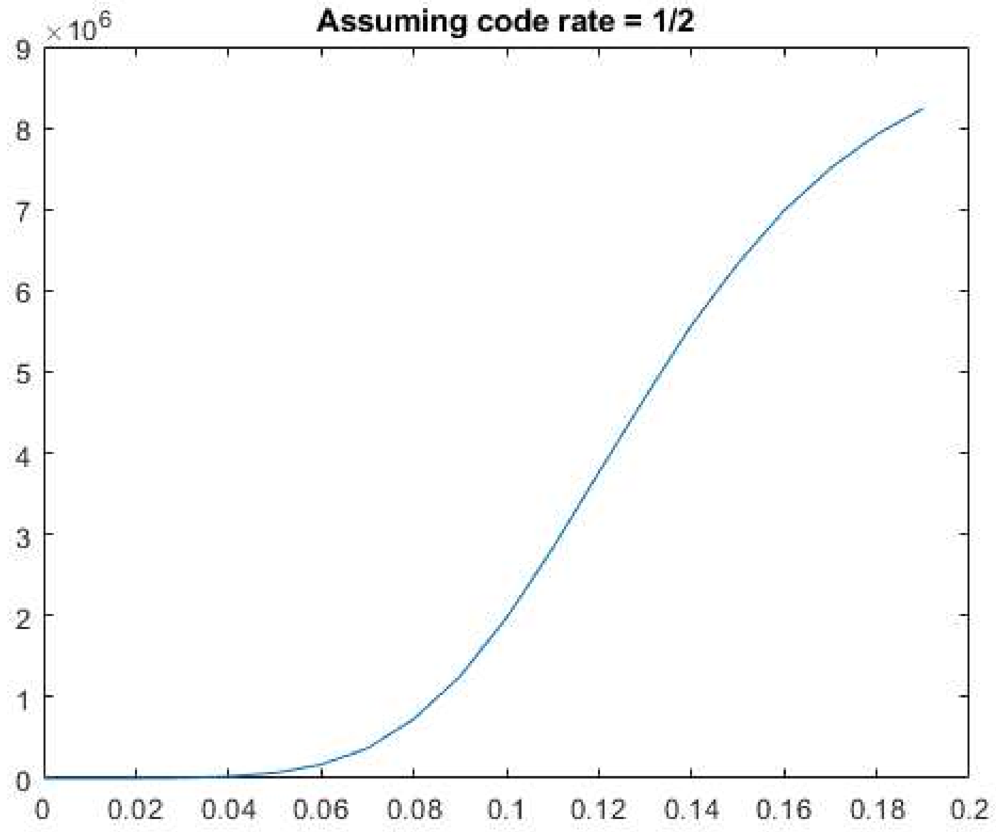
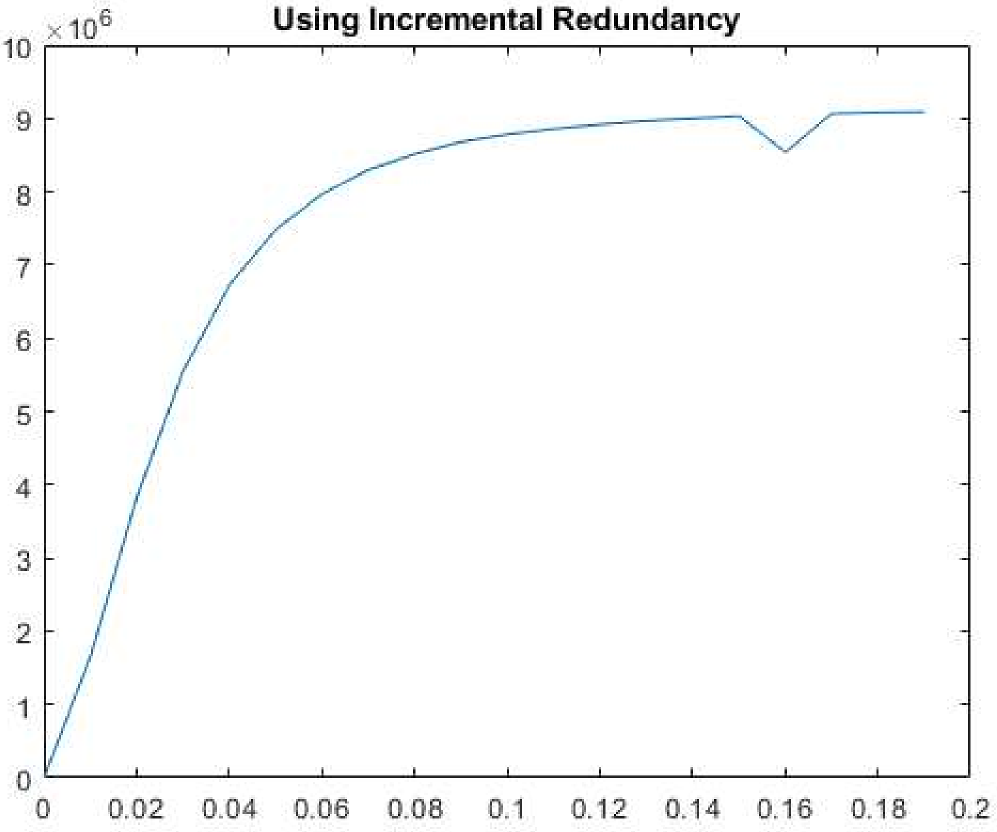
```
clc
close all
clear all

obj = VideoReader ('E:\Downloads\Sample videos\Sample Videos\highway.avi');
a = read(obj);
frames = get(obj,'NumFrames');

for i=1:frames
    I(i).cdata = a(:,:,:,i);
end

s = size(I(i).cdata);

%Getting RGB from frames
for i=1:frames
    %Red Components of the Frame
    R_orginal(:,:,i) = I(i).cdata(:,:,1);
    %Green Components of the Frame
    G_orginal(:,:,i) = I(i).cdata(:,:,2);
    %Blue Components of the Frame
    B_orginal(:,:,i) = I(i).cdata(:,:,3);
end

%Reshaping & double & de2bi & reshape to RGB
Ri = reshape(R_orginal,1,[]);
Gi = reshape(G_orginal,1,[]);
Bi = reshape(B_orginal,1,[]);
Rdouble = double(Ri);
Gdouble = double(Gi);
Bdouble = double(Bi);
Rbin = de2bi(Rdouble);
Gbin = de2bi(Gdouble);
Bbin = de2bi(Bdouble);
R = reshape(Rbin,1,[]);
G = reshape(Gbin,1,[]);
B = reshape(Bbin,1,[]);

%Forming total packets of RGB
TotalPackets = [R G B];

%Dividing Total Packets to Packets with
%each row contains a frame of size 1024
[x,y] = size(TotalPackets);
NumOfPackets = (x*y)/1024;
Packets = reshape(TotalPackets,[NumOfPackets,1024]);

%Code Rates
% 8/9 Code Rate (Puncturing Rule)
X8_9 = [1 1 1 1 0 1 1 1];
Y8_9 = [1 0 0 0 1 0 0 0];
PuncturingRule8_9 = [X8_9;Y8_9];
PuncturingRule8_9 = PuncturingRule8_9(:).';

% 4/5 Code Rate (Puncturing Rule)
```

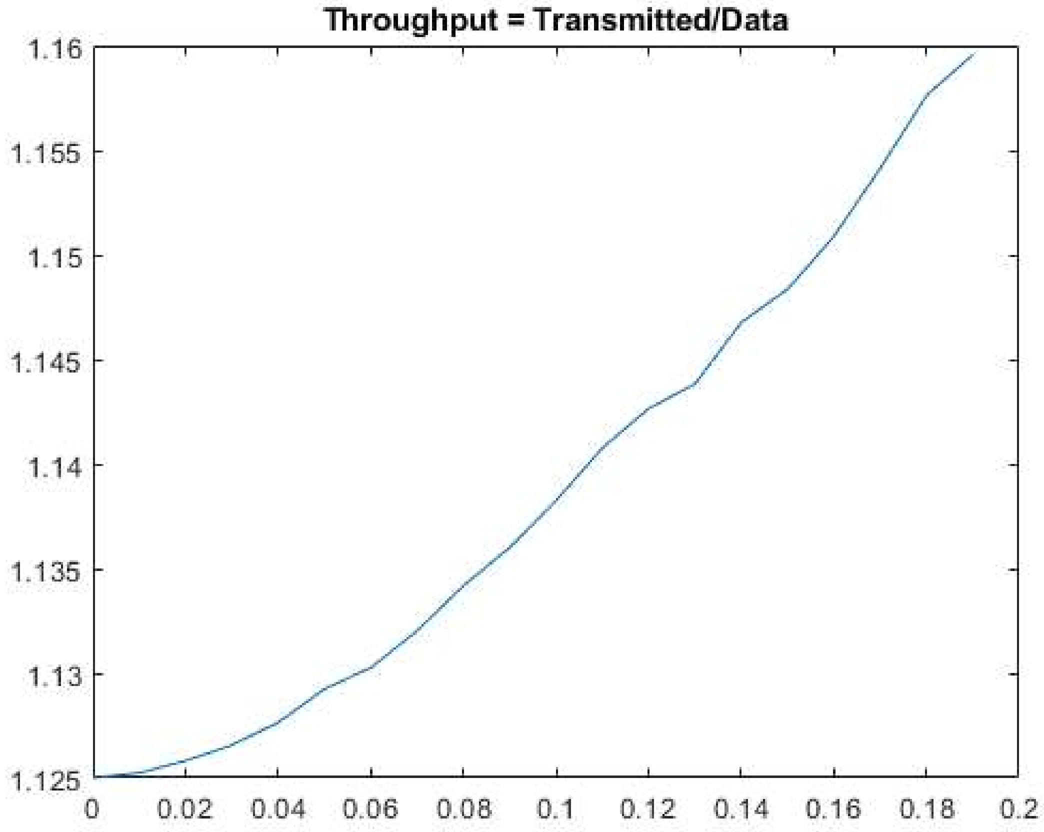
```
X4_5 = [1 1 1 1 1 1 1 1];
Y4_5 = [1 0 0 0 1 0 0 0];
PuncturingRule4_5 = [X4_5;Y4_5];
PuncturingRule4_5 = PuncturingRule4_5(:).';

% 2/3 Code Rate (Puncturing Rule)
X2_3 = [1 1 1 1 1 1 1 1];
Y2_3 = [1 0 1 0 1 0 1 0];
PuncturingRule2_3 = [X2_3;Y2_3];
PuncturingRule2_3 = PuncturingRule2_3(:).';

% 4/7 Code Rate (Puncturing Rule)
X4_7 = [1 1 1 1 1 1 1 1];
Y4_7 = [1 1 1 0 1 1 1 0];
PuncturingRule4_7 = [X4_7;Y4_7];
PuncturingRule4_7 = PuncturingRule4_7(:).';

%Encoding
trellis = poly2trellis(7,[171 133]);
encoded8_9 = Convec(Packets, trellis, PuncturingRule8_9,NumOfPackets);
encoded4_5 = Convec(Packets, trellis, PuncturingRule4_5,NumOfPackets);
encoded2_3 = Convec(Packets, trellis, PuncturingRule2_3,NumOfPackets);
encoded4_7 = Convec(Packets, trellis, PuncturingRule4_7,NumOfPackets);
encoded1_2 = Convec(Packets, trellis, ones(1,16),NumOfPackets);

%Error with p
p = 0.1;
Errored = bsc(Packets,p);
Errored8_9 = bsc(encoded8_9,p);
Errored4_5 = bsc(encoded4_5,p);
Errored2_3 = bsc(encoded2_3,p);
Errored4_7 = bsc(encoded4_7,p);
Errored1_2 = bsc(encoded1_2,p);

%Decoding
decoded8_9 = vdec(Errored8_9,trellis,PuncturingRule8_9,NumOfPackets);
decoded4_5 = vdec(Errored4_5,trellis,PuncturingRule4_5,NumOfPackets);
decoded2_3 = vdec(Errored2_3,trellis,PuncturingRule2_3,NumOfPackets);
decoded4_7 = vdec(Errored4_7,trellis,PuncturingRule4_7,NumOfPackets);
decoded1_2 = vdec(Errored1_2,trellis,ones(1,16),NumOfPackets);

%Comparison
counterTrans = 0;
counterData = 0;

%Loop on the packets and calculate the data
%needed and data transfered to get the throughput
for i=1:max(size(Packets))
    if(decoded8_9(i)~= Packets(i))
        if(decoded4_5(i)~= Packets(i))
            if(decoded2_3(i)~= Packets(i))
                if(decoded4_7(i)~= Packets(i))
                    %use Decoded 1/2
                    counterTrans = counterTrans + 2048;
                    counterData = counterData + 1024;
                    decoded = decoded1_2;
                else
```

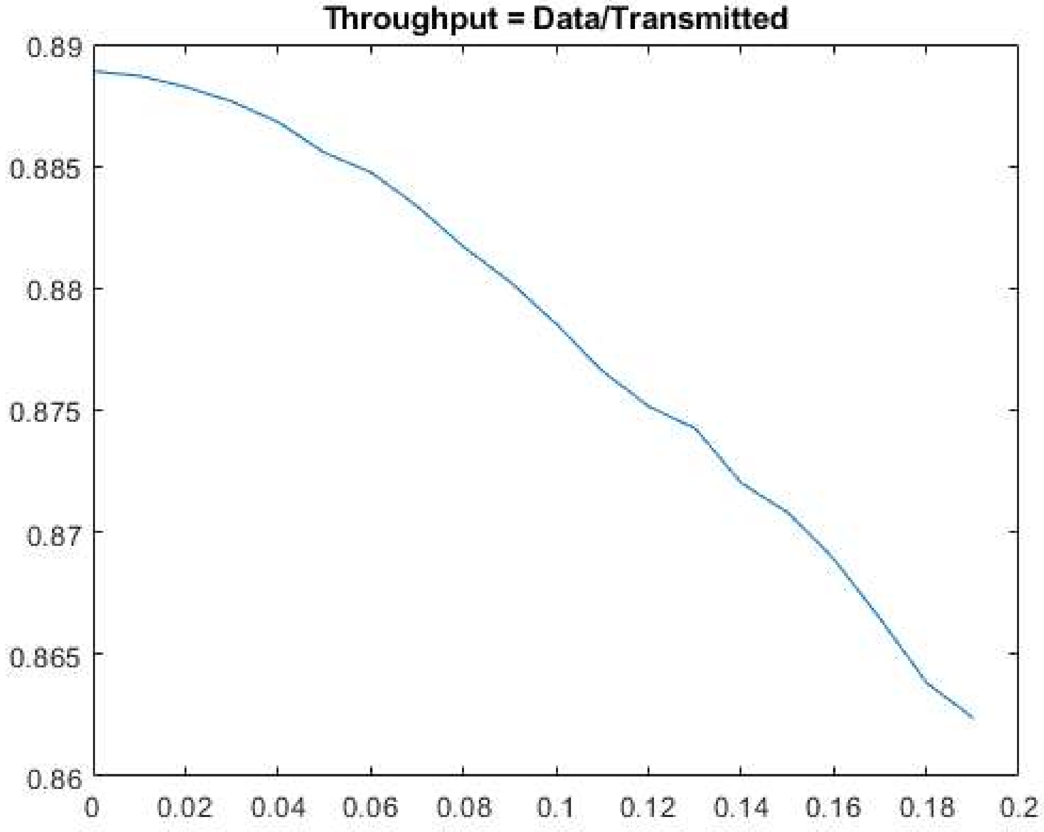
```
                    %use Decoded 4/7
                    decoded = decoded4_7;
                    counterTrans = counterTrans + 1792;
                    counterData = counterData + 1024;
                end
            else
                %use Decoded 2/3
                decoded = decoded2_3;
                counterTrans = counterTrans + 1536;
                counterData = counterData + 1024;
            end
        else
            %use Decoded 4/5
            decoded = decoded4_5;
            counterTrans = counterTrans + 1280;
            counterData = counterData + 1024;
        end
    else
        %use Decoded 8/9
        decoded = decoded8_9;
        counterTrans = counterTrans + 1152;
        counterData = counterData + 1024;
    end
end

p_throughput = counterData/counterTrans;

% Reshape decoded into 1D
Rcvd = reshape(decoded,1,[]);

% Divide each one to RGB
size_Rcvd = max(size(Rcvd));
R_Rcvd = Rcvd(:,1:size_Rcvd/3);
G_Rcvd = Rcvd(:,(size_Rcvd/3)+1:2*size_Rcvd/3);
B_Rcvd = Rcvd(:,(2*size_Rcvd/3)+1:size_Rcvd);

% RGB reshape & bi2de & uint8 & reshape
R_Rcvd_reshaped = restore(R_Rcvd);
G_Rcvd_reshaped = restore(G_Rcvd);
B_Rcvd_reshaped = restore(B_Rcvd);

% Collecting RGB to form a video
mov_create(1:frames) = struct('cdata', zeros(s(1),s(2), 3, 'uint8'), 'colormap',[]);
mov = movFunction(mov_create,R_Rcvd_reshaped,G_Rcvd_reshaped,B_Rcvd_reshaped,frames);

% Creating the video
v = VideoWriter('C:\Users\m8122\Documents\Channel Coding Project NewVideo.avi','Motion JPEG A
VI');
open(v);
writeVideo(v,mov);
close(v);

% implay('C:\Users\m8122\Documents\Channel Coding Project NewVideo.avi');


%Plot of the coded bit error probability vs.
%different values of p from (0.0001 to 0.2) assuming code rate = 1/2.
```

```
prob = 0.0001:0.01:0.2;
i = 1;
for p = 0.0001:0.01:0.2
    Errored1_2 = bsc(encoded1_2,p);
    decoded1_2 = vdec(Errored1_2,trellis,ones(1,16),NumOfPackets);
    z1_2(i) = biterr(Packets,decoded1_2);
    i=i+1;
end
figure
plot(prob,z1_2);
title('Assuming code rate = 1/2');


%Plot of the coded bit error probability vs.
%different values of p from (0.0001 to 0.2) using incremental redundancy

%Plot of the throughput (data rate) vs.
%different values of p from (0.0001 to 0.2) using incremental redundancy.

j=1;
for p = 0.0001:0.01:0.2
    %Error with p
    Errored8_9 = bsc(encoded8_9,p);
    Errored4_5 = bsc(encoded4_5,p);
    Errored2_3 = bsc(encoded2_3,p);
    Errored4_7 = bsc(encoded4_7,p);
    Errored1_2 = bsc(encoded1_2,p);

    %Decoding
    decoded8_9 = vdec(Errored8_9,trellis,PuncturingRule8_9,NumOfPackets);
    decoded4_5 = vdec(Errored4_5,trellis,PuncturingRule4_5,NumOfPackets);
    decoded2_3 = vdec(Errored2_3,trellis,PuncturingRule2_3,NumOfPackets);
    decoded4_7 = vdec(Errored4_7,trellis,PuncturingRule4_7,NumOfPackets);
    decoded1_2 = vdec(Errored1_2,trellis,ones(1,16),NumOfPackets);

    %Comparison
    counterTrans = 0;
    counterData = 0;

    for i=1:max(size(Packets))
        if(decoded8_9(i)~= Packets(i))
            if(decoded4_5(i)~= Packets(i))
                if(decoded2_3(i)~= Packets(i))
                    if(decoded4_7(i)~= Packets(i))
                        %use Decoded 1/2
                        counterTrans = counterTrans + 2048;
                        counterData = counterData + 1024;
                        decoded_test = decoded1_2;
                    else
                        %use Decoded 4/7
                        decoded_test = decoded4_7;
                        counterTrans = counterTrans + 1792;
                        counterData = counterData + 1024;
                    end
                else
                    %use Decoded 2/3
                    decoded_test = decoded2_3;
                    counterTrans = counterTrans + 1536;
                    counterData = counterData + 1024;
                end
            else
                %use Decoded 4/5
                decoded_test = decoded4_5;
                counterTrans = counterTrans + 1280;
                counterData = counterData + 1024;
            end
        else
            %use Decoded 8/9
            decoded_test = decoded8_9;
            counterTrans = counterTrans + 1152;
            counterData = counterData + 1024;
        end
    end

    result(j) = counterData/counterTrans;
    r(j) = counterTrans/counterData;
    z_inc_red(j) = biterr(Packets,decoded_test);
    j=j+1;
end
%Plot BER vs p (incremental redundancy)
figure
plot(prob,z_inc_red);
title('Using Incremental Redundancy');

%Plot throughput vs p (incremental redundancy)
figure
plot(prob,result);
title('Throughput = Data/Transmitted');


figure
plot(prob,r);
title('Throughput = Transmitted/Data');
```


Assuming code rate = 1/2
×10^6
9
8
7
6
5
4
3
2
1
0
0
0.02
0.04
0.06
0.08
0.1
0.12
0.14
0.16
0.18
0.2


Using Incremental Redundancy
×10^6
10
9
8
7
6
5
4
3
2
1
0
0
0.02
0.04
0.06
0.08
0.1
0.12
0.14
0.16
0.18
0.2


Throughput = Transmitted/Data
1.16
1.155
1.15
1.145
1.14
1.135
1.13
1.125
0
0.02
0.04
0.06
0.08
0.1
0.12
0.14
0.16
0.18
0.2


Throughput = Data/Transmitted
0.89
0.885
0.88
0.875
0.87
0.865
0.86
0
0.02
0.04
0.06
0.08
0.1
0.12
0.14
0.16
0.18
0.2

Published with MATLAB® R2019b