UNIVERSIDAD POLITÉCNICA DE MADRID

ESCUELA TÉCNICA SUPERIOR
DE INGENIEROS DE TELECOMUNICACIÓN

MÁSTER EN INGENIERÍA BIOMÉDICA

TRABAJO FIN DE MÁSTER

DISEÑO E IMPLEMENTACIÓN DE LA ADQUISICIÓN DE POSICIÓN DE UN SISTEMA HÁPTICO PARA APLICACIÓN EN CIRUGÍA Y SU PRESENTACIÓN EN UN ENTORNO VIRTUAL

EVA GARCÍA ALONSO

2017

UNIVERSIDAD POLITÉCNICA DE MADRID

ESCUELA TÉCNICA SUPERIOR
DE INGENIEROS DE TELECOMUNICACIÓN

Reunido el tribunal examinador en el día de la fecha, constituido por

Presidente: Dr. D. Julio Muñoz García

Vocal: Dr. D. Jose Luis Muñoz Sanz

Secretario: Dr. D. Bryan Strange

Suplente:

para juzgar el Trabajo Fin de Titulacin titulado:

DISEÑO E IMPLEMENTACIÓN DE LA ADQUISICIÓN DE POSICIÓN DE UN SISTEMA HÁPTICO PARA APLICACIÓN EN CIRUGÍA Y SU PRESENTACIÓN EN UN ENTORNO VIRTUAL

del alumno D. Eva García Alonso
dirigido por Dr. D. Álvaro Gutiérrez Martín
del Departamento de Tecnología Fotónica y Bioingeniería

Acuerdan otorgar la calificación de: _____

Y, para que conste, se extiende firmada por los componentes del tribunal, la presente diligencia

Madrid, 20 de julio de 2017

El Presidente | El Vocal | El Secretario

Fdo: _____ Fdo: _____ Fdo: _____

UNIVERSIDAD POLITÉCNICA DE MADRID

ESCUELA TÉCNICA SUPERIOR
DE INGENIEROS DE TELECOMUNICACIÓN

MÁSTER EN INGENIERÍA BIOMÉDICA

TRABAJO FIN DE MÁSTER

DISEÑO E IMPLEMENTACIÓN DE LA ADQUISICIÓN DE POSICIÓN DE UN SISTEMA HÁPTICO PARA APLICACIÓN EN CIRUGÍA Y SU PRESENTACIÓN EN UN ENTORNO VIRTUAL

EVA GARCÍA ALONSO

2017

Resumen

Este Trabajo de Fin de Máster (TFM) se centra en el desarrollo de un sistema de control para una posterior integración conjunta con un sistema mecánico.

El principal objetivo es la implementación de un sistema de visualización a tiempo real de datos provenientes de un dispositivo de seis grados de libertad. El desarrollo de la estructura empleada para dicha simulación queda fuera de los objetivos del presente Trabajo Fin de Máster.

Las tres señales de los encoders son procesadas en una tarjeta de desarrollo Arduino Due y las posición relativa del dispositivo será enviada con un formato concreto al entorno del programa de visualización Chai3D.

Por otro lado, será en este entorno donde se lleve a cabo el desarrollo de la visualización en tres dimensiones, así como la decodificación de los datos provenientes del Arduino Due y la lectura de la información de los movimientos traslacionales.

Todo el trabajo se ha llevado a cabo en el Laboratorio de Robótica y Control de la ETSIT.

Abstract

This Master Thesis focuses on the development of a control system designed to be integrated with a mechanical system.

The main objective of the thesis is to implement a real-time visualization system of the data taken from a device of six degrees of freedom. The development of the structure used for this simulation stays out of the boundaries of this thesis.

The three signals from the encoders will be processed through an Arduino Due board, and are sent to the environment of the simulation framework, Chai3D.

Moreover, the visualization and the decoding of the data taken from the Arduino Due, as well as the readout of the translational movements are carried out on the same environment.

Agradecimientos

A mi tutor Álvaro, por la paciencia, apoyo y ayuda que me ha prestado durante el desarrollo de este TFM.

A todos los compañeros de Robolabo, en especial a Alberto por la motivación y apoyo constante durante este año.

A mis padres y hermanas, por enseñarme a luchar por lo que quiero y no dejar que me rinda nunca.

Índice general

Índice de figuras

Índice de tablas

Lista de Acrónimos

- **DOF**: Degrees of freedom (Grados de libertad)
- **CMI**: Cirugía de mínima invasión
- **TFM**: Trabajo Fin de Máster

Capítulo 1

Introducción y Objetivos

1.1. Introducción

La cirugía es, según la Organización Mundial de la Salud (OMS, 2017), todo procedimiento que se lleva a cabo en un quirófano y que impique una incisión o manipulación de un tejido y que suele necesitar anestesia local o general. Es un técnica que ha ido evolucionando con la aparición de nuevas tecnologías que se han ido incorporando a las rutinas diarias de los quirófanos.

Sin embargo, ha existido una gran revolución con la aparición de la cirugía laparoscópica y la cirugía robótica (Dávila and Tsin, 2006). La cirugía laparoscópica, también conocida como Cirugía de Mínima Invasión (CMI), es una técnica quirúrgica moderna con la cual la cirugía se lleva a cabo a través de una pequeña incisión (entre 0.5-1.5 cm) denominada comúnmente puerto. Son numerosas las ventajas que aporta esta técnica en comparación con la cirugía tradicional. El dolor y las hemorragias son notablemente reducidas así como el tiempo de recuperación de los pacientes y la disminución de las infecciones postquirúrgicas. El elemento clave de este tipo de cirugía es el laparoscopio, una fibra larga que permite la visualización a tiempo real del área afectada a través de un monitor (Soper et al., 2008).

Por otro lado, la cirugía robótica nace a partir de la cirugía laparoscópica en la cual se aplican los conceptos de ingeniería de control y robótica para incorporar dispositivos que faciliten el trabajo del cirujano en el quirófano. (Castillo and Sánchez-Salas, 2007)

Sin embargo, existen ciertas desventajas con respecto a la cirugía tradicional. Por un lado, existe un limitado rango de movimiento del cirujano así como una disminución en su percepción sensorial. Esto puede provocar que los médicos no sean capaces de medir las fuerzas que se están aplicando pudiendo provocar complicaciones en tareas delicadas como puede ser la sutura (Westebring-Van Der Putten et al., 2008). Por otro lado, es una técnica que juega un papel muy importante durante la residencia de los cirujanos ya que, cada vez más, es empleada a diario en los quirófanos a lo largo del mundo (cirugía general, cirugía oncológica, cirugía pediátrica, etc). Todo esto hace que requiera un entrenamiento previo por diferentes dispositivos que simulan a los equipos de cirugía laparoscópica (Hamad and Curet, 2010)

1.2. Estado del arte

Como ya se adelantaba en el apartado anterior, para llevar a cabo una CMI o cirugía robótica es necesario una gran formación y entrenamiento previo de los cirujanos debido principalmente a la disminución de la percepción tanto visual como sensitiva.

Inicialmente, el procedimiento habitual era el aprendizaje con un médico adjunto en el propio quirófano. Sin embargo, éste era un proceso de aprendizaje lento debido a la baja interacción del alumno en la propia cirugía.

Poco a poco, a lo largo de las dos últimas décadas han ido apareciendo diversos simuladores de realidad virtual a través de los cuales se complementaba la formación de cirujanos con la metodología tradicional. Estos permiten a los cirujanos conocer las técnicas antes de aplicarlas en vivo, lo que les aporta seguridad y eficacia (Rodríguez-García et al., 2006)

A continuación, se mencionarán algunos de los más destacados:

- **eoSIM SurgTract**: Es un dispositivo diseñado principalmente para la formación de estudiantes. Dispone de 4 tipos de intrumentos laparoscópicos intercambiables, una luz interna para mejorar la iluminación así como una imagen a tiempo real con calidad HD (Ver Figura 1.1). Además, es fácilmente portable, lo que lo hace más cómodo de cara al uso por estudiantes. Sin embargo, aunque es lo suficientemente completo para formar estudiantes, se puede quedar corto para completar la formación de cirujanos con algún conocimiento de cirugía de mínima invasión. Para esto último, serán de mayor utilidad los ejemplos siguientes. (eoSurgical, 2017)

Figura 1.1: eoSIM SurgTract Advanced
(eoSurgical, 2017)

- **Lap Trainer**: Desarrollado por Simulab, es un entrenador laparoscópico diseñado para la formación de estudiantes y de cirujanos con una baja formación en tecnologías usadas en Cirugía de Mínima Invasión. Dispone de diferentes posiciones de la cámara, es portable y tiene una buena realimentación háptica. Se muestra en la Figura 1.2. (Simulab, 2017)

Figura 1.2: Lap Trainer
(Simulab, 2017)

- **LapVR Surgical Simulator**: Creado por la empresa CAE Heatlhcare, está diseñado para dar formación completa a estudiantes aunque también para completar la formación de cirujanos. El dispositivo, que se puede observar en la Figura 1.3, reproduce procedimientos laparoscópicos con una gran precisión debido a su tecnología háptica usando modelos fisiológicos reales. Incluye diferentes módulos que permiten simular desde un corte de un tejido hasta una apendectomía por laparoscopía gracias a sus 6 grados de libertad en cada uno de sus dos instrumentos. (Healthcare, 2017)

Figura 1.3: LapVR Surgical Simulator
(Healthcare, 2017)

- **Pyxus HD Move**: Creado por Inovus Medical. Diseñado para instruir a estudiantes y que alcancen las habilidades precisas para llevar a cabo una operación por laparoscopía. El dispositivo, mostrado en la Figura 1.4 permite disección roma y aguda, así como sutura y la formación de nudos. (Inovus, 2017)

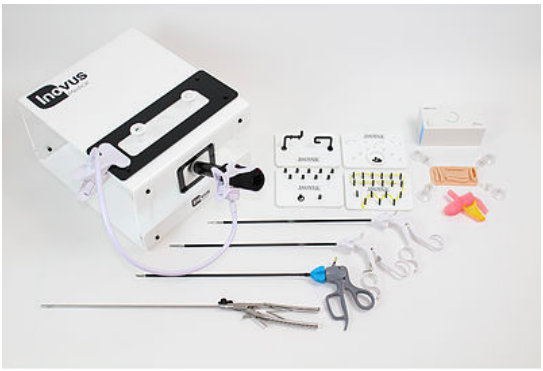

Figura 1.4: Pyxus HD Move
(Inovus, 2017)

1.3. Motivación

La motivación del presente Trabajo Fin de Máster surge para dar solución a una necesidad latente como es la formación y entrenamiento de cirujanos para el buen desarrollo de las operaciones de cirugía de mínima invasión y cirugía robótica. Teniendo en cuenta esto, se busca la creación de un sistema de monitorización y visualización 3D a tiempo real de un dispositivo de 6 grados de libertad (DOF). Para ello, se empleará: un dispositivo de 3 grados de libertad (Novint Falcon), una estructura de 3 grados de libertad creada exclusivamente para este proyecto en el Trabajo Fin de Máster de Alberto Indarte (Indarte, 2017), una tarjeta de desarrollo Arduino DUE y un ordenador con sistema operativo Linux.

1.4. Objetivos

- Estudio del hardware necesario en este trabajo para desarrollar el sistema de visualización 3D.
- Estudio de los sistemas de comunicación entre el Arduino Due y el PC y la velocidad de transmisión de datos.
- Desarrollo de un software de lectura de encoders de los motores del dispositivo.
- Desarrollo de un software de visualización 3D teniendo en cuenta las propiedades del dispositivo para acercarnos lo máximo posible a la realidad.

Capítulo 2

Entrenador para Cirugía de Mínima Invasión

2.1. Estructura

El dispositivo al que se hace referencia en este TFM está formado por 2 dispositivos de 3 grados de libertad cada uno. Uno de ellos es un dispositivo comercial Novint Falcon que se muestra en la Figura 2.1 empleado principalmente como substituto del ratón en videojuegos (Novint, 2017). El segundo de ellos es la estructura que se muestra en la Figura 2.2. Se trata de un instrumento que aporta otros tres grados de libertad que permitirá alcanzar los 6 grados de libertad deseados para este TFM.

Figura 2.1: Novint Falcon

La estructura acoplada al Novint Falcon contiene 3 motores con 3 encoders que permitirán obtener los giros deseados. En lo que concierne a este Trabajo Fin de Máster, no se hará uso de los motores sino que se emplearán las señales que envían los encoders cuando se realiza un movimiento para poder llevar a cabo la visualización 3D. Esta información será procesada en el Arduino Due (Arduino, 2017).

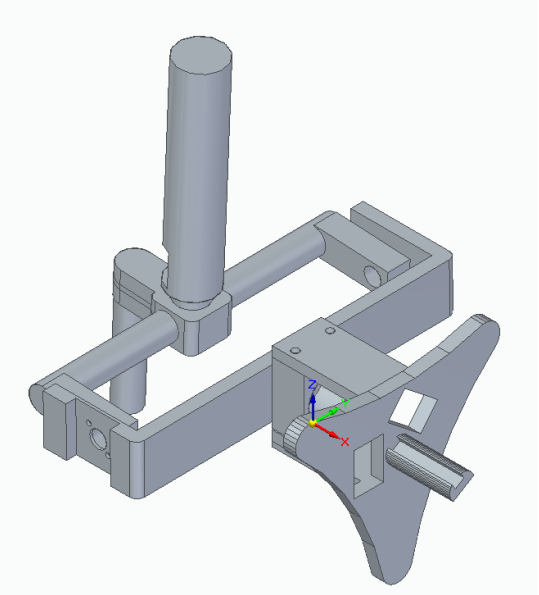

Figura 2.2: Estructura diseñada en el Trabajo Find e Máster de Alberto Indarte (Indarte, 2017)

En la Figura 2.3, se puede observar el resultado del acople de ambas estructuras.

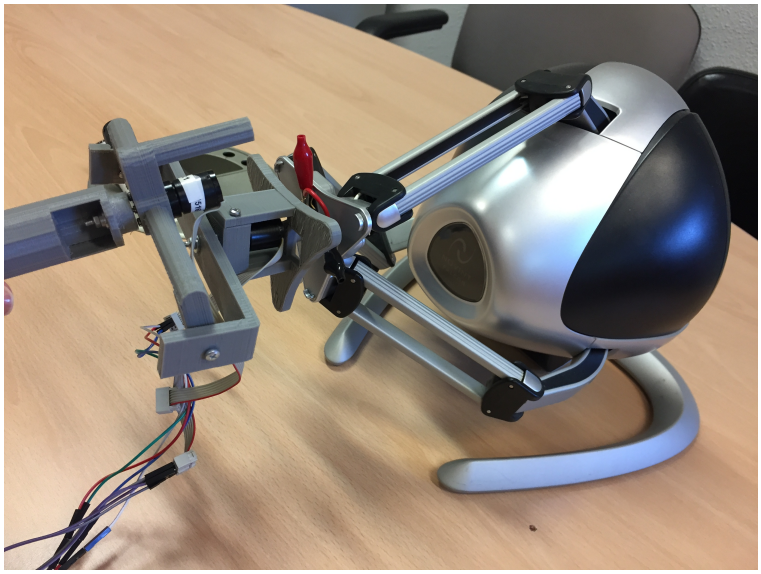

Figura 2.3: Acople Falcon y estructura

2.1.1. Grados de libertad

Como se comentó en el apartado anterior, se dispone de 6 grados de libertad. Los 3 movimientos traslacionales son proporcionados el Novint Falcon que se observa en la Figura 2.1 y los otros 3 movimientos rotacionales se obtendrán de los 3 motores de la estructura creada y que se muestran en la Figura 2.4. Por lo tanto, se pueden realizar 6 movimientos con el dispositivo.

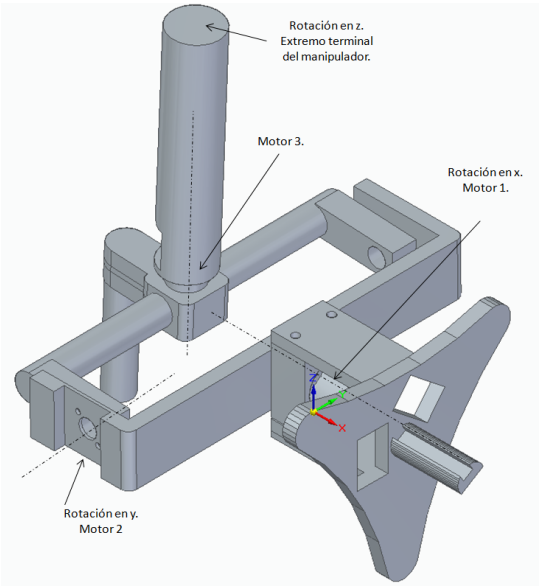

Figura 2.4: Descripción movimientos transicionales

Movimiento	Descripción del movimiento	Dispositivo
Traslación en x	Movimiento traslacional en el eje x	Novint Falcon
Traslación en y	Movimiento traslacional en el eje y	Novint Falcon
Traslación en z	Movimiento tralacional en el eje z	Novint Falcon
Rotación en x	Movimiento rotacional en el eje x	Estructura integrada al Novint Falcon
Rotación en y	Movimiento rotacional en el eje y	Estructura integrada al Novint Falcon
Rotación en z	Movimiento rotacional en el eje z	Estructura integrada al Novint Falcon

Tabla 2.1: Descripción de los movimiento permitidos

2.2. Motores y encoders

Como ya se mencionó en la Sección 2.1, se hará uso de los encoders para obtener la información de los movimientos del dispositivo. En este caso, se emplearán encoders magnéticos rotativos que transforman un movimiento angular en pulsos digitales. Estos pulsos se obtienen de dos canales con un desfase de 90º entre sí.

2.2.1. Especificaciones

Los encoders empleados en este TFM pertenecen a la serie IE2-16 de Faulhaber (Faulhaber, 2017). Están integrados en una estructura con un motor DC de la serie SR de Faulhaber. En la Tabla 2.2, se muestran algunas de las características técnicas de estos encoders.

Características	Motor 1	Motor 2	Motor 3
Vcc Recomendada	4 -18 V	4 -18 V	4 -18 V
Resolución motor	16 LPR	16 LPR	16 LPR
Reductora	41:1	76:1	76:1
Número canales	2	2	2

Tabla 2.2: Especificaciones Motor SR con Encoder iE2-16

Cada ciclo de cuadratura tiene un flanco de subida y otro de bajada por cada uno de los canales tal como se muestra en la Figura 2.5.

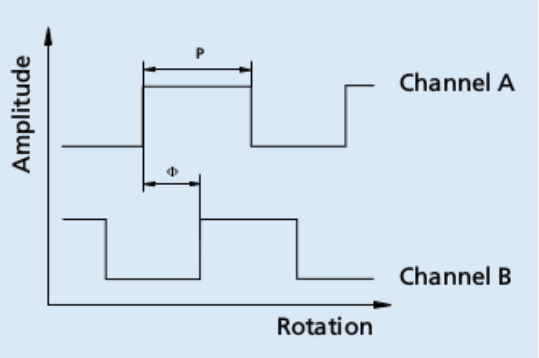

Figura 2.5: Ciclo de cuadratura

De este modo, empleando una decodificación x4, se obtendrá la posición angular final siguiendo la Ecuación 2.1. Por lo tanto, con los motores y encoders empleados se podrá calcular la siguiente posición angular mostrada en la Ecuación 2.2 (Devices, 2017).

$$\frac{Posiciones}{Revolución} = 4 \times ResoluciónEncoder \tag{2.1}$$

$$\frac{Posiciones}{Revolución} = \mathbf{64} \tag{2.2}$$

Como se puede ver en la Tabla 2.2, los motores 2 y 3 tienen un reductor 76:1, mientras que el motor 1 tienen un reductor 41:1. En las ecuaciones 2.4 y 2.5, se puede observar los pulsos teóricos que se deben obtener por revolución para cada uno de los motores incluidos en el dispositivo. En la Figura 2.6 se muestran las etiquetas empleadas para denominar a cada uno de los motores.

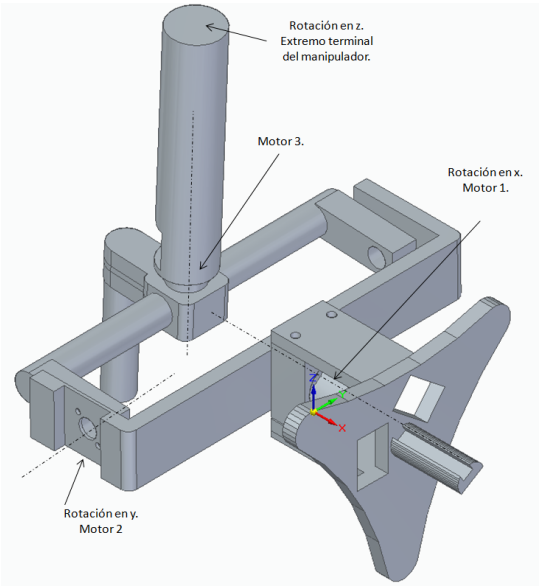

Figura 2.6: Numeración motores

$$\frac{Pulsos}{Revolución} = 64 \times Reductor \tag{2.3}$$

$$\frac{Pulsos}{Revolución} = 64 \times 41 = \mathbf{2624} \tag{2.4}$$

$$\frac{Pulsos}{Revolución} = 64 \times 76 = \mathbf{4864} \tag{2.5}$$

Capítulo 3

Sistema hardware y software

3.1. Necesidades Hardware y Software

A lo largo de la resolución de este Trabajo de Fin de Máster han surgido ciertas necesidades a las que se pretende dar solución:

- Detectar los cambios en las rotaciones del dispositivo.
- Cuantificar y procesar estos cambios para poder ajustarnos lo máximo posible a la realidad en el proceso de realizar la visualización 3D.
- Transmitir los datos al sistema de visualización 3D teniendo en cuenta los tiempos y las velocidades de la comunicación.
- Generar un sistema de visualización que se ajuste estéticamente y funcionalmente a la estructura creada.

Para dar solución a estas necesidades, se precisarán los siguientes elementos:

- **Encoders**: darán la información necesaria para obtener la posición de cada motor.
- **Arduino Due**: será necesario para recoger y procesar los pulsos que obtendremos de los encoders y también como intermediario entre los encoders y la visualización 3D.
- **Chai3D**: es un programa de simulación basada en C++ que se empleará para obtener la visualización a tiempo real de los movimientos del dispositivo.

3.1.1. Encoders

Los encoders descritos en la Sección 2.2 permiten el registro de los tres tipos de rotaciones de la estructura. Para llevar a cabo la comunicación con el Arduino Due se conectan siguiendo el esquema de la Figura 3.1 tomando como referencia la Figura 3.2 de la hoja de características (Faulhaber, 2017).

Como se puede ver en la Figura 3.1, cada uno de los encoders debe estar conectado a las siguientes líneas:

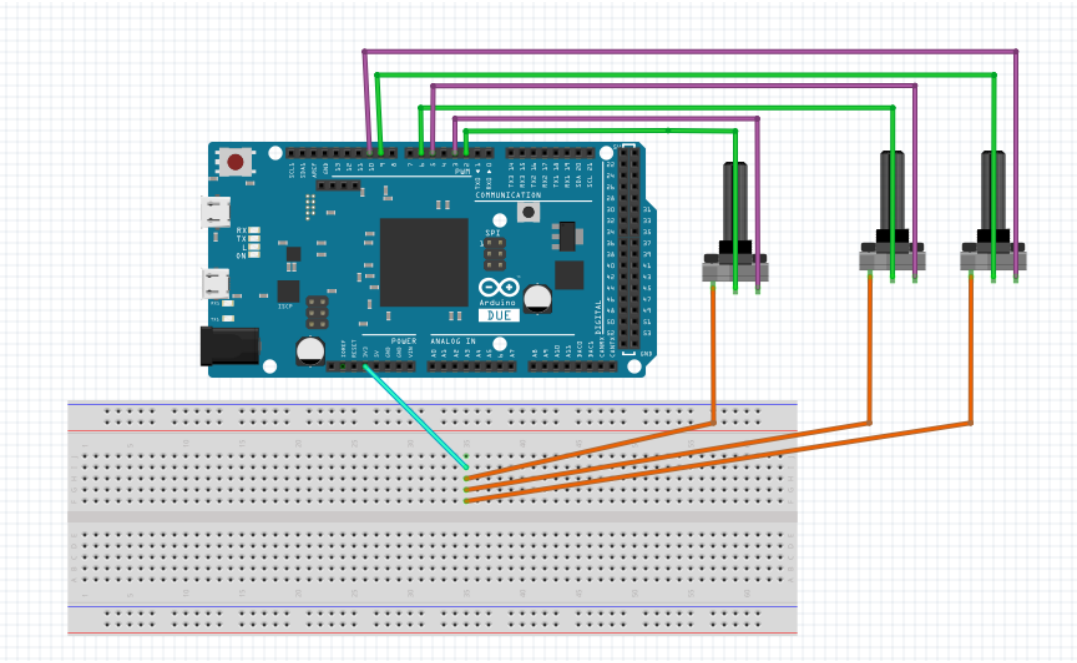

Figura 3.1: Esquema de conexión de los encoders a Arduino Due

- **GND**: Se debe establecer un mismo punto de masa, conectando el del encoder con el correspondiente del Arduino Due.

- **Vcc**: Se debe conectar el encoder a una alimentación. En el caso del Arduino Due, la alimentación es de 3.3 V.

- **Ch. A** y **Ch. B**: Ambos son conectados a un pin de interrupción del Arduino Due a través del cual se obtendrán la señales de los pulsos de los encoders.

- **Motor - y Motor +**: para este TFM no serán empleados ya que no se están activando los motores.

Teniendo en cuenta que los 3 encoders han de estar conectados a masa y a 3.3 V, se hará uso de una protoboard para conectarnos al mismo pin del Arduino Due. Por otro lado, cabe mencionar que las posiciones del Novint Falcon se obtendrán directamente desde Chai3D, no siendo necesario procesar ningún dato por el Arduino Due.

3.1.2. Arduino Due

El Arduino Due es un microcontrolador basado en la CPU Atmel SAM3X8E ARM Cortex-M3. Posee 54 pines digitales de entrada y salida y 12 analógicos de entrada. Además, permite configurar interrupciones en todos sus pines digitales, lo que será de gran utilidad para la lectura de los encoders. De este modo, no existirán problemas a la hora de leer las 6 señales que obtendremos de los encoders, una señal por cada canal y por cada encoder (Arduino, 2017).

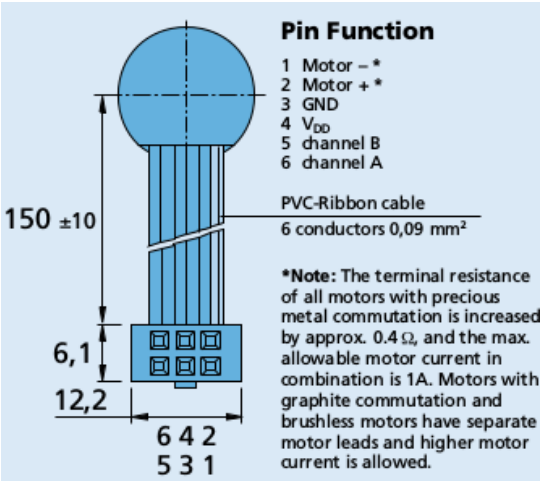

Figura 3.2: Esquema conexión encoders según su Datasheet

3.1.2.1. Implementación en Arduino Due

El Arduino Due trabajará con pulsos para tomar los valores de las posiciones de los encoders y enviarlos de esta manera al PC, donde se transformarán en grados de rotación de cada encoder. De este modo, los valores que tomen los encoders serán relativos a la posición inicial del dispositivo donde tomará valor 0.

Cada uno de los canales de cada encoder estará conectado a los pines digitales del Arduino y se configurarán como interrupciones externas que se activarán cuando haya un cambio en la posición de los encoders, tanto si es un flanco de subida como de bajada.

Para obtener el número de pulsos, se tendrá en cuenta el convenio de signos mostrado en la Figura 3.3.

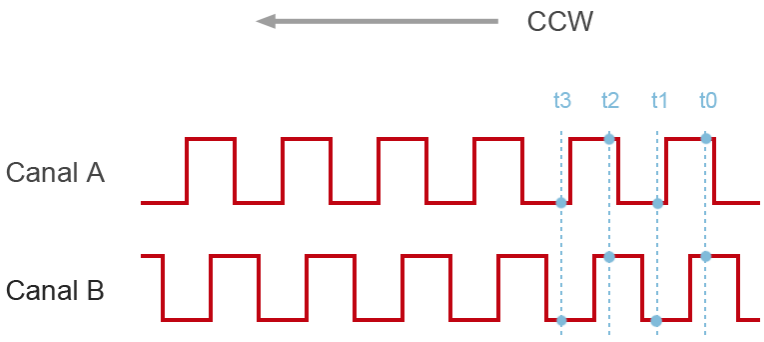

Figura 3.3: Convenio del sentido del movimiento

Teniendo en cuenta este criterio, se procesará de la siguiente manera los pulsos enviados por los encoders:

- **Interrupción en el canal A**: si el valor de las señales A y B es el mismo, se sumará una unidad al número total de pulsos. Si por lo contrario son diferentes, se restará.

- **Interrupción en el canal B**: si el valor de las señales A y B es el mismo, se disminuirá una unidad al número total de pulsos. Si por lo contrario son diferentes, se sumará.

3.1.3. Chai3D

Chai3D (Computer Haptics and Active Interface) es un software abierto basado en librerías de C++ para llevar a cabo visualización, simulación a tiempo real y retroalimentación háptica. (Chai3D, 2017)

Para este Trabajo de Fin de Máster se ha empleado este software que ha permitido la monitorización y visualización a tiempo real del dispositivo de 6 grados de libertad. Como ya se adelantaba en la Sección 2.1, nuestro dispositivo está formado por dos partes: un Novint Falcon que dará los 3 movimientos traslacionales y la estructura creada que aportará los 3 movimiento rotacionales.

En el caso del Novint Falcon, el propio Chai3D es capaz de detectarlo y darnos información acerca de su posición, por lo que podremos obtener los 3 movimientos traslacionales directamente del Chai3D sin necesidad de procesar datos previamente.

Por otro lado, para detectar las posiciones de la estructura, como ya se explicó en las Secciones 3.1.1 y 3.1.2 haremos uso de los encoders y del Arduino Due para obtener los pulsos relativos al movimiento del dispositivo. De este modo, será en Chai3D donde se realizará la transformación de pulsos a grados sexagesimales y se actualizará la posición de la estructura en tiempo real.

Inicialmente, para comprobar la simulación de los 6 grados de libertad se creó una figura lo más simple posible que consiste en una única barra donde se simulaban los 3 movimientos traslacionales y los 3 movimientos rotaciones (Ver Figura 3.4).

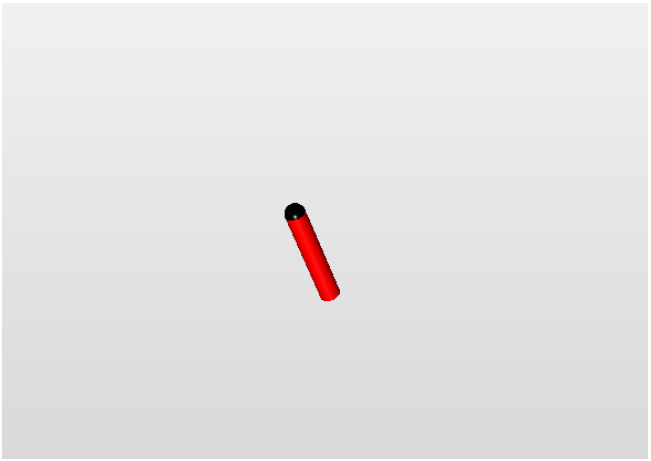

Figura 3.4: Simulación con una única barra

Como segunda alternativa, con el fin de simular los movimientos de rotación y traslacionales generados por la estructura integrada al Novint Falcon, se propuso la disposición de la figura como un sistema de barras en las que cada una de ellas simula el movimiento traslacional. Además, tomando los valores de posición del Novint Falcon directamente desde el Chai3D obtenemos los movimientos rotacionales. De este modo,

es posible realizar la simulación de 6 grados de libertad propuestos. Esta disposición se muestra en la Figura 3.5.

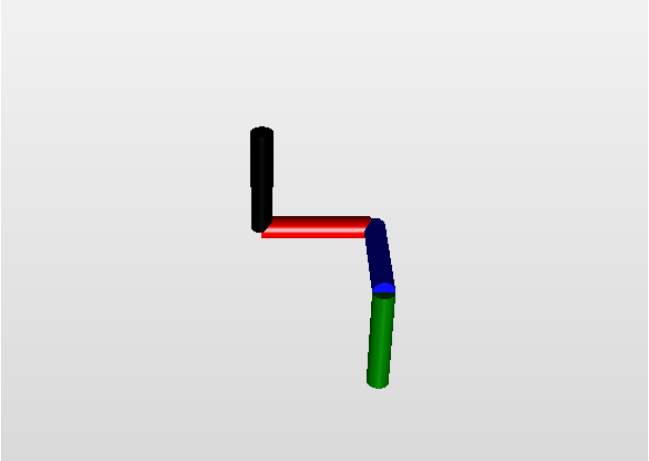

Figura 3.5: Simulación con el sistema de barras

Sin embargo, con el fin de que el sistema se acercarse lo máximo posible a la realidad, se propuso un tercer sistema en el que las piezas empleadas fuesen las propias piezas de la estructura creada en el Trabajo Fin de Máster de Alberto Indarte (Indarte, 2017).

Finalmente, en la Figura 3.6 se muestra el resultado final en el sistema de visualización. Para crearla, se han usado los archivos ".stl"procedentes de un software CAD (Computer-aided design) con el objetivo de que, como ya se ha mencionado, se lograse un mayor parecido con la figura real. Cabe mencionar que todas las piezas están representadas a escala 1:20 respecto a su tamaño real.

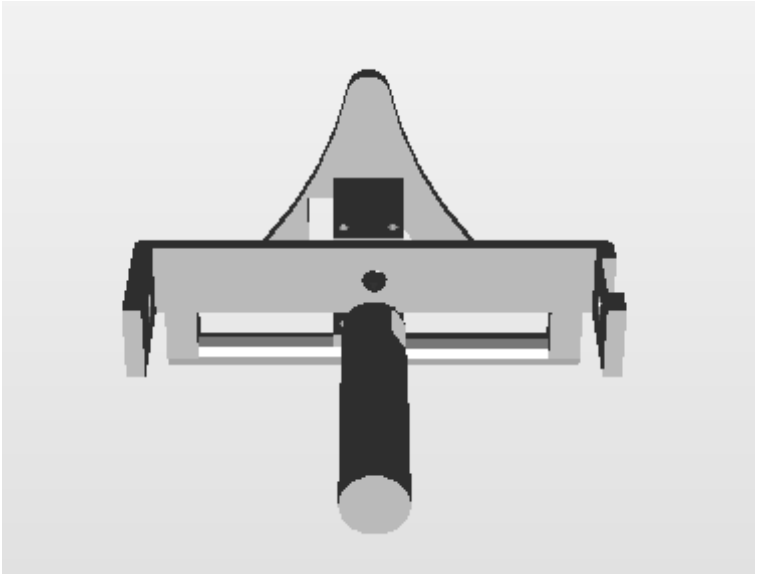

Figura 3.6: Dispositivo visto desde el programa de visualización Chai3D

3.1.3.1. Descripción de los movimientos en Chai3D

En esta sección, se explicará brevemente el desarrollo llevado a cabo para alcanzar los movimientos deseados en la visualización 3D del sistema.

Movimientos traslacionales

Antes de comenzar a explicar cada uno de los movimientos, cabe destacar que el software diferencia entre coordenadas globales y locales. Por un lado, las coordenadas globales son aquellas que hacen referencia al espacio creado en general. Por otro lado, las coordenadas locales son aquellas asociadas a cada una de las piezas incluídas en el sistema. De este modo, es posible trabajar con cada una de ellas por separado.

Los movimientos traslacionales los aporta el Novint Falcon. Sin embargo, no es necesario llevar a cabo un preprocesado de los datos obtenidos, ya que el software Chai3D está configurado para reconocer ciertos dispositivos hápticos como es el Novint Falcon.

De este modo, no se precisa conectar el dispositivo al Arduino para procesar los datos, sino que el propio Chai3D reconoce el Falcon. Una vez reconocido, es posible leer de manera sencilla la posición exacta siguiendo unas coordenadas globales y aplicarlo a nuestra visualiación 3D.

Movimientos rotacionales

Para obtener los movimientos rotacionales, como ya se ha comentado previamente a lo largo de este Trabajo Fin de Máster, es necesario obtener datos a través de microprocesador Arduino Due.

Después de reproducirse la comunicación planteada en la Capítulo 4, el software Chai3D recibe los ángulos en grados de cada uno de los movimientos rotacionales (x,y,z). Estos datos son empleados para rotar la figura de la visualización.

En este caso, es necesario establecer el mismo sistema de signos que los encoders, para que los movimientos se realicen en el mismo sentido. En las siguientes figuras es posible observar las rotaciones en los tres ejes. Además, se establece la positividad de los movimientos según los datos obtenidos de los encoders. En las Figuras 3.7, 3.8 y 3.9 se muestra una flecha que indica el sentido positivo del giro.

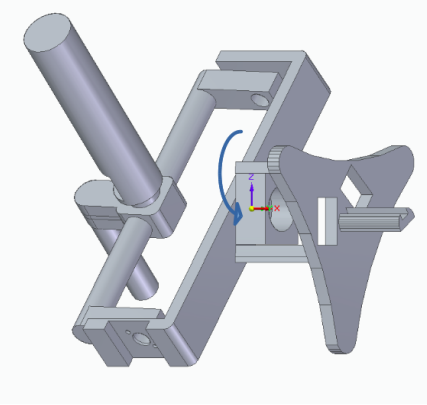

Figura 3.7: Sentido positivo de la rotación en el eje x

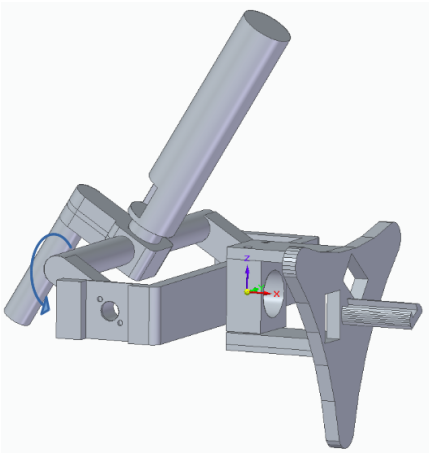

Figura 3.8: Sentido positivo de la rotación en el eje y

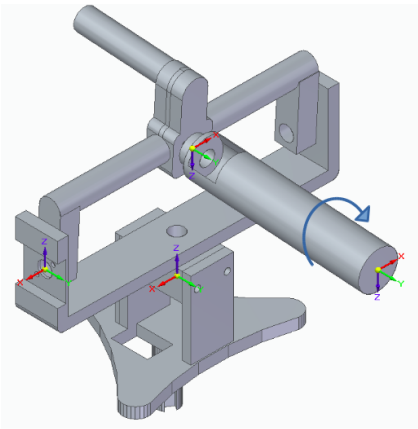

Figura 3.9: Sentido positivo de la rotación en el eje z

Una vez hecho esto, se procesa la rotación de la siguiente manera:

- Rotamos con respecto a las coordenadas locales de la pieza en cuestión.
- Trasladamos las piezas restantes con respecto a sus coordenadas locales, teniendo en cuenta el ángulo de giro de la pieza anterior.

Este procedimiento se realiza de igual manera para cada una de las rotaciones. Todos estas funciones de rotación están incluidas dentro de una función que se reproduce periódicamente de manera que se vaya actualizando la información que se obtiene del Arduino Due.

En este caso, las rotaciones se realizan en una función *void()*, es decir, una función que no tiene ningún tipo de retorno o respuesta. Además, estas rotaciones y traslaciones se realizarán mientras la simulación esté activa empleando un ciclo *while*.

De este modo, mientras estemos ejecutando el código que realiza la simulación 3D,

el programa estará actualizando continuamente la información obtenida del Arduino Due.

La escritura y lectura de estos datos será explicada en el Capítulo 4.

Capítulo 4

Sistemas de comunicación

4.1. Velocidad transmisión de datos

Un punto importante a tener en cuenta para realizar una visualización 3D a tiempo real es la velocidad de muestreo y envío de las señales de posición. Con el fin de que la velocidad sea la máxima posible para conseguir representar los movimientos de los dispositivos a tiempo real, se han hecho pruebas con diferentes velocidades de muestreo para encontrar el periodo de muestreo óptimo. De este modo, finalmente se ha tomado un periodo de T = 10 ms, es decir, una frecuencia f = 100 Hz. Aunque es una frencuencia que está por debajo de la frecuencia mínima para realizar un realimentación háptica, es una frecuencia lo suficientemente alta como para que retrasos en la visualización no sean perceptibles al ojo humano.

Por lo tanto, los datos serán enviados desde el Arduino con esta frecuencia, mientras que Chai3D, como ya se ha mencionado en la Sección 3.1.3.1, trabajará en bucle recibiendo datos a través del puerto serie, calculando las posiciones de las piezas y actualizando la información de la posición de las figuras en la visualización.

4.2. Comunicación

Como se comentaba en el apartado anterior, Arduino Due enviará los datos con una frecuencia de 100 Hz para que la visualización sea a tiempo real y el programa sea capaz de enviar y recibir datos con esa frecuencia.

Para enviar datos a través del puerto serie en Arduino es necesario:

- Conectar el Arduino Due al ordenador.
- Establecer la apertura del puerto serie mediante la siguiente línea de código: *Serial.begin()*. A través de esta línea estableceremos la velocidad de transmisión de datos que en nuestro caso es 38400 bauds.
- Escribir datos en formato ASCII en el puerto serie mediante la siguiente línea de código: *Serial.write()*. Con este comando, se escribirá el dato en formato *int* que se quiere enviar en una nueva línea cada vez que se reproduzca. Esta función envía valores entre 0 y 255 a través del puerto serie. De esta manera, para enviar valores entre 0 y 360º se establecerá una relación lineal donde 0 represente 0º

y 255 represente 360º. Esta función debe estar incluida en una función que se ejecute en bucle, para que haya una actualización periódica. (Arduino, 2017)

En el entorno del Chai3D se recibirán los datos que se envían a través de Arduino. Para ello, se debe tener en cuenta lo siguiente: los datos que se envían a través del puerto serie de arduino lo hacen en formato ASCII extendido. Debido a esto, para recibir los datos en Chai3D de una manera manejable hay que seguir los siguientes pasos:

- Configurar el puerto serie.
- Leer datos con la función *read* que los guardará en un *buffer*. Éste se configura con el tamaño de 1 byte. Además, esta función nos permite extraer un output que hace referencia al número de bytes recibidos a través del puerto serie. De esta manera, se controla que se reciban datos a través del puerto y, de este modo, se detecta si hay algún error durante la conexión entre Arduino y Chai3D. Si el output de esta función es 0, no se están recibiendo datos, mientras que si es 1, sí.
- Transformar los datos guardados en el *buffer* a un número entero que represente una medida de grados entre 0 y 360 (manejable por el programa), para poder aplicarlo y actualizar la posición de las piezas de la visualización. Esta transformación se hará teniendo en cuenta la relación lineal establecida en Arduino (0 representa 0º y 255 representa 360º).
- Realizar los cambios de posición y rotación de las piezas teniendo en cuenta los datos recibidos.

4.3. Funcionamiento del programa

Para poner en funcionamiento el programa de visualización es necesario seguir los siguientes pasos:

- Conectar el Arduino Due al ordenador.
- Cargar el programa de Arduino para que tome periódicamente las señales obtenidas del encoder y envíe dichos datos a través del puerto serie.
- Colocar las piezas en la posición mostrada en la Figura 4.1
- Ejecutar el programa de Chai3D.

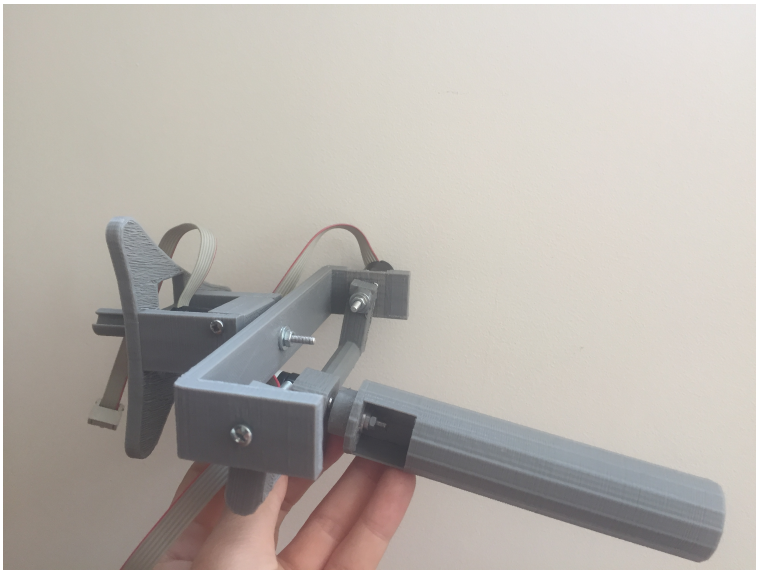

Figura 4.1: Posición inicial para comenzar la visualización

Capítulo 5

Conclusiones y líneas futuras

5.1. Conclusiones

En el presente Trabajo de Fin de Máster se ha realizado un sistema de adquisición de datos y representación en un entorno virtual cumpliendo los objetivos marcados al inicio del mismo, es decir, la visualización a tiempo real de la posición del dispositivo.

Por un lado, el programa de Arduino alcanza las expectativas recogiendo los datos necesarios para llevar a cabo la representación. Por otro lado, el programa de visualización descrito cumple con las funciones principales planteadas, es decir, la lectura de datos del entorno del Arduino y presentación tridimensional de la posición del dispositivo.

Se ha partido de la estructura mecánica descrita en el Trabajo Fin de Máster de Alberto Indarte (Indarte, 2017) para crear un sistema fácilmente reproducible empleando hardware y software libre y de bajo coste, que se ha conseguido mediante el uso de la plataforma libre Chai3D y del Arduino Due y su entorno de desarrollo.

Como reseña indicar que las principales limitaciones fueron encontradas durante la comunicación de Chai3D y Arduino, donde el envío de datos ha resultado complicado debido a la velocidad de comunicación entre ambas plataformas. Sin embargo, al no realizar una comunicación bidireccional, no ha sido un límite infranqueable, sino que se ha obtenido un sistema de visualización a tiempo real y con periodos lo suficientemente rápidos, como para que la percepción visual humana sea de una representación casi simultánea con el propio movimiento del dispositivo.

5.2. Líneas futuras

Aunque se han conseguido los objetivos planteados en este TFM, resultaría interesante seguir trabajando sobre él, no sólo mejorarlo, sino también ampliarlo.

Como líneas futuras se plantea lo siguiente:

- El principal trabajo futuro sería conseguir que el dispositivo fuera un **sistema háptico** con una comunicación bidireccional entre Arduino y Chai3D, es decir, obtener datos relativos a la fuerza generada por el sistema al chocar con algún elemento en el sistema de visualización y obtener así una percepción a través de la respuesta de los motores al choque.

- Con el fin de obtener el objetivo anterior, sería necesario **mejorar la velocidad de comunicación** entre dispositivos, ya que la frecuencia de muestreo para poder realizar una realimentación háptica debe ser mucho más alta de la frecuencia planteada (f = 100 Hz).
- Uso de **timers** en la tarjeta de desarrollo de Arduino Due que regulen el envío de datos al entorno del Chai3D para hacerlo de manera controlada.
- Introducción de otros objetos en el sistema de visualización para **simular órganos humanos** y así poder obtener información háptica cuando se produzca un choque.
- Mejorar el sistema de visualización, de manera que los movimientos sean lo más similares posibles a los **movimiento reales**.

Bibliografía

Arduino (2017). Arduino due. https://store.arduino.cc/arduino-due. [Online; accedido junio 2017].

Castillo, O. A. and Sánchez-Salas, R. (2007). Bases laparoscopicas de la cirugia robotica. *Archivos Españoles de Urología (Ed. impresa)*, 60(4):357–362.

Chai3D (2017). Chai 3d. http://www.chai3d.org/. [Online; accedido junio 2017].

Dávila, Á. and Tsin, D. A. (2006). Cirugía por orificios naturales (notes y manos)¿ la tercera revolución quirúrgica. *Rev Mex Cir Endoscop*, 7(1-4):6–13.

Devices, G. (2017). Quadrature. https://granitedevices.com/wiki/Quadrature. [Online; accedido junio 2017].

eoSurgical (2017). eosurgical. https://www.eosurgical.com/.

Faulhaber (2017). Incremental encoders - two channels. https://fmcc.faulhaber.com/details/overview/PGR_5301_13831/PGR_13831_13812/en/GLOBAL/. [Online; accedido junio 2017].

Hamad, G. G. and Curet, M. (2010). Minimally invasive surgery.

Healthcare, C. (2017). Cae lapvr. https://caehealthcare.com/surgical-simulation/lapvr. [Online; accedido junio 2017].

Indarte, A. (2017). Diseño de las componentes mecánicas, estructurales y motrices de un sistema de control háptico para aplicación en cirugía. Master's thesis, Universidad Politécnica de Madrid.

Inovus (2017). Pyxus hd move. http://www.inovus.org/home-spanish. [Online; accedido junio 2017].

Novint (2017). Novint falcon. http://www.novint.com/index.php/products/novintfalcon. [Online; accedido junio 2017].

OMS (2017). Organización mundial de la salud. http://www.who.int/es/.

Rodríguez-García, J. I., Turienzo-Santos, E., Vigal-Brey, G., and Brea-Pastor, A. (2006). Formación quirúrgica con simuladores en centros de entrenamiento. *Cirugía Española*, 79(6):342–348.

Simulab (2017). Lap trainer. https://www.simulab.com/products/laptrainer%E2%84%A2-simuvision%C2%AE. [Online; accedido junio 2017].

Soper, N. J., Swanström, L. L., and Eubanks, S. (2008). *Mastery of endoscopic and laparoscopic surgery*. Lippincott Williams & Wilkins.

Westebring-Van Der Putten, E., Goossens, R., Jakimowicz, J., and Dankelman, J. (2008). Haptics in minimally invasive surgery–a review. *Minimally Invasive Therapy & Allied Technologies*, 17(1):3–16.