TRANSMISIÓN DE DATOS 2008/09			
Práctica 3: Sistema Codificación- Decodificación		**Grupo**	
		Puesto	
Apellidos, nombre		**Fecha**	
Apellidos, nombre		**10/12/13 de Noviembre 2008**	

El objetivo de esta práctica es el de familiarizar al alumno con el proceso completo de codificación-decodificación de fuente así como la experimentación utilizando ejemplos de transmisión a través de un canal con ruido simulado.

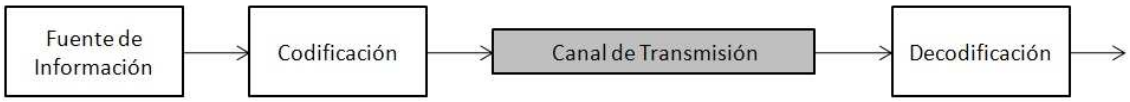
En el fichero Cod-Decod`.zip` se encuentra el código y ficheros de prueba necesarios para la realización de la práctica. Para ello descomprima y copie los ficheros al directorio de realización de la práctica.

Justo antes de finalizar la práctica, comprima el proyecto en un fichero TxDatosP3GXzz.zip (siendo X el grupo –A, B ó C-, y zz el número de pareja) conéctese al sistema de entrega de prácticas de la Intranet y entréguelo en el grupo que corresponda (A, B o C). Guárdese adicionalmente una **copia personal**, para posibles futuras reutilización del código en prácticas posteriores. Recuerde borrar su trabajo del ordenador del puesto de prácticas.

1 Introducción

En esta práctica se llevará a cabo la implementación del proceso completo de codificación de fuente, simulación de transmisión de la información a través de canales con y sin ruido y finalmente la decodificación de la información transmitida.

Para ello se implementarán gradualmente varios sistemas siguiendo el siguiente esquema genérico:



Los sistemas a desarrollar serán los siguientes:

- Codificación/decodificación de cadenas de texto mediante codificación *Huffman*.
- Codificación/decodificación de muestras de audio mediante codificación PCM.
- Codificación/decodificación de una señal mediante combinación de codificación PCM y *Huffman*.

En la primera parte de la práctica el alumno se familiarizará con las funciones proporcionadas para la simulación de transmisión de información con errores. Posteriormente se llevará a cabo la implementación y pruebas del conjunto de sistemas propuestos. En todos los casos se llevará a cabo primero la implementación de la secuencia completa de codificación y decodificación sin ruido. A continuación se realizará la simulación de un canal de transmisión ruidoso para estudiar el impacto de las distintas modalidades de ruido (con o sin ráfagas) en la decodificación.

1.1 *Transmisión de la Información*

En todos los casos se simulará el envió de información como cadenas de bits representadas en forma de vectores fila Matlab. En cada posición se indicará el valor del bit correspondiente (0 ó 1). Por tanto la salida de los módulos de codificación (*Huffman* y PCM) será un vector de longitud *n* (bits). Así mismo, los módulos de decodificación recibirán vectores de bits y devolverán el resultado de la decodificación en forma de vectores en los que cada posición representará un valor numérico. *Nota: No confundir los vectores de bits (con valores 1 ó 0) con los vectores numéricos (en los que cada posición representa un número y, por tanto, también pueden contener los valores 1 y 0).*

2 Simulación de canales con error

Para la simulación de la transmisión de un vector de *n* bits a través de un canal ruidoso generaremos secuencias aleatorias de longitud *n* (las llamaremos máscaras de ruido o de error) en las que cada posición indicará si el *bit* de la secuencia original en esa misma posición ha sufrido un error de transmisión (máscara de ruido con valor = 1) o se ha transmitido correctamente (máscara de ruido con valor = 0). Cuando se produzca un error de transmisión en una posición *i* el bit correspondiente deberá cambiar de valor (de 0 a 1 ó viceversa). Esto se puede llevar a cabo mediante una operación *Xor* (*Or* exclusivo) o mediante una suma módulo 2 del vector original y la máscara de ruido. En la siguiente tabla se muestra un vector de 16 bits de información, una máscara de error aleatoria y el resultado de una transmisión con el patrón de error especificado en la máscara:

Información	1	0	0	**0**	**1**	0	1	**1**	0	1	0	**1**	0	**1**	1	0
Máscara de Error	0	0	0	**1**	**1**	0	0	**1**	0	0	0	**1**	0	**1**	0	0
Resultado	1	0	0	**1**	**0**	0	1	**0**	0	1	0	**0**	0	**0**	1	0

Para la generación de secuencias aleatorias de error se proporciona la función `generarRuido`[1] con la siguiente especificación:

```
function [mascaraRuido] = generarRuido(duracion, probabilidadRafaga,longitudMediaRafaga,
                                       variacionLongitudRafaga, semillaAleatoria)
```

- **Recibe:**
  - duracion == Longitud de la máscara/patrón de error que se desea generar.
  - probabilidadRafaga == Indica la probabilidad de que en una posición cualquiera de la secuencia se produzca un error.
  - longitudMediaRafaga == Indica el número de bits a lo largo de los cuales se va a propagar el error en caso de producirse en un bit determinado. Esto es, la longitud de la ráfaga de error.
  - variacionLongitudRafaga == Indica la variabilidad en la longitud de un error. Si se especifica valor 0 todos los errores que se produzcan en la secuencia se prolongarán a lo largo de 'longitudMediaRafaga' bits. Si se especifica otro valor la longitud de las ráfagas de error podrán variar en el margen indicado.
  - semillaAleatoria == Este valor indicará la semilla con la que se inicializará el generador de números aleatorios. De esta forma, indicando los mismos parámetros y semilla, podremos reproducir las secuencias de error generadas por esta función. El resultado es pues pseudoaletorio.
- **Devuelve:**
  - mascaraRuido == La máscara de ruido generada en la que los bits a '1' indicarán que se ha producido un error en la comunicación y los bits a '0' lo contrario.

[1] Observese que el caso concreto de: longitudMediaRafaga==1 y variacionLongitudRafaga==0 es equivalente a generar el error en un solo bit.

2.1 Ejercicio 1: Simulación de un canal con error

Complete el *script* Matlab de nombre *Ejercicio1.m* proporcionado en el fichero de la práctica en el que se lleve a cabo la simulación de transmisión con error de la secuencia *vectorTransmision* que se proporciona previamente inicializada. Para ello genere una secuencia de error de tamaño adecuado con los siguientes parámetros: *probabilidadRafaga=0.15*, *longitudMediaRafaga=1*, *variacionLongitudRafaga=0* y *semillaAleatoria=13.* A continuación calcule y muestre por pantalla la cadena original, máscara de error generada y el resultado de aplicar dicha máscara de error al vector original. Escriba los resultados:

Información																							
Máscara																							
Resultado																							

Escriba a continuación una repetición del experimento con los anteriores parámetros, pero cambiando los siguientes valores: *longitudMediaRafaga=3.* Escriba los resultados:

Información																							
Máscara																							
Resultado																							

A continuación repita de nuevo el experimento anterior cambiando en esta ocasión los siguientes parámetros: *longitudMediaRafaga=3, variacionLongitudRafaga=2.* Escriba los resultados:

Información																							
Máscara																							
Resultado																							

Comente las diferencias observadas al variar los parámetros de generación de máscaras de ruido:

Explique cuáles son las diferencias entre errores de un solo bit y errores de ráfaga:

3 Codificación/Decodificación *Huffman*

En este apartado se implementará un sistema de codificación/decodificación mediante el método *Huffman* para lo cual se proporciona la función `decodificaHuffman` que cumple con la siguiente especificación:

```
function [simbolos] = decodificaHuffman(bitsHuffman, tablaCodigos)
```

- **Recibe:**
  - bitsHuffman == Una secuencia de bits (como un vector fila) codificado con *Huffman*.
  - tablaCodigos == Una tabla de códigos *Huffman* con la lista de símbolos, probabilidades y códigos *Huffman* asociados (tal y como se definieron en la práctica 1).
- **Devuelve:**
  - simbolos == Vector con los valores decodificados (cada posición corresponde al valor numérico de un símbolo).

El algoritmo de decodificación implementado en la función lee bit a bit de la variable de entrada `bitsHuffman` y los almacena en un buffer. Cada vez que se almacena un bit en el buffer se comprueba si la secuencia de bits completa contenida en el buffer se corresponde con alguno de los códigos *Huffman* especificados en la tabla recibida. En caso afirmativo se sustituye en la salida por su símbolo correspondiente. Si se diese una secuencia de bits para la que no es posible encontrar una correspondencia en la tabla de códigos *Huffman* (una secuencia con errores) se salta dicha secuencia y se continúa la decodificación.

3.1 Ejercicio 2

Complete el *script* que se proporciona en el fichero *Ejercicio2.m* para la implementación y prueba de sistema de codificación/decodificación *Huffman* mediante la codificación y posterior decodificación de la cadena de texto *"Esta es una prueba de codificación y decodificación Huffman para la práctica 3."*.

En el primer paso deberá completar la secuencia de codificación de la cadena de texto mediante el cálculo de probabilidades, generación de tabla de códigos *Huffman* y codificación.

En el segundo paso se llevará a cabo la decodificación de la cadena codificada originalmente almacenando los resultados en la variable indicada.

En el tercer paso se llevará a cabo una simulación de transmisión a través de un canal con ruido. Para ello genere una máscara de ruido con los siguientes parámetros: *probabilidadRafaga=0.003, longitudMediaRafaga=1, variacionLongitudRafaga=0, semillaAleatoria=12341*. Aplique la máscara de ruido a la información codificada y lleve a cabo su decodificación. Observe atentamente el resultado de la decodificación. Explique los efectos del ruido de canal observados en el proceso de decodificación. Comente los resultados:

A continuación complete la última parte del ejercicio llevando a cabo una prueba con mayor probabilidad de error. Genere una máscara de ruido con los siguientes parámetros: *probabilidadRafaga=0.03, longitudMediaRafaga=1, variacionLongitudRafaga=0, semillaAleatoria=12341.* Aplique la máscara de ruido a la información codificada y lleve a cabo su decodificación. Observe el resultado de la decodificación. Explique los efectos del ruido de canal observados en el proceso de decodificación. Comente los resultados:

Se observará que en el segundo caso la longitud de la cadena decodificada varía con respecto al original. Dado el algoritmo de decodificación aplicado explique a que cree que se debe esta reducción del tamaño:

Lleve a cabo pruebas con distintos valores de probabilidad de error y semillas. Observe los resultados y comente la relación entre la probabilidad de error y los efectos que provoca en la decodificación:

4 Codificación/Decodificación PCM

En este apartado se implementara la secuencia completa de codificación y decodificación mediante PCM realizando además simulaciones de canal ruidoso. Para ello se utilizará la función `getSecuenciaBits` (práctica 2) utilizada para convertir una secuencia de valores a una secuencia de bits. Adicionalmente se proporciona la función `getSecuenciaNumerica` para llevar a cabo el proceso inverso:

```
function [secuencia] = getSecuenciaNumerica(secuenciaBits,nbits)
```

- **Recibe:**
  - secuenciaBits == Una secuencia de bits (como un vector fila) en este caso será el resultado de llamar a la funcion getSecuenciaBits sobre el resultado de la cuantificación PCM.
  - nbits == Número de bits por muestra
- **Devuelve:**
  - secuencia == La secuencia de valores decimales correspondientes a la secuencia de bits recibida tomándolos de nbits en nbits.

4.1 Ejercicio 3

Complete el script suministrado *Ejercicio3.m* siguiendo las indicaciones del mismo. En primer lugar se realizará una codificación PCM de la muestra de audio suministrada con 8 bits y valor de sobrecarga V=4. Se utilizará la función `getSecuenciaBits` para obtener la secuencia de bits correspondiente. A continuación (paso 3 en el fichero de código) se llevará a cabo la decodificación de la secuencia de bits mediante el proceso inverso (utilizando `getSecuenciaNumerica` y `decodificaPCM).`

En el cuarto paso se llevará a cabo una primera simulación de canal ruidoso. Genere una máscara de error para la secuencia de bits codificada. Utilice los siguientes parámetros: *probabilidadRafaga=0.01*, *longitudMediaRafaga=1, variacionLongitudRafaga=0, semillaAleatoria=2*. Comente los resultados obtenidos (resulta altamente ilustrativa la audición de la muestra) y el error de reconstrucción:

¿Cómo varía el error de reconstrucción al modificar la probabilidad de ruido? Realice las pruebas que considere oportunas modificando los valores de generación de ruido. Comente los resultados:

Para finalizar (paso 5) lleve a cabo una simulación de ruido a ráfagas con los siguientes parámetros: *probabilidadRafaga=0.0005, longitudMediaRafaga=200, variacionLongitudRafaga=100, semillaAleatoria=2.* Comente los resultados obtenidos:

Compare y relacione los valores de probabilidad de ráfaga y de longitud de las mismas con el efecto producido sobre el error de reconstrucción. Realice las pruebas necesarias y comente los resultados:

Comente las diferencias que aprecia al escuchar las muestras de audio original y reconstruída para un canal ideal, para un canal ruidoso (con y sin ráfagas).

Repita los experimentos para codificación PCM uniforme con 6 y 4 bits. Rellene la siguiente tabla:

Codificación PCM	Potencia de Error (sin canal ruidoso)	Potencia de Error (canal ruidoso sin ráfagas)	Potencia de Error (canal ruidoso con ráfagas)
8 bits			
6 bits			
4 bits			

5 Codificación/Decodificación PCM + Huffman

En este apartado se realizará el proceso de codificación PCM conjuntamente con una codificación *Huffman*. El experimento consistirá en codificar una señal mediante PCM. Posteriormente se codificará mediante *Huffman* el vector de muestras cuantificadas, para lo cual se deberán calcular sus probabilidades y códigos *Huffman* y proceder a la codificación tal y como se llevaba a cabo en la primera práctica. A continuación se realizará el proceso inverso para reconstruir la señal de audio y adicionalmente se realizará un experimento de transmisión con ruido.

4.2 Ejercicio 4

Complete el script suministrado *Ejercicio4.m* siguiendo los pasos especificados en él. En esta ocasión se utilizará para las pruebas una señal de entrada más corta con el objetivo de acelerar el proceso. Dicha señal se codificará mediante PCM uniforme de 6 bits y valor de sobrecarga V=4 (paso 2). En el paso 3 se utilizará la salida de la cuantificación uniforme (sin transformar a una secuencia de bits) para llevar a cabo una codificación *Huffman.* En el paso 4 se llevará a cabo el proceso de decodificación completa (*Huffman* + PCM) y se mostrará por pantalla la señal reconstruida. Comente la diferencia en número de bits necesarios para codificar la señal con PCM únicamente y al hacerlo con PCM+*Huffman*.

En el último paso se generará una máscara de error con los siguientes parámetros: *probabilidadRafaga=0.01, longitudMediaRafaga=1, variacionLongitudRafaga=0* y *semillaAleatoria=1*. Aplique dicha máscara de error a la señal codificada y lleve a cabo la reconstrucción del resultado mostrando por pantalla la señal reconstruida. Comente los resultados obtenidos comparando los resultados de la decodificación tras simular un canal con y sin ruido: