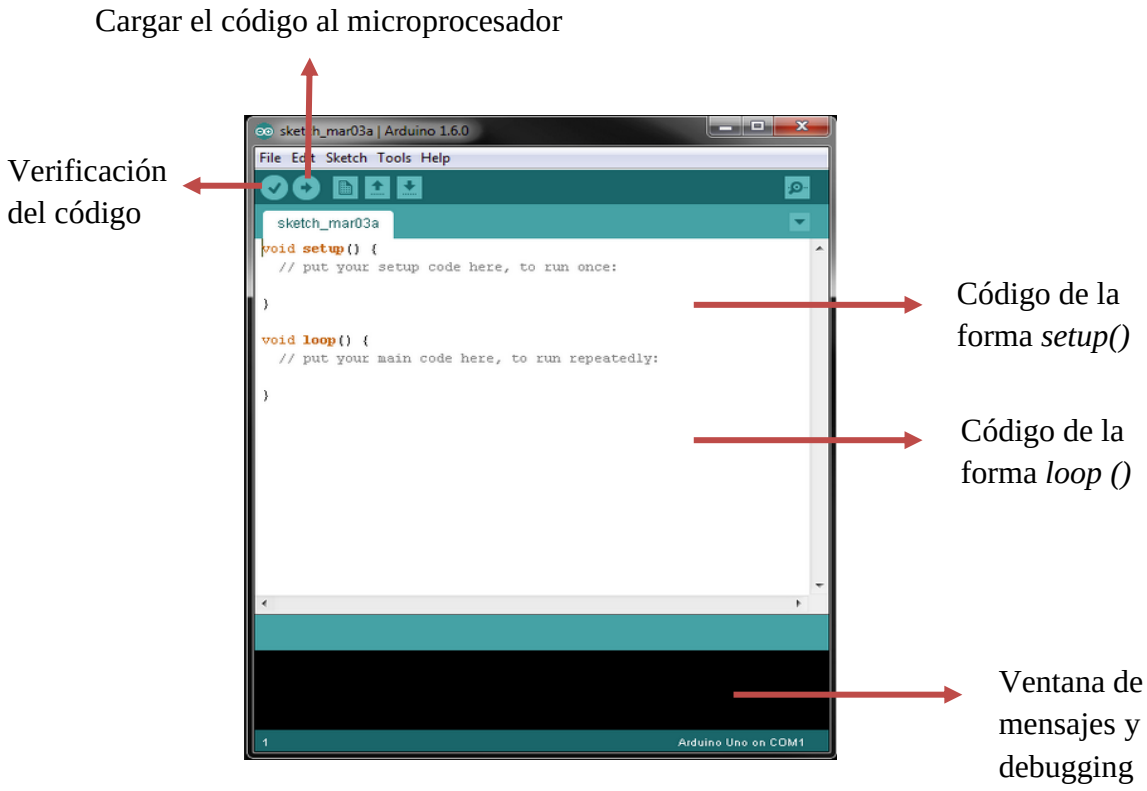


Fig. 3.1.2.3. Visualización del entrono Arduino

Tal y como se ve en la figura 3.1.2.3, el editor de texto donde se escribe el código se divide en dos partes;

- **Void setup ()**

  Las líneas de código escritas en este apartado, serán ejecutadas una sola vez por el programa.
- **Void loop ()**

  El apartado funciona como un bucle y el código se repetirá indefinidamente.

- *Código*

Una vez conocidos los elementos y sus conexiones falta la determinación del código que procesará la información recibida del sensor.

Dicho código es bastante estándar y consiste en indicar, en primer lugar, qué pines del sensor irán conectados a qué pines del microcontrolador así como cuál recibirá información y cuál la mandará (1), en segundo lugar generar el pulso de emisión (2), seguidamente medir el tiempo entre los pulsos emitidos y recibidos (3), convertir dicho tiempo en distancia (4) y finalmente, mandar el dato requerido (5).

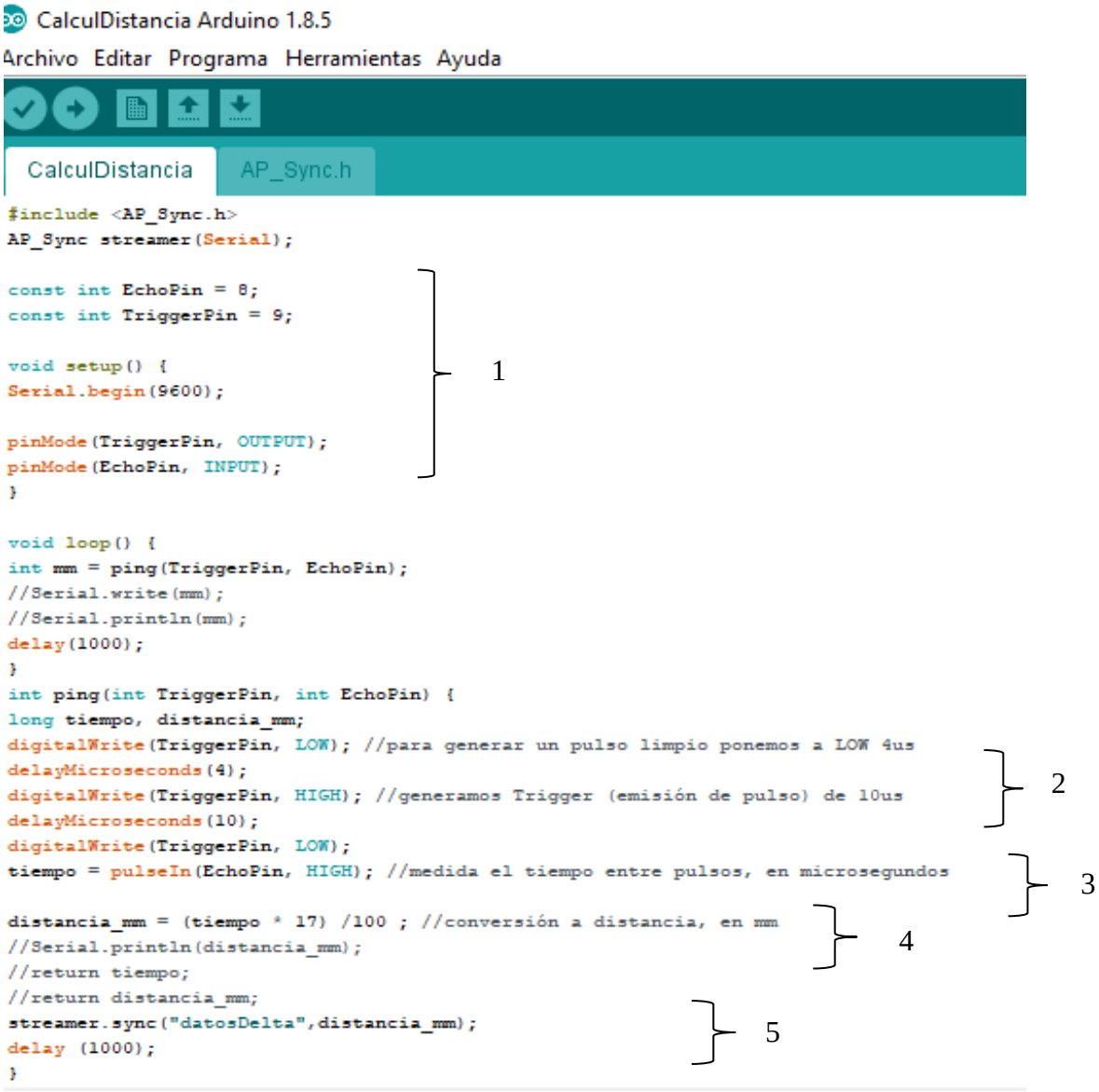


Fig. 3.1.2. 4 Código de Arduino para la obtención de los datos

3.1.3. INTERFAZ GRÁFICA DE USUARIO. ENTORNO DE DESARROLLO PROCESSING.

Una interfaz gráfica de usuario –GUI por sus siglas en inglés– es un programa informático que representa información utilizando imágenes y objetos gráficos proporcionando una interacción visual al usuario.

La GUI escogida para este trabajo, es ***Processing***, una plataforma de código libre que utiliza el lenguaje y entorno de desarrollo basado en Java, compatible con el lenguaje de Arduino y que da la oportunidad de crear proyectos multimedia e interactivos de diseño digital. Se ha escogido esta plataforma debido a su utilidad y sencillez para comunicar, mostrar, guardar y visualizar datos provenientes de Arduino.

Processing es un lenguaje de programación orientado a objetos (OOP) derivado del lenguaje C. Un lenguaje orientado a objetos, combina los datos e instrucciones del programa para crear "objetos". Esto significa que cada objeto es una entidad con atributos –características- y comportamientos –acciones– propios; realizan un conjunto de actividades propias de cada objeto, y son capaces de relacionar sus datos con otros objetos para combinar varias actividades.

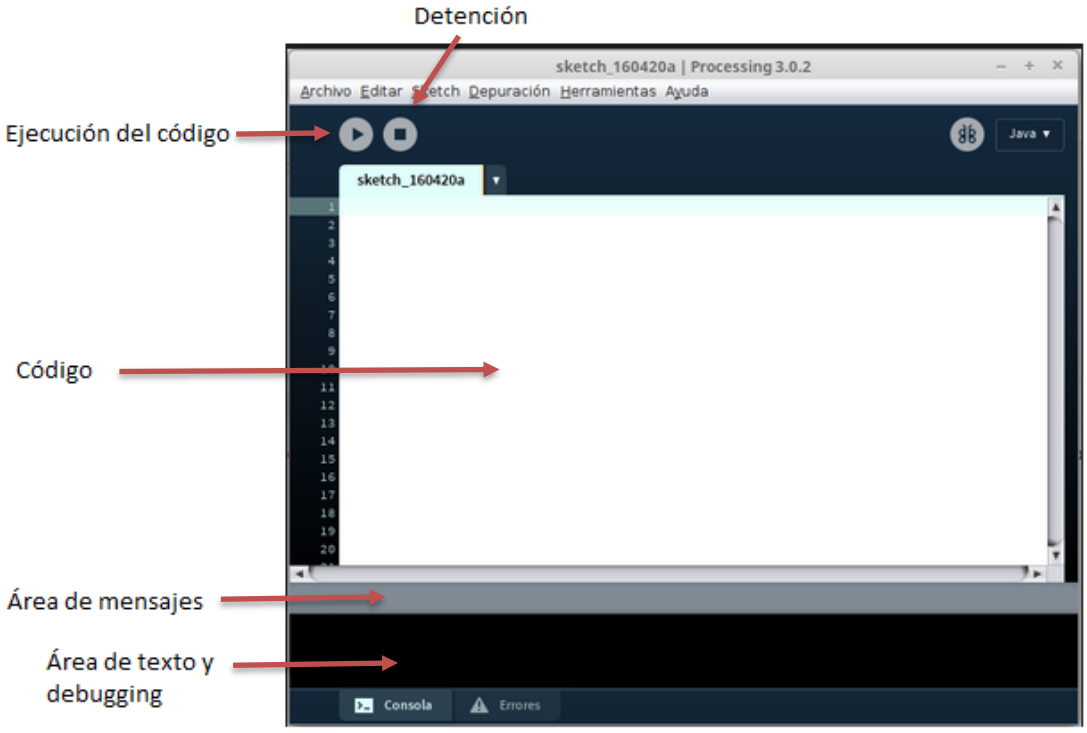


Fig. 3.1.3.1. Entorno de desarrollo Processing

En la figura 3.3.1, se muestra el entorno de Processing, donde el código se escribe mediante un editor de texto, y permite dibujar tanto gráficos en dos como tres dimensiones. El render estándar dibuja en dos dimensiones pero cuenta también con la posibilidad de renderizar mediante el P3D render, que permite controlar la posición visual del objeto, la luz o materiales creando visualizaciones en tres dimensiones.

Al igual que Arduino, Processing divide su entorno en dos partes;

- **Void setup ().** Ejecución del código una sola vez
- **Void draw ().** Ejecución del código en bucle.

Processing cuenta con distintos modos de programación donde Java es el utilizado por defecto, pudiendo agregar de forma sencilla otros modos compatibles.

Para ver el resultado del código que se está programando, solo es necesario apretar el botón de *play* que ejecuta dicho código y, que en caso de haber errores, estos se muestran en la ventana inferior de la pantalla.

Las coordenadas (0,0) del programa se encuentran en la esquina superior izquierda de la pantalla, con lo que por defecto todas las distancias que se le introduzcan estarán referenciadas a ese punto, y con el pixel como unidad de trabajo. Estas coordenadas se pueden trasladar a otros puntos deseados mediante la sintaxis correspondiente.

Para dibujar un elemento, solo es necesario indicarle el objeto y la posición deseada. Por ejemplo, *line (50,50,150,50)* dibujará una línea de la posición (50,50) a la (150,50) de color negro y con un pixel de grosor por defecto, estas características pueden modificase fácilmente dándole otro color, grosor o movimiento.

Con tal de ampliar las prestaciones implícitas del código base de Processing, se pueden utilizar librerías externas a fin de extender las acciones u objetos propios del programa. Estas librerías son de libre acceso y han estado programadas por otros usuarios y se añaden fácilmente desde el propio entorno del programa.

3.2. DISEÑO DEL SISTEMA. CASO LINEAL.

3.2.1. Configuración de la interfaz

Dado que en primer lugar se busca familiarizarse con los entornos descritos, se hará un ensayo reducido consistente en la carga de una pletina de acero al carbono.

Esto se ha decidido debido a que las bases matemáticas que definen el comportamiento del acero al carbono y la configuración estructural de una viga biapoyada son más sencillas y ampliamente estudiadas. De esta forma, en este ensayo los esfuerzos estarán concentrados en el diseño y programación de la interfaz en sí.

Para ello, antes de empezar la programación propiamente dicha, previamente se ha diseñado la configuración visual que se pretende mostrar.

Dicha configuración consistirá en una pantalla dividida en cuatro cuadrantes, donde cada uno mostrará parámetros referentes al ensayo.
La configuración visual que se pretende mostrar es aproximadamente, como la de la siguiente figura:

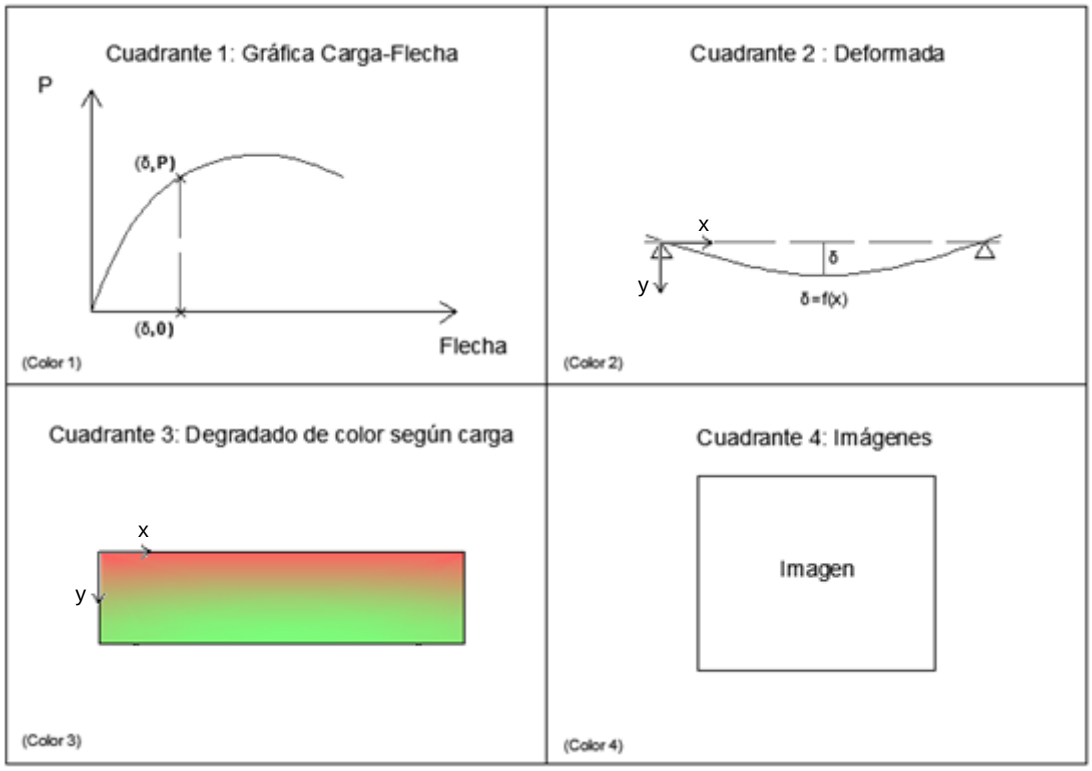


Fig.3.2.1.1. Esquema del diseño de la interfaz gráfica

- Cuadrante 1: Gráfica Carga-Flecha.

Dado que el dato que recibirá Processing del sensor, es la flecha de la viga y, mediante la relación lineal que existe entre la carga aplicada a una viga y su flecha, se calculará la una en función de la otra para visualizarse a tiempo real la correspondiente gráfica Carga-Flecha.

En el apartado punto 4 "*Bases matemáticas"* del presente trabajo, se especifican las ecuaciones utilizadas para relacionar los dos parámetros.

- Cuadrante 2: Deformada de la viga.

También se considera interesante ver la evolución de la deformada de la viga a medida que aumenta la carga. Dicha deformada también vendrá dada por la relación flecha-carga medidas en un punto, aproximando los demás puntos de la curva de acuerdo con las ecuaciones especificadas más adelante.

- Cuadrante 3: Evolución hasta plastificación

Se hará una representación de la sección transversal de la pletina, donde, mediante un degradado de colores, de una forma rápida y visual se aportará al usuario, información sobre la magnitud de la fuerza aplicada desde su inicio (P=0) hasta la eventual plastificación de la sección.

- Cuadrante 4: Fotografías

Finalmente, en el último cuadrante se integrarán fotografías reales del ensayo, que irán asociadas a los incrementos pertinentes y sucesivos de la deformación.

3.2.2. Código

La estructura del código que materializará la configuración visual que se ha diseñado seguirá de forma general, el siguiente orden:

1) Importación de las librerías necesarias.
2) Declaración de variables. Es importante tener claro qué tipo de dato se va a tratar, y esto se define a través de su declaración.
   Los tipos de variable pueden ser *integer* (números enteros), *float* (números reales), *string* (texto) entre otros, y cada uno es utilizado y tratado de una forma distinta por el programa.
   Cada objeto puede aceptar uno o varios tipos de variable.
3) Definición de la parte fija de la interfaz. *Setup()* . Esta parte del código determina el tamaño de la pantalla, el color, etc. Es también en este punto donde se realiza la comunicación entre Arduino y Processing, es decir la recepción de los datos.
4) Definición de la parte variable – ejecutada como bucle– de la interfaz. *Draw()*. Esta sección del código, es la encargada de materializar las acciones principales del Digital Twin, como es calcular la carga en función de la flecha, dibujar la gráfica correspondiente, la deformada, la sección transversal con degradado de color y la relación de las fotografías con el avance del ensayo.

   A continuación se exponen algunos ejemplos de la estructura del código. Dada su longitud, el código completo se adjunta en el Anexo 3 del presente documento.

(1) Importación de librerías.

```
import apsync.*;        //Libreria para sincronzar Arduino con Processing
import processing.serial.*;

AP_Sync streamer;
```

Ap_Sync es una librería que facilita la comunicación entre Arduino y Processing, especificando qué dato de Arduino se quiere enviar a Processing. De esta forma se asegura que no haya errores en los datos enviados, por ejemplo, que se envíe el dato tiempo y no el dato distancia.

(2) Declaración de variables

```
      //VARIABLES CONSTANTES ECUACION DELTA-P


float I=106.67;     //inercia (mm^4)
float E=210000.0;      //modulo de Young (MPa)
float longViga=1500.0;  //mm
float x=longViga/2;
float cte=4.0/3.0;      //cte auxiliar de la ecuacion de la deformada.

float h=600.0;            //Distancia entre sensor y viga


      //VARIABLES DE ENTRADA ARDUINO (DELTA)


float [] delta =new float [5000];
public float datosDelta;
float dist;
int j=0;

    //VARIABLE RESULTADO -CARGA-


float [] P=new float [5000];
```

Antes del *setup* se declaran las variables que se utilizarán a lo largo del programa. En el fragmento de programa mostrado se pueden observar algunos ejemplos como la declaración mediante *float* de las variables necesarias para el cálculo de la ecuación carga-flecha, deformada o las variables de entrada y salida en forma de vector (*float* [ ]).

(3) Definición del *setup*.

```
void setup() {

size(1000,700);

//printArray(Serial.list());              //Saber que COM esta disponible

streamer=new AP_Sync(this,"COM3",9600);  //Comunicación con Arduino (recibe dato distancia)

delay (500);

               //PANTALLA PRINCIPAL

 fill(169,204,227);                    //cuadrante 1
 rect(0,0,width/2,height/2);

 fill(171,235,198);                   //cuadrante 2
 rect(width/2,0,width/2,height/2);

 fill(232,218,239);                    //cuadrante 3
 rect(0,height/2,width/2,height/2);

   fill(235,237,239);                  //cuadrante 4
 rect(width/2,height/2,width/2,height/2);
```

(1

(2

(3

En el ejemplo anterior se observa cómo se define el tamaño de la pantalla (1) mediante la sentencia *size*, la comunicación con Arduino (2) y la definición del tamaño y color de cada cuadrante usando un código RGB (3).

(4) Definición de las acciones a ejecutar en cada cuadrante. Ejemplo cuadrante 2.

```
 pushMatrix();
 translate(X2+100,Y2+40);              //Definicón de coordenadas locales,  cuadrante 2.1


for(int i = 0; i < 150; i++){


     float deformada = (pow(300,2)*i/(16*E*I))*(1-(4.0/3.0*pow(i,2))/(pow(300,2)))*d; //Calcular curva de la deformada

          strokeWeight(1);
            stroke(120,145,169);
                 line(i,0,i,deformada);               //Dibujar deformada (x<L/2)

}
popMatrix();
```

Aquí se muestra cómo, mediante un bucle *for* y la expresión matemática correspondiente, se le indica al programa cómo dibujar la curva de la deformada a medida que recibe el dato de la medida de la flecha.

3.2.3. Dificultades encontradas

Dificultad 1:

La primera dificultad encontrada ha sido respecto al cuadrante 1. Una vez definida la configuración visual base de los cuadrantes –sus colores, los ejes de la gráfica y su posición– al pasar datos de Arduino a Processing, las líneas verticales que tenían que definir la gráfica Carga-Flecha, se visualizaban de una en una. Es decir, recibía un dato, dibujaba la línea correspondiente y al ejecutar otra vez el código, recibía un dato nuevo dibujando una nueva línea, sin mantener la primera.

Solución:

Dado que la intención es que se visualicen todos los datos obtenidos, y de forma acumulativa a lo largo del eje, se solucionó creando un "Array".

Este objeto, es una lista o vector que guarda los datos que se le indiquen, cada uno en una posición de dicho vector. Los datos a guardar en este caso, eran las mediciones de distancia provenientes de Arduino. Con el objeto "Array" y un bucle del tipo "*for*" se consiguió que las líneas verticales permanecieran acumulativamente en la posición que correspondiera a medida que recibía nuevos datos.

Dificultad 2:

En segundo lugar y referente a este mismo cuadrante, se observó que había un fallo en el cálculo de la carga P ya que, el programa solo daba 5 números –siempre los mismos–. Al principio se pensó que el problema radicaba en la propia expresión de P, pero finalmente se concluyó que el problema era la comunicación entre Arduino y Processing.

Arduino por su lado, daba datos correctos de la distancia medida por el sensor, y la expresión para calcular la carga P también era correcta, el problema era que Processing solo recibía 5 tipos de números (aleatorios) y en consecuencia solo calculaba 5 valores de la carga.

Solución:

Buscando referencias del mismo tipo de error se encontró una librería oficial de sincronización de datos entre Arduino y Processing (librería AP Sync). Instalando dicha librería y con su sintaxis correspondiente, se fue capaz de indicarle a Arduino qué parámetro en concreto emitirle a Processing para su posterior procesado. De esta forma, los valores resultantes de delta y el correspondiente cálculo de la carga P y la gráfica asociada fueron coherentes.

Dificultad 3:

La siguiente dificultad fue referente al segundo cuadrante, donde se pretendía dibujar la curva de la deformada. Processing presenta varias opciones que se pueden utilizar con ese fin, como por ejemplo a base de puntos, definiendo líneas cortas a lo largo del trazado de la curva, crear una curva definiendo varios puntos de control (Spline), entre otras.

El problema de la mayoría de estas opciones era lo laborioso que podría suponer la definición de cada punto individual de la curva. También se probó de buscar una librería que contuviera la opción de definir una función y que el programa en sí la dibujara con los datos dados.

Solución:

Al no encontrar librerías de fácil aplicación o que no cumplieran los requisitos, se optó por una solución más fácil: crear líneas verticales a lo largo de la viga que definieran con su longitud, la ecuación de la deformada.

Dificultad 4:

Respecto al tercer cuadrante, fue relativamente complicado conseguir que el degradado de color fuera coherente y mínimamente estético. Se necesitaba una relación entre la carga aplicada y los colores sucesivos que se querían visualizar.

Solución:

Se solucionó buscando la combinación de colores deseada, yendo de verde a rojo a medida que se cargaba la pletina y, definiendo la variación de los colores como variables en un bucle. Para que se visualizara como un degradado y no como rectángulos de colores diferenciados, se hizo una proporción entre el ancho del degradado a lo largo de la representación de la sección transversal y la carga.

Dificultad 5:

Una vez conseguido que en su mayoría, que el programa funcionara se ultimaron los detalles, como el de graduar los ejes de la gráfica del primer cuadrante y que dicha gradación fuera acorde con los resultados que se visualizaban por pantalla.

Solución:

La solución fue relacionar los píxeles correspondientes al eje, con la flecha medida en milímetros por el sensor, para que correspondiera con la correcta graduación de cada eje.

3.3. DISEÑO DEL SISTEMA. CASO NO LINEAL.

3.3.1. Configuración de la interfaz

En este caso, se pretende crear un prototipo de Digital Twin para su aplicación en vigas reales de acero inoxidable, teniendo en cuenta el comportamiento de dicho material en la configuración de la interfaz. El diseño que se querrá visualizar será parecido al modelo prueba descrito anteriormente.

- Cuadrante 1. Gráfica carga-flecha
- Cuadrante 2: Deformada de la viga
- Cuadrante 3: Visualización 3D de la viga y su evolución durante el ensayo. Mostrará la viga en 3 dimensiones con la evolución de la deformación respecto a la carga aplicada. Estas imágenes se han obtenido mediante el cálculo de elementos finitos desarrollado por integrantes del grupo de investigación de estructuras metálicas, como parte del proyecto PINOX.

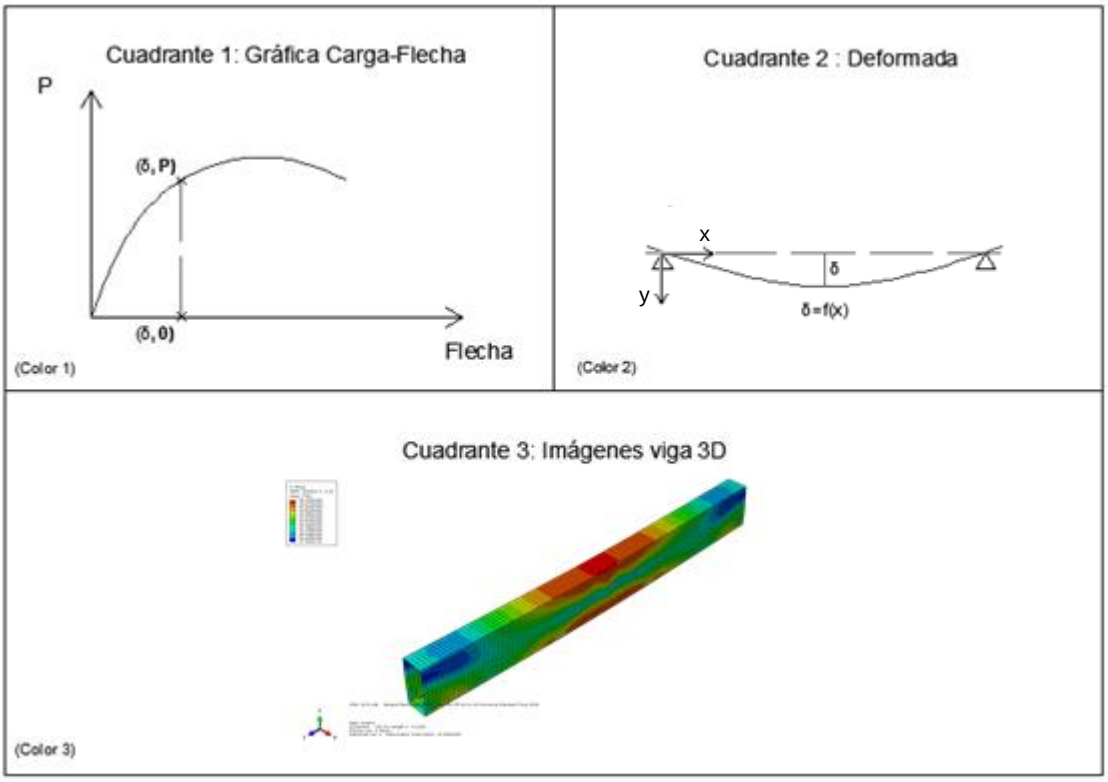


Fig. 3.3.1.1. Esquema del diseño de la interfaz gráfica. Caso no lineal.

3.3.2. Código

El esquema general del código, en este caso, es parecido al del caso lineal.

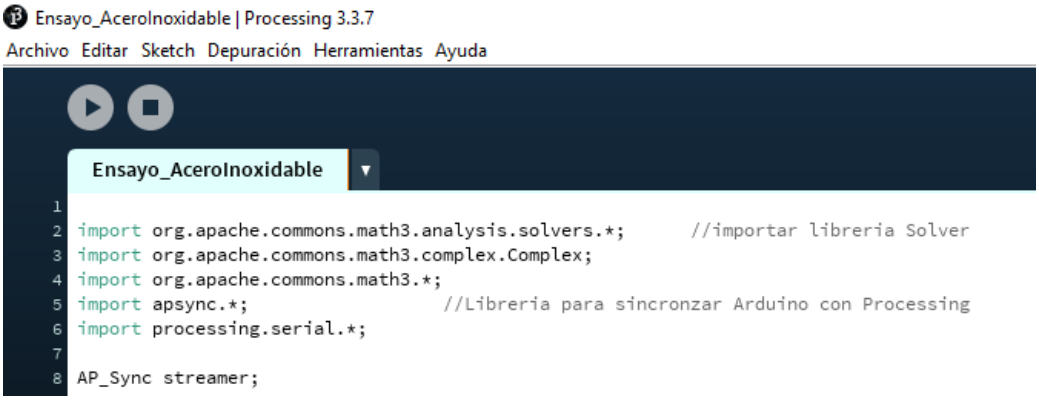
1) Importación de librerías
2) Declaración de variables
3) Definición del *set-up ().*
4) Definición del bucle *draw ().*

La diferencia principal en este código es la no linealidad del comportamiento del material, es decir la notable diferencia de las ecuaciones a resolver por el programa. Con tal de indicarle cuál es la expresión a resolver, relación carga- flecha, se hicieron previamente cálculos –definidos en el capítulo 4 *"Bases matemáticas*"– para la reducción de las expresiones más complejas que definen el comportamiento no lineal del acero inoxidable.

Como se verá más adelante la ecuación a resolver por el programa son de cuarto grado y requiere un cero de funciones con tal de obtener sus raíces. También cuenta con cuatro tramos que determinan la curva de la deformada y, que en consecuencia, será necesario definirle cada uno de estos tramos para que tenga continuidad.

A continuación se muestran algunos ejemplos del código. Dada su longitud, el código completo se adjunta en el Anexo 4 del presente documento.

1)Importación de librerías



Con tal de resolver la ecuación que determinará la carga P en función de la flecha, se importa la librería principal que la resolverá (.math3), esta librería procede del proyecto *Apache Commons Math*[4], y para la resolución concreta de este problema se necesitan dos clases incluidas en ella; el propio *solver* (.analysis.solvers.) y el modo específico para la resolución de raíces complejas (.complex.Complex.).
También se importa, tal y como se ha visto en el caso lineal, la librería Ap Sync, encargada de la correcta comunicación entre Arduino y Processing

[4]Apache Commons Math es una serie de proyectos proveedores de componentes de software Java reutilizables, de código abierto.

2) Declaración de variables

```
            //VARIABLES PARA RESOLICIÓN DE P (a partir de datos Arduino)

double resultadoP; //resultado formato double no admisible para funcion line
float resultadoP_Def;  //resultado final de forma float para funcion line
float[] delta= new float [5000];
float[] Carga= new float[5000];

public float datosDelta;
float d;          //variable que guarda los datos obtenidos
int j=0;


           //COEFICIENTES DE LA ECUACION A RESOLVER -CARGA-

double coeff[] = {0,-1.66e-4,0,0,-1.557e-18};
Complex comp[];   //Declaramos comp como lista de numeros complejos
```

Se muestra la declaración de las diferentes variables necesarias para el cálculo de la carga P. Se han creado entre otros, vectores encargados de almacenar los datos recibidos del sensor (*float delta[]*), así como de la carga resultante de cada dato de entrada (*float Carga[]*).

A dichos vectores se les indica mediante el número entre corchetes, la cantidad de datos que almacenará, pudiéndose cambiar según necesidad.

Para indicarle al programa, que ecuación deberá resolver, y de acuerdo con la sintaxis correspondiente a la librería utilizada, se declara un vector, *coeff[]* –en este caso – y, entre las llaves se indica el valor del coeficiente que multiplica a la variable a resolver, es decir se indica según: {constante,x^1,x^2,x^3,x^4,….,x^n}. Para la resolución posterior, y dado que la ecuación tendrá raíces complejas, también se declara un vector para esos resultados, con el fin posterior de filtrarlos y solo obtener las raíces reales.

3) Definición del *set-up()*

```
void setup() {

  size(1000,700);                              // Tamaño pantalla

//printArray(Serial.list());                   //Para saber que COM esta disponible
 streamer=new AP_Sync(this,"COM3",9600);       // Comunicacion con arduino, recibe dato distancia
    delay (1000);

              //PANTALLA PRINCIPAL

 fill(169,204,227);                     //cuadrante 1
 rect(0,60,width/2,height/2+60);

 fill(171,235,198);                    //cuadrante 2
 rect(width/2,60,width/2,height/2);

 fill(232,218,239);                     //cuadrante 3
 rect(0,height/2+60,width,height/2-60);


  line(width/2,60,width/2,height/2+60);    //linia central vertical
  line(0,height/2+60,width,height/2+60);    //linia central horizontal


 line(50,height/2+10,width/2-20,height/2+10);          //eje "x"  cuadrante 1

  for (int posX=0; posX<21; posX++){                //graduación eje ''x''
       line(50+20*posX,365,50+20*posX,355);
 }

 line(width/2-20,360,width/2-30,height/2);      //flechas eje
 line(width/2-20,360,width/2-30,height/2+20);
```

La definición del set-up es parecida al del caso lineal, se define el tamaño de la pantalla, la comunicación con Arduino y las características de los cuadrantes –tamaño, color, ejes, su graduación, etc.–.

Se ha intentado referenciar las variables de sistema (ancho y alto) de manera que si se deseara cambiar el tamaño de la pantalla, la configuración cambie acorde. Asimismo, la graduación de los ejes se ha hecho con un bucle también para poder cambiarlo en caso necesario.

4) Definición de la estructura principal del código. Draw ()

```
                // RECIBIDA DE DATOS + CÁLCULO DE P

    d=datosDelta;
     println(d);

        LaguerreSolver P = new LaguerreSolver();        //declaracion calculo P

 delta[j] = d;  //guardar entrada arduino
 coeff[0]=  d; //cte de la ecuacion (posicion 0 del vector coeff) correspondiente al dato delta
// println(d);
 comp=P.solveAllComplex(coeff,0.0); //cálculo de todas las raices de la funcion
 //println(comp);        //Ver resultados del cálculo total de P por pantalla


  for(int c=0; c<comp.length; c++){
     if ( (comp[c].getImaginary() == 0) && (comp[c].getReal() >= 0.0)){      //for para escoger resultado de P real
        resultadoP = comp[c].getReal();      //si imaginario =0 y real diferente, guardar real= resultado de P, para dibujar linea
        resultadoP_Def = (float) resultadoP;          //conversion de double a float para funcion line
        Carga[j]=resultadoP_Def;

                        //DIBUJAR DEFORMADA

   for (int k=0; k<=150; k++){

   if (k<=100){
   float deformada = -(((4.9838e-36)*pow(5*k,6)-(3.27065e-21)*5*k)*pow(Carga[j],4)+((2.667e-13)*pow(5*k,3)-(4.10e-7)*5*k)*Carga[j]);
      //println(Carga[j]);
      //println(deformada);
    strokeWeight(1);
      stroke(120,145,169);
        line(k+600,235,k+600,deformada*0.1+235);                //Dibujar deformada x<L1
   }
```

Se explica a continuación de forma sucesiva las acciones que lleva a cabo el fragmento de código mostrado.

En primer lugar se asigna una variable – *d* – a los datos que llegan de Arduino a través de AP Sync -*datosDelta*-. Esta variable se usa a lo largo de todo el programa y se indica a continuación que será el dato que define los elementos sucesivos del vector *delta[]*. Asimismo, se determina que dicho dato "d", será el valor de la posición 0 –coeficiente constante – del vector *coeff[]* definido anteriormente.

Una vez determinados la relación de parámetros con los que trabajará el programa, se ejecuta el *solver*, que utilizará el método numérico de Laguerre[5].

A continuación se crea un bucle para filtrar los resultados, obteniendo así, solo las raíces reales de la ecuación y almacenando cada uno de esos resultados en una posición del vector *Carga[]*.

[5]El método numérico de Laguerre, se caracteriza por su uso exclusivo para resolver ecuaciones algebraicas polinómicas de coeficientes reales, iniciándose con cualquier aproximación inicial y convergiendo a la solución de todas las raíces de forma rápida.

Con los resultados del vector *Carga[]* obtenidos, se procede a dibujar la deformada –en este ejemplo se muestra la curva definida para $x<L_1$–.

La curva de la deformada se dibujará de forma completa para la medida de la flecha recibida con su correspondiente estado de carga.

3.3.3. Dificultades encontradas

Dificultad 1:

La ecuación de la flecha en función de la fuerza P correspondiente al problema no lineal, es una ecuación de cuarto grado, tal y como se ha comentado anteriormente.

Con tal de resolver las raíces de dicha ecuación era necesario encontrar una expresión general para la resolución de ecuaciones de cuarto grado o un método numérico de cero de funciones. Dado que las expresiones necesarias para dicha resolución eran notablemente complicadas, se buscó la existencia de un posible *Solve*r que pudiera resolverlo.

Solución:

Después de algo de investigación, se encontró un *Solver* basado en el método numérico de *Laguerre*, –proporcionado por el proyecto "Apache Commons Maths" – que cumplía la condición de resolver ecuaciones polinómicas de cuarto grado de una forma suficientemente sencilla y sin demasiado coste computacional –.

Ya que este método numérico da todas las raíces –reales e imaginarias– mediante el propio código se filtraron las soluciones de interés, las raíces reales.

Dificultad 2:

Aunque el procedimiento de programación para dibujar la deformada de la viga fuera el mismo que en el caso lineal, el hecho de referenciar la carga al resultado y, la dificultad de comprobar el resultado matemático complicó su ejecución.

Hicieron falta varias pruebas, así como cálculos auxiliares para su correcta continuidad y visualización de la curva.

Las expresiones correspondientes a los puntos situados a partir de L/2, eran laboriosas para su resolución a mano, por lo que se hizo una simetría, de las expresiones de $x<L/2$, aprovechando esta característica de la configuración estructural tratada.

CAPÍTULO 4: BASES MATEMÁTICAS

Cada uno de los ensayos realizados tiene metodologías claramente diferenciadas. El primer caso, como ya se ha ido comentado a lo largo del documento, se trata de una pletina de acero al carbono – caso a escala reducida–, y el segundo una viga de tamaño real y de acero inoxidable – caso a gran escala–.

Dado que los dos casos nada tienen que ver, se divide en dos subcapítulos las bases matemáticas en las que se han basado los cálculos en cada ensayo.

De forma general, las expresiones que se pretenden obtener son la carga en función de la flecha. Todo el proceso de visualización de datos parte de la medición de la flecha en un punto de la viga, a partir de la cual, se calculará la carga a la que está sometida. Esto se hará mediante expresiones analíticas que definan el comportamiento lineal o no lineal según cada caso, para que una vez obtenida la carga aplicada, estimar la flecha en el resto de puntos de la viga para su correspondiente visualización en Processing.

4.1. CASO LINEAL. PLETINA DE ACERO AL CARBONO

El primer caso a estudiar es una pletina biapoyada de acero al carbono. La teoría lineal que lo sustenta –tanto respecto a la configuración de la viga, como el material– son casos ampliamente estudiados dentro de la mecánica y la resistencia de materiales.

A continuación se describen las expresiones que determinan el comportamiento de la configuración estructural de una viga biapoyada con una carga puntual en centro luz, y que durante este trabajo han sido necesarias para determinar los parámetros correspondientes.

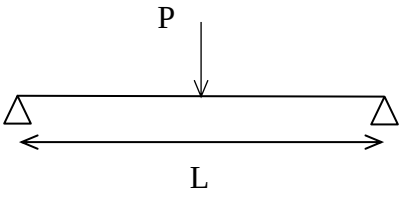


Fig. 4.1. 1. Configuración estructural del ensayo. Viga biapoyada con carga puntual en centro luz.

- *Ecuaciones:*

-Reacciones:

$$R_A = R_B = \frac{P}{2} \qquad \text{(Ec.4.1.1)}$$

-Esfuerzos:

-Cortante:

$$V_{AC} = \frac{P}{2} \qquad \text{(Ec.4.1.2)}$$

$$V_{CB} = -\frac{P}{2} \qquad \text{(Ec.4.1.3)}$$

-Momentos Flectores:

$$M_{AC} = \frac{P}{2}x \qquad \text{(Ec. 4.1.4)}$$

$$M_{CB} = \frac{P}{2}(L - x) \qquad \text{(Ec.4.1.5.)}$$

$$M_{max} = M_C = \frac{PL}{4} \qquad (Para\ x = \frac{L}{2}) \qquad \text{(Ec.4.1.6)}$$

A partir de la expresión del momento máximo (Ec.4.1.6) se estimará la carga máxima que podrá resistir la pletina, con el objetivo de aproximar el rango de las cargas que se le aplicarán. Una vez determinadas las cargas se estimará la flecha resultante con la expresión Ec.4.1.9 para posteriormente compararlas con las medidas por el sensor.

Los resultados específicos de los cálculos para el ensayo para la pletina a tratar se detallan más adelante en el capítulo 5 *"Ensayos realizados"* del presente documento.

-Flechas

A partir de la expresión de la elástica (Ec.4.1.7, Ec.4.1.8), es posible obtener la relación entre la flecha medida y la carga que actúa (Ec.4.1.10), esta será la expresión introducida en el programa *Processing* para su correspondiente visualización.

$$y_{x<\frac{L}{2}} = \frac{PL^2 x}{16EI}\left(1 - \frac{4}{3}\frac{x^2}{L^2}\right) \qquad \text{(Ec.4.1.7)}$$

$$y_{x>\frac{L}{2}} \frac{PL^2(L-x)}{12EI}\left(\frac{3}{4}-\frac{(L-x)^2}{L^2}\right) \quad \text{(Ec.4.1.8)}$$

$$y_{max} = y_C = \frac{PL^3}{48EI} \quad \text{(Ec.4.1.9)}$$

$$P = \frac{16EI*y}{L^2x\left(1-\frac{4x^2}{3L^2}\right)} \quad \text{(Ec.4.1.10)}$$

Siendo:

y: Flecha de la viga
P: Carga aplicada
L: Longitud total de la viga
x: punto de estudio de la viga
E: Módulo de Young del material
I: Inercia de la sección

-Tensión normal en la sección

$$\sigma = \frac{M*z}{Iy} \quad \text{(Ec.4.1.11)}$$

Siendo:

σ : Tensión admisible. Limite elástico
M: Momento máximo aplicado
z: Distancia de la fibra neutra al punto de estudio.
Iy: Inercia

La Ec.4.1.11 junto con la Ec.4.1.6 servirá para garantizar que las cargas aplicadas a la pletina no superen el límite elástico y por lo tanto que la sección no plastifique.

El valor de dicha tensión se determinará mediante un ensayo a tracción de la pletina, hecho en el laboratorio y que se define en el Capítulo 5 *"Ensayos realizados*".

4.2. CASO NO LINEAL. VIGA DE ACERO INOXIDABLE.

El caso de estudio de la viga de acero inoxidable, tiene una base matemática notablemente más complicada que el caso de acero al carbono convencional, estas expresiones han sido proporcionadas por la cotutora de este trabajo, Itsaso Arrayago y se especifican a continuación:

La configuración estructural que se considera en este caso es:

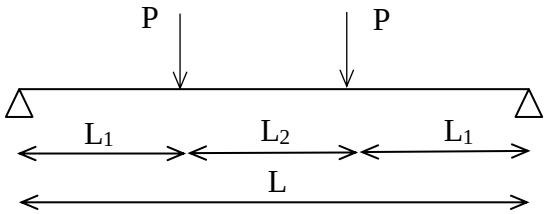


Fig.4.2.1. Configuración estructural de la viga a tratar. Viga de acero inoxidable

L= 1500 mm

L_1=L_2=L_3= 500 mm

- Función de las flechas

En la Ecuación 4.2.1 se presenta la relación momento-curvatura para secciones transversales de acero inoxidable, está basada en un método alternativo de cálculo de flechas, que tiene en cuenta la linealidad del material. Mediante la integración directa de dicha ley (Ec.4.2.2) y el ajuste posterior de las condiciones de contorno, se obtiene la componente inelástica de las flechas.

$$\chi(x) = \frac{M(x)}{EI} + \chi_p \left(\frac{M(x)}{M_{0.2}}\right)^{n-1} \qquad \text{(Ec.4.2.1)}$$

$$d = \iint \chi(x)dx \qquad \text{(Ec.4.2.2)}$$

La componente plástica de la curvatura puede obtenerse mediante las siguientes expresiones:

$$\chi_p = \frac{2}{H}\left(\frac{\sigma_{0.2}}{E} + 0.002\right) - \frac{M_{0.2}}{EI} \qquad \text{(Ec.4.2.3)}$$

donde H es la altura de la sección transversal, $\sigma_{0.2}$ y E son parámetros del material (límite elástico y módulo de elasticidad), I es la inercia y $M_{0,2}$ puede estimarse a partir de:

$$M_{0.2} = \sigma_{0.2} t(B-2t)(H-t) + H^3 \chi_{0.2} 2t \left(\frac{E}{12} - \frac{0.002 E \chi_{0.2} H}{32\left(\frac{\sigma_{0.2}}{E} + 0.002\right)^2} \right) \quad \text{(Ec.4.2.4)}$$

$$\chi_{0.2} = \frac{2}{H}\left(\frac{\sigma_{0.2}}{E} + 0.002\right) \quad \text{(Ec.4.2.5)}$$

El ensayo se realizará sobre una viga con las características definidas en la siguiente tabla:

Sección	H [mm]	B [mm]	t [mm]	E [MPa]	$\sigma_{0.2}$ [MPa]	n (6)
S4 – 200x100x3	100	200	3	192918	415	5.5

Para el caso de una viga biapoyada sometida a dos cargas puntuales la ley de momentos flectores se define según la Ecuación 4.2.6:

$$M(x) = \begin{cases} P \cdot x & x \leq 510mm \\ P \cdot L_1 & 510mm < x \leq 990mm \\ P \cdot (L-x) & x > 990mm \end{cases} \quad \text{(Ec.4.2.6)}$$

La expresión de la flecha puede obtenerse por tanto, tal y como se ha comentado, mediante la integración de la función de curvaturas definida en la Ecuación (4.2.1) con la ley de momentos flectores mostrada en la Ecuación (4.2.6).

Flecha para $x \leq L_1$

$$d = \frac{Px^3}{6EI} + \chi_p \left(\frac{P}{M_{0.2}}\right)^{n-1} \frac{x^{n+1}}{n(n+1)} + Cx \quad \text{(Ec.4.2.7)}$$

donde

$$C = \frac{-PL_1^2}{2EI} - \chi_p \left(\frac{P}{M_{0.2}}\right)^{n-1} \frac{L_1^n}{n} - (L/2 - L_1)\left[\frac{PL_1}{EI} + \chi_p \left(\frac{PL_1}{M_{0.2}}\right)^{n-1}\right] \quad \text{(Ec.4.2.8)}$$

[6]Con tal de simplificar la expresión final a introducir y resolver por el programa, se tomará n= 5.

La ecuación anterior es la que se empleará para la estimación de la carga P^* a partir de la flecha d^* medida, para un punto x^*, utilizando para ello un cero de funciones, ya que es recomendable que la medición de la flecha se realice en un punto situado entre el apoyo y la primera de las cargas puntuales aplicadas, por simplicidad de la ecuación en este tramo.

Flecha para $L_1 < x \leq L/2$

$$d = d_{L_1} + \theta_{L_1}(x - L_1) + \left[\frac{PL_1}{2EI} + \frac{\chi_p}{2}\left(\frac{PL_1}{M_{0.2}}\right)^{n-1}\right](x^2 - L_1^2) - (x - L_1)\left[\frac{PL_1^2}{EI} + \chi_p L_1\left(\frac{PL_1}{M_{0.2}}\right)^{n-1}\right] \quad \text{(Ec.4.2.9)}$$

donde

d_{L_1} es la flecha que corresponde al punto $x = L_1$, calculada en el tramo anterior y

θ_{L_1} es el giro que corresponde al punto $x = L_1$, calculado a partir de

$$\theta_{L_1} = \frac{PL_1^2}{2EI} + \chi_p\left(\frac{P}{M_{0.2}}\right)^{n-1}\frac{L_1}{n} + C \quad \text{(Ec.4.2.10)}$$

El parámetro C es el definido anteriormente.

Flecha para $L/2 < x \leq L - L_1$

Esta curva es la simétrica de la curva establecida para $L_1 < x \leq L/2$. Puede aprovecharse esta simetría o emplear la misma expresión cambiando x por $L - x$:

$$d = d_{L_1} + \theta_{L_1}(L - x - L_1) + \left[\frac{PL_1}{2EI} + \frac{\chi_p}{2}\left(\frac{PL_1}{M_{0.2}}\right)^{n-1}\right][(L-x)^2 - L_1^2] - (L - x - L_1)\left[\frac{PL_1^2}{EI} + \chi_p L_1\left(\frac{PL_1}{M_{0.2}}\right)^{n-1}\right] \quad \text{(Ec.4.2.11)}$$

Los parámetros C, d_{L_1} y θ_{L_1} son los mismos que se han definido antes.

Flecha para $x > L - L_1$

Esta curva es la simétrica de la curva establecida para $x \leq L_1$. Puede aprovecharse esta simetría o emplear la misma expresión cambiando x por $L - x$:

$$d = \frac{P(L-x)^3}{6EI} + \chi_p\left(\frac{P}{M_{0.2}}\right)^{n-1}\frac{(L-x)^{n+1}}{n(n+1)} + C(L-x) \quad \text{(Ec.4.2.12)}$$

Siendo el parámetro C el mismo que se ha definido anteriormente.

Sustituyendo las diferentes expresiones expuestas y con tal de obtener una ecuación δ(P) para introducirla al programa y obtener la correspondiente resolución del cero de funciones, se tiene:

- Para $x \leq L_1$ (Ec.4.2.13)

$$\delta = 4.9838 * 10^{-36} P^4 x^6 - 3.27065 * 10^{-21} P^4 x + 2.667 * 10^{-13} P x^3 - 4 * 10^{-7} P x$$

Para x=500mm se tiene: (Ec.4.2.14)

$$\delta = -1.557 * 10^{-18} P^4 - 1.66 * 10^{-4} P$$

- Para $L_1 < x < \frac{L}{2}$ (Ec.4.2.15)

$$\delta = 4.67 * 10^{-24} P^4 x^2 - 7.94 * 10^{-21} P^4 x + 1.24715 * 10^{-18} P^4 + 4 * 10^{-10} P x^2 - 6 * 10^{-7} P x + 3.4 * 10^{-5} P$$

Las ecuación 4.2.14 será introducida en el programa con el fin de calcular la carga correspondiente a la flecha medida por el sensor, que estará colocado en x=500mm.

Las ecuaciones 4.2.14 y 4.2.15 serán las expresiones que aproximaran la curva de la deformada en los demás puntos de la viga, donde se creará una simetría para los puntos correspondientes a $L/2 < x \leq L - L_1$ y $x > L - L_1$.

CAPÍTULO 5: ENSAYOS REALIZADOS

5.1. ENSAYO DEL MODELO REDUCIDO

5.1.1. DETERMINACIÓN DE PARÁMETROS

5.1.2.

El ensayo que se describirá a continuación servirá a modo de calibración del primer programa realizado y, ayudará a familiarizarse con la recogida de datos, ver su fiabilidad y en general comprobar el funcionamiento, los cálculos hechos por el programa y los resultados obtenidos.

En primer lugar es necesario determinar los parámetros y condiciones de contorno con los que contará el ensayo. El modelo reducido se llevará a cabo con una pletina de acero que estará simplemente apoyada.

La pletina tiene una longitud total aproximada de 1.7 m, por lo que el primer paso es decidir la longitud exacta que tendrá la viga ensayada y la flecha que se querrá alcanzar, para posteriormente acotar los pesos con los que se irá cargando. Cabe destacar que el ensayo se diseña sólo para el aumento de carga, no se considera la posibilidad de descarga de la pletina.

Las características geométricas y mecánicas de la pletina son las siguientes:

Ancho: 20mm
Espesor: 4mm
Módulo elástico: 210.000 MPa
Límite elástico: 343.75 N/mm^2 (determinado mediante ensayo a tracción)
Inercia: $I = \frac{h^3 b}{12} = \frac{4^3 20}{12} = 106'67 mm^4$

Para determinar las condiciones de contorno del ensayo, la primera idea era aplicar como condición de diseño la flecha máxima que se quería alcanzar, asemejando el ensayo a escala reducida al real. Gracias a cálculos previos facilitados por el tutor se estima que la flecha máxima alcanzada en la viga de acero inoxidable será de 200mm.

Haciendo algunos cálculos previos, se vio que para alcanzar una flecha de 200 mm era necesario que la luz de la viga fuera de 2 m. Dado que la pletina disponible media 1.7m, se decidió cambiar la condición de diseño fijando el valor de la longitud en vez de la flecha. Así pues, imponiendo una longitud de 1500 mm, en primer lugar se determinará a qué momento máximo admisible puede estar sometida la pletina con tal de cumplir la

tensión máxima admisible (Ec.5.1.1) para finalmente determinar cuál será la carga máxima aplicada (Ec.5.1.2), sus incrementos y la flecha que se alcanzará (Ec.5.1.3).

- *Cálculos realizados:*

$$\sigma = \frac{M*z}{Iy} \qquad \text{(Ec.5.1.1)}$$

$$M_{max} = \frac{235\,\frac{N}{mm^2}*2mm}{106.67\,mm^2} = 12533.72\,N*mm$$

$$M_{max} = \frac{PL}{4} \qquad \text{(Ec.5.1.2)}$$

$$\boldsymbol{P_{max}} = \frac{4*12533.72\,N*mm}{1500\,mm} = \mathbf{33.42}\,N$$

$$\boldsymbol{m} = P/g = \frac{33.42\,N}{9.8\frac{m}{s^2}} = \mathbf{3.4\ Kg}$$

$$y_{max} = y_C = \frac{PL^3}{48EI} \qquad \text{(Ec.5.1.3)}$$

$$\boldsymbol{y_{max}} = y_C = \frac{33.42*1500^3}{48*210000*106.67} = \mathbf{104.9\ mm}$$

Aunque el ensayo a tracción realizado – que se explica con más detalle en este mismo capítulo– haya resultado un límite elástico de 343.75 N/mm^2 se han hecho los cálculos con el valor de 235 N/mm^2 para asegurar en todo momento que la pletina se encuentra en la zona elástica y que en consecuencia, las hipótesis establecidas son correctas.

Una vez calculado el peso máximo dada la longitud estipulada y cumpliendo la tensión máxima admisible, resultará una flecha de 105 mm –flecha teórica–.Seguidamente se determinarán los incrementos de peso, con el objetivo de que la viga se cargue progresivamente. Esto se ha decidido debido a la gran flexibilidad de la pletina, para no cargarla de forma brusca y de esta forma acercarse más a las condiciones del ensayo a gran escala.

Se ha estipulado que dichos incrementos sean respecto a la deformación y resulten aproximadamente de 1 cm, consiguiendo así 10 escalones de carga, incrementando el peso en **340 gramos** por cada escalón[7].

[7]En un primer momento se aproximó los incrementos de peso a 350 gr, pero al comprobar que el recipiente vacío utilizado para cargar la viga, tenía un peso propio de 680gr se decidió imponer que los incrementos fueran de 340 gr para que entonces el recipiente vacío fuese el segundo escalón de carga, correspondiente a 680 gr.

5.1.2. REALIZACIÓN DEL ENSAYO

Una vez determinadas las condiciones de contorno, parámetros de longitud, carga y flecha se procede al montaje del ensayo.

Este ensayo se realiza en el laboratorio pequeño colindante al laboratorio de estructuras de la *Escola de Camins, Canals i Ports* de la Universidad Politécnica de Catalunya.

El material que se ha usado es el siguiente:

-Pletina de 1.70 m de longitud y sección 20 mm x 4mm, con gancho regulable a lo largo de la longitud de la viga, para la colocación de los pesos.
-Recipientes vacíos a colgar de la viga.
Se pone en la zona inferior del recipiente una base rígida para asegurar una mayor superficie de rebote de las ondas ultrasónicas con el objetivo de minimizar los errores de medida.
-Granalla de plomo a introducir en los recipientes para cargar la pletina progresivamente.
-Balanza con precisión de +/- 10 gramos.
-Embudo.
-Cinta métrica.
-Ordenador portátil.
-Cámara fotográfica.
-Sensor de ultrasonido HC-SR04.
-Placa Arduino Mega 2560.

- *Ensayo a tracción*

En primer lugar, para contar con información y características reales de los materiales utilizados, se ha realizado un ensayo a tracción de un tramo de la pletina, con tal de asegurar que el límite elástico es igual o superior al considerado.

Este ensayo se hace colocando la probeta de acero de sección 20mm x 4mm en la prensa que se muestra en la figura 5.1.2.1. Esta máquina cuenta con un dispositivo encargado de producir la carga a tracción y una banda extensométrica que mide las deformaciones que sufre la probeta al aplicarle la carga.

Se decide imponerle incrementos de carga de 250 Kg. De esta forma, se ha cargado con dichos intervalos hasta su límite elástico correspondiente a 2750 Kg (fy =343.75 N /mm^2). Una vez llegado al límite elástico se sigue aplicando carga hasta llegar a su límite plástico

que se halló en 2950 Kg de carga (368.75 N/mm^2) y finalmente su límite último con 3950 Kg de carga (fu=493.75 N/mm^2).

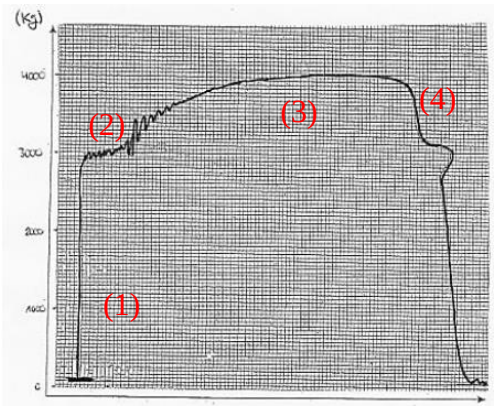
De esta forma, se comprueba que el límite elástico considerado en los cálculos –235 N/mm^2– es inferior al real, por lo que las hipótesis de elasticidad establecidas son válidas en el rango de trabajo.

Fig. 5.1.2.1. Máquina de ensayo a tracción

Fig. 5.1.2.2. Probeta ya ensayada

Se ensayó la probeta para que plastificara pero sin llegar a rotura, como se muestra en la figura 5.1.2.2, donde se ve claramente el estrechamiento de la sección a causa de la tracción sufrida, donde el alargamiento consecuente fue de 8.4 cm.

En este caso solo interesaba conocer el límite elástico de la probeta, pero este tipo de ensayo proporciona además, otros parámetros físicos como el módulo elástico, el coeficiente de Poisson, las deformaciones longitudinales y transversales así como la gráfica tensión deformación correspondiente.



A continuación se muestra, para el mayor entendimiento de los resultados y el ensayo, una gráfica tipo – no correspondiente al ensayo hecho– La gráfica corresponde a la relación carga-alargamiento, donde cada 2 cm del eje vertical corresponde a 1000 Kg de carga, el eje horizontal es a escala real.

Fig. 5.1.2. 3. Gráfica tipo resultante de un ensayo a tracción.

Se observan pues, cuatro zonas claramente diferenciadas; la elástica (1), la zona de fluencia (2), la plástica (3) y el punto de rotura (4). Después del punto de rotura se observa un salto que se debe a que la plumilla que la dibuja es mecánica y ésta se desvía al romperse la probeta.

- *Ensayo de la viga*

Una vez hecho el ensayo a tracción y realizados los cálculos y comprobaciones correspondientes, se prosigue a la colocación de la pletina a ensayar entre dos mesas, asegurando la longitud estipulada y comprobando que el gancho de cuelgue está en el medio de ésta, así como la correcta alineación de la viga entre las mesas para la minimización de errores.

También se prepara el ordenador portátil conectando la placa Arduino y el sensor, confirmando su correcta colocación debajo del gancho, y dentro de un recipiente de plástico para una mejor recogida de la granalla de plomo en caso de derramarse.

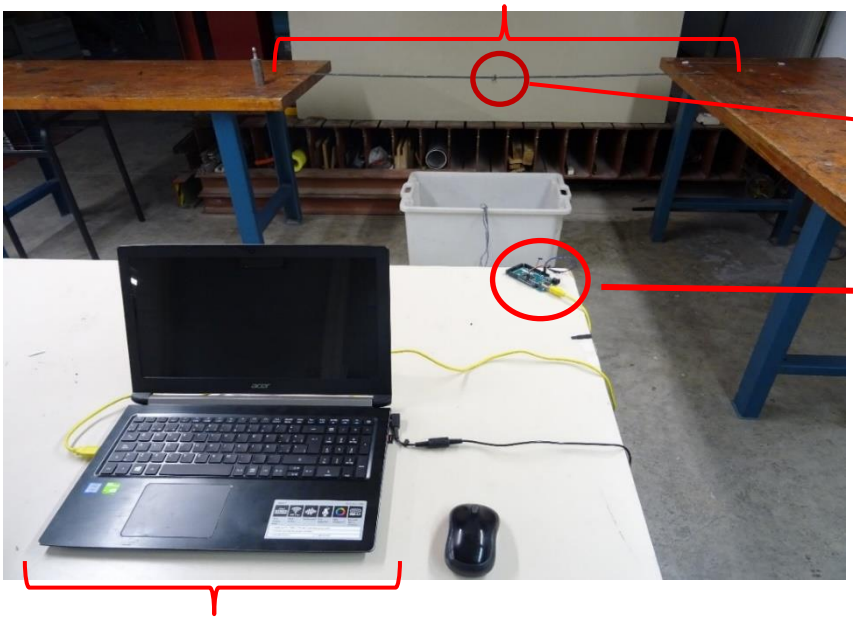


Fig. 5.1.2.4. Disposición de los elementos del ensayo

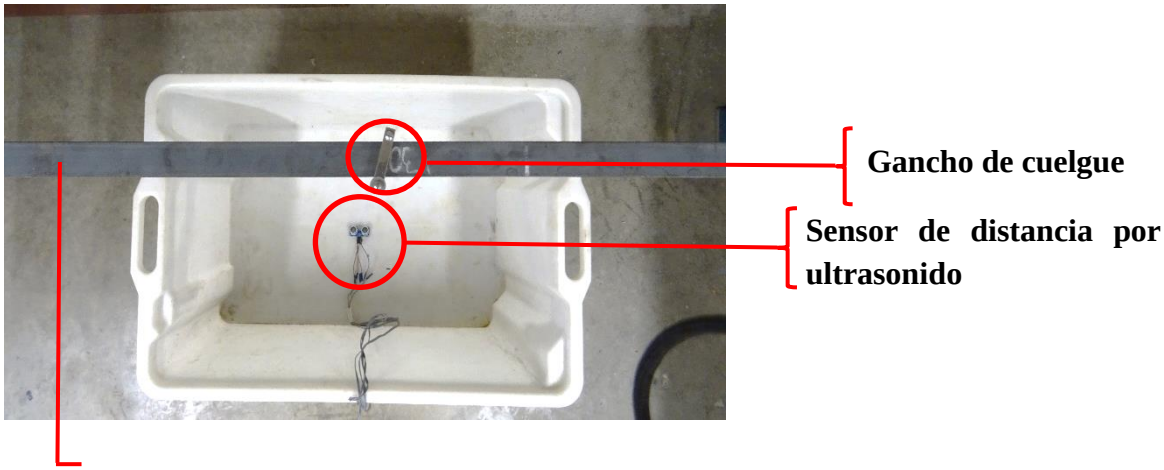


Fig. 5.1.2. 5. Disposición de los elementos del ensayo

A continuación, se prosigue a la preparación de los 10 pesos de 340 gr cada uno, con los que se cargará la viga. Esto se hace en primer lugar tarando los recipientes mediante la balanza e introduciendo la granalla de plomo hasta llegar al peso deseado.

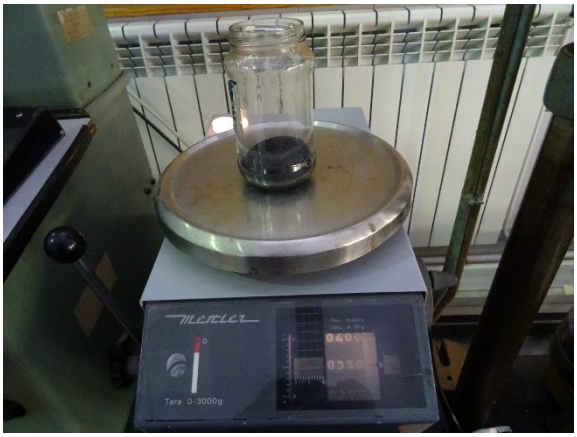


Fig. 5.1.2.6. Preparación de los pesos.

Una vez preparado todo el material, se prosigue a calibrar el sensor, hasta la estabilización de la medida de la distancia. Esto se comprueba a través de la impresión por pantalla de Arduino y comparándolo con la medida a mano con la cinta métrica.

La distancia inicial del sensor a la viga (h), se considera 600 mm, dato que se introduce en el programa para que durante el ensayo, a la distancia inicial (h) se le reste la medida del sensor (d'), para así obtener la flecha en cada momento (d).

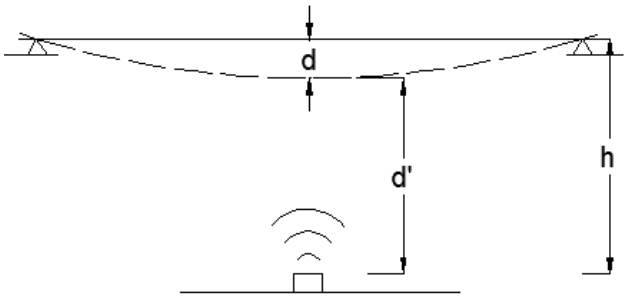


Fig. 5.1.2.8. Esquema de las distancias de la distancia medida por el sensor.

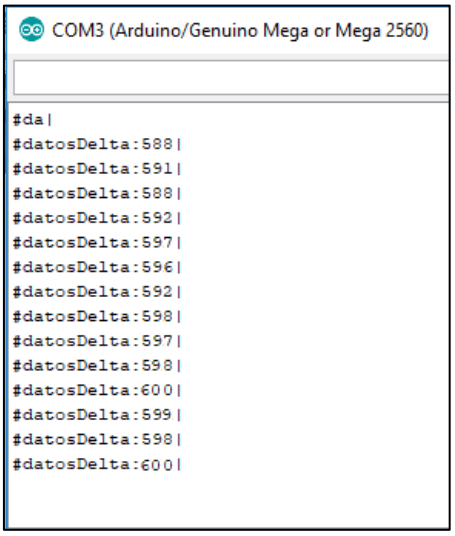


Fig. 5.1.2.7. Impresión de las medidas de distancia por pantalla.

Se coloca la cámara fotográfica para hacer fotografías de cada escalón de carga para introducirlas en el programa y que éstas vayan asociadas a la flecha correspondiente[8]. Una vez comprobada la correcta asociación de imágenes con las medidas, se prosigue a empezar el ensayo

Fig. 5.1.2. 9 Fotografías iniciales del ensayo

[8] Este ensayo se realizó tres veces, una para hacer las fotografías y otras dos para obtener los resultados.

5.1.3. RESULTADOS OBTENIDOS

A continuación se muestran algunas capturas de pantalla del resultado del ensayo con diferentes escalones de carga. Las diferentes medidas que aparecen en el primer cuadrante son consecuencia de la estabilización de la medida.

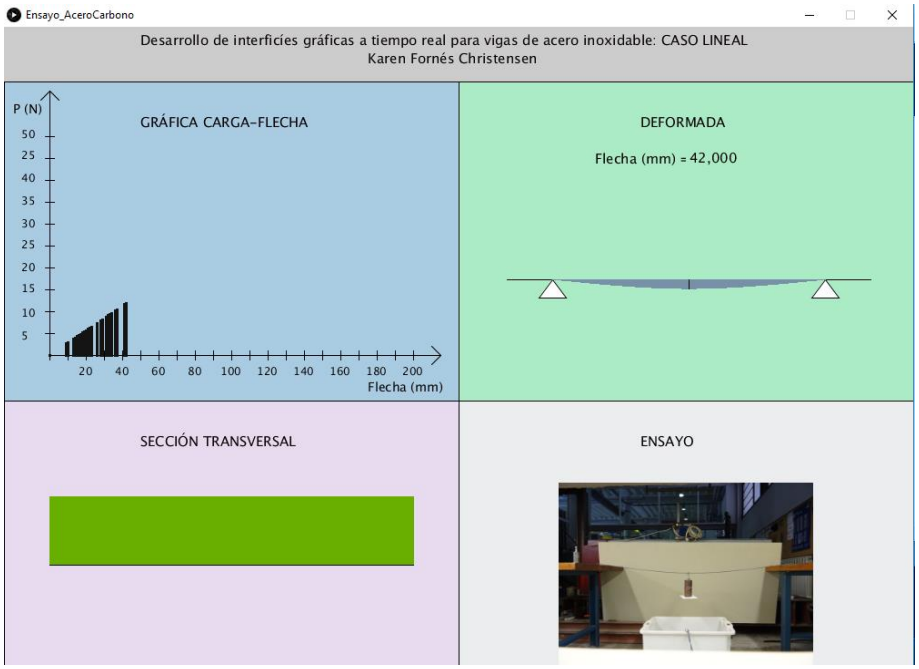


Fig.5.1.3. 1. Muestra del escalón de carga 4, con un peso de 1.38 Kg y flecha de 42mm.

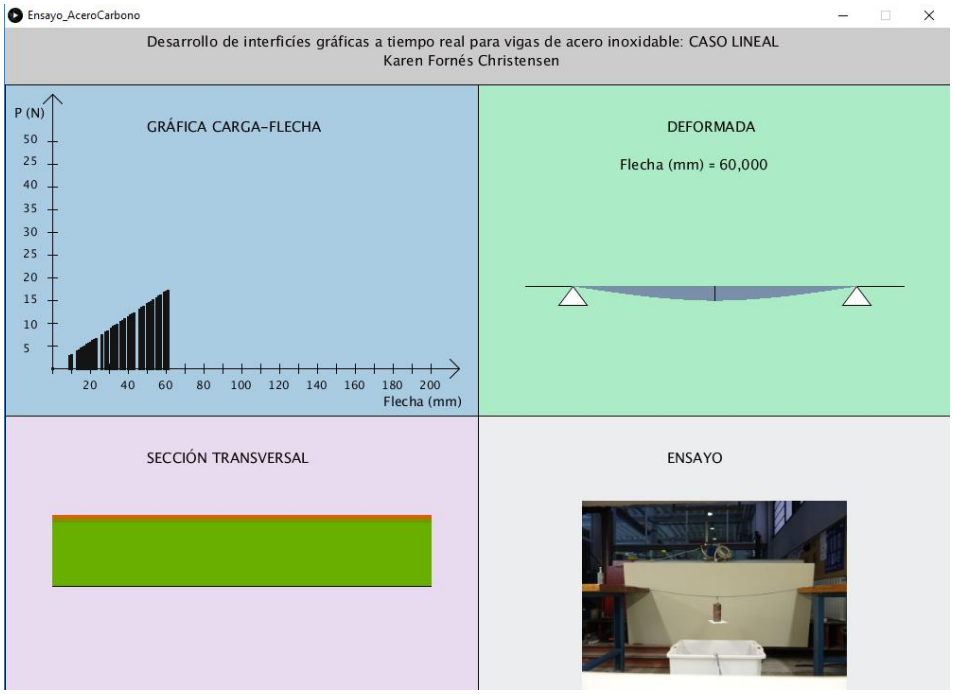


Fig.5.1.3. 2. Muestra del escalón 6, con 2.08 Kg y flecha de 60 mm

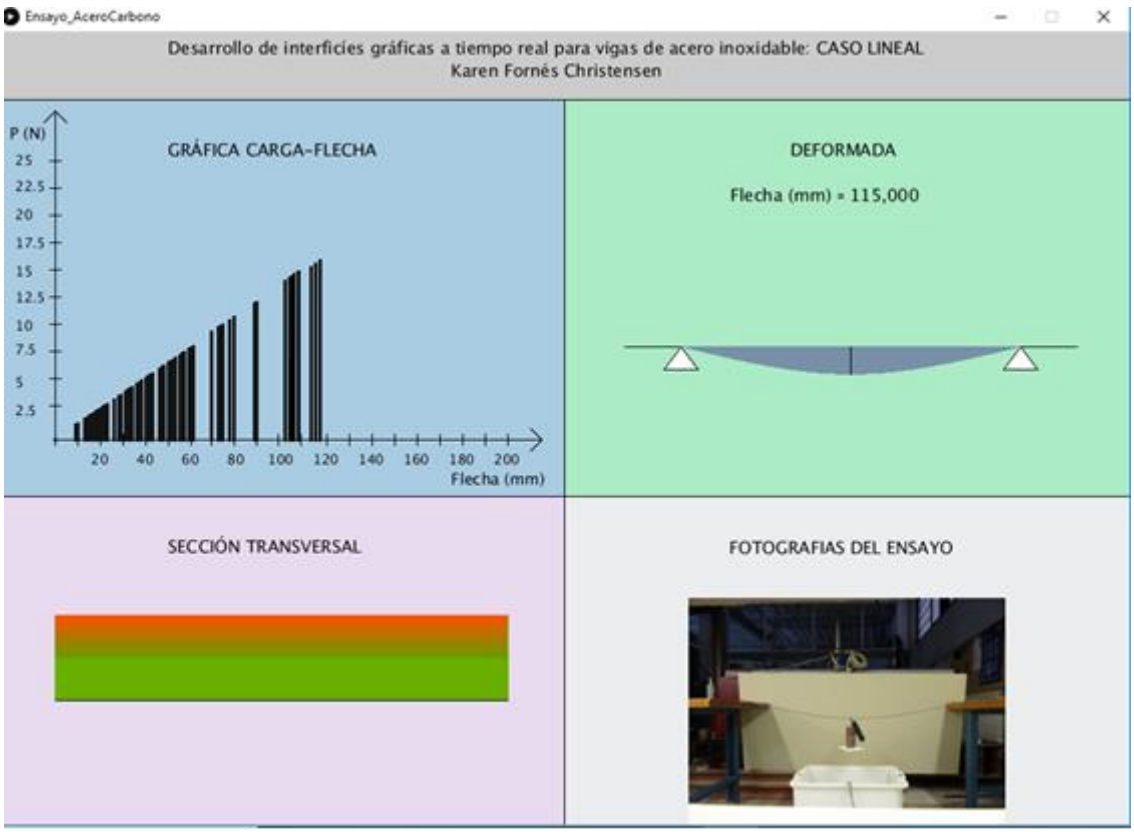


Fig.5.1.3. 3. Último escalón de carga, con un peso final de 3.78 Kg

A continuación se muestra el cuadro con los resultados obtenidos. El primer escalón de carga corresponde a un recipiente vacío de peso 340 gr, y el segundo corresponde al peso del recipiente vacío -más grande que el primero- usado a lo largo del ensayo. Como éste sin granalla pesaba 680 gr se decidió usarlo como segunda carga, y a partir de ahí cargarlo gradualmente en escalones de 340 gr, tal y como estipulaban los cálculos.

Escalón de carga	Incrementos de masa (kg)	Carga (N)	Flecha teórica (mm)	Flecha medida (mm) Ensayo 1	Flecha medida (mm) Ensayo 2	Diferencia entre calculado y medido. Ensayo 1	Diferencia entre calculado y medido. Ensayo 2
	0,000	0	0,00	0	0	0,00	0,000
1	0,340	3,332	10,46	12	11	1,54	0,541
2	0,680	6,664	20,92	22	21	1,08	0,083
3	1,020	9,996	31,38	30	32	-1,38	0,624
4	1,360	13,328	41,83	42	42	0,17	0,165
5	1,700	16,66	52,29	49	50	-3,29	-2,293
6	2,040	19,992	62,75	60	60	-2,75	-2,752
7	2,420	23,716	74,44	72	70	-2,44	-4,441
8	2,760	27,048	84,90	82	78	-2,90	-6,900
9	3,100	30,38	95,36	91	90	-4,36	-5,358
10	3,440	33,712	105,82	110	109	4,18	3,183
11	3,780	37,044	116,28	112	115	-4,28	-1,276

Fig.5.1.3. 4 Tabla de cálculos y resultados obtenidos

De esta forma los intervalos de carga se han distribuido de la siguiente manera:

-Escalón 1: 0.340 Kg. Recipiente pequeño vacío.

-Escalón 2: 0.680 Kg. Recipiente grande vacío

-Escalón 3: 1.020 Kg. Recipiente grande + 0.340 Kg

-Escalón 4 al 8: Escalón anterior + 0.340 Kg

-Escalón 9: Recipiente grande lleno + recipiente pequeño vacío.

-Escalón 10: Recipiente grande lleno + recipiente pequeño con 0.34 Kg

-Escalón 11: Recipiente grande lleno + recipiente pequeño con 0.68 Kg.

En los últimos tres escalones de carga se usó un segundo recipiente dado que el volumen del recipiente principal no abarcaba toda la granalla correspondiente a los 3.76 Kg. Esta entre otras, sería una posible futura mejora, dado que la fiabilidad del ensayo se puede ver afectada por el hecho de que el segundo recipiente interactuaba con el primero, como consecuencia de colgar el uno al lado del otro.

Finalmente se comprueba que las flechas teóricas calculadas con la formulación especificada anteriormente, y la medida por el sensor, así como el cálculo de la carga son suficientemente parecidas.

Se puede observar como los errores máximos medidos se dan en los escalones 8, 9 y 10 – con errores cercanos a medio centímetro- que se atribuyen a la mayor inestabilidad que sufría el sistema por estar muy cargado y a la interacción de los dos recipientes, que en consecuencia aumentaba el tiempo de estabilización y un leve cambio de posición de la base del recipiente correspondiente al lugar de rebote del sensor.

Los demás resultados se consideran dentro de un rango de error muy aceptable –entre 0.1 y 2 mm de diferencia con los cálculos teóricos – que se atribuye sobre todo a que el sensor utilizado no es de alta precisión, así como algunos posibles errores asociados al propio ensayo –tiempo de estabilización –.

5.2. PROTOTIPO A GRAN ESCALA. VIGA DE ACERO INOXIDABLE

Una vez realizado y verificado el Digital Twin a escala reducida correspondiente a una pletina de acero al carbono, se prosigue a la comprobación del prototipo de gemelo digital para ser desarrollado en un ensayo de vigas de acero inoxidable.

Esta comprobación se hará a mano, verificando la correcta recibida de datos, cálculo de la carga P asociada y visualización a tiempo real.

5.2.1. CARACTERÍSTICAS DE LA VIGA

El modelo de la interfaz a gran escala será un prototipo para la aplicación en un ensayo a flexión de una viga real de acero inoxidable. Dicho acero será austenítico de tipo 1.4301, que cuenta con características específicas de alta tenacidad, resistencia a atmosferas corrosivas, brillo superficial, además de unas buenas propiedades de soldabilidad. Es de uso adecuado, entre otras, para aplicaciones o atmósferas urbanas.

De forma general, su composición química es de menos de 0.12 % de carbono, entre 16 y 18 % de cromo y entre 7-8 % de níquel.

La sección a tratar es hueca rectangular con las siguientes características:

Sección	H [mm]	B [mm]	t [mm]	E [MPa]	$\sigma_{0.2}$ [MPa]
S4 – 200x100x3	100	200	3	192918	415

Esta sección forma parte de las secciones a ensayar dentro del proyecto BIA2016-75678-R, AEI / FEDER, UE "Comportamiento estructural de pórticos de acero inoxidable. Seguridad frente a acciones accidentales de sismo y fuego ".

La configuración estructural tratada es una viga biapoyada con dos cargas puntuales a aproximadamente $\frac{L}{3}$ y $\frac{2L}{3}$. La longitud nominal de la viga es de 1700 mm, preparada para ser ensayada con una luz de 1500mm, contando así, con dos voladizos de 100mm a cada lado de los apoyos. Las cargas estarán aplicadas a 510 mm de cada punto de apoyo, resultando una separación entre ellas de 480 mm.

De esta forma, las condiciones de contorno en este caso, a diferencia del caso lineal, están definidas de un inicio, con lo que, el objetivo será aplicar los conocimientos de funcionamiento y programación adquiridos en el Digital Twin correspondiente al modelo reducido, para adaptar el código a este caso en concreto.

Por simplicidad de la realización del código y, dada la complejidad de las expresiones que definen el comportamiento del acero inoxidable, se han incluido implícitamente dichas expresiones, correspondientes a la configuración estructural y a las características de la sección concreta de estudio, en el programa prototipo.

5.2.2. FUNCIONAMIENTO DEL ENSAYO REFERENCIA

El ensayo referencia, con el que se ha creado el prototipo de Digital Twin, consiste en someter la viga de estudio a dos cargas puntuales a un y dos tercios de la luz. Dichas cargas se aplican mediante una viga cargadero cuya transmisión de fuerza proviene de un actuador hidráulico con capacidad de hasta 1000 KN. La viga ensayada estará apoyada en una estructura auxiliar con tal de obtener una altura adecuada de la misma.

Con tal de asegurar que la configuración estructural tratada se mantiene durante el ensayo, se colocan estabilizadores laterales para evitar una eventual caída lateral de la viga, también se incorporan rótulas en los apoyos para asegurar su libre movimiento – configuración de viga biapoyada – y piezas de neopreno en los puntos de aplicación de las cargas con tal de evitar el contacto acero-acero entre la viga cargadero y la viga ensayada.

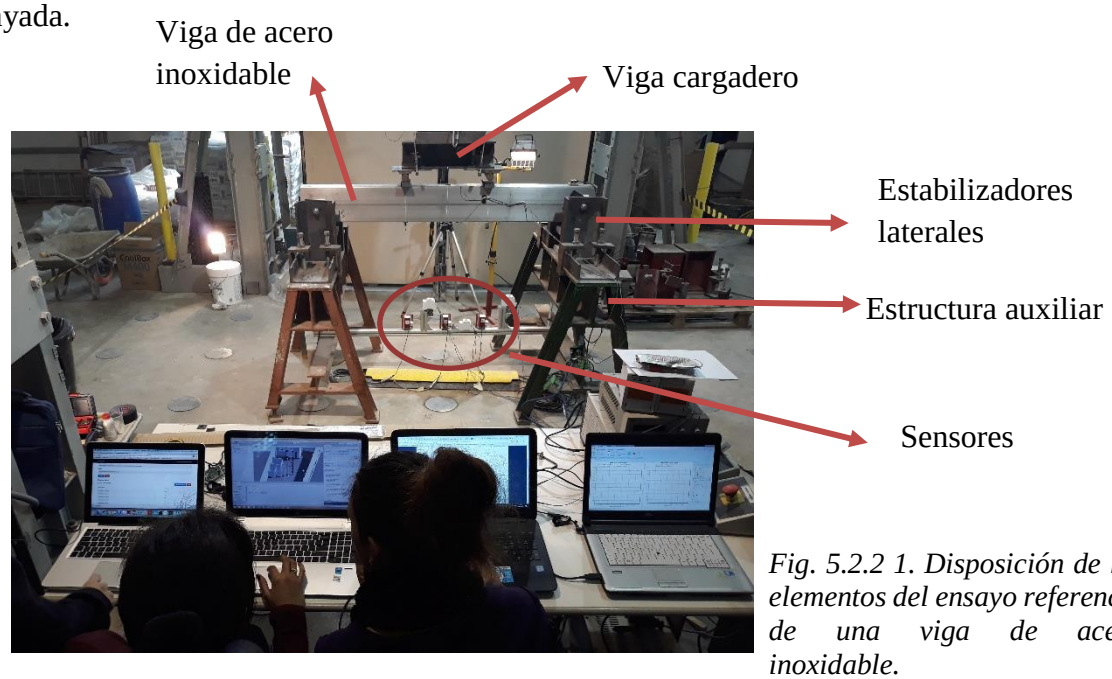


Fig. 5.2.2 1. Disposición de los elementos del ensayo referencia de una viga de acero inoxidable.

Con tal de evitar el fenómeno de inestabilidad o fallo por abolladura producida en elementos verticales o el alma, bajo cargas concentradas ("web crippling"), se colocan piezas de madera en el interior de la sección hueca.

El ensayo se hace realizando un control de desplazamiento en vez de carga, con el objetivo de observar el comportamiento de la viga después de alcanzar la carga máxima.

Según cálculos anteriores se estima que la carga última será alrededor de 100 KN con una flecha de 200mm, estos valores servirán a modo de comprobación en el momento de realizar la prueba del gemelo digital prototipo, para asegurar el correcto resultado y fiabilidad de los cálculos.

5.2.3. COMPROBACIÓN Y RESULTADOS OBTENIDOS DEL PROTOTIPO

- *Comprobación del prototipo*

La comprobación del prototipo se llevará a cabo manualmente, bajo condiciones suficientemente estables, para asegurar la fiabilidad de los resultados obtenidos.

Se decide, igual que en el caso lieal, y debido a las condiciones del ensayo referencia, adoptar la deformación como medida de control. De esta forma y, con incrementos sucesivos de aproximadamente un 1cm de flecha, se comprueba la correcta visualización y cálculos de los datos.

Dicha comprobación se ha hecho colocando el sensor debajo de una estructura auxiliar y, con un elemento rígido y de superficie plana – para la minimizar el error en la detección de la distancia – se ha ido disminuyendo sucesivamente la distancia entre el elemento rígido y el sensor, asemejando el incremento de deformación que sufre la viga en el ensayo.

La distancia inicial entre el sensor y la estructura auxiliar, una vez estabilizado el sensor de distancia por ultrasonido, ha sido de 400mm. Este valor ha introducido en el programa con tal de restarle, a la distancia inicial (h), la distancia medida en cada instante (d') y así obtener el semejante a la flecha de la viga (h).

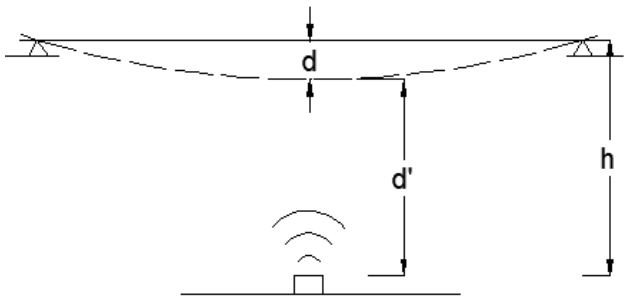


Fig. 5.2.3.1. Esquema de distancias a introducir en el programa

- *Resultados obtenidos*

Los resultados obtenidos de la comprobación se muestran a continuación.

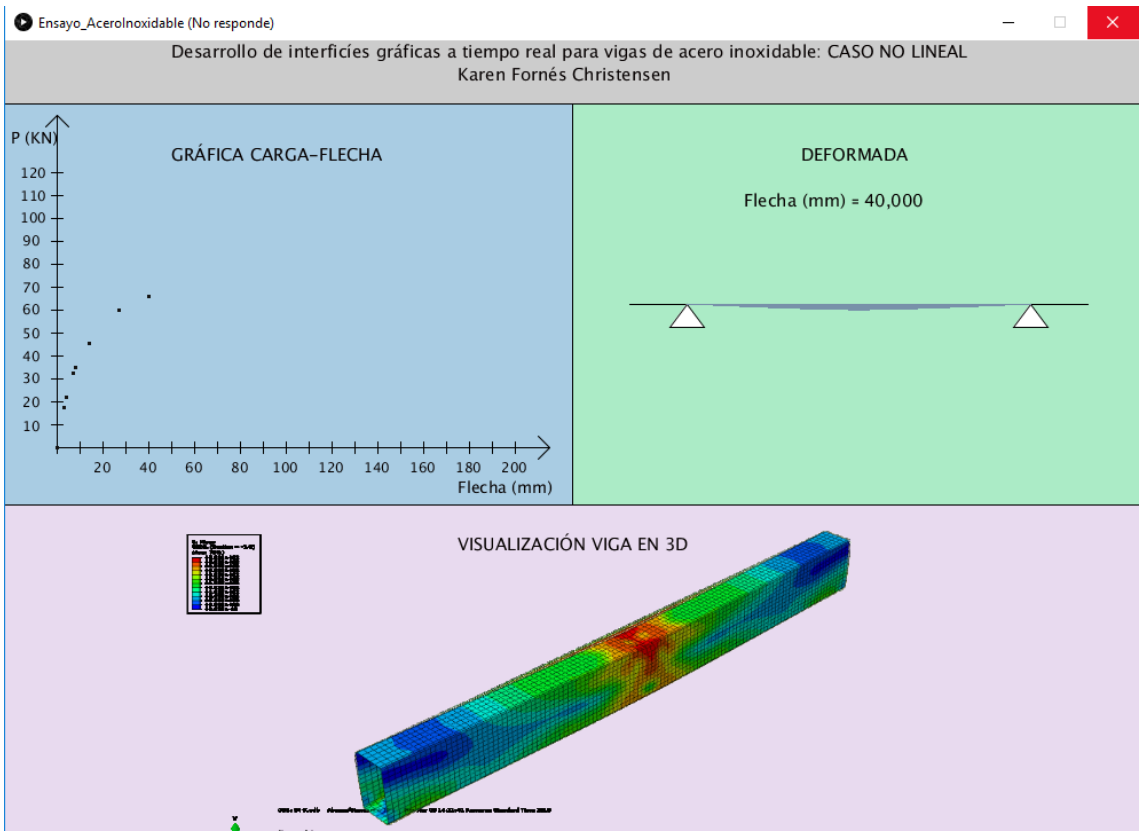


Fig. 5.2.3 2. Muestra del resultado correspondiente al caso no lineal, con una flecha de 40 mm.

En la figura se muestra los incrementos de carga-flecha hasta llegar a una deformación de 40 mm, así como la deformada y la visualización en 3D de la viga asociada a la deformación correspondiente.

En este ejemplo la gráfica se dibuja con puntos, aunque el código también incluye la opción de dibujarla con líneas –como en el caso lineal mostrado –.

Se ha reducido la escala del dibujo de la deformada para que a flecha máxima estuviera suficientemente proporcionada, así pues, para valores iniciales el dibujo es más reducido.

Comprobando por pantalla, las impresiones de los valores de distancia, carga e imagen asociada, se verifican los resultados:

Distancia medida: 40mm
Carga calculada: 6743.25 N = 67.43 KN
Imagen asociada: nº 40, asociada a la deformación de 40 mm.

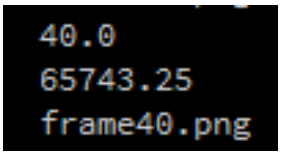


Fig. 5.2.3 3. Ejemplo de impresión de los resultados

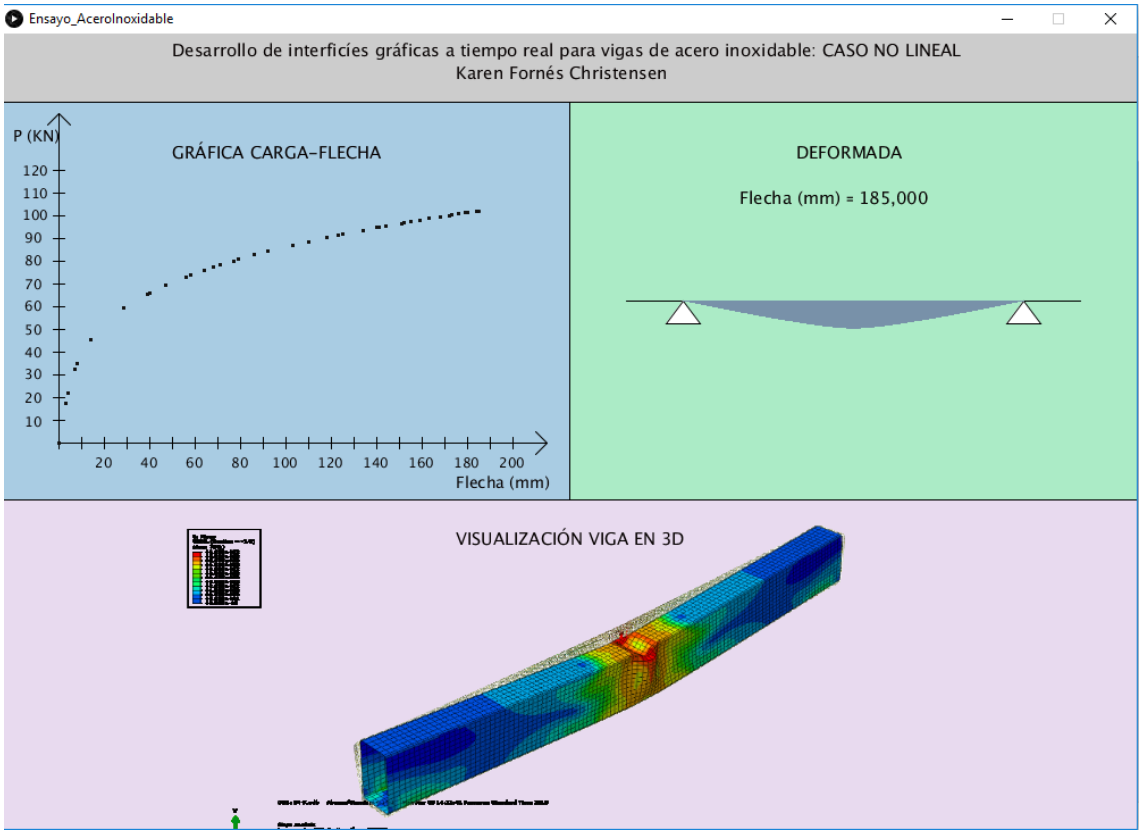


Fig. 5.2.3 4. Ejemplo de la visualización de la interfaz correspondiente al caso no lineal, con flecha de 185 mm

Llegando a los valores limites, y con una visualización general de la comprobación del ensayo, se observa claramente el comportamiento no lineal y redondeado de la gráfica carga-flecha correspondiente a una viga de acero inoxidable. Asimismo se muestra la deformada de la viga, creada a partir de las expresiones correspondientes.

Mientras se lleva a cabo la comprobación, se ve por la pantalla inferior del entorno de desarrollo de Processing, las impresiones de los valores deseados (Fig. 5.2.2.3.). En este caso, y para confirmar que los valores son los correctos, así como la graduación de los ejes, se han visualizado los valores de la flecha medida, la imagen asociada a esta y el valor de la carga calculada.

Se comprueba por pantalla, la correcta recibida de datos, cálculo de la carga P e imagen asociada:

Distancia medida: 185mm
Carga calculada: 101931.05 N = 101.93 KN
Imagen asociada: nº 185 correspondiente a deformación de 185 mm

Fig. 5.2.3 5. Ejemplo de impresión por pantalla de los resultados.

Dados los resultados, se puede concluir que bajo las hipótesis establecidas y, con la información disponible, los resultados tanto gráficos, como los cálculos realizados por el programa son correctos. Asimismo la visualización y progreso de la interfaz gráfica con sus correspondientes cálculos y dibujos constituyen un modelo digital a tiempo real.

CAPÍTULO 6: CONCLUSIONES

6.1. CONCLUSIONES DEL TRABAJO

Analizando los objetivos marcados en el inicio de este trabajo, en líneas generales, se han cumplido satisfactoriamente. Se han ampliado destacablemente los conocimientos en electrónica y programación, pudiendo crear y completar la interfaz gráfica con un resultado correcto, fiable de las medidas y valores de carga, así como comprobar la aplicabilidad del concepto Digital Twin en los diferentes campos estudiados.

- *Ejemplo práctico*

El objetivo principal de este trabajo, que era la realización de un Digital Twin aplicado a la tecnología de estructuras, ha dado unos resultados suficientemente buenos. Si bien la visualización puede ser más detallada, los resultados obtenidos, tanto en el modelo a escala reducida como en el prototipo para la aplicación a vigas reales de acero inoxidable, han sido fiables en medición, cálculo y sincronización del modelo real, parte fundamental del concepto Digital Twin.

- *Digital Twin*

Al iniciar el trabajo, parecía ser relativamente complicado aplicar el concepto de Digital Twin en un sector como el de la construcción pero, rápidamente, se hizo notar el gran potencial que dicho concepto supone en un campo tan amplio, complejo y dinámico como el de la obra civil. Durante el transcurso del trabajo se ha visto y entendido que, aunque el concepto abarca amplias ambiciones, es algo que puede incorporarse de forma relativamente fácil y completarlo a medida que se quiera y en el grado que se desee, consiguiendo un modelo digital o Digital Twin completo. Es decir, no es una idea que requiera una aplicación masiva directa, si no que poco a poco se puede instaurar en los modos de trabajo, sea en el diseño, planificación o ejecución de una estructura o construcción, concluyendo así su total aplicabilidad.

- *Acero inoxidable*

Otro aspecto que ha aportado la realización de esta tesina han sido los conocimientos respecto a un material estructural tan interesante como el acero inoxidable. Durante los estudios de grado no se profundiza en dicho material y los conocimientos iniciales sobre sus características y comportamiento eran nulos. A lo largo del transcurso del trabajo se ha comprobado que es un material más usado estructuralmente de lo pensado y, sobre

todo que, gracias a sus características, tiene un potencial muy elevado digno de incluirse más y más en el diseño y ejecución de estructuras.

- *Arduino y Processing*

Respecto a la realización propiamente de la interfaz fue bastante complicado hacer el código relativo a, sobre todo, la deformada de la viga. Esto fue principalmente debido a que, pese a que la plataforma de programación Processing es de uso relativamente sencillo, su aplicación en resoluciones técnicas y métodos numéricos es más complicada si se empieza con un nivel básico.

Así pues, es fácil conseguir la visualización general de formas y elementos simples, pero dotar dichos elementos de un comportamiento físico real siguiendo resoluciones de expresiones complejas, es más difícil. Esto viene dado por la necesidad de inclusión de librerías específicas de métodos numéricos, que no son abundantes en el entorno de Processing, siendo además la información disponible poco extensa. Conseguir pues, que se calculara y dibujara de forma suficientemente fiel la deformada – sobre todo de la viga de acero inoxidable – ha sido de lo más difícil de este trabajo.
A pesar de lo expuesto, se quiere finalmente concluir que el uso de plataformas como Arduino y Processing son altamente recomendables para la introducción a la programación y la creación de proyectos interactivos, poniendo a alcance de todo el mundo la posibilidad de crear proyectos propios con la satisfacción y practicidad que eso supone. Estas plataformas y su utilización pueden abrir un mundo de posibilidades a estudiantes, profesores y técnicos de cualquier ámbito acercando el mundo de la programación y la tecnología de bajo coste de una forma sencilla y económica.

6.2. FUTURAS MEJORAS

Puesto que este proyecto se ha realizado con unos conocimientos iniciales prácticamente nulos de programación y electrónica y aprendiendo de forma sucesiva, una vez concluido el trabajo, se perciben algunos aspectos del código y el diseño que se pueden mejorar.

- Mejora de la calidad estética de la interfaz.

Dado que se ha concentrado más esfuerzo en la correcta resolución, verificación de los resultados y métodos numéricos, se hace notar que la visualización se puede mejorar con la inclusión de más parámetros referentes al ensayo o detalles visuales.

.

-Gráfica carga-flecha.

Una forma de mejorar la calidad estética de la gráfica correspondiente al primer cuadrante sería una línea o curva de unión entre los puntos de la gráfica carga-flecha.

-Deformada.

Respecto a la visualización de la deformada y, dado que se genera mediante un bucle definido entre 0 y L y a través de líneas verticales que dibujan la curva, si se registra una flecha menor a la que ya está dibujada -descarga de la viga, por ejemplo-, la deformada no se reduce, esto es debido a que las coordenadas van asociadas al bucle y en consecuencia solo aumentan.

- Referenciación de parámetros.

Dado que el programa se ha ido haciendo progresivamente a medida que aprendía, hay aspectos de referenciación de coordenadas que no se han tenido en cuenta en todo el transcurso del programa y que, por tanto, podrían mejorarse. De la misma forma, también se puede mejorar la referenciación de los parámetros propios de la sección.

REFERENCIAS BIBLIOGRÁFICAS

[1] Smart Cities. "Ciudad, tecnologia y sostenibilidad urbana" http://www.smartcities.es/

[2] Medina Romero, Lara. (2006) "Analisis de la viabilidad económica y ambiental del uso de armaduras corrugadas de acero inoxidable en elementos de hormigón armado sometidos a clases de exposición agresivas. Aplicación a elementos en contacto con aguas agresivas". [En línea, abril de 2018].

[3] Valbruna México."Historia del acero inoxidable" http://valbruna.com.mx/historia-del-acero-inoxidable/ [en línea, mayo 2018]

[4] Navarro, Andrés. (2013) "Determinación de parámetros característicos de comportamientos tenso-deformacional de aceros inoxidables ferríticos conformados en frío" [en línea, abril 2018].

[5] Arrayago, Itsaso. (2011)"Comportamiento estructural del acero inoxidable ferritico frente a cargas concentradas". [En línea, abril de 2018].

[6] Baddoo, N.R..(2008) "Stainless steel in construction: A review of research, applications, challenges and opportunities" Journal of constructional steel research 1199-1206 [En línea, abril de 2018].

[7] Houska, Catherine y Wilson, Kirk."El acero inoxidable inspira metamosfosis en el diseño" Aplica Inox. [En línea, mayo de 2018].

[8] Administración portuaria integral de Progreso (2010). "El muelle de Progreso, Ejemplar". http://www.puertosyucatan.com/cgi-bin/news.cgi?folio=238 [en línea, mayo 2018]

[9] EN 1993-1-4:2006, Eurocode 3: Design of steel structures - Part 1-4: General rules - Supplementary rules for stainless steel, CEN, 2006.

[10] Euro Inox and Steel Construction Institute, Design Manual for Structural Stainless Steel, 3rd edition ed., 2006. [En línea,2016]

[11] 6. Parrot, Aaron y Warshaw, Lane. "Industry 4.0 and the digital twin" Deloitte [En línea, abril de 2018].

[12] Grieves, M.; Vickers J.(2016) " Digital Twin. Mitigation Unpredictable, Undesirable Emergent Behaviour in Complex Systems" [En línea, abril de 2018].

[13] Siemens. (2017) "The Digital Enterprise EWA- Electronic Works Amberg" [en línea, junio 2018] https://industrie40.vdma.org/documents/4214230/20960337/Siemens%20Digital%20Enterprise%20EWA_1507187944145.PDF/2ea3080c-855c-45c0-891f-b6743e675373

[14] Intellect Soft (2018) "Advanced imaging algorithms in digital twin reconstruction of constructions sites" [En línea, abril de 2018]. https://www.intellectsoft.net/blog/advanced-imaging-algorithms-for-digital-twin-reconstruction.

[15] European Comission. [En línea, junio de 2018]. https://ec.europa.eu/transport/themes/its/c-its_en.

[16] Carreteras conectadas e inteligentes. Ventosa, Javier R. 2018, Revista del Ministerio de Fomento, págs. 10-21.

[17] Port de Barcelona. [En línea, Junio de 2018] http://news.portdebarcelona.cat/esp/noticia.php?id=91&p=.

OTRAS REFERENCIAS BIBLIOGRÁFICAS

[18] Ciudades del futuro.(2017) "¿Qué es el gemelo digital? Ventajas y Aplicaciones" [En línea, marzo de 2018]. https://ciudadesdelfuturo.es/que-es-el-gemelo-digital.php.

[19] Keane, Phillip. Engineering. [En línea, mayo de 2018]. https://www.engineering.com/DesignSoftware/DesignSoftwareArticles/ArticleID/16772/Digital-Twins-Where-Are-We-Now.aspx.

[20]Paultre, Alix. EETimes. [En línea, marzo de 2018] https://www.eetimes.com/author.asp?section_id=36&doc_id=1333284.

[21] Hernández, Luis del Valle. [En línea, febrero de 2018] https://programarfacil.com/podcast/80-processing-lenguaje-para-graficos/.

[22]Processing. [En línea, mayo de 2018]. https:/processing.org.

[23] Arduino. [En línea, mayo de 2018]. https://arduino.cc.

[24] Apache Commons. [En línea, marzo de 2018]. http://commons.apache.org/proper/commons-math/javadocs/api-3.3/org/apache/commons/math3/analysis/solvers/LaguerreSolver.html.

[25] Nudoss. [En línea, junio de 2018] http://nudoss.com/estrategia-smart-del-port-de-barcelona/.

[26] Woodhead, R.; Stephenson, P.; Morrey's D.(2018) " Digital, construction: From point solutions to IoT ecosystem" Automotion in construction, Elsevier. [Junio 2018]

ANEXOS

ANEXO 1: ESPECIFICACIONES DEL SENSOR DE DISTANCIA POR ULTRASONIDO

HC-SR04 Ultrasonic Range Finder

Manual

Features

- Distance measurement range: 2cm - 400cm
- Accuracy: 0.3cm
- Detect angle: 15 degree
- Single +5V DC operation
- Current comsuption: 15mA

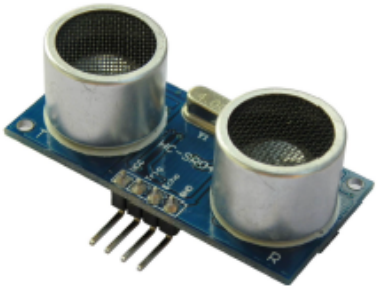
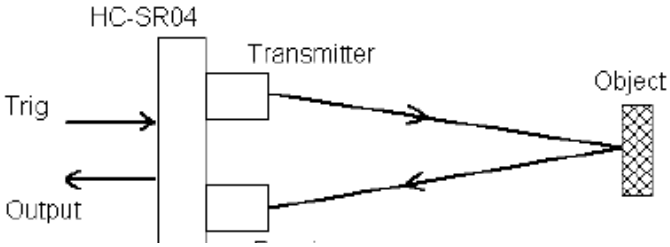


Fig. 1

How It Works

HC-SR04 consists of ultrasonic transmitter, receiver, and control circuits. When trigged it sends out a series of 40KHz ultrasonic pulses and receives echo from an object. The distance between the unit and the object is calculated by measuring the traveling time of sound and output it as the width of a TTL pulse.



How To Use It

To measure distance you need to generate a trig signal and drive it to the Trig Input pin. The trig signal leve must meet TTL level requirements (i.e. High level > 2.4V, low level < 0.8V) and its width must be greater than 10us. At the same time you need to monitor the Output pin by measuring the pulse width of output signal. The detected distance can be calculated by the formula below.

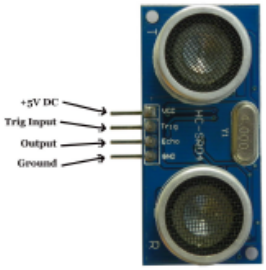


Fig.3

$$\text{Distance} = \frac{\text{Pulse Width} * \text{Sound Speed}}{2}$$

where the pulse width is in unit of second and sound speed is in unit of meter/second. Normally sound speed is 340m/s under room temperature.

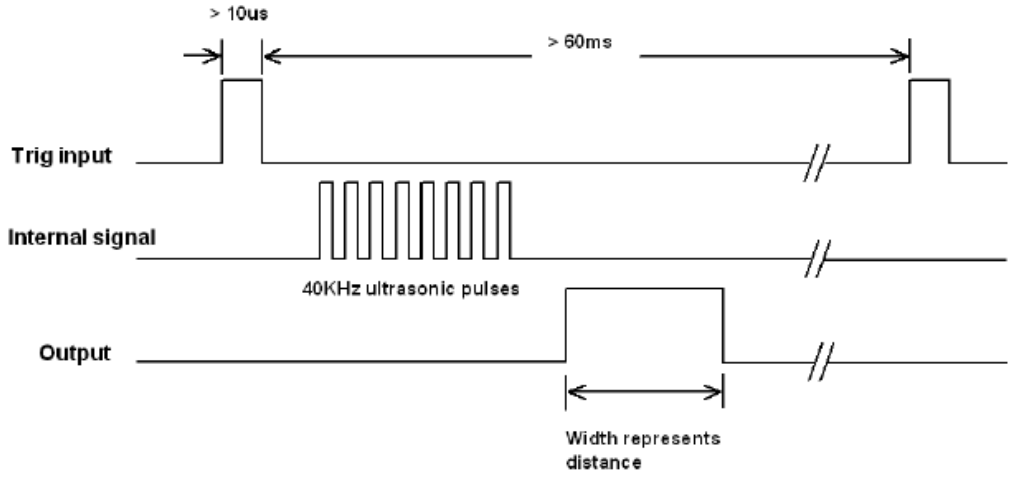


Fig. 4

Notes: 1. The width of trig signal must be greater than 10us
2. The repeat interval of trig signal should be greater than 60ms to avoid interference between connective measurements.

Specifications

Parameters	Specification
Operating Voltage	+5V DC
Operating Current	15mA
Operating Frequency	40KHz
Maximum Distance	400cm
Minimum Distance	2cm
Detect Angle	15 degree
Resolution	0.3cm
Input Trig Signal	>10us TTL pulse
Output Signal	TTL pulse with width representing distance
Weight	
Dimension	45 x 20 x 15 mm

ANEXO 2: CÓDIGO DE ARDUINO DE RECIBIDA DE DATOS

```
#include <AP_Sync.h>
AP_Sync streamer(Serial);

const int EchoPin = 8;
const int TriggerPin = 9;

void setup() {
Serial.begin(9600);

pinMode(TriggerPin, OUTPUT);
pinMode(EchoPin, INPUT);
}

void loop() {
int mm = ping(TriggerPin, EchoPin);
//Serial.write(mm);
//Serial.println(mm);
delay(1000);
}

int ping(int TriggerPin, int EchoPin) {
long tiempo, distancia_mm;
digitalWrite(TriggerPin, LOW); //para generar un pulso limpio ponemos a LOW 4us
delayMicroseconds(4);
digitalWrite(TriggerPin, HIGH); //generamos Trigger (emisión de pulso) de 10us
delayMicroseconds(10);
digitalWrite(TriggerPin, LOW);
tiempo = pulseIn(EchoPin, HIGH); //medida el tiempo entre pulsos, en microsegundos

distancia_mm = (tiempo * 17) /100 ; //conversión a distancia, en mm
//Serial.println(distancia_mm);
//return tiempo;
//return distancia_mm;
streamer.sync("datosDelta",distancia_mm);
delay (1000);
}
```

ANEXO 3: CÓDIGO DE PROCESSING. ENSAYO PLETINA DE ACERO AL CARBONO.

```
import apsync.*;                    //Libreria para sincronzar Arduino con Processing
import processing.serial.*;
AP_Sync streamer;
```

//VARIABLES ORIGEN CUARANTE 1

```
int X1;
int Y1;       //variables de 0.0 locales cuadrante 1
```

//VARIABLES ORIGEN CUADRANTE 2

```
int X2;       //variables de 0.0 cuadrante 2
int Y2;
```

//VARIABLES CONSTANTES ECUACION DELTA-P

```
float I=106.67;    //inercia (mm^4)
float E=210000.0;     //módulo de Young (MPa)
float longViga=1500.0;  //mm
float x=longViga/2;
float cte=4.0/3.0;      //cte auxiliar de la ecuación de la deformada.  (Al ser definido como float necessita una coma en la declaración)

float h=600.0;          //Distancia entre sensor y viga
```

//VARIABLES DE ENTRADA ARDUINO (DELTA)

```
float [] delta =new float [5000];
public float datosDelta;
float dist;
int j=0;
```

//VARIABLE RESULTADO -CARGA-

```
float [] P=new float [5000]
```

//VARIABLES CUADRANTE 4 -FOTOGRAFIAS-

```
float d;
PImage photo;
String a="frame";
String c=".JPG";
String f = str(d);
float d2;
```

//SETUP

```
void setup() {

size(1000,700);
//printArray(Serial.list());          //Saber que COM está disponible

streamer=new AP_Sync(this,"COM3",9600);    //Comunicación con Arduino (recibe dato distancia)
delay (500);
```

//PANTALLA PRINCIPAL

```
fill(169,204,227);              //cuadrante 1
rect(0,60,width/2,height/2+60);


fill(171,235,198);              //cuadrante 2
rect(width/2,60,width/2,height/2+60);


fill(232,218,239);              //cuadrante 3
rect(0,height/2+60,width/2,height/2);

fill(235,237,239);              //cuadrante 4
rect(width/2,height/2+60,width/2,height/2-60);


 line(width/2,60,width/2,height);   //línea central vertical
 line(0,height/2+60,width,height/2+60);   //línea central horizontal
```

//TEXTO

```
 fill(5,0,0);
 textSize(15);
 text("Desarrollo de interficíes gráficas a tiempo real para vigas de acero inoxidable: CASO LINEAL",150,20);       //Titulo
 text("Karen Fornés Christensen",400,40);

textSize(15);
text("GRÁFICA CARGA-FLECHA",150,110);
textSize(14);
text("P (N)",10,95);                          //texto cuadrante 1

text("Flecha (mm)",(width/2)-100,(height/2)+50);

textSize(12);
text("25",15,118);
text("45",15,141);
text("40",15,166);
text("35",15,191);
text("30",15,216);                //números eje Y cuadrante 1
text("25",15,239);
text("20",15,264);
text("15",15,287);
text("10",15,314);
text("5",15,339);
 textSize(12);

 text("20",82,382);
 text("40",122,382);
 text("60",162,382);;
 text("80",202,382);
 text("100",238,382);             //números eje x cuadrante 1
 text("120",278,382);
 text("140",318,382);
 text("160",358,382);
 text("180",398,382);
 text("200",438,382);
```

```
textSize(15);
 text("DEFORMADA",700,110);                    //Texto cuadrante 2
 text("Flecha (mm) =",650,150);

 text("SECCIÓN TRANSVERSAL",150,460);          //Texto cuadrante 3

 text("FOTOGRAFIAS DEL ENSAYO",700,460);               //Texto cuadrante 4
 fill(255);
                    //CUADRANTE 1 -EJES-

X1=50;                  //  0.0 cuadrante 1
 Y1=height/2-50;

  line(X1,70,X1,Y1+55);                //eje "y" cuadrante 1
  line(X1,70,X1+10,X1+30);              //flechas eje "y"
  line(X1,70,X1-10,X1+30);

 for (int posY=0; posY<10; posY++){       //graduacuión eje "y"
   line(45,114+24*posY,55,114+24*posY);
 }

  line(50,height/2+10,width/2-20,height/2+10);   //eje "x"  cuadrante 1

  for (int posX=0; posX<21; posX++){            //graduación eje ''x''
      line(50+20*posX,365,50+20*posX,355);
  }

  line(width/2-20,360,width/2-30,height/2);  //flechas eje "x"
  line(width/2-20,360,width/2-30,height/2+20);

    //CUADRANTE 2 -VIGA-

  X2=width/2;        //coordenadas 0.0 cuadrante 2
  Y2=height/3;
```

```
 line(X2+50,Y2+40,(X2*2)-50,Y2+40);
 triangle(X2+100,Y2+40,X2+115,Y2+60,X2+85,Y2+60);          //representación viga
 triangle(2*X2-100,Y2+40,2*X2-85,Y2+60,2*X2-115,Y2+60);


        //CUADRANTE 3

    pushMatrix();          //Definición sistema de coordenadas de cuadrante 3
    translate(0,height/2);
   stroke(1);
   fill(105,175,0);
   rect(50,165,400,75);
   fill(255,255,255);                    //representación sección transversal
      popMatrix();


}

                //DRAW

void draw (){

                    // RECIBIDA DE DATOS + CÁLCULO DE P

  dist=datosDelta;

 // d=datosDelta;          //leer datos arduino
 //println(dist);           //print de la distancia que llega de Arduino
  d=h-dist;
 //println(flecha);
 delta[j]=d;
P[j]=(d*16*E*I)/((1-(cte*((pow(x,2))/(pow(longViga,2)))))*pow(longViga,2)*x);    //cálculo de P
  println(P[j]);         //Print del cálculo de P

        //CUADRANTE 1  -  DIBUJAR GRAFICA P-DELTA

pushMatrix();
  translate(50,Y1+60);       // definición coordenades locales cuadrante 2
   stroke(20);
```

```
 for (int d=0; d<delta.length; d++){
   if (delta[d]<210){                    //if para no dibujar más allá del eje
   strokeWeight(3);                      //grosor línea
   line(delta[d]*2,0.0,delta[d]*2,-P[d]*10);          //línea vertical delta-P a partir de los datos
(línea solo admite float)
   }

   else {

   }

   //strokeWeight(3);                    //grosor punto
   // point(delta[d],-P[d]);              //opcion grafica delta-P con puntos
}
 j++;

popMatrix();
```

// CUADRANTE 2 - DIBUJAR DEFORMADA

```
 X2=width/2;
 Y2=height/3;            //Varialbles de coordenadas locales cuadrante 2
  pushMatrix();

 translate(X2+100,Y2+40);          //Definicón de coordenadas locales,  cuadrante 2.

for(int i = 0; i < 150; i++){

    float deformada = (pow(300,2)*i/(16*E*I))*(1-(4.0/3.0*pow(i,2))/(pow(300,2)))*d; //Calcular
curva de la deformada
         strokeWeight(1);
         stroke(120,145,169);
            line(i,0,i,deformada);          //Dibujar deformada (x<L/2)
}

popMatrix();
```

```
pushMatrix();
translate(X2+400,Y2+40);                    //Definición coordenadas locales cuadrante 2.2
 for (int i=0; i<=150; i++){
 float deformada = (pow(300,2)*i/(16*E*I))*(1-(4.0/3.0 *pow(i,2))/(pow(300,2)))*d;    //Calcular
deformada
 // println(deformada);
     stroke(120,145,169);
   line(-i,0,-i,deformada);       //Dibujar deformada (L/2<=x<=L)
   strokeWeight(1);
    stroke(0);

    line(-150,0,-150,deformada); // Línea vertical flecha máxima (centro luz)

   }

 popMatrix();

       fill(171,235,198);
       noStroke();
       rect(750,125,100,50);
       fill(5,0,0);
       text(d,750,150);                //Texto que muestra el valor de Delta (recibido de Arduino)
           delay(500);


                  //CUADRANTE 3 -DEGRADADO COLOR-

 int r=255;          //Variables de color
 int g=80;
 int b=0;


 pushMatrix();
 translate(0,height/2+40);          //Definición coordenadas locales cuadrante 3


for (int i=0; i<75; i++){

      if (d<=50) {                    //Degradado de color según valor de delta
        stroke(105,175,0);
        line(50,125+i,450,125+i);

              }
```

```
        else if (d>25 && d<=50) {

         stroke(r-2*i,g+i,b);
                  line(50,125+0.1*i,450,125+0.1*i);
      }

        else if (d>50 && d<=75) {

         stroke(r-2*i,g+i,b);
                  line(50,125+0.1*i,450,125+0.1*i);
      }

        else if (d>75 && d<100) {
              stroke(r-2*i,g+i,b);
                   line(50,125+0.3*i,450,125+0.3*i);
      }

        else if (d>=100 && d<150) {


         stroke(r-2*i,g+i,b);
                  line(50,125+0.5*i,450,125+0.5*i);
        }

        else if (d>150 && d<200) {

         stroke(r-2*i,g+i,b);
                  line(50,125+0.7*i,450,125+0.7*i);
        }

        else {
          // stroke(r,g,b);
          stroke(105,175,0);
                  line(50,125+i,450,125+i);
     }

    }

popMatrix();
```

```
                    //CUADRANTE 4 -FOTOGRAFIAS-

// println(d);

 d=10*round(d/10);                //para que solo muestre las imagenes correspondientes a la deformación cada 1 cm (10mm)

 int d2=(int) d;


  //println(d2);


 String f=str(d2);
 photo = loadImage(a+f+c);
 photo.resize(280,200);
 image(photo,610, 500);
 //println(a+f+c);


}
```

ANEXO 4: CÓDIGO DE PROCESSING. PROTOTIPO PARA UN ENSAYO REAL DE UNA VIGA DE ACERO INOXIDABLE.

```
import org.apache.commons.math3.analysis.solvers.*;     //importar libreria Solver
import org.apache.commons.math3.complex.Complex;
import org.apache.commons.math3.*;
import apsync.*;              //Libreria para sincronzar Arduino con Processing
import processing.serial.*;
AP_Sync streamer;

        //VARIALBLES CUADRANTE 1
int X1;
int Y1;       //variables de 0.0 locales cuadrante 1

       //VARIABLES CUADRANTE 2

int X2;       //variables de 0.0 cuadrante 2
int Y2;
         //VARIABLES PARA RESOLICIÓN DE P (a partir de datos Arduino)

double resultadoP; //resultado formato double no admisible para funcion line
float resultadoP_Def;  //resultado final de forma float para funcion line
float[] delta= new float [5000];
float[] Carga= new float[5000];
public float datosDelta;

float d;        //variable que guarda los datos obtenidos
int j=0;
float h=400.0;
float dist;
        //COEFICIENTES DE LA ECUACION A RESOLVER -CARGA-

double coeff[] = {0,-1.66e-4,0,0,-1.557e-18};
Complex comp[];   //Declaramos comp como lista de numeros complejos

     //COEFICIENTES CUADRANTE 2 -DEFORMADA-

float E=192918.0;         //Módulo elástico MPa
float I=3238892.0;          //Inercia (mm^4)
float longViga=1500.0;
```

```
//  VARIABLES PARA CUADRANTE 3
PImage photo;
String a="frame";
String b = str(1);
String c=".png";


//SETUP
void setup() {
 size(1000,700);                    // Tamaño pantalla
 //printArray(Serial.list());            //Para saber que COM esta disponible
 streamer=new AP_Sync(this,"COM3",9600);        // Comunicacion con arduino, recibe dato distancia
   delay (1000);

//PANTALLA PRINCIPAL

 fill(169,204,227);                //cuadrante 1
 rect(0,60,width/2,height/2+60);

 fill(171,235,198);                //cuadrante 2
 rect(width/2,60,width/2,height/2);

 fill(232,218,239);                //cuadrante 3
 rect(0,height/2+60,width,height/2-60);

  line(width/2,60,width/2,height/2+60);    //linia central vertical
  line(0,height/2+60,width,height/2+60);    //linia central horizontal

//TEXTO
 fill(5,0,0);
 textSize(15);
 text("Desarrollo de interficíes gráficas a tiempo real para vigas de acero inoxidable: CASO NO LINEAL",150,20);       //Titulo
 text("Karen Fornés Christensen",400,40);
 text("GRÁFICA CARGA-FLECHA",150,110);
 textSize(14);
 text("P (KN)",10,95);                          //texto cuadrante 1
 text("Flecha (mm)",(width/2)-100,(height/2)+50);
```

```
textSize(12);
text("120",18,125);
text("110",18,145);
text("100",18,164);
text("90",20,184);
text("80",20,204);
text("70",20,225);
text("60",20,244);                //números eje Y cuadrante 1
text("50",20,265);
text("40",20,285);
text("30",20,304);
text("20",20,325);
text("10",20,346);

 textSize(12);
 text("20",82,382);
 text("40",122,382);
 text("60",162,382);;
 text("80",202,382);
 text("100",238,382);             //numeros eje x cuadrante 1
 text("120",278,382);
 text("140",318,382);
 text("160",358,382);
 text("180",398,382);
 text("200",438,382);
textSize(15);
text("DEFORMADA",700,110);                    //texto cuadrante 2
text("Flecha (mm) =",650,150);
text("VISUALIZACIÓN VIGA EN 3D",400,450);          //texto cuadrante 3
fill(255);
          //CUADRANTE 1 -EJES-

X1=50;                  //  0.0 cuadrante 1
Y1=height/2-50;

 line(X1,70,X1,Y1+55);        //eje "y" cuadrante 1
  line(X1,70,X1+10,80);              //flechas eje
  line(X1,70,X1-10,80);
    for (int posY=0; posY<12; posY++){        //graduacuión eje "y"
```

```
  line(45,120+20*posY,55,120+20*posY);
 }

  line(50,height/2+10,width/2-20,height/2+10);        //eje "x"  cuadrante 1

  for (int posX=0; posX<21; posX++){              //graduación eje ''x''
      line(50+20*posX,365,50+20*posX,355);
 }

 line(width/2-20,360,width/2-30,height/2);     //flechas eje
 line(width/2-20,360,width/2-30,height/2+20);

              //CUADRANTE 2 -VIGA-

X2=width/2;         //coordenadas 0.0 cuadrante 2
Y2=height/4;

 line(X2+50,Y2+60,(X2*2)-50,Y2+60);
 triangle(X2+100,Y2+60,X2+115,Y2+80,X2+85,Y2+80);            //representación viga
 triangle(2*X2-100,Y2+60,2*X2-85,Y2+80,2*X2-115,Y2+80);

}

 void draw(){
            // RECIBIDA DE DATOS + CÁLCULO DE P

    dist=datosDelta;
    // println(dist);
    d=h-dist;
    println(d);

       LaguerreSolver P = new LaguerreSolver();       //declaracion calculo P

 delta[j] = d;  //guardar entrada arduino
 coeff[0]=  d; //cte de la ecuacion (posicion 0 del vector coeff) correspondiente al dato delta
// println(d);
 comp=P.solveAllComplex(coeff,0.0); //cálculo de todas las raices de la funcion
 //println(comp);       //Ver resultados del cálculo total de P por pantalla

  for(int c=0; c<comp.length; c++){
```

```
    if ( (comp[c].getImaginary() == 0) && (comp[c].getReal() >= 0.0)){       //for para escoger resultado de P real
      resultadoP = comp[c].getReal();        //si imaginario =0 y real diferente, guardar real= resultado de P, para dibujar linea
      resultadoP_Def = (float) resultadoP;        //conversion de double a float para funcion line
      Carga[j]=resultadoP_Def;

                    //DIBUJAR DEFORMADA

      for (int k=0; k<=150; k++){

      if (k<=100){
      float deformada = -(((4.9838e-36)*pow(5*k,6)-(3.27065e-21)*5*k)*pow(Carga[j],4)+((2.667e-13)*pow(5*k,3)-(4.10e-7)*5*k)*Carga[j]);
        //println(Carga[j]);
        //println(deformada);
       strokeWeight(1);
        stroke(120,145,169);
          line(k+600,235,k+600,deformada*0.1+235);          //Dibujar deformada x<L1
      }
      else if(k>100){

    float deformada2 = -((3.4e-5*Carga[j])+(1.24715e-18*pow(Carga[j],4))-(6e-7*Carga[j]*5*k)-(7.94e-21*pow(Carga[j],4)*5*k)+(4.10e-10*Carga[j]*pow(5*k,2))+(4.67e-24*pow(Carga[j],4)*pow(5*k,2)));
    strokeWeight(1);
        stroke(120,145,169);
          line(k+600,235,k+600,deformada2*.1+235);          //Dibujar deformada L1<x<L/2
       //  println(deformada2);
}
}
pushMatrix();
translate(900,235);
      for (int k=0; k<=150; k++){
      if (k<=100){
      float deformada = -(((4.9838e-36)*pow(5*k,6)-(3.27065e-21)*5*k)*pow(Carga[j],4)+((2.667e-13)*pow(5*k,3)-(4.10e-7)*5*k)*Carga[j]);
        //println(Carga[j]);
        // println(deformada);
    strokeWeight(1);
```

```
          stroke(120,145,169);
           line(-k,0,-k,deformada*0.1);              //Dibujar deformada x>L-L1

      }
        else if(k>100){
         float        deformada2        =        -((3.4e-5*Carga[j])+(1.24715e-18*pow(Carga[j],4))-(6e-
7*Carga[j]*5*k)-(7.94e-21*pow(Carga[j],4)*5*k)+(4.10e-10*Carga[j]*pow(5*k,2))+(4.67e-
24*pow(Carga[j],4)*pow(5*k,2)));
     strokeWeight(1);
          stroke(120,145,169);
            line(-k,0,-k,deformada2*0.1);             //Dibujar deformada L/2<x<L-L1
              // println(deformada2);

}
}

  popMatrix();
    println(Carga[j]);

  }
  }
            fill(171,235,198);
        noStroke();
        rect(750,125,100,50);
        fill(5,0,0);
        text(d,750,150);                //Texto que muestra el valor de Delta (recibido de Arduino)

                        //DIBUJAR GRÁFICA CARGA-FLECHA
   pushMatrix();
    translate(50,Y1+60);     // coordenades en 0.0 locales
      stroke(20);
      strokeWeight(3);

  for (int q=0; q<delta.length; q++){

       if (delta[q]<210){
  //line(2*delta[q],0.0,2*delta[q],-Carga[q]/500);             // Opcion dibujar grafica delta-P con
lineas verticales(A partir de datosDelta)
   point(2*delta[q],-Carga[q]/500);                    //Opcion grafica con puntos
}
```

```
        else  {
         }
      }
popMatrix();
  j++;
   delay(500);
```

//CUADRANTE 3 -VISUALIZACIÓN EN 3D DE LA VIGA-

```
      int d2=(int) d;
b=str(d2);
 photo = loadImage(a+b+c);
  photo.resize(700,300);
    image(photo, width/2-350, height/2+80);
   println(a+b+c);

 }
```