Departmento de Ingeniería Eléctrica
Facultad de Ciencias Físicas y Matemáticas
Universidad de Chile
EL4112-1 Principios de Comunicaciones

Principios de Comunicaciones

Transmisión Inalámbrica Digital de Imágenes y Texto Empleando Portadoras Audibles

Proyecto Nº 2

Integrantes:
Mattias Prietto
Ignacio Romero

Profesor:
Cesar Azurdia

Auxiliares:
Giovanni Castiglioni
Cesar Azurdia

Ayudantes:
Vicente Aitken
Martín Moreno
Ariel Núñez
Simón Repolt

Entrega:
15 de diciembre de 2023

Índice

1. Introducción

En el fascinante campo de las telecomunicaciones, la continua búsqueda de metodologías más eficientes para transmitir información ha sido el foco principal de la investigación. La modulación, una técnica crucial en este contexto, implica la asociación de variables como amplitud, frecuencia o fase a cada bit o conjunto de bits de una señal, con el propósito de generar una señal modulada que encapsule toda la información relevante para facilitar su transmisión a través de un canal. La demodulación, en contraste, consiste en el proceso inverso, donde se reconstruyen los paquetes de información codificados en la señal modulada.

El objetivo de este informe, es presentar los resultados de un proyecto que busca transmitir, entre dos computadores (enlace SISO), la información digital de una imagen a color y texto usando como canal inalámbrico portadoras audibles (sonido).

2. Marco teórico

En el ámbito de las telecomunicaciones, la investigación se centra principalmente en la búsqueda de diversas y más eficientes metodologías para transmitir información. Una técnica crucial en este contexto es la modulación, que implica asociar alguna variable, como amplitud, frecuencia o fase, a cada bit o conjunto de bits de una señal. Este proceso tiene como objetivo generar una señal modulada que encapsule toda la información relevante, facilitando su transmisión a través de un canal. La demodulación, por otro lado, consiste en el proceso inverso, donde se reconstruyen los paquetes de información codificados en la señal modulada.

La modulación por desplazamiento de frecuencia (FSK), es una técnica de modulación para la transmisión digital de información que utiliza dos o más frecuencias diferentes para cada símbolo. La señal moduladora solo varía entre dos valores de tensión discretos formando un tren de pulsos donde son representadas por 1 (representa la marca) y 0 (representa el espacio).

FSK es una modulación de frecuencia donde la señal moduladora (datos) es digital. Los dos valores binarios se representan con dos frecuencias diferentes (f1 y f2) próximas a la frecuencia de la señal portadora.

$$v(t) = \begin{cases} V_p \ \text{sen}(2\pi \ f_1 \ t) \text{ para un "1" binario} \\ V_p \ \text{sen}(2\pi \ f_2 \ t) \text{ para un "0" binario} \end{cases}$$

Figura 1: Ecuación FSK [1].

Generalmente f1 y f2 corresponden a desplazamientos de igual magnitud, pero en sentidos opuestos de la frecuencia de la señal portadora.

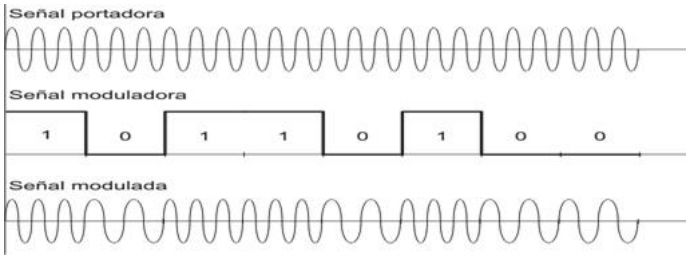


Figura 2: Ilustración de la señal [1].

Para demodular una señal FSK, se emplea un detector de frecuencia que es capaz de distinguir entre las frecuencias utilizadas para representar los bits. Este detector se encarga de identificar las variaciones de frecuencia en la señal FSK.

Después de pasar por el detector de frecuencia, la señal se dirige a un filtro de paso de banda (BPF). Este filtro permite únicamente el paso de las frecuencias específicas asociadas a los valores binarios. Al filtrar las frecuencias no deseadas, se mejora la calidad de la señal demodulada.

Por otro lado, en la técnica de modulación de señal PAM la información a transmitir, señal de mensaje, se codifica en la amplitud de los pulsos de señal. La amplitud de la señal portadora de mayor frecuencia varía de acuerdo con la señal de información. En el proceso de demodulación de la señal, en cada período de símbolo instantáneo la señal original es obtenida del nivel de amplitud de la señal portadora.1

La señal modulada PAM se da del producto de una señal analógica continua por un tren de pulsos de amplitud constante, de la cual se obtiene como resultado un tren de pulsos modulado en amplitud. En los siguientes esquemas se muestra como se obtiene la señal modulada PAM y el comportamiento de las respectivas señales [2].

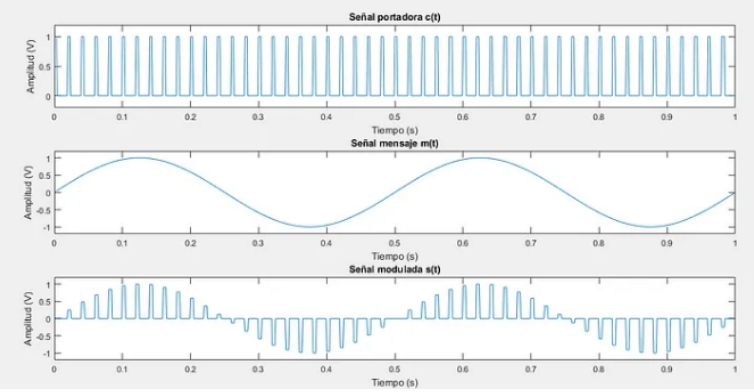


Figura 3: Señal portadora, señal de mensaje y señal PAM modulada [2].

Para la demodulación de una señal PAM, esta debe pasarse por un filtro pasa baja. Este filtro elimina las señales de alta frecuencia generando así, la señal demodulada. A esta señal se le debe aplicar un amplificador inversor para aumentar el nivel de la señal y así obtener la salida demodulada con un valor de amplitud muy parecido a la señal de modulación. En el siguiente esquema se muestra cómo se agrega el LPF a la señal modulada para obtener la señal original. Seguidamente, se muestra el comportamiento de estas señales [2].

3. Descripción de la implementación

En este proyecto, se ha desarrollado un sistema de transmisión de información que utiliza señales audibles para codificar y transmitir tanto imágenes como texto. El código implementado en MATLAB aborda esta tarea mediante una serie de pasos coherentes. De manera general, se puede ver la organización del proyecto en el diagrama de la figura 4, donde cada bloque verde representa un script en MATLAB diferente.

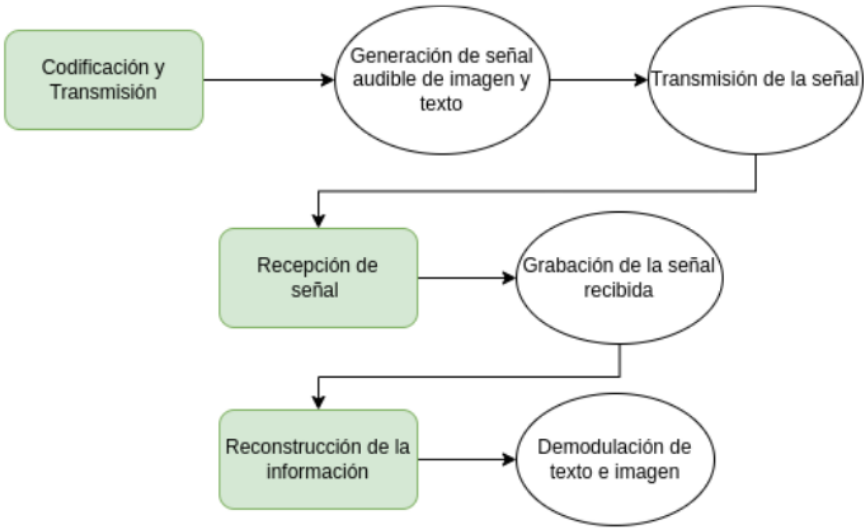


Figura 4: Diagrama de bloques del proyecto.

3.1. Codificación y Transmisión

3.1.1. Transmisión de Imágenes

El proceso de transmisión de imágenes se inicia con la lectura de una imagen en formato PNG. Esta imagen se descompone en sus componentes RGB (rojo, verde y azul), y cada canal se convierte en un vector unidimensional (`v_redChannel`, `v_greenChannel`, `v_blueChannel`). La visualización gráfica de estos canales proporciona una comprensión visual de la información.

```
% Lectura y procesamiento de imagenes

image = imread('img1.png');

% Separar en matrices RGB
[redChannel, greenChannel, blueChannel] = imsplit(image);


```

```
%Se crean vectores unidimensionales de las matrices RGB de cada color
v_redChannel=reshape(redChannel,400,1);
v_greenChannel=reshape(greenChannel,400,1);
v_blueChannel=reshape(blueChannel,400,1);

%Se reconstruyen los canales de color independientes
emptyChannel = zeros(20, 20, 'uint8');
just_redColor = cat(3, redChannel, emptyChannel, emptyChannel);
just_greenColor = cat(3, emptyChannel, greenChannel, emptyChannel);
just_blueColor = cat(3, emptyChannel, emptyChannel, blueChannel);
all_imageColor = cat(3, redChannel, greenChannel, blueChannel);
subplot(1,4,4);
imshow(all_imageColor);
subplot(1,4,3);
imshow(just_blueColor);
subplot(1,4,2);
imshow(just_greenColor);
subplot(1,4,1);
imshow(just_redColor);
```

A continuación, se asigna un espectro de frecuencias específico a cada canal de color. Este espectro se divide en 256 tonos, cada uno correspondiente a un nivel de intensidad de color. La información de cada píxel se codifica en una señal sinusoidal generada a partir de estas frecuencias. Finalmente, las señales de los tres canales se combinan en una única señal (`s1`) que representa la información visual completa.

```
% Generacion de senales para cada canal de color
step = 12;
f_muestreo = 50000;
m_pixel = 8000;
time_header = 1;
symbols = 255;
f1 = 6000;
f2 = 7000;
f3 = 8000;

s_red = [];
s_green = [];
s_blue = [];

tiempo_envio = m_pixel / f_muestreo;
t1 = 0:1/f_muestreo:tiempo_envio-1/f_muestreo;
frec = 0:step:255*step;

init_red = 2000;
for i = 1:400
    y_red = cos(2*pi*(frec(redChannel(i)+1)+init_red)*t1);
    s_red = [s_red, y_red];
end

init_green = init_red + symbols*step + 50;
for i = 1:400
    y_green = cos(2*pi*(frec(greenChannel(i)+1)+init_green)*t1);
    s_green = [s_green, y_green];
end

init_blue = init_green + symbols*step + 50;
for i = 1:400
    y_blue = cos(2*pi*(frec(blueChannel(i)+1)+init_blue)*t1);
```

```
    s_blue = [s_blue, y_blue];
end

s1 = s_red + s_green + s_blue;
```

3.1.2. Transmisión de Texto

La transmisión de texto se realiza mediante la codificación de caracteres en señales audibles. Se define un conjunto de símbolos que abarca letras, números y espacios. Cada símbolo se asigna a una frecuencia única, creando así una correspondencia entre caracteres y frecuencias audibles. El mensaje de texto se codifica en una secuencia de señales, y esta señal de texto (`s_str`) se agrega a la señal de imagen (`s1`).

```
% Codificacion y transmision de texto
simbolos = 'abcdefghijklmnnopqrstuvwxyz ABCDEFGIJKLMNNOPQRSTUVWXYZ0123456789';
n = length(simbolos);
posicion_simbolos = 0:n-1;
vector_simbolos = zeros(1, n);

mensaje = 'Principios de Comunicaciones primavera 2023 EL4112';
vector_mensaje = zeros(1, length(mensaje));

for i = 1:n
    vector_simbolos(i) = simbolos(i);
end

for i = 1:length(mensaje)
    vector_mensaje(i) = mensaje(i);
end

largo_mensaje = length(mensaje);
t_str = 0:1/f_muestreo:m_pixel/f_muestreo-1/f_muestreo;
s_str = [];
vector_ASCII = [];
espectro_str = 2000;

for i = 1:largo_mensaje
    posicion = find(vector_simbolos == vector_mensaje(i));
    vector_ASCII = [vector_ASCII, posicion];
    y = cos(2*pi*(step*(posicion-1)+espectro_str)*t_str);
    s_str = [s_str, y];
end
s1 = [s1, s_str];
```

3.1.3. Generación de Señales y Transmisión

Para asegurar la sincronización durante la transmisión, se generan señales de sincronización (`s_sincr1`, `s_sincr2`, `s_sincr3`). Estas señales marcan claramente el inicio de la transmisión y se combinan con la señal de imagen y texto para formar la señal final de transmisión (`s1`). La longitud temporal de estas señales se ajusta cuidadosamente para garantizar la coherencia temporal durante la transmisión.

```
% Generacion de senales de sincronizacion y transmision
T = 1/f_muestreo;
t_s = 0:T:time_header-T;
s_sincr = [];
s_sincr1 = cos(2*pi*f1*t_s);
s_sincr2 = cos(2*pi*f2*t_s);
s_sincr3 = cos(2

*pi*f3*t_s);
s_sincr = [s_sincr, s_sincr1, s_sincr2, s_sincr3];
s1 = [s_sincr, s1];
```

3.1.4. Reproducción del Sonido

La reproducción del sonido se realiza mediante la función `soundsc`, que reproduce la señal completa (`s1`) con la frecuencia de muestreo especificada (`f_muestreo`). Este paso finaliza el proceso de transmisión, permitiendo que la información codificada en las señales sea audible para el oyente.

```
% Reproduccion del sonido
soundsc(s1, f_muestreo);
```

En conjunto, este sistema integral demuestra la capacidad de transmitir información visual y textual de manera eficiente y coherente utilizando señales audibles. La modularidad y claridad en cada fase del proceso permiten una fácil adaptación y expansión para futuras aplicaciones.

3.2. Grabación del sonido

En esta sección, se describe el código encargado de grabar sonido y visualizar la forma de onda resultante. Es importante destacar que, esta parte de la implementación es necesario reproducirla en un computador diferente al de la subsección 3.1.

```
% Parametros recorder

f_muestreo = 50000;
recObj = audiorecorder(f_muestreo, 16, 1);
seconds = 110;

% Record

disp('Recording.')
recordblocking(recObj, seconds);
disp('End of Recording.');

% Se guarda el audio

z = getaudiodata(recObj);
w = z;

% Se grafica el audio

```

```
t = 0:1/f_muestreo:seconds-1/f_muestreo;
plot(t, w)
```

Este código utiliza la función 'audiorecorder' para configurar un objeto de grabación de audio con una frecuencia de muestreo de 40,000 Hz, una precisión de 16 bits y un solo canal. Luego, inicia la grabación durante 110 segundos con la función 'recordblocking'. Después de la grabación, se obtienen los datos de audio con 'getaudiodata'. La forma de onda se almacena en el vector 'z' y se visualiza utilizando la función 'plot'.

Este código complementa el sistema de transmisión de información, proporcionando una manera de capturar y visualizar el sonido transmitido.

3.3. Reconstrucción de la información

El siguiente código se encarga de procesar la señal de audio grabada para recuperar la información de colores y texto transmitida previamente. El proceso se divide en varias etapas, cada una explicada a continuación.

3.3.1. Parámetros Iniciales

```
red_Channel = [];
green_Channel = [];
blue_Channel = [];
str_Channel = [];

m_pixel = 8000;
fs = 50000;
T = 1/fs;
step = 12;

s = z;
```

Aquí se inicializan variables esenciales, como el tamaño de muestra por pixel (`m_pixel`), la frecuencia de muestreo (`fs`), el paso para la codificación de colores y texto (`step`), y se carga la señal de audio (`s`) previamente grabada.

3.3.2. Sincronización y Header

```
f1 = 6000;
f2 = 7000;
f3 = 8000;
header_time = 1; % segundos
ts = 0:T:header_time-T;

% Se define la senal de sincronizacion

```

```
s_sincr1 = cos(2*pi*f1*ts);
s_sincr2 = cos(2*pi*f2*ts);
s_sincr3 = cos(2*pi*f3*ts);
s_sincr = [s_sincr1, s_sincr2, s_sincr3];

% Se realiza una autocorrelacion en los primeros 30 segundos para encontrar la senal de sincronizacion
% en el audio recibido

header_test=header_time*30;

signal = s(1: header_test*fs);
[acorr, lag] = xcorr(signal, s_sincr);
[~,I] = max(abs(acorr));
lagDiff = I-(length(signal)-length(ts)*3);
```

En esta sección, se identifica el inicio de la señal mediante la correlación cruzada entre la señal grabada y la señal de sincronización (`s_sincr`). Se busca el desplazamiento (`lagDiff`) que maximiza la correlación, permitiendo la sincronización adecuada.

3.3.3. Demodulación de Colores

```
for i = 1:400
    y = s(m_pixel*(i-1)+lagDiff:m_pixel*i+lagDiff);  % filtr. por cant. de muestras para cada pixel
    [i_red, i_blue, i_green] = frec_colores(y, fs, step);

    red_Channel = [red_Channel, i_red];
    green_Channel = [green_Channel, i_green];
    blue_Channel = [blue_Channel, i_blue];
end

% Se reconstruye la imagen
m_red = uint8(reshape(red_Channel,20,20));
m_green = uint8(reshape(green_Channel,20,20));
m_blue = uint8(reshape(blue_Channel,20,20));

imagenRGB = cat(3, m_red, m_green, m_blue);
```

En esta etapa, se demodulan los colores de la señal grabada. Se filtra la señal para cada pixel, y se calculan las frecuencias predominantes en las bandas de color rojo, verde y azul mediante la función auxiliar `frec_colores`. Los resultados se almacenan en matrices 20x20 para cada canal de color.

3.3.4. Visualización de Imagen Decodificada

```
subplot(1,4,1);
imshow(cat(3, m_red, zeros(20,20,'uint8'), zeros(20,20,'uint8')));
title('Red');
subplot(1,4,2);
imshow(cat(3, zeros(20,20,'uint8'), m_green, zeros(20,20,'uint8')));
title('Green');
subplot(1,4,3);
```

```
imshow(cat(3, zeros(20,20,'uint8'), zeros(20,20,'uint8'), m_blue));
title('Blue');
subplot(1,4,4);

imagen_decodificada = cat(3, m_red, m_green, m_blue);
imshow(imagen_decodificada);
title('Imagen decodificada');
```

En esta sección, se visualiza la imagen decodificada separando los canales de color y presentando la imagen resultante. Cada canal se muestra en un subplot individual, y la imagen completa se muestra en otro subplot.

3.3.5. Demodulación de Texto

```
for i = 401:450
    s_prima = s(m_pixel*(i-1) + lagDiff:m_pixel*i + lagDiff);
    y = s_prima;
    str_Channel = [str_Channel, frec_str(y, fs, step)];
end
disp(str_Channel)
```

En esta fase, se demodula el texto de la señal grabada. Se extraen las frecuencias asociadas al texto mediante la función auxiliar `frec_str`.

3.3.6. Funciones Auxiliares

```
function [str] = frec_str(y, fs, step)
espectro_str = 2000;

simbolos='abcdefghijklmn opqrstuvwxyz ABCDEFGIJKLMNOPQRSTUVWXYZ0123456789';
n = length(simbolos);
posicion_simbolos = 1:n ;

%se crean filtros para cada banda de frecuencias
n=5; %orden de los filtros
norma = fs/2;
Wd_r=(espectro_str-25)/norma;
Wu_r=(espectro_str+65*step+25)/norma;
[b,a]=butter(n,[Wd_r,Wu_r],'bandpass');
str_filt = filter(b,a,y);

y_frecuencias_str1 = abs(fft(str_filt,fs));
y_frecuencias1 = y_frecuencias_str1(1:(length(y_frecuencias_str1))/2);
y_frecuencias_str=y_frecuencias1(espectro_str-25:(espectro_str+65*step)+25);
[maxValue_str, maxIndex_str] = max(y_frecuencias_str);


if maxIndex_str<25
    f_str=1;

elseif maxIndex_str <= length(y_frecuencias_str)-25
    f_str = round((maxIndex_str-25)/step)+1;
```

```
27
28  else
29      f_str=63+1;
30  end
31  %disp(simbolos(f_str));
32  str=simbolos(f_str);
33
34  end
35
36  function [i_red, i_blue, i_green] = frec_colores(y, fs, step)
37
38      symbols=255;
39      init_red=2000;
40      order_filter=10;
41      sred_filt = filter_butterworth(y, fs, order_filter, init_red, init_red+symbols*step);
42
43      y_frecuencias_r1=abs(fft(sred_filt,fs));
44      y_frecuencias_r=y_frecuencias_r1(1:length(y_frecuencias_r1)/2);
45      y_frecuencias_red=y_frecuencias_r(init_red-25:init_red+symbols*step+25);
46
47      init_green = init_red+symbols*step+50;
48      sgreen_filt = filter_butterworth(y, fs, order_filter, init_green, init_green+symbols*step);
49
50      y_frecuencias_g1 = abs(fft(sgreen_filt,fs));
51      y_frecuencias_g = y_frecuencias_g1(1:length(y_frecuencias_g1)/2);
52      y_frecuencias_green=y_frecuencias_g(init_green-25:init_green+symbols*step+25);
53
54      init_blue = init_green+symbols*step+50;
55      sblue_filt = filter_butterworth(y, fs, order_filter, init_blue, init_blue+symbols*step);
56
57      y_frecuencias_b1 = abs(fft(sblue_filt,fs));
58      y_frecuencias_b = y_frecuencias_b1(1:length(y_frecuencias_b1)/2);
59      y_frecuencias_blue=y_frecuencias_b(init_blue-25:init_blue+symbols*step+25);
60
61      [~, maxIndex_red] = max(y_frecuencias_red);
62      [~, maxIndex_green] = max(y_frecuencias_green);
63      [~, maxIndex_blue] = max(y_frecuencias_blue);
64
65      %disp(init_blue+symbols*step+25)
66
67      if maxIndex_red<25
68          i_red=0;
69
70      elseif maxIndex_red <= length(y_frecuencias_red)-25
71          i_red = round((maxIndex_red-25)/step);
72
73      else
74          i_red=255;
75      end
76          if maxIndex_green<25
77          i_green=0;
78
79      elseif maxIndex_green <= length(y_frecuencias_green)-25
80          i_green = round((maxIndex_green-25)/step);
81
82      else
83          i_green=255;
84      end
85
86
87      if maxIndex_blue<25
88          i_blue=0;
```

```
89
90      elseif maxIndex_blue <= length(y_frecuencias_blue)-25
91          i_blue = round((maxIndex_blue-25)/step);
92
93      else
94          i_blue=255;
95
96      end
97  end
98
99  function [filtered_signal] = filter_butterworth(signal, fs, order, f_low, f_high)
100     [b, a] = butter(order, [f_low, f_high] * 2 / fs, 'bandpass');
101     filtered_signal = filter(b,a,signal);
102 end
```

3.3.7. Función `frec_str`

La función `frec_str` se encarga de extraer la información de texto demodulada a partir de una señal dada. A continuación, se presenta una descripción detallada de su funcionamiento:

- **Parámetros:** La función toma tres parámetros como entrada:
  - `y`: La señal de entrada.
  - `fs`: La frecuencia de muestreo de la señal.
  - `step`: El paso utilizado en la codificación de frecuencias.
- **Proceso:** La función realiza los siguientes pasos:
  a. Aplica un filtro Butterworth de paso de banda para aislar la banda de frecuencias asociada al texto.
  b. Calcula la transformada de Fourier de la señal filtrada.
  c. Identifica la frecuencia máxima en la banda de interés.
  d. Mapea la frecuencia máxima a un símbolo en el conjunto de símbolos disponibles.
- **Salida:** La función devuelve el símbolo de texto asociado a la frecuencia identificada.

3.3.8. Función `frec_colores`

La función `frec_colores` se encarga de demodular la información de colores a partir de una señal dada. A continuación, se presenta una descripción detallada de su funcionamiento:

- **Parámetros:** La función toma tres parámetros como entrada:
  - `y`: La señal de entrada.
  - `fs`: La frecuencia de muestreo de la señal.

  - `step`: El paso utilizado en la codificación de frecuencias.

- **Proceso:** La función realiza los siguientes pasos:

  a. Aplica filtros Butterworth de paso de banda para cada canal de color (rojo, verde y azul).
  b. Calcula las transformadas de Fourier de las señales filtradas para cada canal.
  c. Identifica las frecuencias máximas en las bandas de interés para cada canal.
  d. Mapea las frecuencias máximas a niveles de intensidad de color (0-255) para cada canal.

- **Salida:** La función devuelve los niveles de intensidad de color demodulados para los canales rojo, verde y azul.

3.3.9. Función `frec_colores`

La función `frec_colores` se encarga de demodular la información de colores a partir de una señal dada. A continuación, se presenta una descripción detallada de su funcionamiento:

- **Parámetros:** La función toma tres parámetros como entrada:

  - `y`: La señal de entrada.
  - `fs`: La frecuencia de muestreo de la señal.
  - `step`: El paso utilizado en la codificación de frecuencias.

- **Proceso:** La función realiza los siguientes pasos:

  a. Aplica filtros Butterworth de paso de banda para cada canal de color (rojo, verde y azul).
  b. Calcula las transformadas de Fourier de las señales filtradas para cada canal.
  c. Identifica las frecuencias máximas en las bandas de interés para cada canal.
  d. Mapea las frecuencias máximas a niveles de intensidad de color (0-255) para cada canal.

- **Salida:** La función devuelve los niveles de intensidad de color demodulados para los canales rojo, verde y azul.

Este código integral demuestra cómo procesar una señal de audio grabada para recuperar información de colores e imágenes codificadas, y cómo visualizar y presentar la información decodificada. Las funciones auxiliares facilitan la modularidad y comprensión del código. Cada sección contribuye a la recuperación exitosa

4. Resultados

Al realizar el experimento en reiteradas ocasiones, se obtuvo que, en más del 80 % de los casos la información transmitida tenía menos de 5 % de error. En este sentido, a continuación, se presentarán los resultados de la transmisión de imagen y texto perfecta (0 % de error): Una vez transmitida la señal codificada, el receptor guardó la señal de la figura 5:

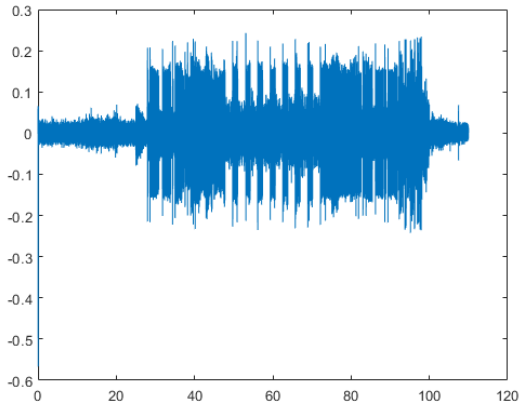
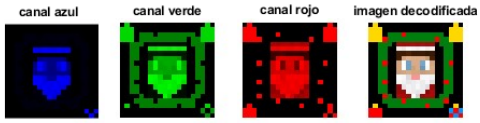


Figura 5: Señal recibida.

Al decodificar esta señal, se obtuvieron la imágenes y el texto de la figura 6:



(a) Imagen recibida.

```
>> decodingRGB
Principios de Comunicaciones primavera 2023 EL4112
```

(b) Texto recibido.

Figura 6: Información decodificada.

Además, al separar los canales de color se obtuvieron los histogramas de la figura 7:

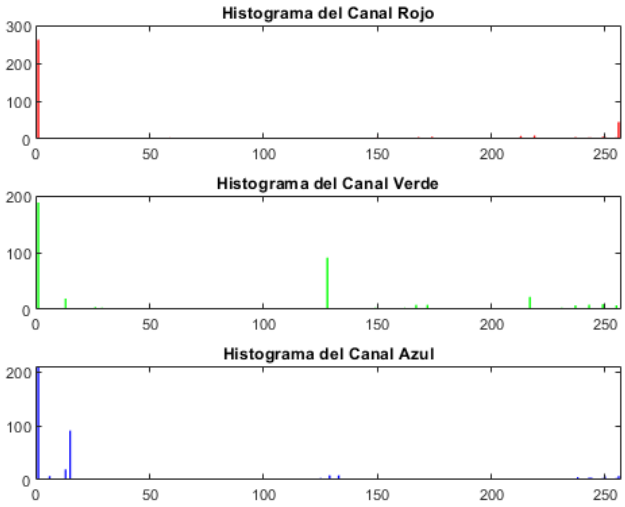


Figura 7: Histograma de color de la imagen recibida.

En contraste con lo anterior, a continuación, en la figura 8 se presentan los resultados de una medición, artificialmente, muy ruidosa. El ruido agregado fue provocado al golpear el micrófono durante la transmisión, hacer sonar un patito de goma, y poner un video de personas hablando, a volumen moderado, junto al micrófono.

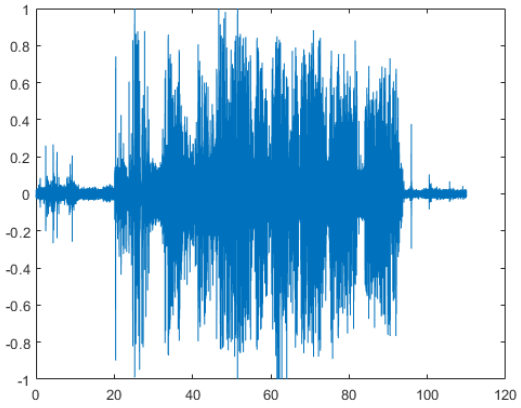
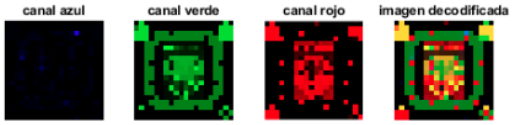


Figura 8: Señal recibida con mucho ruido.

Así, los mensajes decodificados de la señal anterior se pueden ver representados en la figura 9:



(a) Imagen recibida.

```
>> decodingRGB
Principios de Comunicaciones primavera 2023 ELq1lw
```

(b) Texto recibido.

Figura 9: Información decodificada de la señal ruidosa.

Al analizar el histograma por color de la imagen recibida en este caso, se obtienen las gráficas de la figura 10:

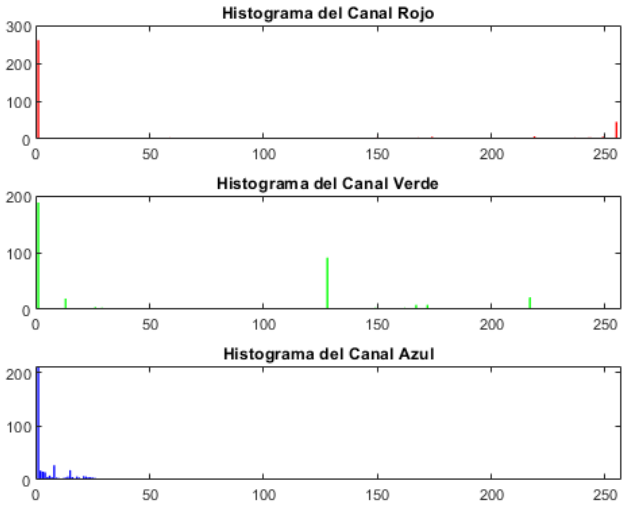


Figura 10: Histograma de color de la imagen recibida.

5. Conclusiones

El equipo logró un hito significativo al realizar una exitosa transmisión de imágenes y texto utilizando portadoras audibles mediante las técnicas de modulación M-FSK y M-PAM. Los resultados obtenidos demostraron la eficiencia del sistema en la comunicación de datos a través de canales de audio, destacándose la robustez de la implementación al transmitir información sin errores aparentes.

A pesar de los logros prácticos, es crucial reconocer que el equipo enfrentó una limitación en la comprensión teórica del proyecto. Aunque se alcanzó un éxito práctico, la falta de una profundización más rigurosa en la teoría subyacente impidió una comprensión completa de los fundamentos técnicos de las implementaciones. Esta carencia resulta fundamental para establecer una conexión sólida entre los logros obtenidos y los principios teóricos discutidos en las clases.

En resumen, aunque se celebra el éxito práctico del proyecto en términos de transmisión efectiva, se reconoce la necesidad de un mayor énfasis en la comprensión teórica. La dualidad entre el éxito práctico y la falta de profundización teórica destaca la importancia de equilibrar la aplicación práctica con una comprensión rigurosa de los fundamentos conceptuales. Abordar esta brecha teórica no solo mejorará la capacidad del equipo para enfrentar desafíos futuros, sino que también consolidará el aprendizaje, permitiendo aplicar de manera más informada los conocimientos adquiridos en las clases.

Bibliografía

[1] R. R. L. Judith y Z. B. J. Gisella. "Transmisión por desplzamiento de Frecuencias." (), dirección: https://www.monografias.com/docs113/transmision-dezplazamiento-frecuencia/transmision-dezplazamiento-frecuencia (visitado 14-12-2023).

[2] S. Cortez. "PAM." (), dirección: https://medium.com/modulaciones-de-pulsos-muestreo-pam-ppm-pcm-y/pam-a5e6d926fcae (visitado 14-12-2023).