PROYECTO 3

Mateo Oyuela, Juan Pablo Orozco

Correcciones realizadas del proyecto 2

- Se corrigieron las pruebas de hipótesis de Mann-Whitney.
- Se evaluó otras características para hallar una que permitiese la discriminación de movimientos desde las señales.
- Orientado al proyecto 3, se adaptó el análisis para la evaluación de una sola señal.

Rutinas encontradas para aplicar a arduino

Teniendo en cuenta las limitaciones de memoria al trabajar con arduino, se tuvieron que escoger las rutinas más importantes vistas en proyectos anteriores para realizar la clasificación de la señal. Estas son:

- **TKEO:** En proyectos anteriores se estableció la relevancia del operador Teager Kaiser Energy Operator (TKEO) para el reconocimiento de contracciones musculares en la señal de EMG.

- **Valor cuadrático medio (RMS):** La característica más apropiada para realizar la discriminación de los movimientos es el rms debido a que el análisis es fundamentalmente una comparación de amplitudes promedio de segmentos de la señal.

Diseño del filtro

El filtro propuesto para aplicar a la señal es un IIR pasa altas con frecuencia de corte de 60Hz. Como se está trabajando con un único canal, los cambios de fase que pueda tener el filtro no son relevantes, es por esto que se decidió utilizar un filtro IIR en vez de un FIR, además el filtro IIR tiene la capacidad de realizar el mismo filtrado que el FIR a un orden considerablemente menor, que se busca por las limitaciones del arduino. La principal función del filtro implementado es eliminar el ruido eléctrico, que es una de las principales fuentes de ruido en la toma de la señal. Según la bibliografía, el ruido eléctrico tiene una frecuencia de 60Hz, por tanto se optó por realizar un filtro notch o un pasa altas. Pese a que el notch es el filtro más apropiado para eliminar este ruido, dadas las limitaciones de procesamiento del arduino se decidió optar por el filtro pasa altas. El orden del filtro se escogió mirando los diagramas del filtro en python en función de su atenuación de la frecuencia de corte, desde los cuales se determinó el orden del filtro, que en este caso es de orden 3 (siendo éste el mínimo orden en el que el filtro se comportaba de manera satisfactoria).

El filtro fue diseñado en pyhton utilizando el código facilitado por el profesor:

```python
N = 3 # filter order

fs = 140

# compute the filter coefficients
b,a = signal.iirfilter(N, 65/(fs/2), btype='highpass', ftype='butter')

mfreqz(b,a,N, fs/2)

print(b)
print(a)
```

✓ 0.6s Python

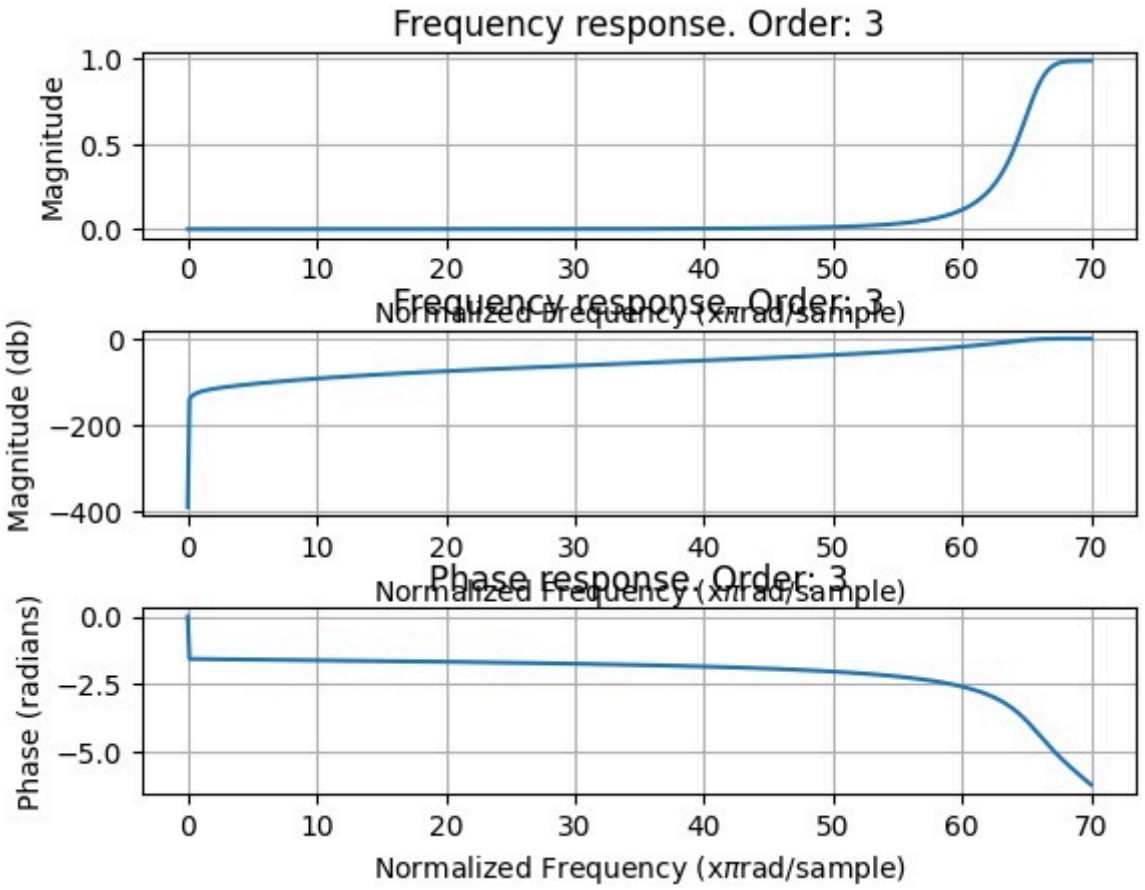
```
C:\Users\MATEO\AppData\Local\Temp\ipykernel_18008\1227699950.py:17: RuntimeWarning: divide by zero encountered in log10
  h_dB = 20 * np.log10 (abs(h));
```

```python
b_round = np.round(b, decimals=4, out=None)
a_round = np.round(a, decimals=4, out=None)

mfreqz(b_round,a_round,N, fs/2);

print(b_round)
print(a_round)
```

✓ 0.8s Python



Y estos los coeficientes del filtro:

```
b= [ 0.0011 -0.0034  0.0034 -0.0011]
a= [1.     2.5521 2.199  0.6378]
```

La implementación en el microcontrolador puede verse en el archivo adjunto.

Diagrama de flujo

Implementación del flujo

La implementación en el controlador se realizó por medio del entorno de Arduino. Allí se definieron las funciones para el filtro, TKEO y cálculo del RMS. La lógica principal por la cual se desarrolló el flujo propuesto consiste en que la salida de una función es la entrada de otra, siguiendo el orden establecido en el diagrama. Finalmente, al final de cada ciclo se realiza la comparación del valor de RMS con el umbral.

Cómo se implementó de manera más específica puede verse en el archivo adjunto que contiene el código en Arduino.

Informe de funcionamiento

SUJETO	VECES QUE DETECTA	VECES QUE NO DETECTA
1	15	5
2	16	4

Puede verse en la tabla que la probabilidad de detectar el movimiento de flexión es de alrededor de 75%. Esto puede hacer referencia a que se debe establecer otro valor de umbral o referirse a otro tipo de ruidos captados por los electrodos (por la manera realizada perfectamente puede ser un problema de fijación de electrodos y/o reutilización de estos).

Comparativa microcontroladores

Microcontrolador	Memoria	Velocidad de procesamiento	Puerto y periféricos	Costo (COP)	Facilidad de programación	Consumo de energía
ARDUINO UNO R3	2KB RAM, 32KB ROM	16 MHz	14 GPIO, 6 ADC	47000	Entorno Arduino IDE	Bajo
RASPBERRY PI 4	1GB RAM	1.5 MHz	GPIO, UART, I2C, SPI	330000	Raspbian OS, Python	Moderado
STM32F407	192 KB RAM, 1MB ROM	168 MHz	GPIO, UART, I2C, SPI	76000	STM32CubeIDE	Bajo
PSoC 5 CY8CKIT-059	16-64 KB RAM	80 MHz	GPIO, UART, SPI, I2C, ADC, DAC, PWM, Temporizadores	118000	PsoC Creator	Variable

Para mejorar la adquisición, filtrado, procesamiento y discriminación de una señal EMG obtenida experimentalmente desde el músculo bíceps y generar una salida que indique flexión o extensión del brazo, considerando la relación costo-beneficio y las especificaciones técnicas, resultan ser muy buenas opciones los microcontroladores STM32F407 y PSoC 5 CY8CKIT-059. Teniendo en cuenta las características de cada una:

- STM32F407: Es una opción sólida en términos de capacidad de procesamiento y periféricos, siendo una opción confiable para aplicaciones que requieran un alto rendimiento y una variedad de interfaces. Tiene una buena capacidad de procesamiento con alrededor de 168 MHz, lo que podría llegar a una tasa de frecuencia de adquisición de datos bastante similar a las frecuencias en las que se conocen las señales EMG.

- PsoC 5 CY8CKIT-059: Destaca por su arquitectura flexible y la inclusión de CapSense para detección capacitiva, lo cual podría ser útil para aplicaciones que involucren interfaces táctiles o sensores capacitivos adicionales. También es importante destacar que es ampliamente conocido por nosotros, conocemos acerca de su efectividad y eficiencia en la adquisición de datos analógicos y su conversión para datos digitales para procesamiento y programación multifuncional, con una buena capacidad de procesamiento de 80 MHz, y capacidad de programación para el filtrado y demás operaciones de procesamiento de señales.

Las conclusiones a las que podemos llegar, es que un microcontrolador, para una buena adquisición de señales EMG, debe tener una buena capacidad de memoria y una buena capacidad en la velocidad de procesamiento, esto es necesario para tomar la cantidad de muestras necesarias en una frecuencia de muestreo que pueda adquirir eficientemente el comportamiento de las señales EMG en la discriminación de cada tipo de movimiento, esto será fundamental en el proceso de filtrado, utilizando frecuencias de filtrado de interés que permitan un buen filtrado.

Bibliografía

- Boyer, M., Bouyer L., Roy J., Campeau A. (2023). *Reducing Noise, Artifacts and Interference in Single-Channel EMG Signals: A Review.*
- *Proyecto Arduino : Arduino UNO R3: https://proyectoarduino.com/arduino-uno-r3/#google_vignette*
- *Raspberry Pi 4 Tech Specs: https://www.raspberrypi.com/products/raspberry-pi-4-model-b/specifications/*
- *Discovery STM32F407: https://moviltronics.com/tienda/tarjeta-discovery-stm32f4/*