# Diseño y construcción de un animatronic controlado por KINECT

A. Cifuentes [1], E.F. Garzón [2],
L.A. Rodríguez [3], D. Lancheros [4]

(1) acifuentes00@unisalle.edu.co
(2) gedward44@unisalle.edu.co
(3) rluis33@unisalle.edu.co
(4) dlancheros@unisalle.edu.co

Facultad de Ingeniería en Automatización
Universidad de La Salle

# Diseño y construcción de un animatronic controlado por KINECT

*Palabras clave:*
***Animatronic; Kinect; visión artificial; esqueletización.***

**RESUMEN**

El juego es la manera más natural que tiene el ser humano de aprender, ejercitarse y compartir socialmente. En el caso de los niños, desarrollar nuevas estrategias pedagógicas que cautiven sus sentidos genera grandes desafíos para los pedagogos y este reto se magnifica cuando los niños por condiciones múltiples o por traumas secundarios de alguna enfermedad no desarrollan la fuerza necesaria en los músculos de las extremidades superiores o inferiores, por tan motivo este proyecto tiene como principal objetivo desarrollar una herramienta innovadora, tecnológica y de bajo costo, que de los medios necesarios para la rehabilitación de dichas extremidades, esperando que su proceso de recuperación sea lo más satisfactorio posible. Este proyecto da a conocer una interfaz humano-máquina controlado mediante un Kinect, donde sus algoritmos permiten controlar puerto serial de dos motores, los cuales están ubicados estratégicamente en una máscara diseñada con elementos de fácil acceso con el fin de realizar movimientos precisos de ojos y mandíbula. En conclusión el prototipo final ofrece una interfaz interactiva y dinámica de fácil uso y a bajo costo. [1] [2]

## I. INTRODUCCIÓN

La recuperación fisiológica en la mayoría de los casos toma un tiempo determinado con un cierto grado dedicación, esto traducido a términos monetarios puede llegar representar costos excesivos, consecuente a ello la accesibilidad solo se centraría para aquellas personas que cuentan con los recursos necesarios, es por esto que nace la necesidad de tener una herramienta de bajo costo con el cual se puedan abarcar gran variedad de ejercicios que faciliten las terapias y puedan ser llevados a cabo en cualquier lugar, de manera versátil y eficiente, dinámica, y entretenida que aporte en el progreso de recuperación del paciente. [3]

El presente artículo describe el desarrollo de la interfaz de reconocimiento de movimientos y su implementación en movimientos en una máscara con actuadores mecánicos. En el desarrollo del documento podemos observar el progreso para llevar a cabo la construcción y su implementación. La primera fase contempla el diseño de la máscara y la segunda la conexión entre el sensor Kinect y la programación de la IDE Arduino.

## II. ANTECEDENTES

Realizando una revisión de trabajos relacionados con el tratamiento de imágenes en sistemas de calidad, se encuentran trabajos como el de Fontan y Abascal [4] donde el objetivo principal fue explorar el potencial de diferentes estructuras de redes neuronales para realizar la segmentación de imágenes de resonancia magnética (IRM). La investigación permitió validar procesos de segmentación de imágenes mediante la utilización de RNA. [4]

Como avance en la adquisición de imágenes para el procesamiento de movimientos Gutiérrez *et al*. [5], Desarrollo una plataforma con una gran velocidad de operación cuyo algoritmo propone la manipulación de la plataforma mediante la estimación de pose y coordenadas e la mano del operario, como siempre para la adquisición de imágenes del sistema se lleva a cabo mediante el equipo Kinect de Microsoft y cuyo medio de accionamiento es el Arduino desde donde podemos observar su sistema de comunicación como elemento importante en el desarrollo de la aplicación. [5]

Como se puede observar el desarrollo de sistemas que involucren Kinect para el procesamiento de movimientos utilizan algoritmos y programas para el desarrollo final de interfaces humano máquina con gran eficiencia y calidad de diseño. [6]

## III. CONSTRUCCIÓN DEL PROTOTIPO

El diseño del prototipo se lleva a cabo en etapas que abarcan la selección de la estructura soporte de accionamientos, los sistemas mecánicos, la caracterización del personaje y el diseño del programa para el reconocimiento de movimientos mediante el Kinect para transmitirlos mecánicamente.

### *Estructura de soporte*

Estas estructuras normalmente emplean platinas metálicas rectangulares para distribuir el peso de los componentes del robot. Las estructuras del cráneo seleccionados para este prototipo serán analizadas en el siguiente cuadro comparativo:

**TABLA I.** CUADRO COMPARATIVO DE VENTAJAS Y DESVENTAJAS DE DIFERENTES SOPORTES DE CRÁNEO.

| *Soporte de cráneo* | *Ventajas* | *Desventajas* |
|---|---|---|
|  | • Facilidad de acople de elementos<br>• Robustez y solidez | • Difícil elaboración<br>• Alto costo<br>• Estructuras pesadas |
|  | • Facilidad de acople de los actuadores<br>• Costo moderado<br>• Estructura liviana<br>• Forma apropiada del rostro | • Acople vertebral no incluido |
|  | • Bajo costo económico<br>• Forma apropiada del rostro<br>• Estructura liviana | • Estructura maciza |

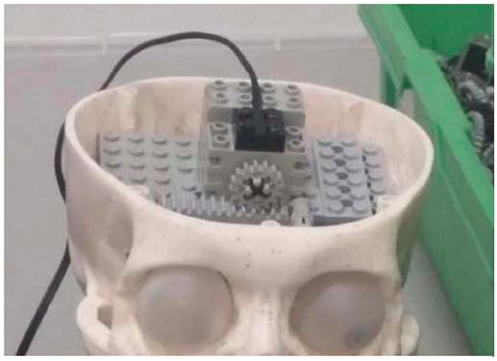
Con respecto a estas tres posibilidades se seleccionó el cráneo didáctico, pues ofrecía mayores ventajas y mejores posibilidades en los resultados a esperar.

*Sistemas mecánicos*

- ***Movimiento de los ojos:***

Materiales: 2 bolas de desodorante pequeñas
1 motor 12 Voltios
1 Cremallera
1 Piñón

**Fig. 1.** Sistema mecánico movimiento de ojos



El sistema se sujeta a una base con fichas de lego dentro la caja cerebral tal como se observa en la figura 1, El motor se acoplada a una cremallera y un piñón, de esta manera si el motor giran en un sentido los ojos se desplazara hacia un lado y cuando el motor gire en el otro sentido los ojos cambiaran de posición.

El resultado que obtuvimos después de la implementación de este sistema se ve reflejado en las Figura 2.

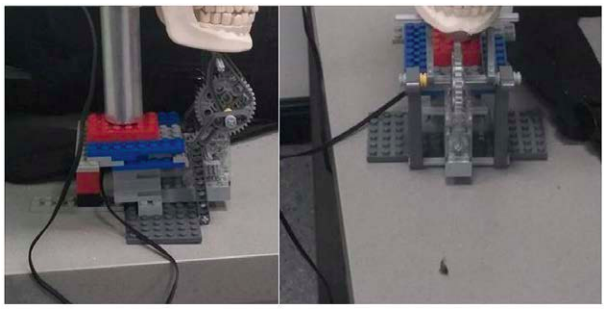
**Fig. 2.** Sistema mecánico movimiento de ojos

- ***Movimiento de la boca:***

Materiales: 1 motor 12 Voltios
1 Caja con piñón sinfín
Vigas y columnas lego.

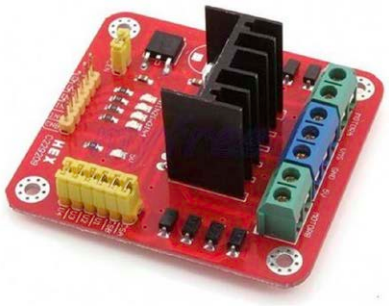
El sistema es accionado por un motor y una caja con un piñón sinfín que convierte un movimiento rotacional en un movimiento traslacional, de esta manera si el motor gira en un sentido la plataforma subirá y si el motor gira en sentido contrario, la plataforma bajara. El diseño mecánico obtenido después de la implementación del sistema se ve reflejado en la Figura 3.

**Fig. 3.** Sistema mecánico movimiento de la boca



Adicionalmente se incluyó un Puente H en la alimentación de los motores con el fin de tener la corriente necesaria y no llegar a quemar circuito impreso del Arduino, el puente H seleccionado es el observado en la Figura 4.

**Fig. 4.** Puente H LM298



*Caracterización del personaje*

El personaje elegido para el prototipo es el cantante Gene Simmons de la banda de Rock KISS, donde los criterios de selección de dicho personaje son:

- Apariencia llamativa (pelo alborotado y actitud alternativa.
- Facilidad en la caracterización: al tener su cara pintada de blanco y negro no requiere gran esfuerzo para copiar dichos rasgos.
- Fama y reconcomiendo.

Al existir la necesidad de una máscara cuyo diseño sea realista se tiene propuesto dos alternativas para la piel:

- Máscara flexible: Elaborada con látex y otros polímeros que le permitían plena movilidad a cada articulación del rostro.

- Porcelanicron: Macilla rígida la cual podía ser pintada muy fácilmente.

Las dos alternativas se pusieron a prueba, la máscara de látex a pesar de su flexibilidad tiende a reducir movimientos debido a su peso con respecto a la fuerza del motor por lo cual se decide usar porcelanicron, su facilidad de moldeo y escaso peso lo hace adecuado para la aplicación.

Ya determinado el material se agrega una capa sobre el cráneo buscando captar y plasmar los mayores detalles del rostro de Gene Simmons sin descuidar los movimientos de los ojos y la boca. Finalmente se aplican los detalles característicos con el fin de hacerlo llamativo para cualquier persona, en la Figura 5 podemos observar la máscara de Kiss con sus detalles de cabello, maquillaje y ojos basados en el personaje.

**Fig. 5.** Caracterización del Animatronics

## IV. SISTEMA DE CONTROL

Se lleva a cabo una investigación previa del estado del arte para determinar el software que será usado en la plataforma de reconocimiento, de la cual se llega a dos alternativas:

- Matlab: Entorno de programación que permite diseñar archivos ejecutables mediante interfaz y texto estructurado y posee comunicación por puerto serial.

- Proccesing: Entorno de programación de código abierto que permite diseñar mediante texto estructurado conexiones usando bloques descargables y puede llevar a cabo múltiples conexiones.

Después de un análisis se determinó que a pesar del amplio conocimiento en MatLab este presentaba conflicto con el hardware implementado por lo que finalmente se optó por realizar un estudio mediante una guía de programación [7] desde la cual se diseñó finalmente la interfaz con Proccesing mediante un bloque de conexión con el Kinect basándonos en el diseño de la Figura 6.

**Fig. 6.** Modelo de control del sistema

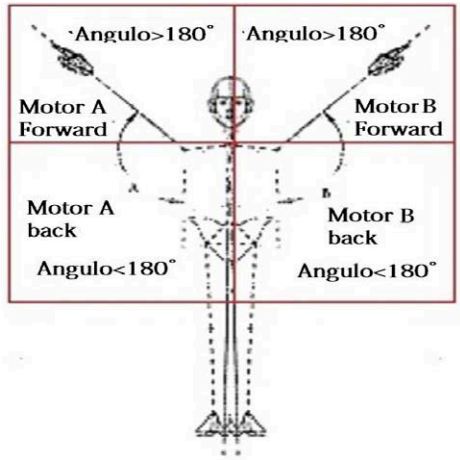
*Funcionalidad del sensor Kinect en el proyecto*

El Kinect es una cámara de profundidad, las cámaras normales recogen la luz que rebota en los objetos en frente de ellos, se asemeja a lo que vemos con nuestros propios ojos. El Kinect, por su parte, registra la distancia de los objetos que se colocan en frente de ella. Se utiliza luz infrarroja para crear una imagen de profundidad, donde están en el espacio.

Como se mencionó antes este proyecto está enfocado en la recuperación de los miembros superiores (brazos). El sensor Kinect capta una imagen donde se mide una separación angular entre los brazos izquierdo y derecho con respecto a los hombros y dependiendo de los rangos de trabajo se

envía un dato para el control del motor, en la figura 7 se observa los rangos de trabajo.

**Fig. 7.** Áreas de trabajo del sensor Kinect




*Limitaciones del Kinect.*

La cámara de profundidad del Kinect tiene algunas limitaciones debido a cómo funciona. Dicha cámara de tiene un rango mínimo de aproximadamente 20 pulgadas. Más cerca de eso, el Kinect no puede calcular con precisión distancias basado en el desplazamiento de los puntos de infrarrojos. Como no puede imaginar una profundidad exacta, el Kinect simplemente trata nada más cerca que este rango mínimo como si tuviera un valor de profundidad de 0, en otras palabras, como si fuera infinitamente lejos. Por lo tanto se muestran partes negras en la profundidad de la imagen, que es más cerca que la distancia mínima de la Kinect, como se muestra en la Figura 8.

**Fig. 8.** Profundidad inaccesible para el Kinect

*Arduino y Proccesing*

El cerebro de nuestro animatronic será un microcontrolador Arduino. El Arduino es una pequeña computadora que puede controlar sensores y actuadores de forma muy simples. Para el proyecto usaremos dos motores DC, uno que controlara el movimiento de los ojos y el otro el de la boca. Entonces vamos a comunicarnos con el Arduino desde Proccesing para que podamos colocar nuestro animatronics basado en los datos que captamos del Kinect.

*Procesamiento de la imagen*

Un píxel es la unidad más pequeña de una imagen. Cada pixel es un solo color sólido, y mediante la organización de muchos píxeles una al lado de la otra, las imágenes digitales puede crear ilusiones como degradados suaves y sutiles matices. Con suficientes píxeles para crear imágenes lo suficientemente suave, usted consigue las imágenes digitales realistas que vemos todos los días. Dependiendo del tipo de imagen, la variedad de colores posibles un píxel puede ser puede variar.

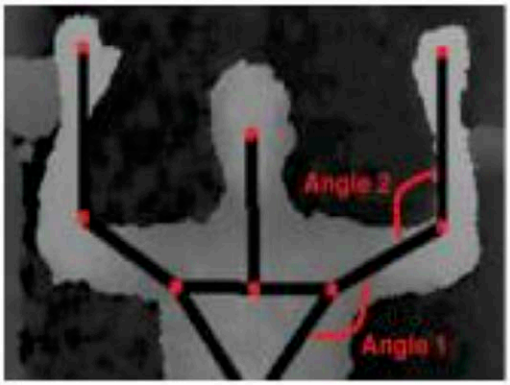
El procesamiento de estos píxeles como números de 8 bits que pueden almacenar los valores de 0 a 255. Dependiendo de su resolución, cada imagen digital tiene miles o millones de píxeles dispuestos en una rejilla. Esta rejilla se compone de columnas y filas. Al igual que con cualquier red, el número de columnas es igual a la anchura de la imagen, y el número de filas es igual a su altura. La imagen capturada por la cámara de profundidad Kinect es de 640 píxeles de ancho por 480 píxeles de alto. Esto significa que cada cuadro que viene adentro de la cámara de profundidad del Kinect tendrá un total de 307.200 píxeles. Y, a diferencia de los de escala de grises y en color imágenes que hemos discutido hasta ahora sobre todo, cada píxel va a hacer una doble función. El número de cada pixel, entre 0 y 255, representará tanto un color de gris y una distancia en el espacio. Podemos utilizar ese mismo número para calcular y analizar los datos obtenido.

*Cálculo de los ángulos de los brazos*

Para llevar a cabo el movimiento cinemático usamos los datos esqueleto del Kinect para

determinar la ubicación del hombro, el codo y el antebrazo de nuestro usuario. Estos ángulos son valores muy importantes ya que serán enviados al Arduino para que pueda coincidir con los servos del cráneo, el cálculo de los ángulos se lleva a cabo mediante SimpleOpenNI el cual se encuentra añadido a la programación realizada en programming. Se desea calcular los ángulos entre cada una de las extremidades y la parte del cuerpo adyacente. Servos van a reproducir los ángulos del hombro del usuario y el codo. Cuando se habla del ángulo del hombro, se quiere decir el ángulo que el hombro crea entre el brazo superior y el torso y del mismo modo cuando se habla del codo se quiere decir el ángulo que crea el brazo y el antebrazo. Al girar el hombro, la posición de su codo y los cambios de mano, pero el ángulo entre ellos sigue siendo el mismo. Del mismo modo al inclinar el tronco hacia la izquierda o la derecha, el ángulo de su hombro no necesariamente cambia a pesar de que su posición.

**Fig. 9.** Ángulos entre las articulaciones que queremos calcular para el movimiento del cráneo.




Lo mismo sucede con el cráneo, la figura 10 muestra el diseño básico de cómo se planeó armar el cráneo el cual cuenta con 90 grados en la boca, y 45 grados de los ojos, cuando se levanta el brazo el ángulo entre el cuerpo y el brazo determina la apertura de la boca mientras el movimiento del antebrazo el ángulo entre el brazo y el antebrazo determina el movimiento de los ojos.

Esta disposición física corresponde precisamente a la estrategia del cálculo del ángulo que se propone para los movimientos del cráneo. Se deben ahora calcular los ángulos correctos entre los miembros con relación al eje correcto para que puedan traducir correctamente a la orientación de cada uno de los motores. Se deben calcular aproximadamente el rango correcto para que el diseño del programa sea correcto al movimiento de los motores.

### *Programación en Proccesing*

A continuación se observa detalladamente como se conviertes las posiciones de las articulaciones en los ángulos exactamente en la orientación correcta que se ajuste a los motores.

El diseño del programa contiene código familiar, en totalidad encontramos llamadas habituales de esqueleto y su seguimiento, funciones como onNewUser para agregar usuarios, onStartPose para iniciar la esqueletización y EndCalibration para ver cuando la calibración ha concluido con éxito. Una vez la calibración se ha completado vamos al acceso de las cuatro articulaciones, la mano derecha, el codo derecho, el hombro derecho y la cadera derecha y así mismo con la parte izquierda.

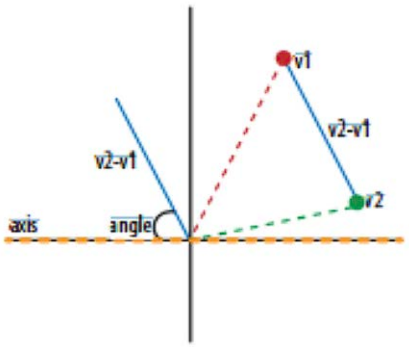
Una vez tenemos todas estas partes conjuntas las debemos convertir de vector 3D a 2D.

No queremos usar el eje Z por lo que descartamos la información del eje Z por lo que creamos un nuevo vector que tome solo los dos componentes necesarios como podemos observar en la figura 9,

Por lo tanto antes de calcular los ángulos, es necesario crear vectores que correspondan a cada uno de los ejes, mediante la resta de vectores, por ejemplo para el caso del hombro se usa en eje vertical del torso, y para el codo se toma el antebrazo y el brazo.

Para facilitar el cálculo de los miembros se usa la función auxiliar angleOf, esta función toma 3 argumentos, los dos vectores que representan los dos extremos de la extremidad cuyo ángulo queremos encontrar y un tercer valor que representa el eje de orientación, la lógica de esta función la podemos observar en la figura 10.

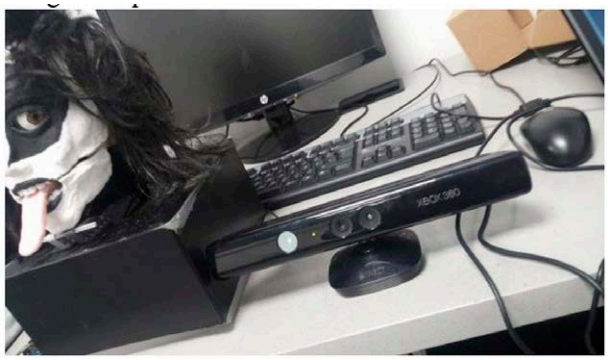
**Fig.10.** Representación de la conversión de vectores.




A continuación usamos la función PVector angleBetween para calcular el ángulo entre las extremidades, este valor se retorna en radianes que se pasaras a grados.

Las últimas líneas complementan la visualización en pantalla. En la figura 11 podemos observar la visualización respecto a la programación implementada.

**Fig 11.** Uso del Kinect en el proyecto



Para evitar problemas con respecto a los movimientos de los motores es posible usar sentencias if que hagan que nuestros movimientos no afecten el sistema mecánico del cráneo cuando sus motores llegan a sus límites y deseamos frenar, aunque es posible modificar programación para usar un método mejor.

### *Conexión Proccesing-Arduino*

El primer se debe establecer la conexión entre el Proccesing y nuestra placa Arduino, para ellos tenemos que importar la biblioteca de serie y declarar nuestro objeto de serie:

```
import processing.serial.*;
Serial port;
```

Luego, en el interior tenemos que encontrar el puerto serial en el cual nuestro Arduino está conectado y utilizando eso para inicializar nuestro objeto puerto. Hay que recordar usar 9600 como la velocidad de transmisión para que coincida con el que nos propusimos en nuestro código de Arduino, y colocar 0 al número de puerto serial, que es el que se va usar:

```
println (Serial.list ());
Cadena PortName = Serial.list () [0];
port = nueva serie (esto, PortName, 9600);
```

Justo después de que mostremos los ángulos de los brazos en la pantalla serán enviados por el puerto serial, tal como se muestra en el siguiente código. Como hemos mencionado, queremos enviar los ángulos de los brazos de forma alterna: primero el lado derecho, y luego lado izquierdo. La manera más fácil de hacer que el sketch de Proccesing envié ambos ángulos cada vez de forma consecutiva lo hacemos mediante la construcción de una matriz de bytes que representan los dos valores de ángulo en orden y luego enviar toda esa matriz través del puerto serie a la vez. Esto es lo que parece en código:

```
byte out[] = new byte[2];
out[0] = byte(shoulderAngle);
out[1] = byte(elbowAngle);
port.write(out);
```

Una cosa importante a señalar aquí es que llamamos a la función de bytes en cada uno de nuestros valores de ángulo antes de guardarlos en nuestra matriz. Además de la conversión de ellos al tipo correcto para evitar un error, este redondea estos valores para el número entero más cercano.

### *Programación de Arduino*

La placa Arduino establecerá una conexión serial con nuestro ordenador, esto permitirá la comunicación del Proccesing (u otro software) para enviar información a nuestro Arduino y viceversa. El Arduino abrirá la conexión en serie y luego escuchar las instrucciones. Para poder realizar los

movimientos se necesita enviarnos dos números: el ángulo del brazo izquierdo y el ángulo del brazo derecho. El problema que se genera al realizar este procedimiento es que la conexión serial solo puede enviar un dato a la vez. Así que, para evitar esta limitación, se genera la siguiente alternativa:

- En primer lugar se enviará el ángulo para el brazo derecho, luego para el brazo izquierdo, y luego para el brazo derecho de nuevo, entonces el hombro de nuevo, y así sucesivamente. Esto significa que nuestro código Arduino tendrá que llevar un registro de las cuales valor tiene y esto se realizara en una variable nextMotor, esto con el fin de no tener conflictos a la hora de mover los motores.

En muestro programa de Arduino se declaran los dos motores a usar, la matriz ángulos y nextMotor:

```
int motor3 =10;
int motor4 =9;
int Angles[] ={0,0,0,0};
int nextMotor=0;
```

Se declara después las salidas de los motores y se inicia el puerto serial a 9600 baudios.

```
pinMode(motor1,OUTPUT);
pinMode(motor2,OUTPUT);
Serial.begin(9600);
```

En el Void loop se inicia el puerto serial y se captará los datos usando la variable ángulo con la siguiente sentencia:

```
int Angle=Serial.read();
```

Por último se cuadran los rangos de trabajos descritos anteriormente en la Figura 7, tanto para los ojos como para la boca, un pequeño ejemplo de esto se muestra a continuación:

```
if (Angles[0] > 60 & Angles[0] < 80 ) // Derecha
{
digitalWrite (motor1,250);
digitalWrite (motor2,0);
}
...
```

## V. PRUEBAS DEL SISTEMA

Los movimientos obtenidos mediante los motores accionados por Puente H en función de las respuestas realizadas mediante el Arduino transmitidas usando puerto serial utilizando respuestas realizadas mediante Proccesing en relación a los movimientos fueron realmente los esperados por lo que pudimos realizar este prototipo y probar el funcionamiento correcto en cualquier persona que lo desee.

Como prueba se ellos se plasmas las siguientes fotografías que muestran lo anterior:

**Fig. 12.** Diseño de la interfaz en proccesing

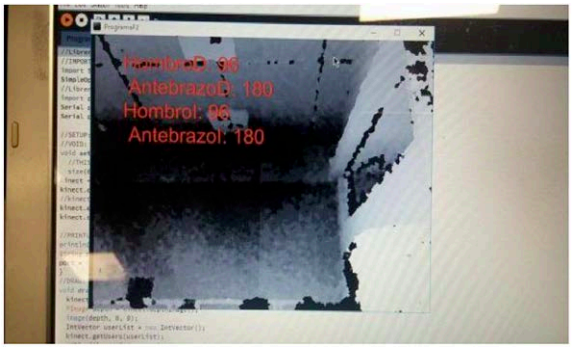



**Fig. 13.** Modelo final del animatronic

## VI. CONCLUSIONES

A continuación las conclusiones que deja el desarrollo del actual del proyecto

- Los entornos de programación de código abierto como lo es Proccesing nos ofrecen herramientas para el desarrollo de aplicaciones a bajo costo con millones de posibilidades de uso en distintos campos.

- El desarrollo del prototipo demostró ser muy entretenido para cualquier persona con lo cual podemos decir que puede ser una excelente fuente de diversión y entrenamiento para ejercicios en niños.

- Para un próximo prototipo es posible introducir más movimientos, pero estos aumentan la complejidad del diseño mecánico así como el software por lo que es necesario una evaluación del programa para llevar a cabo su implementación.

## REFERENCIAS

[1] «catarina.udlap.mx,» [En línea]. Available: http://catarina.udlap.mx/u_dl_a/tales/do cumentos/ldf/galvez_s_mi/capitulo3.pdf. [Último acceso: Junio 2015].

[2] A. I. M. G. M. T. K. Caro, «Apoyando las terapias de coordinación motriz de coordinación motriz de niños con autismo a través de videojuegos serios basados en movimiento».

[3] M. A. N. G. P. C. M. J. G. Martínez, «Avances tecnológicos en neurorrehabilitación,» Avances tecnológicos en neurorrehabilitación. , pp. 8-23, 2014.

[4] D. J. A. G. D. Jose Antonio Fontan Inza, «Interacción corporal con el ordenador para niños con restricciones motoras,» 2012.

[5] C. E. P. B. L. E. S. Q. F. R. J. Q. O. E. U. M. E. A. Gutíerrez Caseres, «Teleoperación Robot DELTA mediante Captura de Coordenadas y Pose de la mano,» 2014.

[6] M. C. C. S. K. M. D. G. J. P. Chuya Sumba, «Diseño e implementacion de un sistema para el analisis del movimiento humano usando sensores kinect,» Cuenca - Ecuador, 2013.

[7] G. Borenstein, in Making Things See, Canada, Makermedia, 2012, pp. 0-437.