Practica 2:
Sistema de alerta en caidas

Pulido Bejarano Raymundo

7 de Noviembre del 2020

1. Introducción

En esta practica, se desarrollará un sistema que mediante el dispositivo *Sense Hat*, en colaboración con la Raspberry, se obtendran los datos de *Aceleración* y *Orientación median Giroscopio* sobre el usuario, el cual portara esta, en posición vertical, una vez con estos datos, estos se procesaran para determinar si es que el usuario sufrió una caída, y en dado caso de esto, se le enviara un mensaje a un contacto de confianza que previamente fue ligado, a su vez se deberá poder visualizar los datos sensados en tiempo real, mediante una interfaz gráfica, para poder ser consultados de una forma clara.

2. Desarrollo

El desarrollo de esta practica consistió en implementar un flujo en el sistema Node-Red, que consiste en obtener los datos de la tarjeta Sense Hat.

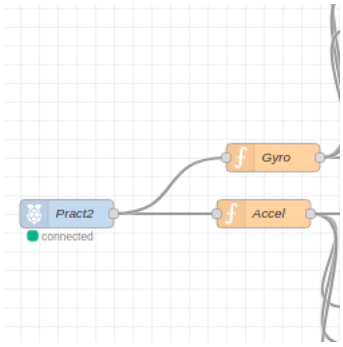

Figura 1: Nodo de Entrada de Datos

Una vez obteniendo estos datos se procedió a procesar los dichos datos de manera que primeramente obtuvimos el valor absoluto de los datos, ya quela

Sense Hat, maneja estos como números Reales, donde el signo represéntala dirección en la que se efectuó el movimiento, pero una vez haciendo la correlación entre la posición designada de nuestro dispositivo de censado con la relación con las variables de aceleración y giroscopio, pude determinar un área de acción óptima. Donde si con las coordenadas se encuentran en dicha área, se generara una bandera que determina que esa variable se encuentra en el rango buscado.

A su vez se separaron las distintas coordenadas, tanto de Aceleración como de Giro, para poder ser mostradas posteriormente por la interfaz Gráfica del Dashboard.

Esto se puede mostrar en las siguientes imágenes:

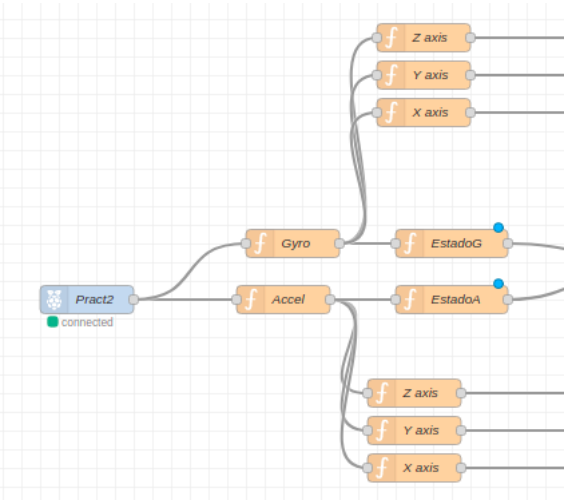

Figura 2: Arquitectura del Sistema de Procesamiento

Como se puede observar en la imagen anterior una vez separadas las variables de Aceleración y de Giroscopio, son procesadas de manera que una vez que sus datos se encuentran en el rango deseado, son enviados a un post proceso, para separar los sub-valores y poder ser mostrados en el Dash Board o seguir con el proceso.

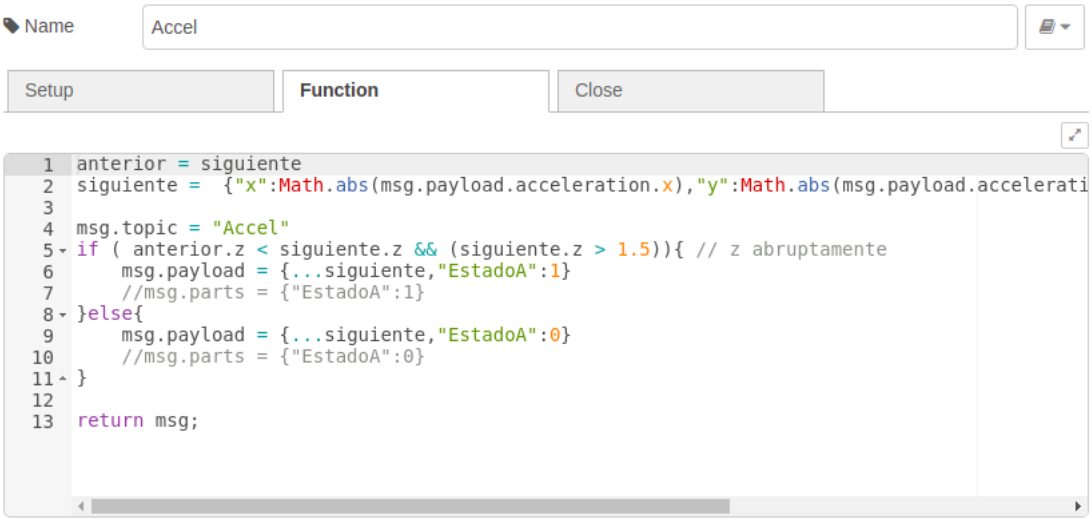

Figura 3: Procesamiento de la variable: Aceleración

Como se puede ver en la imagen anterior, el procesamiento de la variable de Aceleración fue el siguiente, primeramente se obtuvo el valor absoluto del valor de cada eje de Aceleración.

Posteriormente se comprueba si el valor anterior en el eje Z es menor al actual y si también el valor actual es mayor a 1.5. Donde en dado caso de que fuera de esa forma se determina que debe ser pasado al siguiente proceso, ya que cumple con los valores deseados para el fin del sistema

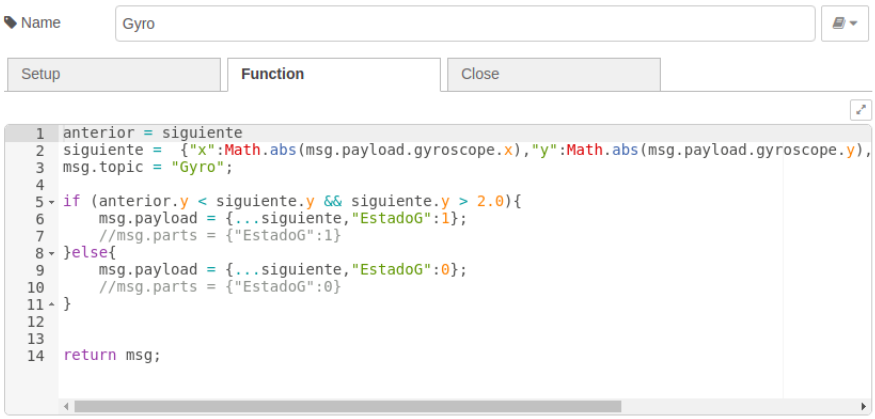

Figura 4: Procesamiento de la variable: Giroscopio

Como se puede ver en la imagen anterior, el procesamiento de la variable de Giroscopio fue el siguiente, primeramente se obtuvo el valor absoluto del valor de cada eje de Giro.

Posteriormente se comprueba si el valor anterior en el eje Y es menor al actual y si también el valor actual es mayor a 2. Donde en dado caso de que fuera de esa forma se determina que debe ser pasado al siguiente proceso, ya que cumple con los valores deseados para el fin del sistema.

En esta parte del proceso, los datos viajan por dos canales, el primero donde estos separarán para posteriormente ser mostrados en el Dash Board y la segunda que conlleva seguir el proceso, puesto que se obtienen las banderas de ambas variable donde se indica si es que estas obtuvieron los valores buscados para el objetivo de la practica A.Z >1.5 y G.y >2

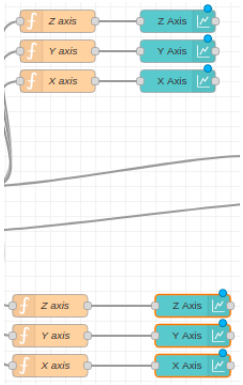

Figura 5: Proceso para el DashBoard

En la imagen anterior se puede observar los nodos que realizan la separación por ejes de las variables de Aceleración y Giroscopio, para posteriormente ser mostradas en el DashBoard

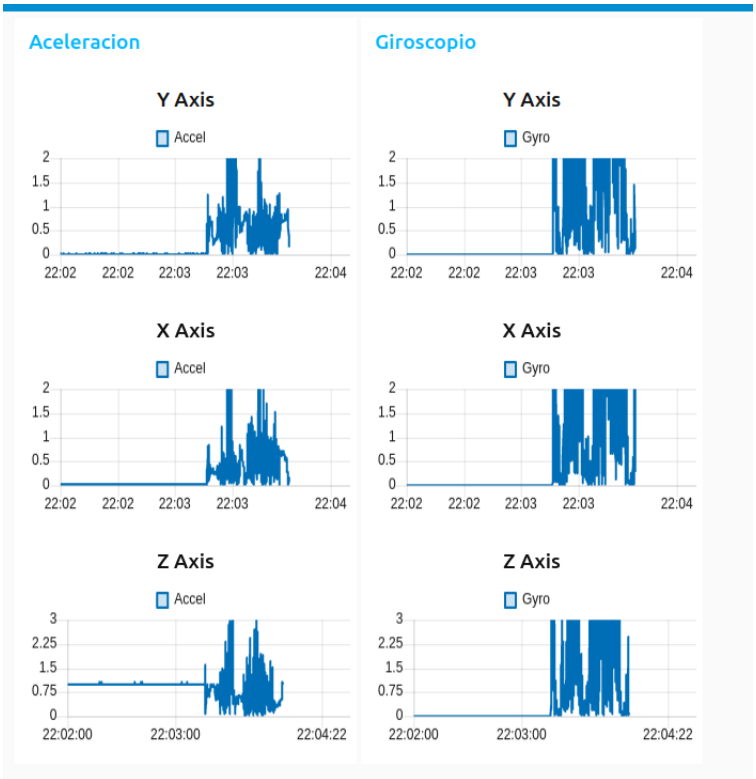

Figura 6: DashBoard

En la Imagen anterior se puede observar que los datos censados plasmados en una grafica respecto al tiempo en que fueron obtenidos los dichos datos.

Continuando con el segundo camino de los datos.

Una vez que se separaron las banderas de estado generadas por él procesamiento anterior, debemos unirlas en el mismo mensaje, para de esta forma poder realizar una comparación entre ambas banderas donde solo en caso de que las dos estén activas se considerara una caída del usuario.

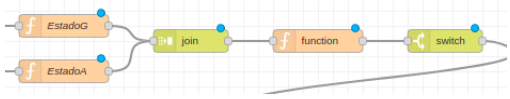

Figura 7: Manejo de Banderas

En la imagen anterior se puede observar que una vez obtenidas las banderas generadas por el procesamiento anterior, estas serán unidas por un nodo join, para posterior ser evaluada la condición de que ambas deben estar presentes

para que posteriormente ser evaluadas por un nodo switch, que nos ayudara como filtro intermedio al envió de los mensajes, ya que solo en caso de que la condición sea aceptada se genera una salida del nodo.

Como paso anteriormente, en este caso el desarrollo del sistema sufrió otra división, la primera conlleva la creación de un bot en el sistema de mensajería Telegram, ya que este será el que recibirá los mensajes que le notificara al usuario, y la segunda parte es la gestión de los comandos necesariamente recibidos para obtener los datos del chat al que se le enviara los mensajes en caso de una caída.

Para realizar el primer paso, se deberá buscar el "BootFather", ya que según la documentación de Telegram, este es el único medio para realizar la creación y gestión de boots en la plataforma, por lo que se procedió a buscarlo. Una vez encontrado se debe de iniciar el boot, como si de cualquier otro se tratase, y únicamente quedaría seguir los pasos, claramente descritos por el mismo boot.

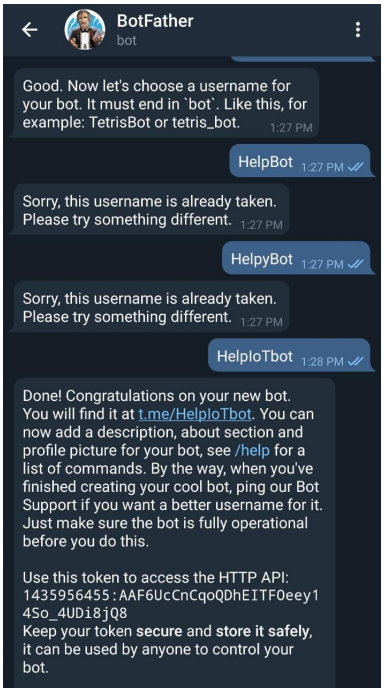

Figura 8: Creacion del Boot receptor

Como se puede apreciar el la imagen anterior, una vez seguimos los pasos descritos por el "BootFather", este nos otorgara un "Token", el cual es el que nos permitirá realizar la comunicación desde nuestra aplicación con el boot, lo próximo que se debe realizar el asignar al menos un comando, el cual designe

/start"para iniciar la comunicación.

Una vez creado el bot, procederemos a crear el manejador del evento, para esto se usarán los nodos provistos por node-red para trabajar con Telegram, una vez configurado el token de boot creado en node-red, podemos gestionar los comando y obtener todos los datos del chat al que le enviaremos mensajes, en este caso nuestro bot.

Para manejar la recepción del comando /start, que como marca el nombre inicia la conversación con el bot, esta información será almacenada en una variable global para de esta forma realizar el envió del aviso cuando sea necesario, ya que estos mensajes son asíncronos a los comandos.

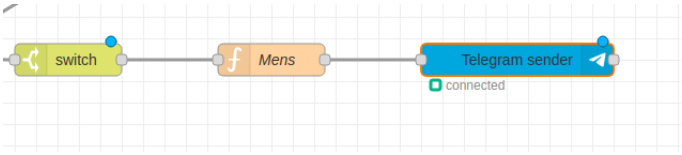

Figura 9: Manejo de Mensajes Telegram

```
if (global.get("myData")){
    //msg.payload = {chatId: global.get("myData"),type: "message",content: "Corre!! algo le paso"}
    msg.payload = {chatId: global.get("myData"),type: "message",content: "Alerta!!, Sufrio una caida"}

    return msg;
}
```

Figura 10: funcio de Mensajes Telegram

Como se puede observar en las imágenes anteriores, en caso de que la variable global no tenga datos, es decir en caso que no haya ingresado el comando /start, no se enviaran los mensajes de alerta, ya que el bot no estaría activo.

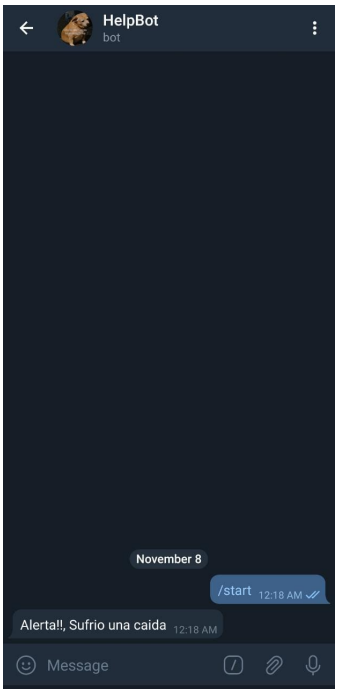

Figura 11: Recepcion de alerta

Como se puede ver en la imagen anterior, con esta topología se puede realizar la notificación de una caída de un usuario, notificando a otro usuario del suceso mediante Telegram.

A continuación se mostrará la topología de flujo que se creó completa.

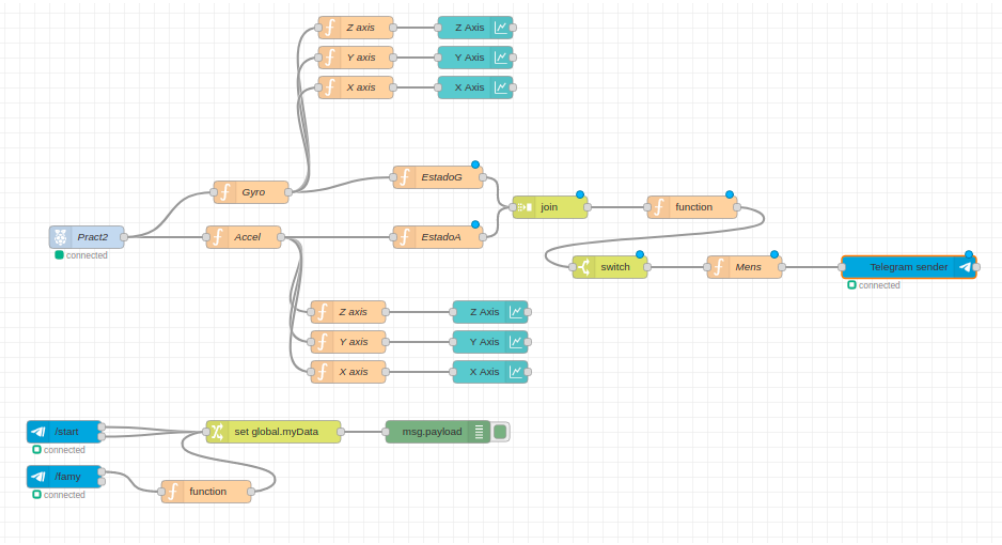

Figura 12: Topologia completa

En la imagen anterior se puede observar la topología completa del proceso o flujo de acción general, desde el manejo de los datos hasta la recepción de los comandos.

3. Conclusión

En conclusión, en esta practica pude obtener gran cúmulo de experiencia y habilidad en el manejo de los nodos de node-red y en el manejo de comunicación mediante los nodos de Telegram.

A su vez se aprendió a manejar los nodos de desarrollo y los nodos para desarrollar funciones sobre los datos, y de esta forma filtrarlos, para posteriormente manejarlos.

También se logró gestionar el envió de mensajes mediante Telegram y un boot, como también recibir comandos y realizar acciones en respuesta, como también mostrar los datos procesados en una interfaz gráfica web.

En sítensis se pudo identificar cuando el usuario sufrió una caída y a su vez realizar un mensaje como aviso de urgencia.