Universidad del País Vasco
Euskal Herriko Unibertsitatea
BILBOKO INGENIARITZA ESKOLA
ESCUELA DE INGENIERÍA DE BILBAO

GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

TRABAJO FIN DE GRADO

CONSTRUCCIÓN Y CONTROL DE UNA MAQUETA PUENTE COLGANTE

Alumno: Cuesta Arrillaga, Mikel

Director: Gómez Garay, Vicente

Curso: 2017-2018

Fecha: 23/07/2018

Resumen Trilingüe

El objeto de este Trabajo Fin de Grado ha sido recrear el funcionamiento básico del Puente Colgante de Vizcaya mediante el uso de la plataforma Lego Mindstorms principalmente. El trabajo ha incluido el estudio de viabilidad, el diseño previo, la construcción del Puente y su control. Para realizar el diseño previo se ha utilizado el software AutoDeskFusion 360. En cuanto a la construcción se refiere, los elementos utilizados han sido diversos, incluyendo piezas de Lego, barras Tetrix y materiales adicionales. Para el control del Puente, la programación se ha llevado a cabo mediante Bricx Command Center con NXC, un lenguaje de programación similar a C.

Gradu Amaierako Lan honen helburua Bizkaiko Zubiaren funtzionamendua islatzea izan da, batik bat Lego Mindstroma kit-aren laguntzaren bidez. Lanaren osotasunean hiru zati nagusitan banatu da: aurretiko diseinua, maketaren eraikuntza eta zubia beraren kontrola. Aurretiko diseinua aurrera eramateko AutoDeskFusion 360 softwarea erabili da. Eraikuntzari dagokionez, elementu erabiliak hainbat izan dira, hala nola Lego piezak, Tetrix-ko barrak eta eskulanetarako hainbat material gehigarri. Kontrola Bricx Command Center softwarearen bidez eraman da, NXC programazioa erabiliz, C programazio lengoaiaren antzeko programazioa.

The purpose of this Final Degree Project has been to recreate in scale the Suspension Bridge of Biscay by using mainly the Lego Mindstorms kit. The work has included a previous design, the construction of the bridge and its respective control. The previous design has been carried out by AutoDeskFusion 360 software. As far as the construction is concerned, diverse elements have been used, including pieces of Lego, Tetrix bars and additional materials. In order to control the bridge, programming has been carried out by Bricx Command Centre software, using NXC, a programming language similar to C with minor modifications.

Palabras clave: Lego Mindstorms NXT, Servomotor, Puente Colgante, Maqueta.

Índice

Índice de Figuras

Índice de Tablas

1. MEMORIA

1.1 Introducción

Este documento recoge los diferentes aspectos tratados en la construcción y control de una maqueta del Puente Colgante de Vizcaya. La primera parte presenta el contexto del trabajo, sus requerimientos y los beneficios que aporta. Posteriormente, se desarrollan tres alternativas y se evalúan según diferentes criterios de selección. Una vez seleccionada, se realiza una descripción de la solución escogida.

Partiendo de la alternativa seleccionada, se expone la metodología llevada a cabo: diseño de la estructura principal, construcción del Puente, diseño del sistema de control y la programación.

Otro de los aspectos importantes del trabajo es su planificación. La metodología implantada se refleja en el listado de tareas y su diagrama de Gantt.

Asimismo, este informe incluye un presupuesto ejecutado durante el transcurso del trabajo, separado en cuatro apartados.

Por último, el documento contiene dos anexos: el Anexo I con el código elaborado para el cálculo de distancias de los sensores ultrasonidos, el diagrama de flujo de control del sistema de la maqueta y su código, y el Anexo II con el listado de materiales empleados en la construcción.

1.2 Objetivo y alcance de trabajo

El objetivo principal de este Trabajo Final de Grado(TFG) es diseñar y construir un prototipo capaz de replicar el funcionamiento básico del Puente Colgante de Vizcaya a partir de la plataforma Lego Mindstorms.

Construir la estructura real y sus rasgos más significativos como son las cuatro torres con sus cables de sujeción, los dos travesaños y la barquilla.

Dada la imposibilidad de replicar con exactitud el funcionamiento real del transbordador, la maqueta simulará mediante el control, servomotores y sensores los siguientes aspectos:

- El movimiento horizontal de la barquilla transbordadora.
- La parada del trayecto si alcanza un obstáculo, sea un barco o cualquier objeto que ponga en riesgo el transporte.

1.3 Contexto

Llegados a la parte final del grado, se presenta al requisito de hacer el TFG tras un periodo de estudios académicos de 4 años en la Escuela de Ingeniería de Bilbao.

El trabajo se lleva a cabo dentro del Departamento de Ingeniería de Sistemas y Automática de la misma Escuela.

Con el fin de contextualizar el trabajo, primero, se expondrá un breve texto sobre la historia del Puente real y, a continuación, se tratará en detalle la plataforma Lego Mindstorms.

Por un lado, el Puente de Vizcaya también conocido como Puente Vizcaya, Puente Colgante o Puente Palacio es un puente transbordador de peaje, concebido, diseñado y construido por la iniciativa privada entre 1887 y 1893, que une las dos márgenes de la ría de Bilbao en Vizcaya (España), y fue inaugurado el 28 de julio de 1893, siendo el primero de su tipología en el mundo y uno de los 8 que aún se conservan.

Figura 1: El puente transbordador de Portugalete a Las Arenas.

El Puente enlaza la Villa de Portugalete con el barrio de Las Arenas, que pertenece al municipio de Getxo, así como las dos márgenes de la ría de Bilbao. Su construcción se debió a la necesidad de unir los balnearios, destinados a la burguesía industrial y a los turistas de finales del siglo XIX, existentes en ambas márgenes de la ría.

Por otro lado, Lego Mindstorms es un set de construcción de robots programables fabricado por la empresa LEGO. El kit posee elementos de construcción, un ladrillo microprocesador, servomotores y sensores. El set incluye software que facilita la construcción, la programación y el control del robot desde un PC o un Smartphone.

Este producto de Lego fue comercializado por primera vez en septiembre de 1998.

Figura 2: Kit Lego Mindstorms.

Hasta 2015 ha habido tres generaciones de Lego Mindstorms: el bloque RCX, el bloque NXT y el EV3. Las tres generaciones se diferencian por el microprocesador, los servomotores y los sensores. La última generación de la plataforma es el EV3 que incluye mejoras respecto a sus predecesoras como la conexión wi-fi con el microprocesador e integración de nuevos sensores como un giróscopo, sensores infrarrojos de proximidad, etc. En este TFG se utiliza la versión NTX del producto.

Figura 3: Microprocesador NXT conectado a tres servomotores y cuatro sensores.

Como se puede observar en la figura anterior, el hardware se compone de tres partes:

- **Ladrillo**. El ladrillo, la "mente" del sistema. Recibe señales desde los puertos de entrada de los sensores y ejecuta distintas acciones como mandar señales a distintos servomotores, a través del programa diseñado por el usuario. Estas señales se reciben y se mandan mediante cable o bluetooth, a un máximo de tres motores y de cuatro sensores. Para programar este dispositivo, existen diferentes lenguajes de programación de diferentes niveles.

- **Servomotores**. Originan movimiento circular mediante una señal eléctrica.

- **Sensores**. Perciben cambios en el entorno y envían señales eléctricas al microprocesador. Existen varios tipos de sensores como podemos observar en la imagen anterior, de izquierda a derecha: sensor de contacto (se activa al producirse un contacto en el extremo del sensor), sensor de sonido (se activa al recibir sonidos de determinadas frecuencias), sensor de luz (se activa al disminuir la luminosidad entre dos umbrales de blanco y negro) y por último el sensor de ultrasonidos (calcula la distancia al objeto más cercano mediante el reflejo de las ondas ultrasónicas en dicho objeto).

1.4 Beneficios que aporta el trabajo

Técnicos: Se adquieren conocimientos tanto de diseño por ordenador como de programación, dos aspectos fundamentales en las actividades de la Ingeniería industrial. Concretamente, se profundiza en el Modelado Paramétrico 3D, creación de planos y sus acotaciones, y en la programación estructurada. Además, se han puesto práctica conocimientos a la hora de diseñar el sistema de control. Y por último, se redacta un proyecto de ingeniería simplificado que incluye los elementos básicos de una memoria y la utilización de un lenguaje adecuado y preciso en su redacción.

Y por último la utilización, en general, de Internet como fuente de información.

Personales: Tras la construcción de este prototipo, además de destreza manual, se adquiere capacidad en resolución de problemas, en toma de iniciativa para hacer frente a distintos retos surgidos durante el desarrollo del TFG, y así como en el gusto estético. Finalmente, se aplican conocimientos previamente adquiridos de distintas materias cursadas durante el Grado, como Automática y Control para el diseño y la implantación del controlador, Mecánica para la construcción de la maqueta e Informática para la programación.

Sociales: La maqueta construida podrá ser expuesta en las diferentes jornadas de puertas abiertas organizadas por la Escuela, así como en cualquier tipo de eventos donde se considere oportuno. Gracias a la vistosidad de la misma, ésta podrá crear expectativa y, quizás, captar el interés de futuros ingenieros.

Otro aspecto a tener en cuenta es que esta estructura, y su programación, puede ser el punto de partida para otros futuros proyectos del próximo alumnado de 4º curso, tanto para realizar nuevas maquetas como para realizar modificaciones y mejoras sobre la misma.

1.5 Estado del Arte

Después de completar una investigación en Internet se obtiene la siguiente información:

- La utilización del kit Lego Mindstorms se limita únicamente a usar los componentes estructurales, es decir, no incorporan ladrillos ni motores. Ejemplos de ello son la construcción de maquetas del Golden Gate de San Francisco y London Bridge.

Figura 4: Golden Gate y London Bridge [11].

- Existen varias maquetas realizadas con diferentes elementos y materiales del Puente de Vizcaya. Prueba de ello son las dos figuras siguientes:

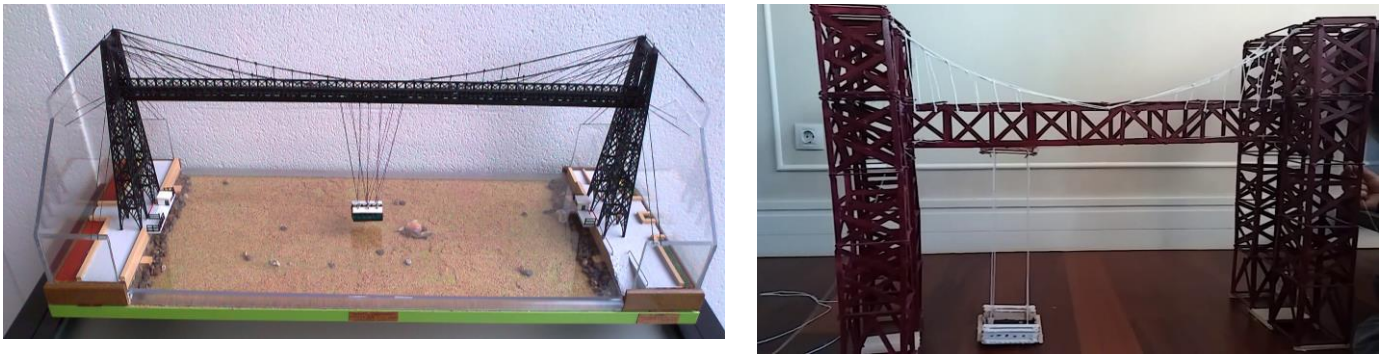

Figura 5: Maquetas del Puente de Vizcaya [12].

Con todo ello, se llega a la conclusión de que este Puente si se ha construido previamente pero sin haberlo dotado de la capacidad de movimiento de la barquilla y la correspondiente automatización.

1.6 Análisis de Alternativas

Para llevar a cabo esta tarea, inicialmente se ha realizado un profundo análisis de la estructura real y sus aspectos técnicos, a continuación se ha elaborado un análisis del material disponible en el Departamento de Ingeniería de Sistemas y Automática y, por último, se han propuesto tres alternativas viables para su posterior valoración, utilizando diferentes criterios, de cara a seleccionar la solución final.

Análisis técnico del Puente de Vizcaya

El análisis técnico del puente original se ha realizado para obtener el aspecto, funcionalidad y las dimensiones principales de la estructura.

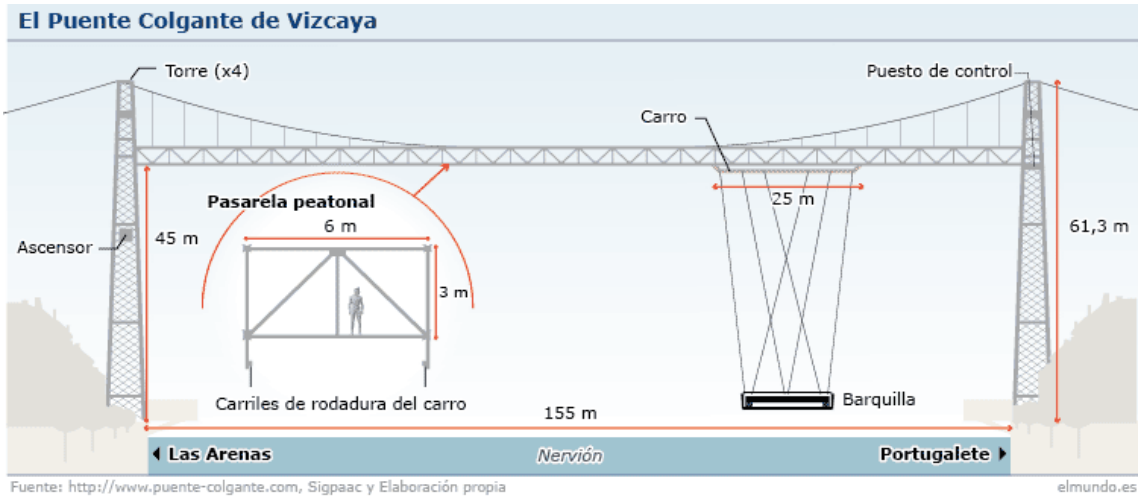

Figura 6: Dibujo esquemático con las dimensiones principales [3].

Álvaro García [3] describe los siguientes aspectos técnicos:

"La estructura básica del Puente está formada por cuatro torres de hierro de 61 metros de altura, levantadas por pares a ambas márgenes de la Ría de Bilbao y que constituyen los pilares del Puente, unidas entre sí por un travesaño de 160 metros de longitud, situado a 45 metros de altura sobre el nivel de pleamar.

La estructura está armada íntegramente con piezas de hierro laminadas en taller y unidas entre sí mediante remaches al rojo vivo, dado que las técnicas de soldadura en la época en que se construyó el Puente estaban poco desarrolladas.

Para fijar la estructura, el Puente utiliza 8 cables de sustentación de acero (4 por cada lado) anclados en bloques de cimentación sitos en los dos extremos del Puente (en Portugalete y el Getxo), a unos 110 metros de distancia de las torres. Además, las torres de la estructura están arriostradas en el sentido perpendicular al travesaño mediante cables de acero anclados en los muelles que discurren paralelos a la Ría, a unos 60 metros de distancia.

El travesaño horizontal superior, de donde pende la barquilla, "gravita" entre las torres mediante 70 cables de acero denominados péndolas. La estructura, en realidad, no está soldada ni remachada a las torres o pilares, sino tan solo sujetada por estas péndolas, que soportan gran parte del peso, y apoyada en las ménsulas, una especie de capiteles adosados a las torres que ayudan a resistir el peso del tablero de forma equilibrada. De ahí el sobrenombre de colgante, ya que realmente al travesaño cuelga sobre las torres, aunque muchos lo asocien a la suspensión de la barquilla.

El transporte de vehículos y pasajeros se realiza por medio de la barquilla, la cual cuelga de un carro de 36 ruedas y 25 metros de longitud que se desplaza a través de los carriles del travesaño horizontal. El actual carro es de 1999 y dispone de doce motores eléctricos (es su origen se utilizó un sistema de tracción por cables y poleas que se accionaba gracias a una caldera de vapor situada en una de las dos torres del Puente). La actual barquilla data de 1998 y es la quinta en la historia del Puente. El puesto de control está situado en la planta baja de una de las torres de Portugalete, para su mejor visión.

La duración actual de viaje entre las dos márgenes de la ría es de un minuto y medio; la frecuencia de transbordo es de ocho minutos.

La barquilla dispone de una capacidad para 6 vehículos (tipo turismo) y para 200 pasajeros (100 en cada sala), pudiendo completarse con 6 motocicletas o bicicletas.

Se han habilitado para subir a las pasarelas, con el fin de realizar visitas turísticas, dos ascensores panorámicos, uno en cada lado del Puente, y a los que se accede a través de las terrazas.

El Puente Vizcaya fue el primer puente transbordador construido en el mundo, pero su originalidad tecnológica no descansa sólo en el desarrollo y puesta en marcha de un mecanismo de transbordo (barquilla, carro y carril), sino también en su espectacular estructura de hierro y en el empleo de cables de acero para fijar dicha estructura."

Análisis del material disponible en el Departamento de Ingeniería de Sistemas y Automática

En esta fase, se ejecutó un estudio sobre los recursos útiles para la construcción de la maqueta.

1. El set de Lego Mindstorms NXT disponible en el Departamento incluye ladrillos NXT, servomotores NXT y sensores de contacto, sonido, luz y ultrasonidos, todos ellos descritos anteriormente.

   Además, en el mismo kit se encontraron diferentes piezas interesantes para la construcción de la estructura:

   - Piezas estructurales: Vigas y ladrillos principalmente.
   - Piezas de unión: Elementos para unir distintos componentes.
   - Elementos de transmisión: Poleas, correas, engranajes, etc.

2. Otro kit de piezas disponible en el Departamento es Tetrix. En él se localizaron piezas más rígidas y robustas (barras metálicas aluminio). Asimismo, en el mismo kit se hallaron tornillos y tuercas para la unión de las barras.

3. Herramientas útiles del Laboratorio I del Departamento. Utensilios como taladros, llaves inglesas, celos y colas existen en el Departamento y son habitualmente utilizadas para la realización de diferentes proyectos.

Alternativas posibles

En este apartado se expondrán diferentes opciones a la hora de crear la estructura del Puente, lenguaje de programación y software de diseño. En cada una de ellas se incluye una breve descripción y las ventajas/desventajas más significativas.

- **Estructura del Puente**. Las diferentes alternativas son:

1. ***Estructura íntegramente compuesta por piezas Lego***. Esta opción se basa en utilizar solamente elementos incorporados en el kit Lego como vigas, ladrillos y elementos de unión.

   Ventajas: Previamente disponibles (reducción de costes), manejo fácil, uniones sencillas y piezas ligeras.

   Desventajas: Imposibilidad de variar las dimensiones de las piezas y problemas de pandeo y no linealidades al tratarse de piezas de plástico.

2. ***Estructura íntegramente creada por piezas de madera***. Crear diferentes vigas y piezas estructurales a partir de láminas de madera y unirlos mediante cola especial.

   Ventajas: Rigidez suficiente requerida por la estructura, piezas creadas a medida y estética visual.

   Desventajas: Costes adicionales por la adquisición del material, material no resistente al agua, el deterioro del material durante el tiempo y estética visual disminuida al incorporar los elementos de control.

3. ***Estructura combinada de piezas Lego con barras Tetrix***. Crear la base de la estructura haciendo uso de las barras de aluminio y después cubrirlas de piezas Lego.

   Ventajas: Rigidez suficiente requerida por la estructura, todas las piezas previamente disponibles (reducción de costes), manejo fácil, uniones sencillas y estéticas.

   Desventajas: Mayor peso, requerimiento de herramientas adicionales para el ensamblaje de la estructura de aluminio.

- **Lenguaje de programación**. Las diferentes alternativas son:

1. ***NTX-G***. Lenguaje de programación en modo gráfico basado en iconos, se trata de un software oficial de LEGO basado en LabVIEW. Orientado a público sin conocimientos previos de programación.

   Ventajas: Es muy útil durante el primer contacto con la plataforma dado que permite hacer programas relativamente rápido sin tener apenas conocimientos de programación.

   Desventajas: A medida que la complejidad del código aumenta resulta algo engorroso dado que aumenta el número de iconos. El código gráfico hace que el programa se ejecute muy lentamente ocupando una gran cantidad de memoria.

2. ***ROBOT-C***. Este lenguaje de programación es un lenguaje textual basado en AnsiC.

   Ventajas: Las posibilidades aumentan enormemente con respecto al NXT-G, principalmente debido a que los lenguajes textuales ocupan mucha menos memoria que los gráficos, algo muy importante debido a las limitaciones de memoria del ladrillo NXT. Los programas son livianos y de rápida ejecución.

   Desventajas: Son necesarios conocimientos de programación.

3. ***NXC(Not Exactly C)***. Este lenguaje de programación textual ha sido creado por la comunidad de usuarios de LEGO con ciertas similitudes con el lenguaje C.

   Ventajas: Se puede encontrar como software libre. Los programas son muy ligeros y de rápida ejecución.

   Desventajas: Los programas desarrollados en el lenguaje Robot-C y NXC no son compatibles dado que cada uno usa sus propias librerías.

- **Software de diseño**. Las diferentes alternativas son:

1. ***Catia***. Creado y comercializado por Dassault Systèmes, está desarrollado para proporcionar apoyo CAD desde la concepción del diseño hasta la producción y análisis de productos. Fue inicialmente desarrollado para la industria aeronáutica, pero en los últimos años se ha integrado también en la industria del automóvil para el diseño y desarrollo de carrocería, y en menor medida el sector de la construcción también ha incorporado su uso para diseñar edificios de gran complejidad.

   Ventajas: La principal peculiaridad de Catia es que provee una arquitectura abierta para el desarrollo de aplicaciones y para personalizar el programa.

   Desventajas: Necesidad de aprendizaje previo.

2. ***Solid Edge***. Creado en 1996 por Intergraph y adquirido posteriormente por Siemens, engloba un grupo de aplicaciones CAD/CAM/CAE. Debido a las distintas aplicaciones que contiene, podemos decir que se trata de un sistema CAD híbrido 2D/3D.

Ventajas: Su mayor ventaja radica en que utiliza tecnología síncrona para acelerar las fases del diseño, lo cual permite realizar los cambios rápidos y mejorar la reutilización. Suele utilizarse para el modelado de piezas. Es usado sobre todo en el diseño de electrodomésticos.

Desventajas: Precio elevado.

3. ***AutoCAD***. Actualmente es desarrollado y comercializado por la empresa Autodesk. Tuvo su primera aparición en 1992.

   Ventajas: Es reconocido a nivel internacional como el más usado. Tiene amplias capacidades de edición, que hacen posible el dibujo digital de planos de edificios o la recreación de imágenes en 3D con mucha calidad. Es compatible con multitud de programas que lo complementan o se apoyan en él. Es sobre todo usado por arquitectos, ingenieros y diseñadores industriales.

   Desventajas: Se requiere un ordenador potente.

Elección entre las alternativas posibles

- **Estructura del Puente**.

A través de la siguiente tabla se seleccionará la solución más adecuada a los objetivos. Los criterios de selección han sido 4: coste, rigidez, estética y durabilidad. Cada uno de los criterios tiene una ponderación en el resultado final y el grado de cumplimiento de las alternativas puede variar entre 1-10:

Estructura del Puente	Coste (x0.2)	Rigidez (x0.5)	Estética (x0.2)	Durabilidad (x0.1)	Total
Alternativa 1 (Lego)	8	4	7	8	5.8
Alternativa 2 (Madera)	5	8	7	7	7.1
Alternativa 3 (Combinada)	8	9	7	8	8.3

Tabla 1: Elección de la alternativa de la Estructura del Puente.

La alternativa elegida ha sido la de combinar las barras de Tetrix con las piezas Lego ya que ha obtenido la mayor puntuación de las tres. Esta solución propuesta se expondrá con más detalle en el siguiente apartado.

- **Lenguaje de programación**.

A través de la siguiente tabla se seleccionará la solución más adecuada a los objetivos. Los criterios de selección han sido 3: tipo(gráfico o textual), ocupación de memoria y rapidez. Cada uno de los criterios tiene una ponderación en el resultado final y el grado de cumplimiento de las alternativas puede variar entre 1-10:

Lenguaje de programación	Tipo (x0.1)	Memoria (x0.5)	Rapidez (x0.4)	Total
Alternativa 1 (NXT-G)	5(Gráfico)	5	5	5
Alternativa 2 (Robot-C)	8(Textual)	7	8	7.5
Alternativa 3 (NXC)	8(Textual)	8	8	8

Tabla 2: Elección de la alternativa del Lenguaje de Programación.

Como se puede observar en la anterior, en la ponderación la memoria adquiere gran importancia debido a la escasez de la misma en el ladrillo.

La alternativa elegida ha sido la de utilizar el lenguaje de programación NXC ya que ha obtenido la mayor puntuación de las tres. Esta solución propuesta se expondrá con más detalle en el siguiente apartado.

- **Software de diseño**.

A través de la siguiente tabla se seleccionará la solución más adecuada a los objetivos. Los criterios de selección han sido 4: extendido en diferentes ámbitos, funcionalidades, facilidad de uso y coste. Cada uno de los criterios tiene una ponderación en el resultado final y el grado de cumplimiento de las alternativas puede variar entre 1-10:

Software de diseño	Extendido (x0.2)	Funcionalidades (x0.3)	Facilidad de uso (x0.2)	Coste (x0.3)	Total
Alternativa 1 (Catia)	5	8	5	6	6,2
Alternativa 2 (Solid Edge)	6	7	7	8	7,1
Alternativa 3 (AutoCAD)	9	8	7	7	7,7

Tabla 3: Elección de la alternativa del Software de Diseño.

La alternativa elegida ha sido la de utilizar AutoCAD ya que ha obtenido la mayor puntuación de las tres. Esta solución propuesta se expondrá con más detalle en el siguiente apartado.

1.7 Descripción de la solución propuesta. Diseño básico

- **Estructura del Puente**.

Dados los materiales disponibles y limitaciones de espacio para su exposición, se decide que la escala a utilizar en el modelo sea de 1:2425. Estos son las partes principales de la maqueta:

  - **Estructura**. Se construye a partir de las barras de aluminio del kit de Tetrix. Cuatro torres y dos travesaños unidos por tornillos tuercas y una chapa de unión. Se decide colocar dos travesaños con el fin de facilitar el tránsito del carrito. A partir de esta estructura rígida, se colocan diferentes vigas de Lego, cubriendo las barra metálicas mediante elementos de unión del propio kit de Lego, y adicionalmente algunas tuercas y tornillos. Para finalizar, se hace uso de un hilo de color negro para simular los cables de suspensión y distintos detalles adicionales.

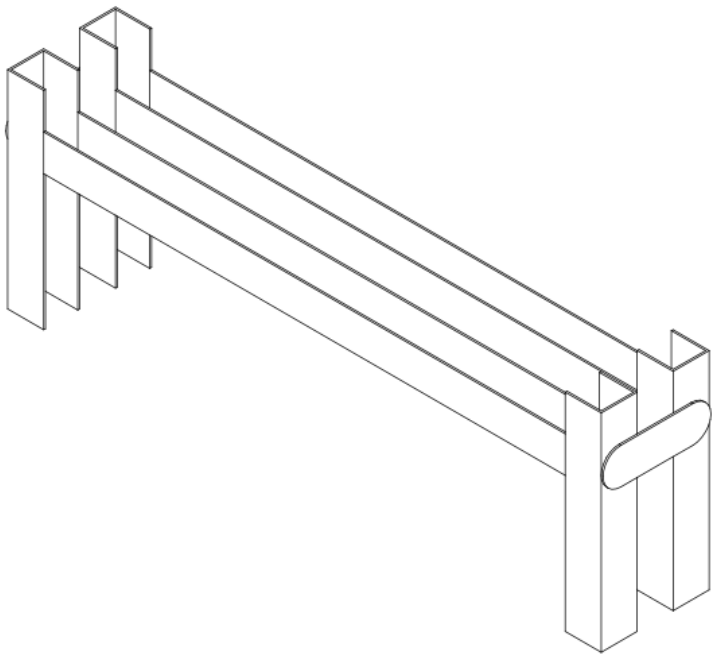

Figura 7: Boceto de la estructura principal.

- **Mecanismo de transbordador**. ¿Cómo conseguir transformar el movimiento circular del servomotor a la traslación de la barquilla? Este problema se soluciona utilizando un mecanismo de polea-correa. En cada extremidad del puente se colocan dos poleas, después la correa se une a los lados de carrito superior. Finalmente, se coloca otra polea a un lado del puente que va unido al servomotor.

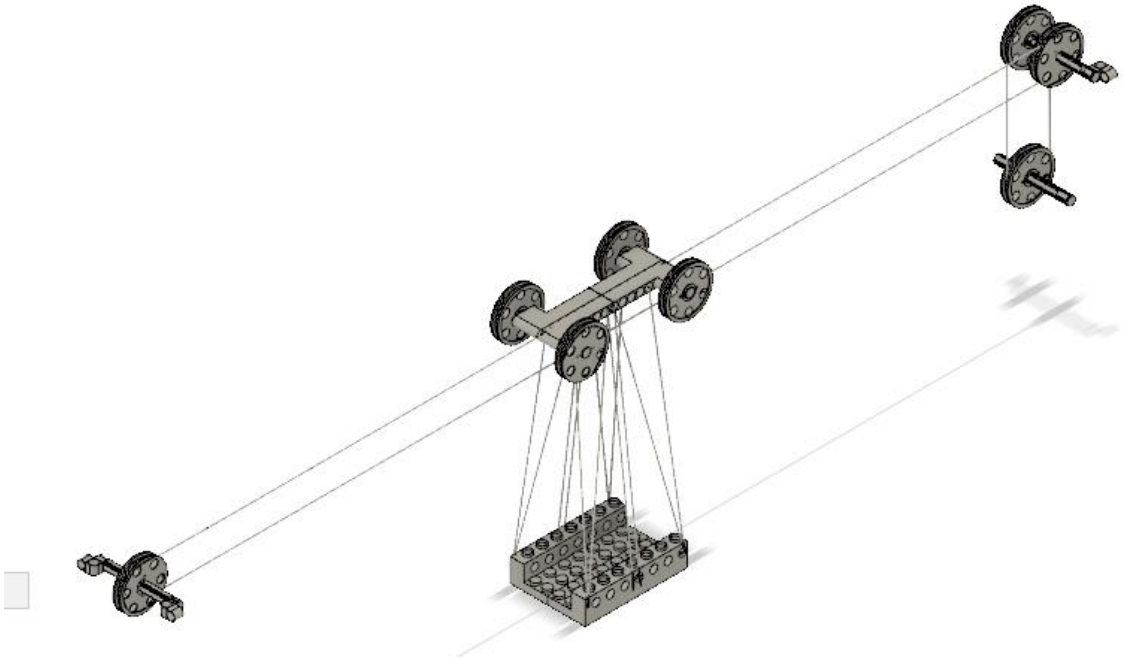

Figura 8: Mecanismo de transbordador.

- **El carrito y la barquilla**. El carrito superior se compone de cuatro ruedas iguales a las poleas y un cuerpo formado por vigas de diferentes tamaños y elementos de unión. Los cables de suspensión se simulan mediante cuerdas. Siendo la escala tan pequeña, se reducen las opciones de detallar los distintos componentes.

- **Control de la barquilla transbordadora**. Para llevar a cabo el control del movimiento horizontal del Puente Colgante, se emplea tanto componentes físicos (hardware) como lógicos (software de programación). El ladrillo utilizado es el modelo NXT de Lego y a él se conectan por cable el servomotor encargado de crear el movimiento y los diferentes sensores.

  Los sensores seleccionados son de dos tipos: el sensor de contacto y el sensor de ultrasonidos. Los sensores de contacto se encargan de hacer

llamadas desde las dos torres. Los sensores de ultrasonidos se utilizan para localizar exactamente a la barquilla. Los sensores se colocarán en las extremidades de las torres. Para programar en este lenguaje, se utiliza el software BricxCC que permite enviar distintos programas al ladrillo mediante cable USB o Bluetooth. En este caso se opta por utilizar el primer medio.

- **Lenguaje de programación**.

En cuanto a la programación se refiere, Not Exactly C (NXC) es el lenguaje de programación estructurada utilizado, de alto nivel diseñado por John Hansen específicamente para los robots de Lego. NXC se basa en Next Byte Codes, un lenguaje ensamblador que permite ser ejecutado por el microprocesador (el ladrillo en este caso) NXT.

Para programar en NXC, se dispone de Bricx Command Center (BricxCC), un entorno de programación creado por Mark Overmans. En su origen, se creó para programar en lenguaje NQC (Not Quite C) haciendo uso del anterior modelo de ladrillo, RXC. Durante los últimos años, se han realizado cambios en el entorno original con el objetivo de poder ser utilizado para el modelo NXT y el lenguaje NXC.

El entorno de programación permite escribir programas y descargarlos al robot, ejecutarlos, detenerlos, moverse por la memoria flash del robot, convertir ficheros de sonido a un formato propio del robot y otras funcionalidades.

- **Software de diseño**.

Para la elaboración del diseño previo se ha decidido utilizar el software AutoDesk Fusion 360 que incluye la aplicación AutoCAD seleccionada. Al contrario que otras aplicaciones de modelado paramétrico, esta versión integra diferentes funcionalidades como la de modelar, renderizar, animar, simular y si es necesario convertir el diseño en un archivo CAM para posterior uso.

Autodesk Fusion 360 es un paquete de modelado paramétrico de sólidos en 3D producido por la empresa de software Autodesk. Entró en el mercado en 1999, muchos años después que los anteriormente mencionados, y se agregó a las Series de Diseño Mecánico de Autodesk como respuesta de la empresa a la creciente migración de su base de clientes de diseño mecánico en dos dimensiones hacia la competencia, permitiendo que los ordenadores personales puedan construir y probar montajes de modelos extensos y complejos.

Dentro del extenso catálogo de funcionalidades, el modelado paramétrico ha sido el que se ha utilizado para desarrollar las ideas principales, diseñar el primer prototipo de Puente Colgante y el mecanismo de transmisión de movimiento de la barquilla. Asimismo, a partir del modelo 3D se han creado diferentes vistas y sus respectivas acotaciones.

2. TAREAS DESARROLLADAS EN EL TRABAJO

2.1 Descripción de tareas y fases

Para lograr los objetivos descritos anteriormente, se han realizado las siguientes tareas y actividades:

1. Estudio de viabilidad del proyecto. Estudio de la estructura y mecanismo del puente original y comparación con el material disponible.
2. Diseño básico de la estructura principal mediante el software AutoDesk Fusion 360.
3. Construcción de la maqueta física añadiendo diferentes elementos al diseño realizado anteriormente.
4. Diseño y programación del sistema de control mediante Matlab-Simulink y BricxCC (NXC) para el control de los motores y sensores del kit de Lego Mindstorms.

2.1.1 Estudio de viabilidad

Esta fase se ha descrito anteriormente en análisis de alternativas incorporada en el apartado de la memoria.

2.1.2 Diseño de la estructura

Después del análisis de viabilidad del proyecto, se pasó a materializar la idea. En esta fase, no se consideraron las vigas de Lego que cubren las barras de aluminio de Tetrix ya que estas piezas comparadas con las barras no aportan ninguna rigidez al puente. Otra de las razones de no incluir las vigas es la de simplificar el sistema.

Los diferentes componentes usados han sido: barras de aluminio, ruedas, ejes, chapas de unión, piezas de sujeción de los ejes, carrito, cables de suspensión y barquilla. Sin embargo, no todos los elementos utilizados en el modelo han sido creados manualmente ya que el mismo programa ofrece un catálogo de piezas normalizadas. De este catálogo se utilizaron cuatro tornillos de tipo 91292021 para unir las torres a pares.

Las uniones entre los distintos componentes son rígidas exceptuando la unión del carrito a los travesaños y el movimiento de los ejes respecto a las

sujeciones. En estos casos se utilizaron las uniones de deslizamiento y rotatorias para unir los distintos elementos.

En las figuras que se muestran a continuación se pueden encontrar diferentes aspectos del diseño, desde el isométrico hasta las dimensiones más relevantes. Cabe destacar que el apartado de las vistas acotadas no es del todo correcto puesto que las medidas se reflejan directamente encima del conjunto y no de cada componente. Esta decisión ha sido tomada para mayor claridad del conjunto.

La escala utilizada en los distintos es 1:4 respecto a la maqueta (no respecto al puente original).

Figura 9: Isométrico de la estructura.

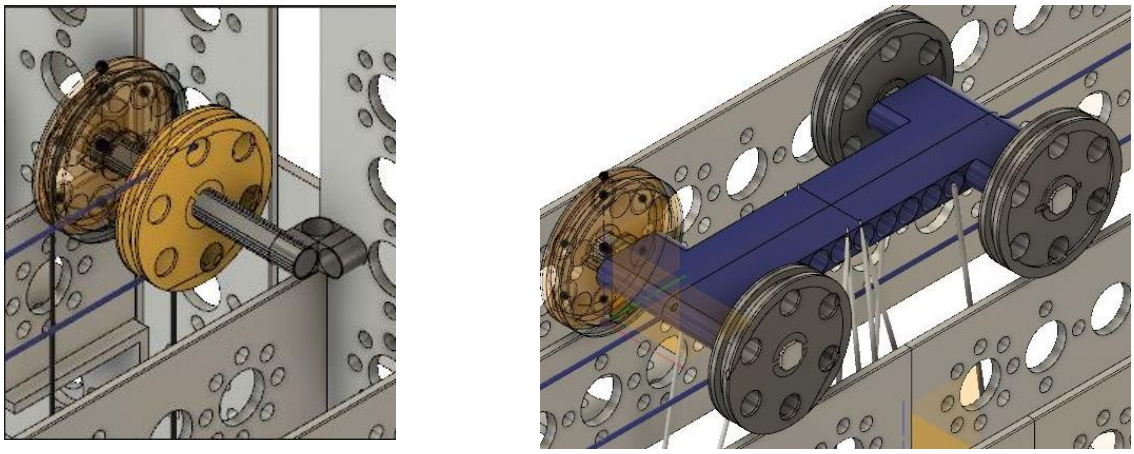

Figura 10: Detalle de las poleas y el carrito.

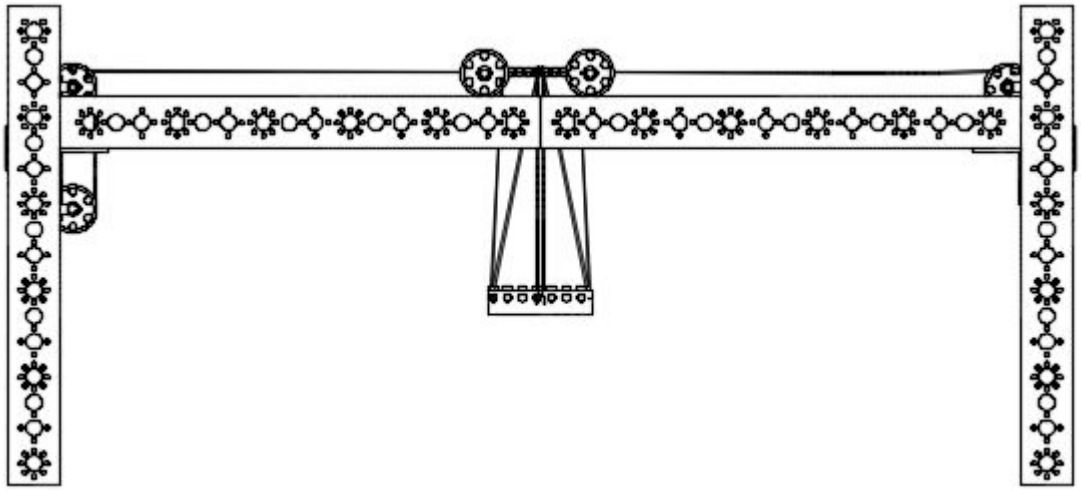

Figura 11: Vista Frontal.

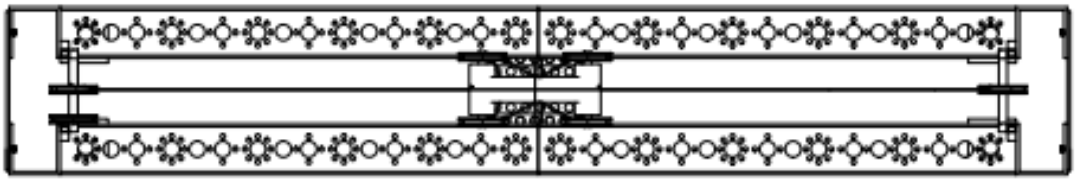

Figura 12: Vista Superior.

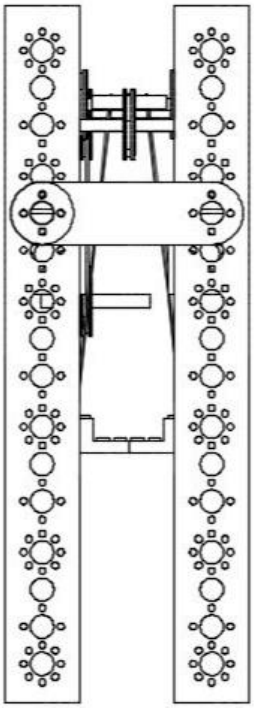

Figura 13: Vista Lateral.

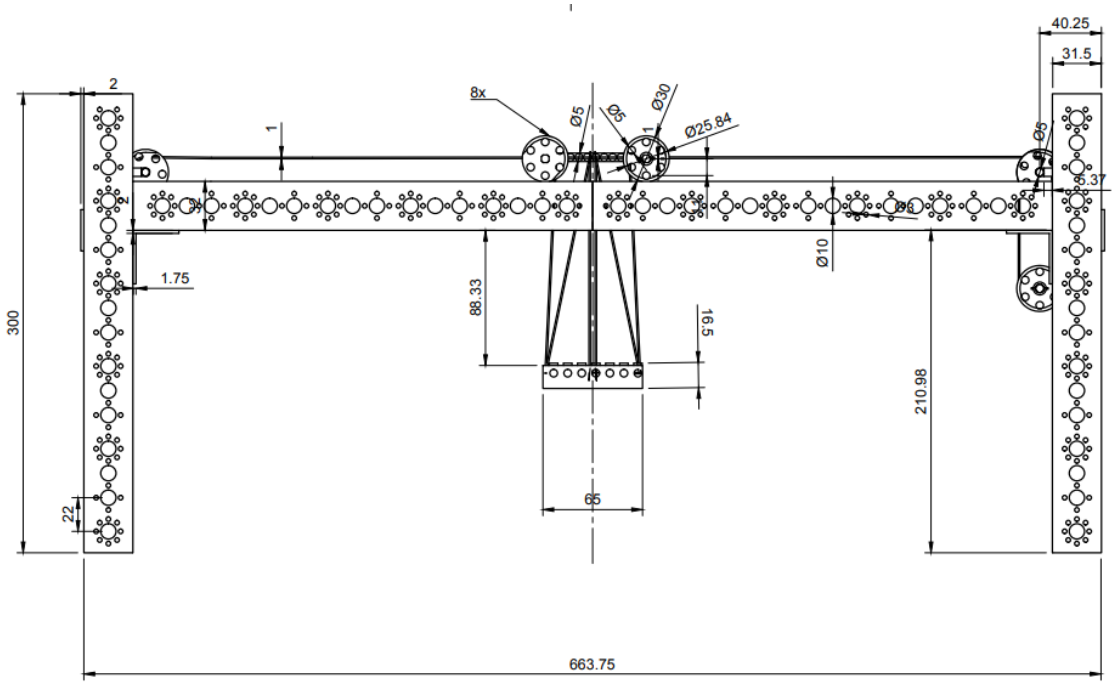

Figura 14: Vista frontal con dimensiones principales.

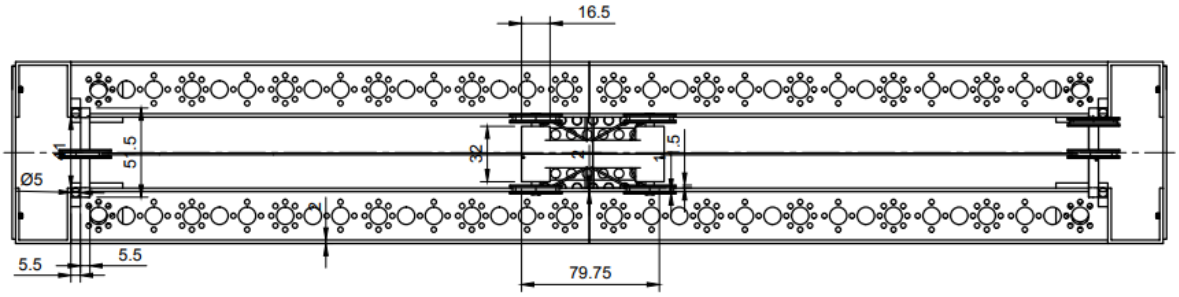

Figura 15: Vista superior con dimensiones principales.

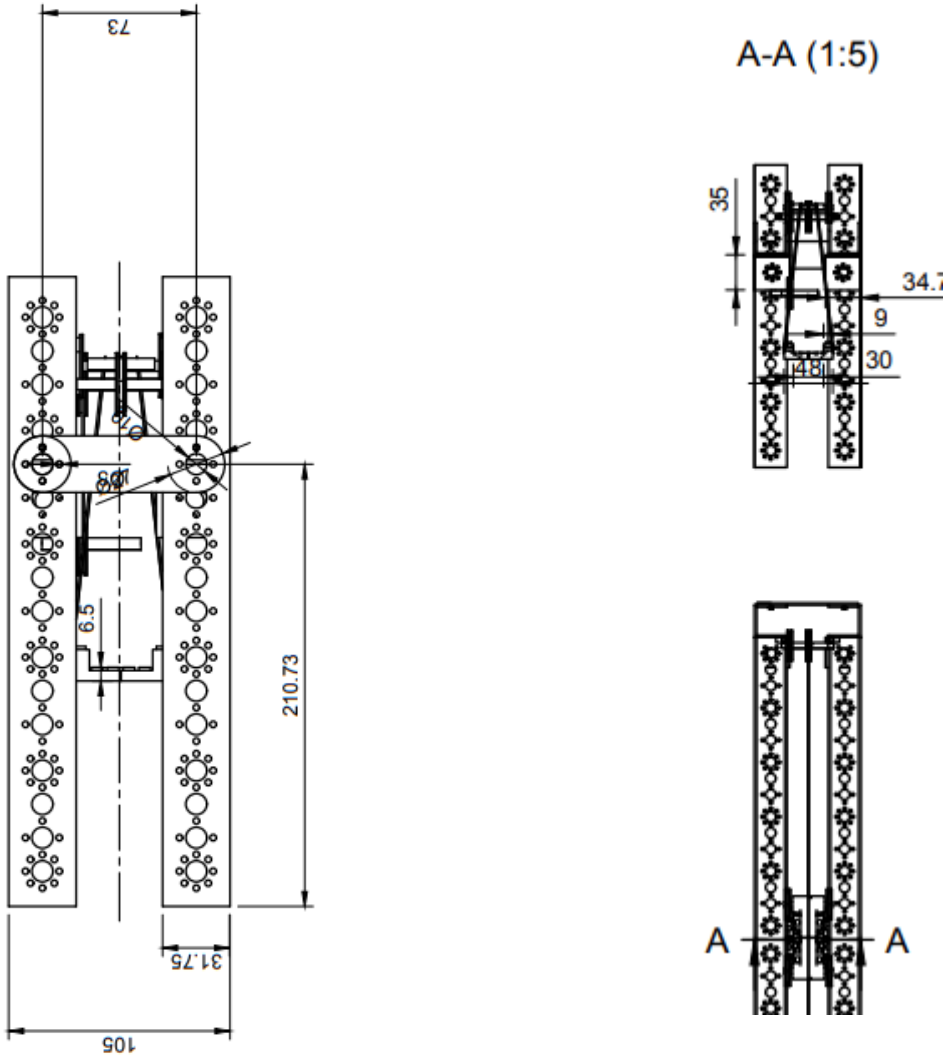

Figura 16: Vista lateral con dimensiones principales.

2.1.3 Construcción

Una vez diseñada la estructura y el mecanismo de traslación de la barquilla, el proyecto se centró en la parte del montaje.

El primer paso del montaje fue ensamblar las barras de aluminio para cimentar la base del puente. En este paso, se utilizaron barras de aluminio del kit de Tetrix, tornillos y tuercas (también encontradas en el kit anterior), chapas de unión y una pequeña llave inglesa.

El primer problema surgió cuando no se hallaban las barras del tamaño deseado. Con el fin de conseguir el tamaño requerido, se unieron barras de diferente longitud mediante tornillo-tuerca (los agujeros no tenían rosca) y apretadas mediante una llave inglesa. A continuación, se ensamblaron dos partes constituidas por dos torres con un travesaño a través de piezas de unión rectangulares. Finalmente, la estructura quedó concluida en el momento que se acoplaron las dos estructuras por medio de una chapa de aluminio y tornillo-tuercas.

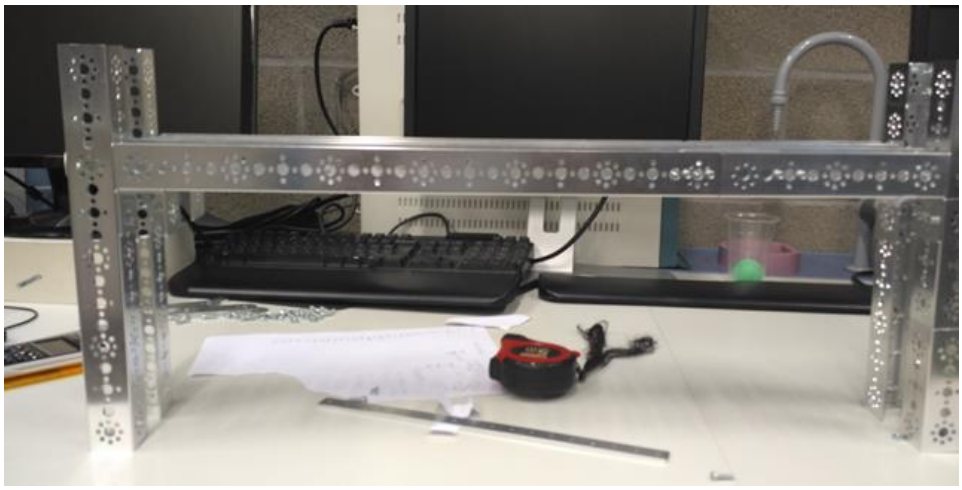

Figura 17: La estructura metálica ensamblada.

Tras construir la base del puente, llegó la hora de cubrir la estructura de metal con vigas Lego. Las torres de la estructura real se componen de cuatro plantas separadas por vigas horizontales y entre ellas tirantes de gran diámetro. Con el objetivo de representar los grandes rasgos de la estructura original, se colocaron cuatro ladrillos para crear la base y a continuación vigas verticales acopladas por piezas de unión triangulares.

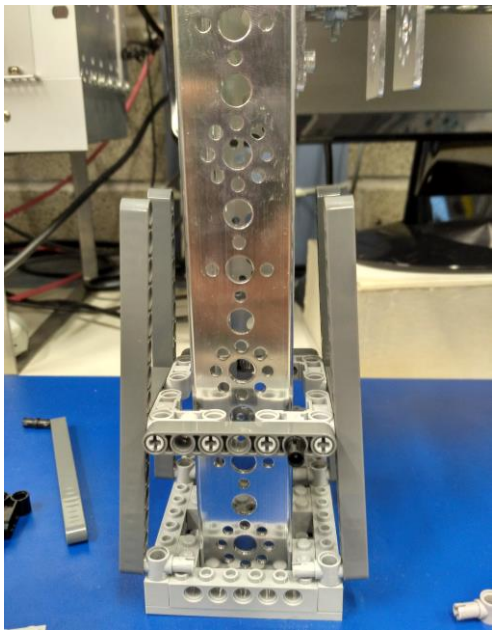

Figura 18: Construcción de una de las torres.

No obstante, cabe recalcar que en esta parte de la construcción no se ha respetado la escala establecida. La razón principal ha sido que las dimensiones de las vigas y las piezas de unión no eran compatibles con la escala, ya que la anchura de estas no permitía representar adecuadamente la característica de inclinación de las torres.

Figura 19: La estructura de vigas de Lego prácticamente terminada.

Como se puede apreciar en la fotografía anterior, el travesaño se cubrió también de vigas de menor tamaño.

Figura 20: La base de la estructura finalizada.

Una vez finalizada la base de la estructura mediante vigas, se simularon los tirantes del puente con hilo negro.

Figura 21: Torre con tirantes.

Para finalizar la construcción, se montaron las diferentes poleas junto al conjunto carrito-baquilla y se incorporaron los últimos detalles como los cables de sujeción y los cables de los travesaños:

Figura 22: La estructura finalizada.

2.1.4 Diseño y programación del sistema de control

Tras la construcción de la maqueta, se ha diseñado e implementado el sistema de control. Para ello, el primer paso ha sido analizar el sistema real, identificar las variables que se desean controlar y establecer las especificaciones que debe cumplir el sistema. Posteriormente, se han clasificado los elementos de control de la maqueta, se han obtenido los modelos matemáticos y se ha diseñado el controlador. Finalmente, se ha simulado el control en el software Matlab Simulink y se ha trasladado a la maqueta para su ejecución.

- **Análisis del sistema y especificaciones**

El carro real se acciona mediante doce motores. Asimismo, el puente real dispone de salas de embarque y una sala de control. En la maqueta construida se emplea un único motor para replicar el funcionamiento.

La variable que se desea controlar en este trabajo es la velocidad de translación de la barquilla. No obstante, en un control más sofisticado del control del trayecto de la barquilla se deberían de tener en cuenta otras variables como son los ángulos de balanceo. Se ha optado por un sistema de control realimentado donde el actuador es el servomotor NXT.

La perturbación principal es el viento, tanto su fuerza como su dirección. Su influencia se debería tener en cuenta al final del bucle, sumando a la señal de salida la perturbación. El sistema de control realimentado disminuiría el efecto de la perturbación en la señal de salida. No obstante, cn el objetivo de minimizar al máximo el efecto del viento se optaría por otro sistema de control diferente del cursado durante el grado. En este trabajo, con el propósito de simplificar lo máximo posible el problema, no se ha considerado ninguna perturbación.

Tras analizar el sistema real, simplificar el funcionamiento e identificar la variable a controlar, se han establecido las siguientes especificaciones:

- Error nulo en régimen estacionario ante entrada escalón.
- Mp (sobreimpulso)<%3 con el objetivo de no sufrir grandes oscilaciones en la velocidad.

Las especificaciones presentadas deben cumplirse al aplicar una señal de referencia que se compone de tres partes: una rampa con pendiente positiva (arranque con aceleración constante), un tramo de velocidad constante (pendiente cero) y una rampa con pendiente negativa (parada con aceleración constante).

Las razones principales de elegir las rampas han sido dos: el arranque brusco de la barquilla causaría balanceo afectando directamente al confort de los pasajeros, y un cambio brusco de la señal de voltaje de entrada al motor podría causar picos que podrían afectar a la vida útil del motor.

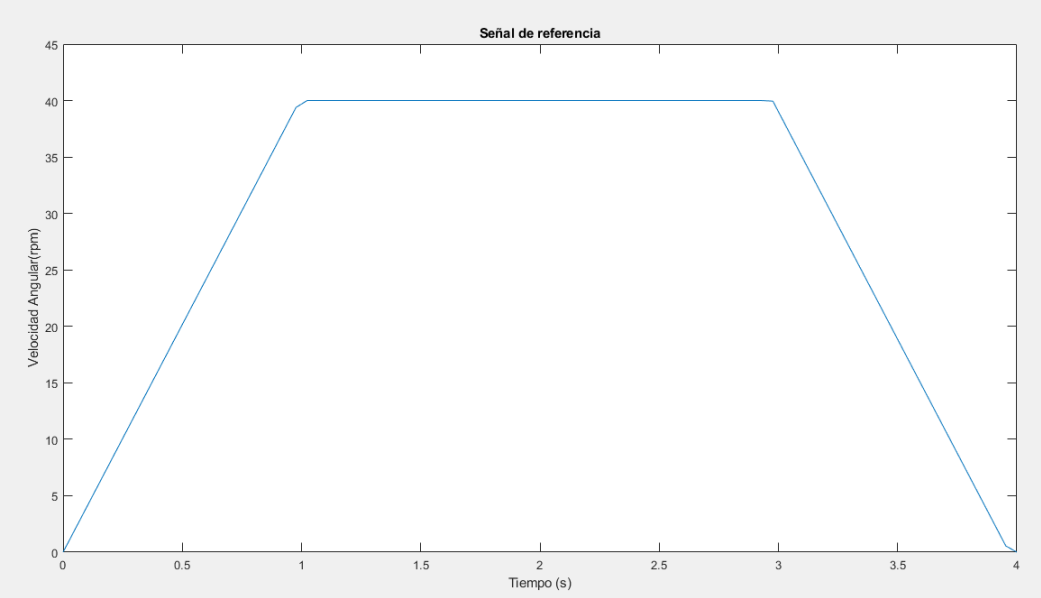

Figura 23: Señal de referencia.

Como se puede apreciar en el Figura 23, la duración del trayecto de la barquilla es de 4 segundos, con arranque y parada de 1 segundo. Los valores de las velocidades angulares se han escogido para que el trayecto sea apreciable para el observador.

- **Configuración del sistema de control realimentado**

    1. Actuador-Planta. Se compone del servomotor con su carga mecánica (barquilla).
    2. Sensor. Tacómetro que calcula el ángulo recorrido por el eje de salida.
    3. Controlador. Implementado en el microprocesador mediante el software Brixcc.

- **Obtención de modelos matemáticos**

Después de analizar el sistema y configurar el sistema de control, el siguiente paso es la obtención del modelo del sistema físico. En este caso, la obtención de las funciones de transferencia se ha realizado mediante experimentación.

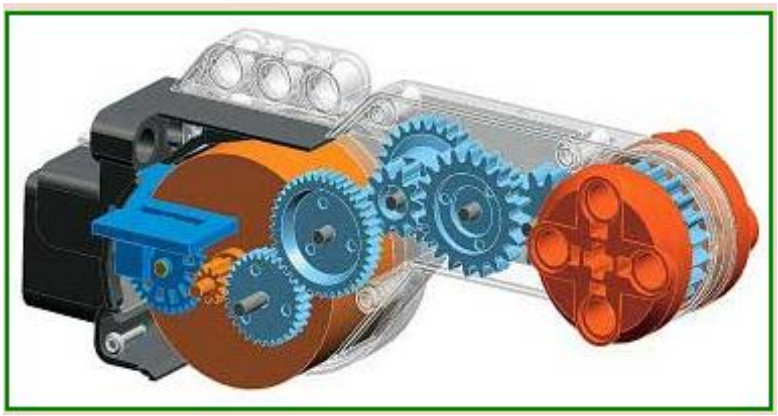

Figura 24: Motor Lego NXT.

Antes de obtener la función de transferencia del conjunto actuador-planta, se han realizado diferentes pruebas enviando al motor diferentes valores de voltaje (con carga mecánica) para obtener la curva que relaciona el voltaje aplicado y el valor estacionario de la velocidad angular. La pendiente de este gráfico refleja la ganancia que tiene el conjunto. Asimismo, se observa que el comportamiento del conjunto es lineal.

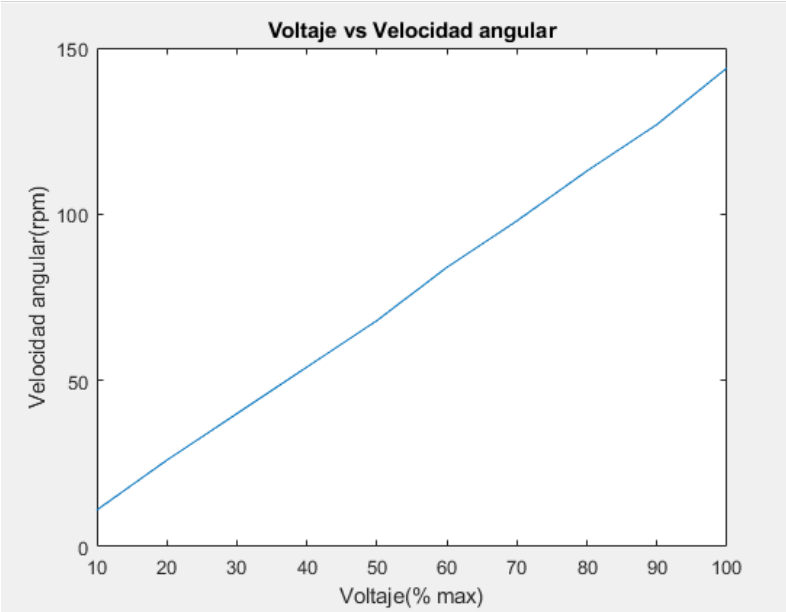

Figura 25: Voltaje (%) vs Velocidad Angular de Salida (rpm).

Tras la prueba inicial, se alcanzó la función de transferencia de tal forma que la recogida de datos (Anexo I) se produjo con una entrada escalón de valor u=20%. Los valores obtenidos se han representado en el siguiente gráfico:

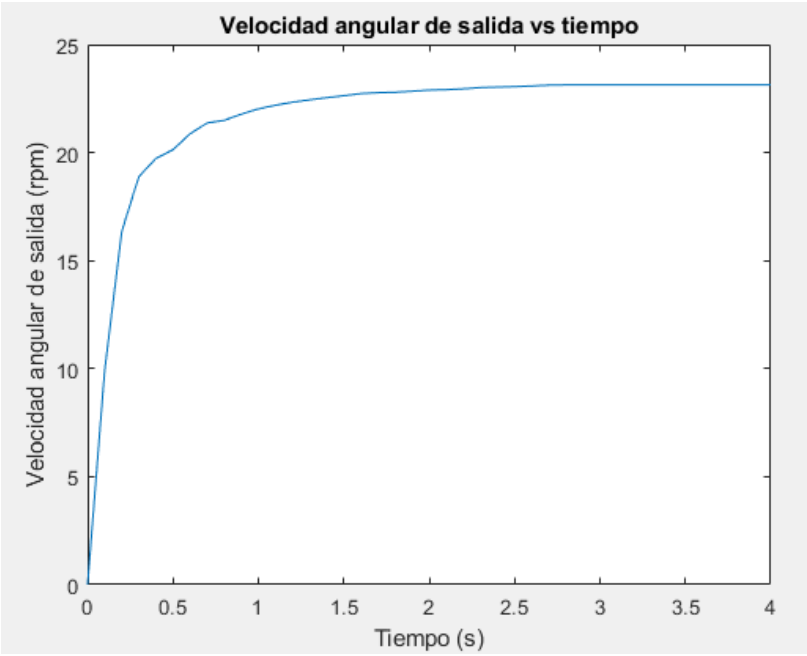

Figura 26: Velocidad Angular de Salida (rpm) vs Tiempo (s).

La función de transferencia obtenida se asemeja a un sistema de primer orden, puesto que posee un aspecto semejante: un valor estacionario en régimen permanente y sin oscilaciones en régimen transitorio. La pendiente inicial descarta la opción de modelizar el sistema mediante un sistema de segundo orden. La forma general de la función de transferencia de primer orden:

$$\mathrm{Gp(s)} = \frac{K}{\tau s + 1}\,(\mathrm{rpm}/\%)$$

A partir del gráfico se obtienen los dos parámetros necesarios:

$$K = \frac{23.14}{20} = 1.157$$

$$\tau = t(0.63 * \Delta y) = 0.23447s$$

$$Gp(s) = \frac{1.157}{0.23447s + 1}$$

Una vez obtenido el modelo del actuador-planta, el sensor se ha modelizado mediante una ganancia unitaria.

- **Diseño del controlador y Simulación**

Antes de realizar pruebas en el sistema real, se han realizado simulaciones utilizando el software Matlab-Simulink. Para ello, se ha creado el sistema de control con realimentación unitaria formado por dos bloques: el controlador continuo PID(s) y la planta.

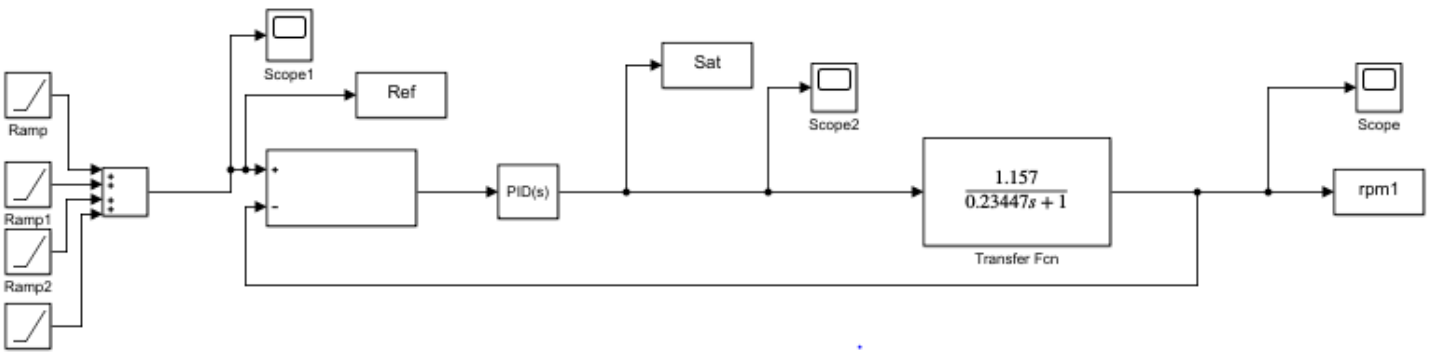

Figura 27: Control lazo cerrado (Simulink).

El controlador PID(s) se representa de siguiente manera y los parámetros a especificar son Kp, Kd y Ki:

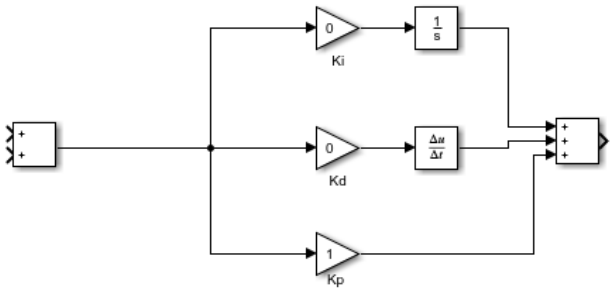

Figura 28: Controlador PID(s).

Teniendo en cuenta las especificaciones, se ha escogido un controlador PI(s). El error estacionario nulo a entrada escalón exige acción integral. La acción derivativa puede causar grandes picos en sistemas con mucho ruido (sensores de poca calidad) y por tanto se descarta su uso.

La sintonización del PI se ha llevado a cabo mediante la opción "Tune" que incorpora el mismo bloque PID escogido. Este comando incorpora la opción de escoger entre diferentes especificaciones y sintoniza las constantes necesarias para cumplirlas. El resultado de la sintonización imponiendo las especificaciones del control queda:

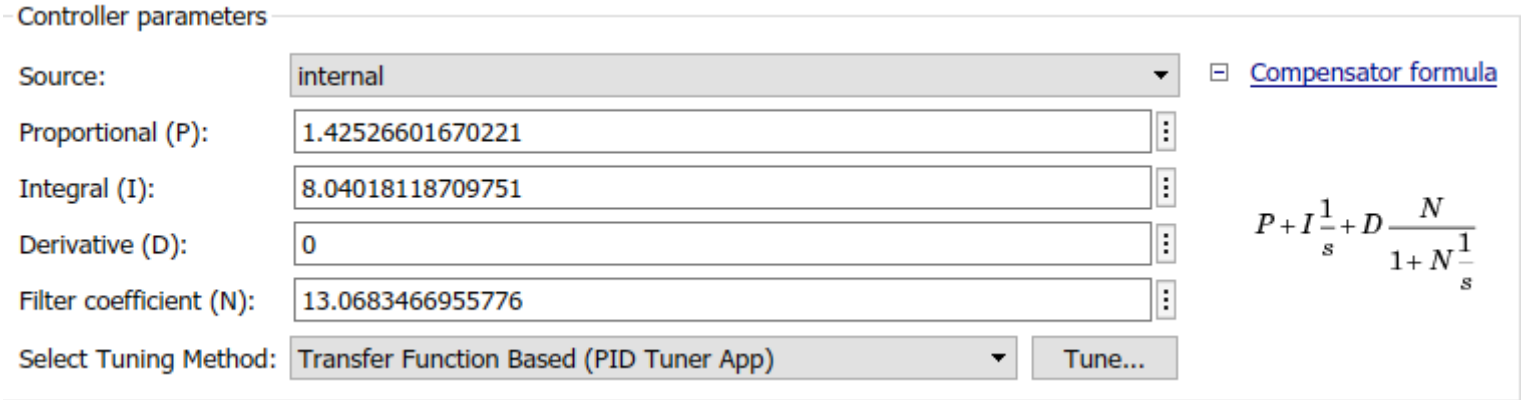

Figura 29: Parámetros del controlador.

Basado en los resultados obtenidos en la sintonización del controlador, la función matemática es la siguiente:

$$PI(s) = 1.4253 + \frac{8.04018}{s}$$

Después de sintonizar el controlador, se iniciaron las simulaciones y se representaros dos gráficos: la señal de referencia y salida en lazo cerrado vs tiempo(s), y la señal de control para ver si se produce saturación.

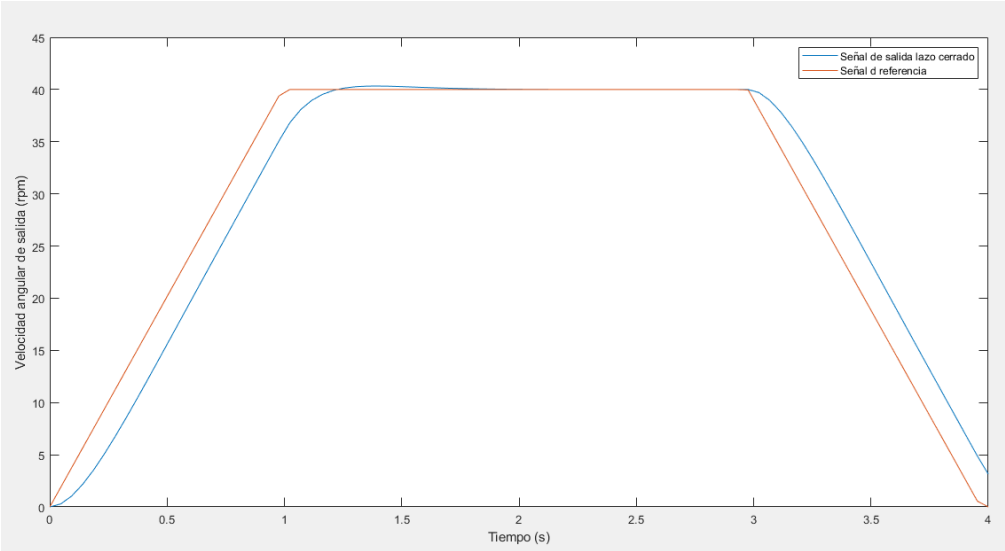

Figura 30: Señales de salida y referencia.

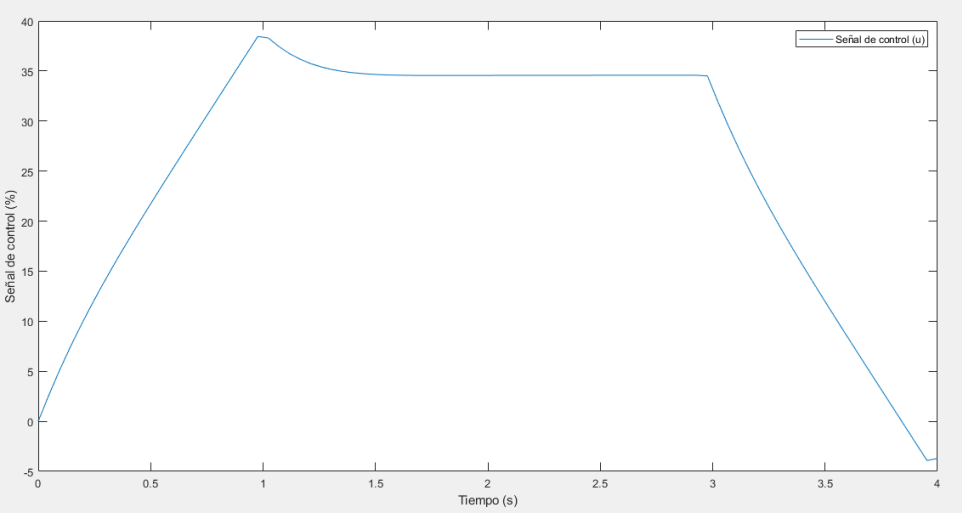

Figura 31: Señal control.

Con el objetivo de asegurar que las especificaciones se cumplen, se ha prestado atención al vector de salida en el workspace. En él se ha observado que el valor máximo de salida es 40.3305, es decir, el Mp=0.3304*100/40=0.826%(<3%). Asimismo, en el tramo de pendiente cero, el error en régimen permanente es cero ya que los valores de los vectores de referencia y salida coinciden. En cuanto a la

señal de control, se puede apreciar que todos los valores obtenidos se encuentran entre +-100, es decir, no existe un problema de saturación. Asimismo, la señal de control es suave y no origina ningún problema adicional.

- **Traslado del control al sistema real**

Una vez comprobado en Simulink, se ha programado el controlador en BricxCC (ver Anexo I). El controlador implantado es el siguiente:

```
// Implementación PI
  error = desiredRpm - motorRpm;
  integralOfError = prevError + error;
  motorCommand = Kp*error + Ki*integralOfError;
  if(motorCommand >= 100.0) motorCommand = 100.0;
  if(motorCommand <= 0.0) motorCommand = 0.0;
  OnFwd(MOTOR, motorCommand);
```

Figura 32: Controlador PID NXC.

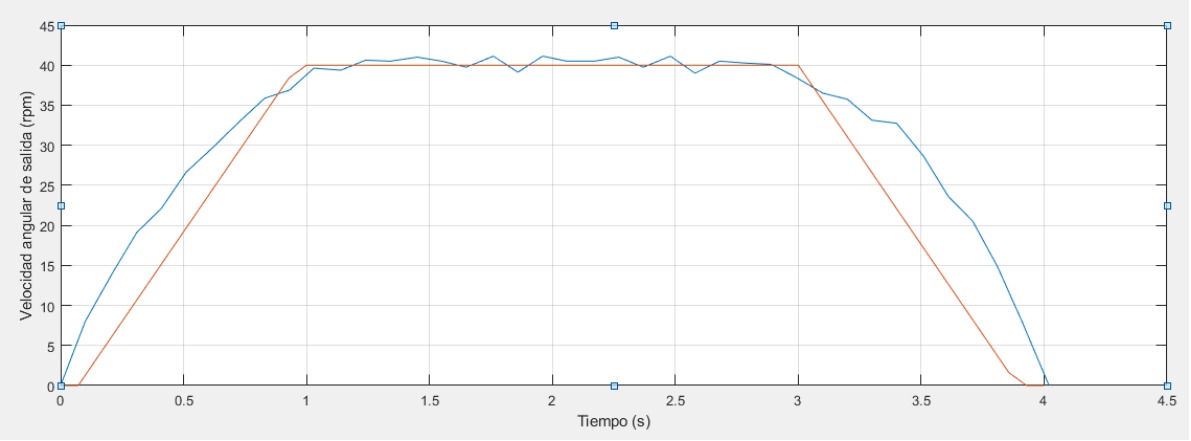

Figura 33: Señal de salida implementando el controlador.

La imagen superior representa el funcionamiento real del sistema al aplicar el controlador diseñado. Tal y como se puede apreciar, la señal de salida cumple razonablemente las especificaciones impuestas. La condición del sobreimpulso máximo se cumple puesto que el valor máximo que se alcanza es 41.125, es decir, MP=1.125*100/40=2.81% (<3%). No obstante, en la región de pendiente nulo se puede observar que el error no es cero. Existe un rizado y el valor medio de la señal en esa región es de 40.5 aproximadamente (ess=1.125%). Estas variaciones pueden ser debidas a la falta de precisión del tacómetro (fuente principal del ruido) y las imprecisiones de los componentes mecánicos que forman el servomotor, puesto que son elementos de plástico y de media-baja calidad.

Parada de emergencia

Completando el sistema de control anterior, se ha incluido otra una condición para el inicio del trayecto de la barquilla. El objetivo impedir que la barquilla se ponga en marcha si se detecta el paso de cualquier nave o aeronave que suponga un riesgo para la seguridad del funcionamiento del puente. En el sistema real, la detección de naves o aeronaves se lleva a cabo mediante radares que envían señales a la sala de control. En la maqueta construida, se hace uso de los sensores de ultrasonidos para replicar este aspecto de seguridad. El funcionamiento se basa en detectar cualquier objeto mediante los dos sensores ultrasonidos colocados a los lados. En condiciones normales de funcionamiento si la barquilla está situada en una de las dos torres, la distancia calculada por los sensores es un valor concreto. No obstante, cuando un objeto es colocado entre las dos torres, la distancia calculada varía e impide el inicio del trayecto, emitiendo un sonido agudo que indica la necesidad de no iniciar el transporte. Ver Anexo I.

2.2 Planificación del proyecto. Diagrama de Gantt

A continuación, se presenta la planificación del proyecto desde el principio hasta el final. En primer lugar, se presenta una lista con las diferentes tareas realizadas a lo largo del proyecto y después el diagrama de Gantt correspondiente.

La fecha de inicio del proyecto es el 2 de Febrero de 2018 y la fecha de finalización el 23 de Julio de 2018. El ritmo de trabajo ha sido irregular a lo largo del periodo del proyecto a causa de los periodos de exámenes y diferentes trabajos docentes de las demás asignaturas. Por ello, se han tenido en cuenta los siguientes periodos a la hora de realizar la planificación y la suma total de los días en cada tarea(se han computado estos períodos aunque no se han trabajado en ellos):

- Del 28 de Marzo de 2018 hasta el 8 de abril de 2018.
- Del 20 de mayo de 2018 hasta el 5 de junio de 2018.

2.2.1 Listado de tareas

El proyecto en su totalidad está compuesto por una serie de tareas principales que se dividen, a su vez, en tareas de segundo orden. Las tareas principales se han introducido como hitos. El proyecto en su totalidad se ha realizado conjuntamente entre el alumno y su director del proyecto. A continuación, se enumeran las distintas tareas de la planificación:

1. **Planteamiento del problema**. En este apartado se incluyen todas las actividades relacionadas con la familiarización del problema. La fuente de información principal para este apartado ha sido Internet, las páginas oficiales de las herramientas utilizadas y webs de artículos relacionados con el proyecto. La duración total de esta labor fue de 20 días, comenzando el 2 de febrero de 2018. Las subtareas llevadas a cabo fueron:

- Documentación.
- Análisis del estado del arte.
- Análisis de alternativas y estudio de viabilidad.
- Análisis del material disponible y su clasificación.

2. **Diseño previo.** Tras realizar el estudio de viabilidad, se desarrolló la alternativa escogida. Para ello, se utilizó el programa AutoDesk Fusion 360. La duración total de esta tarea fue de 26 días, comenzando el 3 de marzo de 2018. Las subtareas llevadas a cabo:
   - Descarga e instalación.
   - Visualización de tutoriales y manuales.
   - Diseño básico 3D y sus respectivos planos.
3. **Construcción de la maqueta.** Una vez terminado el diseño por ordenador, se llevó a cabo el ensamblaje de la maqueta. La mayor parte de esta tarea se realizó en el laboratorio del Departamento de Automática y Control de la Escuela de Ingeniería de Bilbao. El material utilizado en esta sección se puede encontrar en el apartado listado de materiales. La duración total de esta tarea fue de 30 días, comenzando el 4 de Abril de 2018. Las subtareas llevadas a cabo fueron:
   - Ensamblaje de la estructura metálica.
   - Cubrir la estructura mediante las vigas Lego.
   - Construcción de la barquilla-carrito.
   - Insertar el mecanismo de movimiento de traslación.
   - Últimos detalles.
4. **Programación**. Haciendo uso del lenguaje NXC, se diseñó y programó el controlador de velocidad angular del motor, y el control de traslación de la barquilla que componen la maqueta. Para ello, se hizo uso de las aplicaciones BricxCC y MATLAB-Simulink. La duración total de esta tarea fue de 36 días, comenzando el 21 de Mayo. Las subtareas llevadas a cabo en esta sección fueron:
   - Descarga e instalación del software.
   - Visualización de tutoriales y manuales.
   - Programación de la maqueta.
5. **Redacción de la memoria.** Consiste en el tiempo empleado en redactar todo el trabajo realizado en tareas previas y generando un documento que se amolde a las directrices marcadas por la normativa del TFG. Las herramientas utilizadas han sido Microsoft Word y Microsoft Project, para la edición del texto y el diagrama de Gantt respectivamente. La duración total de esta tarea fue de 10 días iniciado en el 10 de Julio.

A continuación se muestra el diagrama de Gantt correspondiente:

Planteamiento del problema	**20 days**	**Fri 2/2/18**	**Thu 3/1/18**
Documentación	2 days	Fri 2/2/18	Mon 2/5/18
Historia del arte	2 days	Tue 2/6/18	Wed 2/7/18
Estudio de viabilidad	13 days	Thu 2/8/18	Mon 2/26/18
Análisis del material disponible	3 days	Tue 2/27/18	Thu 3/1/18
Diseño previo	**26 days?**	**Fri 3/2/18**	**Fri 4/6/18**
Descarga e instalación de Fusion 360	1 day?	Fri 3/2/18	Fri 3/2/18
Visualizació de tutoriales y manuales	10 days	Mon 3/5/18	Fri 3/16/18
Diseño básico 3D y sus respectivos planos	15 days	Mon 3/19/18	Fri 4/6/18
Construcción de la maqueta	**30 days**	**Mon 4/9/18**	**Fri 5/18/18**
Ensamblaje de la estructura metálica	10 days	Mon 4/9/18	Fri 4/20/18
Cubrir la estructura de vigas Lego	8 days	Mon 4/23/18	Wed 5/2/18
Construcción de la barquilla-carrito	5 days	Thu 5/3/18	Wed 5/9/18
Insertar el mecanismo de transbordo	4 days	Thu 5/10/18	Tue 5/15/18
Últimos detalles	3 days	Wed 5/16/18	Fri 5/18/18
Programación	**36 days?**	**Mon 5/21/18**	**Mon 7/9/18**
Descarga e instalación Brixcc	1 day?	Mon 5/21/18	Mon 5/21/18
Visualizació de tutoriales y manuales	15 days	Tue 5/22/18	Mon 6/11/18
Programación de la maqueta	20 days	Tue 6/12/18	Mon 7/9/18
Redacción de la memoria	**10 days**	**Tue 7/10/18**	**Mon 7/23/18**
Redacción de la memoria	10 days	Tue 7/10/18	Mon 7/23/18

Figura 34: Diagrama de Gantt.

3. ASPECTOS ECONÓMICOS. PRESUPUESTO EJECUTADO

En este apartado se expone el gasto realizado en este TFG, dividido en cuatro apartados: relacionados con el montaje, relacionados con el hardware de Lego Mindstorms, software utilizado y mano de obra. En todos los precios mostrados el IVA está incluido.

3.1 Gastos relacionados con el montaje

Producto	Precio/Ud.	Unidades	Precio
Barra Tetrix de aluminio (416mm)	17,99 €	2	35,98 €
Barra Tetrix de aluminio (160 mm) Pack de 2	14,53 €	3	43,59 €
Barras Tetrix de aluminio (100mm) Pack de 2	11,99 €	2	23,98 €
Tuercas y tornillos kit Tetrix (Pack 50)	11,99 €	1	11,99 €
LEGO Technic Light Gray 1x15 agujeros Mindstorms Parte 32277 (Cantidad 8)	8,53 €	7	59,71 €
LEGO Technic Light Gray 1x7 agujeros 32275 (Cantidad 8)	5,52 €	2	11,04 €
LEGO Technic Light Gray 1x5 agujeros 32273 (Cantidad 8)	4,53 €	8	36,24 €
LEGO Technic Light Gray 1x3 agujeros 32271 (Cantidad 8)	3,59 €	3	10,77 €
LEGO Technic 10 piezas LIGHT GREY CONNECTOR 3x3 BENT 4	10,97 €	5	54,85 €
LEGO Technic 4 piezas CHASIS MARCO LIFTARM BeamStudless 5x7	10,24 €	1	10,24 €
Neumático 24mmX7mm, Rueda y Ejes Largos -Pack 4	9,86 €	1	9,86 €
Cordón de algodón	1,50 €	1	1,50 €
		Subtotal 1	309,75 €

Tabla 4: Presupuesto Apartado 3.1.

3.2 Gastos Hardware Lego Mindstorms

Producto	Precio/Ud.	Unidades	Precio
Ladrillo (Microprocesador) NXT	69,99 €	1	69,99 €
LEGO MINDSTORMS Interactive Motor (9842)	25,70 €	1	25,70 €
LEGO Mindstorms NXT Sensor de Ultrasonidos (9846)	21,41 €	2	42,82 €
LEGO MINDSTORMS Sensor de contacto(9843)	7,71 €	2	15,42 €
Cables de conexión (Pack 5)	14,99 €	1	14,99 €
		Subtotal 2	168,92 €

Tabla 5: Presupuesto Apartado 3.2.

3.3 Mano de obra

Concepto	Precio/H.	Horas	Precio
Mano de obra	8,00 €	150	1.200,00 €
		Subtotal 3	1.200,00 €

Tabla 6: Presupuesto Apartado 3.3.

3.4 Software utilizado

Software Utilizado	€/H.	H.	Gasto
AutoDesk Fusion 360	20	50	1.000,00 €
MATLAB 2018	15	20	300,00 €
Microsoft Project	0,83	20	16,60 €
Microsoft Word	0,83	60	49,80 €
Microsoft Excel	0,83	10	8,30 €
		Subtotal 4	1.374,00 €

Tabla 7: Presupuesto Apartado 3.4.

3.5 Presupuesto ejecutado

Presupuesto ejecutado	Gasto
Subtotal 1	309,75 €
Subtotal 2	168,92 €
Subtotal 3	1.200,00 €
Subtotal 4	1.374,00 €
Total	3.053,37 €

Tabla 8: Presupuesto final.

4. CONCLUSIONES

Este TFG ha sido una buena oportunidad para poner en práctica conocimientos adquiridos durante el grado. En él se integran conceptos trabajados en diferentes asignaturas con un objetivo común. Ejemplo de esto es el diseño del sistema de control mediante la utilización de MATLAB-Simulink, software trabajado en el aula y puesto ahora en práctica, para después trasladarlo a código NXC donde se utiliza la programación estructurada.

Una de las aportaciones más importantes ha sido el de enfrentarse a un problema de ingeniería y ser capaz de resolver todos los problemas encontrados con solvencia; el diseño se ha realizado utilizando el software Autodesk Fusion 360 del que no se tenía conocimiento. Estos retos sirven de experiencia para situaciones en las que el/la ingeniero/a se encuentra habitualmente.

Por último, ha sido muy satisfactorio realizar un trabajo de principio a fin, es decir, comenzar un diseño, continuar con su construcción y finalmente ponerlo en marcha pasando por todas sus fases.

5. BIBLIOGRAFÍA

[1] Daniele Benedettelli revisado por John Hansen (2007), *Manual_para_programar_Robots_Lego_Mindstorms_NXT* [Formato PDF], Billund, Dinamarca, Lego Inc.

[2] John Hansen .NXC Version 1.2.1 r5 Programmer's Guide (2010). Consulta [24/03/2018]. Disponible: http://bricxcc.sourceforge.net/nbc/nxcdoc/nxcapi/group__button_module.html

[3] Álvaro García. Aspectos técnicos e innovación. Puente Colgante (2018). Consulta [21/02/2018]. Disponible online: http://puente-colgante.com/aspectos-tecnicos-e-innovacion

[4] Dave Astolfo-Mario Ferrari-Giulio Ferrari (2007), *Building robots with Lego Mindstroms NXT,* Burlington, Estados Unidos, Syngress Pubilshing Inc

[5] *Soporte* Fusion 360. Autodesk (2015). Consulta [03/05/2018]. Disponible online: https://www.autodesk.es/products/fusion-360/subscribe?plc=F360B&term=1-YEAR&support=ADVANCED&quantity=1

[6] Normativa trabajo fin de grado. Escuela de Ingeniería de Bilbao(2017). Consulta [27/01/2018]. Disponible: https://www.ehu.eus/eu/web/ingeniaritza-bilbo/graduen-araudiak

[7] Lego Mindstorms Downloads. Lego(2012). Consulta [25/04/2018]. Disponible online: https://www.lego.com/en-us/mindstorms/downloads/download-software

[8] Tetrix Products Beams. Tetrix Robotics (2018). Consulta [05/06/2018] Disponible online: https://www.tetrixrobotics.com/Structural-Elements/Square-Beams/TETRIX-PRIME-Square-Beams

[9] Puente de Vizcaya. Wikipedia (2018). Consulta [12/12/2017]. Disponible online: https://es.wikipedia.org/wiki/Puente_de_Vizcaya

[10] Matlab Answers. MathWorks (2018). Consulta [05/05/2018]. Disponible online: https://es.mathworks.com/support.html?s_tid=gn_supp

[11] Pinterest. Golden Gate Bridge(2014). Consulta[16/07/2018]. Disponible online: https://www.pinterest.es/pin/515240013605400133

Deviantart. London Bridge(2013). Consulta[16/07/2018]. Disponible online: https://www.deviantart.com/botskey/art/Lego-London-Tower-Bridge-353812082

[12] ElCorreo. Tusfotos. Maqueta Puente Colgante (2012). Consulta[16/07/2018]. Disponible online: http://tusfotos.elcorreo.com/vizcaya/fotos-vibe/maqueta-puente-colgante-1107557.html

Youtube. Puente Colgante Mi Nueva Maqueta.(2018). Consulta[16/07/2018]. Disponible online: https://www.youtube.com/watch?v=tHtX4wibcr0

ANEXO I: Código en lenguaje NXC

Programa para el control de la velocidad angular del motor.

Control PID y recogida de datos (común para la obtención de la función de transferencia y el control del PID)

```
#define MOTOR OUT_A // Asignar a Motor la entrada A
#define DEG2RPM 166.667 // deg/msec a RPM
#define RPM2RADPERSEC 0.105; // RPM a rad/sec

// Variables globales para crear un fichero
unsigned int result;
byte fileHandle;
short bytesWritten;
string fileHeader;
int fileNumber, filePart;
string fileName;
string strFileNumber;
string strFilePart;
string text;

// Crear y inicializar el archivo
void InitWriteToFile() {
 fileNumber = 0;
 filePart = 0;
 fileName = "nxtMotorData.csv" ; // Nombre del fichero de datos
 result=CreateFile(fileName, 60000, fileHandle);

 // Chequear si ya existe el archivo
 while (result==LDR_FILEEXISTS)
   {
    CloseFile(fileHandle);
    fileNumber = fileNumber + 1; // crear nuevo si previamente existe
    fileName=NumToStr(fileNumber);
    fileName=StrCat("nxtMotorData" , fileName, ".csv");
    result=CreateFile(fileName, 60000, fileHandle);
    } // end while

 PlayTone(TONE_B7, 5);
 fileHeader = "Tiempo(s), Velocidad angular(RPM)" ; // títulos
 WriteLnString(fileHandle, fileHeader, bytesWritten);

}
```

```
void WriteToFile(string strTempText) {
 // función para recoger los datos en forma de string

 result=WriteLnString(fileHandle, strTempText, bytesWritten);
 if (result==LDR_EOFEXPECTED) // LDR_EOFEXPECTED is flagged when end-of-file
   {
   CloseFile(fileHandle);

   filePart = filePart + 1;
   strFileNumber = NumToStr(fileNumber);
   strFilePart = NumToStr(filePart);
   fileName = StrCat("nxtMotorData" , strFileNumber,"-", strFilePart ,".csv");
   DeleteFile(fileName);
   CreateFile(fileName, 1024, fileHandle);
   PlayTone(TONE_B7, 5);
   WriteLnString(fileHandle, strTempText, bytesWritten);
   }
}

void StopWriteToFile() {
 // cerrar el archivo
 CloseFile(fileHandle);
}

task main() { //programa principal
 // Declaración de variables relacionados con el motor
 long prevAngleInDegrees; // ángulo en deg del motor previo
 long curAngleInDegrees; // ángulo actual en deg del motor
 long deltaAngleInDegrees; // diferencia de ángulo
 string strDeltaAngleInDegrees; // string de ángulos
 float motorRpm; // velocidad del motor [RPM]
 string strMotorRpm; // string de los valores de velocidad
 float motorRadPerSec; // velocidad del motor [rad/s]
 string strMotorRadPerSec; // string de velocidad del motor [rad/s]

 // Variables PID
float Kp, Ki; // constantes de PID
float desiredRpm; // valor de referencia
float error;
float prevError;
float deltaError;
float derivativeOfError;
float integralOfError;
float motorCommand; // señal u

 // Timing related variables
 long prevTick; //valor anterior
 long curTick; // valor actual
```

```
long deltaT; // la diferencia entre la anterior y la actual ticks
string strDeltaT; // string de DelT
float elapsedTimeInSeconds; // tiempo en s
string strElapsedTimeInSeconds; // string de tiempo de muestra en s
long firstTick; // valor al principio
long timePass;

//Creacion de valiables utilizadas para la recogida de datos
long deltaTWrite; //time to write
long  prevAngleInDegrees1;
long curAngleInDegrees1;
long deltaAngleInDegrees1;
string strDeltaAngleInDegrees1;
long motorRpm1;
string strMotorRpm1;

// Variables relacionados con los botones
bool orangeButtonPushed, rightArrowButtonPushed;

// Crear archivo para recogida de datos de motor
InitWriteToFile();

// Inicializar variables
elapsedTimeInSeconds = 0.0;
prevAngleInDegrees = 0;
deltaError = 0.0;
prevError = 0.0;
derivativeOfError = 0.0;
integralOfError = 0.0;

//Inicialiación de valiables utilizadas para la recogida de datos
deltaTWrite=0.0;
prevAngleInDegrees1=0.0;
 elapsedTimeInSeconds1 = 0.0 ;

// Inicializar variables PID
Kp = 1.42526;
Ki = 8.04 ;

// La condición para iniciar
TextOut (0, LCD_LINE1, " Pulsar el botón derecho para comenzar." );
do { // esperar hasta que se pulse el botón derecho
rightArrowButtonPushed = ButtonPressed(BTNRIGHT, FALSE);
} while(!rightArrowButtonPushed);
prevTick = CurrentTick();
TextOut (0, LCD_LINE1, "Pulsar boton naranja para abandonar." );
```

```
firstTick=CurrentTick();
do {
 timePass=CurrentTick()-firstTick;
 if (timePass<1000) { // Señal de referencia
  desiredRpm = (0.04)*timePass;
 }
 else if (timePass<3000){
         desiredRpm = 40.0;
         }
     else {
        desiredRpm = 40.0-((0.04)*(timePass-3000));
        }
  TextOut(0, LCD_LINE3, NumToStr(desiredRpm));

 // Implementación PID
  error = desiredRpm - motorRpm;
  deltaError = error - prevError;
  integralOfError = prevError + error;

// señal de control
  motorCommand = Kp*error + Ki*integralOfError;
  if(motorCommand >= 100.0) motorCommand = 100.0;
  if(motorCommand <= 0.0) motorCommand = 0.0;
  OnFwd(MOTOR, motorCommand);

curTick = CurrentTick();
deltaT = curTick - prevTick;
deltaTWrite=deltaTWrite+deltaT;

// Direferencia del angulo del motor
curAngleInDegrees = MotorRotationCount(MOTOR);
deltaAngleInDegrees = curAngleInDegrees - prevAngleInDegrees;
strDeltaAngleInDegrees = NumToStr(deltaAngleInDegrees);

motorRpm = deltaAngleInDegrees * DEG2RPM / deltaT;
strMotorRpm = NumToStr(motorRpm);
motorRadPerSec = motorRpm * RPM2RADPERSEC;
strMotorRadPerSec = NumToStr(motorRadPerSec);
elapsedTimeInSeconds = elapsedTimeInSeconds + (deltaT/1000.0);
strElapsedTimeInSeconds = NumToStr(elapsedTimeInSeconds);
// Mostrar la velocidad y el tiempo en la pantalla
TextOut(0, LCD_LINE4, NumToStr(motorRpm));
TextOut(0, LCD_LINE6, NumToStr(elapsedTimeInSeconds));

//Cada 60 milisegundos se recoge los datos de tiempo y rpm y se guardan en el archivo
if (deltaTWrite>60.0){
  curAngleInDegrees1 = MotorRotationCount(MOTOR); // Obtener la posicion relativa
  deltaAngleInDegrees1 = curAngleInDegrees1 - prevAngleInDegrees1;
```

```
  strDeltaAngleInDegrees1 = NumToStr(deltaAngleInDegrees1);
  motorRpm1 = deltaAngleInDegrees1 * DEG2RPM / deltaTWrite;
  strMotorRpm1 = NumToStr(motorRpm1);
  // Escribir el dato al fichero. El texto será finalizado con un fin de linea - EOL
  text=StrCat(strElapsedTimeInSeconds, "," , strMotorRpm1,",");
  //Escribir en el fichero
  WriteToFile(text);
  deltaTWrite=0.0;
  prevAngleInDegrees1 = curAngleInDegrees1;
 }

 // // inicializar para el siguiente ciclo
 prevTick = curTick;
 prevAngleInDegrees = curAngleInDegrees;
 prevError = integralOfError;
 orangeButtonPushed = ButtonPressed(BTNCENTER, FALSE);
 } while( !orangeButtonPushed && (FreeMemory()>=2000) );


 // terminar el ciclo
 ClearScreen();
 TextOut(0, LCD_LINE2, "Quitting", false);
 // Parar el motor
 OnFwd(MOTOR, 0);
 StopWriteToFile();
 PlaySound(SOUND_LOW_BEEP);
 Wait(SEC_2);
} // fin del programa
```

Programa cálculo de distancias de los sensores ultrasonidos

```
#define SENSOR1 IN_4
#define SENSOR2 IN_1

int dist1,dist2;
taskmain()
{
SetSensorLowspeed(SENSOR1);
SetSensorLowspeed(SENSOR2);
while(true)
{
dist1 = SensorUS(SENSOR1);// leer el valor del ultrasonidos
dist2= SensorUS(SENSOR2);//leer el valor del segundo sensor

TextOut(0, LCD_LINE3, StrCat("DIST = ", NumToStr(dist1)));
TextOut(0, LCD_LINE4, StrCat("DIST = ", NumToStr(dist2)));
}
}
```

Parada de emergencia.

El diagrama de flujo es el siguiente:

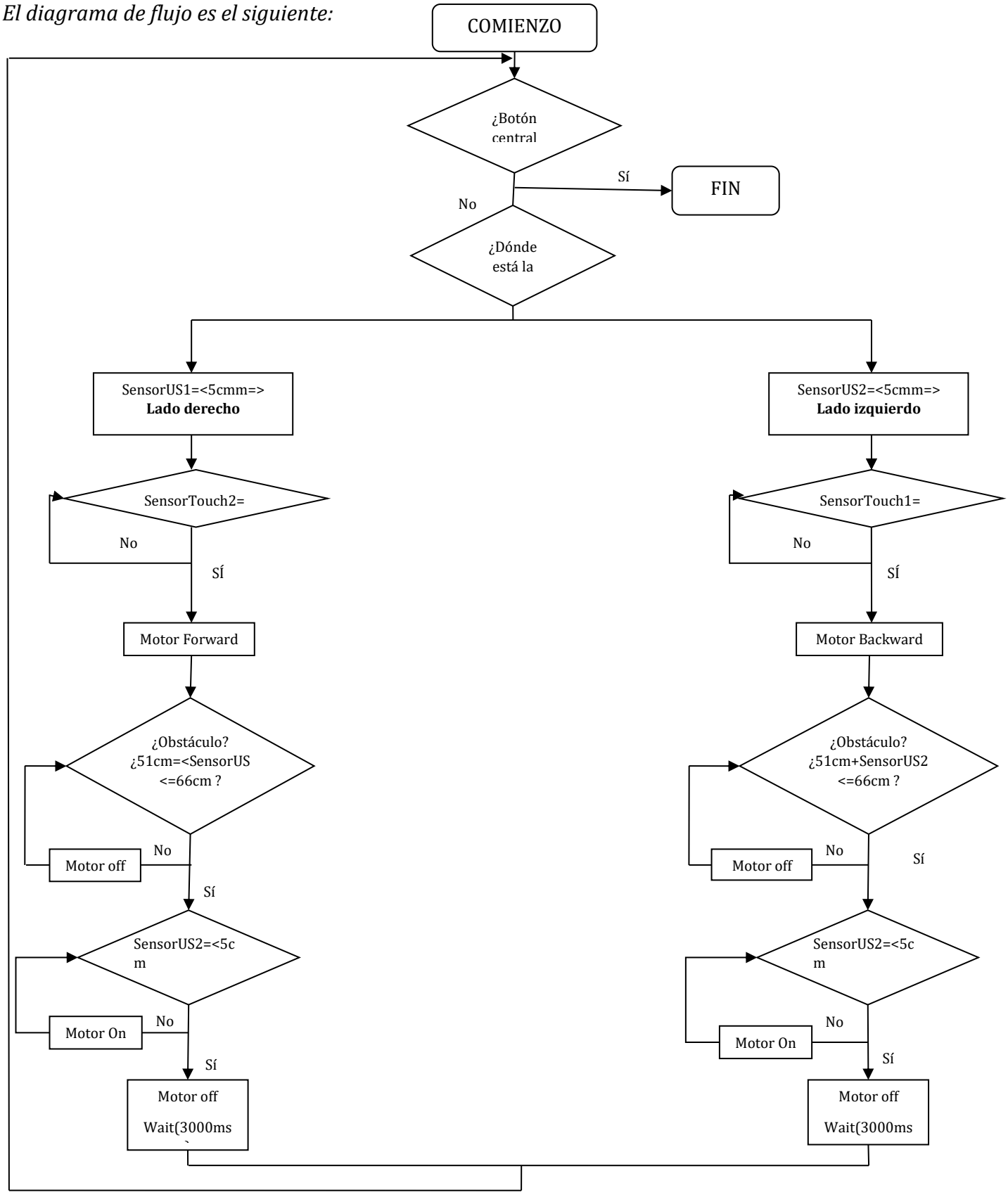

Figura 35: Diagrama de flujo del programa principal.

Código NXC Parada de emergencia

```
#define MOTOR OUT_A// definir motor
#define SenUS1 IN_1 // definir sensores de ultrasonidos
#define SenUS2 IN_2
#define TouchSen1 IN_3// definir sensores de contacto
#define TouchSen2 IN_4
#define LongituBarq// definir longitud de barquilla
#define Separacion 66 // definir separación entre torres
#define Velocidad 25 // porcentaje de la potencia del motor previamente calculado
#define Cerca  5
taskMov_Der()
 {
while(SensorUS(SenUS2)<=Cerca)
  {
OnFwd(MOTOR,Velocidad);
while(SensorUS(SenUS1)+SensorUS(SenUS2)<Separacion-LongituBarq-2)
   {
Float(MOTOR);
TextOut(10, LCD_LINE1, "OBSTÁCULO");
PlayTone(440,100);
   }
  }
Float(MOTOR);//parar motor al llegar al destino
Wait(3000);//esperar hasta el siguiente viaje
}
taskMov_Izq()
{
while(SensorUS(SenUS1)<=Cerca) // hasta llegar al destino seguir en marcha
  {
```

```
OnFwd(MOTOR,Velocidad);
while(SensorUS(SenUS1)+SensorUS(SenUS2)<Separacion-LongituBarq-2)  // detección de obstaculo
   {
Float(MOTOR);
TextOut(10, LCD_LINE1, "OBSTÁCULO");
PlayTone(440,100);
   }
 }
Float(MOTOR);//parar motor al llegar al destino
Wait(3000);//esperar hasta el siguiente viaje
}
taskmain()
{
SetSensorLowspeed(SenUS1);//definir que tipo de sensor en cada conexión
SetSensorLowspeed(SenUS2);
SetSensorTouch(TouchSen1);
SetSensorTouch(TouchSen2);
while(ButtonPressed(BTNCENTER, false))
 {
if (SensorUS(SenUS1)<=Cerca & TouchSen2==1)  // llamada desde la torre de la izquierda
   {
StartTask(Mov_Der)
   }
if (SensorUS(SenUS2)<=Cerca & TouchSen1==1) //llamada desde la  torre de la derecha
   {
StartTask(Mov_Izq)
   }
 }
 }
}
```

ANEXO II: Listado de elementos

Elemento	Unidades
Barra Tetrix de aluminio (416mm)	2
Barra Tetrix de aluminio (160 mm)	6
Barras Tetrix de aluminio (100mm)	4
Tuercas y tornillos kit Tetrix	30
LEGO Technic Light Gray 1x15 agujeros Mindstorms Parte 32277	56
LEGO Technic Light Gray 1x7 agujeros Mindstorms Parte 32275	16
LEGO Technic Light Gray 1x5 agujeros Mindstorms Parte 32273	64
LEGO Technic Light Gray 1x3 agujeros Mindstorms Parte 32271	24
LEGO Technic LIGHT GREY CONNECTOR 3x3 BENT 4	48
LEGO Technic CHASIS MARCO LIFTARM BeamStudless 5x7	4
Neumático 24mmX7mm, Rueda y Ejes Largos	7
Cordón de algodón	25 m
Ladrillo (Microprocesador) NXT	1
LEGO MINDSTORMS Interactive Motor (9842)	1
LEGO Mindstorms NXT Sensor de Ultrasonidos (9846)	2
LEGO MINDSTORMS Sensor de contacto (9843)	2
Cables de conexión	5