QPSK Modulator/Demodulator Example

This documents describes/implements the QPSK modulation and demodulation of a song signal.

**Contents**

* [Program Initialization](file:///D:\ku\7%20Spring_21\ELEC301\projects\project5\html\partb.html#5)
* [Read and Display an Example Image](file:///D:\ku\7%20Spring_21\ELEC301\projects\project5\html\partb.html#6)
* [Convert Image to a Binary Vector](file:///D:\ku\7%20Spring_21\ELEC301\projects\project5\html\partb.html#9)
* [Generate Modulated Signal](file:///D:\ku\7%20Spring_21\ELEC301\projects\project5\html\partb.html#14)
* [to keep BER values for different SNR values](file:///D:\ku\7%20Spring_21\ELEC301\projects\project5\html\partb.html#21)
* [Channel Effect](file:///D:\ku\7%20Spring_21\ELEC301\projects\project5\html\partb.html#28)
* [The QPSK Receiver Processing](file:///D:\ku\7%20Spring_21\ELEC301\projects\project5\html\partb.html#35)
* [Constellation Estimates](file:///D:\ku\7%20Spring_21\ELEC301\projects\project5\html\partb.html#45)
* [Bit Estimates](file:///D:\ku\7%20Spring_21\ELEC301\projects\project5\html\partb.html#46)
* [disp theta vs ber graph](file:///D:\ku\7%20Spring_21\ELEC301\projects\project5\html\partb.html#50)
* [Reconstruct Image](file:///D:\ku\7%20Spring_21\ELEC301\projects\project5\html\partb.html#51)
* [Reconstruct Image](file:///D:\ku\7%20Spring_21\ELEC301\projects\project5\html\partb.html#53)

Prepared for ELEC 301

by Alper T. Erdogan

\*16.03.2020\*

edited by Yakup Enes Güven 64045

**Program Initialization**

%Clear Variables and Close All Figure Windows

% Clear all previous variables

clear

% Close all previous figure windows

close all

**Read and Display an Example Image**

**cameraman.tif** is an example gray-level image provided my matlab

Load the Cameraman Image

Im = imread('cameraman.tif');

% Extract part of the image

Im=Im(51:100,101:150);

Display the image

imshow(Im);

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

**Convert Image to a Binary Vector**

We need to convert the image to a binary bit sequence

Convert 256x256 image matrix to an image (column) vector (of size 256^2x1) by concatenating columns

Imv=Im(:);

Convert each the number in each row to a binary vector

Imvb=de2bi(Imv);

Note that **Imvb** has size 256^2x8

Now generate a row vector containing all bits

Imvbt=Imvb';

s=Imvbt(:)';

**Generate Modulated Signal**

QPSK Modulated Signal

From the single bit sequence generate a vector sequence

sv=[s(1:2:end);

s(2:2:end)];

QPSK Constellation Mapper [0;0]-> -1-i [0;1]-> -1+i [1;0]-> 1-i [1;1]-> 1+i

for k=1:size(sv,2)

switch num2str(sv(:,k)')

case '0 0'

c(k)=1-i;

case '1 0'

c(k)=-1-i;

case '0 1'

c(k)=1+i;

otherwise

c(k)=-1+i;

end

end

% Normalize the power to 1

c=c/sqrt(2);

Rectangle Modulation

% Sample Rate

Fsampling=2^19;

% Sample Intervale

Tsampling=1/Fsampling;

% Symbol Rate

Fsymbol=2^13;

% Symbol Period

Tsymbol=1/Fsymbol;

% Number of Samples per Symbol Period

Ns=Tsymbol/Tsampling;

Baseband Signal (samples)

xb=kron(c,ones(1,Ns));

Carrier frequency:

$f_c=60kHz$

fc=60e3; % 60 kHz;

%define theta

theta=(0:pi/10:2\*pi);

**to keep BER values for different SNR values**

BER10=(0:length(theta)-1);

BER1=(0:length(theta)-1);

for a=1:2

for i=1:length(theta)

% Carrier signal: \_

$c(t)=cos(2\pi f_c t)$

t=(0:1:(length(xb)-1))\*Tsampling;

cost=cos(2\*pi\*fc\*t);

sint=sin(2\*pi\*fc\*t);

Transmitter output

$x(t)=Re(xb(t))cos(2\pi f_c t)-Im(xb(t))sin(2\pi f_c t)$

x=real(xb).\*cost-imag(xb).\*sint;

**Channel Effect**

We add some noise

First calculate average signal energy (per sample)

sigpow=mean(x.^2);

Define SNR level in (dB)

if a==1

SNR=10;

else a==2

SNR=1;

end

Noise Level

NoiseAmp=sqrt(10^(-SNR/10)\*sigpow);

Generate Noise signal as Gaussian Noise

noise=NoiseAmp\*randn(1,length(x));

Noisy received signal

$y(t)=x(t)+n(t)$

y=x+noise;

**The QPSK Receiver Processing**

Coherent QPSK Receiver operation

First extract real component baseband signal

$u_r(t)=2x(t)cos(2\pi f_c t)$

ur=2\*y.\*cos((2\*pi\*fc\*t)+theta(i));

Then low pass filter this signal

$z_r(t)=u_r(t)*h_{LP}(t)$

zr = lowpass(ur,30e3,Fsampling);

Then extract the imaginary component baseband signal

$u_i(t)=2x(t)sin(2 \pi f_c t)$

ui=-2\*y.\*sin((2\*pi\*fc\*t)+theta(i));

Then low pass filter this signal

$z_i(t)=u_i(t)*h_{LP}(t)$

zi = lowpass(ui,30e3,Fsampling);

Basband signal

z=zr+i\*zi;

**Constellation Estimates**

We sample the baseband received signal to get noisy estimates of transmitted constellation point. This is not the best way though. Any other suggestions for improvement?

ce=z(ceil(Ns/2):Ns:length(z));

**Bit Estimates**

We implement QPSK Demapper to extract bits from constellation estimates

Check which quadrant ce lies in

ser=real(ce)>0;

sei=imag(ce)>0;

se(1:2:(2\*length(ser)))=ser;

se(2:2:(2\*length(ser)))=sei;

Calculate Bit Error Rate

if a==1

BER10(i)=sum(se~=s)/length(s)

else a==2

BER1(i)=sum(se~=s)/length(s)

end

end

**disp theta vs ber graph**

if a==1

figure(2)

plot(theta, BER10)

title('impact of the Receiver s Phase Error with SNR = 10')

else a==2

figure(3)

plot(theta, BER1)

title('impact of the Receiver s Phase Error with SNR = 1')

end

xlabel('Receiver oscillator s phase')

ylabel('BER values')

grid

if a==1

**Reconstruct Image**

From the bits we estimated, we reconstruct 8-bit gray level image

Imvbe=reshape(se,8,length(s)/8)';

% Vectorized image estimate in decimals

Imve=bi2de(Imvbe);

% Image estimate in matrix form

Ime=reshape(Imve,50,50);

figure(4)

subplot(1,2,1)

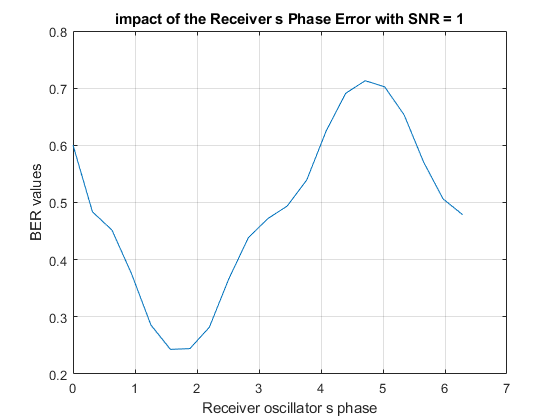
imshow(Im)

title('Transmitted')

subplot(1,2,2)

imshow(uint8(Ime))

title(['Received: BER=10'])



else a==2

**Reconstruct Image**

From the bits we estimated, we reconstruct 8-bit gray level image

Imvbe=reshape(se,8,length(s)/8)';

% Vectorized image estimate in decimals

Imve=bi2de(Imvbe);

% Image estimate in matrix form

Ime=reshape(Imve,50,50);

figure(5)

subplot(1,2,1)

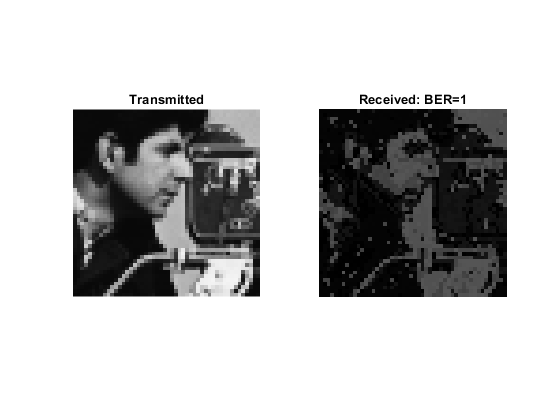
imshow(Im)

title('Transmitted')

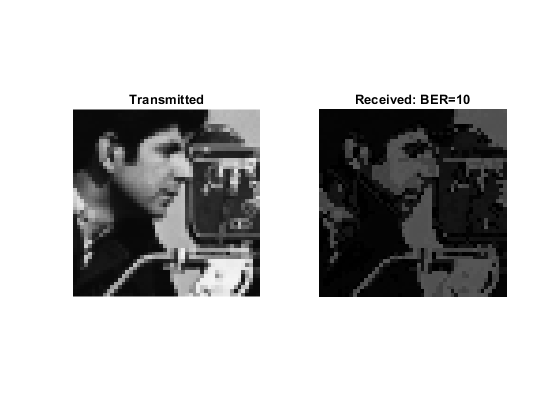
subplot(1,2,2)

imshow(uint8(Ime))

title(['Received: BER=1'])



end



end

[*Published with MATLAB® R2020b*](https://www.mathworks.com/products/matlab/)