

avances meses Octubre-Noviembre

Elaborado por Moises Stevend Meza Rodriguez

Se ha realizado lo siguiente:

Octubre

Semana 01:

- Se probó un enlace mqtt entre los 2 esp8266, verificando que trabajen conjuntamente.
- Se probó el software ROS en ubuntu 18, pero no se obtuvo avances ya que el software no contaba con compatibilidad con la kinect v1
- Se probó C# por ser el lenguaje que mantiene compatibilidad con la kinect v1. Este lenguaje nos podría servir si queremos algún punto en particular.
- Se instaló C# en visual studio y se procedió a programar un script de reconocimiento de trayectorias.

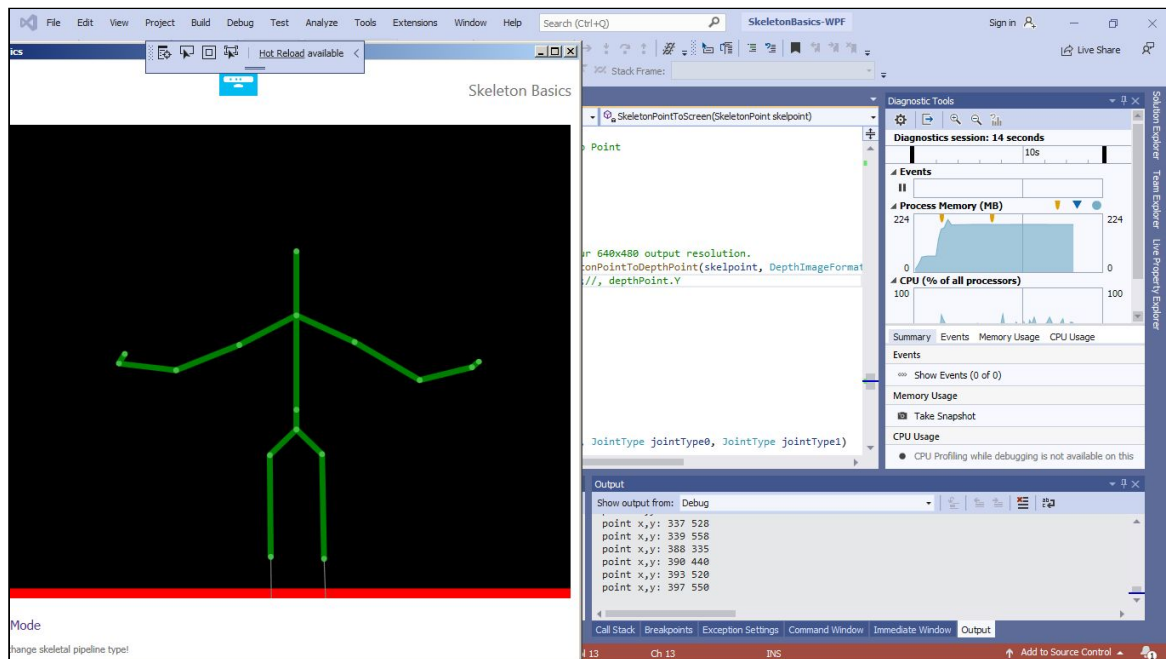


FIG.1- Esqueleto mediante líneas y puntos mediante C#

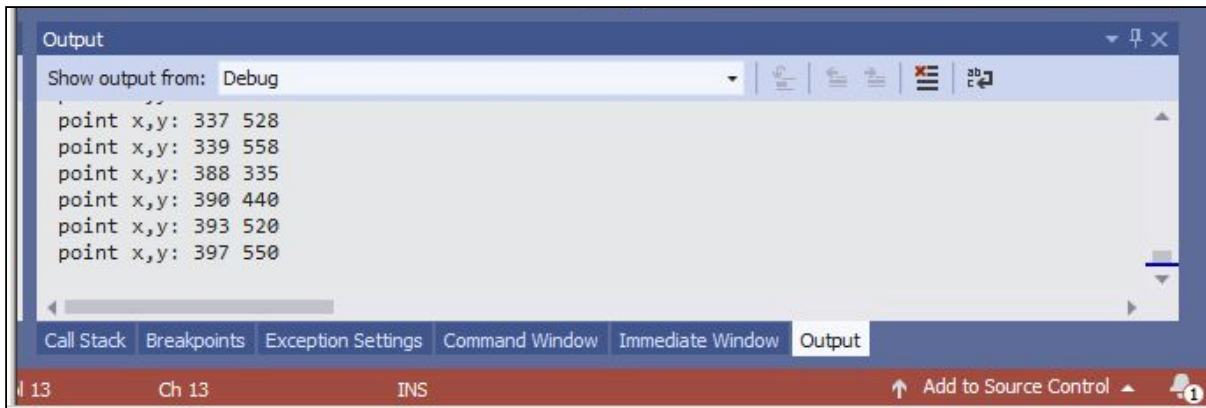


FIG.2- Puntos del esqueleto en C#

- Se instaló el software BREKEL como alternativa para obtener los puntos de manera rápida, se logró tener una visualización de los objetos y exportar los puntos de la parte superior del cuerpo, la desventaja es que solo se puede hacer la toma de puntos del cuerpo por 4 segundos.

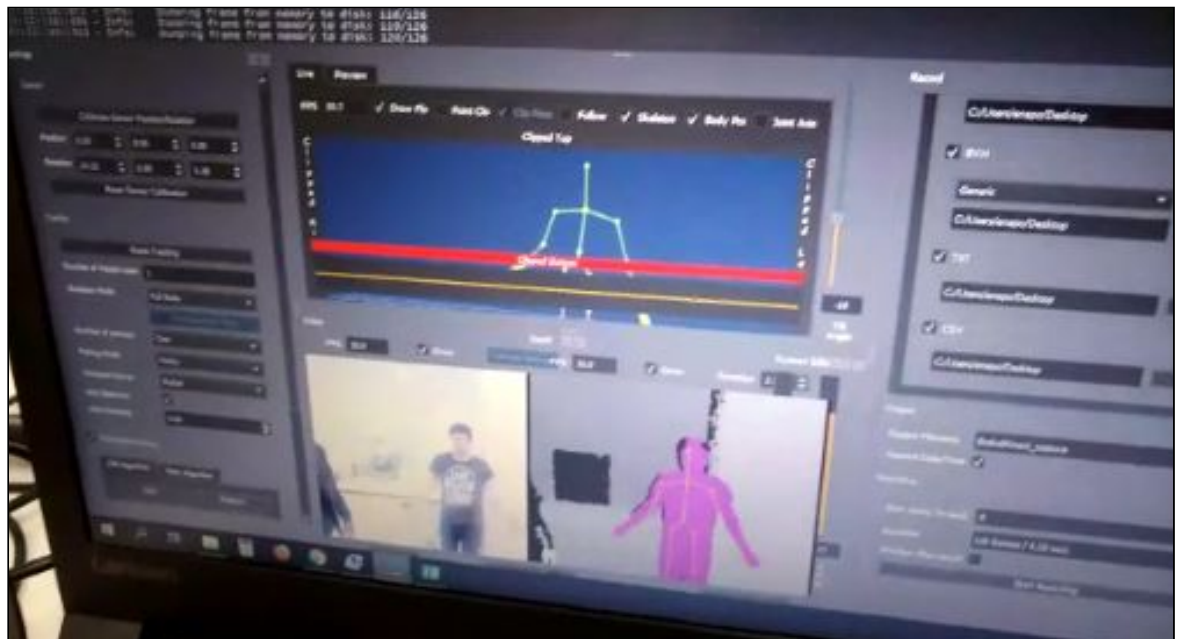


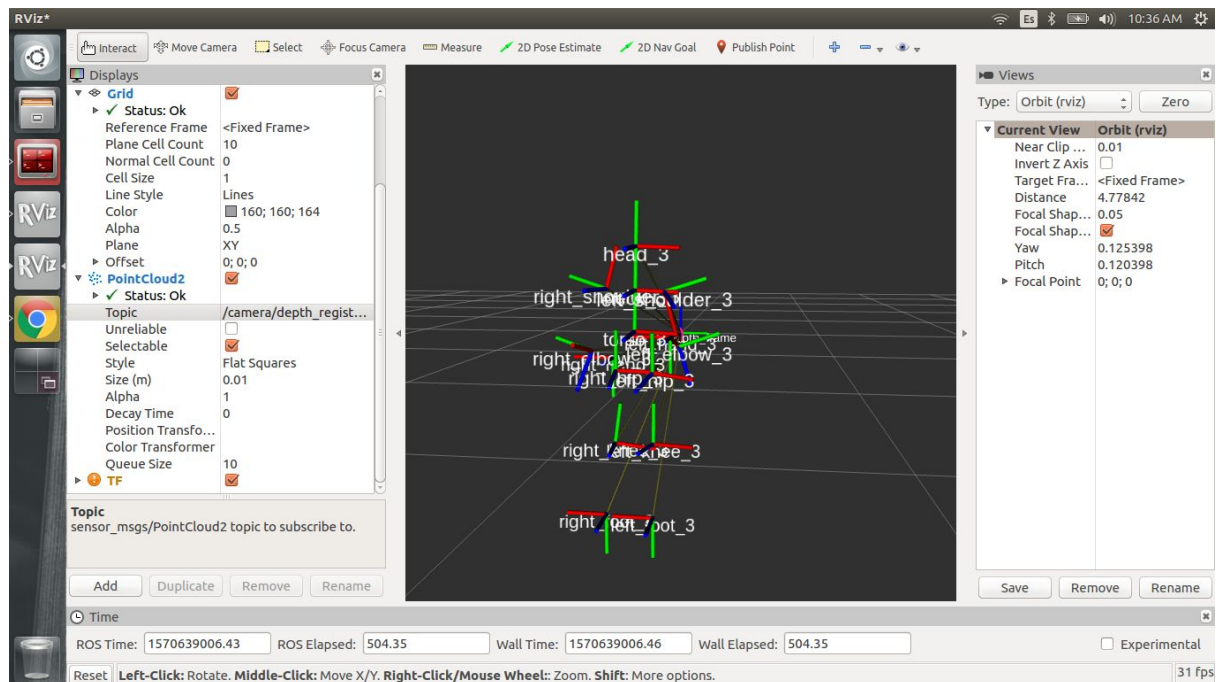
FIG.3- Entorno de trabajo de BREKEL

frame	time	waist_conf	waist_tx	waist_ty	waist_tz	waist_rx	waist_ry	waist_rz	spine_conf	spine_rx	spine_ry	spine_rz	neck_conf	neck_rx	neck_ry	neck_rz
0	0	0.5	-49.051884	94.489738	-291.946899	155.679993	-0.000005	174.699997	0.5	128.606949	-8.079885	176.39798	1	84.621513	-3.876397	13.5
1	0.033333	0.5	-49.051884	94.489738	-291.946899	155.679993	-0.000005	174.699997	0.5	128.606949	-8.079885	176.39798	1	84.621513	-3.876397	13.5
2	0.066667	0.5	-49.051884	94.489738	-291.946899	155.679993	-0.000005	174.699997	0.5	128.606949	-8.079885	176.39798	1	84.621513	-3.876397	13.5
3	0.1	0.5	-49.051884	94.489738	-291.946899	155.679993	-0.000005	174.699997	0.5	128.606949	-8.079885	176.39798	1	84.621513	-3.876397	13.5
4	0.133333	0.5	-49.051884	94.489738	-291.946899	155.679993	-0.000005	174.699997	0.5	128.606949	-8.079885	176.39798	1	84.621513	-3.876397	13.5
5	0.166667	0.5	-49.051884	94.489738	-291.946899	155.679993	-0.000005	174.699997	0.5	128.606949	-8.079885	176.39798	1	84.326378	-3.366278	12.5
6	0.2	0.5	-49.051884	94.489738	-291.946899	155.679993	-0.000005	174.699997	0.5	128.606949	-8.079885	176.39798	1	84.326378	-3.366278	12.5
7	0.233333	0.5	-49.051884	94.489738	-291.946899	155.679993	-0.000005	174.699997	0.5	128.606949	-8.079885	176.39798	1	84.018715	-2.8464	12.4
8	0.266667	0.5	-49.051884	94.489738	-291.946899	155.679993	-0.000005	174.699997	0.5	128.606949	-8.079885	176.39798	1	83.637886	-2.396238	11.4
9	0.3	0.5	-49.051884	94.489738	-291.946899	155.679993	-0.000005	174.699997	0.5	128.606949	-8.079885	176.39798	1	83.090569	-1.856264	10.5
10	0.333333	0.5	-49.051884	94.489738	-291.946899	155.679993	-0.000005	174.699997	0.5	128.606949	-8.079885	176.39798	1	82.313416	-1.203928	9.4
11	0.366667	0.5	-49.051884	94.489738	-291.946899	155.679993	-0.000005	174.699997	0.5	128.606949	-8.079885	176.39798	1	81.705162	-0.793394	8.4
12	0.4	0.5	-49.051884	94.489738	-291.946899	155.679993	-0.000005	174.699997	0.5	128.606949	-8.079885	176.39798	1	80.767357	-0.23107	7.4
13	0.433333	0.5	-49.051884	94.489738	-291.946899	155.679993	-0.000005	174.699997	0.5	128.606949	-8.079885	176.39798	1	79.933479	0.086915	5.4
14	0.466667	0.5	-49.051884	94.489738	-291.946899	155.679993	-0.000005	174.699997	0.5	128.606949	-8.079885	176.39798	1	79.224602	0.196676	4.4
15	0.5	0.5	-49.051884	94.489738	-291.946899	155.679993	-0.000005	174.699997	0.5	128.606949	-8.079885	176.39798	1	78.316612	0.356631	3.4
16	0.533333	0.5	-49.051884	94.489738	-291.946899	155.679993	-0.000005	174.699997	0.5	128.606949	-8.079885	176.39798	1	77.753174	0.131185	2.4
17	0.566667	0.5	-49.051884	94.489738	-291.946899	155.679993	-0.000005	174.699997	0.5	128.606949	-8.079885	176.39798	1	77.065926	0.006049	0.4
18	0.6	0.5	-49.051884	94.489738	-291.946899	155.679993	-0.000005	174.699997	0.5	128.606949	-8.079885	176.39798	1	76.388039	-0.045741	-0.4
19	0.633333	0.5	-49.051884	94.489738	-291.946899	155.679993	-0.000005	174.699997	0.5	128.606949	-8.079885	176.39798	1	75.814148	-0.180051	-1.4
20	0.666667	0.5	-49.051884	94.489738	-291.946899	155.679993	-0.000005	174.699997	0.5	128.606949	-8.079885	176.39798	1	75.155602	-0.278246	-2.4
21	0.7	0.5	-49.051884	94.489738	-291.946899	155.679993	-0.000005	174.699997	0.5	128.606949	-8.079885	176.39798	1	74.553093	-0.404904	-3.4
22	0.733333	0.5	-49.051884	94.489738	-291.946899	155.679993	-0.000005	174.699997	0.5	128.606949	-8.079885	176.39798	1	74.086411	-0.587667	-3.5
23	0.766667	0.5	-49.051884	94.489738	-291.946899	155.679993	-0.000005	174.699997	0.5	128.606949	-8.079885	176.39798	1	73.576462	-0.715103	-4.5
24	0.8	0.5	-49.051884	94.489738	-291.946899	155.679993	-0.000005	174.699997	0.5	128.606949	-8.079885	176.39798	1	72.655655	-1.022734	-6.4
25	0.833333	0.5	-49.051884	94.489738	-291.946899	155.679993	-0.000005	174.699997	0.5	128.606949	-8.079885	176.39798	1	72.655655	-1.022734	-6.4

FIG.4- Formato de puntos obtenido de una grabación de secuencial en BREKEL

Semana 02:

- Se probó el metasisistema operativo ROS(kinectic-ubuntu16 xenial) con la kinect v1, obteniendo los puntos a nivel de nodos de ROS (lo que nos permitiría enlazarlo con otros nodos-arduino, otra gran ventaja es poder obtener los puntos de manera ilimitada).



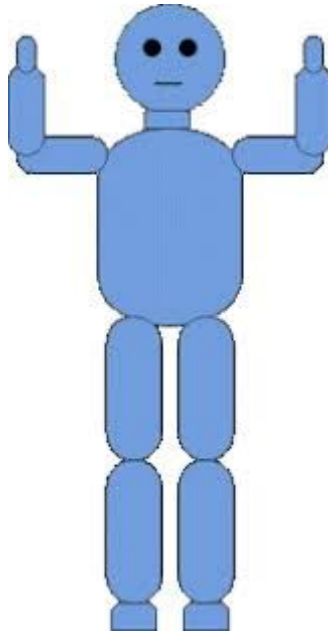


Fig .- Posición de inicio: “Pose PSI”

Se deja algunos enlaces usados:

- https://www.reddit.com/r/ROS/comments/6qejy0/openni_kinect_installation_on_kinetic_indigo/
- https://answers.ros.org/question/10325/how-do-i-run-the-openni_tracker/
- <http://dougsbots.blogspot.com/2012/02/ros-kinect-skeleton-tracking.html>
- <https://github.com/arnaud-ramey/NITE-Bin-Dev-Linux-v1.5.2.23>
- <http://mitchtech.net/ubuntu-kinect-openni-primrose/>
- Se probó el MYO-armband y se obtuvo los valores del IMU en ROS kinetic.


```
stevend@stevend-Satellite-L755: ~
x: -0.86279296875
y: 0.34716796875
z: -0.3603515625
linear_acceleration_covariance: [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
---
header:
  seq: 13162
  stamp:
    secs: 1570643069
    nsecs: 224992036
  frame_id: "myo_raw"
orientation:
  x: 0.327764620088
  y: -0.445076277408
  z: 0.692517202889
  w: 0.463570258764
orientation_covariance: [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
angular_velocity:
  x: -0.1875
  y: 0.375
  z: -2.125
angular_velocity_covariance: [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
linear_acceleration:
  x: -0.85400390625
  y: 0.33154296875
  z: -0.3642578125
linear_acceleration_covariance: [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
---
header:
  seq: 13163
  stamp:
    secs: 1570643069
    nsecs: 247529983
  frame_id: "myo_raw"
orientation:
  x: 0.32825216407
  y: -0.445197338178
  z: 0.692576665247
  w: 0.46301988037
orientation_covariance: [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
angular_velocity:
  x: 2.6875
  y: -0.8125
  z: -1.8125
angular_velocity_covariance: [0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
linear_acceleration:
```

Fig Datos obtenidos de la MYO ARM band

- Adquisición de datos del mpu mediante mqtt con la laptop.
- Se tuvo una reunión sobre las modificaciones que tendrá las futuras versiones.
- Se instaló raspbian en la raspberry pi3
- Se creó un script para adquirir datos del mpu6050

Datos raspberry pi3:

- ❖ **Red wifi:** LAB.ING.BIOMEDICA
- ❖ **Acceso via ssh:** ssh -X pi@healthrecoverpi
- ❖ **Username:** pi
- ❖ **Password:** MicroRobotica19

semana 03:

- Se hizo una pcb para la conexión entre mpu6050 y rpi3
- Se optó por rediseñar el dispositivo para un mejor control de trayectorias, basado en un esp32 y un mpu9250.
- Se diseñó la nueva arquitectura de comunicación de los dispositivos hardware y el smartphone.

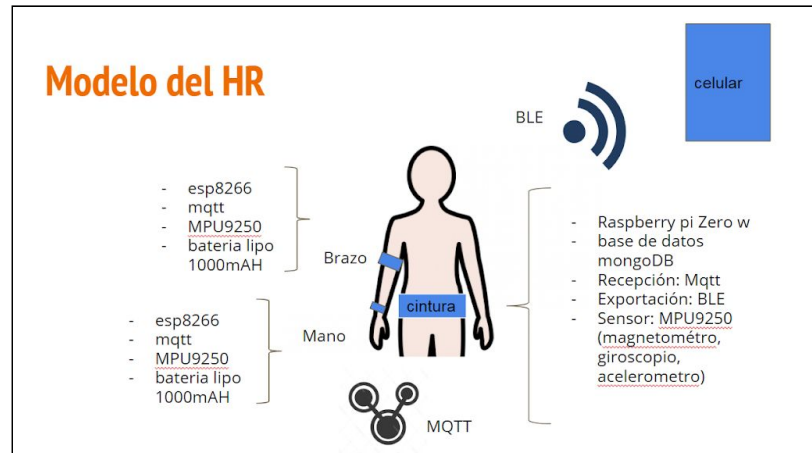


Fig. Nuevo modelo del HR

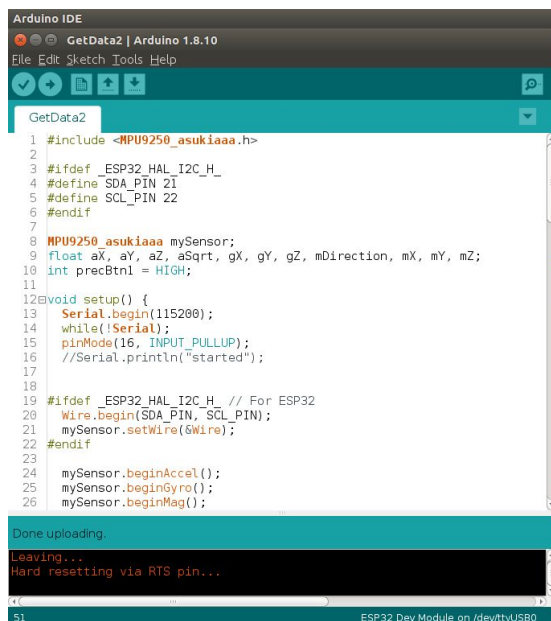
Semana 4:

- Pruebas de enlace entre el raspberry pi y los nodos del health recover.
- Se probó la instalación del ROS en el raspberry pi.
- Se analizó la identificación de patrones en WEKA
- Se analizó la Visualización de Datos en Python
- Se analizó la Base de Datos en MongoDB
- Se realizaron las compras para el nuevo dispositivo.

Noviembre

Semana 1

- Se coordinó las compras restantes.
- Se validó el nuevo modelo de hardware
- Se instaló ROS es el raspberry pi zero



```
1 #include <MPU9250_asukiaaa.h>
2
3 #ifndef _ESP32_HAL_I2C_H_
4 #define SDA_PIN 21
5 #define SCL_PIN 22
6 #endif
7
8 MPU9250_asukiaaa mySensor;
9 float aX, aY, aZ, aSqrt, gX, gY, gZ, mDirection, mX, mY, mZ;
10 int precBtn1 = HIGH;
11
12 void setup() {
13   Serial.begin(115200);
14   while(!Serial);
15   pinMode(16, INPUT_PULLUP);
16   //Serial.println("started");
17
18 #ifndef _ESP32_HAL_I2C_H_ // For ESP32
19   Wire.begin(SDA_PIN, SCL_PIN);
20   mySensor.setWire(&Wire);
21 #endif
22
23   mySensor.beginAccel();
24   mySensor.beginGyro();
25   mySensor.beginMag();
26 }
```

Done uploading.
Leaving...
Hard resetting via RTS pin...

ESP32 Dev Module on /dev/ttyUSB0

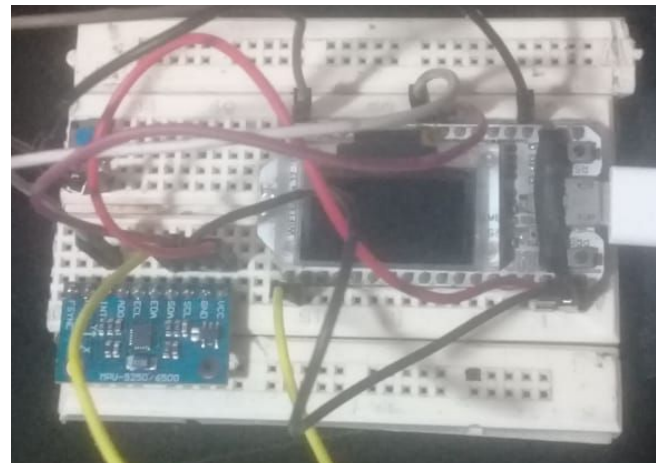


Fig.- Código base y circuiteria para el esp32 y el mpu9250

Semana 2:

- Se analizó patrones de la data obtenida del mpu9250
- Se empezó a implementar algoritmos de IA para obtener secuencias de movimientos.

Semana 3:

- Se realizó el diseño de la versión 2 del HR en eagle, se desarrolló en un modelo de 2 capas para los nodos del brazo.

Resumen

Desarrollo del hardware.-

En cuanto al desarrollo del hardware se optó por pasar el dispositivo central esp8266 a un esp32, además se optó por cambiar el mpu6050 por un mpu9250. Se completó la verificación de la comunicación y se desarrolló un modelo de pcb en eagle para la parte de los nodos del brazo. Se optó por adicionar un nodo central para mejorar el procesamiento de los datos, este nodo central está basado en una raspberry pi zero w. Se desarrolló un esquema electrónico para la comunicación entre el raspberry pi, mpu9250 y la electrónica suficiente para el buen desempeño.

Grabación de movimientos.-

Se optó por trabajar con el metasisistema operativo ROS(Robotics Operating System) y la Kinect v1.0 para la obtención de los movimientos y su análisis.

Compra de componentes para tarjetas. Se llegó a completar las compras para esta segunda versión del HR. Se requirieron componentes adicionales para la versión final del pcb.

Grabaciones de movimientos.-

Se grabó unas secuencias mediante ROS y Kinect y se estuvo programando el reconocimiento de patrones, también se completó los códigos de programación para los nodos del brazo. El algoritmo aún está en depuración.