ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA
GRADO EN INGENIERÍA DE LA SALUD

DISEÑO E IMPLANTACIÓN DE UN SISTEMA DE CONTROL DE MOVIMIENTOS DE UNA CÁMARA PARA CIRUGÍA MÍNIMAMENTE INVASIVA DE PUERTO ÚNICO

CONTROL MOTION DESIGN AND ITS IMPLANTATION FOR A SINGLE SITE MINIMALLY INVASIVE SURGERY CAMERA

Realizado por
Javier Esteban Muñoz
Tutorizado por
Isabel García Morales
Irene Rivas Blanco
Departamento
Ingeniería de Sistemas y Automática

UNIVERSIDAD DE MÁLAGA
MÁLAGA, JULIO DE 2015

Resumen: En el presente trabajo se describe el diseño e implantación de un sistema de control de movimientos de una cámara intra-abdominal para cirugía mínimamente invasiva de puerto único. La cámara se encuentra integrada en un mini-robot, el cual se introduce en la cavidad abdominal a través de una incisión realizada por los cirujanos. Para llevar a cabo el mencionado control de movimientos será necesario la integración de diversos elementos hardware: microcontroladores de la marca Arduino y la shield EasyVR para el reconocimiento del habla, entre otros, que junto a sus correspondientes software de control y servomotores permitirán obtener el fin deseado. Los programas generados incluyen el control de los motores y dos interfaces distintas, dependiendo de si se quiere utilizar comandos de voz o el teclado de un ordenador. Además, el sistema fruto de este trabajo se debe integrar en la plataforma robótica que dispone de una arquitectura basada en entorno ROS en la que trabaja el grupo de Robótica Médica del departamento de Ingeniería de Sistemas y Automática, para así poder comunicarse con el resto de sistemas con los que ya cuentan. Tanto el hardware necesario como el software desarrollado se describen a lo largo de esta memoria, explicando y justificando cada decisión tomada, para finalmente terminar con una serie de experimentos que validan el funcionamiento del mini-robot.

Palabras clave: control, mini-robot, mini cámara, CMI, SILS

Abstract: In this work a control motion design and implantation of an intra-abdominal single site for minimally invasive surgery camera is described. The camera is integrated in a mini-robot which is introduced into the abdominal cavity through an incision made by the surgeons. In order to achieve this control, it is necessary the integration of several hardware devices, mainly Arduino microcontrollers and an EasyVR shield for a speech recognition activity. These hardware devices working together with their control software and servomotors will get the expected goal. The generated programs include the servomotors controller and two different interfaces, depending on the use of human voice or keyboard commands. Besides, the system created in this work must be integrated in a robotic platform developed by the Medical Robotic group, which has ROS architecture to connect all its systems. The Medical Robotic group belongs to the Systems and Automation Engineering Department in the University of Malaga. Both software and hardware architecture developed in this work, are described along this memory, trying to explain and justifying every decision that has been taken, in order to finish with a set of experiments which verify that the mini-robot designed works successfully.

Keywords: control, mini-robot, mini camera, MIS, SILS

Índice General

Capítulo 1: Introducción

1.1 Robótica aplicada a la cirugía mínimamente invasiva de puerto único

Desde hace siglos, la medicina se ha servido de la tecnología disponible en la época para llevar a cabo sus fines, pero es hoy en día cuando dicha relación está creciendo a gran velocidad. Por lo tanto es necesario un esfuerzo doble: por una parte los ingenieros han de tener ciertos conocimientos de medicina para poder crear dispositivos cada vez más especializados, y por otra los médicos han de adaptarse a dichos dispositivos y aprender a utilizarlos.

Los procedimientos médicos pueden clasificarse, entre otras cosas, según sean invasivos o no invasivos, pero a su vez dentro de estos dos grandes grupos puede haber distinciones. La cirugía laparoscópica es un tipo de cirugía mínimamente invasiva (CMI) en la que tanto los instrumentos, llamados de caña larga, como el endoscopio, se introducen en la cavidad abdominal a través de pequeñas incisiones, denominadas puertos. Dicho procedimiento se puede ver en la figura 1. Cabe destacar que en este tipo de procedimientos no hay visión directa del campo quirúrgico, sino que se obtiene de forma indirecta a través de imágenes de video mediante un endoscopio. Además, se pierde la sensación de tacto al no tener contacto directo con las estructuras anatómicas y se necesita una fuerte coordinación entre el cirujano y el asistente, el cual mueve y orienta el endoscopio. Aun así, este procedimiento presenta numerosas ventajas frente a la cirugía abierta tradicional (laparotomía), entre las que se puede destacar la reducción de las complicaciones post-operatorias, reducción del tiempo de recuperación del paciente y una mejora estética, ya que las cicatrices son menores [1].

Figura 1: Cirugía laparoscópica

En los últimos años han surgido nuevas técnicas con el fin de potenciar dichos beneficios, entre ellas la cirugía de puerto único o SILS (de su acrónimo en inglés Single Incision Laparoscopic Surgery) [2]. Dicha técnica se basa en la introducción tanto de los instrumentos como del endoscopio por una única incisión, a través de un trocar multipuerto. A pesar de las ventajas que presenta para los pacientes, esta técnica presenta dos grandes limitaciones para los cirujanos. Por un lado, al introducir todas las herramientas por un mismo puerto, se limita el rango de movimiento de los instrumentos, y por otro, la cercanía de las herramientas y el endoscopio produce una pérdida de triangulación, lo que se traduce en una pérdida de sensación de profundidad en la imagen que recibe el cirujano.

Una de las soluciones a las que se ha recurrido para resolver estos problemas es utilizar instrumentos flexibles o curvos especiales con el fin de abarcar un mayor espacio de trabajo. Sin embargo, esto no solventa el problema de la pérdida de triangulación y requiere un gran entrenamiento por parte del cirujano [3]. Otra alternativa es el uso de robots miniaturizados, los cuales se pueden introducir en la cavidad abdominal a través de la incisión realizada para introducir las herramientas quirúrgicas, y que se fijan a la pared abdominal mediante algún procedimiento. De esta manera se reduce el número de elementos que comparten el puerto de entrada y se evita tanto el problema de la pérdida de triangulación mencionado anteriormente, en caso de que el mini-robot incorpore una cámara, como la pérdida de grados de libertad debido al efecto de fulcro. El posicionamiento y la fijación de los robots a la pared abdominal se puede realizar a través de varios métodos, como sutura [4], fijación con agujas [5]-[6], o interacción magnética [7]-[8]. La solución que más se puede encontrar en la literatura

científica es mediante interacción magnética, ya que ésta permite el desplazamiento del mini-robot a lo largo de la pared abdominal, lo cual permite cambiar el punto de vista de la imagen laparoscópica durante el procedimiento quirúrgico.

1.2 Motivación del trabajo de fin de grado.

La línea de investigación de Robótica Médica del Departamento de Ingeniería de Sistemas y Automática de la Universidad de Málaga lleva muchos años trabajando en el campo anteriormente mencionado, dando finalmente a luz al proyecto MARCUS [9] y aportando ideas innovadoras.

En este sentido, abordaron el diseño y desarrollo de una plataforma robótica capaz de situar en el interior de la cavidad abdominal un mini-robot dotado de una cámara que trabaja de forma colaborativa con el cirujano. El sistema se puede programar fuera de línea para establecer el plan maestro durante la intervención y de forma intraoperatoria mediante un interfaz persona-máquina multi-modal. Todos estos sistemas están integrados en una arquitectura basada en ROS (Robot Operating System) [11].

De esta forma, I. Rivas-Blanco y otros, [2] y [10], proponen la introducción de un mini-robot en la cavidad abdominal, el cual incorpora una cámara y se fija a la pared abdominal mediante interacción magnética. A diferencia de otros trabajos anteriores, [12]-[13]-[14], en los que la sujeción magnética externa se desplaza manualmente a lo largo de la pared abdominal, estos trabajos proponen que la sujeción externa esté acoplada a un brazo robótico, como se muestra en la figura 2, y que sea éste el que desplace el mini-robot a través de una interfaz persona-máquina. De esta manera, el movimiento de la cámara se puede realizar de forma automática mediante un interfaz humano-máquina, o bien de forma autónoma dotando al sistema de ciertas capacidades cognitivas [15].

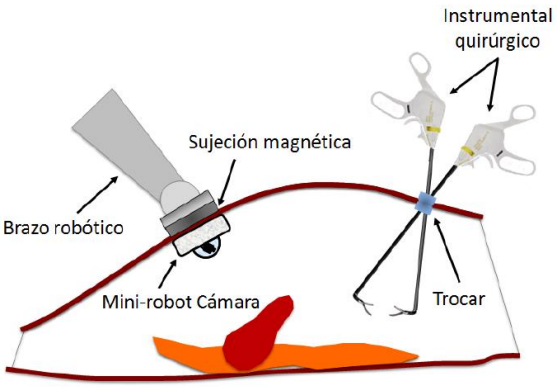

Figura 2: Disposición de la cámara y los instrumentos en la cavidad abdominal

Inicialmente el mini-robot se diseñó de manera que fuese totalmente inalámbrico. El dispositivo cuenta con dos grados de libertad: rotación e inclinación. El primero se realiza gracias a la rotación de la última articulación del manipulador externo; dicha rotación se traduce en una cambio en la orientación del elemento de sujeción o holder magnético y, por lo tanto, del mini-robot. El movimiento de inclinación se realiza mediante interacción magnética, tal y como se describe en [9]. Ambos grados de libertad se pueden ver en la figura 3.

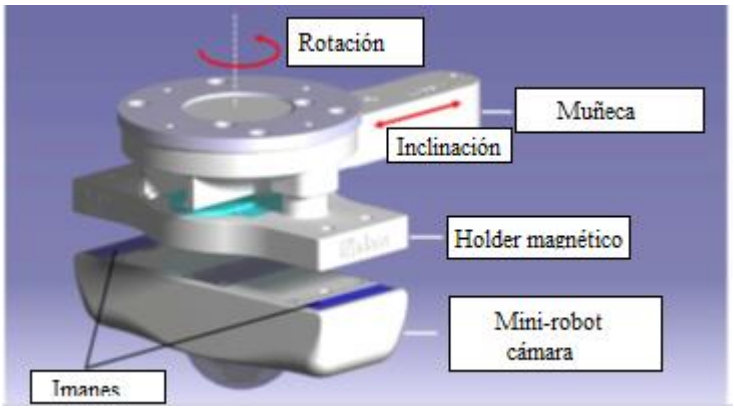

Figura 3: Componentes y grados de libertad del dispositivo

Mediante una serie de experimentos in-vitro realizados para validar el dispositivo, se encontró una limitación importante: el desplazamiento del mini-robot provoca un desplazamiento del sistema magnético que controla la inclinación de la cámara, y por tanto se produce un cambio indeseado de dicha inclinación. Además, posteriores pruebas demostraron, entre otras cuestiones, la necesidad de introducir algún mecanismo que permitiera recoger el mini-robot en caso de que se soltara de la sujeción magnética. Por lo tanto, si el robot no fuese inalámbrico sino que estuviese alimentado por un cable, permitiría recogerlo en caso de caída. Esto a su vez reduciría el tamaño de la cámara, evitaría el problema de la autonomía de la batería y permitiría introducir un motor en el dispositivo para controlar la orientación de la cámara [2].

Como resultado de estos experimentos se ha realizado un nuevo diseño del mini-robot cámara, en el que la orientación de la cámara se controla mediante un conjunto de motores. De esta manera se evitan los cambios indeseados en la inclinación de la cámara debido al movimiento de los imanes. Este nuevo diseño requiere de un sistema de control de los motores que permita controlar la inclinación de la cámara. Es aquí donde se enmarca el presente trabajo fin de grado, que tiene como objetivo principal el diseño e implantación de dicho sistema de control, y su integración en la plataforma desarrollada por el grupo de investigación de Robótica Médica del Departamento de Ingeniería de Sistemas y Automática.

1.3 Objetivos del proyecto

Como ya se ha mencionado, el nuevo diseño del mini-robot requiere de un control mediante motores de la rotación y de la inclinación de la cámara. Por lo tanto, el objetivo general del presente trabajo fin de grado se define como el diseño e implantación del sistema de control de la orientación de la cámara de un dispositivo intra-abdominal diseñado para cirugía laparoscópica de puerto único. Asimismo, será necesaria la integración de dicho sistema de control en una plataforma robótica que dispone de una arquitectura basada en ROS. Para ello será necesario cumplir con los siguientes objetivos específicos:

1. Diseño y desarrollo de un sistema de control de la orientación de la cámara de un dispositivo intra-abdominal basado en componentes comerciales de bajo costo con el objeto de visualizar una zona de interés dentro de la imagen laparoscópica. En este punto reside el grueso del trabajo, y puede desglosarse en los siguientes subobjetivos:

   1.1. Diseño y desarrollo de la interfaz persona-máquina. La zona de interés de la imagen laparoscópica se selecciona mediante un interfaz persona-máquina, que acepta tanto comandos de voz como comandos introducidos por teclado.
   1.2. Estudio del mini-robot cámara. Se ha realizado un estudio del diseño del mini-robot cámara para la selección del sistema electrónico de control, necesario para controlar los grados de libertad de orientación del mismo.
   1.3. Diseño y desarrollo del sistema de control. Se ha diseñado una arquitectura de control de los ángulos de giro de los motores que proporcionan una determinada orientación de la cámara que permite visualizar la zona de interés seleccionada.

2. Integración del sistema de control desarrollado en una arquitectura basada en ROS. Con este objetivo se busca una integración funcional completa del dispositivo dentro de un sistema más amplio, que engloba otros componentes desarrollados dentro de las líneas de investigación del grupo de investigación. Este punto puede desglosarse en los siguientes subobjetivos:

   2.1. Estudio de la arquitectura en la que va a ser incluido el sistema de control y la correspondiente interfaz, para su comunicación con el resto de sistemas.
   2.2. Integración del sistema de control diseñado y su correspondiente interfaz.
   2.3. Validación del sistema. Se ha validado todo el sistema desarrollado a través de pruebas experimentales que demuestran tanto el funcionamiento del sistema de forma aislada según las especificaciones de control impuestas, como su correcta integración dentro del sistema global basado en la arquitectura ROS.

1.4 Estructura de la memoria

La memoria del presente trabajo fin de grado se encuentra dividida en seis capítulos y las referencias bibliográficas. Excepto este primer capítulo y el dedicado a las *Conclusiones y Trabajos futuros*, los restantes se inician con una introducción, que expone la problemática a resolver junto con una breve descripción del contenido, y terminan con unas conclusiones, que destacan los aspectos más importantes de cada capítulo. La organización del contenido de este proyecto se describe en el presente apartado.

El **capítulo 2*, Descripción del mini-robot cámara,*** trata sobre el diseño del mini-robot, donde se explica sus diferentes partes, funcionamiento y componentes. El **capítulo 3, *Arquitectura hardware,*** trata sobre el hardware empleado para el desarrollo del trabajo, describiendo la función de cada elemento y el motivo de su elección.

El **capítulo 4*, Arquitectura software,*** describe el software desarrollado así como el esquema de control implementado, detallando el funcionamiento de las interfaces, los cálculos requeridos para el diseño del control, la implementación de los distintos programas y la arquitectura ROS creada para la interconexión de los distintos códigos.

El **capítulo 5*, Experimentación, presenta*** tanto las simulaciones teóricas como las pruebas experimentales que se han realizado para validar el funcionamiento del esquema de control desarrollado.

Finalmente, el **capítulo 6*, Conclusiones y trabajos futuros,*** se tratarán, como su nombre indica, las conclusiones que han surgido de la finalización del presente trabajo y los trabajos futuros que se pueden derivar de éste.

Capítulo 2: Descripción del mini-robot cámara

2.1 Introducción

En el presente capítulo se describe el actual prototipo de mini-robot cámara, sobre el cual se realizará el control de los motores. Para ello se tratarán tanto su diseño físico, como sus mecanismos de transmisión de movimiento, los actuadores de los que precisa y la mini cámara, necesaria para la captación de la imagen. Además se hablará sobre el brazo robótico que interacciona con él y lo desplaza cuando es necesario.

2.2 Diseño del mini-robot

El actual prototipo de mini-robot presenta una forma casi cilíndrica y redondeada en sus extremos, como se puede ver en la figura 4, en la que la cámara adopta una posición central. El dispositivo tiene unas dimensiones de 100 mm de largo por 20 mm de diámetro, las cuales presentan una mejora importante con respecto a sus predecesores [2]. Dicha mejora consiste en que su sección (entendiendo ésta como un corte en un plano perpendicular a su eje longitudinal) es menor y, por lo tanto, resulta más sencillo de introducir en la cavidad abdominal a través de la incisión que realizan los cirujanos.

Así mismo, el robot cuenta con un total de cuatro leds situados a ambos lados de la mini cámara y que se pueden observar en la figura 4. Las cubiertas en las que se apoyan éstos se pueden quitar retirando los cuatro tornillos situados al lado de los leds y permiten inspeccionar su interior.

Una vez retiradas las cubiertas, véase la figura 5, se puede observar que el mecanismo de transmisión de movimiento, del cual se hablará detalladamente más adelante, consiste en un sistema de cuerdas y elementos elásticos. Dicho mecanismo

permite que la mini cámara rote tanto en el eje longitudinal del mini-robot como en su eje transversal. Además, a diferencia de los anteriores modelos [2], éste dispone los imanes para sujetarse a la pared de la cavidad abdominal en el interior de la cubierta, en lugar de en su zona exterior, lo que reduce el rozamiento durante el desplazamiento del robot.

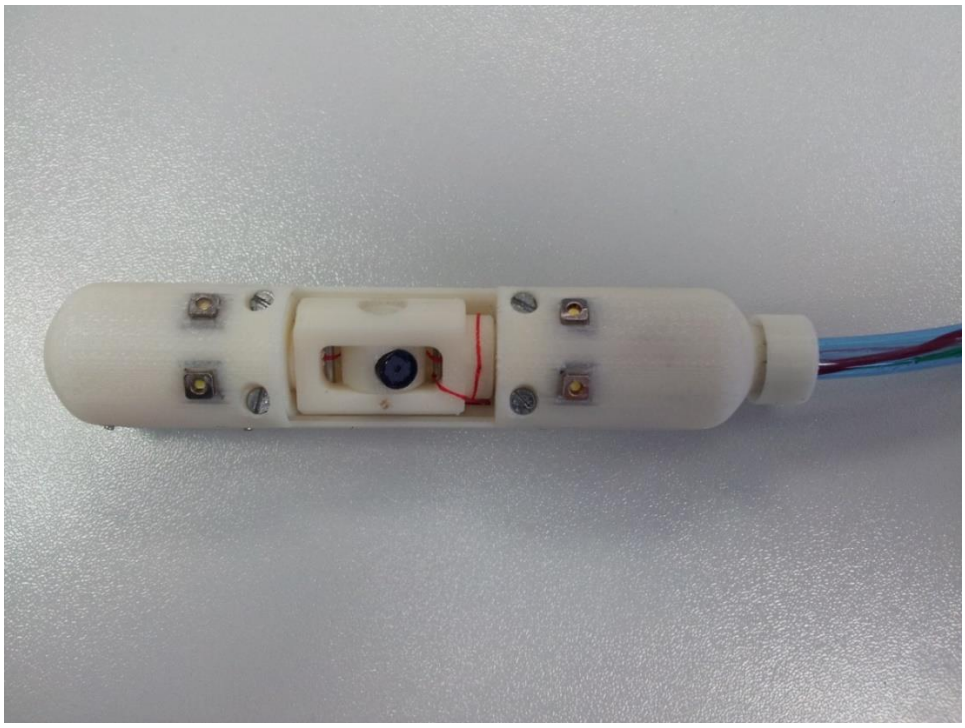

Figura 4: Mini-robot cámara montado

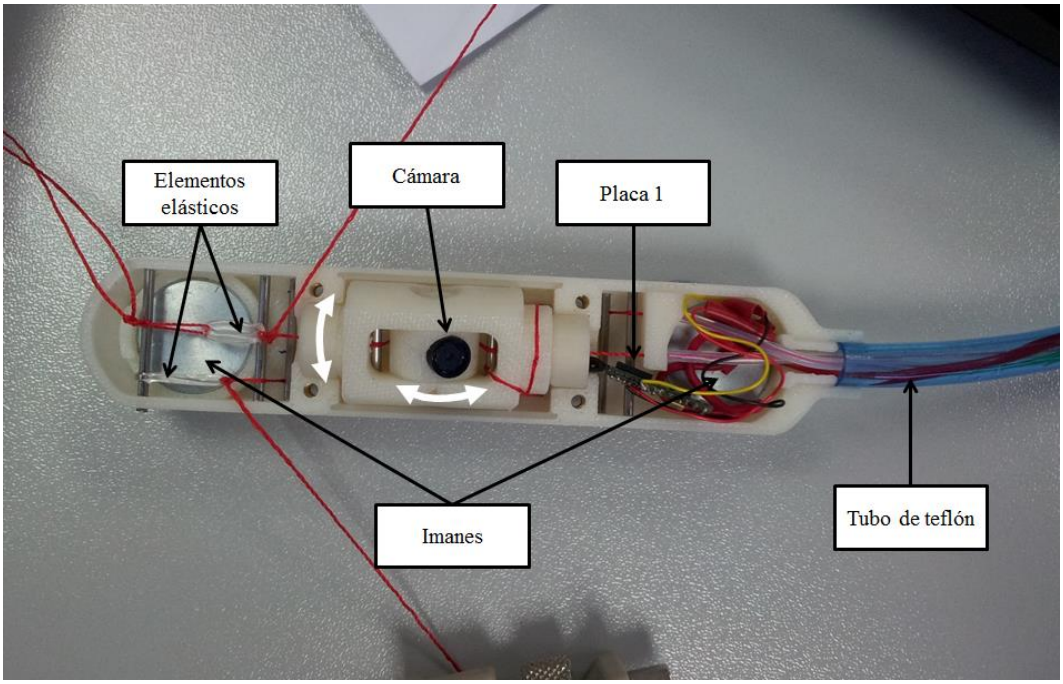

Figura 5: Mini robot cámara con las cubiertas retiradas

Tanto las cuerdas necesarias para transmitir el movimiento de la cámara, como los cables necesarios para su alimentación y la transmisión de la imagen (soldados a la placa de la cámara) se encuentran dentro de un tubo de teflón flexible. Dicho tubo comunica el mini-robot con una caja externa (Figura 6), que contiene los motores y una segunda placa de la cámara. Tal y como se puede ver en la mencionada figura, el giro de los motores provocará un giro de sus respectivas aspas en el sentido que marcan las flechas, tensando la cuerdas y desencadenando la transmisión de movimiento que se detallará en el siguiente apartado. Los motores, al igual que la cámara y sus respectivas placas, se tratarán en sendos apartados posteriores de este mismo capítulo.

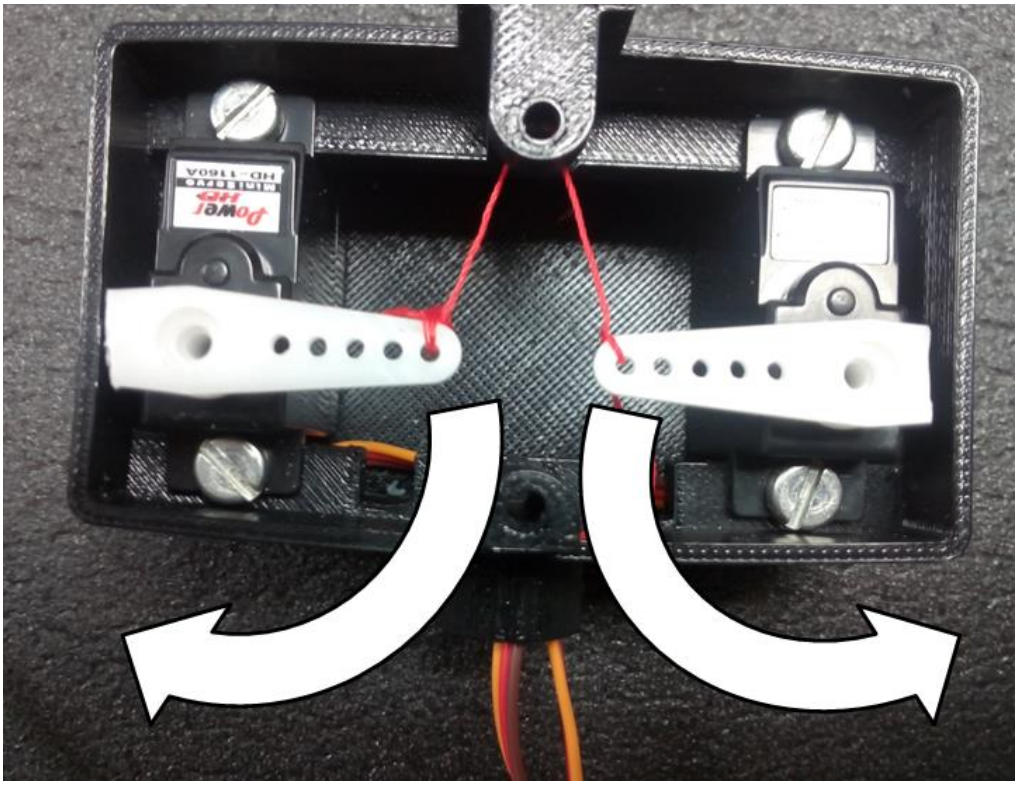

Figura 6: Caja que integra los motores

2.3 Mecanismos de transmisión de movimiento

Tal y como se ha mencionado anteriormente, el mecanismo de transmisión del movimiento de los motores a la mini cámara consiste en un sistema formado por varias cuerdas y dos elementos elásticos. El funcionamiento de éstos, el cual se puede observar en la figura 7, se detalla a continuación.

En primer lugar, atendiendo a la figura 7, si se produce un giro en el motor 1 el aspa situado sobre éste tensará la cuerda 1 en el sentido que marca la flecha amarilla, produciendo el giro longitudinal de la mini cámara en el sentido que marca la flecha del

mismo color y estirando el elemento elástico 1. Si tras tensar la cuerda ésta se libera con un movimiento del motor en sentido contrario, el elemento elástico 1 recupera su forma, volviendo a trasladar la mini cámara a su posición inicial.

En cuanto al segundo eje, el funcionamiento es idéntico al anterior. Si se produce un giro en el motor 2, la rotación de su aspa tensa la cuerda 2 produciendo a su vez el giro transversal de la mini cámara en la dirección marcada por la flecha azul, estirándose el elemento elástico 2. Si se libera la cuerda, dicho elemento recupera su forma original desplazando la mini cámara a su posición inicial.

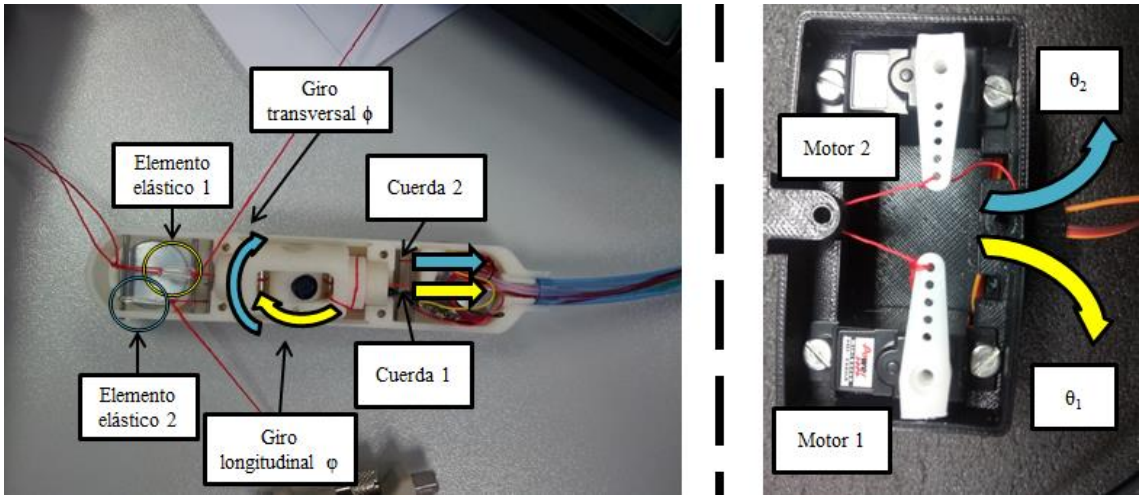

Figura 7: Funcionamiento de los mecanismos de transmisión de movimiento

2.4 Actuadores

Por definición, un actuador es un dispositivo capaz de transformar energía, en este caso eléctrica, en la activación de un proceso con la finalidad de generar un efecto sobre un sistema. Existen una gran variedad de actuadores y, dependiendo de su finalidad, se ha de escoger uno u otro.

Para esta ocasión, por sus características, se ha decidido escoger dos servomotores miniatura HD-1160ª [17]. Se han elegido éstos frente a otros tipos de motores debido a que incorporan un control de posición, tienen un tamaño reducido, y ofrecen un par de salida suficiente para conseguir el giro deseado de la cámara. Los servomotores en cuestión cuentan con una amplitud de 180º ± 10º, un peso de 17 gramos y se alimentan a 4.8-6V y 680-800mA respectivamente con un par máximo de 2.7 kg-cm. En la figura 8 se puede observar el mencionado servomotor.

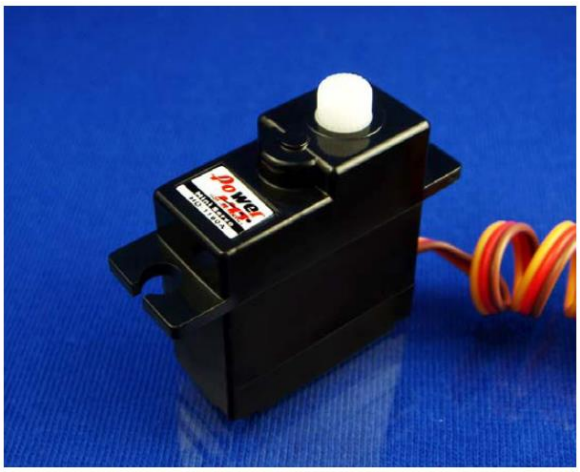

Figura 8: Servomotor utilizado

2.5 Mini cámara

La mini cámara en cuestión que se ha utilizado para el presente trabajo se trata de una cámara ultra mini CMOS [16] de la casa MISUMI, mostrada en la figura 9. Cuenta un diámetro de 5.5mm, una resolución de 656 x 496 píxeles, una distancia focal de entre 2 y 7 cm y se alimenta con 5-12V y 160mA. Además, la cámara tiene un rango de temperatura de funcionamiento de hasta 45ºC, temperatura que en un principio, teniendo en cuenta que la media del cuerpo suele estar entorno a los 37ºC, se considera suficiente.

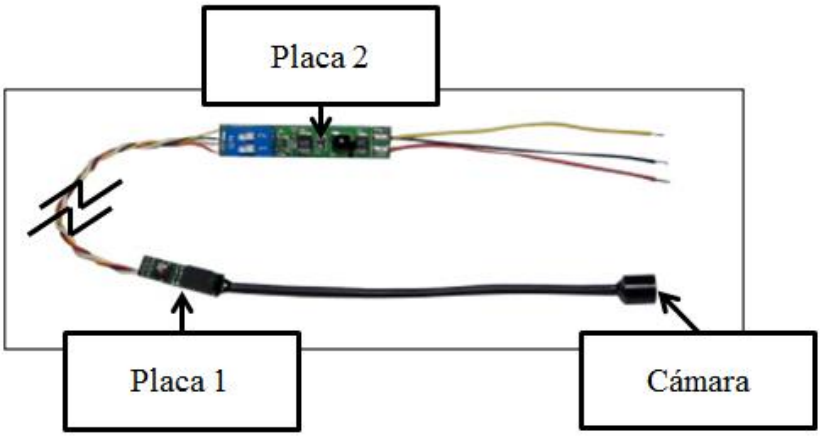

Figura 9: Componentes de la mini cámara

La cámara, señalada en la figura 9, se sitúa en el centro del mini robot mientras que la placa más pequeña, denominada placa 1, es la que se coloca en la parte derecha

del mismo y se encarga de procesar la señal proveniente de la cámara. La placa más grande, denominada placa 2, es la que se encuentra en la caja de los motores, debajo de éstos, y se encarga tanto de convertir la señal que le llega de la placa 1 en una salida analógica, como de de controlar la alimentación de la cámara. En cuanto a los cables que se pueden observar en la figura a la salida de la placa 2, el rojo se conecta a la 5V, el negro a 0V y el amarillo contiene la salida de vídeo.

Debido a que el cable que une las dos placas es demasiado corto y tiene que ir por dentro del tubo de teflón, es necesario cortarlo, colocarlas en sus respectivas posiciones y volverlas a unir mediante las mencionadas soldaduras, con unos cables de la longitud apropiada.

2.6 Brazo robótico

Tal y como se ha mencionado, es necesaria la presencia de un brazo robótico para poder desplazar el holder magnético a lo largo de la pared abdominal de forma automática [2]. Para ello se utiliza un manipulador de la marca Barrett Technology, modelo WAM, de siete grados de libertad. En la figura 10 puede observarse dicho modelo. En su efector final es necesario colocar el holder magnético que sujetará al mini-robot, el cual se puede ver también en la figura..

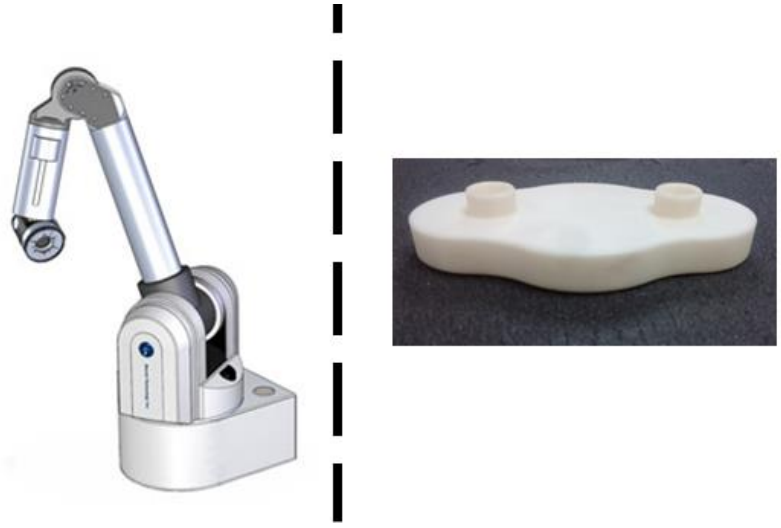

Figura 10: Barrett WAM y holder magético.

El brazo incorpora un control híbrido de fuerza-posición [10], diseñado por el grupo de investigación, con la finalidad de que el desplazamiento se produzca sin ejercer una fuerza excesiva que pueda dañar al paciente. Además, dicho control incluye un módulo de compensación de pares que mantiene el efector final del brazo perpendicular a la pared abdominal en todo momento.

2.7 Conclusiones

A lo largo del capítulo se ha descrito tanto el diseño del mini robot como sus mecanismos de transmisión de movimiento, además de los dispositivos de los que se sirve, tanto los servomotores como de la mini cámara, y del sistema con el que interactúa, el robot WAM. En su conjunto, se trata de un sistema dotado de una cierta complejidad, la cual hay que tener en cuenta a la hora de diseñar el control deseado.

Capítulo 3: Arquitectura Hardware

3.1 Introducción

En el presente capítulo se describe la arquitectura hardware empleada para la realización del trabajo y se explicará, brevemente, cada uno de sus componentes y porqué se ha escogido cada uno de ellos. En primer lugar se describirán las características principales que todo microcontrolador ha de tener, para a continuación detallar los microcontroladores escogidos y los elementos adicionales de los que hacen uso.

3.2 Microcontrolador

Por definición, un microcontrolador es un circuito integrado programable, es decir, que es capaz de ejecutar una serie de instrucciones que serán almacenadas previamente en su memoria. Para ello el dispositivo ha de contar con:

- **CPU:** Es el componente capaz de procesar la información y ejecutar el código almacenado en su memoria, generando unas "salidas" en función de unas "entradas".
- **Memorias:** Alojan tanto las instrucciones (el código) como los datos que necesita la CPU durante la ejecución de éstas. Pueden ser volátiles, si se pierden los datos al cortar la alimentación, o persistentes si se mantienen al cortarla.
- **Diferentes patillas de E/S:** Necesarias para comunicar la CPU con el exterior, permitiendo tanto la entrada como la salida de datos del mismo.

Para llevar a cabo el control de movimientos de la cámara integrada en el mini-robot, ha sido necesaria la elección de dos microcontroladores de bajo coste, por lo que se ha escogido la marca **Arduino** [18]-[19]. Además, se escogió porque dispone de una serie de drivers que permiten la integración del mismo en una arquitectura basada en ROS.

Arduino cuenta con diversas PBCs (Printed Circuit Board) que incorporan un microcontrolador reprogramable y una serie de pines hembra unidas internamente a las patillas E/S de éste, permitiendo la conexión de componentes de forma sencilla y cómoda. Asimismo cuenta con un lenguaje de programación propio, inspirado en otro ya existente denominado processing, y un software gratis, libre y multiplataforma que facilita el desarrollo del software con una gran cantidad de librerías y que permite cargarlo al microcontrolador para su ejecución a través de un cable USB.

Además, Arduino se caracteriza por su versatilidad. Existen una gran cantidad de placas que se interconectan con éste añadiéndole funcionalidades muy diversas. Las placas Arduino que se han escogido con sus correspondientes características y motivos de su elección se exponen a continuación.

3.2.1 Arduino UNO

Arduino UNO [20] es una placa de microcontrolador basada en el chip (circuito integrado) ATmega328. La placa en cuestión cuenta con 14 pines de E/S digitales, 6 pines de entrada analógica, conexión USB, conexión para alimentación por corriente y botón de reset, entre otras cosas. Se puede ver una ilustración de la placa en la figura 11.

Figura 11: Arduino UNO

Así mismo, la placa, una vez conectada a la corriente ya sea por puerto USB o por conexión directa, puede suministrar 5V o 3.3V si es necesario, con una intensidad de 40 o 50mA respectivamente. También cuenta con una serie de pines especiales, de entre los cuales se necesitarán los pines A4 (SDA) y A5 (SLC) para la interconexión mediante protocolos I^2C/TWI entre los dos microcontroladores. Las características más importantes de la placa se muestran a continuación.

Características	
Microcontrolador	ATmega 328
Voltaje operativo	5V
Voltaje de entrada (recomendado)	7-12V
Voltaje de entrada (límites)	6-20V
Pines E/S digitales	14
Pines de entrada analógica	6
Corriente DC por pin E/S	40 mA
Corriente DC en pin 3.3V	50mA
Memoria flash	32 KB, 0,5KB reservados para bootloader
SRAM	2KB
EEPROM	1KB
Frecuencia de reloj	16MHz
Longitud	68.6mm
Ancho	53.4mm
Peso	25g

Cabe destacar que la Memoria flash, la SRAM y la EEPROM son las memorias típicas del microcontrolador, en este caso el ATmega 328. La primera y la última son persistentes, siendo la flash donde se almacenan los sketches (programas) que se cargan en la placa, y la SRAM es una memoria volátil que el procesador utiliza para ir realizando sus cálculos.

Se ha escogido una placa Arduino UNO por ser de las más baratas, y por tener las características necesarias de memoria para almacenar uno de los sketches, el que controla el movimiento de los servomotores. Los cables de control de dichos servomotores se conectan a los pines 3 y 5, los cables de alimentación a 5V y los cables de tierra a GND.

3.2.2 Arduino MEGA ADK

Arduino MEGA ADK [21] se trata de una placa de microcontrolador basada en el ATmega 2560 que incorpora y mejora las funcionalidades de Arduino UNO. Principalmente se caracteriza por poseer una interfaz host de conexión USB con teléfonos Android, pero no se ha escogido por esto sino por sus características de memoria. El dispositivo cuenta con 54 pines digitales de E/S, 16 pines analógicos de entrada, conexión USB y de toma de corriente y botón de reset, entre otras cosas. Los pines SDA y SLC, necesarios para la comunicación mediante protocolos I^2C/TWI entre mircocontroladores, son los pines 20 y 21. En la figura 12 se puede observar una ilustración de la susodicha placa.

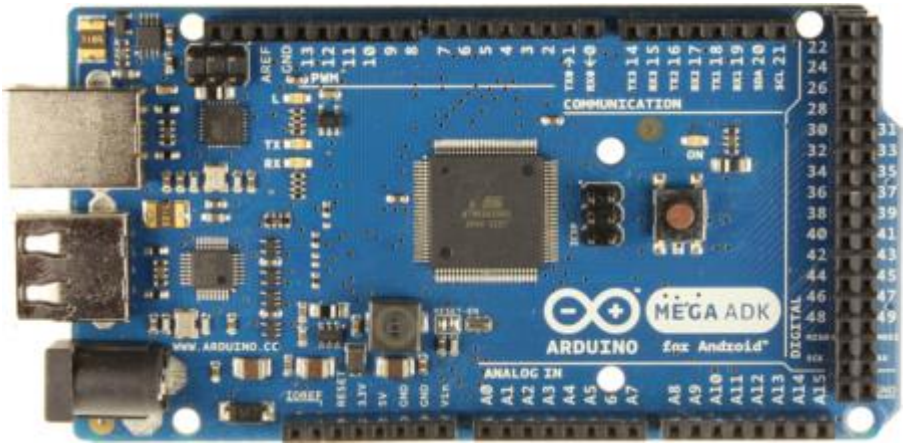

Figura 12: Arduino MEGA ADK

Las principales características del dispositivo se muestran a continuación:

Características	
Microcontrolador	Atmega 2560
Voltaje operativo	5V
Voltaje de entrada (recomendado)	7-12V
Voltaje de entrada (límites)	6-20V
Pines E/S digitales	54
Pines de entrada analógica	16
Corriente DC por pin E/S	40 mA
Corriente DC en pin 3.3V	50mA
Memoria flash	256 KB, 8KB reservados para bootloader
SRAM	8KB
EEPROM	4KB
Frecuencia de reloj	16MHz
Longitud	101,52mm

Características	
Ancho	53.3mm
Peso	36g
Chip host USB	MAX3421E

Se ha escogido también para este trabajo un Arduino MEGA ADK porque será necesario desarrollar dos sketches diferentes, uno para el control de los motores y otro para la interfaz. El sketch de control de los motores tiene un tamaño suficientemente pequeño como para poder ejecutarlo en un Arduino UNO, mientras que el sketch de la interfaz necesita un microprocesador con una capacidad de almacenamiento mayor. Es por eso que se ha elegido un Arduino MEGA ADK.

3.3 Reconocimiento Automático del Habla

El Reconocimiento Automático del Habla (RAH) [22] o reconocimiento automático de voz, es una rama de la Inteligencia Artificial en la que el objetivo último es permitir la comunicación hablada entre seres humanos y computadores. Esto, para éste caso en particular, puede resultar extremadamente útil, ya que el cirujano realiza su labor de pie, con las manos ocupadas, y necesitará de un reconocimiento de voz para poder accionar los comandos que rigen el movimiento del mini-robot si no dispone de un asistente. Además, la comunicación mediante comandos de voz resulta muy intuitiva para el cirujano, puesto que es la forma natural en la que se comunica con un asistente humano. Es por esto que se ha decidido hacer uso de la shield de Arduino EasyVR.

3.3.1 EasyVR 2.0

EasyVR es un circuito integrado que se puede encontrar incrustado en una shield o como un dispositivo independiente. Una shield [19] no es más que una PBC que se coloca en la parte superior de una placa Arduino y que se conecta a ella mediante el acoplamiento de sus pines, sin necesidad de cables, ampliando y complementando las funcionalidades de ésta.

En concreto, EasyVR [23] es un módulo multipropósito de RAH, diseñado para añadir un reconocimiento de habla versátil y robusto a casi cualquier tipo de aplicaciones. En la figura 13 se puede observar dicha shield, la cual se alimenta con 5V, por lo que la tensión que suministra Arduino es suficiente.

Figura 13:Shield EasyVR 2.0

La shield incorpora una serie de comandos predefinidos y permite almacenar hasta 28 comandos personalizados. Así mismo el dispositivo se caracteriza por poder reproducir audios y soportar varios lenguajes de interlocutor.

En cuanto a su funcionamiento, el dispositivo cuenta con cuatro modos de operación:

- **UP:** Para actualizaciones de firmware o para cargar sonidos en la memoria flash de la shield.
- **PC:** Permite utilizar EasyVR con su interfaz gráfica en el ordenador, a través del puerto USB de la placa Arduino.
- **HW:** Permite controlar la shield desde el sketch de arduino, a través de los pines TX y RX (Hardware serial port).
- **SW:** Similar al anterior, pero utilizando los pines 12 y 13 como TX y RX (Software serial port), permitiendo de esta manera el uso del Serial de arduino.

Como EasyVR junto con Arduino MEGA ADK se utilizará para la implementación de la interfaz, se necesita de una comunicación serie del microcontrolador con el ordenador para mostrar información por pantalla y recogerla por teclado. Por lo tanto, para este caso en concreto se utilizará en todo momento el modo de operación SW, para así poder utilizar el Serial de Arduino.

Cabe destacar que la shield [24] trae consigo un pequeño micrófono que se puede quitar y poner cuando sea necesario. También cuenta con salida de audio para altavoces de hasta ocho ohmios, salida para auriculares y una interfaz gráfica para diseñar los comandos de voz con facilidad. Ésta última no se utilizará en el presente trabajo ya que se programarán directamente en el sketch de Arduino.

3.4 Conclusiones

A lo largo del capítulo se han tratado, tanto los microcontroladores necesarios para la implementación del control deseado, con sus respectivas características y motivos de su elección, como la shield de la que hace uso uno de ellos. En consecuencia, como se puede comprobar, el presente trabajo se ciñe al requisito de bajo coste que se le impuso desde un principio.

Capítulo 4: Arquitectura Software

4.1 Introducción

En este capítulo se tratará tanto la arquitectura de control de orientación de la cámara diseñada como su implementación, así como los estudios y cálculos que han sido necesarios para ello. En cuanto a la implementación, se planteará la arquitectura ROS propuesta para el sistema y se describirán cada uno de sus componentes, prestando especial atención tanto a la implementación del control de los motores como a las interfaces que el cirujano utilizará para así interactuar con el mini-robot.

4.2 Arquitectura de control

Un cirujano es un profesional que trabaja de pie, con las manos ocupadas y, en el tipo de procedimiento en el que se engloba este trabajo (cirugía laparoscópica de puerto único), su única referencia es la imagen que le llega a través de una cámara y que se plasma en un monitor. Dicha imagen no abarca todo el campo de trabajo, por lo que para visualizar ciertas zonas de la cavidad abdominal es necesario un cambio en la perspectiva de la imagen que se consigue a través de una variación en orientación y/o posición de la cámara, accionada por comandos que el cirujano introducirá al sistema (por ejemplo, ante el comando *derecha* el centro de la imagen visualizada en la pantalla se desplazará una distancia determinada hacia esta dirección). Para ejecutar dicho cambio de perspectiva, el mini-robot está dotado de dos grados de libertad, visibles en la figura 14: rotación de la mini cámara respecto al eje *x*, también llamada rotación transversal (ϕ), rotación respecto a *y*, también llamada rotación longitudinal (φ). Por otro lado, se dispone de dos grados de libertad adicionales de desplazamiento a lo largo del eje *x* (x_R) y desplazamiento a lo largo del eje *y* (y_R) proporcionados por el brazo robótico WAM al que está fijado el mini-robot mediante el holder magnético.

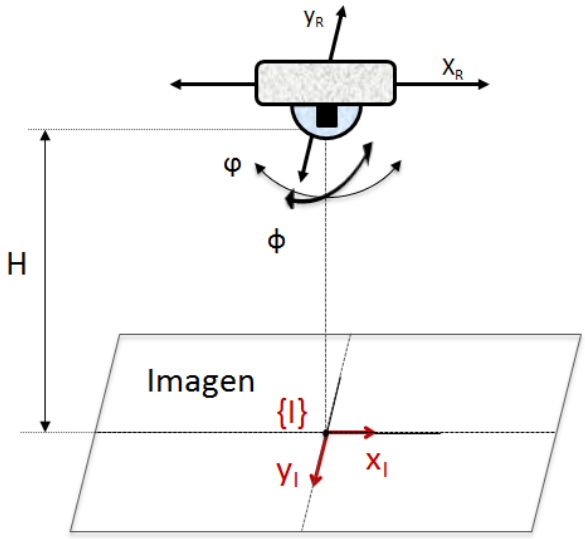

Figura 14: Grados de libertad de la mini-cámara

Cabe destacar que el sistema de referencia local de la imagen *{I}* se sitúa justo debajo de la cámara, a una distancia *H* de la misma, coincidente con la vertical, y se desplaza junto con el mini-robot en caso de intervenir el brazo robótico.

Además de las características indicadas en cuanto a los grados de libertad del dispositivo, es posible que el cirujano necesite de un mayor nivel de detalle en la imagen capturada en determinadas ocasiones, por lo que la mini cámara debe precisar de un zoom digital.

Por lo tanto, se expone a continuación tanto la arquitectura de control necesaria para el control de la orientación de la cámara, visible en la figura 15 como un diagrama de bloques, como los cálculos requeridos para ello. En primer lugar se dará una visión general de la arquitectura para así comprender el funcionamiento de la misma, seguida de la explicación detallada de cada uno de sus componentes.

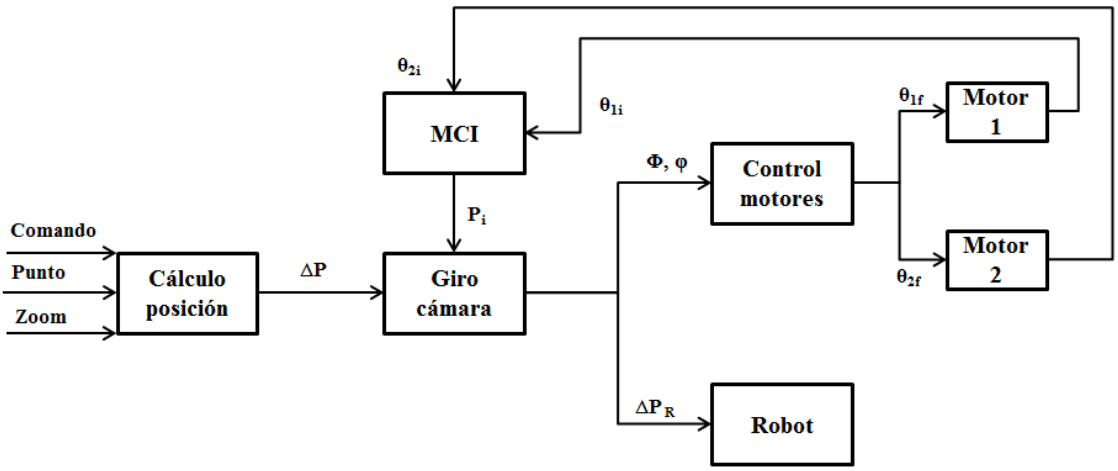

Figura 15: Diseño de arquitectura de control

El primer bloque, denominado ***Cálculo posición***, puede recibir como entrada un *comando* (orden de desplazamiento predefinida), un desplazamiento a un *punto* determinado o el valor de *zoom* a realizar, y traduce las entradas mencionadas a una orden de desplazamiento ΔP del centro de la imagen. El segundo bloque, ***Giro cámara***, recibe como entrada desde el anterior la orden de desplazamiento del centro de la imagen ΔP, además la posición inicial *Pi* en la que se encuentra el centro de la imagen antes de realizar el desplazamiento desde del bloque ***MCI*** (modelo cinemático inverso), y ha de decidir si para realizar el movimiento deseado ha de recurrir al desplazamiento del brazo robótico. En caso de que dicho desplazamiento no sea necesario, le transfiere al bloque de **control de motores** los ángulos ϕ y φ (figura 18) que la cámara ha de adoptar para que los motores obren en consecuencia, alcanzando el punto deseado. Si el desplazamiento del brazo fuese necesario, el bloque ***Giro cámara*** ha de desencadenar tanto el desplazamiento del WAM, pasándole el incremento de posición $\Delta P_R=(x_R,y_R)$ al bloque ***Robot***, como el control adecuado de los motores, pasándole sus correspondientes ángulos ϕ y φ.

4.2.1 *Cálculo posición*

Como se puede ver en la figura 15, el sistema puede recibir como entrada un *comando*, es decir, una orden de desplazamiento del centro de la imagen predefinida, un desplazamiento de dicho centro a un punto determinado o el valor de *zoom* a realizar. Tanto el *comando* como el desplazamiento a un *punto* generan un vector de desplazamiento relativo del centro de la imagen $\Delta P_0=(\Delta x_0, \Delta y_0)$ en mm, el cual se verá influenciado por el valor de zoom para acabar generando el valor ΔP que proporcionará el bloque a su salida.

Comando

Los comandos que acepta la interfaz con sus correspondientes valores en *x* e *y* de ΔP_0 se muestran a continuación.

Comando	ΔP_0 **(mm)**
Arriba	(0, 10)
Abajo	(0, -10)
Derecha	(10, 0)
Izquierda	(-10, 0)

Punto

El punto introduce en el sistema unas posiciones concretas Δx_0, Δy_0 en lugar de un comando predefinido, generando de igual forma una orden de desplazamiento del centro de la imagen ΔP_0.

Zoom

Para que, independientemente del valor del zoom que tenga la cámara, la sensación de movimiento de la imagen que obtenga el cirujano con cada orden de desplazamiento sea la misma, el presente bloque ha de recalcular los valores de ΔP_0, adaptándose a dicho zoom. Con este fin, se parte del supuesto de la expresión 1 el cual se va a demostrar, siendo *z* el valor de zoom y *H* la altura.

$$\Delta z \rightarrow H = \frac{H_0}{z} \qquad (1)$$

Es decir, que una variación de zoom es equivalente a que la cámara adopte una nueva altura, a razón de H_0/z, donde H_0 es la altura real a la que se encuentra la cámara. Para demostrar dicho supuesto se realizó un estudio en el que se utilizó la cámara digital de un teléfono móvil Aquaris E4, ya que al igual que la mini cámara cuenta con un zoom digital. Este estudio se dividió en dos fases.

Fase 1. En primer lugar se colocó la cámara a 20 cm de altura y se varió el zoom desde 1 hasta 3, marcando en un folio en blanco para cada valor de zoom las esquinas de la imagen que se mostraba en la pantalla del móvil. A partir de dichas esquinas, con una regla se puede medir la superficie que se visualizaba en la imagen. En la figura 16 se puede ver cómo se realizó este estudio.

Fase 2. Se repitió el mismo proceso pero variando las alturas en lugar del factor de zoom. Las alturas utilizadas se calcularon a partir de la fórmula anterior con los distintos valores de zoom utilizados en la Fase 1, y con un valor de zoom de uno en la cámara.

Los resultados obtenidos del estudio se pueden ver en la siguiente tabla donde en la primera columna se enumera cada uno de los casos contemplados, en la segunda y tercera columna se muestran los factores de zoom y la superficie de la imagen para la Fase 1 del estudio, y finalmente, la tercera y cuarta columna muestran la altura de la imagen respecto a la cámara y la superficie de la imagen para la Fase 2 del estudio.

Caso	Fase 1		Fase 2	
	Factor de zoom a 20cm	Superficie de la imagen (cm^2)	Altura con zoom 1 (cm)	Superficie de la imagen (cm^2)
1	1.00	432.00	20.00	432.00
2	1.50	186.44	13.33	183.69
3	2.00	103.84	10.00	108.00
4	2.60	60.30	7.69	61.20
5	3.00	43.32	6.67	46.61

Como se puede observar a partir de los datos de la tabla anterior, los valores de superficie de la imagen son los mismos colocando la cámara a una altura H_0/z que al hacer un zoom con un factor z con la cámara situada a una altura H_0, por lo que queda demostrada la validez del supuesto.

Figura 16: Estudio práctico para la determinación de la relación entre el zoom y la altura

Por lo tanto, teniendo en cuenta que la altura disminuye de ésta manera, si se pretende que la sensación de movimiento en la imagen que capta la cámara sea siempre la misma, indiferentemente del valor de zoom, es necesario que la distancia d_0 (figura 17) recorrida por el centro de la imagen disminuya de igual forma, obteniendo de esta forma una distancia d. Dicha disminución se muestra en la expresión 2.

$$d = \frac{d_0}{z} \qquad (2)$$

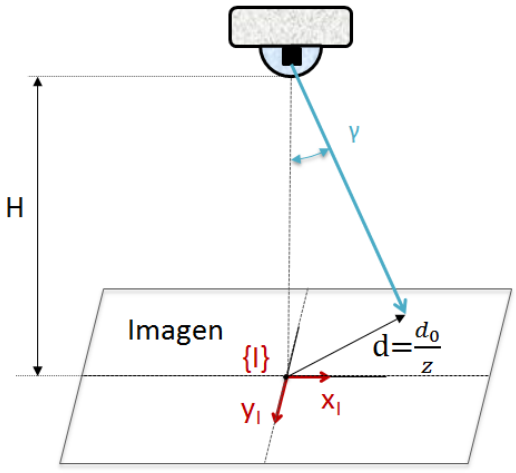

Figura 17: Adaptación del mini-robot al valor de zoom

En consecuencia, Δx y Δy se ven afectados de la forma indicada en las expresiones 3 y 4.

$$\Delta x = \frac{\Delta x_0}{z} \qquad (3)$$

$$\Delta y = \frac{\Delta y_0}{z} \qquad (4)$$

Este parámetro $\Delta P=(\Delta x, \Delta y)$ que se acaba de calcular es el que el bloque de ***Cálculo de posición*** le pasa al bloque ***Giro cámara*** cuando recibe como entrada un *comando* o un *punto*.

4.2.2 *MCI*

El presente bloque, implementa el modelo cinemático inverso de la cámara, que recibe como entrada los valores angulares en los que se encuentran los servomotores antes de realizar el desplazamiento y, en función de éstos, calcula la posición inicial P_i del centro de la imagen con respecto al sistema local de referencia (la vertical de la cámara). Teniendo en cuenta las expresiones 5 y 6:

$$\varphi_i = \frac{\theta_{1rep} - \theta_1}{k1} \qquad (5)$$

$$\phi_i = \frac{\theta_{2rep} - \theta_2}{k2} \qquad (6)$$

Siendo θ_1 y θ_2 los ángulos en los que se encuentran los servomotores, θ_{1rep} y θ_{2rep} los ángulos de los servomotores para los que la orientación de la mini cámara coincide con la vertical (ángulos de reposo) y *k1* y *k2* los coeficientes que relacionan la variación del ángulo θ_1 con la variación de φ y la variación del ángulo θ_2 con la variación de ϕ respectivamente. Dicha relación se muestra en las expresiones 7 y 8.

$$k1 = \frac{\Delta\theta_1}{\Delta\varphi} \qquad (7)$$

$$k2 = \frac{\Delta\theta_2}{\Delta\phi} \qquad (8)$$

Conocidos estos valores, la posición inicial P_i en la que se encuentra el centro de la imagen se muestra calculada en las expresiones 9, 10 y 11.

$$p_i = (x_i \quad y_i) \qquad (9)$$

$$x_i = \tan(\varphi_i) \cdot H \qquad (10)$$

$$y_i = \tan(\phi_i) \cdot H \qquad (11)$$

4.2.3 *Giro cámara*

El actual bloque recibe como entrada información desde dos bloques distintos: desde el bloque ***Cálculo posición*** llega una orden de desplazamiento del centro de la imagen ΔP, mientras que desde el bloque ***MCI*** le llega la posición inicial P_i en la que se encuentra el centro de la imagen antes de realizar el movimiento ordenado. En consecuencia, la posición P_f, visible en la figura 18, a la que se desplazará el centro de la imagen cuando le llegue una orden de desplazamiento ΔP, se muestra en la expresión 12.

$$P_f = P_i + \Delta P \qquad (12)$$

Llegados a este punto, hay que decidir si para realizar dicho desplazamiento a P_f es necesario o no el desplazamiento del mini-robot. Para ello se hará uso de un ángulo γ, el cual, como se puede ver en la figura 18, se define como el ángulo que forman la vertical con el vector de orientación de la cámara, trazado desde la mini-cámara hasta P_f, contenido en el plano del desplazamiento.

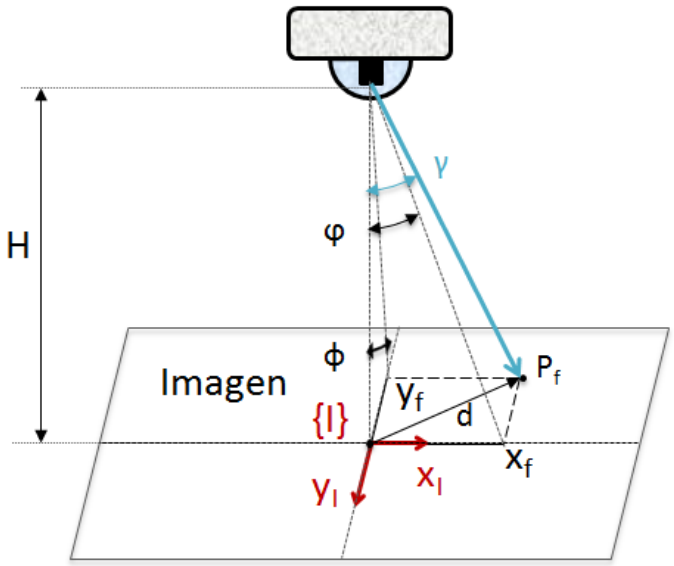

Figura 18: Desplazamiento del centro de la imagen

Dicho ángulo γ no puede ser mayor de un ángulo γ_{max}, el cual viene impuesto por las limitaciones físicas de la cámara. En el caso de que lo sea, se necesitará del desplazamiento del mini-robot. Conociendo los valores x_f e y_f en los que se situaría el centro de la imagen tras su desplazamiento y la altura en la que se encuentra la cámara, el valor de γ es el calculado en las expresiones 13 y 14.

$$d = \sqrt{{x_f}^2 + {y_f}^2} \qquad (13)$$

$$\gamma = atan2(d, H) \qquad (14)$$

Siendo *d* la distancia que hay desde el sistema local de referencia *{l}* hasta P_f. Una vez calculado γ, se pueden dar dos casos:

Ángulo γ menor que γ_{max}

En esta situación, los valores de φ y ϕ que se le pasarían al bloque ***Control motores***, y el valor de ΔP_R que se le pasaría al bloque ***Robot*** serían los mostrados en las expresiones 15, 16, 17.

$$\varphi = atan2(x_f, H) \qquad (15)$$

$$\phi = atan2(y_f, H) \qquad (16)$$

$$\Delta P_R = (x_R, y_R) = (0,0) \qquad (17)$$

Los valores x_R e y_R son nulos ya que no se necesita del desplazamiento del mini-robot.

Ángulo γ mayor que γ_{max}

En el caso de que el ángulo sea mayor que γ_{max}, el control deberá tanto asignarle a γ un valor de $\gamma_{max}/2$, como calcular el valor de ΔP_R que ha de desplazarse el mini-robot (por la acción del brazo robótico). La elección de un valor γ de $\gamma_{max}/2$ se debe a que permite a la mini cámara tanto seguir avanzando en el mismo sentido del desplazamiento, como un amplio recorrido en sentido contrario, todo ello sin tener que volver a recurrir al robot WAM.

Por lo tanto, es necesario calcular en primer lugar los valores de desplazamiento $\Delta P_R = (x_R, y_R)$ del robot, para posteriormente calcular los valores φ y ϕ. Ambas acciones se pueden ver en la figura 19.

Por un lado, de dicha figura se deduce la expresión 18

$$d_R = d - \left(H \cdot \tan\left(\frac{\gamma_{max}}{2}\right)\right) \qquad (18)$$

Siendo d_R la distancia en línea recta que ha de recorrer el brazo. En consecuencia, dado que se puede calcular el valor de β, ángulo que se puede ver en la figura 19, los valores de x_R e y_R serían los calculados en las expresiones 19, 20 y 21.

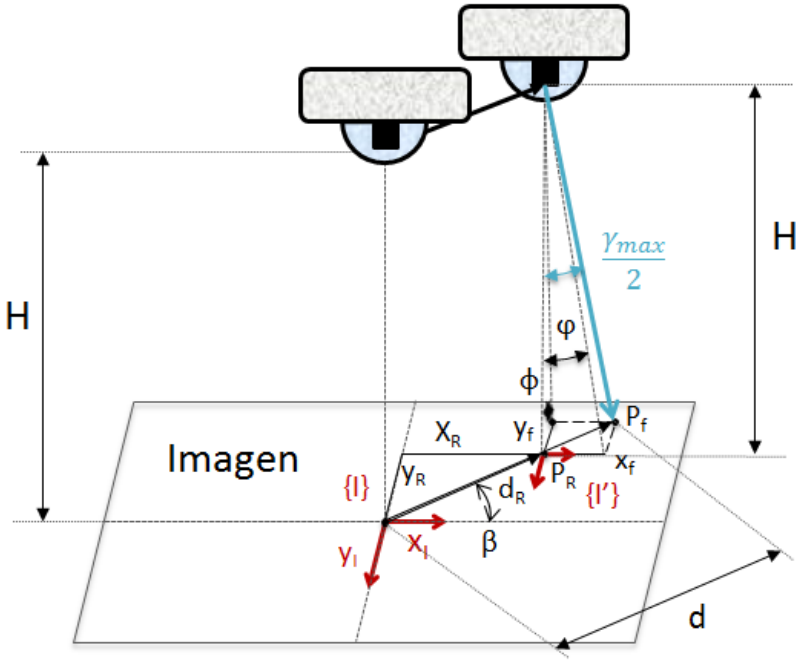

Figura 19: Desplazamiento del mini-robot

$$\beta = atan2(x_f, y_f) \qquad (19)$$

$$x_R = \sin(\beta) \cdot d_R \qquad (20)$$

$$y_R = \cos(\beta) \cdot d_R \qquad (21)$$

Una vez conocidos estos parámetros se pueden calcular los valores de φ y ϕ, mostrados en las expresiones 22 y 23.

$$\varphi = atan2\left((x_f - x_R), H\right) \qquad (22)$$

$$\phi = atan2\left((y_f - y_R), H\right) \qquad (23)$$

Los valores de φ y ϕ el actual bloque se los pasará al bloque ***Control motores***, y los valores x_R e y_R se los pasará al bloque ***Robot*** para que realice el desplazamiento. Cabe destacar que el operador atan2 utilizado anteriormente se trata de la función arcotangente, función inversa a la tangente, cuya solución pertenece al intervalo $[-\pi, \pi]$, y que nos permite conocer su ángulo.

4.2.4 *Control motores*

El presente bloque recibe como entrada los valores de φ y ϕ y, en función de éstos, calcula qué ángulos θ_1 y θ_2 hay que transmitirle a los bloques de ***Motor 1*** y ***Motor 2*** para que los servomotores obren en consecuencia, alcanzando el centro de la imagen la

posición deseada. Como se ha mencionado anteriormente, los servomotores cuentan con unos valores angulares de reposo θ_{1rep} y θ_{2rep}, para los que la orientación de la mini cámara coincide con la vertical y el centro de la imagen coincide con el sistema de referencia local, además de unos valores *k1* y *k2* que relacionan la variación de θ_1 y θ_2 con la variación de φ y ϕ respectivamente. En consecuencia los valores angulares a calcular son mostrados en las expresiones 24 y 25.

$$\theta_1 = \theta_{1rep} - \varphi \cdot k1 \qquad (24)$$

$$\theta_2 = \theta_{2rep} - \phi \cdot k2 \qquad (25)$$

Los valores $\varphi \cdot k1$ y $\phi \cdot k2$ restan a θ_{1rep} y θ_{2rep} porque los valores por debajo de los ángulos de reposo producirán un desplazamiento positivo del centro de la imagen, mientras que los valores por encima de éstos producirán desplazamientos negativos del mismo.

4.3 Implantación del esquema de control

4.3.1. Entorno ROS

ROS (Robotic Operative System) [11]-[20] se trata de un conjunto de librerías de código abierto, cuyo objetivo es simplificar la tarea a la hora de crear sistemas robóticos complejos y robustos a partir de diferentes plataformas robóticas, integrando todas ellas e intercomunicándolas, formando una red. Aunque se trata de un sistema multiplataforma, principalmente funciona en Ubuntu. Entre los principales elementos de cualquier arquitectura ROS destacan:

- **Nodos:** Son la parte más básica de la red que forma toda arquitectura basada en ROS. Los nodos son procesos ejecutables, es decir, programas que se ejecutan en el entorno de ROS, que cumplen una función específica y que se comunican entre ellos.
- **Nodo maestro (Roscore):** Es el nodo principal de ROS, necesario para el funcionamiento de la arquitectura. Sin él, los nodos no se encontrarían ni podrían intercambiar mensajes entre ellos.
- **Mensajes:** Los nodos de ROS se pueden comunicar entre ellos asíncronamente a través de mensajes. Éstos son simples estructuras de datos compuestas por campos tipificados.
- **Topics:** Son "intermediarios" en la comunicación entre nodos, "depósitos" en los que se almacenan los mensajes. Los nodos pueden publicar mensajes en los topics o subscribirse a éstos. El funcionamiento es simple, si un nodo

publica un mensaje determinado en un topic, todos los nodos subscritos a éste lo leerán. Puede haber múltiple publicadores y subscriptores para un único topic y un único nodo puede publicar y/o subscribirse a múltiples topics.

- **Servicios:** Se trata de otra forma de comunicación, esta vez síncrona, entre nodos. Para ello atiende a protocolos de petición/respuesta.

4.3.2 Estructura propuesta

Para el presente trabajo se necesita diseñar el control de movimientos de la cámara y su correspondiente interfaz, y todo ello ha de estar integrado en una arquitectura ROS para que puedan comunicarse con las plataformas que ya tiene diseñadas el proyecto MARCUS. Por lo tanto, la arquitectura ROS en cuestión que se ha propuesto se compone de cuatro nodos además del nodo maestro, y puede verse en la figura 20. Los nodos *imagen cámara* y *robot WAM* hacen referencia al software que controla la imagen de la cámara y los movimientos del brazo robótico respectivamente, dichos nodos no entran en el alcance de éste trabajo ya que los aporta el grupo de investigación, pero sí que se necesita incluirlos ya que hay que enviarles ciertos mensajes.

Además de los cuatro nodos anteriormente mencionados, en la figura 20 se ha incluido un quinto nodo con líneas discontinuas, el nodo *interfaz voz*, que realmente no forma parte de la arquitectura pero que se comunica con ésta a través del nodo *control de motores*. La no inclusión de éste en la arquitectura ROS se debe a motivos de incompatibilidad que se tratarán más adelante en el subapartado implementación del nodo *interfaz voz*. Por lo tanto, los nodos que integran el sistema y los topics de los que hacen uso son:

Nodo *control de motores*

El nodo *control de motores* implementa los bloques ***MCI***, ***Giro cámara*** y ***Control motores*** de la arquitectura de control, y está desarrollado en un sketch de Arduino. Dicho nodo está subscrito al topic *ordenPos*, del cual obtiene las órdenes de desplazamiento, y publica en el topic *moverWAM* la orden de desplazamiento del brazo robótico en caso de que sea necesario. Además, publica en el topic *ordenZoom* los valores de *zoom* que le llegan desde el nodo ficticio *interfaz voz* y puede recibir de éste las órdenes de desplazamiento del centro de la imagen a través de ΔPos. Así mismo, cabe destacar que el presente nodo, implementado en Arduino, se conecta a la red ROS a través de un nodo puente, ya que es la forma que tiene Arduino de integrarse en ROS.

Nodo *interfaz teclado*

El nodo *interfaz teclado* implementa el bloque ***Cálculo posición*** de la arquitectura de control y está desarrollado en C++. Dicho nodo publica las órdenes de

desplazamiento en el topic *ordenPos* y los valores de zoom introducidos en el topic *ordenZoom*.

Nodo *interfaz voz* (ficticio)

El nodo *interfaz voz* se trata de un nodo ficticio (no conectado directamente a la arquitectura ROS sino a través del nodo *control de motores*) que, al igual que el nodo *interfaz teclado*, implementa bloque ***Cálculo posición*** de la arquitectura de control, pero esta vez en un sketch de Arduino. El actual nodo pasa las órdenes de desplazamiento del centro de la imagen al nodo *control de motores* a través de ΔPos y los valores de zoom introducidos a través de *zoom*, para que éste a su vez los publique.

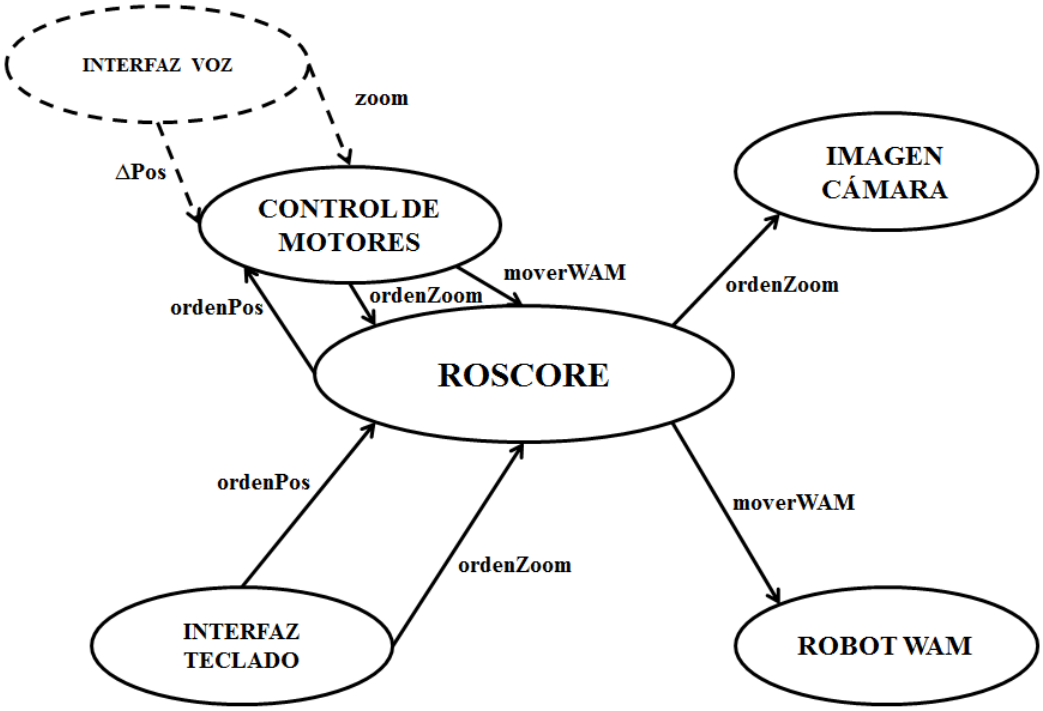

Figura 20: Diseño de arquitectura basada en ROS

Nodo *imagen cámara*

Este nodo controla la imagen que capta la cámara y se suscribe al topic *ordenZoom*, del que saca el valor de zoom que ha de hacer.

Nodo *robot WAM*

El actual nodo controla el movimiento del brazo robótico WAM y está subscrito al topic *moverWAM*, del que extrae la orden de desplazamiento.

Topics

Los diferentes topics, mencionados anteriormente durante el desarrollo de los diversos nodos, se exponen a continuación:

- ***ordenPos*:** Los mensajes que en él se publican son de tipo String y contienen la orden de desplazamiento ΔP del centro de la imagen, mencionada en el apartado Arquitectura de control. En consecuencia, el formato de dichos mensajes está compuesto por dos números enteros de tres dígitos cada uno. Además, ambos valores van precedidos por su signo y se encuentran separados por *";"*, tal y como se puede ver en el ejemplo expuesto en la expresión 26.

$$+000;+000; \qquad (26)$$

- ***moverWAM*:** Los mensajes que en él se publican son también de tipo String y contiene los valores x_R e y_R que el brazo robótico ha de desplazarse. Para ello, el formato de los mensajes está compuesto por dos números enteros de dos dígitos cada uno, acompañados de sus correspondientes signos, tal y como se puede ver en la expresión 27.

$$+00;+00 \qquad (27)$$

- ***ordenZoom*:** Los mensajes que en él se publican son de tipo Float32 y contienen los valores de zoom que ha de hacer la cámara.

4.3.4 Implementación del nodo *interfaz voz*

El nodo *interfaz voz* implementa el bloque ***Cálculo posición*** de la arquitectura de control. Esta interfaz, implementada en un sketch de Arduino y que debido al espacio que ocupa se carga en un Arduino MEGA ADK, utiliza la shield EasyVR 2.0 para la introducción de comandos de voz. Dichos comandos necesitan de un entrenamiento previo que se realiza nada más ejecutar la interfaz, tras indicarle el usuario que está preparado para realizarlo. Las funciones que permite esta interfaz se muestran a continuación.

Modo interfaz por voz:

- Introducir comandos predeterminados de desplazamiento del centro de la imagen (arriba, abajo, derecha, izquierda). Para ello basta con decir "arriba", "abajo", "derecha", "izquierda" y se desplazará una distancia predefinida.
- Aumentar o disminuir el valor de zoom. Diciendo "aumenta" o "disminuye" se produce una variación de zoom prefijada de ±0.5.
- Volver a entrenar los comandos de voz. Para ello hay que decir "entrenar".
- Cambiar al modo interfaz por teclado, diciendo "interfaz teclado".

Además, en caso de que no funcione correctamente el modo interfaz por voz, en cualquier momento el usuario puede cambiar a modo interfaz por teclado pulsando en el mismo la letra *t*, ya sea en mayúscula o en minúscula.

Modo interfaz por teclado:

- Introducir comandos predeterminados de desplazamiento del centro de la imagen (arriba, abajo, derecha, izquierda pulsando "w", "s", "d", "a" respectivamente).
- Aumentar o disminuir el valor de zoom. Pulsando "+" y "-" se producen variaciones de ±0.5.
- Mover el centro de la imagen a un punto concreto respecto a la posición actual, pulsando "p".
- Cambiar al modo interfaz por voz pulsando "v".

Cuando se introduce un comando a través del modo interfaz por teclado, a la interfaz le es indiferente si es mayúscula o minúscula ya que las entiende por igual.

Es importante mencionar que la comunicación de la interfaz con el Arduino que implementa el nodo *control de motores* se hace a través de protocolos I^2C/TWI, utilizando la librería *Wire* y siendo la interfaz el maestro en dicha comunicación. Se ha realizado este tipo de conexión ya que, al tratarse de una interfaz, necesita hacer uso del Serial de Arduino para mostrar información por pantalla y recogerla por teclado. Por lo tanto, el puerto USB queda ocupado por la comunicación serie de Arduino y no se puede conectar al nodo puente de ROS, quedando "aislada" de la arquitectura. Como se ha visto anteriormente, ese aislamiento no es tal ya que si se necesita enviar un valor de zoom a la arquitectura ROS, por comunicación I^2C la interfaz se lo pasa al nodo *control de motores* y éste tal y como lo recibe lo publica en el topic ordenZoom.

Para el desarrollo del sketch que implementa la interfaz por voz ha sido necesario incluir una serie de librerías. Éstas son:

- ***EasyVR***
- ***SoftwareSerial***
- ***Wire***

La primera se puede descargar de la página web de EasyVR y aporta una serie de funciones que facilitan el trabajo con la shield, mientras que la tercera habilita la comunicación entre Arduinos mediante protocolos I^2C/TWI. Así mismo, la segunda librería permite hacer un puerto serie software en los pines 12 y 13 para la shield, para así ponerla en el modo de operación SW y poder utilizar el Serial de Arduino (como se mencionó en el capítulo de arquitectura hardware). Además, el sketch cuenta con una serie de constantes y variables globales, necesarias para el correcto funcionamiento del mismo. Entre ellas destacan:

- ***despCentro*** (constante): Desplazamiento (d_0) del centro de la imagen por cada comando predefinido (derecha, izquierda…).
- ***variacionZoom*** (constante): Valor de zoom que se incrementa o decrementa con cada orden de variación del mismo.
- ***Zoom*** (variable global): almacena el valor de zoom en el que se encuentra la cámara.

De entre todas las funciones presentes en el sketch, las más importantes son las siguientes:

- ***setup( )***: Inicializa la configuración de EasyVR comprobando si está conectada, eliminando todos los comandos almacenados anteriormente y reseteándola, poniéndole un tiempo de espera infinito y así esté todo el tiempo escuchando, indicándole que el idioma en el que se van a introducir los comandos es español, creando los comandos y entrenándolos. Para esto último llama a la función ***EntrenarTodo***.
- ***loop( )***: Está compuesta por una serie de instrucciones que se repiten de forma cíclica, indefinidamente. En concreto, en este caso se llama a las funciones ***InstruccionesVoz***, ***InterfazVoz***, ***InstruccionesTeclado*** y a ***InterfazTeclado***.
- ***InstruccionesVoz( )***: Muestra por pantalla las instrucciones de uso del modo interfaz por voz.
- ***InterfazVoz( )***: Reconoce los comandos de voz y desencadena la ejecución de las instrucciones necesarias para cada uno de ellos.
- ***InstruccionesTeclado( )***: Muestra por pantalla las instrucciones de uso del modo interfaz por teclado.
- ***InterfazTeclado( )***: Reconoce los comandos introducidos por teclado y desencadena la ejecución de las instrucciones necesarias para cada uno de ellos.
- ***HacerZoom(float var)***: Calcula el nuevo valor de zoom y manda al nodo de control de motores dicho valor para que éste a su vez lo publique en ROS.
- ***EnviarPosicion(int x, int y)***: Función que es llamada tanto desde la función InterfazVoz como desde InterfazTeclado, y es la encargada de enviar la orden de desplazamiento ΔP al nodo control de motores, a través de comunicación I^2C, con su correspondiente formato.
- ***EntrenarTodo( )***: Espera a que el usuario esté listo, preguntando por pantalla y esperando una respuesta por teclado. Cuando la respuesta es afirmativa (Pide pulsar "s" para continuar y es indiferente si dicho carácter viene en mayúsculas o en minúsculas) desencadena todo el proceso de entrenamiento de los comandos.

Todo el código generado durante la implementación de la interfaz cumple los estándares de programación de la BSSC [26]. Un problema que se encontró durante

dicha implementación fue la escasa documentación sobre el uso de la biblioteca EasyVR, teniendo únicamente la facilitada por el manual.

4.3.5 Implementación del nodo *interfaz teclado*

El nodo *interfaz teclado* implementa el bloque ***Cálculo posición*** de la arquitectura de control. Esta interfaz se presenta como una alternativa a la anterior, si no se quiere o no se puede hacer uso del Arduino MEGA ADK y la shield EasyVR. Se implementa en C++ directamente como un nodo de ROS y se comunica con el nodo de control de motores y el programa de control de la imagen de la cámara a través de éste, publicando en los topics *ordenPos* y *ordenZoom* respectivamente. La estética y las funciones que implementa esta interfaz son idénticas al modo interfaz por teclado de la anterior interfaz, con la excepción del comando cambio a comandos de voz, que en ésta no está presente. Por lo tanto las funcionalidades que implementa son:

- Introducir comandos predeterminados de desplazamiento del centro de la imagen (arriba, abajo, derecha, izquierda pulsando “w”, “s”, “d”, “a” respectivamente).
- Aumentar o disminuir el valor de zoom. Pulsando “+” y “-” se producen variaciones de ±0.5.
- Mover el centro de la imagen a un punto concreto respecto a la posición actual, pulsando “p”.

La implementación de esta interfaz es muy similar a la anterior, salvando las diferencias entre el lenguaje propio de Arduino y C++, el modo de comunicación con el resto de la arquitectura y que esta interfaz es únicamente por teclado. Todo el código generado durante la implementación cumple los estándares de programación de la BSSC [26] y las librerías que incluye para llevar a cabo su función son:

- ***Iostream***
- ***String***
- ***Sstream***
- ***Ros***
- ***Std_msgs/String***
- ***Std_msgs/Float32***

La primera es necesaria para el flujo de entrada/salida de información en el programa, mientras que la segunda y la tercera se utiliza para la generación de la orden que se le mandará al nodo de control de motores. Así mismo, las tres últimas permiten la comunicación a través de ROS y el paso de información a través de mensajes, ya sea tipo String o Float. Las constantes que utiliza el programa son las mismas que en la anterior interfaz, y las principales funciones que se pueden encontrar en ésta son:

- ***main( )***: En este programa, a diferencia de en la primera interfaz, no hay una función *setup* que inicializa el mismo y otra *loop* que ejecuta una serie de instrucciones repetidamente, sino que ésta función cubre ambas acciones. Por lo tanto, en primer lugar inicializa la comunicación con ROS y llama a la función ***Instrucciones*** para, en segundo lugar, ejecutar dentro de un bucle lo que en la anterior interfaz era la función *InterfazTeclado* y que en este caso no es una función aparte sino parte del *main*.
- ***Instrucciones ( )***: Muestra por pantalla las instrucciones de uso de la interfaz por teclado.
- ***HacerZoom(float var, float &zoom, int &alturaVar, ros::Publisher ordenZoom)***: Calcula el nuevo valor de zoom y publica en el topic correspondiente en ROS dicho valor.
- ***PublicarPosicion(int x, int y, int alturaVar, ros::Publisher ordenPos)***: Crea la orden de desplazamiento del centro de la imagen ΔP con el formato adecuado y la manda a través de ROS, publicándola en el topic correspondiente.

4.3.6 Implementación del nodo *control de motores*

El nodo *control de motores*, que implementa los bloques ***MCI***, ***Giro cámara*** y ***Control motores*** de la arquitectura de control, se ha implementado en un sketch de Arduino, el cual cumple los estándares de programación que marca la BSSC [26]. Para ello, en primer lugar ha sido necesario incluir una serie de librerías:

- ***Servo***
- ***Wire***
- ***Ros***
- ***Std_msgs/String***
- ***Std_msgs/Float32***

La primera librería está compuesta de funciones que facilitan el control de servomotores en Arduino, mientras que la segunda permite la comunicación entre Arduinos mediante protocolos I^2C/TWI. Por otra parte, las tres últimas librerías permiten la comunicación del sketch con el resto de componentes de la arquitectura ROS.

En la comunicación entre Arduinos el presente sketch actúa como esclavo, siendo el maestro el sketch de interfaz por voz. Es por eso el control de motores no hará nada hasta que le llegue una orden del maestro, funcionando de forma completamente pasiva. Dicho esto, las principales constantes y variables globales que se pueden encontrar en el sketch son:

- ***altura***: altura *H* a la que se encuentra la cámara respecto a las vísceras en el interior de la cavidad abdominal.
- ***pi***: número π, necesario para conversiones entre grados y radianes.
- ***angMax***: ángulo máximo γ_{max} que el vector de orientación de la mini cámara puede realizar con la vertical.
- ***kX***: coeficiente *k1*, el cual relaciona la variación del ángulo θ_1 con la variación de φ.
- ***kY***: coeficiente *k2*, el cual relaciona la variación del ángulo θ_2 con la variación de ϕ.

Además de éstas, el sketch cuenta con una serie de constantes necesarias para la calibración de los motores, cuyos valores se detallarán en el capítulo 5. Por otra parte, las principales funciones que se pueden encontrar en el nodo de control de los motores son:

- ***setup( )***: Inicializa todo el programa. Inicializa la comunicación con ROS y a través de Wire (I^2C). Además Enlaza los servomotores a sus correspondientes pines de control, para finalmente alinear la orientación de la mini cámara con la vertical.
- ***loop( )***: Contiene una serie de instrucciones que se repiten de forma cíclica, indefinidamente. Es aquí donde el control une los distintos valores que le han llegado desde la comunicación I^2C y llama entonces a la misma función que si le llegase la orden desde ROS. Además, si desde dicha comunicación le llega un valor cuyo primer carácter es "z", significa que es un valor de zoom y le envía dicho valor al programa de visualización de la imagen, publicándolo en el topic *ordenZoom*.
- ***ReceiveEvent(int howMany)***: Es la función que llama el programa cuando recibe información a través de comunicación I^2C. Se encarga de distinguir qué valores le han llegado y almacenarlos.
- ***MessageCb(const std_msgs::String& str_msgR)***: Es la función que llama el programa cuando se recibe información a través de ROS en forma de mensaje. Dicha función extrae la información del mensaje y llama a la función ***DecidirMover*** con la orden recibida.
- ***DecidirMover(String orden)***: Es la que implementa el control anteriormente mencionado y a la que se llama cuando se recibe una orden, tanto desde ROS como a través de I^2C.

Cabe destacar que se han desarrollado las mismas funciones por duplicado, para el servo que controla los desplazamientos sobre el eje *X* y los desplazamientos sobre el eje *Y*, porque Arduino no permite pasar los objetos que controlan los servomotores como parámetros a las distintas funciones. Además ha sido necesario incluir dos funciones, ***GradosRadianes*** y ***RadianesGrados***, para pasar de grados a radianes y viceversa, ya que las funciones matemáticas de Arduino (seno, coseno, tangente y arcotangente) trabajan en radianes, mientras que los servomotores trabajan con grados.

4.4 Conclusiones

A lo largo del capítulo se ha detallado, en primer lugar, la arquitectura de control para el control de movimientos de la cámara y, en segundo lugar, su implementación en una arquitectura ROS compuesta por diversos nodos. Dichos nodos (control de motores, interfaz voz, interfaz teclado,…) se intercomunican entre ellos para alcanzar el control deseado y dotan de gran flexibilidad al sistema, permitiendo la posterior adicción de nuevos elementos en el mismo.

Capítulo 5: Experimentos

5.1 Introducción

En el presente capítulo se realizarán una serie de experimentos con la finalidad de demostrar que el control diseñado funciona correctamente. Para ello, en primer lugar será necesario fijar una serie de parámetros para la calibración de los motores y determinar el coeficiente de transmisión de movimiento *k*. A continuación se realizará una simulación del funcionamiento del mismo y se procederá a las pruebas experimentales pertinentes.

Cuando se experimentó con la movilidad de ambos ejes de inclinación del mini-robot, se comprobó que tan sólo el giro longitudinal funcionaba correctamente, ya que, debido a su alto rozamiento, el giro transversal resultaba muy difícil de accionar, haciendo que sus movimientos no fuesen precisos. Cabe destacar que éste es un problema de diseño, por lo que queda fuera del alcance del ámbito del presente trabajo. Es por eso que la calibración de los motores, el cálculo del coeficiente de transmisión de movimiento y la experimentación in-vitro se realizarán únicamente sobre el giro longitudinal, producido por el motor 1. La reducción de la experimentación a un solo eje de giro no resta valor científico al trabajo, ya que los algoritmos desarrollados son iguales para ambos ejes de giro.

5.2 Calibración de los motores

Tal y como se ha mencionado en la introducción, antes de realizar la experimentación in-vitro y la determinación del coeficiente de transmisión de movimiento, se necesita una calibración previa de los motores. Dicha calibración incluye la determinación experimental de:

- **Ángulos iniciales**. Tal y como se vio en el capítulo 2, es necesario fijar las cuerdas a los motores en una posición en la que éstas estén en tensión con el

objetivo de que, en el momento que los motores sufran una variación en su posición angular θ, ésta se transmita a la mini cámara. Estos ángulos pasarán a ser los valores iniciales o mínimos del recorrido que pueden realizar los motores, y se les llamará ángulos iniciales.

- **Ángulos finales**. El recorrido que puede realizar la cámara está condicionado por la elongación máxima que pueden alcanzar los elementos elásticos que tiene en su interior, por lo que los servomotores tendrán un valor angular a partir del cual no podrán ir más allá. Dicho valor se convertirá en el valor máximo o final del recorrido que puedan realizar los motores y se les llamará ángulos finales.
- **Ángulos de reposo**. Debido a la naturaleza del mini-robot, el valor de los ángulos para los que la orientación de la cámara coincide con la vertical no corresponden con los valores medios exactos de los límites de los rangos de movimiento en los dos servomotores, y será necesario determinarlos experimentalmente. Estos ángulos son los llamados ángulos de reposo θ_{rep}.
- **Ángulo máximo γ_{max}**. Una vez determinados los rangos de movimientos de los dos servomotores y sus respectivos ángulos de reposo, a partir de éstos se determinará la libertad de movimiento que tendrá la mini cámara hacia ambos lados de la vertical en sus dos giros (longitudinal y transversal). Conocidos estos cuatro rangos de libertad (dos φ y dos ϕ), se tomará el menor de ellos como el ángulo máximo $\gamma_{\max}$ (mencionado en la arquitectura de control) que puede girar la cámara desde la vertical en el plano del movimiento. Si el punto al que se quiera orientar la cámara produce que este ángulo se supere, será necesario mover el brazo robótico.

Para determinar estos parámetros experimentalmente de forma fácil, se creó un sketch de Arduino denominado *sketch_calibracion*. Dicho programa varía el ángulo de los motores dependiendo de los ángulos que le lleguen por teclado, permitiendo modificarlos gradualmente en función de las necesidades. Como se ha mencionado en la introducción del presente capítulo, estos valores se han calculado únicamente para el motor 1, que controla el giro longitudinal de la cámara.

Por lo tanto, los ángulos inicial, final y de reposo de los servomotores se hallaron experimentalmente, haciendo uso del sketch de calibración. Sin embargo, para la determinación de γ_{max} se introdujo el mini-robot en una caja de metacrilato que simulaba la cavidad abdominal y se sujetó a su parte superior mediante interacción magnética. Una vez situado en dicha posición, se colocó el motor 1 en su ángulo de reposo θ_{1rep}, se marcó en una cuadrícula la posición en la que se situaba el centro de la imagen y a continuación se le introdujo al motor 1 dos giros, uno hacia su ángulo inicial y otro hacia su ángulo final. Se midió en ambos casos la distancia recorrida del centro de la imagen con respecto al punto marcado y se calculó para ambos casos el valor de φ teniendo en cuenta la expresión 28.

$$\varphi = \operatorname{atan2}(d, H) \qquad (28)$$

Siendo *d* dicha distancia recorrida y *H* la altura a la que se encontraba la cámara. De entre estos dos valores de φ se tomó, como se mencionó anteriormente, el menor de ellos como γ_{max}. En caso de funcionar correctamente el giro transversal, se habría escogido como γ_{max} el menor de entre los dos posibles valores de φ y los dos posibles valores de ϕ calculados. En consecuencia, los parámetros necesarios para la calibración son:

Motor 1	
Ángulo inicial	180º
Ángulo final	0º
Ángulo de reposo θ_{rep}	90º
γ_{max}	11º

5.3 Cálculo del coeficiente de transmisión de movimiento

El coeficiente de transmisión de movimiento, *k1*, relaciona la variación del ángulo θ_1 del motor 1 con la variación de ángulo φ, tal y como se vio en la arquitectura de control y se muestra en la expresión 29.

$$k1 = \frac{\Delta\theta_1}{\Delta\varphi} \qquad (29)$$

Donde $\Delta\varphi$ se calculó de la manera expuesta en la expresión 30.

$$\Delta\varphi = \text{atan2}(\Delta x, H) \qquad (30)$$

Para el cálculo de *k1* el procedimiento fue muy similar al utilizado para el cálculo de γ_{max}, ya que se hizo uso de una plantilla cuadriculada. Se marcó en dicha cuadrícula la posición del centro de la imagen cuando el motor 1 estaba en su ángulo de reposo θ_{1rep} y se variaron los ángulos del servomotor, midiendo sobre la cuadrícula los desplazamientos del centro de la imagen con respecto al punto marcado. Dichas variaciones se reflejan en la tabla siguiente.

ΔP (mm)	**Δφ**	**$\Delta\theta_1$**	**k1**
(-15, 0)	-11.3	-60	5.31
(-12, 0)	-9.1	-50	5.49
(-10, 0)	-7.6	-40	5.26
(-6, 5)	-4.9	-30	6.12
(-4, 0)	-3.0	-20	6.67
(3, 5)	2.7	10	3.70
(7, 0)	5.3	20	3.77
(8, 0)	6.1	30	4.92
(14, 0)	10.5	40	3.81

	ΔP (mm)	Δφ	$\Delta\theta_1$	k1
	(16, 0)	12.0	50	4.17
	(19, 0)	14.2	60	4.23
	(23, 0)	17.0	70	4.12
	(26, 0)	19.1	80	4.19
	(27, 0)	19.8	90	4.55
Media				**4.73**

El valor de *k1* empleado en el control de movimiento de la cámara corresponde con la media de los valores obtenidos experimentales, obteniéndose un valor de k1=4.73. La desviación típica es de 0.7546, lo que se considera un valor aceptable para el presente experimento.

Cabe destacar que los valores de φ en la dirección positiva del desplazamiento son mayores que γ_{max} porque, como se dijo anteriormente, éste se corresponde con el menor de los ángulos máximos que puede hacer la cámara con la vertical. En este caso el menor ángulo máximo se produce en la dirección negativa del desplazamiento, pudiendo desplazarse en la dirección positiva un rango mayor.

5.4 Simulación teórica

Dado que la experimentación in-vitro únicamente se ha podido realizar en un sentido de giro, se ha realizado una simulación teórica del comportamiento del sistema teniendo en cuenta los dos grados de libertad con los que se ha diseñado el mini-robot. Para ello se supone una altura del mini-robot de 75 mm, unos coeficientes de transmisión de movimiento de *k1* y *k2* de 4.73 y se le introducirán una serie de puntos, consecutivos, a los que ha de ir desplazándose el centro de la imagen. Por cada orden de desplazamiento ΔP_0 introducida y el zoom aplicado a la imagen en ese instante se registraron los valores ΔP, ΔP_R, P_f, d, φ, ϕ, γ, θ_1 y θ_2 que el control iba calculando, los cuales se recogen en la tabla siguiente.

ΔP_0 (mm)	zoom	ΔP (mm)	ΔP_R (mm)	P_f (mm)	d (mm)	γ (º)	φ (º)	ϕ (º)	θ_1 (º)	θ_2 (º)
(10, 0)	1	(10, 0)	(0,0)	(10,0)	10	7.6	7.6	0.0	54.1	90
(-10, 10)	1	(-10, 10)	(0,0)	(0,10)	10	7.6	0.0	7.6	90	54.1
(20, 20)	1	(20, 20)	(16,24)	(4,6)	7.22	5.5	3.1	4.6	75.6	78.3
(-5, -5)	1	(-5, -5)	(0,0)	(-1,1)	1.42	1.1	-0.8	0.8	93.6	86.3
(-10, -10)	1	(-10, -10)	(0,0)	(-11,-9)	14.2	10.7	-8.3	-6.8	129.4	122.3
(30, 15)	1	(30, 15)	(12.1,3.8)	(6.9,2.2)	7.22	5.5	5.2	1.7	65.2	82.1
(10, -10)	2	(5, -5)	(0,0)	(11.9,-2.8)	12.22	9.2	9.0	-2.2	47.4	100.2
(24, 10)	2	(12, 5)	(16.7,1.5)	(7.2,0.6)	7.22	5.5	5.5	0.5	64.1	87.6

La representación gráfica de los valores obtenidos se muestra en las figuras 21 y 22.

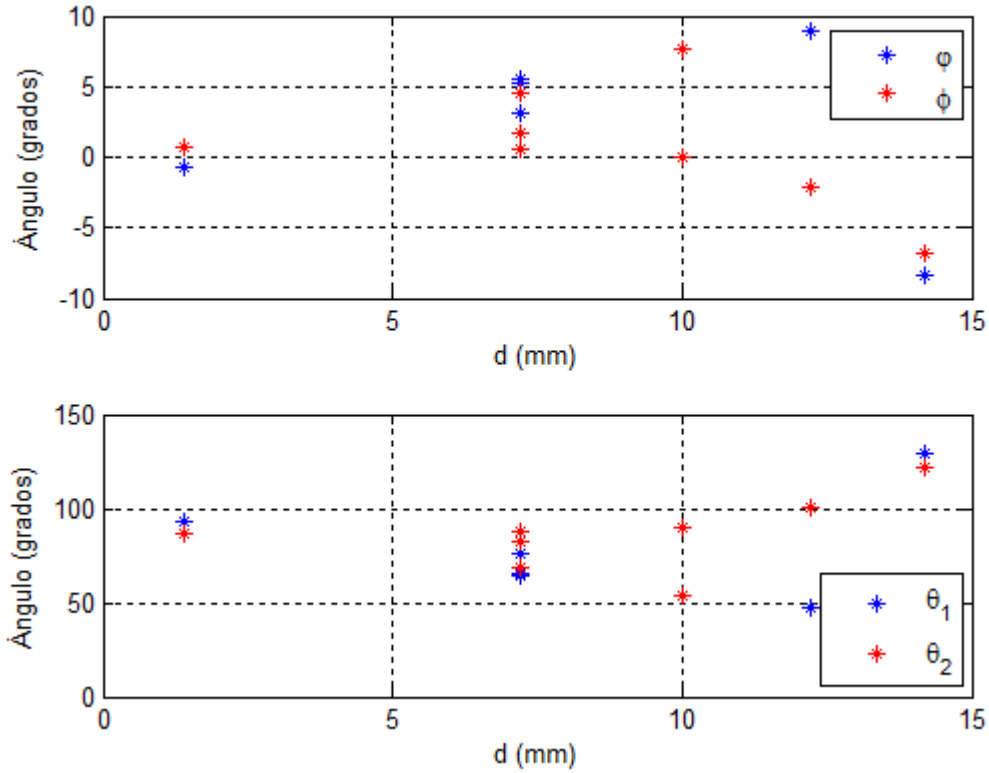

Figura 21: valores de φ, ϕ, θ_1 y θ_2 que la cámara iría adoptando frente a la distancia d

Como se puede observar en la figura 21, los ángulos φ y ϕ, y en consecuencia θ_1 y θ_2 también, trabajan en conjunto para conseguir que la cámara adopte una distancia d determinada con respecto a la vertical. La existencia de varios ángulos φ, ϕ, θ_1 y θ_2 para una misma distancia d se debe a que una misma distancia se puede alcanzar con distintas configuraciones de pares de ángulos. Además, como se puede observar, la segunda gráfica presenta una forma idéntica a la primera pero invertida, y a diferente escala en el eje *Y*. La diferente escala del eje *Y* se debe a los coeficientes *k1* y *k2* de transmisión de movimiento ya mencionados, mientras que la forma invertida se debe a que los valores de θ_1 y θ_2 superiores a sus respectivos θ_{1rep} y θ_{2rep} (ambos de 90º) generan valores negativos en sus correspondientes giros de la cámara, mientras que los valores inferiores a éstos generan valores positivos.

Además, en la figura 22 se puede ver de forma más clara un fenómeno que también aparece en la figura 21, una alta concentración de puntos en *d=7.22*. Esto se debe a que, cada vez que se desplaza el robot WAM, la cámara adopta un ángulo γ de $\gamma_{max}/2$ y, en consecuencia, la distancia d es siempre la misma. Así mismo, se puede observar en la figura 22 que el ángulo γ nunca supera el valor de γ_{max} ya que en tal caso se produce dicho desplazamiento del brazo robótico.

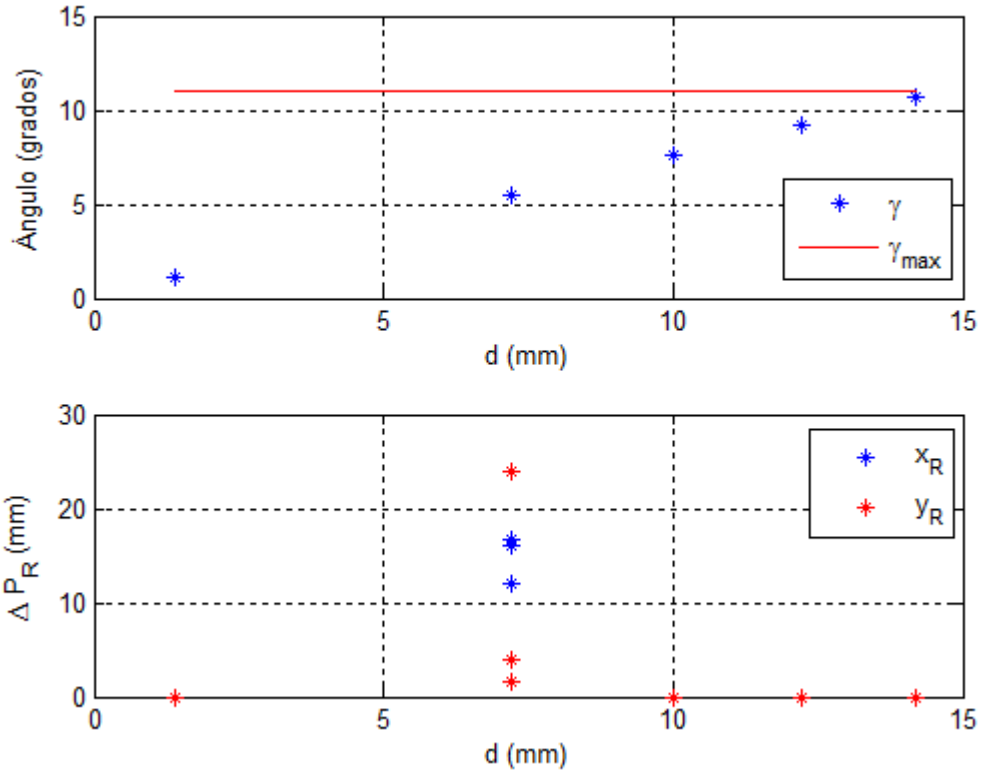

Figura 22: Valores de γ y ΔP_R frente a la distancia d

5.5 Experimentación in-vitro

Una vez calibrado el motor 1 y determinado el coeficiente de transmisión de movimiento *k1*, se procedió a la validación del control desarrollado mediante experimentación in-vitro. Dicha experimentación nos permitió descubrir varias limitaciones del diseño del mini-robot:

- Tan solo admite valores positivos de desplazamiento sobre el eje *X*. Debido a la naturaleza de los elementos elásticos, si se pretende hacer un desplazamiento negativo se precisa de una recuperación en dichos elementos, los cuales están ligeramente tensados cuando el servomotor se encuentra en su posición de reposo θ_{1rep}. Aunque los servomotores liberan cuerda, dicha recuperación no llega a ocurrir a no ser que se produzcan variaciones grandes de ángulo en el servomotor.
- La variación del ángulo en el servomotor ha de ser como mínimo de 20º para que se produzca un desplazamiento del centro de la imagen. Esta limitación también se debe a los mencionados elementos elásticos.

- La interacción magnética entre el mini-robot y el holder magnético provoca que cuando el brazo robótico se desplaza, el rozamiento entre el holder y la superficie sea demasiado grande, lo que hace que el brazo robótico no se desplace la cantidad deseada. Es por esto que no se muestran a continuación órdenes que desencadenen un movimiento del WAM, ya que, si bien conducen a un desplazamiento con las coordenadas correctas, el brazo robótico no realiza un correcto desplazamiento.

Para la realización de las pruebas experimentales se volvió a hacer uso de una cuadrícula. Al igual que en los procedimientos anteriores, se marcó en dicha cuadrícula la posición del centro de la imagen cuando el motor 1 estaba en su ángulo de reposo θ_{1rep} y, una vez hecho esto, se introdujeron las órdenes pertinentes y se anotaron los respectivos desplazamientos del centro de la imagen. El montaje experimental se muestra en las figuras 23 y 24. En la primera de ellas se puede observar el exterior de la caja de metacrilato (que simula la cavidad abdominal), mientras que en la segunda se muestra el mini-robot adherido a la superficie de la caja mediante interacción magnética.

Figura 23: Brazo robótico WAM sujetando el mini-robot

Figura 24: Mini-robot en el interior de la caja de metacrilato que simula la cavidad abdominal

Dicho esto, las órdenes introducidas ΔP_0 junto con sus valores de zoom, ΔP, la distancia teórica (distancia de desplazamiento esperada con respecto a la vertical) y la distancia real que se desplazó el centro de la imagen, junto con el error encontrado, se muestran a continuación. Cabe destacar que las órdenes de desplazamiento no son consecutivas, sino que todas se realizan desde la vertical.

Orden	**ΔP_0 (mm)**	**Zoom**	**ΔP (mm)**	**Distancia teórica (mm)**	**Distancia real (mm)**	**Error (mm)**
Derecha	(10, 0)	1	(10, 0)	10	9.5	0.5
Punto	(12, 0)	1	(12, 0)	12	10.75	1.25
Punto	(5, 0)	1	(5, 0)	5	6.4	-1.4
Punto	(8, 0)	1	(8, 0)	8	7.8	0.2
Derecha	(10, 0)	2	(5, 0)	5	5.35	-0.35
Punto	(12, 0)	2	(6, 0)	6	6.6	-0.6

Con una desviación típica de los errores y una media de los valores absolutos de los mismos de 0.92 y 0.72 respectivamente. En base a dichos errores se puede afirmar que el control funciona correctamente, ya que nunca exceden de milímetro y medio, y se deben a que el parámetro *k1* se ha calculado de forma experimental, lo que

inevitablemente implica un margen de error respecto a su valor real. La representación gráfica de los errores obtenidos frente a las distancias teóricas se muestra en la figura 25.

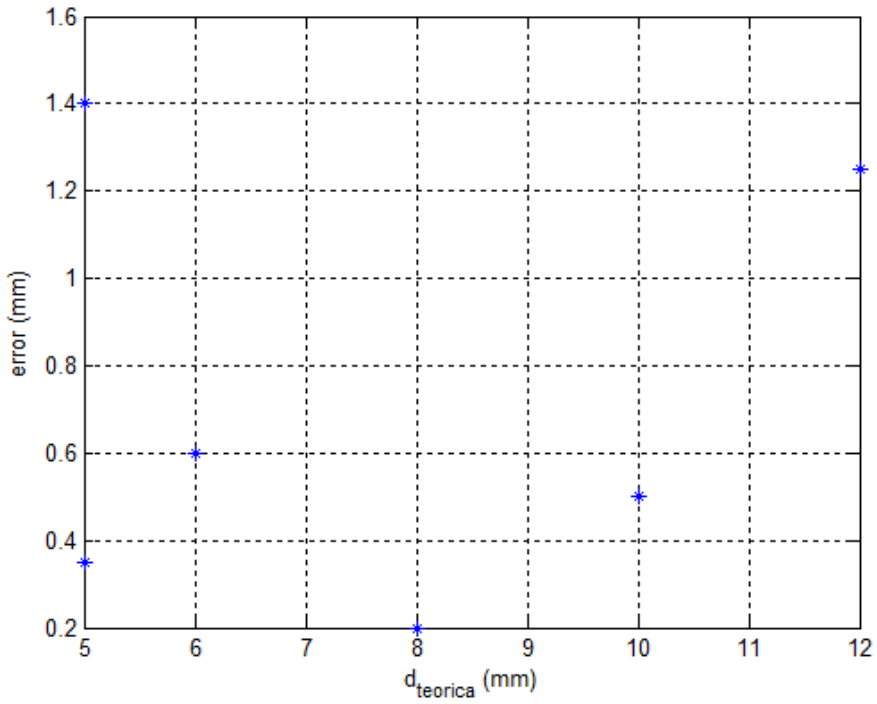

Figura 25: valores de desplazamiento reales frente a estimados para cada orden de desplazamiento

Como se puede observar en la figura, el error debido al parámetro *k1* es aleatorio y no depende de los valores ΔP_0 de entrada al sistema.

5.6 Experimentación in-vivo

Con el objetivo de verificar la validez del diseño del mini-robot, así como la calidad de la imagen proporcionada por la cámara, se ha realizado un experimento in-vivo con un modelo porcino en el Centro IACE (Instituto Andaluz de Cirugía Experimental) de Málaga. Algunas de las fotografías que se tomaron en dicho experimento se muestran en las figuras 26, 27, 28 y 29.

Gracias a este experimento se comprobó que, pese a que la fuerza de los imanes era demasiado fuerte en la caja de metacrilato (utilizada durante los experimentos in-vitro para simular la cavidad), en el animal, debido al grosor de su pared abdominal, la atracción ejercida entre los imanes era la suficiente como para que el mini-robot no se despegara de la misma, pero no tan fuerte como para que no se pudiera desplazar sin dañarla.

Figura 26: Momento de la sujeción del mini-robot mediante interacción magnética.

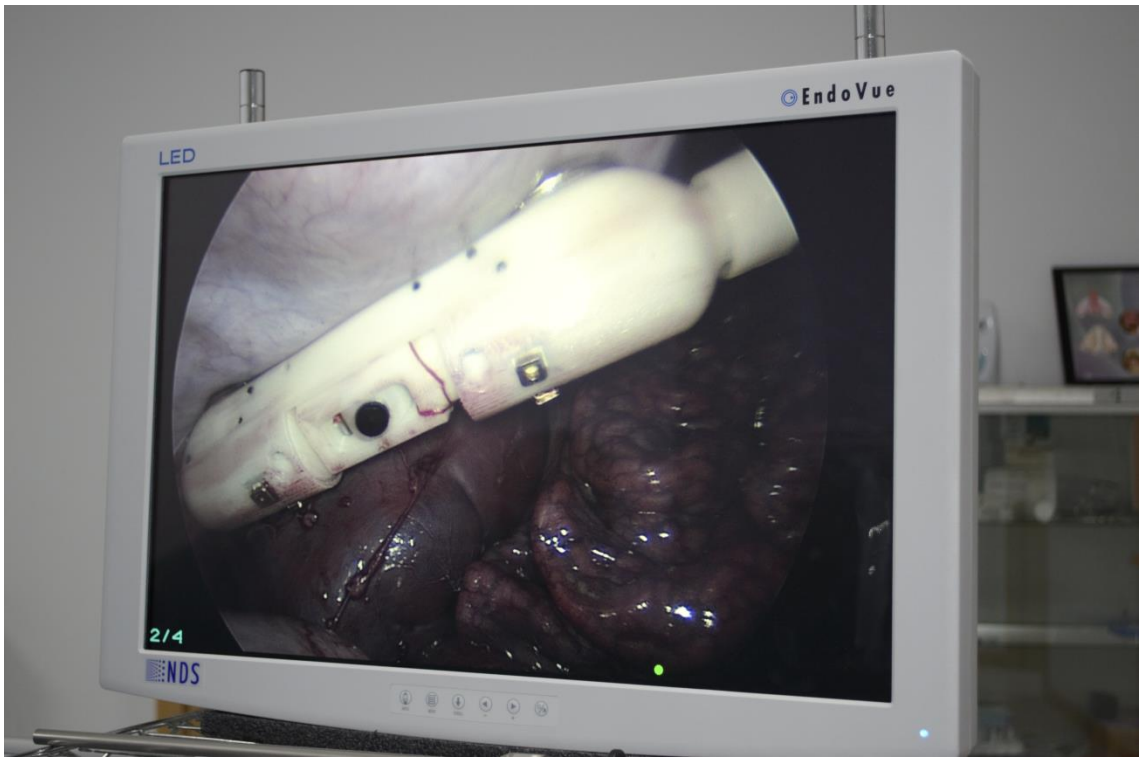

Figura 27: Mini-robot alojado en el interior de la cavidad abdominal, imagen mostrada por un endoscopio

Figura 28: Monitores para visualización de imágenes. A la derecha se muestra la imagen mostrada por el endoscopio, y a la izquierda la imagen mostrada por el mini-robot.

Como se puede observar en la figura 29, la imagen captada por la cámara tiene una calidad más que óptima para la realización de cualquier tipo de procedimiento quirúrgico enmarcado en la cirugía laparoscópica de puerto único. Además, cabe destacar que la mini cámara tiene el valor añadido de poder hacer un zoom digital de la imagen, visualizando pequeños detalles en caso de ser necesario.

Tras la finalización del experimento los cirujanos mostraron su satisfacción con el diseño desarrollado, tanto por la innovación que supone, como por la buena calidad de la imagen y por los grados de libertad del mini-robot, gracias a los cuales se puede abarcar todo el campo de trabajo sin perder la calidad de la misma.

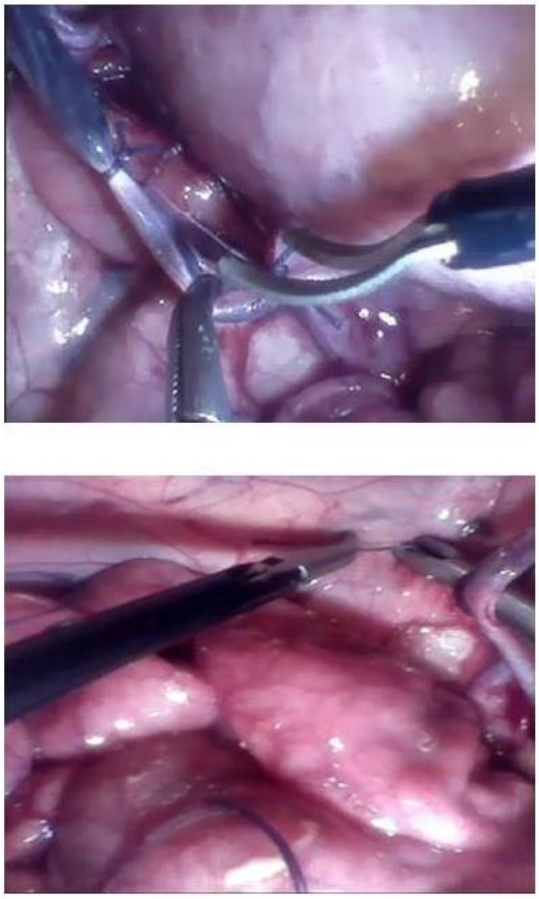

Figura 29: Imágenes captadas por la mini cámara

5.7 Conclusiones

A lo largo del capítulo se ha visto cómo se ha realizado la calibración del motor 1, la determinación de su coeficiente de transmisión de movimiento *k1*, la simulación teórica a la que se ha sometido el sistema y las pruebas experimentales que se le han hecho al mismo, con el objetivo de comprobar su validez. En el subapartado que trata la simulación teórica quedó demostrado que funcionan aquellas características del control que no pudieron ser utilizadas durante la experimentación in-vitro, mientras que en dicha experimentación se comprobó que, dentro de los rangos que permitía el mini-robot, el control diseñado funciona de forma satisfactoria, cumpliendo con las expectativas puestas en él.

Capítulo 6: Conclusiones y desarrollos futuros

6.1 Conclusiones

A lo largo de la memoria se han ido exponiendo los trabajos realizados durante el desarrollo del presente trabajo de fin de grado, demostrando que se han alcanzado todos los objetivos expuestos en el anteproyecto. En primer lugar se expuso una introducción al trabajo y una descripción del mini-robot, viendo tanto el diseño del mismo como sus mecanismos de transmisión de movimiento, además de los dispositivos de los que se sirve y del brazo robótico con el que interactúa. En segundo lugar, se presentó la arquitectura hardware diseñada, tratando tanto los microcontroladores necesarios, con sus respectivas características y motivos de su elección, como la shield EasyVR, demostrando que se cumple el requisito inicial de bajo coste. Además, en tercer lugar se describió la arquitectura de control diseñada para el control de movimientos de la cámara y su implementación en una arquitectura ROS, compuesta por diversos nodos, detallando desarrollo de cada uno de ellos. Finalmente, se realizaron una serie de experimentos para validar el correcto funcionamiento del control, entre los que se encuentran una simulación teórica y una experimentación in-vitro. Los resultados de dichos experimentos demuestran la viabilidad del control desarrollado. Además de la experimentación in-vitro, se realizó un experimento in-vivo, en la que se han podido analizar la viabilidad del diseño del mini-robot.

6.2 Líneas de desarrollo futuro

Si bien queda demostrado que el control diseñado funciona correctamente, se derivan de éste diversos problemas que se plantean como líneas de desarrollo futuro.

- El primero y más importante es mejorar el sistema de transmisión de movimientos del mini-robot, ya que la mayor parte de las dificultades que han ido surgiendo derivan de la naturaleza de los elementos elásticos escogidos para el mismo.

- El segundo problema que se ha de solventar es corregir el mini-robot para que pueda realizar correctamente su giro transversal. Cuando se solucione este problema se podrá estudiar su libertad de movimiento hacia ambos lados de la vertical.

- El tercer problema que se ha de solventar es la determinación de la altura inicial de la cámara con respecto a las vísceras cuando está sujeta a la pared abdominal, en el interior de la cavidad, ya que dicha altura es complicada de medir.

- El cuarto, que se deriva del tercero, se debe a que cuando la pared abdominal se encuentra hinchada y lista para la intervención, la superficie a la que se adherirá el mini-robot no es plana sino curva, por lo que si el robot WAM produce un desplazamiento del mini-robot, la altura a la que se encuentra el dispositivo podría aumentar o disminuir. Por lo tanto sería necesario algún sistema que compensase este efecto.

Bibliografía y referencias

[1] J.R. Romanelli, L. Mark and P.A. Omotosho, "Single port laparoscopic cholecystectomy with the TriPort system: a case report," *Surgical Innovation*, vol. 15, no. 3, pp: 223-228, 2008.

[2] I. Rivas-Blanco, B. Estebanez, M. Cuevas-Rodríguez, I. García-Morales, V.F. Muñoz. Diseño de un asistente camarógrafo para técnicas de cirugía laparoscópica de puerto único, XXXV Jornadas de Automática, Valencia, 2014

[3] Cox D.R., Zeng W., Frisella M.M., Brunt L.M., (2011) Analysis of standard multiport verus single site access for laparoscopic skills training. Surg Endosc, vol. 25.

[4] Hu T., Allen P.K., Hogle N.J., Fowler D.L., (2008) Insertable surgical imaging device with pan, tilt, zoom and lighting, 2008 IEEE International Conference on Robotics and Automation.

[5] Castro C.A., Smith S., Alqassis A., Ketterl T., Sun Y., Ross S., Rosemurgy A., Savage P.P., Gitlin R.D., (2012) MARVEL: a wireless miniature anchored robotic videoscope for expedited laparoscopy, 2012 IEEE Internatinal Conference on Robotics and Automation RiverCentre.

[6] Sun Y., Anderson A., Castro C., Lin B., Gitlin R., Ross S., Rosemurgy A., (2011) Virtually transparent epidermal imagery for laparo-endoscopic single-site surgery, 33rd Annual International Conference of the IEEE EMBS.

[7] Valdastri P, Quaglia C, Buselli E, Arezzo A, Di Lorenzo N, Morino M, Menciassi A, Dario P, (2010) A magnetic internal mechanism for precise orientation of the camera in wireless endoluminal applications, Endoscopy, vol. 42(6).

[8] Zeltser I. S., Bergs R., Fernandez R., Baker L., Eberhart R., Cadeddu J. A., (2007) Single trocar laparoscopic nephrectomy using magnetic anchoring and guidance system in the porcine model, J. Urol., vol. 178.

[9] Proyecto MARCUS [en línea]. http://www.roboticamedica.uma.es/marcus [Consulta: 8/3/2015].

[10] I. Rivas-Blanco, E. Bauzano, M. Cuevas-Rodríguez, P. del Saz-Orozco and V.F. Muñoz. Force-Position Control for a Miniature Camera Robotic System for Single-Site Surgery, IEEE/RSJ Int. Conf. On Intelligent Robots and Systems (IROS), Tokyo, Japan, 2014.

[11] ROS. Robot Operating System [en línea]. http://www.ros.org [Consulta: 8/3/2015].
[12] Lehman A.C., Dumpert J., Wood N.A., Redden L., Visty A.Q., Farritor S., Varnell B., Oleynikov D., (2009) Natural orifice cholecystectomy using a miniature robot, Surgical Endoscopy Journal, Vol. 23, pp. 260-266.
[13] Simi M., Silvestri M., Cavallotti C., Vatteroni M., Valdastri P., Menciassi A., Dario P., (2013) Magnetically Activated Stereoscopic Vision System for Laparoscopic Single-Site Surgery, in IEEE/ASME Transactions on Mechatronics, Vol. 18, No.3, pp. 1140-1151.
[14] Tognarelli S., Salerno M., Tortora G., Quaglia C., Dario P., Menciassi A., (2012) An endoluminal robotic platform for minimally invasive surgery, fourth IEEE RAS/EMBS International Conference on Biomedical Robotics and Biomechatronics, pp. 7-12.
[15] I. Rivas-Blanco, B. Estebanez, M. Cuevas-Rodriguez, E. Bauzano, and V.F. Muñoz, *Member, IEEE,* Towards a Cognitive Camera Robotic Assistant, BIOROB (2014)
[16] ULTRA MINI CMOS COLOR CAMERA. Specification. MISUMI. MO-TS0804L-P10.
[17] Pololu. Servo HD-1160A [en línea]. https://www.pololu.com/file/0J318/HD-1160A.pdf [Consulta: 6/6/2015].
[18] Arduino - Home [en línea]. http://www.arduino.cc/ [Consulta: 11/6/2015].
[19] ARDUINO, curso práctico de formación. Óscar Torrente Artero. Ed Alfaomega.
[20] Arduino – UNO [en línea]. http://www.arduino.cc/en/Main/ArduinoBoardUno [Consulta: 12/6/2015].
[21] Arduino – MEGA ADK [en línea]. http://www.arduino.cc/en/Main/ArduinoBoardMegaADK?from=Main.ArduinoBoardADK [Consulta: 12/6/2015].
[22] Congreso de Sevilla. Análisis y síntesis de la señal acústica [en línea]. http://cvc.cervantes.es/obref/congresos/sevilla/tecnologias/mesaredon_casacuberta.htm [Consulta: 12/6/2015].
[23] EasyVR 2.0 User Manual, release 3.6.7. VeeaR.
[24] EasyVR shield 2.0 [en línea] http://www.veear.eu/products/old-products/easyvr-arduino-shield/ [Consulta: 12/6/2015].
[25] Tutoriales ROS, elementos de una arquitectura basada en ROS [en línea] ftp://ftp.heanet.ie/disk1/sourceforge/r/ro/robotqbo/TUTORIALES%20ROS.pdf [consulta: 10/06/2015]
[26] C and C++ Coding Standards. ESA Board for Software Standardisation and Control (BSSC). European space agency.