Desarrollo de un prototipo de brazo robótico

Marco Ney Rojas Jiménez

e-mail: xecklet@gmail.com

Resumen: *En el campo de la robótica se desarrollan muchos trabajos relacionados a recrear el funcionamiento de diversas partes del cuerpo humano, una de las cuales es el brazo humano. Un brazo robótico es un robot que cuenta con las mismas capacidades de movimiento del brazo humano, los cuales abarcan amplios y diversos aspectos. En este contexto, se presenta el planeamiento, desarrollo y resultados de la implementación de un brazo robótico controlado por el sensor Leap Motion, como parte de un proyecto de investigación del Instituto tecnológico de Costa Rica.*

PALABRAS CLAVE: Leap Motion, Arduino, Servo Motor, Control de versiones, API, Micro controlador, Unity, Brazo Robótico.

1. Introducción

En el mundo en el que actualmente nos encontramos, los robots se encuentran en una variedad de formas y tamaños, diseñadas para hacer una extraordinaria gama de tareas. Típicamente cuando se habla de robots, las personas se imaginan un robot humanoide, como los que aparecen en películas de ciencia ficción, pero la realidad es que hoy en día, los robots comerciales e industriales son de utilizados para realizar trabajos de forma más barata, con mayor precisión y fiabilidad que los humanos.

Cuando se habla de robots, a lo que realmente se refiere es a la máquina que hace lo que normalmente se esperaría que los seres humanos hicieran. Estas máquinas imitan el funcionamiento del ser humano o al menos ciertas partes del mismo. Por lo que en el presente documento se plantea una problemática relaciona con el uso de robots, especialmente se enfoca en los brazos robóticos, donde se exponen los objetivos del proyecto y posteriormente se presentan los resultados obtenidos del mismo.

2. Planteamiento del problema

2.1. Descripción del problema

En el campo de la robótica se han desarrollado diferentes tipos de robots que ayudan a realizar trabajos pesados como el proceso de manufacturación, la manipulación de materiales radiactivos, la desactivación de bombas y muchas otras áreas. La mayoría de los robots que se utilizan en las áreas mencionadas, corresponden a los robots que tratan de simular el movimiento de un brazo human, los brazos robóticos.

Por lo tanto, en este proyecto se sientan las bases para desarrollar un brazo robótico con diferentes capacidades de movimiento, a un bajo costo. Además, el control principal del brazo será realizado mediante el sensor de movimientos Leap Motion.

2.2. Objetivo general

- Investigar y desarrollar un brazo robótico controlable mediante el uso de sensores gesticulares, basándose en desarrollos similares.

2.3. Objetivos específicos

- Investigar los diferentes modelos de brazos robóticos que existen en el mercado.
- Desarrollar una aplicación en un micro controlador que envié las señales necesarias para controlar los diferentes grados de libertad del brazo robótico.
- Estudiar el API del sensor Leap Motion para obtener señales que controlen un brazo robótico.
- Desarrollar una aplicación de escritorio donde se procesen las señales del Leap Motion.
- Establecer una forma de comunicación entre la aplicación de escritorio y el micro controlador.

3. Marco teórico

3.1. Micro controlador Arduino

En [10] se define un micro controlador como una microcomputadora diseñada para controlar

las operaciones de sistemas embebidos en motores de vehículos, robots, dispositivos médicos complejos, transmisores de radio móviles y otros dispositivos. Un micro controlador típico incluye un procesador, memoria y periféricos.

El Arduino es una herramienta para hacer que un computador pueda detectar y controlar ciertos componentes del mundo físico, con respecto a un computador normal.

El Arduino se puede utilizar para desarrollar objetos interactivos, tomando las entradas de una variedad de interruptores o sensores y el control de una variedad de luces, motores, y otras salidas físicas. Los proyectos de Arduino pueden ser independientes, o pueden comunicarse con el software que se ejecuta en su computadora.

Figura 1: Imagen de un Arduino Uno, tomado de [1]

3.2. Servo motores

En un dispositivo que esta compuesto por un circuito que se encarga de controlar la posición (en grados) de los engranajes utilizando un reductor de velocidad y un multiplicador de fuerza. Los servo motores estándar tienen un radio de giro de 0 a 180 grados, los cuales pueden ser modificados para obtener un radio completo de giro, es decir, 360 grados.

La posición del servo motor es configurada dependiendo de la longitud de un pulso enviado. Los servo motores esperan recibir un pulso aproximadamente de 20 milisegundos. Un pulso menor a 1 milisegundo implica un ángulo de 0 grados, para 1.5 milisegundos el ángulo es de 90 grados y para mayores a 2 milisegundos es de 180 grados.

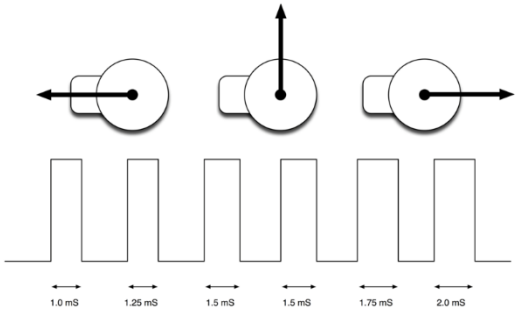

Figura 2: Posición de los servo motores según el pulso enviado, tomado de [8]

3.3. Brazo Robótico

Un brazo robótico básicamente corresponde con una seria de enlaces que se encuentran conectados de tal manera que los servo motores puedan ser utilizados para controlar cada articulación de forma independiente. El ordenador de control está programado para controlar los diversos motores en el robot de una manera que le permite realizar tareas específicas.

El brazo robótico puede ser diseñado de diferentes de maneras, tamaños y formas, lo cual es crítico en la arquitectura del robot dependiendo del trabajo que vaya a realizar. El brazo es el encargado de posicionar el efector final en la posición correcta de trabajo, para que posteriormente ejecute su acción. Si el diseño del brazo es demasiado grande o pequeño, este posicionamiento puede o no ser posible. La mayoría de los brazos robóticos son construidos con seis grados de libertad, exactamente la misma cantidad de movimiento de un brazo humano. En la siguiente figura se muestra las relaciones de grados de libertad de un brazo robótico con el brazo humano.

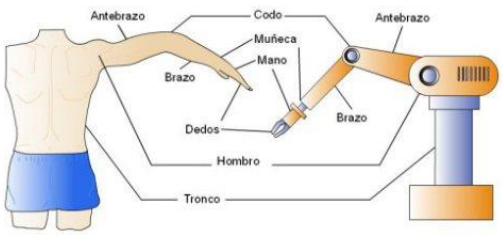

Figura 3: Brazo robótico con seis grados de libertad tomado de [5]

3.4. Leap Motion

El controlador Leap Motion es un pequeño dispositivo USB, que rastrea el movimiento de las manos y los dedos como entrada. Este dispositivo utiliza dos cámaras monocromáticas infrarrojas y tres LEDs infrarrojos. La distancia de visión del dispositivo corresponde con aproximadamente de 25 a 900 milímetros.

Los LEDs generan un patrón de luces infrarrojas y las cámaras generan alrededor de trecientos cuadros por segundo de datos reflejados, los cuales son enviados por el cable USB al computador, donde son analizados por el software controlador del Leap Motion. Este software utiliza algoritmos complejos para obtener la posición en tres dimensiones de los datos, al comparándolos con imágenes en dos dimensiones generados por las dos cámaras. Este dispositivo cuenta con precio muy competitivo, $99.99.

Algunas de las características que ofrece el dispositivo son:

- Rastreo de los 10 dedos hasta un centésimo milímetro. Es dramáticamente más sensible que la tecnología de control de movimiento existente.
- Tiene un campo de visión de 150° y una componente Z de profundidad. Lo que permite mover las manos en 3D, al igual que el mundo real.

Leap Motion cuenta con una comunidad de desarrolladores bastantes grande. El SDK para el desarrollo de aplicaciones en Windows, MAC y Linux, es totalmente gratuito. Además, es importante rescatar el hecho de que la cantidad de leguajes soportados es bastante amplio, entre las opciones oficiales se encuentran: Unity, Unreal Engine, JavaScript, C++, Vuo.org y Cylon.js, y las no oficiales se encuentra Processing (biblioteca desarrollada por terceros).

Figura 4: Imagen del funcionamiento del Leap Motion, tomado de [9]

3.5. Unity

Unity es una plataforma de desarrollo flexible y poderosa para el desarrollo de juegos en 3D y 2D, que además cuenta con la capacidad de integración de diferentes dispositivos de hardware y programas de software, por lo que lo convierte en una herramienta perfecta para el desarrollo de esta aplicación.

Una de las principales características es la capacidad de moverse libremente entre veintiuna plataformas de desarrollo, por lo que el desarrollo de una aplicación o juego para Windows, puede ser fácilmente exportable a Macintosh.

Unity cuenta con un ambiente de desarrollo integrado muy intuitivo, donde se le presenta una sección de la pantalla donde se muestra el espacio de trabajo en 3d y otra donde se muestra lo que realmente aparecerá en pantalla al ejecutar la aplicación.

Entre los lenguajes de programación que se pueden utilizar se encuentran: JavaScript, C# y Boo, los cuales cuentan con una fuerte documentación y un foro de desarrolladores.

Algunas de las herramientas que posee el IDE es la creación de animación, uso de físicas, monitoreo de uso del procesador y memoria, control de versiones, entre otras. Además cuenta con una tienda donde se puede adquirir nuevas y mejores herramientas.

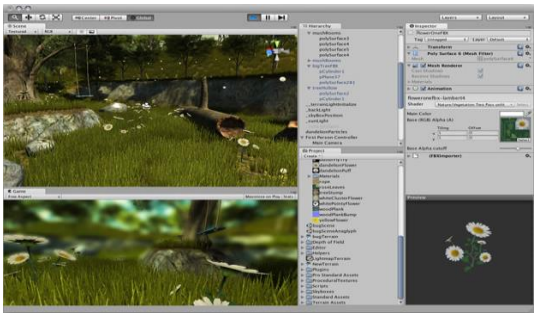

Figura 5. IDE de Unity donde se muestran las diferentes secciones, tomado de [6]

4. Solución planteada

4.1. Construcción

Como se planteó en el plan de proyecto, antes de comenzar la construcción del brazo robótico se realizó una investigación de las opciones de brazos robóticos que existían en el mercado, a continuación se presentan las tres mejores opciones:

Tabla 1: Modelos de brazo robótico evaluado

Nombre	Kit de brazo robot AL5A	Brazo robótico impreso en 3D	Brazo robótico de acrílico
Imagen			
Dificultad de construcción	Ninguna	Alta	Alta
Costo	$278	$125	$154

Como el proyecto consiste en la creación de un brazo robótico de bajo costo, la primera opción queda descartada. Por lo que las mejores opciones son las dos últimas.

La opción final de brazo robótico corresponde con la segunda opción, obtenido de (Ancastrog, 2012). Esto se debe a que en primer lugar es la opción más económica, pero además de esto, cuando se manda a fabricar piezas en acrílico se utilizan láminas de dicho material, las cuales son cortadas posteriormente con la maquinaria necesaria, por lo que tienen un ancho predefinido por pieza, mientras que con una impresora en 3D no se presenta el mismo problema. Cuando se realizar la impresión en 3d esta puede crecer en las tres direcciones y no se ve limitada a una lámina de acrílico.

Una vez elegido el modelo de brazo robótico, se procedió con la impresión. En ese momento la única impresora disponible al alcance del estudiante, correspondía a la que se encontraba en el laboratorio de la escuela de diseño, por lo que se tuvo que hablar con el encargado.

Luego de coordinar con el encargado, se llegó a la conclusión de que para utilizar la impresora en 3D, era necesario comprar un cartucho, debido a que la cantidad que tenían podía ser insuficiente y no podía quedarse sin material, pues los estudiantes de diseño lo podrían necesitar. Para poder adquirir este material, se coordinó con el profesor asesor para realizar el trámite. El cartucho tuvo un costo de 50 dólares.

La impresión de las partes del robot se realizó en la impresora Cubify Pro de segunda generación y tardó aproximadamente tres semanas, debido a que la impresión era bastante lenta y la superficie de impresión muy reducida.

Una parte clave de la construcción del brazo robótico, corresponde con el movimiento de las articulaciones, las cuales son realizadas utilizando servomotores. En la siguiente tabla, se muestra el costo de total de cada servomotor:

Tabla 2: Costo de los servo motores

Nombre	Servo Tower Pro MG995	Servo Futaba S3003	MicroServo MG90S
Imagen			
Cantidad	1	3	4
Costo	$10,99	$38,64	$18,59

El costo total de la compra de los servomotores 68.22 dólares. Al comprar los servomotores por Amazon, estos duraron aproximadamente cuatro semanas en llegar, lo que generó un importante retraso.

Los últimos materiales por conseguir, fueron los tornillos. Esto fue un problema, debido a que los tornillos que se necesitaba principalmente eran de 2x15 mm, los cuales no se pueden conseguir en ferreterías. Al hablar con varias personas, se llegó a la conclusión de que los tornillos debían están en la Casa del Tornillo, donde se especializan en la construcción de los mismos. Al final el brazo ocupo aproximadamente 18 tornillos de 2x6 mm, 8 tornillos de 3x25 mm y 36 tornillos de 2x15 mm.

Una vez con todos los materiales, la construcción del brazo robótico pudo dar inicio. El brazo fue completado al cabo de dos semanas. Para comprobar que todas las articulaciones se movían correctamente, se programó una sub rutina para el movimiento de las articulaciones.

Uno de los problemas que se presentó con las pruebas, fue que la pinza no cerraba correctamente debido a la calidad de la impresión.

4.2. Propuestas para la obtención de los datos gestual

Instalar el ambiente de desarrollo del Leap Motion no fue ningún problema, descargar e instalar el SDK es lo que se necesita para hacer funcionar el dispositivo.

Por otro lado, la propuesta inicial de la obtención de datos para el manejo del brazo robótico consistió en utilizar ambas manos para mover las diferentes articulaciones del brazo.

Utilizando la posición de la mano derecha se iba a controlar un total de tres articulaciones. Recordemos que el espacio está compuesto por tres componentes: X, Y y Z, las cuales son representadas como izquierda-derecha, adelante-atrás y arriba-abajo. En este caso la componente X movería la base, la Z movería el hombro y la Y movería el codo. Para la mano izquierda, se utilizó la orientación de la mano para hacer rotar la muñeca y cuando esta era cerrada, cerrar la pinza.

La propuesta anterior corresponde con el movimiento inicialmente propuesto, más adelante se menciona la propuesta final utilizada.

4.3. Micro controlador escogido

El micro controlador escogido fue el Arduino, que aunque cuenta con capacidades básicas de recursos, su nivel de procesamiento es el ideal y cuenta con la cantidad necesario de salidas digitales. Otra de los puntos por el cual se escogió, fue el hecho de cuanta con una gran cantidad de ejemplos y no era necesario adquirir uno, pues ya se contaba con uno. Por otro lado, no era la primera vez que el estudiante trabaja con un Arduino.

Otros micro controladores considerados fueron la Beaglebone y una FPGA. Ninguno de las opciones anteriores se encontraba disponible, por lo que se tenía que mandar a traer y en el caso de la FPGA, ronda los 250 dólares.

4.4. Alimentación y control de los servomotores

En primer lugar cabe destacar que el Arduino por sí solo, no es capaz de poner a funcionar ocho servomotores al mismo tiempo, su máxima capacidad es de dos o tres servos. En resumen, la potencia del Arduino era un problema, principalmente por el hecho de que su máxima corriente es de 500 mA, que es la corriente que provee el puerto USB.

La solución al problema de potencia fue el uso de una fuente de alimentación externa, que consistió en una apartador de corriente alterna, que convierte el voltaje de un tomacorriente a 5V, con una corriente máxima de 1.5 A. Al realizar esto, el problema de la potencia fue resuelto.

Por otro lado, para controlar cada servo motor de manera independiente, se utilizaron 8 pines digitales donde se define el ángulo de cada servo motor de manera independiente. En la siguiente imagen se muestran las conexiones:

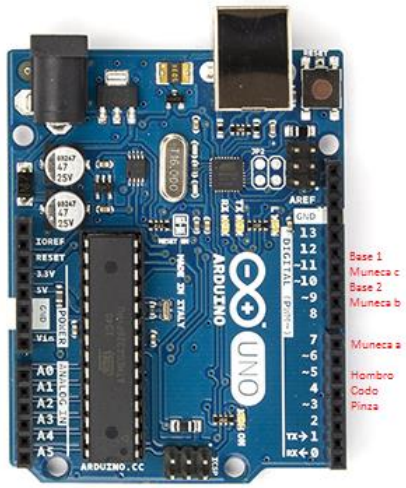

Figura 6.Conexión de los pines del Arduino con partes específicas del brazo robótico.

El algoritmo que cambia el ángulo es bastante sencillo, este utiliza dos arreglos: el primero de ellos posee el ángulo actual de cada servo motor y el segundo contiene la posición a la cual debe de ir. Para llegar hasta la posición deseada, se compara la posición final con la inicial y dependiendo del resultado se suma o se resta un grado al servo motor en cada ciclo de reloj.

4.5. Comunicación por puerto serial

Al realizar el método de la comunicación del puerto serial, se presentó un problema importante en el rendimiento del brazo robótico.

El método inicial lee en cada ciclo la entrada serial, si encontraba algún bit, comenzaba a leerlo bit por bit, hasta recuperar de forma exitosa la cadena de caracteres. Luego tomada la cadena, la decodificaba y actualizaba la posición a la cual los servo motores debían ir.

La lectura del puerto serial siempre era exitosa, al igual que la decodificación y la actualización de las posiciones, el problema radicaba en la consulta del puerto serial. Cuando el método preguntaba si existía algo en dicho puerto, se retrasaba en aproximadamente 500 ms, lo cual hacia que el brazo robótico se moviera de forma pausada y lenta.

La solución a este problema fue utilizar un método que funcione con eventos. A diferencia del método anterior, este comienza a leer el puerto serial cada vez que llegaba un nuevo mensaje, por lo que no se generaba ningún retraso en el ciclo principal del Arduino.

4.6. Lenguaje y programación de la aplicación

Como se observó en el marco teórico, la cantidad de lenguajes de programación de Leap Motion, es variada.

La decisión final fue programar en Unity 4.6f, utilizando C#. La razón principal, es debido a que posee una librería muy poderosa. Dicha librería cuenta con la capacidad de *renderizar* el modelo de las manos en tiempo real. Además, el estudiante ya tenía conocimiento previo de Unity.

Al ser un IDE multiplataforma, se puede aprovechar esta cualidad para exportarlo a diferentes sistemas operativos.

Una vez seleccionado el lenguaje de programación, se procedió al desarrollo de la aplicación. Entre las características que debía de tener se encontraba: transformar los datos obtenidos en ángulos, enviar los datos al puerto serial, mostrar el modelo de las manos y la posición de los servo motores en tiempo real.

La parte más complicada de la aplicación, consistió en la transformación de los datos de posición y rotación en ángulos. La solución implementada fue tomar dos puntos fijos en el espacio e implementar una función lineal con base en esos puntos.

4.7. Decisiones finales tomadas

Para terminar con la solución planteada, hay que mencionar las decisiones que se tuvieron que tomar en el desarrollo del proyecto.

La primera de ellas fue limitar el movimiento de los servomotores, esto para que evitar que chocaran contra la placa de pruebas, que se encuentra detrás del brazo robótico, estropeando el circuito controlador. Las

articulaciones que fueron limitadas fueron: la base, el hombro y el codo, por lo que sus rangos de rotaciones son de 45 a 135 grados, 60 a 180 grados y 60 a 180 grados, respectivamente. Además, debido a la poca rotación de la pinza, el servo se limitó de 100 a 150 grados.

La segunda decisión tomada fue el hecho de que la base fue diseña para utilizar dos servo motores funcionando de forma opuesta, pero cuando se trató de mover ambos dispositivos al mismo tiempo, entre ellos se oponían al movimiento y movían la base en forma extraña. La solución final fue dejar un servomotor estático y dejar que el otro mueva toda la estructura.

La última decisión importante fue la de limitar la cantidad de articulaciones que se rotan, a solamente cuatro. Esto se debe, a que cuando se estaban realizando las pruebas de movimiento, el Leap Motion, muchas veces perdía la posición exacta de las manos; y al realizar dos rotaciones en la mano izquierda, se afectaban mutuamente, como resultado se generaban movimientos inesperados. La solución final fue mover dos articulaciones por mano:

La mano derecha mueve la base con la posición X y se encarga de abrir o cerrar la pinza cuando cierra o abre la mano respectivamente.

La mano izquierda mueve el hombro con la posición Z y controla la inclinación de la muñeca del brazo robótico, con la inclinación de la mano

Al controlar el brazo robótico de esta forma, fue más fácil su movimiento. Además, como ciertas articulaciones fueron limitadas a una posición estática, se puede dar la posibilidad que esa articulación no esté funcionando o simplemente esta estático en el código.

Para evitar esas posibles interrogantes, se desarrolló una nueva forma de movimiento de cada articulación por separado utilizando las letras del teclado. Su funcionamiento consiste aumentar o disminuir el ángulo utilizando las siguientes teclas:

1. Cadera: Aumentar "a" y disminuir "z"
2. Hombro: Aumentar "s" y disminuir "x"
3. Codo: Aumentar "d" y disminuir "c"
4. Rotación hombro: Aumentar "f" y disminuir "v"
5. Inclinación Muñeca: Aumentar "g" y disminuir "b"
6. Rotación Muñeca: Aumentar "h" y disminuir "n"
7. Pinza: Aumentar "m" y disminuir "j"

5. Resultados obtenidos

Al utilizar la impresora en 3D como método de construcción de las partes del brazo robótico, se logró un muy buen acabado con una resistencia apropiada para el movimiento de las articulaciones. Aunque algunas de las partes impresas presentaban cierta deformación, esto no generó ningún problema. En la siguiente figura se puede apreciar el resultado final.

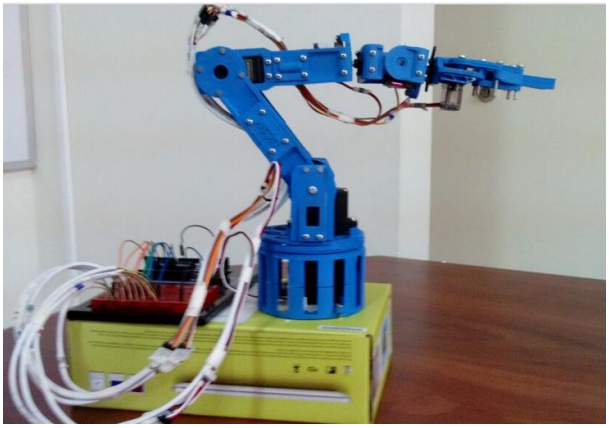

Figura 7. Brazo robótico terminado.

Por otro lado, es claro que se logró alcanzar el objetivo de construir un brazo robótico de bajo costo, pues el material de impresión costo $50, los servo motores $68 y con el Arduino ronda los $24.95, lo cual genera un costo total de aproximadamente de $143. Este precio es muy competitivo si es comparado con la primera opción de brazo robótico planteada, el kit AL5A que costaba $278.

El brazo robótico posee un muy buen tiempo de respuesta, por lo que al utilizarlo no se siente mucho retraso entre los movimientos

realizados por las manos y el resultado final de la posición.

Con las mejoras que se implementaron al limitar la cantidad de articulaciones móviles, el control del brazo se facilitó, pero aun así, es difícil controlarlo con precisión, pues tener las manos en el aire es cansado y difícil mantenerlas inmóviles.

Al utilizar las flechas para mover las articulaciones por separado se comprueba que todas las articulaciones se encuentran funcionando a la perfección. Además, se evidencia un mayor control de cada articulación con respecto al uso del Leap Motion.

La aplicación muestra de forma apropiada el ángulo que posee cada servo motor en forma de barras, lo cual hace intuitiva su comprensión. La conversión de los datos es muy fluida y no presenta saltos importantes entre los movimientos de posición de la mano y la posición del brazo robótico. Además, la aplicación muestra cuando pudo abrir el puerto serial de forma correcta y cuando no, y detecta la conexión del Leap Motion, en caso de que no se haya conectado.

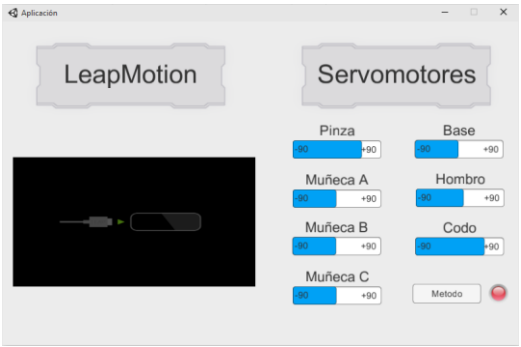

Figura 8. Aplicación de escritorio encargada de obtener datos del Leap Motion y enviarlos al puerto serial

La mayor parte del tiempo, el Leap Motion reconoce en su totalidad la posición exacta de ambas manos, pero otras veces simplemente pierde la orientación exacta, ocasionando errores de movimiento. Una mejora futura es el uso de un sensor más estable.

El micro controlador Arduino, si bien se sabe que es una placa que se encuentra muy limitada por sus capacidades, en la mayoría de las aplicaciones que requiere un componente de hardware, se convierte en un poderoso aliado.

6. Conclusiones

- El principal problema planteado en el desarrollo del proyecto se logró, la construcción de un brazo robótico de bajo costo.
- Las expectativas de velocidad de movimiento ante cualquier cambio en la posición de las manos, fueron cumplidas debido a la estupenda fluidez que presenta el brazo robótico.
- Aunque el sensor del Leap Motion muchas veces es muy preciso al calcular la posición espacial de las manos, en varias ocasiones se pierde dependiendo del movimiento de las mismas.
- 8Unity es una poderosa herramienta que se puede utilizar en una gran cantidad de aplicaciones y no únicamente en juegos.
- Aunque el Arduino pueda parecer una placa limitada por sus capacidades de procesamiento, es el perfecto aliado para el desarrollo de muchos proyectos.

7. Mejoras futuras

- La principal mejora futura esta relaciona con el movimiento del brazo robótico. En robótica existe el concepto de cinemática de transformación inversa, que en términos simples consiste en un proceso que se realiza para obtener la rotación de las articulaciones con base en la posición XYZ de un punto en el espacio.
- Otra mejora futura consiste en el uso de un sensor diferente, como por ejemplo el Kinect v2 que utilizaría el esqueleto de una persona como entrada.

8. Bibliografía

Ancastrog. (10 de Septiembre de 2012). *Robotic arm with 6 DOF*. Obtenido de Thingiverse: http://www.thingiverse.com/thing:30163

Arduino. (s.f.). *Arduino UNO Rev3*. Obtenido de Arduino: http://store.arduino.cc/product/A000066

Arduino. (s.f.). *What is Arduino?* Obtenido de Arduino: http://www.arduino.cc/en/pmwiki.php?n=Guide/Introduction

Desconocido. (s.f.). *Technology*. Obtenido de Studyclix: https://www.studyclix.ie/Content/Uploads/2fc4fda1-4221-40a4-afc1-9ff110ea462b.pdf

Han, J., & Gold, N. (s.f.). *Lessons Learned in Exploring the Leap Motion Sensor for Gesture-based Instrument Design*. Obtenido de New Interfaces for Musical Expression: http://nime2014.org/proceedings/papers/485_paper.pdf

Hernández, J. (26 de Agosto de 2013). *ESTRUCTURA DE LOS ROBOTS*. Obtenido de roboticajh: https://roboticajh.wordpress.com/category/clasificacion/estructura-en-general/

HIGGINS, T. (20 de Marzo de 2009). *UNITY 2.5 FOR MAC AND WINDOWS NOW AVAILABLE!* Obtenido de http://blogs.unity3d.com/2009/03/20/unity-25-for-mac-and-windows-now-available/

KUMAR, R., ANAND, S., & GAUTAM, R. (s.f.). *ROBOTIC ARM* . Recuperado el 01 de Junio de 2015, de Student's Gymkhana IIT Kanpur: http://students.iitk.ac.in/roboclub/old/data/projects/summer11/arm.pdf

Monk, s. (04 de Mayo de 2015). *Servo Motors*. Obtenido de Adafruit: https://learn.adafruit.com/controlling-a-servo-with-a-beaglebone-black/servo-motors

Motion, L. (s.f.). *Leap Motion 3D Motion Controller*. Obtenido de Robot Shop: http://www.robotshop.com/en/leap-motion-3d-motion-controller.html

Rouse, M. (Marzo de 2012). *Microcontroller*. Obtenido de WhatIs.com: http://whatis.techtarget.com/definition/microcontroller