Digital Video

HW 4

Ambika Verma

Av28944

Q 1)

Matlab Code –

```
clear all
close all
clc
vid = VideoReader('C:\Users\user1\Downloads\cat.mp4');
outputVideo = VideoWriter('C:\Users\user1\Downloads\q1_diff2');
outputVideo.FrameRate = vid.FrameRate;
open(outputVideo)
vframes=vid.NumberofFrames;
vid = VideoReader('C:\Users\user1\Downloads\cat.mp4');
framecell=[];
 for i=1:1:vframes-1
   frame1 = read(vid,i);
   frame_gray1=double(rgb2gray(frame1));
            frame2 = read(vid,i+1);
   frame_gray2=double(rgb2gray(frame2));

c=frame_gray2;
j=13; k=9;
c1=frame_gray1;
c2=c(j:end-j+1,k:end-k+1);
n1=16*ones(1,66);
n2=16*ones(1,119);
d=mat2cell(c2,n1,n2);
blk=16;
result=[];
z3=[];
for hor=1:1:66
   for ver=1:1:119
% first step
m=j+(blk)*(hor-1); n=k+(blk)*(ver-1);
sad=[];
for x=m-4:4:m+4
   for y=n-4:4:n+4
      e=c1(x:x+blk-1,y:y+blk-1);
      s=sum(sum(abs(d{hor,ver}-e)));
      sad=[sad s];
   end
end
[smin,ind]=min(sad);
switch ind
   case 1
      p=m-4;q=n-4;
   case 2
      p=m-4;q=n;
   case 3
```

```
        p=m-4;q=n+4;
    case 4
        p=m;q=n-4;
    case 5
        p=m;q=n;
    case 6
        p=m;q=n+4;
    case 7
        p=m+4;q=n-4;
    case 8
        p=m+4;q=n;
    case 9
        p=m+4;q=n+4;
end
% second step
sad2=[];
for x2=p-2:2:p+2
    for y2=q-2:2:q+2
        f=c1(x2:x2+blk-1,y2:y2+blk-1);
        s2=sum(sum(abs(d{hor,ver}-f)));
        sad2=[sad2 s2];
    end
end
[smin2,ind2]=min(sad2);
switch ind2
    case 1
        u=p-2;v=q-2;
    case 2
        u=p-2;v=q;
    case 3
        u=p-2;v=q+2;
    case 4
        u=p;v=q-2;
    case 5
        u=p;v=q;
    case 6
        u=p;v=q+2;
    case 7
        u=p+2;v=q-2;
    case 8
        u=p+2;v=q;
    case 9
        u=p+2;v=q+2;
end
%third step
sad3=[];
for x3=u-1:1:u+1
    for y3=v-1:1:v+1
        g=c1(x3:x3+blk-1,y3:y3+blk-1);
        s3=sum(sum(abs(d{hor,ver}-g)));
        sad3=[sad3 s3];
    end
end
```

```
[smin3,ind3]=min(sad3);
switch ind3
   case 1
      a=u-1;b=v-1;
   case 2
      a=u-1;b=v;
   case 3
      a=u-1;b=v+1;
   case 4
      a=u;b=v-1;
   case 5
      a=u;b=v;
   case 6
      a=u;b=v+1;
   case 7
      a=u+1;b=v-1;
   case 8
      a=u+1;b=v;
   case 9
      a=u+1;b=v+1;
end
res=c1(a:a+blk-1,b:b+blk-1);
result=[result res];
end
end
for pos=1:1904:size(result,2)-1904+1
    z2=result(:,pos:1904+pos-1);
   z3=cat(1,z3,z2);
end
diff=abs(c2-z3);
z4=padarray(diff,[12 8]);
z4 = (z4 - min(z4(:))) / (max(z4(:)) - min(z4(:)));
z4 = im2uint8(z4);
writeVideo(outputVideo,z4)
end
close(outputVideo)
```

** video is available separately in folder Q1

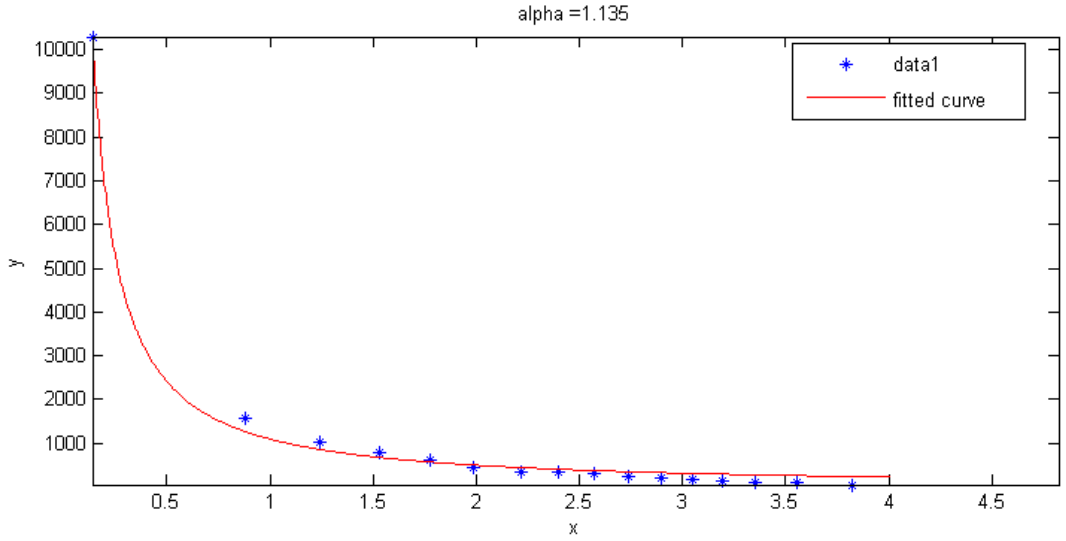
Greatest prediction errors are around moving objects, especially high for fast moving objects (such as the cat's paw).
The background is rather stationary resulting in no or negligible prediction error.

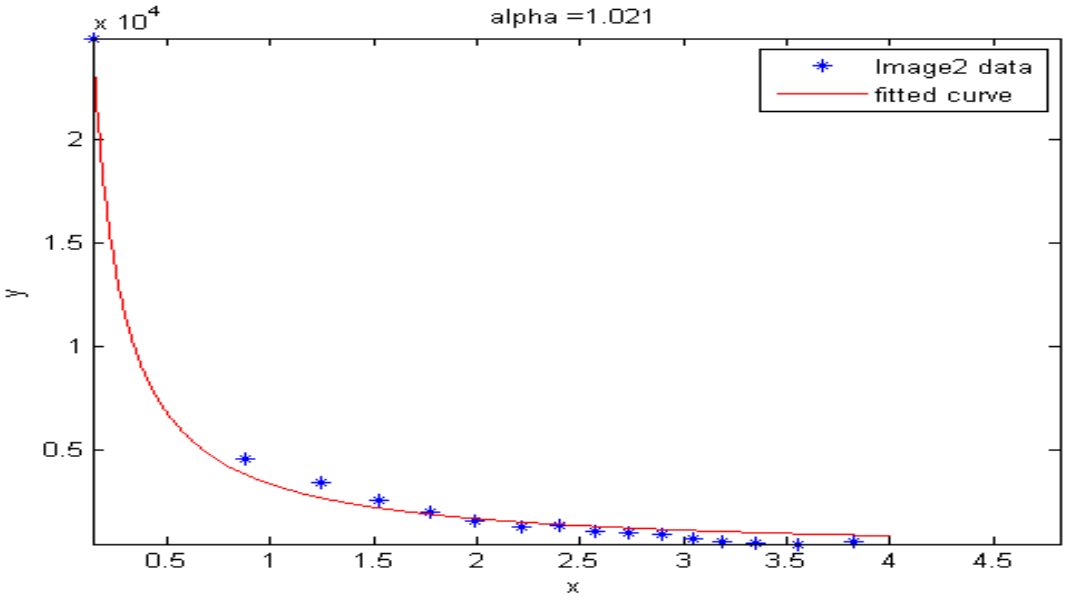
Q 2)

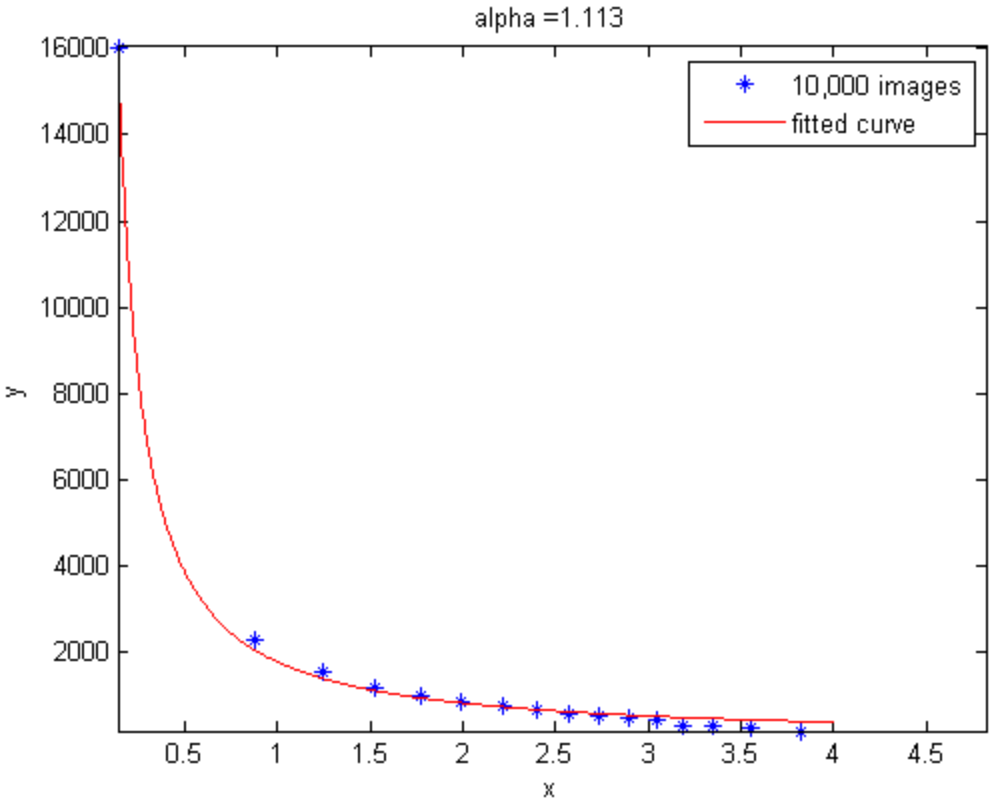
```
clear all
close all
clc

file = fopen('natural-image-patches.dat');
A = fread(file);
images = reshape(A,64,64,10000);

for u=1:1:32
```

```
   for v=1:1:32
      w(u,v)=sqrt(u^2+v^2);
   end
end
w3=unique(w);
w3=w3(1:end-2);
final=[];
for x=1:1:10000     %1 for first image, 2 for second image
ii=images(:,:,x);
iif=abs(fft2(ii));
iif=iif(1:32,1:32);

f1=[];f2=[];f3=[];f4=[];f5=[];f6=[];f7=[];f8=[];
f9=[];f10=[];f11=[];f12=[];f13=[];f14=[];f15=[];f16=[];

for i=1:1:32
   for j=1:1:32
   if w3(1)<=sqrt(i^2+j^2) && sqrt(i^2+j^2)<=w3(27)
      f1=[f1 iif(i,j)];
   elseif w3(28)<=sqrt(i^2+j^2) && sqrt(i^2+j^2)<=w3(54)
      f2=[f2 iif(i,j)];
   elseif w3(55)<=sqrt(i^2+j^2) && sqrt(i^2+j^2)<=w3(81)
      f3=[f3 iif(i,j)];
   elseif w3(82)<=sqrt(i^2+j^2) && sqrt(i^2+j^2)<=w3(108)
      f4=[f4 iif(i,j)];
   elseif w3(109)<=sqrt(i^2+j^2) && sqrt(i^2+j^2)<=w3(135)
      f5=[f5 iif(i,j)];
   elseif w3(136)<=sqrt(i^2+j^2) && sqrt(i^2+j^2)<=w3(162)
      f6=[f6 iif(i,j)];
   elseif w3(163)<=sqrt(i^2+j^2) && sqrt(i^2+j^2)<=w3(189)
      f7=[f7 iif(i,j)];
   elseif w3(190)<=sqrt(i^2+j^2) && sqrt(i^2+j^2)<=w3(216)
      f8=[f8 iif(i,j)];
   elseif w3(217)<=sqrt(i^2+j^2) && sqrt(i^2+j^2)<=w3(243)
      f9=[f9 iif(i,j)];
   elseif w3(244)<=sqrt(i^2+j^2) && sqrt(i^2+j^2)<=w3(270)
      f10=[f10 iif(i,j)];
   elseif w3(271)<=sqrt(i^2+j^2) && sqrt(i^2+j^2)<=w3(297)
      f11=[f11 iif(i,j)];
   elseif w3(298)<=sqrt(i^2+j^2) && sqrt(i^2+j^2)<=w3(324)
      f12=[f12 iif(i,j)];
   elseif w3(325)<=sqrt(i^2+j^2) && sqrt(i^2+j^2)<=w3(351)
      f13=[f13 iif(i,j)];
   elseif w3(352)<=sqrt(i^2+j^2) && sqrt(i^2+j^2)<=w3(378)
      f14=[f14 iif(i,j)];
   elseif w3(379)<=sqrt(i^2+j^2) && sqrt(i^2+j^2)<=w3(405)
      f15=[f15 iif(i,j)];
   elseif w3(406)<=sqrt(i^2+j^2) && sqrt(i^2+j^2)<=w3(432)
      f16=[f16 iif(i,j)];
   end
   end
end
```

```
f=[mean(f1) mean(f2) mean(f3) mean(f4) mean(f5) mean(f6) mean(f7) mean(f8) mean(f9) mean(f10) mean(f11)
mean(f12) mean(f13) mean(f14) mean(f15) mean(f16)];
final=cat(1,final,f);
end

w4=w3(1:27:406);
final2=mean(final,1);
plot((2*pi*w4)/64,final2,'*')
fn = fit(((2*pi*w4)/64),final2','b/x^m')
hold on
plot(fn)
axis([min((2*pi*w4)/64) max((2*pi*w4)/64)+1 min(final2) max(final2)+1])
title(['alpha =1.021'])
```







Alpha values obtained are –
Image 1 – α=1.135
Image 2 – α=1.021
All images – α=1.113

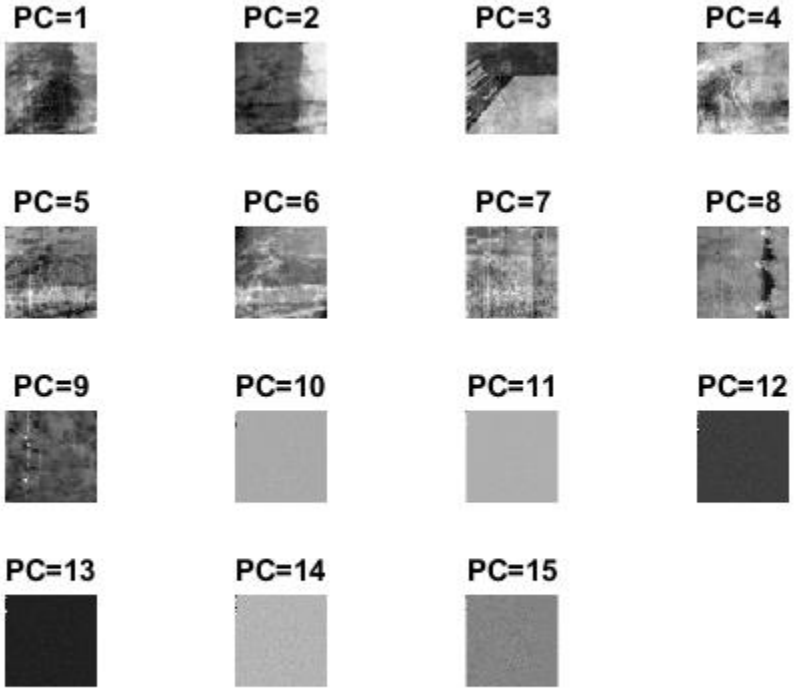
The alpha values obtained are fairly close to 1 (as assumed by the Reciprocal Law).
The individual images as well as the set of 10000 image patches satisfy the statistics expected from natural scene images very closely and the spectrums are observed to be consistent, with the alpha value being in a range of {0.8,1.5}.

Q 3)

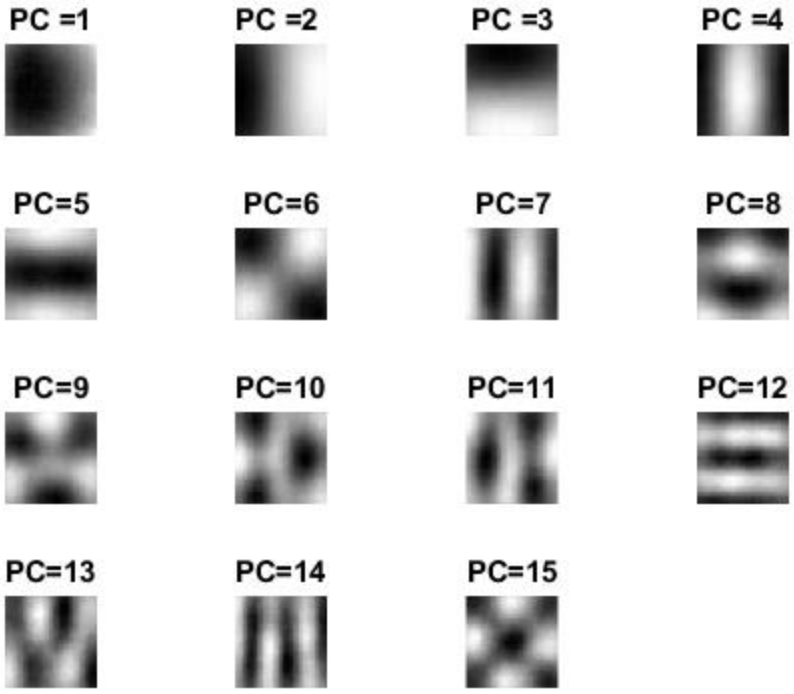
```
clear all
close all
clc
file = fopen('natural-image-patches.dat');
A = fread(file);
images = reshape(A,64,64,10000);
f=[];
 for i=1:1:10
 ii=images(:,:,i);
 ii2=ii(:)';
f=cat(1,f,ii2);
 end
c=cov(f);
[p,b,pt]=svd(c);
figure
for j=1:1:15
```

```
p2=reshape(p(:,j),64,64);
subplot (4,4,j)
   imshow(p2,[])
end
```

Principal components obtained with 10 image patches –



Principal components obtained with 10000 image patches –



Principal components obtained with 10 image patches are highly inconsistent and do not match with the components from Hancock et al presented in the notes. This is expected due to the limited amount of data used (10 patches) for computing the principal components.

With using 10000 image patches the principal components obtained are much better (with distinct directions and orientations) and consistent with the components presented in class.

Q 4)

```
clear all
close all
clc
v = VideoReader('C:\Users\user1\Downloads\q1_diff2.avi');
outputVideo = VideoWriter('C:\Users\user1\Downloads\q1_dct2');
outputVideo.FrameRate = v.FrameRate;
v.NumberofFrames
v = VideoReader('C:\Users\user1\Downloads\q1_diff2.avi');
i=0;
open(outputVideo)
cr1=[];
cr2=[];
%%
while hasFrame(v)
   frame = readFrame(v);
   frame_gray=double(rgb2gray(frame));
dctn2 = @(block_struct) dct2(block_struct.data);
C = blockproc(frame_gray,[8 8],dctn2,'PadPartialBlocks',true);
%%
```

```
mquant=1;
q=mquant*[16 11 10 16 24 40 51 61;
12 12 14 19 26 58 60 55;
14 13 16 24 40 57 69 56;
14 17 22 29 51 87 80 62;
18 22 37 56 68 109 103 77;
24 35 55 64 81 104 113 92;
49 64 78 87 103 121 120 101;
72 92 95 98 112 100 103 99];
B2 = blockproc(C,[8 8],@(block_struct) block_struct.data./q);
B4=round(B2);
c=nnz(B4);
c2=numel(B4);
r2=c/c2;
cr1=[cr1 r2];
B5= blockproc(B4,[8 8],@(block_struct) q.* block_struct.data);
c=nnz(B5);
c2=numel(B5);
r=c2/c;
cr2=[cr2 r];
%%
invdct2 = @(block_struct) idct2(block_struct.data);
I2 = blockproc(B5,[8 8],invdct2);
writeVideo(outputVideo,uint8(I2))
i=i+1
end
close(outputVideo)
```

** videos are available separately in the folder Q4

Average Compression ratio for cat video –
No of non-zero elements / Total No of elements = 0.0727
Or No of total elements / No of non-zero elements =13.787

Average Compression ratio for residual error video from Q1 –
No of non-zero elements / Total No of elements = 0.0355
Or No of total elements / No of non-zero elements =29.9867

We obtain higher compression for residual error video from question 1, as it only contains motion information and is mostly zero over stationary areas (leading to reduced redundancy).