LABORATORIO 2: Programación en radio definida por software (GNURADIO)

Diego Ferney Beltran Rodríguez - 2210401
Arman Yerlaith Abello Plata - 2201523
Kevin Sayari Durán Lizarazo - 2194097

Escuela de Ingenierías Eléctrica, Electrónica y de Telecomunicaciones
Universidad Industrial de Santander

Repositorio: `https://github.com/KESADULI/CommunicationsII_2024_2_lab`

8 de septiembre de 2024

Abstract

In this report, GNURADIO, a software development environment specialized in software-defined radios (SDR), was used. It allows the design and testing of signal processing systems without the need for specialized hardware, using blocks that process information in real time. In practice, a low-pass filter was implemented using accumulator blocks programmed in Python language to study the statistical characteristics of noisy signals, using the measurements of mean, root mean square, RMS, power and standard deviation, allowing a better understanding of the behavior and noise of the processed signals.

1. Introducción

GNURADIO es un kit de herramientas de desarrollo de software que proporciona bloques de procesamiento de señales para implementar radios de software, las cuales realizan el procesamiento sin el requerimiento de circuitos integrados especializados [1]. De esta forma, podemos diseñar y probar una gran cantidad de sistemas de procesamiento que además nos permitan entender el funcionamiento y limitaciones de estos mismos, lo que la hace una herramienta de gran utilidad académica.

El formato de GNU Radio está orientado a la programación de sistemas en tiempo real, por lo que no opera enviando información muestra por muestra a los bloques, sino por arrays [2].

GNURADIO no solamente provee una listas de bloques, sino que permite la creación de bloques por programación en Python. Así, podemos generar bloques, por ejemplo, para mediciones estadísticas y filtros.

Durante esta práctica, se aplicaron los bloques acumulador y diferenciador, que corresponden a un filtro pasa bajas y uno pasa altas, en un posible caso real para evaluar los efectos de los filtros en algunas mediciones estadísticas, buscando además la recuperación de una señal con ruido. Las mediciones estadísticas aplicadas representan las cuestiones determinísticas de una señal aleatoria, estas son: La media, media cuadrática, valor RMS, potencia promedio y desviación estándar. Estas nos permiten, entre otras cosas, detectar ruido, darnos una visión de la intensidad de una señal, la amplitud promedio y la dispersión de la señal respecto de esta (desviación estándar). Estas señales son promedios de tiempo, y están definidas de la siguiente manera:

- Media
$$X_m = \langle x(t) \rangle, \tag{1}$$
- Media cuadrática
$$X_C = \langle x^2(t) \rangle, \tag{2}$$
- Valor RMS
$$X_{RMS} = \sqrt{\langle |x(t)|^2 \rangle} \tag{3}$$
- Desviación Estándar
$$\sigma = \sqrt{\langle [x(t) - X_m]^2 \rangle}, \tag{4}$$
- Potencia Promedio
$$P = \langle |x(t)|^2 \rangle \tag{5}$$

2. Metodología

Inicialmente, se diseñaron en GNU Radio los bloques acumulador y diferenciador con el uso de un Python Block para cada función. De la misma manera, se desarrolló un bloque que calcula los parámetros de promedio de tiempo: Media, media cuadrática, valor RMS, desviación estándar y potencia promedio como se aprecia en la figura 1.

Se creó un vector de entrada equivalente a una sección que se repite para generar un vector real. Esto quiere decir que el vector final corresponde a repeticiones del vector ingresado, teniendo una longitud establecida por GNU Radio. Luego, se desarrolló un sistema conectando los bloques, que permite la lectura de los promedios de tiempo de una señal definida por el vector de entrada. Se evaluó después el efecto de los filtros, graficando sus salidas. Finalmente, se implementó un sistema utilizando los bloques de medición de promedios de tiempo, los bloques acumulador y diferenciador, un vector de entrada correspondiente a una señal cardíaca en un electrocardiograma (ECG) con una adición de ruido sinusoidal, de forma que se puedan medir los valores promedio antes y después de la aplicación de filtros a la señal para comprobar su eficacia en la recuperación de una señal original con ruido añadido.

3. Resultados

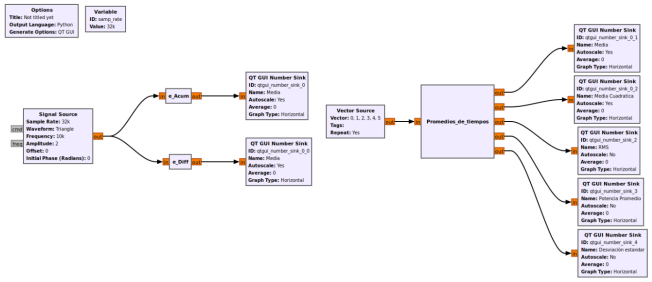

Fig. 1: Bloques de acumulación, diferenciación y promedios

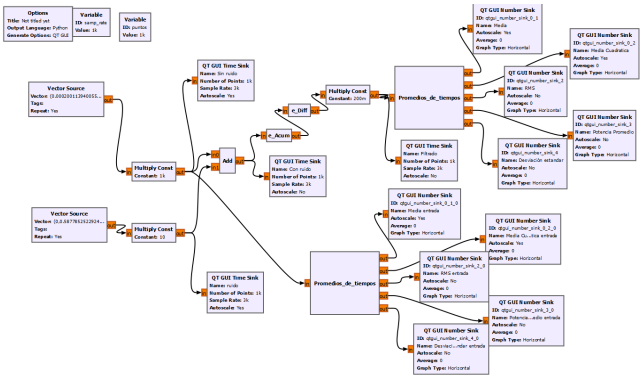

Fig. 2: Diagrama de bloques

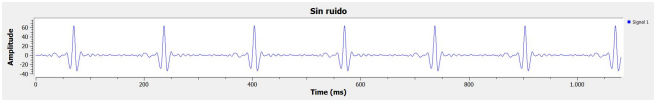

Fig. 3: Gráfico de un electrocardiograma

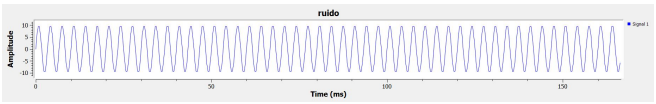

Fig. 4: Señal de ruido seleccionada

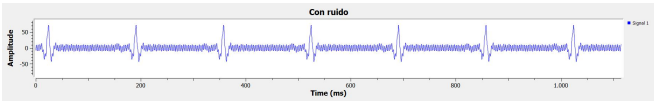

Fig. 5: Señal electrocardiograma con ruido

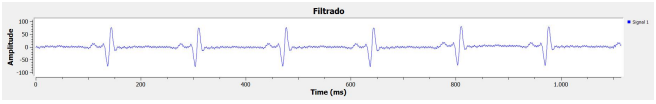

Fig. 6: Señal filtrada

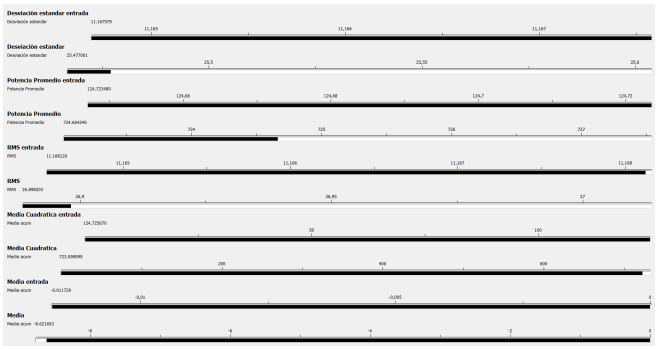

Fig. 7: Datos estadísticos

En la figura 2 tenemos el diagrama de bloques por el cual pasará un vector de datos que contiene el resultado de un electrocardiograma (figura 3), el cual, al agregarle el ruido de la figura 4 obtendremos como resultado la figura 5.

Dado que el ruido que se añadió al electrocardiograma es de alta frecuencia, se optó por pasar la señal por un bloque integrador, obteniendo así la señal de la figura 6

Como resultado obtenemos una señal similar al del electrocardiograma, con frecuencias altas atenuadas pero sin llegar a ser igual, esto es debido a nuestro bloque acumulador, si bien se comporta como un filtro pasa bajas, su precisión de filtrado deja mucho que desear. Ahora por medio del bloque creado para el cálculo estadístico de la señal se podrá comparar la entrada con la salida (figura 7).

4. Conclusiones

Al inicio de la práctica, enfrentamos dificultades debido al limitado conocimiento que teníamos sobre GitHub, lo que resultó en errores al ejecutar comandos e implementar las descargas y subidas de los documentos y archivos. Sin embargo, a medida que avanzábamos, fuimos consolidando nuestra comprensión de esta herramienta, lo que nos permitió, al finalizar, apreciar los múltiples beneficios que ofrece. Uno de los aspectos más relevantes en esta práctica fue la integración del trabajo colaborativo. Al utilizar las ramas de la plataforma, aprendimos a realizar tareas de manera independiente, pero con un objetivo común, lo que nos ayudó a reducir la carga académica y, en el futuro, podría disminuir la carga laboral.

Nos encontramos con obstáculos relacionados con la creación de llaves SSH, las cuales son vitales para asegurar la autenticación en GitHub. Si su configuración no es correcta, pueden surgir inconvenientes al intentar acceder al repositorio. Además, aunque la capacidad de acceder a las ramas de otros colaboradores fomenta la cooperación y ofrece seguridad mediante la encriptación, un manejo inapropiado podría comprometer la seguridad del proyecto. Por esta razón, es fundamental aplicar una gestión cuidadosa y seguir estrictas prácticas de seguridad para maximizar los beneficios y reducir los riesgos que acompañan el uso de las llaves SSH.

La efectividad del filtro al recuperar la señal es mala, pues su precisión era baja y se perdieron muestras al eliminar el ruido, por esta razón también se dio esa diferencia en las cinco medidas estadísticas.

Se presentaron inconvenientes durante el desarrollo en la parte de la implementación y codificación del bloque acumulador e_Acum, debido a que la suma se hacía muy grande y este seguía aumentando la señal a la salida, dando como resultado datos que alcanzaron magnitudes exponenciales, o dando como resultado la gráfica filtrada en forma de triángulos, pero se pudieron solucionar esos problemas disminuyendo el tamaño del vector de entrada, cambiando de 10000 muestras a 500, dando como resultado la gráfica 7.

Se observó que existía un problema con la recepción de datos en los filtros, debido a que los bloques operan con un conjunto de datos que no corresponde con el período de la señal. No es efectivo limitar el número de datos que toma el bloque en el código solamente, dado que GNURADIO no envía los datos muestra por muestra sino en arrays.

Referencias

[1] “What is gnu radio?” [Online]. Available: https://wiki.gnuradio.org/index.php?title=What_is_GNU_Radio%3F

[2] H. Ortega and O. Reyes, *Comunicaciones digitales basadas en radio definida por software*. Editorial UIS, 2019.