Práctica 1
PROGRAMACIÓN EN GNURADIO

Carlos Fernando Carreño Jerez - 2201729
Juan Esteban Pinto Orozco - 2215585

https://github.com/CMarsJ/CommunicationsII_2024_2_CFJP

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

8 de Septiembre del 2024

abstract

This report details the practice nº 1 of the digital communications laboratory, this practice focuses on the implementation of custom blocks as accumulators and differentiators for signal processing. Throughout the report, several examples of custom block design in GNU-radio software and its corresponding implementation in a simulation environment are described, as well as the results obtained after filtering a signal contaminated with white Gaussian noise. Finally, statistical techniques were applied to analyze block performance and filtering efficiency using real-time signal processing, comparing theoretical and experimental results.

Keywords: GNURADIO, Programación, Python Blocks.

1. Introducción

Este informe describe el desarrollo de una práctica enfocada en la creación de bloques personalizados en el software GNU Radio para el procesamiento de señales en tiempo real. La práctica se centró en diseñar e implementar bloques programados específicamente para realizar tareas como la acumulación y diferenciación de señales. Estos bloques se aplicaron a señales afectadas por ruido, con el objetivo de analizar su comportamiento y su capacidad para filtrar fluctuaciones indeseadas en tiempo real. La implementación de bloques personalizados es fundamental en el procesamiento digital de señales, ya que permite un manejo eficiente y adaptable de las señales, mejorando su calidad y precisión en sistemas de comunicación.[**?**]

2. Metodología

El desarrollo del laboratorio se realizó en 4 fases metodológicas. Primero, se trabajó en GitHub. En la segunda fase, se implementaron y probaron tres tipos de bloques en Python con GNU Radio. La tercera fase consistió en crear una aplicación utilizando esos bloques. Finalmente, en la cuarta fase, se documentó el proceso y se subió el trabajo finalizado a la rama principal en GitHub.

Git Hub

Este desarrollo se estructuró en los siguientes momentos:

1. Creación de una rama correspondiente a la práctica 1.
2. Creación de subramas dentro de la rama de la práctica 1, asignadas a cada uno de los integrantes.
3. Configuración y revisión para asegurar que cada integrante esté trabajando en su respectiva subrama.

Python Blocks

En esta sección se realiza la siguientes implementaciones:

1. Implementación del bloque Acumulador.
2. Implementación del bloque Diferencia.
3. Implementación de un bloque de probabilidad que calcula el promedio, media, RMS y desviación estándar de una señal.

Aplicacción

En esta parte, realizamos una aplicación que nos permitió visualizar cómo se utilizarían estos bloques en casos más realistas y aplicados.

Git merge

En esta parte, se ilustra cómo se realizó la subida a la rama principal mediante la función merge de GitHub.

3. Resultados

Git Hub

1. Se creó una nueva rama en Git a partir de la rama main y se verificó su correcta creación mediante un push. La nueva rama se configuró con una carpeta llamada "Práctica 1"1, que contenía dos subcarpetas: una para el informe y otra para los archivos de GitHub.

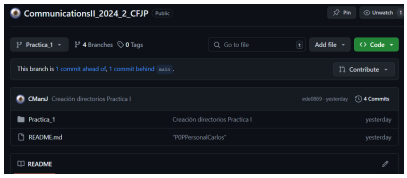

Figura 1: Rama Practica 1

2. Se crearon dos subramas basadas en la rama "Práctica 1", las cuales fueron nombradas como "P1_Nombre", donde Ṅombreẗcorresponde al primer nombre de cada integrante, para definir la zona de trabajo de cada uno.

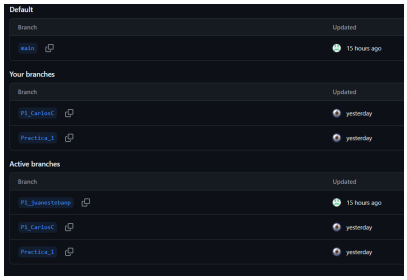

Figura 2: Vista de las ramas existentes

3. Cada integrante configuró su subrama desde Git Bash como la rama de trabajo personal.

Python Blocks

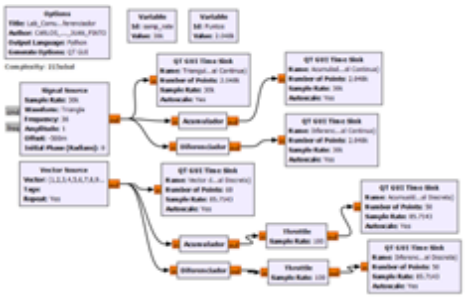

Figura 3: GNU Radio Acumulador y Divisor

1. Implementación del bloque Acumulador.

La acumulación es el proceso de sumar continuamente muestras de una señal para atenuar las oscilaciones rápidas, funcionando como un filtro pasa bajas. Esto suaviza y estabiliza la señal, reduciendo el ruido y resaltando las frecuencias bajas. Es equivalente a la integración en el tiempo discreto.

$$Y(N) = X(N) + Y(N-1) \tag{1}$$

Para implementar el bloque en GNU Radio, se utilizó el código del libro guía [1], pero fue necesario modificarlo debido a que el bloque no retornaba la salida de datos por una asignación incorrecta de la variable. Además, se añadió una variable para almacenar los datos acumulados y evitar el reinicio, lo que aseguraba una correcta captura de los valores de la señal.

Cuadro I: Bloque Acumulador

Señal	Señal Esperada	Señal Obtenida
Triangular	Sinusoidal	Sinusoidal
Cuadrada	Triangular	Triangular
[1,2,3,4]	[1,3,6,10]	[1,3,6,10]
[5,-5,5,-5]	[5,0,5,0]	[5,0,5,0]

No se lograron apreciar discrepancias entre el modelo experimental I y el teorico 1 en el acumulador, al corregir el codigo.

2. Implementación del bloque Diferencia.
La diferenciación es el proceso de calcular las diferencias entre muestras sucesivas de una señal para resaltar las oscilaciones rápidas, actuando como un filtro pasa altas. Amplifica los cambios rápidos en la señal y destaca las frecuencias altas, siendo equivalente a la derivación en el tiempo discreto.

$$Y(N) = X(N) - X(N-1) \tag{2}$$

Para implementar en GNU Radio, se usó el código del libro guía [1], pero se realizaron modificaciones para su correcto funcionamiento. Se calculó manualmente la primera diferencia de la señal y se utilizó la función "np.diff"para los valores siguientes. Además, se añadió una variable para almacenar el último valor del stream anterior.

Cuadro II: Bloque Diferenciador

Señal	Señal Esperada	Señal Obtenida
Triangular	Cuadrada	Cuadrada
Cuadrada	Impulsos en los Cambios	Señal de Pulsos
[1,2,3,4]	[1,1,1,1]	[1,1,1,1]]
[5,-5,5,-5]	[5,-10,10,-10]	[5,-10,10,-10]

Se evidencio una concordancia directa entre la ecuacion2 y el procesamiento de datos en GNUradio II.

3. Implementación del bloque que calcula promedio, media, RMS, potencia y desviación estándar.
   La probabilidad se refiere a la medida de la ocurrencia de ciertos eventos o valores dentro de una señal.

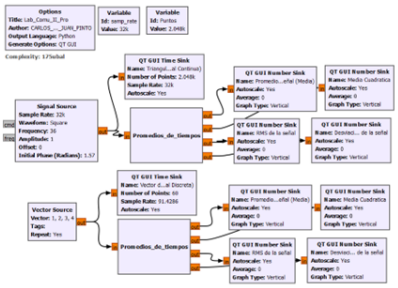

Figura 4: Estadistica

Para implementar el bloque de probabilidad, basado en el libro guía [1], se redujo el número de operaciones propuestas, ya que el libro utiliza el cálculo de la potencia promedio como si fuera el valor de la media cuadrática.

Cuadro III: Bloque Estadistica

Datos teoricos				
Offset = 0				
Señal	Promedio	mean square	RMS	Desviación
Cuadrada	0.5	0.5	0.7071	0.5
Triangular	0.5	0.335	0.579	0.292
[1,2,3,4]	2.5	7.5	2.7386	1.1180
[5,-5,5,-5]	0	25	5	5
Datos GNU Radio				
Cuadrada	0.500008	0.500008	0.707125	0.500014
Triangular	0.500003	0.333329	0.577349	0.288699
[1,2,3,4]	2.500135	7.500402	2.738686	1.118071
[5,-5,5,-5]	0	25.001263	5.000128	5.000128

No existe una discrepancia significativa entre los datos teóricos y experimentales. Sin embargo, se observa en la tabla III que la toma de datos por software es más precisa y su precisión mejora en tiempo real a medida que se recibe cada stream.

Aplicacción

La aplicación desarrollada en GNU Radio tiene como propósito general filtrar ruido blanco gaussiano por medio de un pasa bajas (Acumulador) y un pasa altas (Diferenciador). Además, se realiza un análisis de la variación de los valores en distintas secciones del proceso mediante bloques de probabilidad, con el fin de evidenciar los cambios en la señal durante el filtrado y comparar la señal obtenida con la original5.

Se generó una señal arbitraria con ruido blanco gaussiano de magnitud 500 m. Esta señal fue procesada

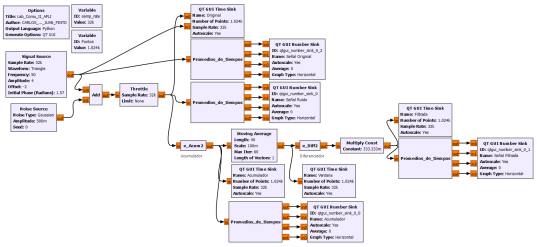

Figura 5: Aplicacion Esquimatico

por un bloque acumulador, que actúa como un filtro pasa bajas y suaviza el ruido. Sin embargo, la acumulación indefinida haría que la señal de salida creciera sin límite y también acumulara el ruido atenuado, lo que puede hacer que el ruido llegue a ser igual o mayor que el original debido al movimiento del offset por culpa del ruido cancelando la funcion del offset. Esta acumulación excesiva distorsionaría la señal y no permitiría un filtrado eficaz del ruido. Sin una limitación en la acumulación, el diferenciador al retornar la señal original restauraría las frecuencias del ruido, ya que este no habría sido filtrado adecuadamente.

Para dar solución a esta problemática, debíamos delimitar el filtrado del acumulador mediante una ventana, de tal forma que acumulara por bloques la señal en cuestión y así poder realizar correctamente la eliminación del ruido. Inicialmente planteamos esta delimitación en la misma estructura del "python block"del acumulador; sin embargo, se encontró un bloque en la interfaz de GNU Radio que cumplía esta función de una forma más sencilla y eficiente, denominado "moving average"[2]. Este bloque solo requiere determinar los parámetros de la ventana que limitarían al acumulador, además de permitir realizar una normalización sobre dicha acumulación para reducir el cambio de amplitud causado por el ruido. Con esto realizado, se pudo utilizar el bloque diferenciador, permitiendo retornar la señal original con un menor nivel de ruido y obtener así la señal arbitraria suministrada inicialmente.

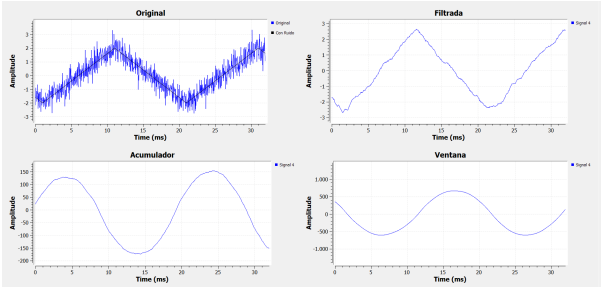

Figura 6: Aplicacion

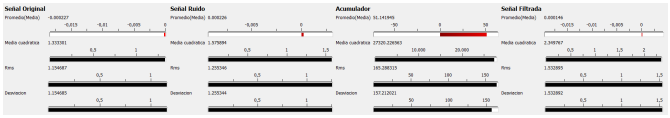

Figura 7: Aplicacion Estadistica

Analizando la señal desde el punto de vista de los valores de probabilidad obtenidos entre la señal original y la señal filtrada, podemos visualizar que el valor promedio y la desviación estándar entre las dos señales no presentan variaciones muy considerables. Sin embargo, el valor RMS y la media cuadrática sí presentan diferencias. Esto se debe a que, a pesar de que el ruido en la señal fue atenuado, no fue eliminado por completo. Este ruido ocasiona que la energía correspondiente a la señal aumente, así como las fluctuaciones que presenta, provocando un aumento en el valor RMS y en la media cuadrática de la señal. No obstante, estos valores no son tan alejados, lo que permite deducir que se ha logrado atenuar un valor considerable del ruido mediante este procedimiento.

Git merge

Siguiendo la documentación [3], se llevaron a cabo los pasos correspondientes, resaltando que primero era necesario establecer como base la rama con la cual se deseaba realizar la fusión. Posteriormente, se ejecutó el comando git merge, obteniendo una unión exitosa sin errores.

4. Conclusiones

- Durante la configuración de los bloques de acumulación y de diferencia, se comprendió la discrepancia entre vectores y streams. Se observó que, al producirse un cambio en el tamaño de los datos, era necesario revisar constantemente dicho tamaño para poder almacenar la información requerida de manera adecuada.
- El uso de ramas permitió una gestión eficiente en el desarrollo de la práctica. Esto facilitó que cada integrante pudiera trabajar en su propio espacio de trabajo en paralelo al proyecto original, sin causar inconvenientes a los demás compañeros. De este modo, se pudo lograr un mejor resultado en el trabajo colaborativo.
- Los bloques personalizados en GNU radio ofrecen al usuario una heramienta adaptable, flexible y eficaz a la hora de realizar proyectos de procesamiento de señales, ya que permite a los usuarios diseñar soluciones específicas, eficientes y reutilizables.
- El uso del bloque "Moving Average"permitió implementar la limitación requerida en el bloque acumulador, lo que facilitó la observación de su comportamiento como filtro pasa-bajas. Este bloque no solo realiza un promedio móvil, sino que también normaliza los datos, generando ventanas de datos específicos y contribuyendo así al filtrado efectivo de la señal.
- Se logro apreciar por medio de una variacion en el codigo del diferenciador y el acumulador como varian las señales stream al momento de su envio y recepcion, evidenciando la variacion de los arrays correspondientes a estos en el software de GNUradio.
- Los bloques de probablidad permiten analizar parametros especificos en las señales aleatorias, permitiendo asi analizar diversos tipos de señales.

Referencias

[1] H. O. B. y Oscar Mauricio Reyes Torres, “Comunicaciones digitales basadas en radio definida por software.” *ESCUELA DE INGENIER IAS ELECTRICA ELECTR ONICA Y TELECOMUNICACIONES–E3T UNIVERSIDAD INDUSTRIAL DE SANTANDER – UIS*, 19 de octubre del 2019.

[2] W. G. RADIO. Moving average. [Online]. Available: https://wiki.gnuradio.org/index.php/Moving_Average

[3] GIT. Git - git-merge documentation. [Online]. Available: https://git-scm.com/book/es/v2/Ramificaciones-en-Git-Procedimientos-B%C3%A1sicos-para-Ramificar-y-Fusionar