BASE PARA SISTEMA DE ENTRENAMIENTO QUIRURGICO: ROBOT HIBOU

JORGE ALEJANDRO SAMBONI LASSO
CARLOS ANDRES FUENTES

Universidad del Cauca
Facultad de Ingeniería Electrónica y Telecomunicaciones
Departamento de Electrónica, Instrumentación y Control
Línea de I+D en Robótica y Control
Ingeniería en Automática Industrial

Popayán, Septiembre de 2013

BASE PARA SISTEMA DE ENTRENAMIENTO QUIRURGICO: ROBOT HIBOU

JORGE ALEJANDRO SAMBONI LASSO
CARLOS ANDRES FUENTES

Tesis presentada a la Facultad de Ingeniería Electrónica y Telecomunicaciones de la Universidad del Cauca para la obtención del Título de

Ingeniero en Automática Industrial

Director:
Phd. Oscar Andrés Vivas

Popayán, Septiembre de 2013

Hoja de Aprobación

_____

_____

_____

Director _____
Phd. Oscar Andrés Vivas

Jurado _____

Jurado _____

Fecha de sustentación: Popayán, Octubre de 2013

Agradecimientos

A nuestro director de tesis Andrés Vivas que nos brindó su colaboración en el ámbito intelectual y administrativo, para la realización de este proyecto.
A los profesores que contribuyeron con sus enseñanzas y experiencias, aportadas en el transcurso de la carrera.
A nuestros compañeros de estudio por sus consejos, apoyo y su invaluable amistad.
A los evaluadores de este proyecto que con sus críticas constructivas fomentan el crecimiento y la mejora continua.
A los miembros del Departamento de Electrónica, Instrumentación y Control por su entereza y responsabilidad en el cumplimiento de sus labores.
A la Universidad del Cauca por su contribución administrativa y económica.

Resumen

“Base para sistema de entrenamiento quirúrgico: robot HIBOU” es el proyecto que se desarrolla a continuación, y forma parte de un proyecto macro que es la construcción de un sistema de entrenamiento quirúrgico para cirugía laparoscópica. Para la consecución del proyecto se realizó la construcción de Hibou un robot porta endoscopio que es encargado de la visión al interior del paciente y es operado mediante un joystick, existe un segundo robot que es objeto de otro trabajo de grado realizado paralelamente, este es un manipulador el cual realiza el procedimiento quirúrgico al paciente, los dos robots integrados conforman el sistema completo.

Inicialmente se realizó la instalación del software necesario para lograr la manipulación del robot desde el computador y mostrar en pantalla los movimientos del mismo, el software incluye una plataforma de desarrollo y las librerías graficas necesarias para correr este tipo de programas.

Posteriormente se realizó el diseño de la tarjeta de adquisición de datos necesaria para soportar la comunicación entre el robot y el computador, la función de la tarjeta es generar un canal bidireccional de comunicación. Desde el robot hacia el computador se envían las posiciones articulares del robot, y desde el computador se envían tramas que indican la posición deseada para el robot. La tarjeta de adquisición soporta el protocolo de comunicación USB, esto debido a que es el más usado en la actualidad.

La siguiente tarea fue desarrollar el software para lograr la comunicación entre el computador y el robot, para esto se realizaron dos códigos: uno en la plataforma de desarrollo en el computador, que se encarga de captar y enviar las posiciones deseadas del robot, el otro es un código que se embebe en el micro controlador que tiene como función la recepción de las consignas desde el computador para posteriormente realizar la activación de los actuadores y la lectura de los sensores.

Enseguida se realizó la elección de la instrumentación del robot que son básicamente sensores, actuadores y etapas de potencia para su funcionamiento. Los sensores que se escogieron son potenciómetros lineales que permiten una lectura más exacta de las posiciones del robot, los actuadores escogidos son motor reductores de alto torque que permiten mover las articulaciones del robot con mayor facilidad. Para conocer con más exactitud el torque necesario que se requería para cada articulación se realizó un proceso de simulación que permite determinar el par necesario que requiere cada articulación por medio de una trayectoria determinada.

Las piezas del robot se diseñaron en un software CAD que permite darle la forma deseada a cada uno de los componentes y que respete las distancias establecidas entre las articulaciones. El software permite exportar los diseños para que posteriormente una máquina láser realice los cortes de cada una de las piezas que forman parte del robot. Una vez cortadas todas las piezas se procede a ensamblarlas mediante dispositivos mecánicos como tornillos, tuercas y partes metálicas hechas a la medida para el robot.

A continuación se hace el acople de todas las partes del sistema, para esto se fija el robot a la mesa y se procede a colocar todos los elementos que forman parte del robot, esto incluye los sensores, actuadores, tarjetas de adquisición de datos y etapas de potencia.

Finalmente se realiza la etapa de pruebas, para ello se ubica un recipiente plástico el cual se asemeja al abdomen del paciente, este recipiente incluye el trocar, por donde se introduce el órgano terminal. Seguidamente se manipula con el joystick dicho órgano del robot HIBOU, esto se logra mediante el software que calcula los ángulos deseados para cada articulación.

Abstract.

“Base para sistema de entrenamiento quirúrgico: robot HIBOU” is the project that develops below and is part of a macro project that is the construction of a surgical training system for laparoscopic surgery. To achieve the project was carried Hibou building a robot endoscope holder is responsible for the vision into the patient and operated by a joystick, there is a second robot that is the subject of another degree work done in parallel, this is a handle which makes the surgical procedure to the patient the two robots make up the complete system integrated.

Initially performed software installation necessary to achieve manipulation robot from the computer and display the movements of the same, the software includes a development platform and libraries needed to run graphics programs such.

Subsequently performed designing the data acquisition board needed to support communications between the robot and the computer, card function is to generate a bidirectional communication channel. Since the robot to the computer are sent in the robot joint positions, and from computer frames are sent indicating the desired position for the robot. The acquisition card supports USB communication protocol, this because it is the most used at present.

The next task was to develop the software to achieve communication between the computer and the robot, for this were two codes: one on the development platform on the computer, which is responsible for capturing and sending the desired positions of the robot, the other is a code that is embedded in the microcontroller whose function is the receipt of the instructions from the computer and subsequently to activate the actuators and sensors reading.

This was followed by the choice of instrumentation sensors are basically robot, actuators and power amplifiers for operation. Sensors chosen are linear potentiometers which allow a more accurate reading of the position of the robot, chosen actuators are high torque engine reducers that allow the robot joints move more easily. For more details of the necessary torque for each joint was required was a simulation process that determines the necessary torque required for each joint using a given path.

Robot parts are designed on a CAD software that allows the desired shape to each of the components that comply with preset distances between joints. The software allows layouts to export subsequently perform a laser machine cuts each of the pieces that form part of the robot. Once cut all the pieces necessary to assemble by mechanical devices such as screws, nuts and metal parts made to order for the robot.

The following is the coupling of all parts of the system, for this robot is fixed to the table and proceeds to place all items that are part of the robot, for this robot is fixed to the table and proceed to place all items that are part of the robot.

We carried out the testing stage, for it is located a plastic container that resembles the patient's abdomen, the container includes the trocar, through which you enter the terminal organ. Then the joystick is manipulated organ robot said HIBOU, this is achieved by software that calculates the desired angles for each joint.

Tabla de contenido

Lista de figuras

Lista de tablas

1. Robótica y su aplicación en el campo médico.

1.1. Conceptos Generales.

La robótica es un campo de la ciencia que ha tomado mucha fuerza en los últimos años, esto se debe a que un robot es un sistema electromecánico programado capaz de realizar múltiples tareas en tiempos cortos y con un grado de error muy bajo.

Debido a las grandes ventajas de la utilización de robots para tareas complejas y de alta precisión existen robots dedicados a participar como asistentes en procedimientos quirúrgicos, entre estos procedimientos se destacan la tele-cirugía y la cirugía mínimamente invasiva, cirugía cardiaca, gastrointestinal, pediátrica y neurocirugía [1].

A continuación se presentan las principales definiciones en el campo de la robótica [2].

Articulación: Mecanismo que une dos cuerpos sucesivos, accionado por un motor. Las articulaciones son principalmente rotoides (de giro) y prismáticas (de desplazamiento lineal).

Articulación rotoide: El movimiento de rotación se realiza alrededor de un eje común entre dos cuerpos.

Articulación prismática: El movimiento de traslación se realiza a lo largo del eje común entre dos cuerpos. La situación relativa entre los dos cuerpos está dada por la distancia a lo largo de este eje.

Grado de libertad: Define cada movimiento independiente del robot. Para situar un objeto en un espacio tridimensional son necesarios tres grados de libertad, uno por cada dimensión.

Espacio articular: Es el espacio en el cual se representa la situación de todos los cuerpos del robot; corresponde al lenguaje que maneja el mecanismo en si mismo (movimientos rotacionales y prismáticos).

Espacio operacional: Es aquel donde se representa la situación del órgano terminal. Para definir esta situación se utilizan las coordenadas cartesianas en tres dimensiones.

1.2. Modelo geométrico de un robot

El modelo geométrico de un robot me permite representar las características físicas en un sistema de coordenadas referenciado a cada articulación que lo compone [2].

Modelo geométrico directo: Este modelo expresa la situación del órgano terminal, es decir que si se conocen las posiciones articulares de un robot, el modelo permite conocer la posición cartesiana y la orientación del órgano terminal. Esto permite al programador saber con exactitud donde se encuentra el órgano terminal del robot en cada momento, sin hacer uso de un sistema de visión.

Modelo geométrico inverso: Este modelo provee todas las soluciones posibles del cálculo de las coordenadas articulares, correspondientes a una situación cartesiana determinada. Es decir para una posición y orientación deseadas del órgano terminal, el modelo geométrico inverso entrega todas las posibles soluciones de las posiciones articulares con el fin de alcanzar esa situación deseada.

1.3. Modelo dinámico de un robot.

El modelo dinámico de un robot se define como la relación entre las fuerzas aplicadas a los actuadores y las posiciones, velocidades y aceleraciones articulares [2].

El modelo dinámico es utilizado para:

- El dimensionamiento de los actuadores, esto debido a que en simulación se pueden obtener los pares y fuerzas necesarios para generar movimiento en cada una de las articulaciones que componen el robot.
- Para el control del robot.

Modelo dinámico directo: Describe las aceleraciones articulares en términos de las posiciones, velocidades y pares.

Modelo dinámico inverso: Provee los pares articulares y fuerzas en términos de las posiciones, velocidades y aceleraciones articulares.

1.4. Robótica aplicada a la medicina.

La robótica puede ser de gran utilidad en la medicina ya que es una herramienta muy apropiada en el momento de realizar cirugías complejas o donde al ojo humano le es difícil acceder.

Algunos métodos sirven para la captación de imágenes, que permiten reconstruir modelos virtuales de los órganos que se necesitan examinar, para luego manipularlos y mejorar un diagnóstico o simular operaciones quirúrgicas.

Los grandes avances en informática y comunicaciones no solo se ven reflejados en la industria, también vemos un gran desarrollo en la medicina y de forma más particular en la cirugía, que ha dado como resultado un acelerado proceso de informatización de todas las áreas de la medicina. Así, los robots se están convirtiendo en un instrumento indispensable en la medicina, ya que son herramientas que han ayudado a mejorar múltiples procesos quirúrgicos [3].

1.5. Cirugía laparoscópica.

La laparoscopia es un procedimiento mediante el cual se separa la pared abdominal de los órganos, elevando el abdomen para que el cirujano tenga acceso al interior del paciente y pueda manipular los instrumentos con relativa facilidad. La elevación se realiza mediante insuflación empleando CO_2, dado que suprime la combustión y el cuerpo lo absorbe con más facilidad en comparación a la insuflación con aire [4]. La aguja de Veress es el instrumento que se utiliza para puncionar la cavidad abdominal y e insuflar el abdomen, está constituida por una camisa metálica de punta aguda, en su interior posee un vástago retráctil y su longitud esta entre 0.08 y 0.2m. Después de obtener acceso al abdomen se realizan incisiones de 0.005 a 0.01m para introducir los trocares que son instrumentos con un punzón cortante, revestidos por una funda o cánula, y su composición es acero quirúrgico [5]. A continuación se muestra gráficamente como se realiza el procedimiento.

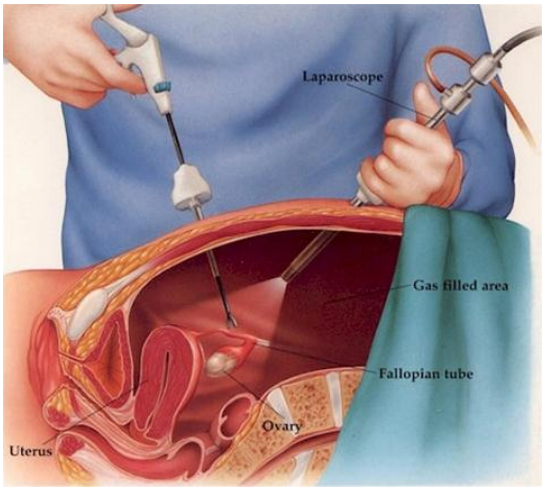


Figura 1. Cirugia laparoscopica.

Actualmente los procedimientos quirúrgicos laparoscópicos más comunes son:

Colecistectomía: Intervención que se realiza para la extracción de la vesícula biliar que presente inflamación o se encuentre obstruida por cálculos biliares.

Apendicetomía: Extirpación de un apéndice inflamado o infectado (apendicitis). Se hace una incisión pequeña en el cuadrante inferior derecho del abdomen y se extirpa el apéndice [6].

Histerectomía: Cirugía para extirpar el útero (matriz), el cirujano realiza incisión por la cual corta las trompas de Falopio y separa el útero en su unión con el cuello [7].

Gastrectomía: Extirpación total o parcial del estómago, en caso de presentarse problemas gástricos crónicos como ulceras o cáncer [8].

Entre las principales ventajas de la cirugía laparoscópica se encuentran:

- Mejores resultados estéticos.
- Disminución de riesgo a infecciones.
- Reducción en la estadía hospitalaria; menor tiempo de recuperación, reducción en la medicación.

1.6. Robots quirúrgicos.

1.6.1. Robot DAVINCI

Es comercializado por la empresa *Intuitive Surgical* [9], y tiene licencia de la FDA (*Food and Drug Administration*) de los Estados Unidos para ser empleado en pacientes humanos desde 1999 [10].

El sistema quirúrgico Da Vinci permite a los cirujanos realizar operaciones delicadas y complejas a través de unas incisiones pequeñas con una mayor visión, precisión, agilidad y control. El sistema quirúrgico Da Vinci se compone de varios elementos importantes incluyendo: una consola de diseño ergonómico, donde el cirujano se sienta durante el procedimiento, cuatro brazos robóticos interactivos, un sistema de alta definición de la visión en 3D [9].

A continuación se muestra el sistema quirúrgico DAVINCI.

Figura 2. Robot Quirúrgico DA VINCI [9].

1.6.2. Robot BLACK FALCON.

Desarrollado a manera experimental por Madhani en 1998 en el MIT (*Massachusetts Institute of Technology*) [11].

Este robot manipulador se caracteriza por la eficiencia en la realización de suturas complejas. Tiene una interfaz háptica Phantom modificada, que le permite controlar los movimientos del robot y retroalimentar al cirujano los esfuerzos entre el instrumento y los tejidos.

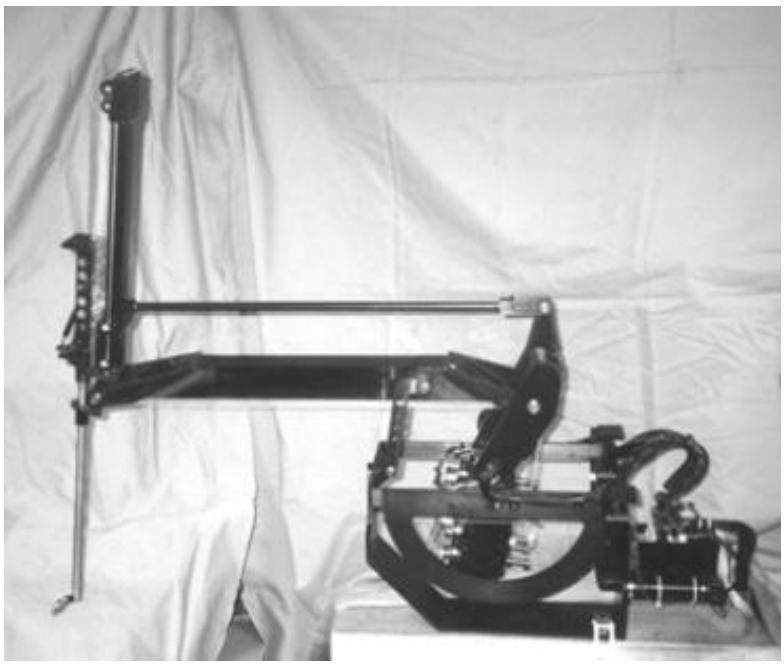


Figura 3. Robot quirúrgico Black Falcón.

1.6.3. Robot ZEUS.

Zeus es una plataforma modular (ver Figura 4) que contiene al robot AESOP para operar la cámara, una mesa de cirugía con brazos independientes que emplean instrumentos miniaturizados de 5 grados de libertad y una consola de mando donde se ubica el cirujano.

El único límite de separación entre los componentes de Zeus es la latencia del video, por tanto puede ser usado en telecirugía a grandes distancias [12].

Figura 4. Robot quirúrgico ZEUS.

1.7. Entrenadores quirúrgicos.

El entrenamiento basado en simulación busca entrenar a los médicos para crear condiciones en las cuales cometer errores no sea un factor perjudicial o peligroso para los pacientes, esto proporciona la posibilidad de que los médicos residentes obtengan destreza y reciban realimentación constructiva del especialista a cargo [13].

El entrenamiento basado en simulación presenta las siguientes ventajas:

- Reducción en el tiempo de aprendizaje, esto se debe a que no es necesario esperar a que se presente un paciente, al que se le deba practicar un determinado tipo de cirugía.
- Por medio de un simulador los médicos experimentados pueden realizar procedimientos que no hayan realizado anteriormente.
- No existe contacto directo con el paciente, se realiza una simulación de cómo afrontar un determinado procedimiento.
- Proporciona a los estudiantes y profesionales una oportunidad de aprender de sus propios errores y generar realimentación.

1.7.1. Simuladores de cirugía laparoscópica.

Un simulador de cirugía laparoscopia es un dispositivo que permite reproducir actividades propias de este procedimiento, su objetivo principal es propiciar destreza en la manipulación de los instrumentos quirúrgicos y la familiarización con el procedimiento a seguir [14].

A continuación se muestran simuladores quirúrgicos para entrenamiento en cirugía laparoscópica.

1.7.2. LAP MENTOR.

El LAP Mentor es un simulador quirúrgico multi-disciplinario, permite la práctica de uno o varios alumnos. El sistema ofrece oportunidades de capacitación a los cirujanos nuevos y experimentados, desde el perfeccionamiento de habilidades básicas de laparoscopia hasta realizar procedimientos quirúrgicos laparoscópicos completos [15].

Figura 5. Simulador quirúrgico Lap Mentor.

1.7.3. Simulador para entrenamiento en cirugía avanzada.

Es un simulador de cirugía laparoscópica desarrollado en la Universidad Politécnica de Valencia, España, permite familiarizar a los cirujanos tanto con el manejo de las herramientas como en las diferentes técnicas de intervención en este tipo de cirugías. Este simulador posee dos ventajas frente a los sistemas de entrenamiento tradicionales: permite repetir una técnica tantas veces como

sea necesario hasta su correcto aprendizaje y permite simular la intervención sobre la reconstrucción 3D de los órganos del paciente previamente a su intervención quirúrgica real [13].

1.7.4. Software manipulador del robot porta endoscopio (Hibou) para Cirugía laparoscópica.

Es un software de simulación desarrollado en la Universidad del Cauca, en la Facultad de Ingeniería Electrónica y Telecomunicaciones. En esencia es una aplicación que muestra al robot HIBOU en un ambiente tridimensional en una sala de operaciones. En este ambiente es posible manipular el robot mediante un dispositivo de mando (joystick), teniendo en cuenta sus características dinámicas. Adicionalmente se muestran en dos pantallas separadas la cámara en el abdomen del paciente y el robot en la sala de cirugía [16].

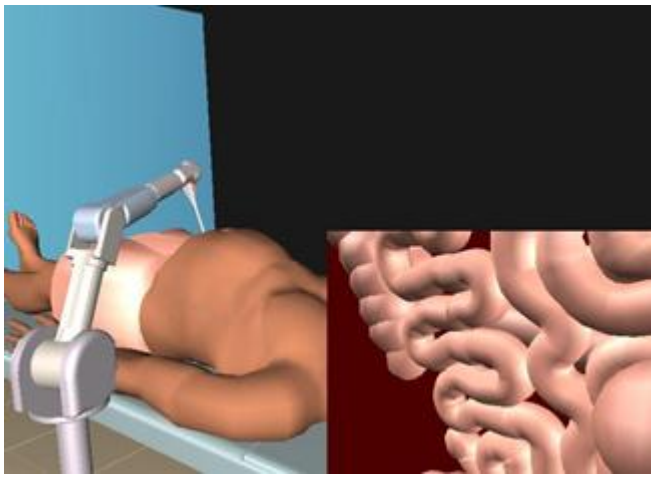


Figura 6. Simulador quirúrgico del robot Hibou.

1.7.5. Robot porta endoscopio Hibou.

Es un robot porta endoscopio desarrollado en la Universidad Del Cauca, diseñado con 7 grados de libertad, matemáticamente modelado a nivel geométrico y dinámico, y simulado en un ambiente tridimensional. El robot sigue con eficiencia las trayectorias que se necesitan en procedimientos laparoscópicos, manteniendo un punto fijo en el espacio para su movilidad (trocar), este sirve como entrada a la cavidad abdominal y como punto fijo de rotación del efector final que se introduce al paciente [3].

Figura 7. Robot porta endoscopio Hibou.

1.8. Robots porta endoscopios comerciales.

1.8.1. Endoassist.

Sistema creado por Amstrong Healthcare Ltd. en Inglaterra. El Endoassist es un dispositivo independiente que sostiene el laparoscopio, controlado por el cirujano en lugar del asistente. Se activa mediante un pedal y es controlado por los movimientos de la cabeza del cirujano. Está diseñado para facilitar la cirugía laparoscópica mediante la mejora de la imagen sobre la que el cirujano trabaja.

Este robot permite fácilmente insertar el sistema de video dentro de la cavidad pélvica-abdominal; la naturaleza de su interfaz humano-computador permite lograr el dominio del robot en poco tiempo [17]. El Endoassist se muestra en la siguiente figura.

Figura 8. Robot Endoassist.

1.8.2. Aesop.

Robot creado por Computer Motion Incorporated en Santa Bárbara, California. Este sistema está compuesto por un brazo mecánico acoplado a un ordenador que reconoce comandos de voz específicos, que se captan a través de un micrófono. La computadora registra estas órdenes verbales y las transforma en movimientos que guían la cámara hacia la posición deseada por el cirujano. Lo particular del sistema es que solo obedece a las voces de los cirujanos registrados en la base de datos de la computadora, con lo cual se evita problemas de interferencia con otras voces. Este robot cuenta con una serie de sistemas de seguridad que protegen al paciente. Al inicio de la cirugía, el cirujano fija un margen inferior límite para el brazo robótico de Aesop. Esto previene al brazo robótico de herir al paciente durante la operación, de la misma manera evita comandos inadvertidos por parte del cirujano al conducir el telescopio a través de estructuras anatómicas, como el hígado [18].

Figura 9. Robot Aesop.

1.8.3. Robot LER.

Light Endoscopio Robot, robot creado por TIMC, Grenoble, Francia. Brazo mecánico articulado unido a un computador, el cual se encarga de interpretar órdenes verbales simples que el robot convierte en movimientos de la cámara laparoscópica. Este sistema consta de tres partes: una base móvil, un laparoscopio de flexión, y un manipulador externo. La particularidad de este robot es su reducido tamaño, por tanto es muy útil ya que se puede acoplar en cualquier quirófano sin necesitar de grandes espacios [19].

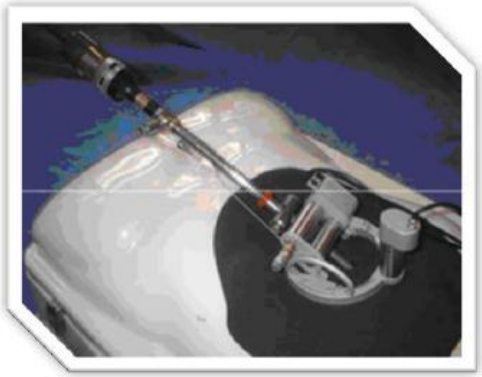


Figura 10. Robot LER.

1.8.4. Robot Lapman.

Dieñado por *Medsys,* Bélgica. Es un robot que puede ser fácilmente trasladado desde y hacia la mesa de operaciones. El robot Lapman consta de una base móvil para un fácil manejo del sistema y un brazo que ejecuta los movimientos reales, mientras sostiene el endoscopio. Su eje tiene un laparoscopio que se mueve mediante órdenes a distancia, estas órdenes son enviadas a través del mando de control RF LapStick®. Posee un sistema que se ubica sobre el mango del instrumento quirúrgico, el cual es activado por el cirujano, el mecanismo asegura la estabilidad de la imagen, con lo cual el control de los movimientos de la cámara recae únicamente en el cirujano y libera al asistente de cirugía para ampliar sus funciones de apoyo [20].

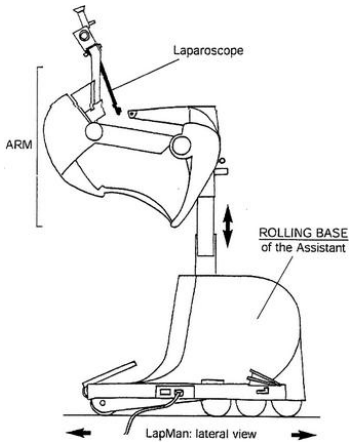


Figura 11. Robot Lapman.

1.8.5. Tonatiuh

Diseñado en el Laboratorio de Bioelectrónica del Centro de Investigación y Estudios Avanzados del Instituto Politécnico Nacional, México**.** Es el brazo robótico Tonatiuh, el primero en su tipo en América Latina, diseñado por el doctor en ingeniería eléctrica Arturo Minor**.** El robot está constituido por un

brazo mecánico de cinco grados de libertad con tres articulaciones y un efector final. El grado de libertad rotatorio ubicado en la base se usa para la ubicación del manipulador sobre el puerto de inserción del laparoscopio y las articulaciones restantes se utilizan para la navegación. Este sistema puede ser operado por el cirujano mediante: movimientos cefálicos a través de luz infrarroja, reconocimiento de voz o por un control manual físico [21].

Figura 4. Robot Tonatiuh.

2. Construcción del robot Hibou.

Para realizar la construcción del robot Hibou se realizó inicialmente la construcción de todos los componentes del robot mediante un software de simulación, esto se hace para tener un diseño preliminar antes de su construcción real. Enseguida se realiza la escogencia del material sobre el cual se va a soportar toda la cadena cinemática, posteriormente se realiza el corte de todas las piezas que lo componen y se realiza el ensamble de estas mediante dispositivos mecánicos como tornillos, tuercas y uniones en metal hechas a la medida. El siguiente paso es determinar el torque necesario que deben generar los actuadores para levantar el robot sin ningún inconveniente, y se escoge la instrumentación necesaria para generar la potencia que permita activar los actuadores, además de los sensores que determinan la posición de las articulaciones del robot y la tarjeta de adquisición de datos que fue diseñada, En seguida se procede a realizar el ensamble de todos los componentes para conformar el robot. Finalmente se ubica en la mesa el recipiente plástico con su respectivo trocar.

2.1. Diseño del robot mediante software CAD.

El diseño del robot se realizó en el software Solid Edge y se tuvo en cuenta que el cuerpo de las dos primeras articulaciones debía tener un espesor de 1 cm, y el cuerpo de la articulación 3 y las articulaciones pasivas debía tener un espesor de 4mm, esto debido a que el grosor del material utilizado para la construcción tiene estas dimensiones, y que respetara las distancias que previamente se habían establecido entre las articulaciones.

La construcción del robot mediante simulación se realizó partiendo desde la primera articulación hasta el efector final, para la articulación 1 se diseñaron tres piezas que ensambladas permiten conectar el motor 1 al cuerpo del robot (ver figura 13).

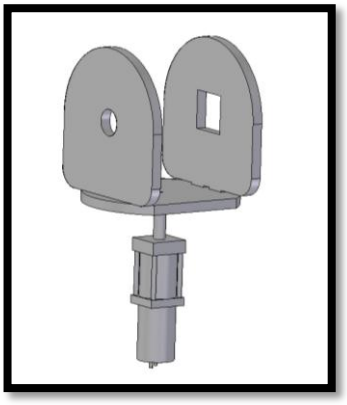


Figura 5. Articulación 1 del robot diseñado

Para soportar el movimiento de la articulación 2 se diseñaron 14 piezas que ensambladas permiten unir la estructura del robot con el actuador de la articulación 2, estas piezas deben soportar casi la totalidad del peso del robot en movimiento, por este motivo se conectaron 5 piezas al eje del motor, las piezas restantes forman el cuerpo de la articulación 2 (ver figura 14).

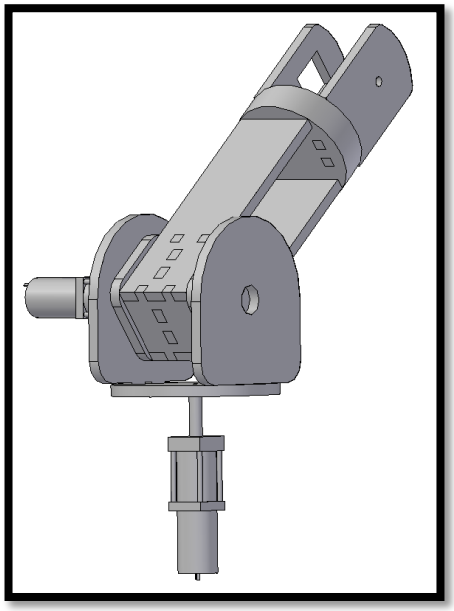


Figura 6. Piezas de la articulación 2.

El cuerpo de la articulación 3 es la siguiente parte que se diseña, esta parte incluye un orificio para ensamblar un dispositivo mecánico para la implementación de la primera articulación pasiva, este dispositivo es un rodamiento o balinera.

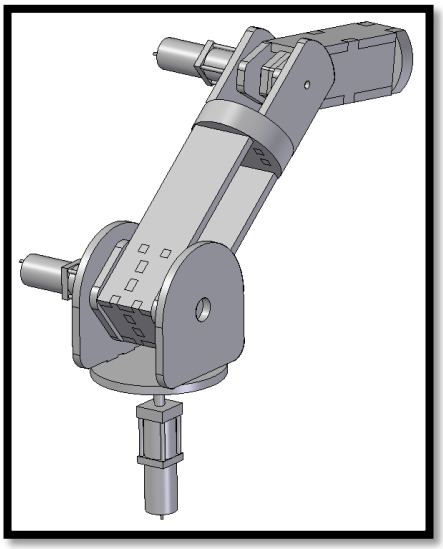


Figura 7. Piezas de la articulación 3.

Finalmente se diseña la etapa que incluye el porta endoscopio, esta parte incluye también dos balineras para la segunda articulación pasiva. Con esto queda diseñada la totalidad del robot.

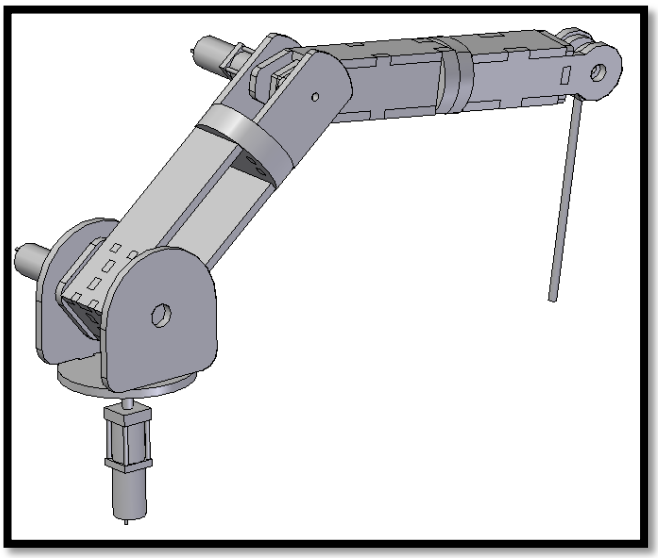


Figura 8. Estructura del robot en el software CAD.

Los archivos que genera Solid Edge de cada pieza son guardados en una carpeta, y posteriormente se realiza un cambio de formato para poder ser exportados a la máquina que va a realizar los cortes.

2.2. Material seleccionado para la construcción.

El robot está conformado básicamente por un material que soporte el peso de toda la estructura del robot, que permita cortes finos mediante una maquina a laser, y que sea fácil de encontrar en el mercado. Teniendo en cuenta esto se determinó que el material ideal para la construcción de la base de sistema de entrenamiento quirúrgico es la madera, más específicamente dos láminas de triplex de 1 cm y de 4mm de espesor.

2.2.1. Características del material escogido.

- Presenta resistencia a la compresión, si es sometida a una fuerza las fibras que la componen no tienden a doblarse con facilidad.
- Tiene una estructura homogénea, esto es debido a que la composición de las fibras que la componen es uniforme.
- Presenta alta resistencia a la torsión, si se fija una pieza al extremo no presenta deformación en esta parte.

2.3. Corte de las piezas del robot y ensamble.

Para realizar el corte de las piezas del robot se introducen los archivos generados mediante el software CAD al controlador de la máquina cortadora, una vez introducidos los bocetos de corte se procede a colocar la lámina de madera y se asegura para que no se mueva durante el procedimiento.

El corte de la totalidad de las piezas tiene un tiempo de duración de aproximadamente 115 minutos, una vez cortadas se procede a realizar el proceso de estucado y pintado para darle una mejor presentación a la estructura del robot.

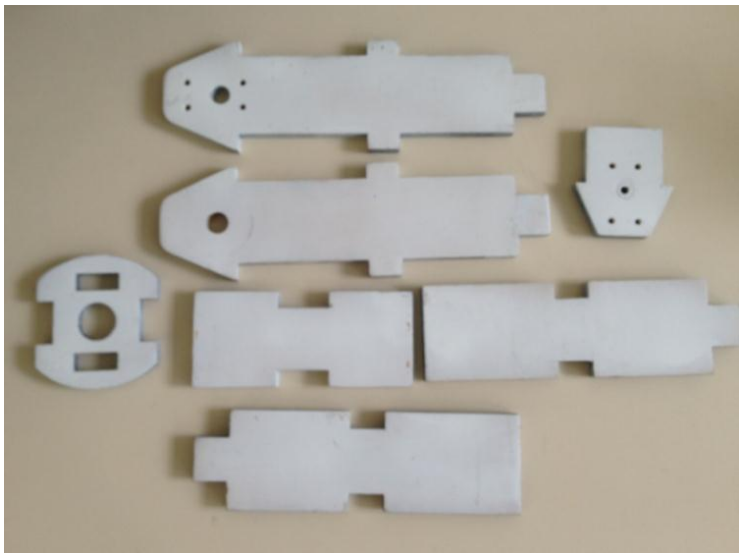
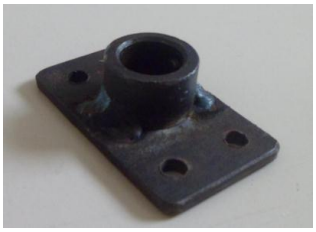
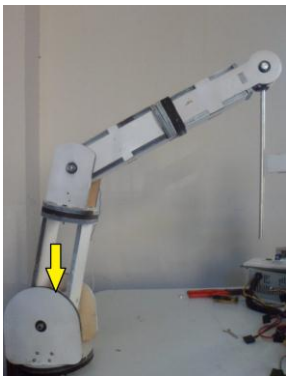
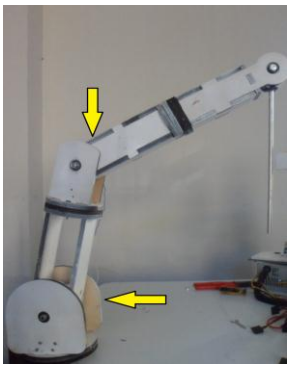
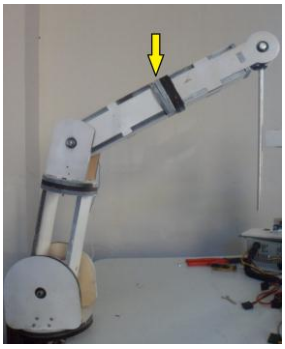


Figura 17. Piezas cortadas y pintadas.

Para realizar el ensamble de todas las piezas se requiere de una serie de dispositivos mecánicos que permiten acoplar todas las piezas, estos dispositivos y su descripción se muestran a continuación en la tabla 1.

Dispositivo	Imagen de dispositivo	Ubicación en el robot
Unión: Es un dispositivo en forma de L que sirve para acoplar dos piezas para que unidas formen un ángulo de 90 grados, el robot presenta 6 de estos elementos.		

Acople motor: Este dispositivo permite conectar el motor con el resto de la articulación, el sistema presenta 2 de estos elementos.		
Soporte motor 2 y 3: Este dispositivo fija el motor reductor a la pieza que va ubicada en cada una de las articulaciones, en el robot se encuentran 4 de estos elementos.		
Soporte motor 1: Este es un dispositivo mecánico hecho a la medida que conecta el primer motor a la articulación 1, este dispositivo sirve además como eje para mover el potenciómetro 1.		
Unión para articulación tres y cuatro: Es un dispositivo que permite la unión que permite acoplar la articulación tres y cuatro, en su parte trasera presenta una balinera.		

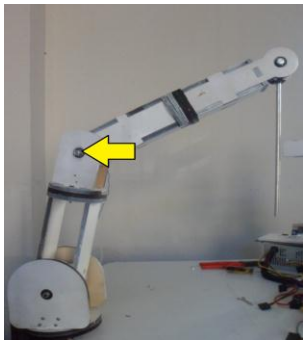
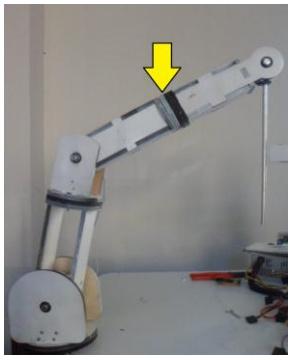
Eje 2 para sensor 2: Este eje permite transmitir el movimiento del motor 2 al potenciómetro 2.		
Eje 3 para sensor 3: Este eje permite transmitir el movimiento del motor 3 al potenciómetro 3.		
Eje 4 para sensor 4: Este eje permite transmitir el movimiento de la articulación pasiva 4 al potenciómetro 4.		
Eje 5 para sensor 5: Este eje permite transmitir el movimiento de la articulación pasiva 5 al potenciómetro 5.		

Tabla 1. Dispositivos mecánicos.

2.4. Escogencia de los actuadores.

En la elección de cada motor encontramos dos parámetros importantes, la velocidad y el torque. Debido a que nuestro sistema debe ser lento y con movimientos suaves, la velocidad debe ser baja por lo cual se pensó en el uso de motorreductores. Teniendo en cuenta que estos poseen un alto torque se opta por el uso de estos dispositivos en las primeras 3 articulaciones las cuales son motorizadas y requieren un alto torque.

2.4.1. Cálculo de torques.

El cálculo de torques para cada articulación se realiza mediante un controlador CTC cartesiano, con una consigna circular ubicando su centro en (0.4, 0) metros, estas coordenadas están en los ejes X y Y respectivamente. Este tipo de controlador involucra el modelo dinámico, el cual está dividido en dos, modelo dinámico directo (MDD) y modelo dinámico inverso (MDI) [2].

Para lograr el dimensionamiento de los actuadores, se deben identificar los parámetros inerciales pertenecientes al modelo dinámico de nuestro robot, por lo cual se hace uso del programa CAD Solid Edge. Esta herramienta calcula el valor de parámetros necesarios en el modelo dinámico como lo son los momentos de inercia: primer y segundo momento de inercia o de área. A continuación se muestra como se obtuvieron estos parámetros.

Teniendo el diseño CAD en Solid Edge de cada una de las piezas de nuestro robot adicionamos a cada una su respectivo material, seguidamente se procede a realizar el ensamble del brazo robótico en un conjunto (ver figura 18).

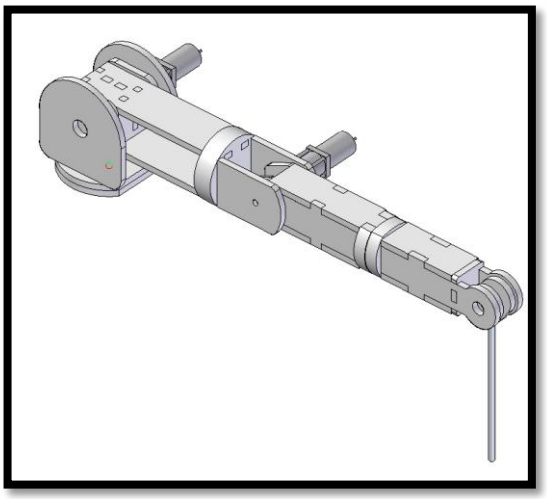


Figura 18. Brazo robótico ensamblado.

Una vez terminando el conjunto se procede en Solid Edge a ubicar los ejes de giro de cada articulación, según su movimiento, el eje z será el eje de giro. Por medio de la opción sistema de coordenadas (ver figura 19), orientar mediante la opción “Teclado” (ver figura 20) se puede trasladar y rotar el nuevo sistema de coordenadas en los ejes X, Y y Z relativo a los ejes globales que se crean al poner la primera pieza en el conjunto o a un sistema de coordenadas creado previamente (ver figura 21).

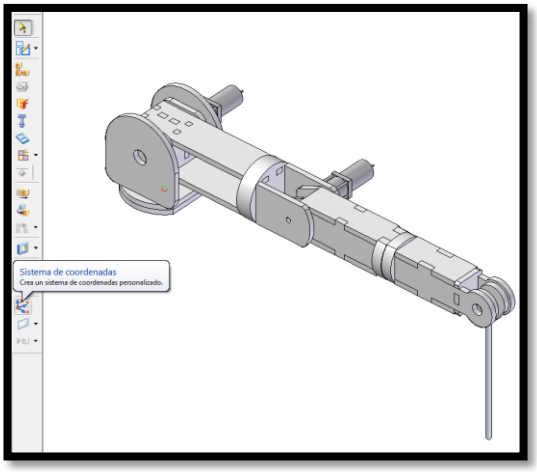


Figura 19. Herramienta sistema de coordenadas.

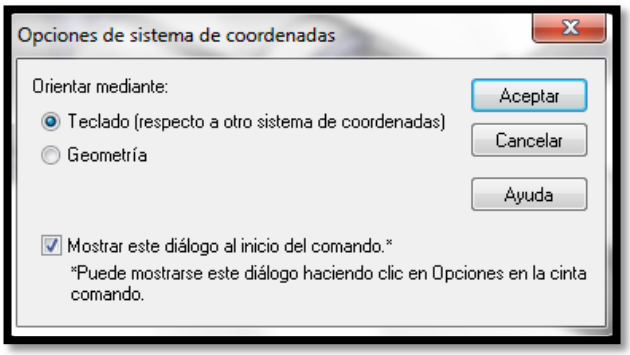


Figura 20. Orientación mediante teclado.

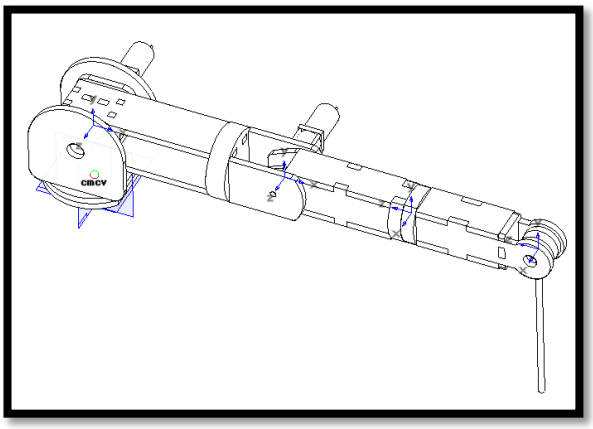


Figura 21. Robot Hibou con sus ejes globales.

Una vez ubicados los ejes de giro, por medio de la opción Verificar >> Propiedades Físicas (ver figura 22), obtenemos los datos para calcular el primer momento de inercia y los valores del segundo momento de inercia.

En seguida buscamos la opción verificar >> propiedades físicas, esta opción permite calcular la masa, centro de volumen y finalmente la matriz de momentos de inercia de la masa (segundo momento de inercia). (Ver figura 23)

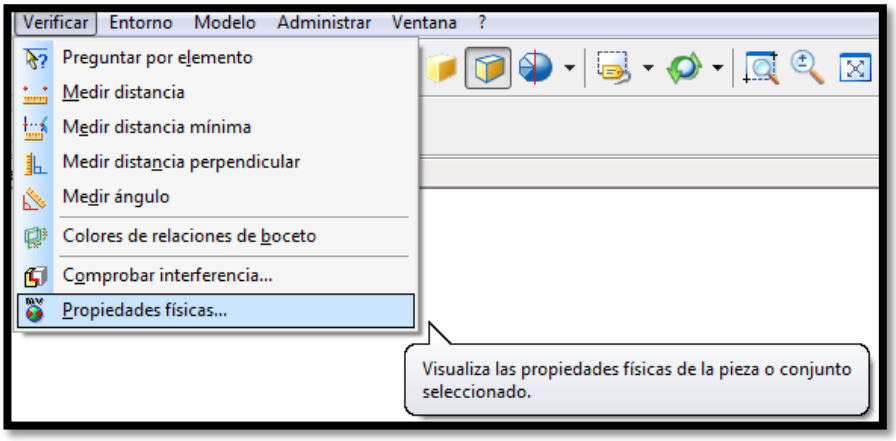


Figura 22. Herramienta verificar y propiedades físicas en el software.

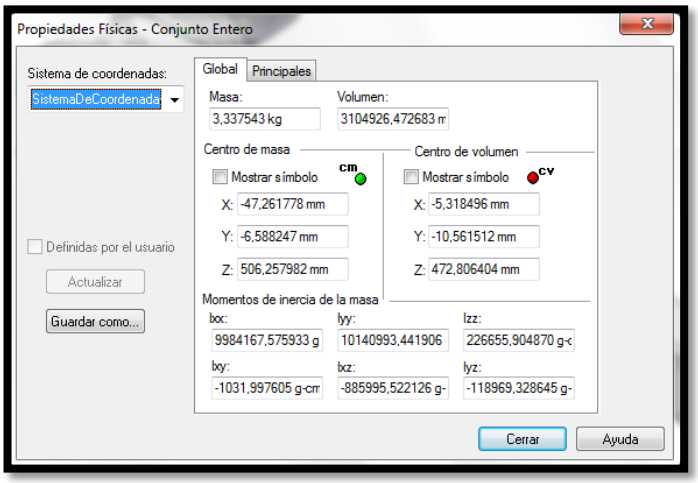


Figura 23. Propiedades físicas del conjunto.

ARTICULACIONES	MASA DE CADA ARTICULACION
Articulación1	M1= 3.337543
Articulacion2	M2= 2.076420
Articulacion3	M3= 0.609924
Arrticulacion4	M4= 0.319843
Articulacion5	M5= 0.062715
Articulacion6	M6= 0.034950

Tabla 2. Masas por articulación obtenidas en Solid Edge.

Para calcular el primer momento de inercia multiplicamos la masa por la distancia al centro de masa en cada una de sus coordenadas X, Y y Z, obteniendo el vector del primer momento de inercia, una vez hecho esto obtenemos los siguientes resultados.

Articulación1	MX1=0.646622315981774	MY1=0.18035522641454	MZ1=0.295078027342879
Articulacion2	MX2=0.64736698749228	MY2=-0.00508320489804	MZ2=-0.08508030035988
Articulacion3	MX3=0.128160309776352	MY3=-0.005083714710228	MZ3=-0.000808470120024
Arrticulacion4	MX4=-0.000064354330658	MY4=-0.005083714498258	MZ4=-0.035829789980365
Articulacion5	MX5=0	MY5=-0.00508372606512	MZ5=-0.00001261863429
Articulacion6	MX6=0	MY6=-0.00503853560955	MZ6=-0.0000070321497
Articulacion7	MX7=-0.0000070321497	MY7=-0.00503853560955	MZ7=0

Tabla 3. Primer momento de inercia por articulación.

Articulación1	XX1=0.0506826332051	XY1=0.0258068455096	XZ1=0.0579340574515
	YY1=0.2942355062036	YZ1=0.0169089597824	ZZ1=0.2860481419585
Articulacion2	XX2=0.0108054856510	XY2=-0.0035584202268	XZ2=-0.0258067393355
	YY2=0.2732103461651	YZ2=0.0000323100817	ZZ2=0.2658782190951
Arrticulacion3	XX3=0.0014320689369	XY3=-0.0035586011664	XZ3=-0.0019121398544
	YY3=0.1689594972974	YZ3=0.0000323894032	ZZ3=0.1698855679591
Articulacion4	XX4=0.0067417485355	XY4=0.0000010228757	XZ4=0.0000071308887
	YY4=0.0058230043082	YZ4=0.0010167431904	ZZ4=0.0011788284671
Articulacion5	XX5=0.0000064382387	XY5=0	XZ5=0
	YY5=0.0000106452053	YZ5=0.00000102287579	ZZ5=0.0009272569600
Articulacion6	XX6=0.0009152125280	XY6=0	XZ6=0
	YY6=0.0000005356860	YZ6=0.0000010137807	ZZ6=0.0009150633006
Articulacion7	XX7=0.0009150633006	XY7=0.0000010137807	XZ7=0
	YY7=0.0000005356860	YZ7=0	ZZ7=0.0009152125280

Tabla 4. Segundo momento de inercia por articulación.

Con los parámetros inerciales obtenidos se procede a introducirlos en el control CTC realizado en la tesis de pregrado de la universidad del cauca “DISEÑO Y SIMULACIÓN EN 3D DE UN ROBOT PORTA ENDOSCOPIO PARA CIRUGÍA LAPAROSCÓPICA” [3], que se muestra en la figura 24. Seguidamente se simula en MATLAB dicho control obteniendo de las gráficas el error cartesiano y el torque máximo, se varían las ganancias proporcionales y derivativas, hasta lograr un error de 1.5 mm con un torque de 142 N*m.

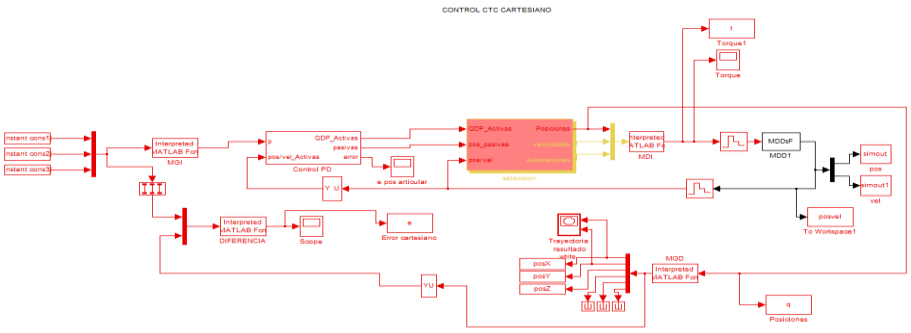


Figura 24. Control CTC cartesiano.

En la siguiente figura podemos observar que el máximo error cartesiano para una trayectoria circular se produce al inicio de la simulación y tiene un valor máximo de $1.5x10^{-3}$ m, es decir 1.5 mm.

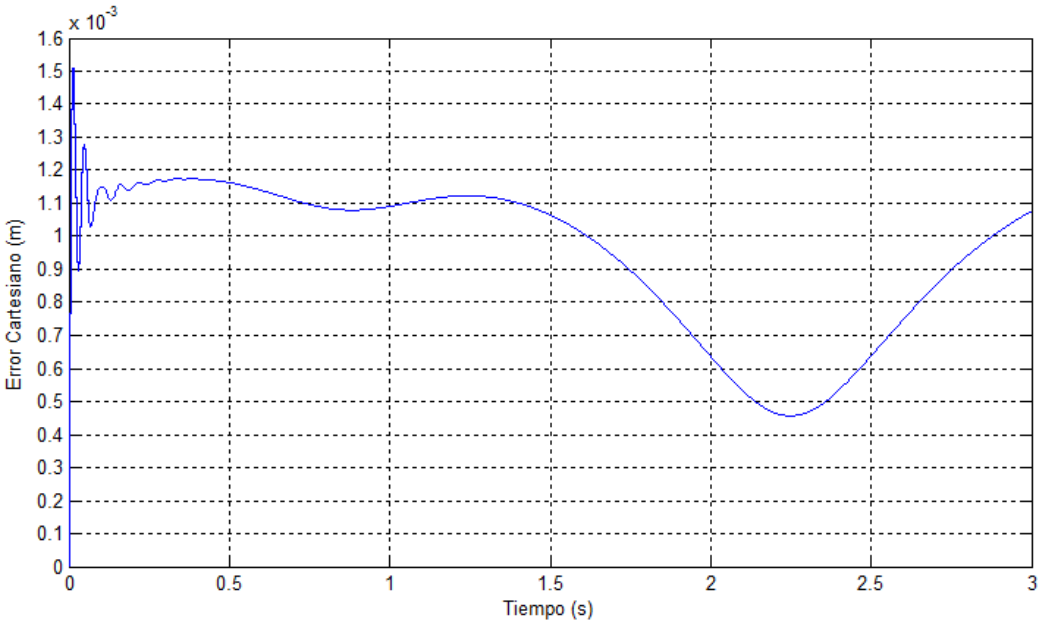


Figura 25. Error cartesiano para una trayectoria circular.

Finalmente en la figura 26 podemos observar que el máximo torque se produce en la articulación 2, una articulación motorizada con un torque de 14.2075 N*m o 142.075 $Kg*cm^2$. Por tal razón se decide escoger un motor reductor que genere un torque mayor al torque obtenido mediante la simulación.

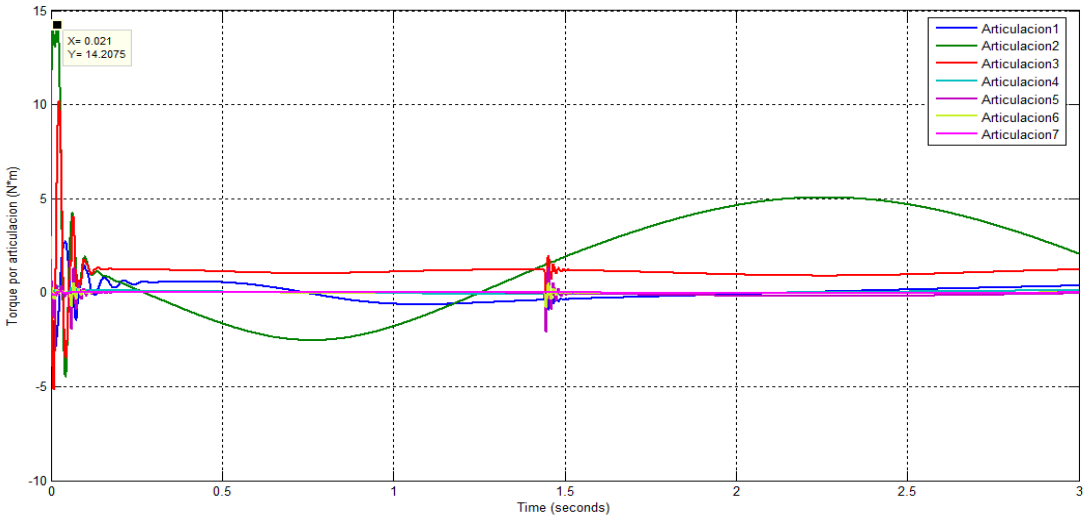


Figura 26. Torque generado para cada articulación.

2.5. Escogencia y puesta a punto de los sensores.

Para la escogencia de los sensores se tuvo en cuenta que estos debían tener la capacidad de medir la posición angular de cada una de las articulaciones que conforman el robot, por esto se determinó la escogencia de potenciómetros lineales que están ubicados en el centro de las articulaciones, exactamente en el eje de cada motor, de esta forma se logra obtener las posiciones exactas del robot. Los potenciómetros escogidos deben generar valores de voltaje proporcionales a la cantidad de grados que gira en torno a su eje, esto se verifica realizando pruebas con el potenciómetro.

La prueba realizada consistió en polarizar el potenciómetro a un valor de voltaje fijo y empezar a variar el ángulo de su eje, la variación que se realizó para obtener los valores es de 10 grados. A continuación se muestran los valores obtenidos para el voltaje y ángulo obtenidos para el potenciómetro.

Ángulo (grados)	voltaje
10	0,1158
20	0,231
30	0,35
40	0,463
50	0,579
60	0,688
70	0,81
80	0,931
90	1,05
100	1,16
110	1,27

Tabla 5. Valores de voltaje y Angulo del potenciómetro.

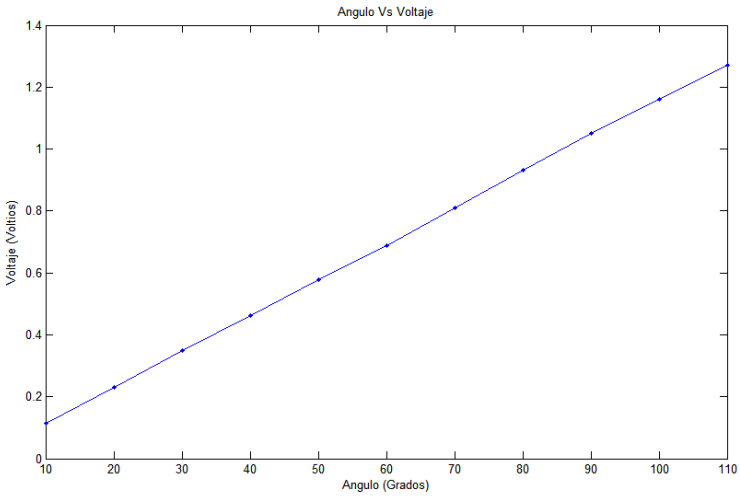


Figura 27. Grafica de Angulo Vs Voltaje.

El potenciómetro utilizado es un Vishay 533 a 3 vueltas, en su hoja de datos el fabricante especifica que es lineal, lo cual se demuestra en su puesta a punto mediante la experimentación.

Figura 28. Potenciómetro de precisión lineal Vishay 533.

2.6. Fuente de alimentación del robot.

Para la escogencia de la fuente de alimentación se tuvo en cuenta que debía generar la corriente suficiente para alimentar los motores y la instrumentación que conforman el robot.

Para realizar la puesta a punto de la fuente se conectan tres motores y se verifica que no haya una caída de tensión en la fuente en ningún momento.

Dentro de las especificaciones de la fuente está el valor nominal de corriente que soporta a una tensión de 5 voltios, este valor es de 31 amperios, suficiente para alimentar la instrumentación del robot.

Figura 29. Fuente de alimentación del robot.

2.7 Circuito de potencia puente h.

Un circuito puente h es un dispositivo electrónico que tiene la capacidad de cambiar la polaridad de la tensión de alimentación de un motor de corriente directa, esto permite que el dispositivo electro mecánico gire en ambos sentidos.

El puente h escogido para realizar la manipulación de los motores es un “pololu high-power” de 18 voltios y 15 amperios, este cuenta con 12 pines que se muestran a continuación en la siguiente figura.

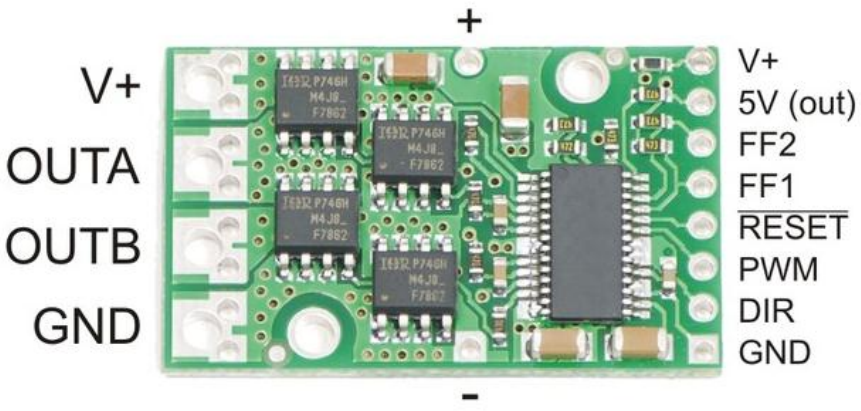


Figura 30. Tarjeta puente h

PIN V+: Este pin corresponde a la conexión de alimentación del motor, el rango de conexión es de 5v a 30v.

PIN OUT A: Primer pin de salida de la tarjeta.

PIN OUT B: Segundo pin de salida de la tarjeta.

GND: Conexión a tierra de la tarjeta.

DIR: Este es el pin de dirección que permite cambiar el sentido de giro del motor.

PWM: Es una entrada de modulación de ancho de pulso, esta permite disminuir la energía que se entrega a la salida del motor, lo que permite disminuir la velocidad de giro de este.

RESET: es una entrada que permite reiniciar la tarjeta.

FF1: Indicador de fallo 1.

FF2: Indicador de fallo 2.

3. Dispositivo electrónico para comunicación entre computador y robot.

El dispositivo electrónico encargado de la comunicación entre el robot y el computador es básicamente una tarjeta de adquisición de datos que tiene como base un micro controlador 18f 4550, esto debido a que soporta comunicación USB y tiene la suficiente cantidad de canales que permiten la lectura de sensores y manipulación de motores para lograr el funcionamiento del robot. Para la elaboración del hardware y el software del dispositivo se dividió el trabajo en tres partes, la primera es el diseño de la tarjeta de adquisición de datos, la segunda es el análisis y estudio del protocolo de comunicación USB, la tercera parte es la implementación del software que se embebe en la tarjeta de adquisición, y la última es la integración de la tarjeta con la instrumentación del robot.

3.1. Diseño de la tarjeta de adquisición de datos.

Para el diseño de la tarjeta de adquisición de datos se determinó que debía tener las siguientes características:

- Conexión mediante protocolo de comunicación USB.
- Capacidad de utilizar 7 conversores análogos-digitales.
- Bajo consumo de energía.
- Capacidad de utilizar 7 salidas digitales para manipulación de los motores
- Que se pueda integrar a la plataforma de desarrollo Visual Studio.

Inicialmente se determinó que la tarjeta debía tener un micro controlador que soportara comunicación USB, además este debía tener las entradas y salidas que requeríamos para el funcionamiento del robot. El dispositivo escogido es el micro controlador 18F4550 que hace parte de la gama alta de estos dispositivos. Además son necesarios algunos periféricos para completar la estructura de la tarjeta, estos son: cristal de cuarzo, capacitores para el filtrado de ruido a la entrada del micro controlador, capacitores para el filtrado de ruido en los pines de comunicación, resistencia de polarización, conector USB.

Figura 31. Micro controlador utilizado para construcción de tarjeta.

El micro controlador 18f4550 presenta las siguientes especificaciones técnicas:

- Micro controlador de 8 bits.
- 24 KB de memoria flash para programación.
- Memoria RAM de 2KB.
- EEPROM de 256 KB.
- Velocidad 48 MZ.
- 35 puertos de entrada-salida.
- 13 canales ADC de 10 bits.
- 4 timers.
- Voltaje de alimentación 3V - 5.5V.
- USB 2.0

Los terminales USB 2.0 se componen de cuatro pines internos a través de los cuales se transmiten los datos. Es necesario tener en cuenta que la potencia necesaria para la alimentación de la tarjeta debe estar dentro de un determinado rango. El modo de transmisión de los datos es diferencial, por lo que existen dos pines de comunicación, uno positivo y otro negativo, por lo tanto la conexión con el puerto del micro controlador permite la transferencia bidireccional de información, así como la alimentación directa del dispositivo.

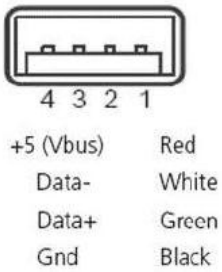


Figura 32. Terminal USB tipo A.

Una vez determinados todos los elementos se procede a su diseño en el software Eagle que permite la integración de instrumentos electrónicos para realizar el circuito impreso de toda la tarjeta.

Realizando las debidas conexiones para cada uno de los elementos se obtiene el resultado que se muestra en la figura 33. Finalmente este archivo generado se envía a un sitio especializado donde se realizan las pistas de este y se ensamblan los demás componentes, obteniendose la tarjeta que se muestra en la figura 34.

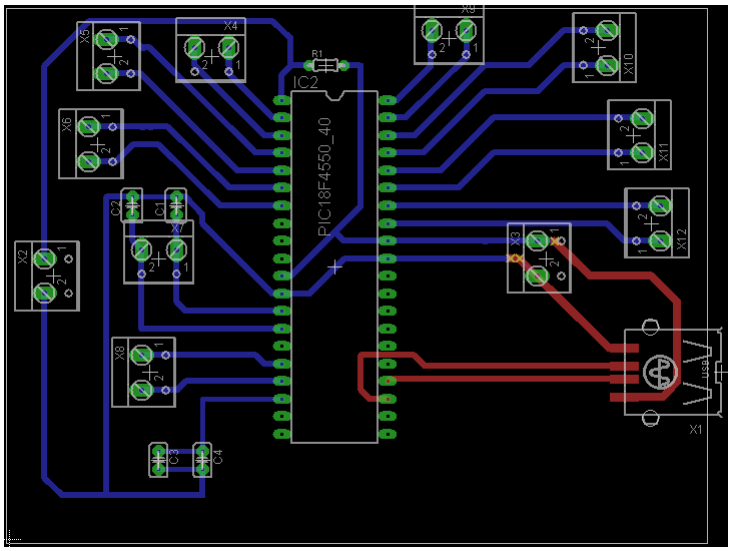


Figura 33. Circuito PCB de la tarjeta de adquisición.

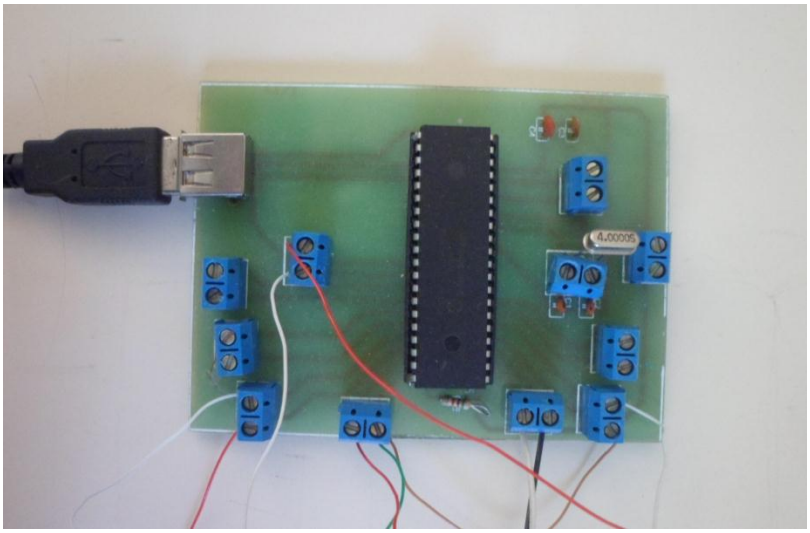


Figura 34. Tarjeta de adquisición.

3.2. Protocolo de comunicación USB.

Para desarrollar una aplicación que implemente el protocolo USB es importante conocer cómo es su funcionamiento, y que acciones se deben tener en cuenta para crear una aplicación que responda en tiempos cortos y realice las múltiples tareas que debe cumplir.

El protocolo USB es un sistema punto a punto, esto se da debido a que el lugar de partida es el host (PC o *hub*), y el destino es un periférico u otro *hub*, en una arquitectura USB solo existe un host y hasta 127 periféricos.

Los computadores estándar tienen hasta 5 entradas USB, lo que implica que para permitir que más de dos periféricos funcionen simultáneamente se implemente un hub [22].

3.2.1. Interfaz física del protocolo USB

A nivel eléctrico el cable USB transfiere la señal y la alimentación sobre 4 hilos.

Figura 35. Nivel eléctrico del cable USB.

A nivel de alimentación el cable proporciona la tensión nominal de 5 voltios. Es necesario definir correctamente el diámetro del hilo con el fin de que no se produzca una caída de tensión demasiado importante en el cable.

La señal es transmitida por un par trenzado con una impedancia característica de 90 Ω. La velocidad puede ser tanto de 12 Mbits/s como de 1.5 Mbits/s. La sensibilidad del receptor puede ser de al menos 200 mv y debe admitir un buen factor de rechazo de tensión en modo común.

El consumo de energía es bajo, el computador es el encargado de proporcional la tensión para su funcionamiento, además el periférico puede ser autoalimentado (fuente externa).

3.2.2. Conceptos generales en el protocolo USB.

Host: Dispositivo maestro que inicia la comunicación (generalmente la computadora).

Hub: Dispositivo que contiene uno o más conectores o conexiones internas hacia otros dispositivos USB, este habilita la comunicación entre el host con diversos dispositivos, cada conector representa un puerto USB.

Dispositivo compuesto: Es aquel dispositivo con múltiples interfaces independientes. Cada una tiene una dirección sobre el bus para cada interfaz.

Puerto USB: Cada host soporta solo un bus, cada conector en el bus representa un puerto USB, por lo tanto sobre el bus puede haber uno o varios conectores, pero solo puede haber una ruta y solo un dispositivo puede transmitir información a un tiempo.

Driver: Es un programa que habilita aplicaciones para poderse comunicar con el dispositivo. Cada dispositivo sobre el bus debe tener un driver, algunos periféricos utilizan los drivers que están incluidos en el sistema operativo Windows.

Puntos terminales (*endpoints*): Es una localidad específica dentro del dispositivo. El *endpoint* es un *buffer* que almacena múltiples bytes, típicamente es un bloque de la memoria de datos o un registro dentro del microcontrolador. Todos los dispositivos deben soportar el punto terminal cero. Este punto terminal es el que recibe todo el control y las peticiones de su estado durante la enumeración cuando el dispositivo esta sobre el bus.

Tuberias (*pipes*): Es un enlace virtual entre el host y el dispositivo usb, este enlace configura los parámetros asociados con el ancho de banda y que tipo de transferencia se va a utilizar (*control*, *bulk*, *isócrona* o *interrupt*).

La norma USB define dos tipos de enlaces virtuales (*pipes*), *stream* y *message*.

***Stream pipes*:** Se trata de un flujo sin formato USB definido, esto significa que se puede enviar cualquier tipo de dato. Este tipo de *pipe* soporta las transferencias (*bulk*, *isócronas*, *interrupt*), además tanto el dispositivo USB como el host pueden controlarlas.

***Message pipes*:** Este tipo de enlace virtual tiene un formato USB definido y solo puede soportar el tipo de transferencia *control*.

El enlace *virtual* o *pipe* puede ser de cuatro tipos:

Control: Modo utilizado para realizar configuraciones, existe siempre sobre el punto terminal cero (*endpoint* cero). Todos los dispositivos USB deben soportar este tipo de transferencia. Los datos de control sirven para configurar el periférico en el momento de conectarse a USB. Algunos drivers específicos pueden utilizar este enlace para transmitir su propia información de control. Este enlace no tiene perdida de datos puesto que los dispositivos de detección de recuperación están activados a nivel USB.

Bulk: Este modo se utiliza para la transición de importantes cantidades de información. Como el tipo control, este tipo de enlace no tiene perdida de datos, es útil cuando la razón de transferencia no es crítica. Solo los dispositivos de media y alta velocidad utilizan este tipo de transferencia.

***Interrupt*:** Modo utilizado para transmisiones de pequeños paquetes, rápidos, orientados a percepciones humanas (datos, puntero). Este tipo de transferencia es utilizado en dispositivos que deben recibir atención periódicamente y lo utilizan dispositivos de baja velocidad.

Este tipo de transmisión garantiza la transferencia de pequeñas cantidades de datos. El tiempo de respuesta no puede ser inferior al valor especificado por la interfaz.

Isócrona: Este tipo de enlace da prioridad a la correcta transmisión de los datos, y los tiempos de ejecución son muy cortos, la transmisión de voz es un ejemplo de esta aplicación.

Para establecer la comunicación entre un dispositivo USB y un computador debemos tener en cuenta que se deben establecer canales lógicos unidireccionales que llamamos *pipes*, por medio de los cuales circulan los datos que se conectan desde el controlador del *host* hasta una entidad lógica en el dispositivo llamada *endpoint*.

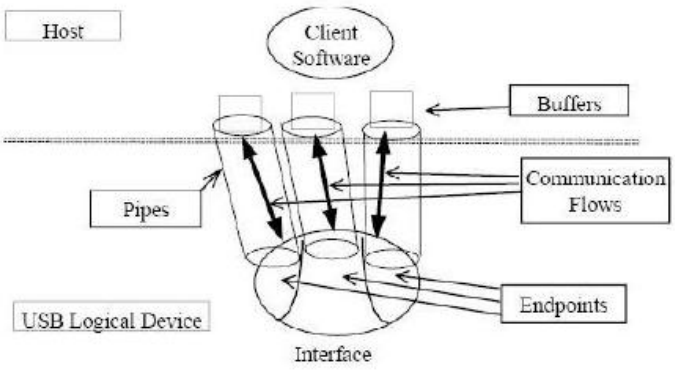


Figura 36. Diagrama lógico de comunicación USB.

Dependiendo del tipo de transferencia que se va a manejar en la comunicación, se deben definir los *pipes* correspondientes, por ejemplo si se va a establecer una transferencia *isócrona*, se debe crear un *pipe* de ese tipo [23].

El protocolo USB maneja una estructura semejante a la del modelo OSI pero reducido a tres niveles. La capa superior es la capa lógica de los niveles, esta se asemeja al nivel de aplicación que es donde se realiza la implementación del programa de acuerdo a los requerimientos establecidos. La capa intermedia se encarga de la gestión de los datos entre el nivel tres y el uno, esta capa está conformada por los drivers y vinculadores que se requieren desde la tarjeta y el computador para poder utilizar el protocolo, por ultimo está el nivel más bajo, que está directamente relacionado con las conexiones eléctricas que se deben tener en cuenta para la implementación de este tipo de comunicación [23].

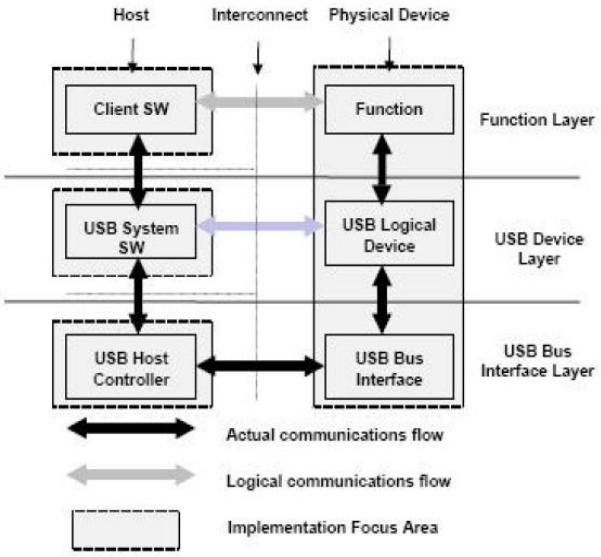


Figura 37. Modelo estructural USB.

3.3. Software para la tarjeta de adquisición de datos.

El programa que realiza las diferentes tareas para lograr la manipulación del robot se desarrolló dentro del IDE de programación para sistemas embebidos MPLAB V8.43, utilizando el compilador CCS, este último debe ser instalado en conjunto con un *plugin* para que pueda ser asociado con el IDE. Debe ser configurado como un lenguaje de compilación, utilizando la opción *select lenguaje tool suite* (Figura 38), que se encuentra dentro del menú Project de MPLAB, y buscando la localización donde se produjo la instalación de CCS por parte del usuario [23].

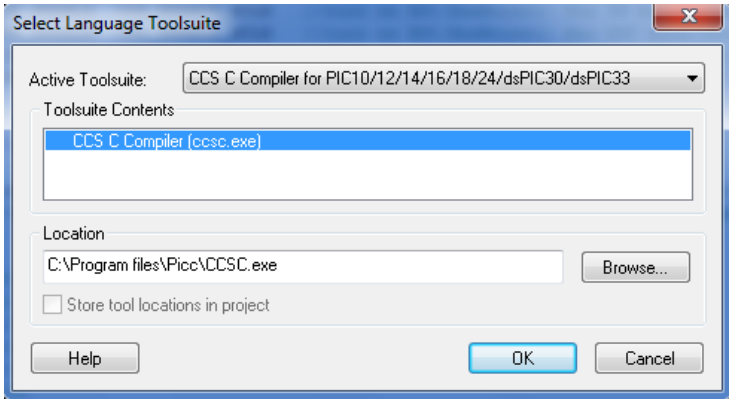


Figura 38. Configuración del lenguaje de compilación.

El software de manipulación embebido en la tarjeta de adquisición de datos comienza su ciclo de trabajo al recibir las primeras consignas provenientes del host, estas consignas son las posiciones angulares de las articulaciones del robot las cuales se reciben en grados, seguidamente de ser recibidas se convierten de grados a bits, posteriormente se activa un temporizador el cual fijara nuestro periodo de muestreo. Dentro de la rutina del temporizador primeramente se leen los conversores analógicos digitales, es decir la posición de los sensores, seguidamente se calcula la ley de control, se envía la ley de control, se envían los datos del conversor analógico digital, se espera hasta completar el tiempo de muestreo y con ello se reinicie el ciclo dentro del temporizador. Si se reciben nuevas consignas provenientes del host, estas se convertirán nuevamente de grados a bits y se actualizan estas consignas dentro de la rutina del temporizador.

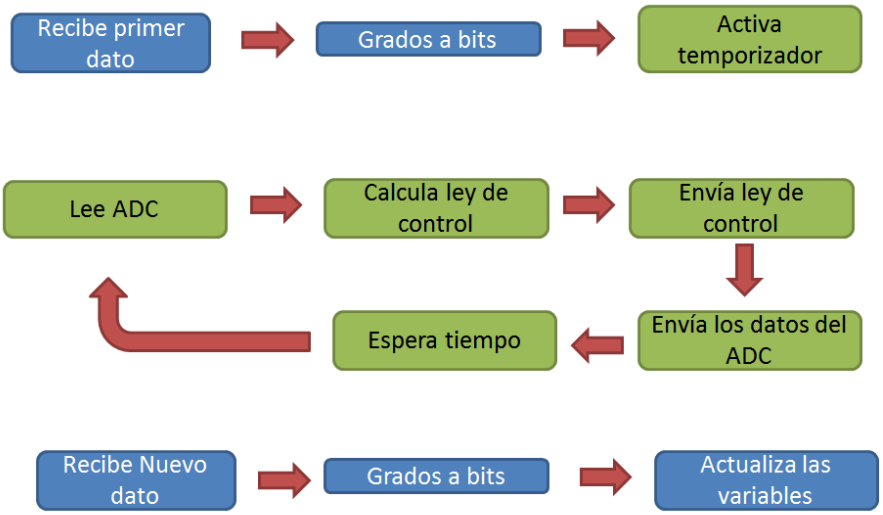


Figura 39. Esquema general del software embebido en la tarjeta.

3.3.1. Archivos de cabecera.

Inicialmente cuando se crea un nuevo proyecto en la plataforma MPLAB se deben incluir los archivos de cabecera necesarios para el desarrollo del proyecto, estos son 18f 4550.h que se encuentra en la línea 1 del programa principal, este archivo contiene los descriptores asociados con este microcontrolador. El segundo es pic18_usb.h; está en la línea 2, este archivo contiene los descriptores físicos de comunicación USB para PICs de la familia 18. Y finamente tenemos usb.c en la línea 3 que contiene los manejadores y los *tokens* de configuración. Todos estos archivos de cabecera son proporcionados por MPLAB y la librería de CCS por lo que basta con incluirlos en el proyecto mediante el comando #include. A continuación se muestran las líneas de código anteriormente mencionadas.

```
#include <18F4550.h>
#include <pic18_usb.h>
#include <usb.c>
```

3.3.2. Rutina de comunicación USB.

La rutina de comunicación y transmisión de datos se implementa en el archivo principal cuya extensión es .c. En forma general este archivo contiene en primera instancia las configuraciones iniciales y la declaración de variables a utilizar en la rutina, en segundo lugar el establecimiento de comunicación USB con el ordenador y finalmente el bucle infinito de comunicación y transmisión de los datos.

Inicialmente realizamos la configuración de los *fuses* del microcontrolador (configuraciones del micro controlador), estos sirven para configurar ciertos aspectos del dispositivo. Cada uno activa o desactiva una opción de funcionamiento, de tal forma que con un cristal externo de 4 MHZ se obtenga a la entrada del procesador del micro controlador una frecuencia de 4.8 MHZ que es la requerida para establecer la comunicación USB [24].

A continuación se muestra la línea para la configuración de los fuses.

```
#fusesTPLL,NOWDT,NOPROTECT,NOLVP,NODEBUG,USBDIV,PLL1,CPUDIV1,VREGEN
```

Una vez configurados los fuses se procede a realizar la definición de los descriptores USB de acuerdo al tipo de comunicación que se desea implementar. Lo que inicialmente hacemos es deshabilitar el uso de directivas HID (línea 1). Luego habilitamos el *endpoint* 1(EP1) para entrada y salida de datos tipo BULK (línea 2 y 3) [25].

```
#define USB_HID_DEVICE
#define USB_EP1_TX_ENABLE  USB_ENABLE_BULK
#define USB_EP1_RX_ENABLE  USB_ENABLE_BULK
```

En la tarjeta de adquisición de datos se incluyen leds que nos permiten conocer el estado en el cual se encuentra el dispositivo, cuando se conecta la tarjeta al computador se enciende el primer led (línea 1 y 2). Este me indica que las conexiones eléctricas están correctamente establecidas, pero que aún no se ha establecido comunicación USB, ya sea porque no se han instalado los controladores de dispositivo o porque existe algún problema con el computador o la tarjeta. Cuando los procesos de inicialización USB de habilitación de periférico, interrupciones y enumeración por parte del host se realizan de una manera satisfactoria, el primer led se apaga y el segundo led se enciende (línea 3 a 7). Estos indicadores se encuentran asociados a los pines B5 y B6 del puerto B del micro controlador.

```
output_low(PIN_B4);
output_high(PIN_B5);
usb_init();
usb_task();
usb_wait_for_enumeration();
 output_low(PIN_B5);
 output_high(PIN_B4);
```

Cuando se establece la comunicación USB de manera correcta comienza el bucle infinito, dentro de cual se produce el intercambio de datos entre el ordenador y el microcontrolador. Inicialmente e PIC se debe configurar, y está a la espera de los datos provenientes desde el computador (línea 1 y 2).

```
if(usb_enumerated())
if (usb_kbhit(1))
```

3.3.3. Rutina de adquisición y transmisión de datos.

Lo que se hace inicialmente para la adquisición y transmisión de información es definir el tamaño del buffer para transmisión (línea 1) y recepción (línea 2), asociados también a cada uno de los *endpoints*. Dichos tamaños están dados en paquetes de 8 bits.

```
#define USB_EP1_TX_SIZE  10

#define USB_EP1_RX_SIZE    3
```

Para este caso en particular el tamaño del paquete de salida es 10 bits, que es un paquete que contiene información de la conversión ADC a través de los canales analógicos del micro controlador, cada canal lee la posición angular de cada una de las articulaciones del robot a través de los potenciómetros lineales. El conversor ADC del microcontrolador es de 10 bits, esto quiere decir que puede convertir valores de voltaje de 0-5 voltios en valores numéricos de 0-1023, lo que implica que para enviar las posiciones de cada una de las articulaciones se requieran 2 bytes, y para enviar la información en paquetes de 2 bytes se deben concatenar.

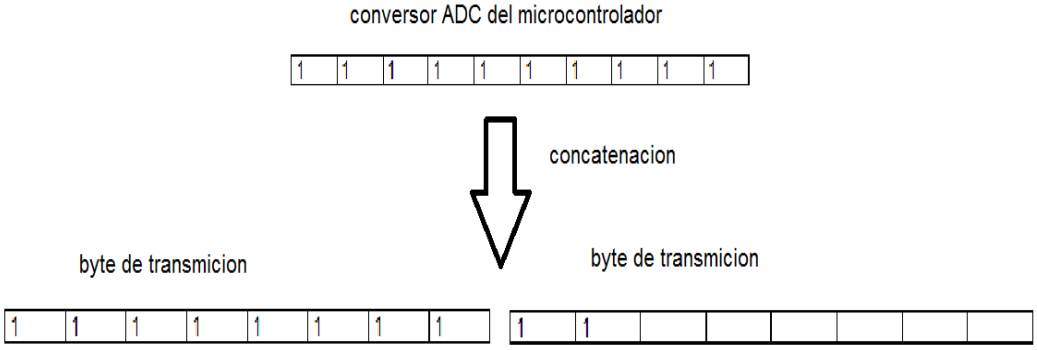


Figura 40. Concatenación de información del ADC.

El paquete de recepción va a tener un tamaño de 3 bytes, la explicación de esto es que son las posiciones angulares deseadas que se generan desde el joystick y que gracias al modelo geométrico inverso generado en el ordenador se conoce el valor exacto que debe moverse una articulación para llegar a una determinada posición, cada byte contiene la información de cuánto se debe mover cada articulación.

Una vez es definida la cantidad de información de transmisión y recepción se procede a realizar la declaración de variables.

Para la transmisión se declaran así:

```
#define h envia[0]
#define l envia[1]
#define h1 envia[2]
#define l1 envia[3]
#define h2 envia[4]
#define l2 envia[5]
#define h3 envia[6]
#define l3 envia[7]
#define h4 envia[8]
```

```
#define l4 envia[9]
```

Para recepción se declaran así:

```
int16 x1;
int16 x2;
int16 x3;
int16 rk;
int16 rk1;
int16 rk2;
int16 rk3;
int16 rk4;
```

Después de realizar la declaración de variables se entra al ciclo infinito que se nombró anteriormente en la rutina de comunicación. Una vez dentro del ciclo y establecida la comunicación entre el computador y la tarjeta se puede leer los conversores ADC.

Para leer los conversores inicialmente se declara el canal sobre el cual se quiere realizar la lectura (línea 1), para nuestro caso en particular se usaran cuatro canales desde el cero hasta el cuatro, el siguiente paso es colocar un retardo para realzar el salto de canal (línea 2), y finalmente a una variable se le asigna la lectura de dicho canal.

```
set_adc_channel(0);
delay_us(20);
rk = read_adc();
```

Continuando con el programa se debe realizar el encapsulamiento de los datos del ADC en las variables que van a ser enviadas por USB. Para esto se asignan los descriptores para la parte alta y baja de los 10 bits del ADC(línea 1 y 2). Cada descriptor se asigna en una variable de 8 bits que a su vez se asigna a la variable **envía**, y finalmente se realiza la salida de la información mediante la función usb_put_packet. Aquí se declara el nombre de la variable y el tamaño de la trama a enviar (línea 3).

```
h4=ADRESH;
l4=ADRESL;
usb_put_packet(1, envia,10, USB_DTS_TOGGLE);
```

3.3.4. Rutina para manipulación de los actuadores.

Para realizar la rutina de manipulación inicialmente tomamos el valor de voltaje que entrega el potenciómetro lineal y se escaliza para obtener la posición de cada articulación en unidades de medida angular, para este caso se trabaja con grados.

Debido a que el potenciómetro lineal es de tres vueltas, es necesario conectar este dispositivo a un valor de voltaje de 12 voltios que es el máximo valor que entrega la fuente utilizada, de esta forma se obtiene que para una vuelta existe una variación de 4 voltios.

La escalización consiste básicamente en realizar una regla de tres, en la cual se multiplica el valor de la variable del conversor análogo digital por la cantidad angular máxima a una vuelta, que es 360 grados, y lo dividimos por el valor máximo en unidades decimales que lee el conversor.

```
ref1=rk*(360.0/819.0);
```

El siguiente paso es medir el error en la articulación (línea 1). Para esto se realiza una resta entre las posiciones deseadas que provienen desde el computador y las posiciones articulares del robot real, este valor es asignado a una variable para posteriormente ser utilizada.

```
error1=p1-ref1;
```

Una vez calculado el error se procede a realizar la lógica de activación para los actuadores. Inicialmente la referencia y las posiciones deseadas están en un mismo valor numérico, para esto se ubica el robot real en las mismas posiciones que genera el modelo geométrico inverso en el computador.

Si existe un movimiento de manipulación desde el computador las posiciones deseadas se modificarán y por lo tanto se producirá un valor de error positivo o negativo dependiendo de la dirección en que se mueva el joystick. Si el error es positivo se entra a un condicional, enseguida se pregunta si el valor de error excede los 4 grados, si ocurre esto se activa el actuador en un sentido, y si no ocurre esto se apaga el actuador. Si lo que ocurre es que se da un valor de error negativo se entra a un condicional y se pregunta si el error es mayor a -4 grados, si ocurre esto se activa el motor a la izquierda, y si no se apaga el actuador, a continuación se muestran las dos lógicas de activación.

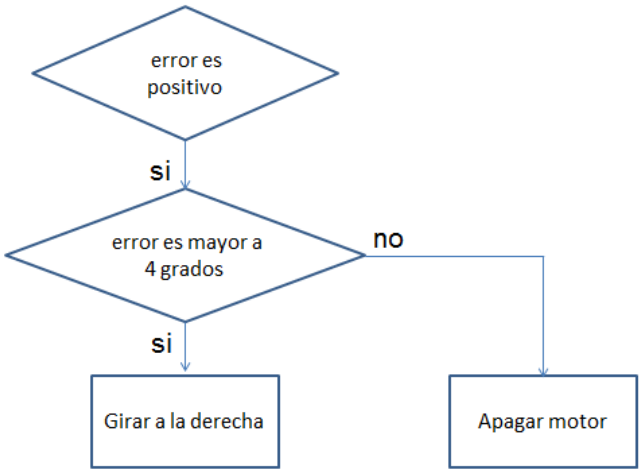


Figura 41. Lógica de activación 1.

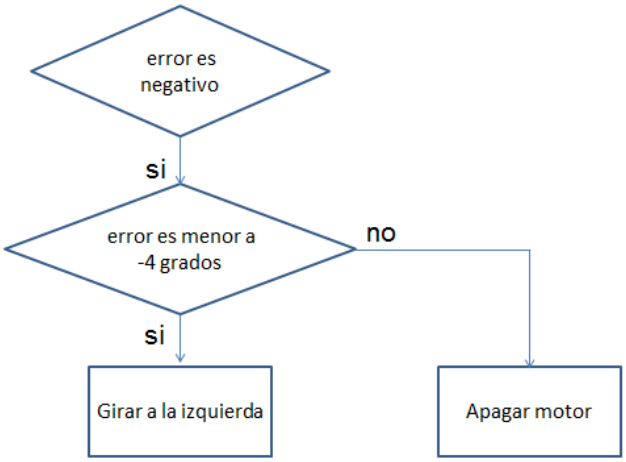


Figura 42. Lógica de activación 2.

El mismo procedimiento se realiza para las tres primeras articulaciones que son las motorizadas.

4. Software para manipulación del robot Hibou.

Para la implementación del software de manipulación se hace uso de un código creado por los ingenieros Luis Daladier Guerrero y Cristian Méndez, desarrollado en una materia de la Maestría en Automática de la Universidad del Cauca, Colombia, este software fue modificado para realizar la comunicación entre el robot y el ordenador, este presenta la siguiente estructura de funcionamiento.

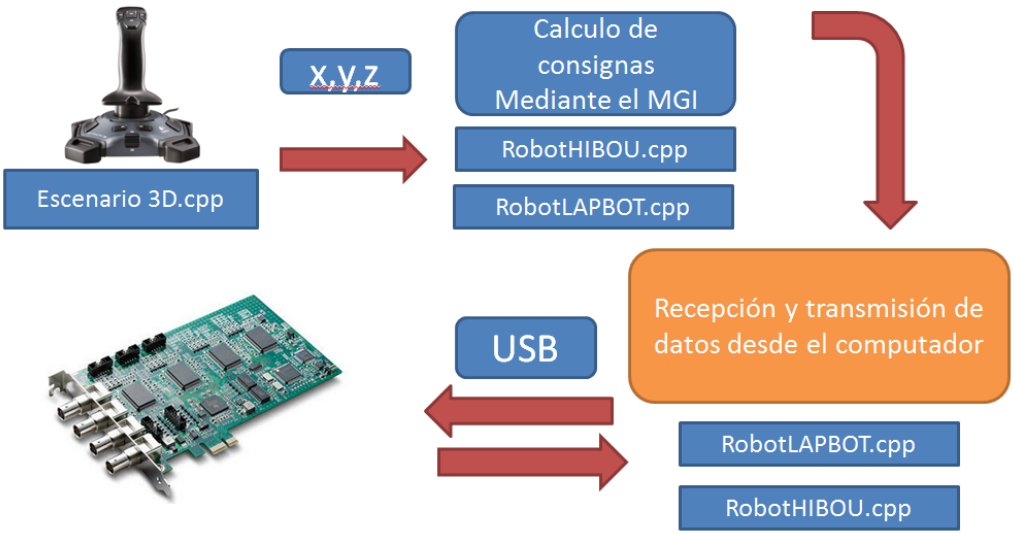


Figura 43. Estructura del software de manipulación.

Inicialmente el programa realiza la lectura de las posiciones cartesianas X,Y,Z provenientes del joystick, estos datos corresponden a la posición deseada en el efector final. Posteriormente estos datos pasan como entradas al Modelo Geométrico Inverso, aquí se realizan una serie de cálculos matemáticos que al final generan las consignas deseadas para cada articulación, estas consignas son las posiciones angulares que debe moverse cada una de las articulaciones para alcanzar la posición final.

Una vez calculadas las posiciones deseadas se procede a realizar la transición de las consignas, para esto se utiliza la librería que permite enviar y recibir datos utilizando el protocolo de comunicación USB.

Para la realización del software se utilizaron las siguientes librerías y plataformas:

- Visual Studio (plataforma de desarrollo).
- Librería VTK (motor gráfico).

4.1. Plataforma de desarrollo Visual Studio.

Microsoft Visual Studio es un entorno de desarrollo integrado (IDE por sus siglas en ingles) para sistemas operativos Windows, soporta varios lenguajes de programación como Visual C++, Visual C#, Visual J#. ASP:NET y Visual Basic .NET, aunque en la actualidad existe compatibilidad y extensiones para otros [26].

Microsoft Visual Studio permite crear aplicaciones que integran librerías y motores gráficos para la implementación de simuladores y al mismo tiempo otras librerías para conexión a hardware externo, lo que permite realizar aplicaciones externas al computador mediante un protocolo de comunicación como USB.

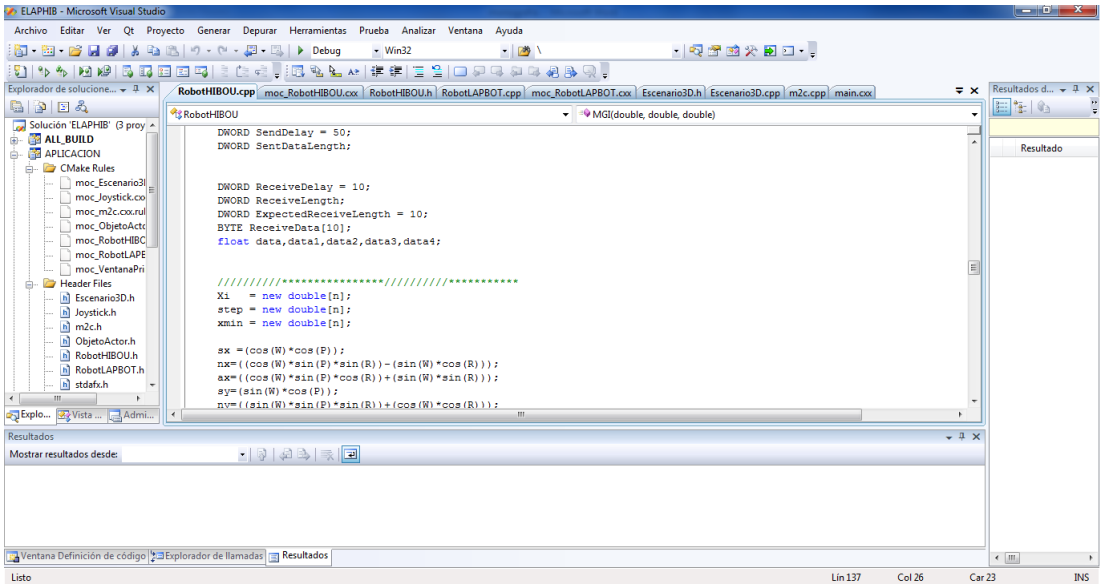


Figura 44. Interfaz gráfica Visual Studio 2008.

4.2. Librería para interfaz gráfica VTK.

VTK es un conjunto de librerías de código abierto orientada a objetos y disponible para visualización y tratamiento de imágenes. VTK está implementado en C++, utiliza la aplicación CMake para el proceso de compilación y además puede integrarse en herramientas como Qt.

VTK es una de las herramientas estándar más importante en la programación para el análisis de imágenes ya que cuenta con una amplia funcionalidad tanto para la imagen, como para tratamiento de superficies y visualización. No sólo permite visualizar figuras geométricas, sino que además soporta una amplia variedad de algoritmos de visualización y otras modernas técnicas de

modelado en 2D y 3D. Debido a su gran potencia, se hacen necesarios amplios recursos de memoria en el ordenador para poder aprovechar la totalidad de sus funcionalidades [27].

4.3. Herramienta de vinculación CMAKE.

CMaKe es una herramienta multiplataforma basada en un sistema de código abierto. Este programa se utiliza para controlar el proceso de compilación. Básicamente realiza el proceso de traducción de las instrucciones escritas en un determinado lenguaje de programación a lenguaje máquina, el cual es interpretado por la computadora.

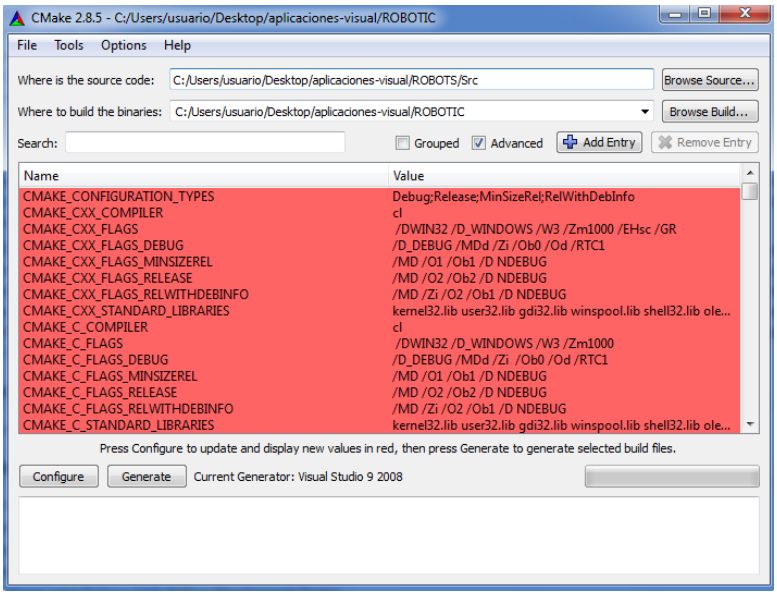


Figura 45. Interfaz gráfica herramienta CMake.

4.4. Integración de la librería MPUSBAPI.DLL al software para manipulación.

Se realiza la integración de la librería USBAPI al software de manipulación para el envío de datos desde el computador hacia la tarjeta de adquisición, los datos enviados son las posiciones articulares deseadas calculadas por medio del modelo geométrico inverso del robot implementado en este software.

4.4.1. Vinculación explicita de MPUSBAPI.DLL a Visual Studio 2008.

Lo que debemos hacer para lograr la vinculación es primero agregar a la carpeta donde se encuentra nuestro proyecto 3 de los 4 archivos que componen la librería, usb4550.h, usb4550.lib y stdafx.h. Después estando

dentro de la interfaz de Visual Studio nos ubicamos en el explorador de soluciones sobre nuestro proyecto, damos click derecho, vamos a la opción propiedades y se nos desplegará una ventana, damos click izquierdo sobre Propiedades de la configuración >> Vinculador >> Entrada y aquí agregamos usb4550.lib como una dependencia adicional (ver figura 46).

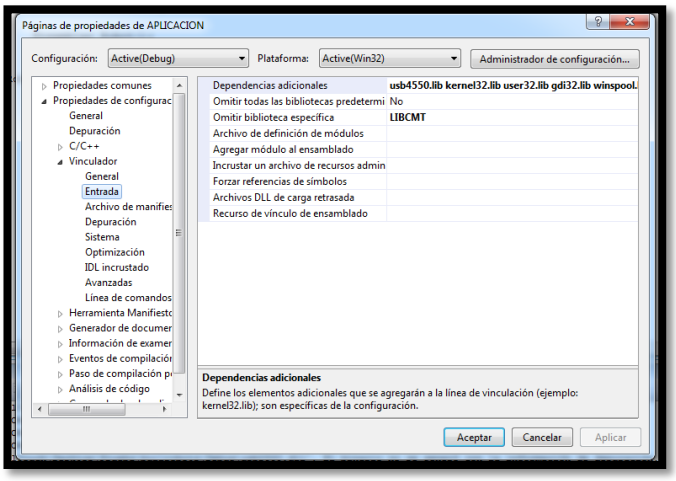


Figura 46. Vinculación de la librería MPUSBAPI a Visual Studio 2008.

Posteriormente en esta misma ventana vamos a Propiedades de la configuración >> General Dentro de los valores predeterminados del proyecto se escoge Uso de MFC y elegimos la opción Utilizar MFC en una archivo DLL compartido (ver figura 47).

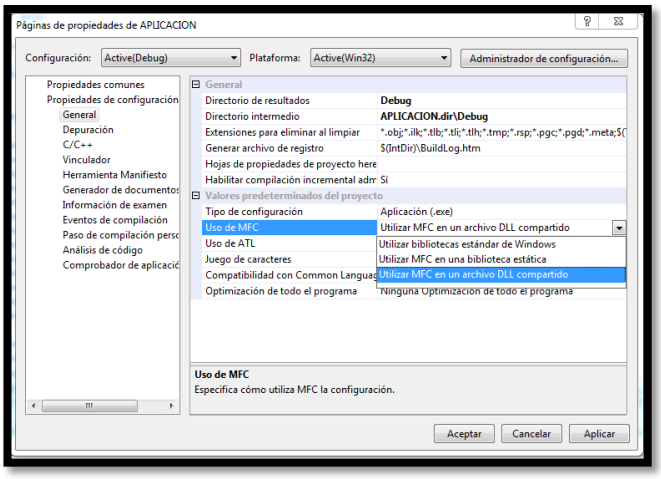


Figura 47. Pasos a seguir para configuración de la DLL.

Por último debemos agregar el archivo usb4550.dll a la carpeta donde se genera nuestra aplicación. La aplicación generada y el archivo usb4550 siempre deben estar juntos debido a que la aplicación dependerá de la librería ubicada en el archivo dll.

4.4.2. Declaración de variables locales para hacer uso de las funciones de la librería MPUSBAPI.DLL.

A continuación se explicaráa cómo se realiza la declaración de variables locales en Visual Studio.

PCHAR vid_pid = "vid_04d8&pid_0011"; es la cadena de caracteres del número de identificación asignado del puerto USB, su formato es “vid_xxxx&pid_yyyy” donde xxxx y yyyy son los valores hexadecimales del VID y el PID respectivamente.

Si un dispositivo tiene un VID=0x04d8 y un PID=0x000b, la cadena de entrada es: “vid_0x04d8&pid_0x000b”

```
PCHAR out_pipe= "\\MCHP_EP1";
PCHAR in_pipe= "\\MCHP_EP1";
```

Cadena de caracteres con el número del *Endpoint* que se va a abrir para establecer la comunicación. El formato es \\MCHP_EPz, donde z es el número del *endpoint*. En este caso *out_pipe* es el *endpoint 1* para escribir datos en el puerto USB e *in_pipe* es el *endpoint 1* para leer los datos del puerto USB.

```
HANDLE myOutPipe;
```

Es la variable que identifica el *pipe* del *endpoint* que se va a escribir.

```
HANDLE myInPipe;
```

Es la variable que identifica el *pipe* del *endpoint* que se va a leer.

```
BYTE SendData[2];
```

Variable donde se almacenan los datos que se envían por el puerto USB.

```
BYTE ReceiveData[10];
```

Variable donde se almacenan los datos que se reciben por el puerto USB.

```
DWORD SendLength = 2;
```

Número de bytes que se van a escribir en *el pipe*.

```
DWORD ExpectedReceiveLength = 10;
```

Número de bytes que se van a leer en el *pipe*.

```
DWORD SentDataLength;
```

Puntero al número de bytes que se escriben.

```
DWORD ReceiveLength;
```

Puntero al número de bytes que se leen.

```
DWORD SendDelay = 5;
```

Especifica el intervalo de *time-out* en milisegundos.

```
DWORD ReceiveDelay = 10;
```

Especifica el intervalo de *time-out* en milisegundos.

4.4.3. Funciones de la librería MPUSBAPI.DLL.

- **MPUSBOpen:** Función que devuelve el acceso al *pipe* del *endpoint* con el VID_PID asignado.

```
myOutPipe = MPUSBOpen(0,vid_pid,out_pipe,0,0);
```

- **MPUSBRead:** Función que permite leer los datos del puerto USB.

```
MPUSBRead (myInPipe, (void*)ReceiveData, ExpectedReceiveLength, &ReceiveLength,ReceiveDelay);
```

- **MPUSBWrite:** Función que permite enviar los datos en el puerto USB.

```
MPUSBWrite (myOutPipe,(void*)SendData,SendLength,&SentDataLength,SendDelay);
```

- **MPUSBClose:** Función que cierra la comunicación.

```
MPUSBClose( myOutPipe);
```

4.4.4. Procesamiento de los datos recibidos y enviados por el puerto USB.

Se programó el conversor analógico digital del PIC 18F4550 de 10 bits para tener un rango de lectura de los potenciómetros lineales más amplio, teniendo en cuenta que por el puerto USB solo se puede trasmitir tramas de un byte es decir 8 bits con este microcontrolador. Se usan dos cadenas de 8 bits de los cuales se harán uso de solo 10 bits y se envían atreves del puerto USB, por tal razón al recibir estos datos para verificar el valor del canal analógico digital debemos unificar los dos bytes.

```
MPUSBRead(myInPipe, (void*)ReceiveData, ExpectedReceiveLength,
&ReceiveLength,ReceiveDelay);

data=(ReceiveData[0]*256+ReceiveData[1]);
data1=ReceiveData[2]*256+ReceiveData[3];
data2=ReceiveData[4]*256+ReceiveData[5];
data3=ReceiveData[6]*256+ReceiveData[7];
data4=ReceiveData[8]*256+ReceiveData[9];
```

Imprimimos el valor de los datos en la pantalla secundaria, ya que estos datos solo serán usados para verificación de la posición de los 5 potenciómetros lineales, información que es relevante para el usuario.

```
cout<<data;
cout<<"  pos"<<pos;
cout<<"      "<<data1;
cout<<"      "<<data2;
cout<<"      "<<data3;
cout<<"      "<<data4<<endl;
```

Hacemos un escalamiento a los datos obtenidos por el modelo geométrico inverso antes de proceder a enviarlos a la tarjeta de adquisición de datos.

5. Etapas de prueba del sistema.

En esta sección se abordarán las pruebas realizadas al sistema. Es decir la manipulación en lazo abierto y lazo cerrado de las articulaciones del robot que ubican el órgano terminal en una determinada posición.

5.1. Manipulación en lazo abierto.

En esta manipulación el sensor (potenciómetro lineal) no se acopla al eje del motor, se ubica en una posición fija la cual corresponderá a la posición central del joystick, para obtener movimiento en el motor solo cuando se realiza un movimiento en el joystick.

Se prueba articulación por articulación independientemente, para verificar que cada una de ellas tenga un adecuado giro a favor de las manecillas del reloj o en contra de las manecillas, según corresponda a un determinado movimiento en el joystick.

5.2. Manipulación en lazo cerrado.

Para la manipulación en lazo cerrado como su nombre lo específica se cierra el lazo acoplando el sensor al eje del motor, así el sensor proporcionará la posición en la cual se encuentra la articulación.

Se prueba igualmente cada articulación por separado, se realiza manipulación según el algoritmo embebido en la tarjeta de adquisición de datos que nos permite comparar la referencia con la salida y así producir un giro en contra o a favor de las manecillas del reloj, hasta llegar a la posición deseada.

A continuación se muestran los movimientos del órgano terminal del robot en las diferentes direcciones X, Y y Z.

Movimientos en Y:

- Figura 48. El órgano terminal se encuentra en la posición central no se realiza ningún movimiento en el joystick.
- Figura 49. El órgano terminal se mueve hacia atrás, cuando el joystick se mueve hacia adelante.
- Figura 50. El órgano terminal se mueve hacia adelante, cuando el joystick se mueve hacia atrás.

Movimientos en X:

- Figura 51. El órgano terminal se encuentra nuevamente en la posición central, cuando no se realiza movimiento en el joystick (vista lateral).

- Figura 52. El órgano terminal se mueve hacia la derecha, cuando el joystick se mueve hacia la izquierda.
- Figura 53. El órgano terminal se mueve hacia la izquierda, cuando el joystick se mueve hacia la derecha.

Movimientos en Z:

- Figura 54. El órgano terminal se introduce al máximo, cuando en el joystick se mueve el botón deslizante totalmente hacia abajo.
- Figura 55. El órgano terminal sale al máximo, cuando en el joystick se mueve el botón deslizante totalmente hacia arriba.

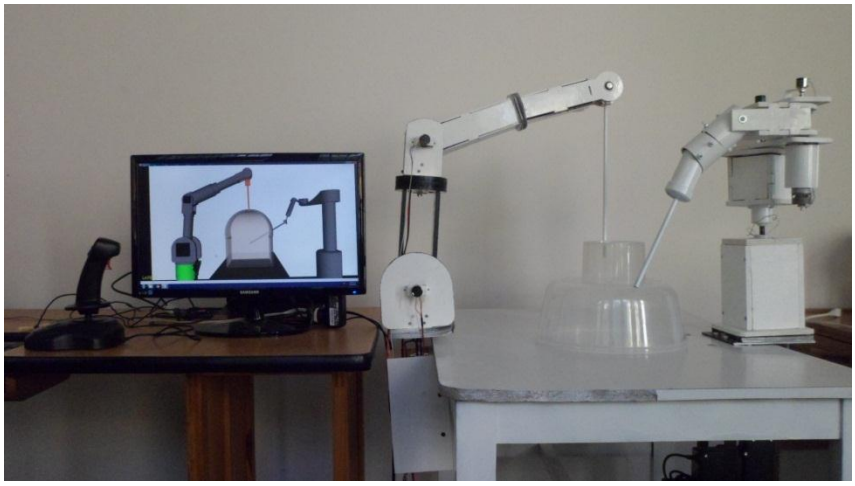


Figura 48. Órgano terminal en la posición central.

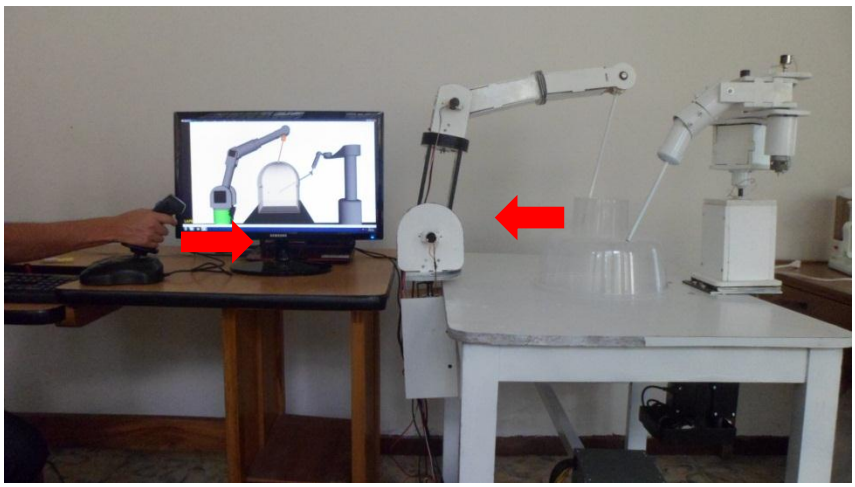


Figura 49. Órgano terminal ubicado hacia atrás.

Figura 50. Órgano terminal hacia adelante.

Figura 51. Órgano terminal en la posición central (vista lateral).

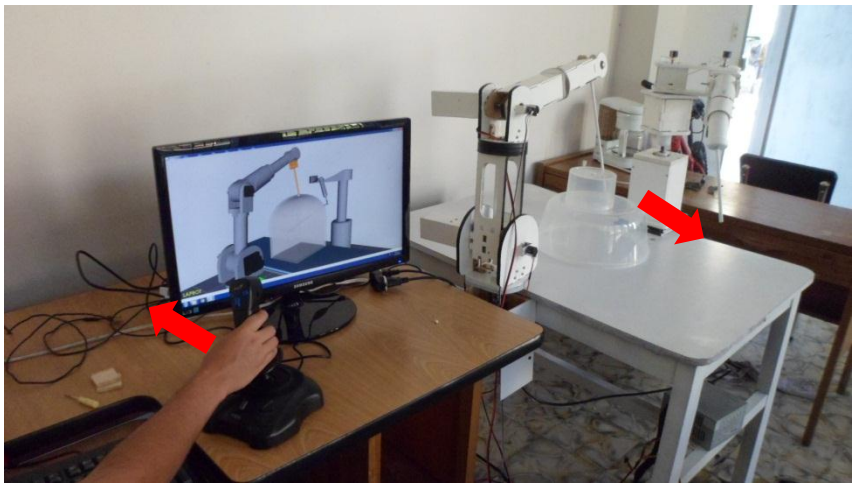


Figura 52. Órgano terminal en la derecha.

Figura 53. Órgano terminal en la izquierda.

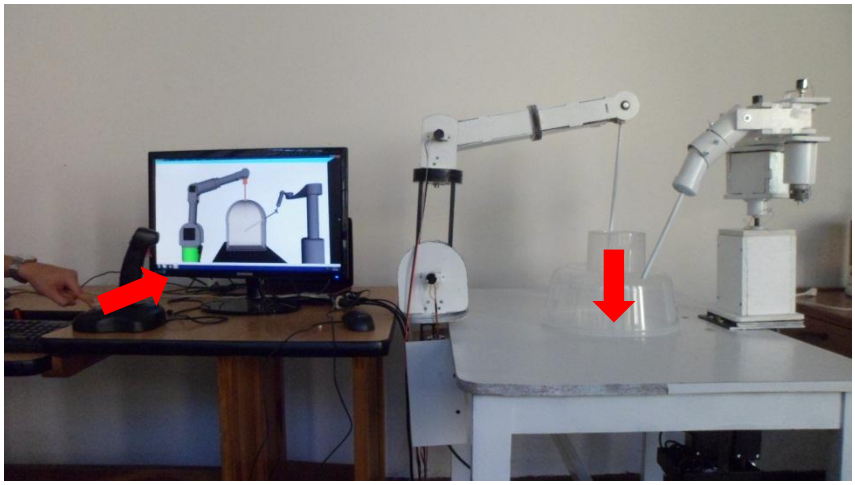


Figura 54. Órgano terminal introducción máxima en el abdomen.

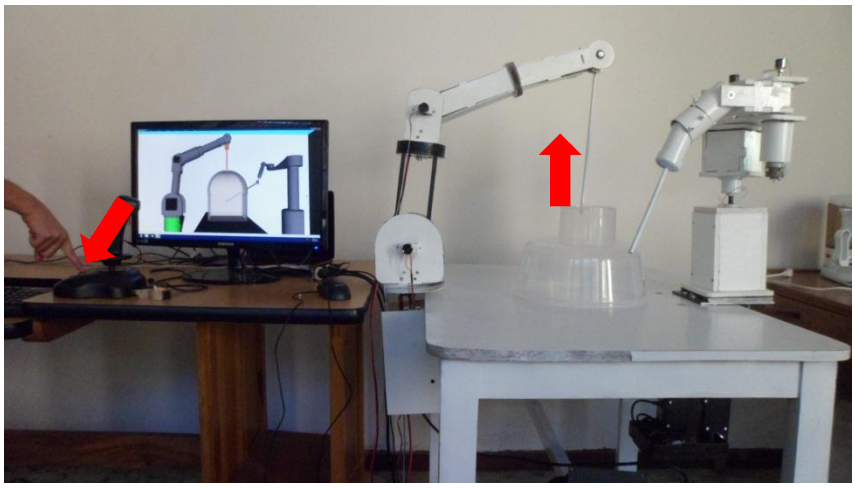


Figura 55. Órgano terminal introducción mínima en el abdomen.

6. Conclusiones y trabajos futuros.

6.1. Conclusiones.

- Se logró realizar el diseño del robot porta endoscopio con miras a su construcción real, para esto se utilizó un software de diseño que permite la creación de piezas que unidas forman la estructura completa del robot, y esta a su vez respeta las distancias establecidas previamente entre las articulaciones.

- Se determinaron por medio de una simulación los torques máximos requeridos para cada articulación, para esto se utilizó la estructura de un controlador CTC que calcula la fuerza necesaria para cada articulación, siguiendo una trayectoria circular. Con esto se logra un dimensionamiento aproximado del par necesario para la escogencia de los actuadores, estos últimos debían tener una par mayor a los pares máximos para garantizar el movimiento.

- Se pudo determinar la instrumentación adecuada para la construcción del sistema, esta incluye sensores que permiten realizar la medición de las posiciones articulares del robot, etapas de potencia dispositivos que permiten manipular el sentido de giro de los motorreductores, elementos electrónicos y eléctricos como fuentes de alimentación y cableado para conexión entre las tarjetas.

- En el proyecto se logró realizar una comunicación exitosa entre el computador y el robot, para esto se diseñó una tarjeta de adquisición de datos basada en el microcontrolador 18f 4550 cuya principal función es la recepción de las consignas generadas desde el computador para posteriormente ser procesadas, y de esta forma lograr la manipulación correcta de los actuadores. Estos datos son transmitidos mediante el protocolo de comunicación USB.

- En la realización de este proyecto se logró realizar el diseño y la posterior construcción completa de “base para sistema de entrenamiento quirúrgico (robot Hibou)”, es un robot porta endoscopio que permite realizar pruebas de manipulación desde un joystick, en el interior de un recipiente asemejando al abodomen del paciente.

6.2. Trabajos futuros.

- Diseñar un algoritmo de control que permita realizar el seguimiento de las trayectorias deseadas. Para esto inicialmente se debe realizar la identificación paramétrica del robot, enseguida el controlador debe leer las posiciones articulares del robot mediante los potenciómetros lineales y después transformar las diferentes órdenes de consignas en señales de movimiento que serán ejecutadas por los motorreductores en cada una de las articulaciones.

- Diseñar un entorno virtual que permita realizar prácticas orientadas a los diferentes tipos de procedimientos que se realizan cuando se hace este tipo de intervención, para esto el software debe indicar al usuario paso a paso cómo se debe realizar un determinado procedimiento.

- Implementar otros dispositivos de mando en el sistema para la interacción entre usuario-computador, que es la forma en que el operador humano se comunica con el sistema robótico. Entre las alternativas tenemos: interacción por voz, movimientos de la cabeza utilizando un casco, o dispositivo de mando desde el pie.

Referencias bibliográficas.

[1] V. Muñoz, “Control Cartesiano de un Asistente Robótico para Cirugía Laparoscópica”. Primer Work Shop Español de Robótica, Zaragoza, 2007.

[2] A. Vivas, *Diseño y control de robots industriales: teoría y práctica*. Buenos Aires: Elaleph, 2010.

[3] C. Méndez y V. Torres, “Diseño y Simulación en 3D de un Robot Porta Endoscopio para Cirugía Laparoscópica”, Tesis de Grado de Ingenieria en Automática Industrial, Universidad del Cauca, Colombia, 2010.

[4] S. Shwartz y J. Hunter, *Principios de cirugía,* 7ª ed, Vol. II. México: McGraw Hill Interamericana, 1999.

[5] DIMEDA INSTRUMENTE, “Dimeda Surgical Instruments” http://www.dimeda.de/en/surgical.htm. Consultado: mayo 2013.

[6] E. Valle, “Experiencia en Cirugía Laparoscópica en HEORDRA en el periodo de enero de 2003 a febrero de 2004”, Tesis, Universidad Nacional Autónoma de Nicaragua, León, Nicaragua, 2004.

[7] *Surgical alternatives to hysterectomy in the managment of leiomyomas.* American College of Obstetrican and Gynecologists (ACOG). No. 16, pp. 1-10, 2000.

[8] M. Meinero y G. Meloti, *Cirugía laparoscópica.* Buenos Aires: Editorial Médica Panamericana. S.A. 1994.

[9] Intuitive Surgical, Inc: http://www.intuitivesurgical.com. Consultado: mayo 2012.

[10] Da Vinci. Surgery: http://www.davincisurgery.com/. Consultado abril 2012.

[11] A.J. Madhani “Design of Teleoperated Surgical Instrument for Minimally Invasive Surgery”, PHD. Tesis, Massachusetts Institute of Technology (MIT), United States, 1998.

[12] Robot Zeus: http://en.wikipedia.org/wiki/ZEUS_robotic_surgical_system. Consultado junio 2012.

[13] C. Monserrat, M. Alcañiz, U. Meier, J. Poza, M. Juan, V. Grau, “Simulador para el Entrenamiento en Cirugías Avanzadas”. MedICLab/DSIC, Universidad Politécnica de Valencia, Valencia, España, 2000.

[14] J. García, M. Arias y E. Valencia, “Diseño de Prototipo de Simulador para Entrenamiento en Cirugía Laparoscópica”, Revista de Ingeniería Biomédica. Vol. 5, No. 9. Medellín, Colombia 2011.

[15] SIMBIONIX: Lap Mentor http://www.simbionix.com/LAP_Mentor.html. Consultado junio 2012.

[16] J. Ordóñez y J. Muñoz. “Software Manipulador del Robot Porta Endoscopio (Hibou) para Cirugía Laparoscópica”, Tesis de Grado en Ingeniería Automática Industrial, Universidad del Cauca, Colombia, 2012.

[17] S. Kommu, P. Rimington, C.Anderson, y A.Rane “Initial experience with the Endoassist camera-holding robot in laparoscopic urogical surgery ”, *Journal of robotic surgery,* Vol1, pp.133-137, 2007.

[18] A. Cordoba, G. Ballantyne, “Sistemas quirúrgicos robóticos y telerobóticos para cirugía abdominal”. Revista de Gastroenterología del Perú, Vol. 23, pp. 58-66, 2003.

[19] G. Morel and P. Poignet**,** “Kinematics, position and force control issue in minimally invasive surgery”, *International Conference on Computerized Medical Imaging and Computer Assisted Intervention,* France 2004.

[20] Medsys, “Technology”, *Medsys,* Enero, 2010, [Online], Disponible: http: //www.medsys.be/surgical-robots/lapman/technology.asp. Consultado: Septiembre 2009.

[21] J. Mosso, A. Minor, V. Lara, y E. Maya, “Brazo robótico para sujetar y posicionar laparoscópicos. Primer diseño y construcción en México”, *Academia Mexicana de cirugía*, Vol. 69, No 6, pp. 295-299, 2001.

[22] E. López, “El Protocolo USB“. [Online]. Disponible: http://www.imicro. com/pdf/articulos/usb.pdf. Consultado: junio 2013.

[23] Soporte técnico OEM, ”un paseo por USB 2.0”. [online]. Disponible: http://www.fujitsu.com/downloads/EU/es/soporte/discosduros/UnpaseoporUSB-2.pdf.

[24] Data Sheet PIC 18f 4550 enhanced Flash USB microcontrollers http://ww1.microchip.com/downloads/en/devicedoc/39632c.pdf. Consultado junio 2012.

[25] C. Tosse y E. Lasso. “Dispositivo de Mando Auxiliar Para el Robot Porta endoscopio Hibou”, Tesis de Grado en Ingeniería Automática Industrial, Universidad del Cauca, Colombia, 2012.

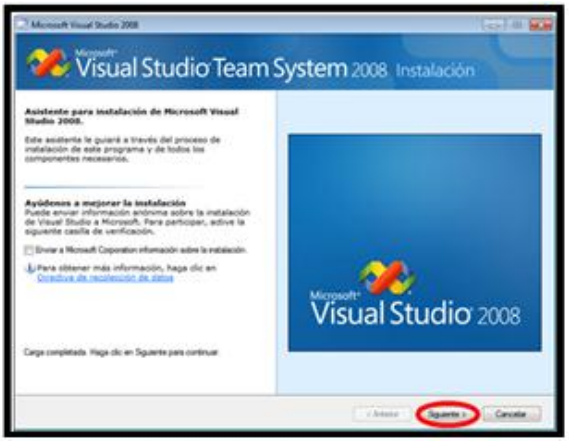
[26] Microsoft Visual Studio. msdn.microsoft.com/es-co/vstudio. Consultado: junio 2013.

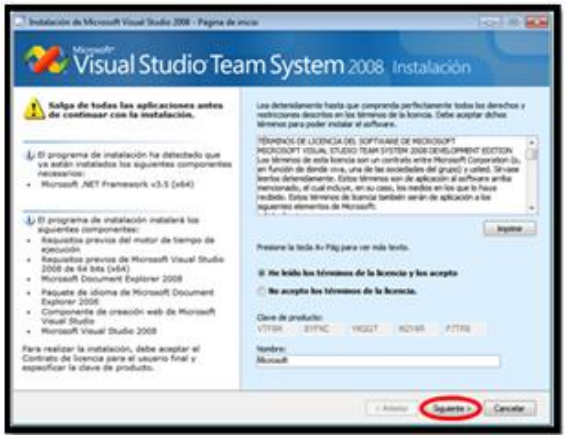
[27] M. Martínez, “Herramienta software para la segmentación y visualización en 3D de hígados reales a partir de imágenes de escaneos 3D no invasivos”, Tesis de Grado en Ingeniería en Telecomunicaciones, Universidad Politécnica de Valencia, España, 2010.

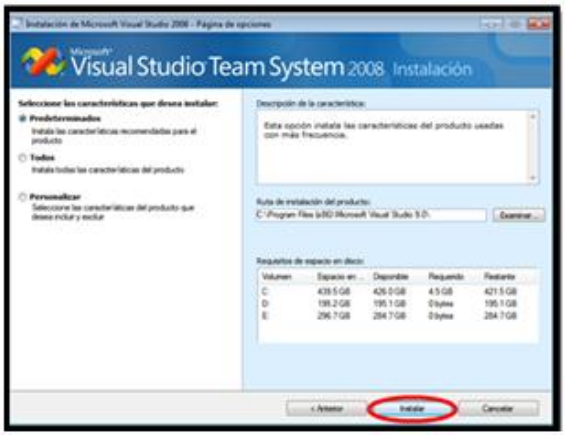
Anexo A. Instalación herramientas software utilizadas.

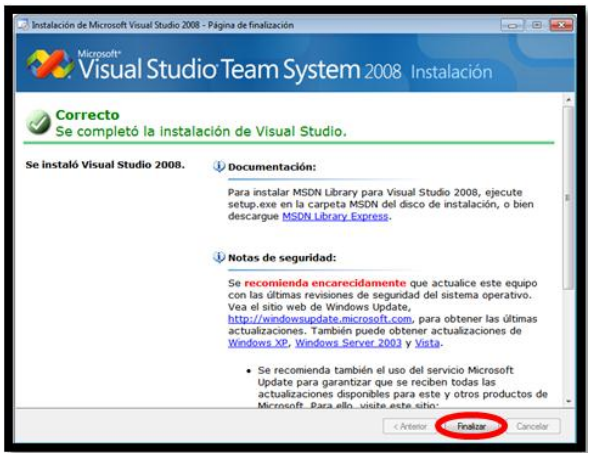
- ## Instalación de Visual Studio 2008

Se instalara esta herramienta con las características que tiene por defecto como lo mostraran las siguientes imágenes.





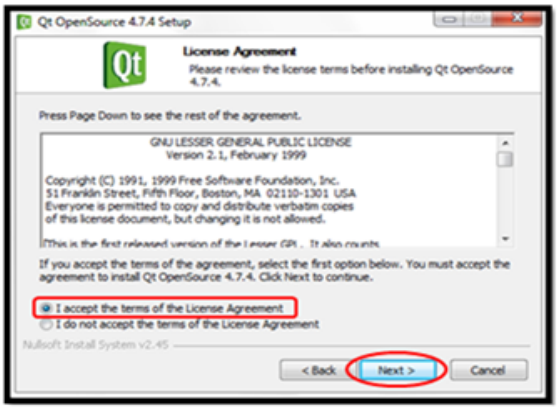


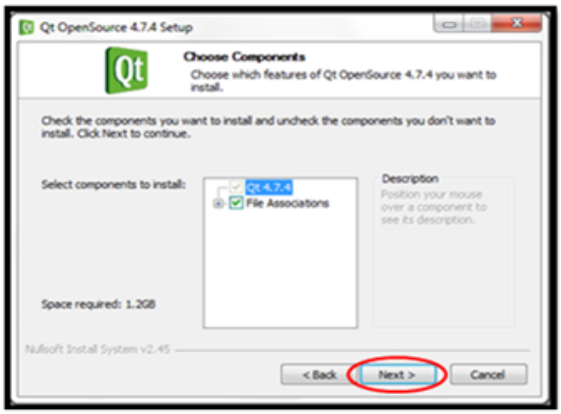


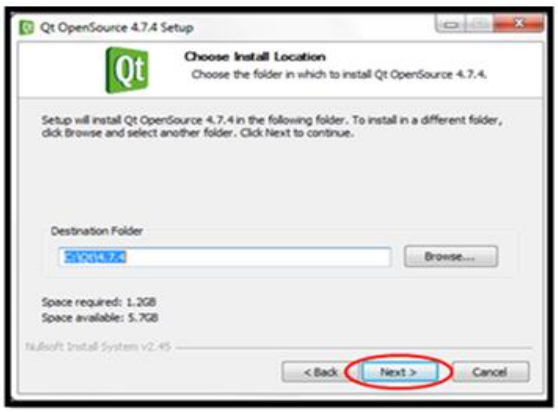
- **Instalación de Qt.**

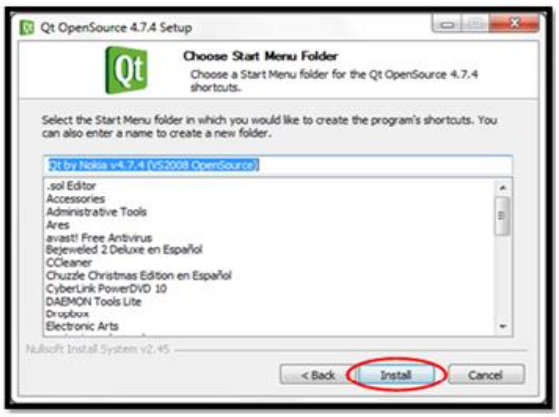
Para el desarrollo de la interfaz se utilizó la versión “qt-win-opensource-4.7.4-vs2008” para generar la ventana secundaria donde se visualizan los dos robots. Descargado desde la página http://qt.nokia.com/downloads. Se instala como veremos a continuación las imágenes.

Marcar la opción “I accept the terms of the License Agreement” y dar click en “Next”.

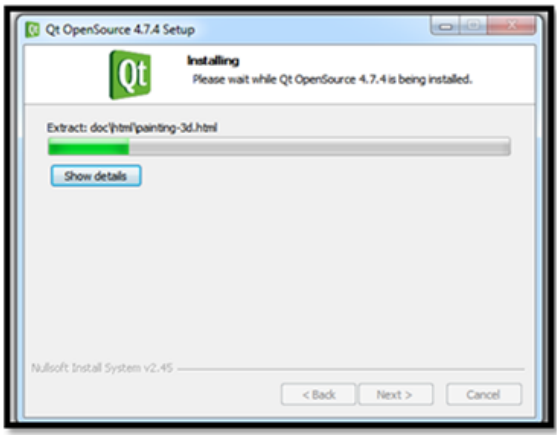


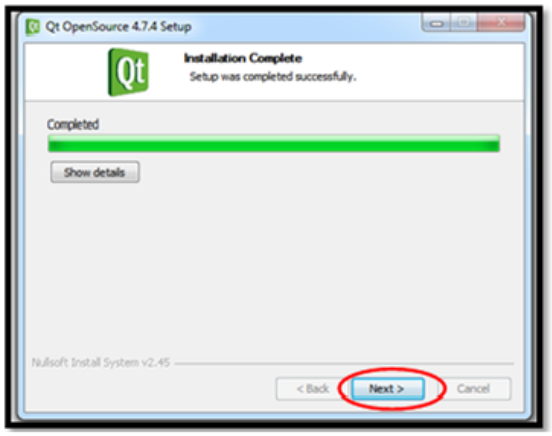




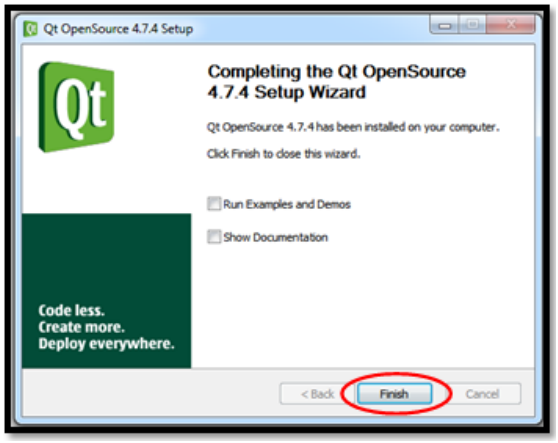


El programa empezara a instalar todas sus características, esperar hasta que se llene toda la barra verde de instalación y aparezca “completed”



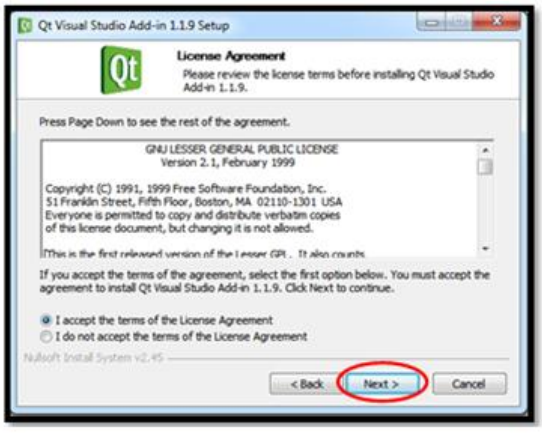


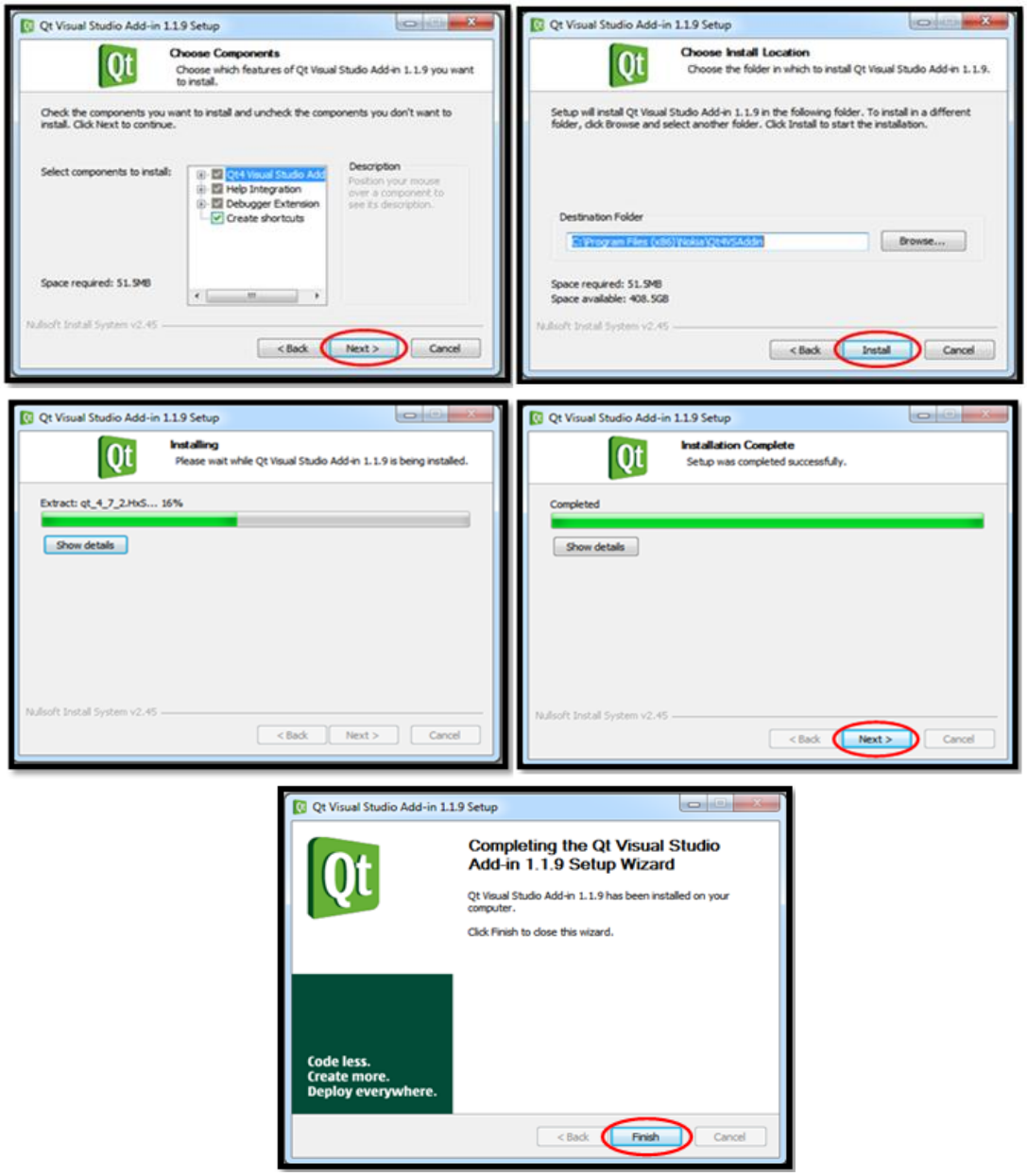
Para no correr ejemplos, demostraciones o nos muestre documentación del programa desmarcamos las opciones “Run Examples and Demos” y “Show Documentation” y damos click en finish.



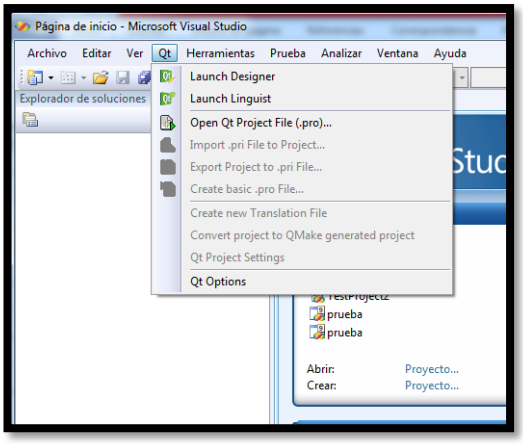
- **Integración de Qt con Visual Studio 2008.**

Para la integración de estas dos herramientas hacemos uso del fichero “qt-vs-addin-1.1.9”, el cual se puede descargar a través de la página http://qt.nokia.com/downloads/visual-studio-add-in



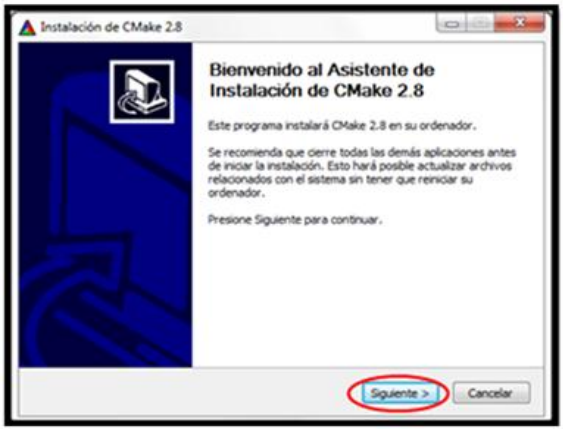


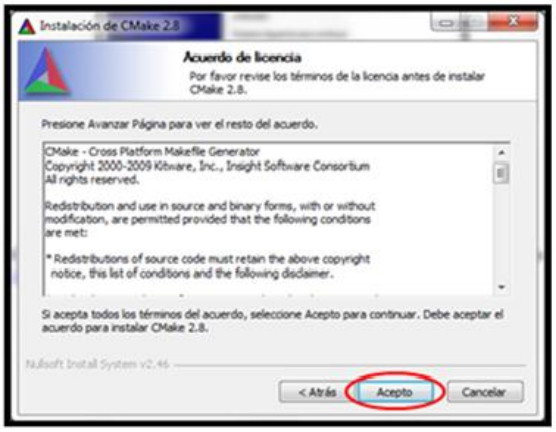
Para comprobar la integración de estas dos herramientas abrimos Visual Studio 2008 y verificamos que aparece un nuevo menú llamado Qt.

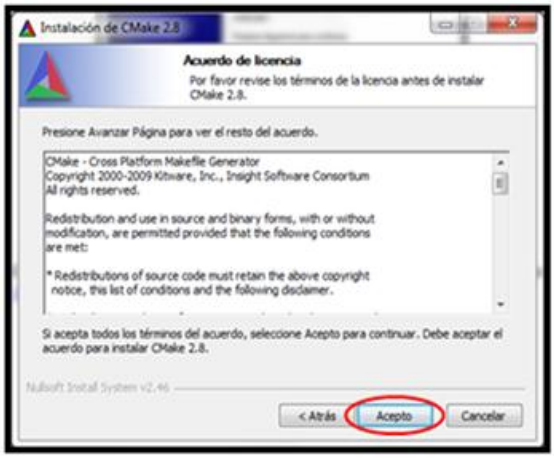


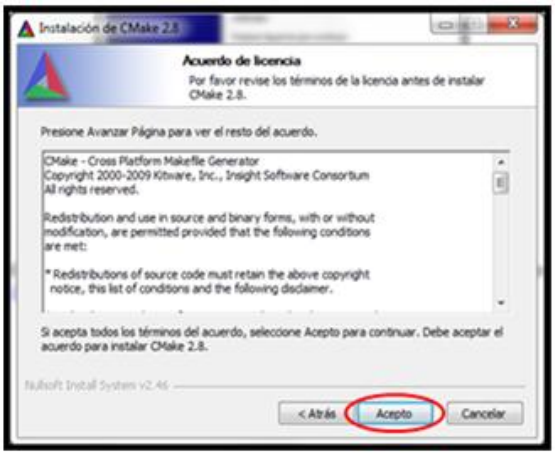
- **Instalación de Cmake.**

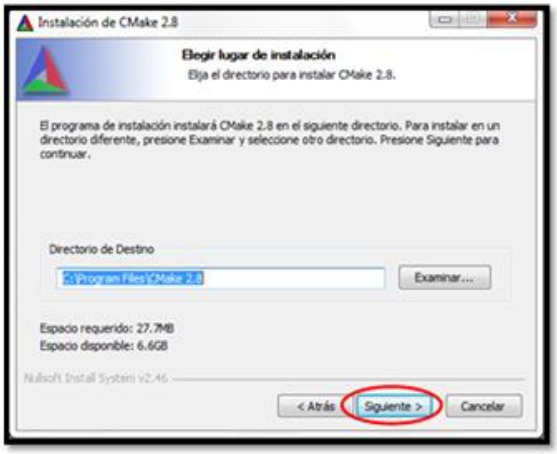
Es un compilador libre el cual podemos descargar de la pagina web http://www.cmake.org/cmake/resources/software.html. Instalaremos la versión CMake 2.8 con las opciones predeterminadas como se muestra en las figuras.







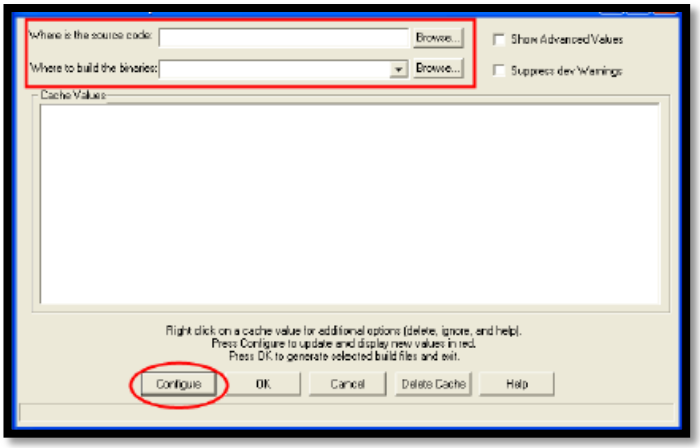




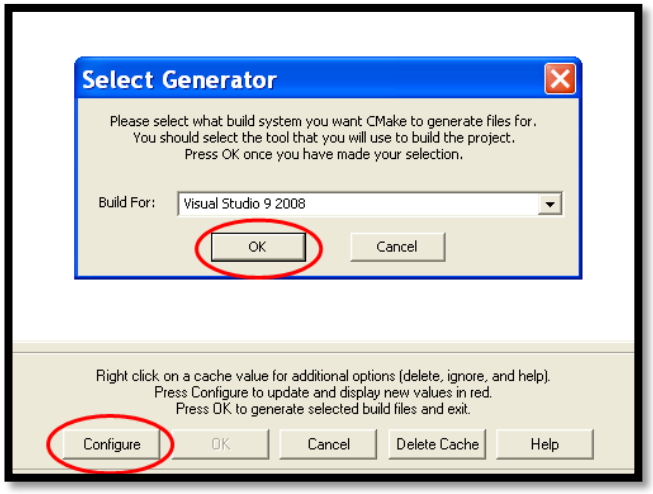
- **Instalación de VTK.**

VTK es una librería libre por lo tanto podemos descargarla de su pagina web http://www.vtk.org/VTK/resources/software.html. Se descargará un archivo .zip el cual hay que descomprimirlo.

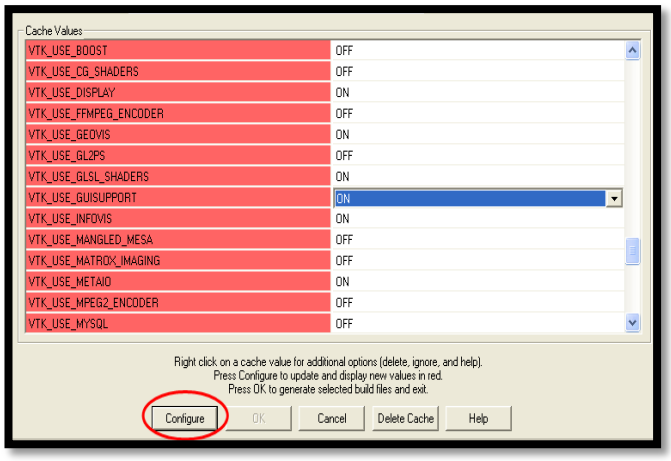
Ejecutamos la aplicación CMake para compilar la librería VTK, seguidamente en la ventana de Cmake donde indica “Where is the source code” habrá que poner el directorio donde descomprimimos la carpeta de VTK. Luego en “Where to build the binaries” se ubica la dirección de una nueva carpeta creada por el usuario donde quedará la librería VTK compilada que creará Cmake. Luego de ello dar click en Configure.



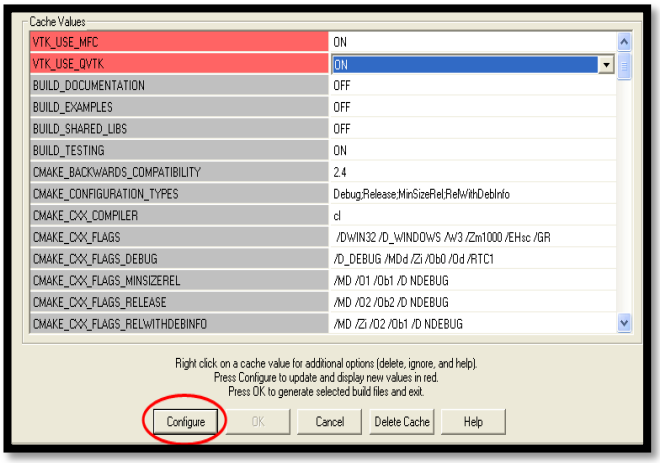
A continuación pedirá seleccionar la plataforma en la cual se hará la pre compilación de la librería. Se elige la opción Visual Studio 2008 y se da click de nuevo en Configure.



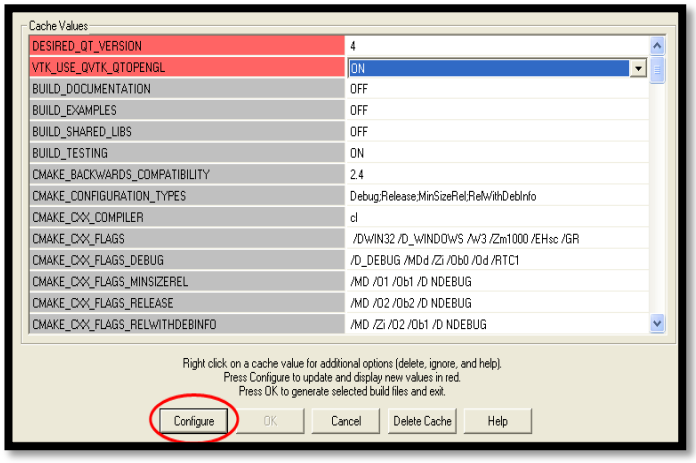
Aparecerán varias variables en rojo dentro de la ventana de compilación de las cuales debemos cambiar la variable “VTK_USE_GUISUPPORT” de “OFF” a “ON” y dar de nuevo click en Configure.



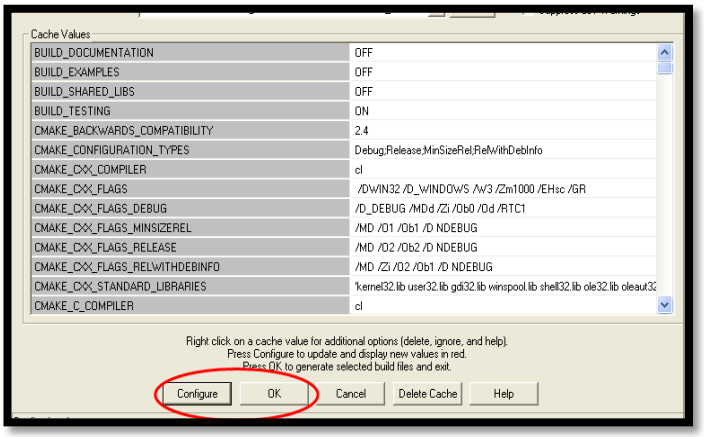
Aparecerán nuevamente algunas variables en rojo esta vez debemos cambiar “VTK_USE_QVTK” de “OFF” a “ON” y dar click en Configure.



Una vez más se repetirá este proceso, ahora se cambiara la variable “VTK_USE_QVTK_QTOPENGL” de “OFF” a “ON” y se dará click en Configure.



Por ultimo se vuelve a hacer click en configure y cuando las variables aparezcan en gris dar click en OK.



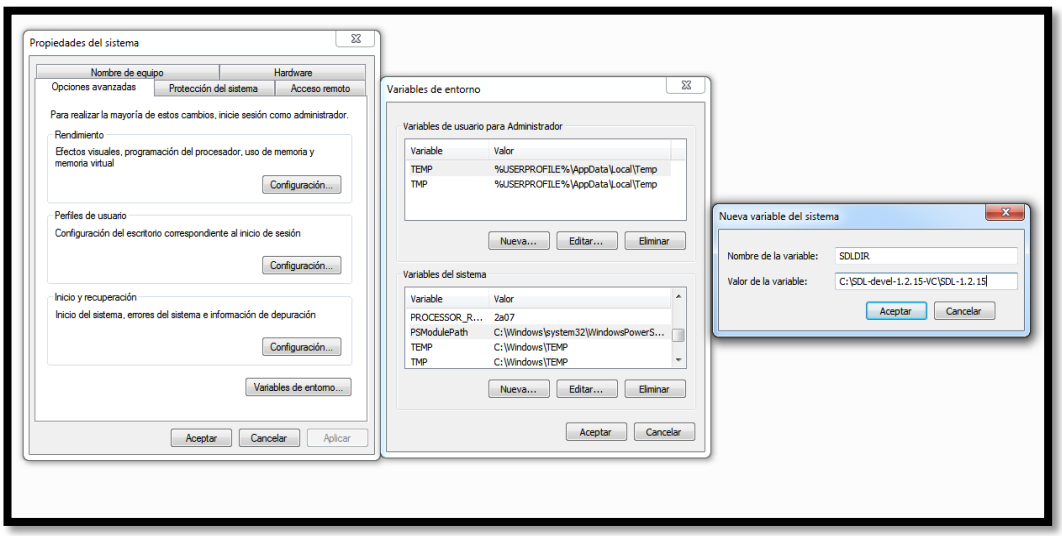
La librería ya esta compilada, ir a la nueva carpeta que se creo para compilar VTK, es la carpeta que se ubico en “Where to build the binaries” anteriormente. Ahí aparecerá un proyecto llamado “Vtk.sln” dar doble click sobre el y se abrirá con Visual Studio 2008.

Dentro de la ventana de Visual Studio ubicarse en “ALL_BUILD”, dar click izquierdo y luego click derecho en “Generar”. Este proceso tardará mas de una hora una vez terminada la compilación estas librerías se podrán usar en cualquier programa creado en Visual Studio 2008.

- **Instalación de SDL.**

SDL es una librería de uso libre la cual podemos descargar de la página http://www.libsdl.org/download-1.2.php, se busca “Development Libraries” y descargamos el archivo “SDL-devel-1.2.15-VC.zip (Visual C++)” como se muestra en la figura. Una vez descargada descomprimimos esta carpeta en el disco C.

En el equipo vamos a: Panel de control >> Seguridad y sistema >> sistema, se da click izquierdo en configuración del sistema, la cual desplegara una venta de propiedades del sistema. Dar click izquierdo en el botón Variables de entorno, se desplegará una ventana de variables de entorno. En variables de entorno dar click derecho en Nueva, en esta nueva ventana se agrega nuestra nueva variable de entorno, como nombre poner SDLDIR y en el valor de la variable se escribe C:\SDL-devel-1.2.15-VC\SDL-1.2.15. Como se muestra en la siguiente figura.



Anexo B. Manual de usuario del sistema.

El siguiente manual de usuario va dirigido a las personas que continúen con la consecución del proyecto, para finalmente obtener un sistema de entrenamiento quirúrgico operativo orientado a cirugía de laparoscopia.

Introducción.

Este manual pretende describir los pasos que se deben seguir para poner en funcionamiento el sistema, el manual enumera tanto los pasos que se deben seguir en el computador como los que se deben aplicar al hardware. Al finalizar el usuario debe comprender cuál es el orden establecido para el encendido del dispositivo.

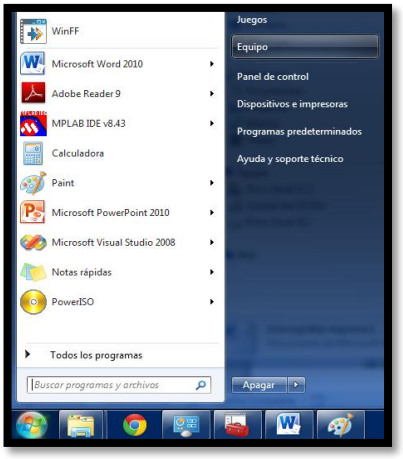
Paso 1.

Encender el computador y conectar la tarjeta de adquisición de datos mediante el cable de conexión USB.

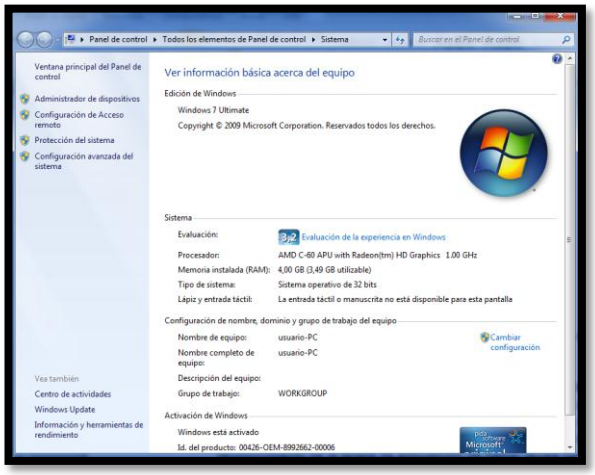
Paso 2.

- Verificar que la conexión se ha establecido, para esto vamos al menú **inicio**.

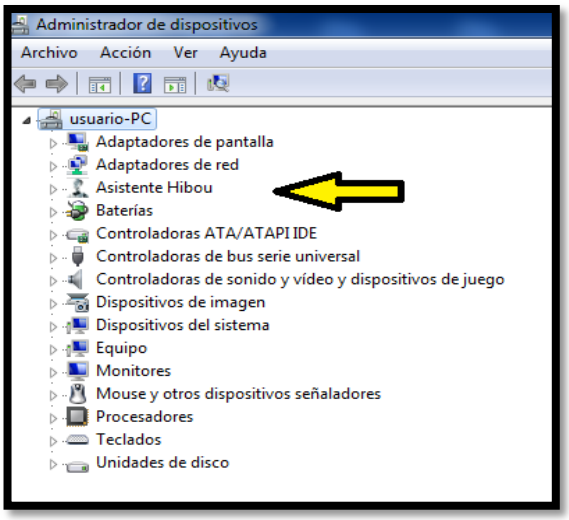
- Buscamos el botón **equipo** oprimimos click derecho sobre este y buscamos la opción **propiedades.**



- Enseguida se muestra una ventana para ver información básica del computador, una vez aquí buscamos en la parte izquierda la opción **administración de dispositivos** se da click izquierdo sobre esta opción**.**



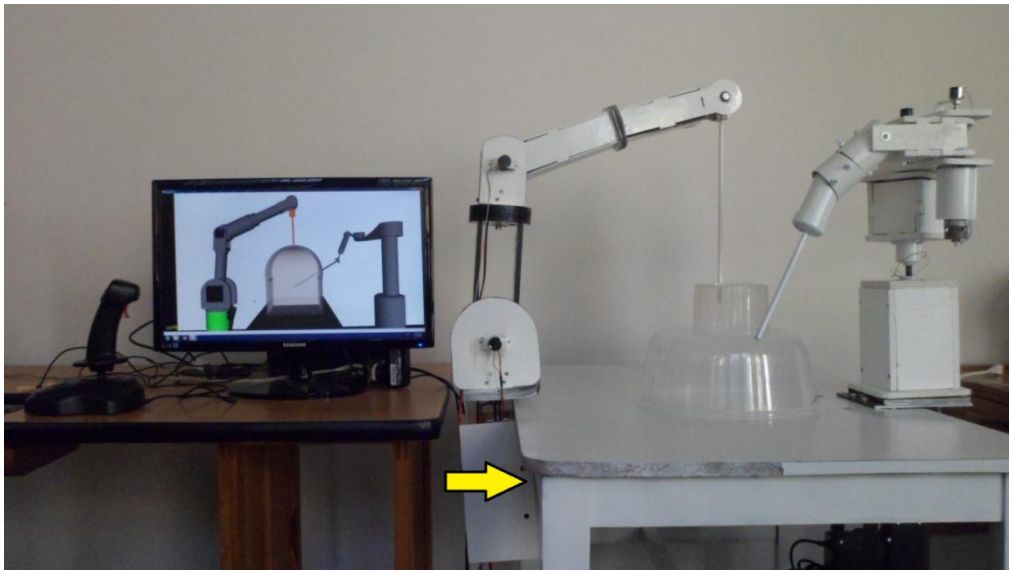
- Aparece una ventana en la cual debe aparecer el dispositivo con el nombre de **Asistente Hibou.**



- Si el dispositivo no aparece quiere decir que existe un conflicto en el reconocimiento de la tarjeta de adquisición de datos (revisar capítulo 4).

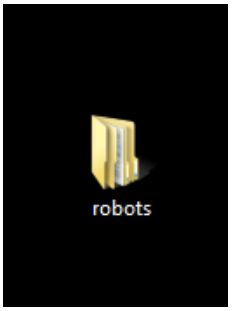
Paso 3.

Realizar el encendido de la instrumentación del robot, esto se hace a través de un switche que está ubicado en la parte inferior izquierda de la mesa, a continuación se muestra su ubicación.

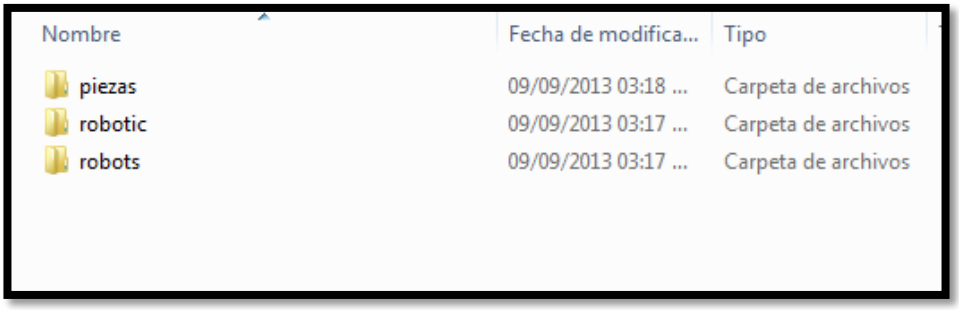


Paso 4.

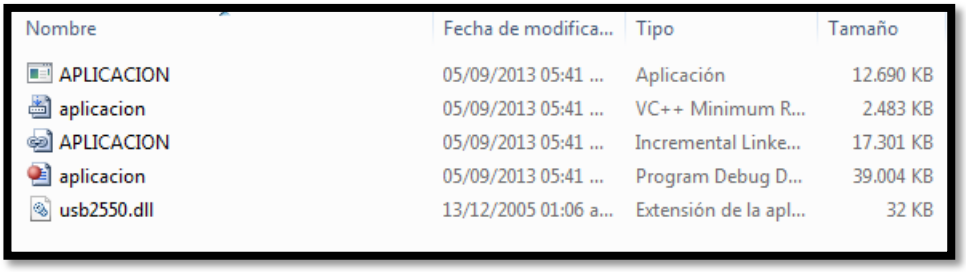
Buscar la carpeta donde se encuentra el archivo ejecutable de la interfaz. Esta carpeta se encuentra en el escritorio con el nombre **robots**.



Aquí se encuentran tres carpetas, seleccionamos la carpeta con el nombre **robotic**.

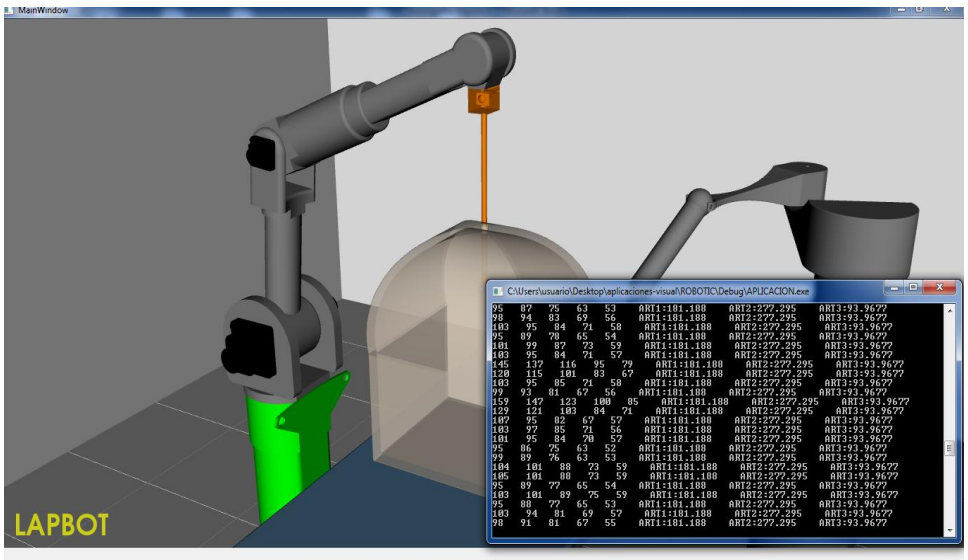


Una vez aquí se busca la carpeta **Debug** y dentro de esta se encuentra el archivo ejecutable que contiene una extensión **.exe**, una vez ubicados aquí conocemos la ubicación del archivo de ejecución**.**



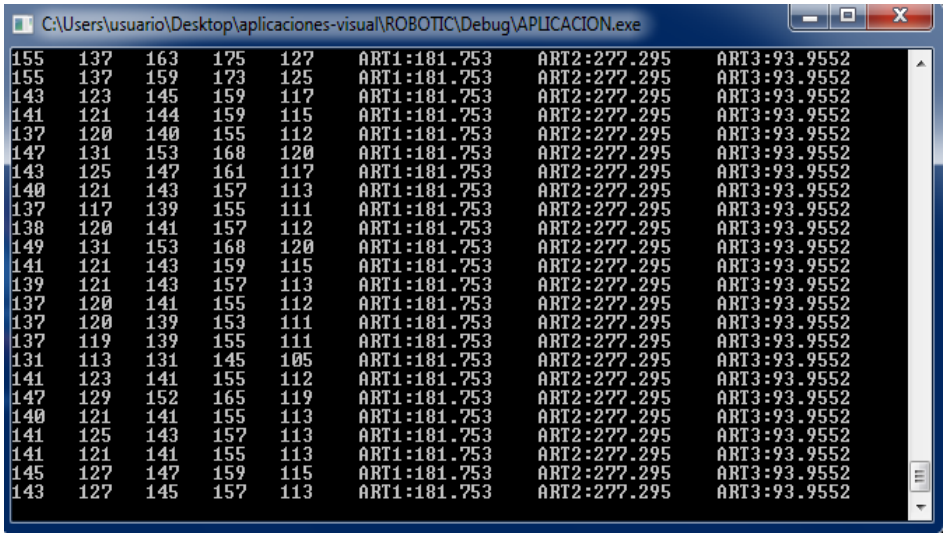
Paso 5.

A continuación se procede a realizar la ejecución del archivo ejecutable, para esto damos click derecho en el archivo y buscamos la opción ejecutar como administrador, enseguida aparecerán dos pantallas.



La pantalla grande es donde se muestra la interfaz gráfica en la cual están los dos robots en la sala de cirugía, la segunda es una ventana de color negro que muestra en el lado derecho las consignas generadas desde el

computador y en el lado izquierdo la posición articular leída por los potenciómetros lineales.



Se recomienda desactivar el antivirus que se tenga activado en el momento, esto debido a que este puede bloquear el sistema impidiendo su correcto funcionamiento.

Paso 6

Finalmente ya se puede realizar el proceso de manipulación del robot a través del joystick.

Anexo C. Diagrama circuital y de conexión del sistema.

A continuación se muestran las etiquetas y los esquemas para realizar conexiones cableadas si se requiere hacer modificaciones en el hardware del sistema.

- **Tarjeta de adquisición de datos.**

ADC1	Entrada analógica 1 proveniente de la salida 1 de la tarjeta de potenciómetros.
ADC2	Entrada analógica 2 proveniente de la salida 2 de la tarjeta de potenciómetros.
ADC3	Entrada analógica 3 proveniente de la salida 3 de la tarjeta de potenciómetros.
VREF+	Voltaje de polarización fijo proveniente de la tarjeta de potenciómetros.
OUT-B0	Salida para verificación de conexión USB (fallo de conexion)
OUT-B1	Salida para verificación de conexión USB (conexión exitosa)
OUT-B2	Salida digital para manipulación de motor reductor 1
OUT-B3	Salida digital para manipulación de motor reductor 1
OUT-B4	Salida digital para manipulación de motor reductor 2
OUT-B5	Salida digital para manipulación de motor reductor 2
OUT-B6	Salida digital para manipulación de motor reductor 3
OUT-B7	Salida digital para manipulación de moto r reductor 3
GND-1	Conexión a tierra común en la parte digital

- **Tarjeta de optocopladores.**

OUT-B2	entrada digital para manipulación de motor reductor 1
OUT-B3	entrada digital para manipulación de motor reductor 1
OUT-B4	entrada digital para manipulación de motor reductor 2
OUT-B5	entrada digital para manipulación de motor reductor 2
OUT-B6	entrada digital para manipulación de motor reductor 3
OUT-B7	entrada digital para manipulación de motor reductor 3
PWM1	Entrada de PWM 1 proveniente de la tarjeta (actuador 1)
PWM2	Entrada de PWM 2 proveniente de la tarjeta (actuador 2)
PWM3	Entrada de PWM 3 proveniente de la tarjeta (actuador 3)
GND-1	Conexión a tierra común en la parte digital
A1	Conexión a 5 voltios de la fuente de alimentación.
PH1-IN1	Salida digital 1 para manipulación de motorreductor 1
PH1-IN2	Salida digital 2 para manipulación de motorreductor 1
PH2-IN1	Salida digital 3 para manipulación de motorreductor 2
PH2-IN2	Salida digital 4 para manipulación de motorreductor 2
PH3-DIR	Salida digital 5 para manipulación de motorreductor 3
PH3-RESET	Salida digital 6 para manipulación de motorreductor 3
PWM1	Salida de PWM 1 proveniente de la tarjeta (actuador 1)
PWM2	Salida de PWM 2 proveniente de la tarjeta (actuador 2)
PWM3	Salida de PWM 3 proveniente de la tarjeta (actuador 3)
A2	Conexión a tierra común en la etapa de potencia

- **Circuito puente H1.**

A1	Conexión a 5 voltios de la fuente de alimentación.
A2	Conexión a tierra común en la etapa de potencia
PH1-IN1	Entrada digital 1 para manipulación de motorreductor 1
PH1-IN2	Entrada digital 2 para manipulación de motorreductor 1
PWM 1	Entrada de PWM 1 proveniente de la tarjeta (actuador 1)
OUT 1	Salida de manipulación motorreductor 1
OUT 2	Salida de manipulación motorreductor 1

- **Circuito puente H2.**

A1	Conexión a 5 voltios de la fuente de alimentación.
A2	Conexión a tierra común en la etapa de potencia
PH2-IN1	Entrada digital 1 para manipulación de motorreductor 2
PH2-IN2	Entrada digital 2 para manipulación de motorreductor 2
PWM 2	Entrada de PWM 2 proveniente de la tarjeta (actuador 2)
OUT 1	Salida de manipulación motorreductor 2
OUT 2	Salida de manipulación motorreductor 2

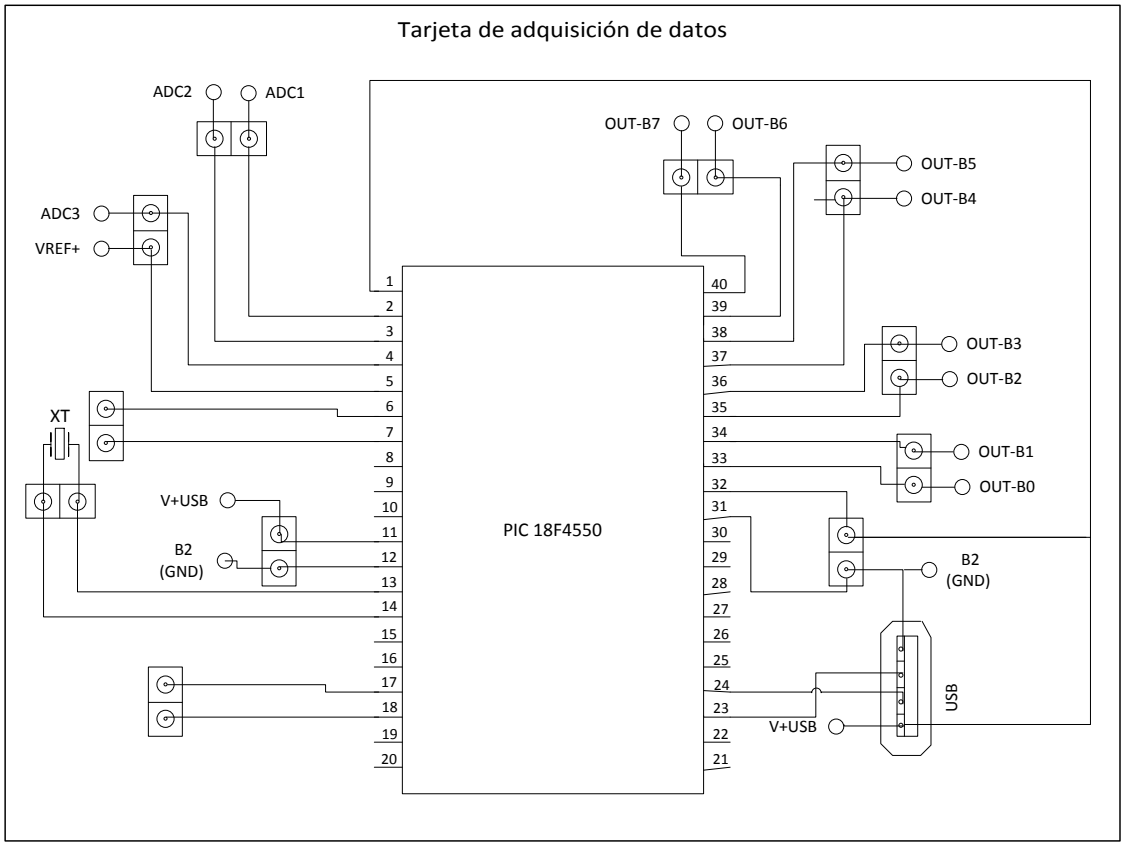
- **Circuito puente H3.**

A1	Conexión a 5 voltios de la fuente de alimentación.
A2	Conexión a tierra común en la etapa de potencia
DIR	Entrada digital 1 indica sentido de giro del motor
RESET	Entrada digital 2 bit de reset negado
PWM 3	Entrada de PWM 3 proveniente de la tarjeta (actuador 3)
OUT A	Salida de manipulación motorreductor 1
OUT B	Salida de manipulación motorreductor 2

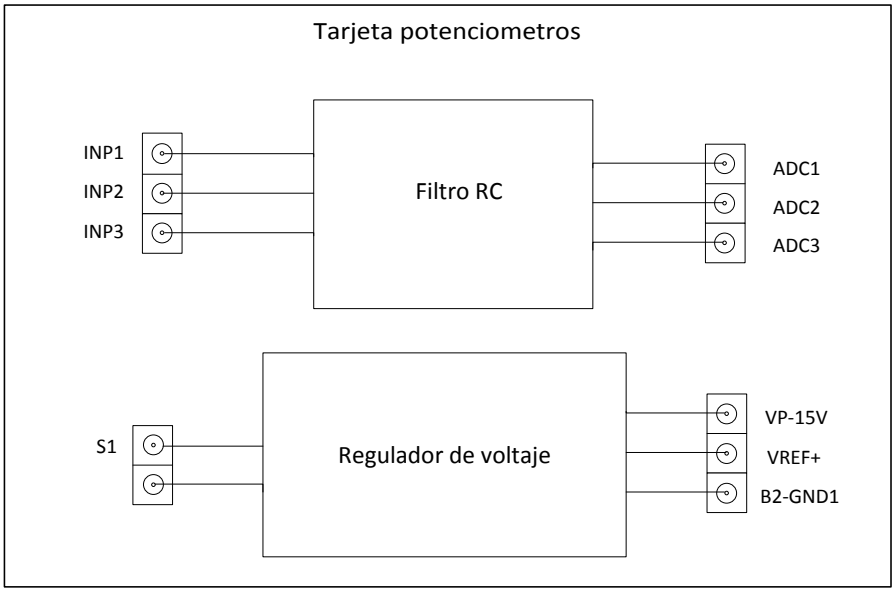
- **Tarjeta de potenciómetros.**

INP1	Entrada proveniente del potenciómetro 1
INP2	Entrada proveniente del potenciómetro 2
INP3	Entrada proveniente del potenciómetro 3
ADC1	Salida potenciómetro 1
ADC2	Salida potenciómetro 2
ADC3	Salida potenciómetro 3
S1	Entrada proveniente de la fuente 2
VP-15V	Voltaje de polarización para potenciómetros
VREF+	Voltaje de polarización fijo de 5V
B2-GND1	Conexión a tierra común en la parte digital

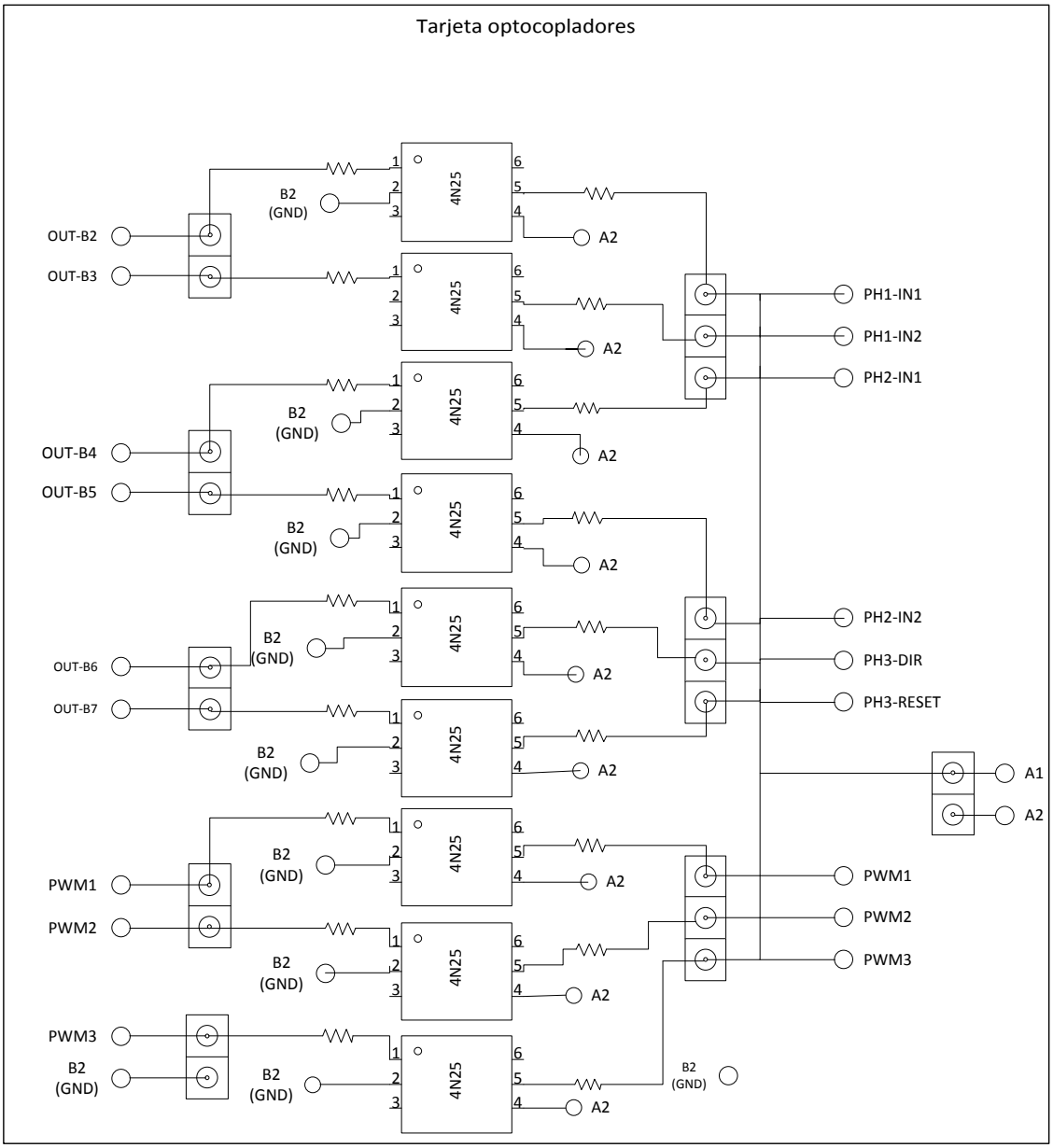
- **Esquema circuital tarjeta de adquisición de datos.**



- **Esquema circuital de la tarjeta de potenciómetros.**



- **Esquema circuital tarjeta optocopladores.**



- **Esquema circuital circuitos puente h.**