Department of Signals & Systems
Chalmers University

Laboratory Exercise 3

Video Compression Using Block-Matching Motion Compensation

May 3, 2009

Submitted By

Group 43

Rayyan Ali Qureshi

Department of Signals & Systems

Chalmers University

rayyan@student.chalmers.se

Sohaib Maalik

Department of Signals & Systems

Chalmers University

sohaib@student.chalmers.se

Submitted To

Irene Y. H. Gu

Department of Signals & Systems

Chalmers University

irenegu@student.chalmers.se

Table of Contents

List of Figures

1. Introduction

This lab focuses on how lossy video compression can be implemented. It utilizes a simplest and a common technique referred as Block-Matching which incorporates motion vector estimation and compensation for inter and intra mode. [1].

2. Task 2 – Computing the Motion Pixels and Motion Blocks

2.1. Threshold Value

The finally selected threshold value is.

$$Threshold\ Value = \frac{50}{255}$$

2.2. Different Plots

I_{old}

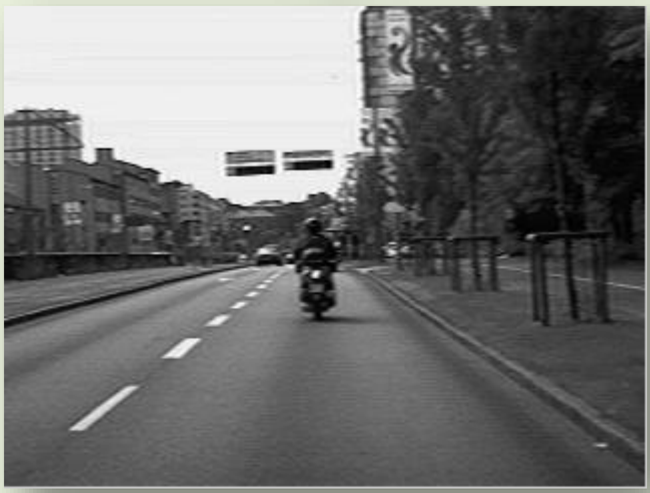


Figure 1: Original Old Image

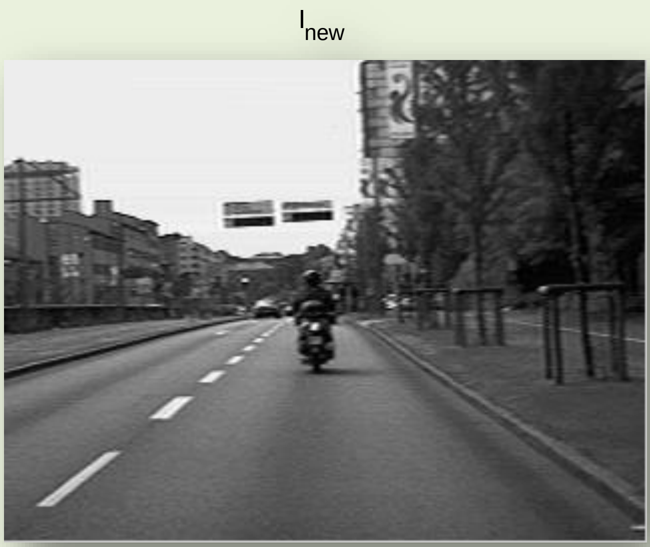


Figure 2: Original New Image

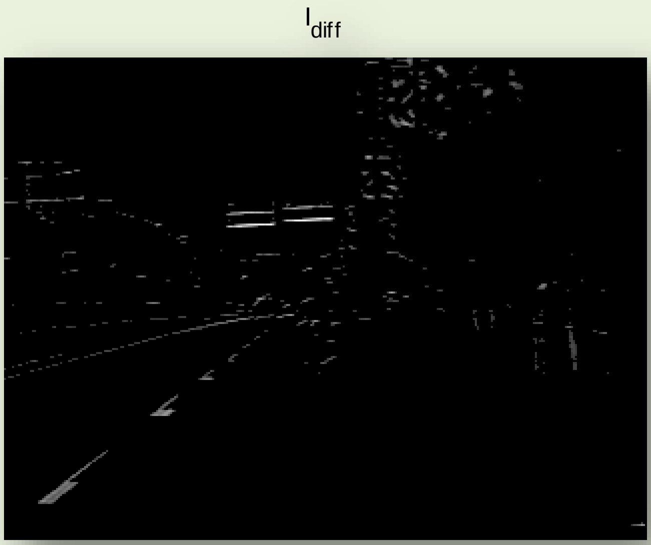


Figure 3: Difference Image

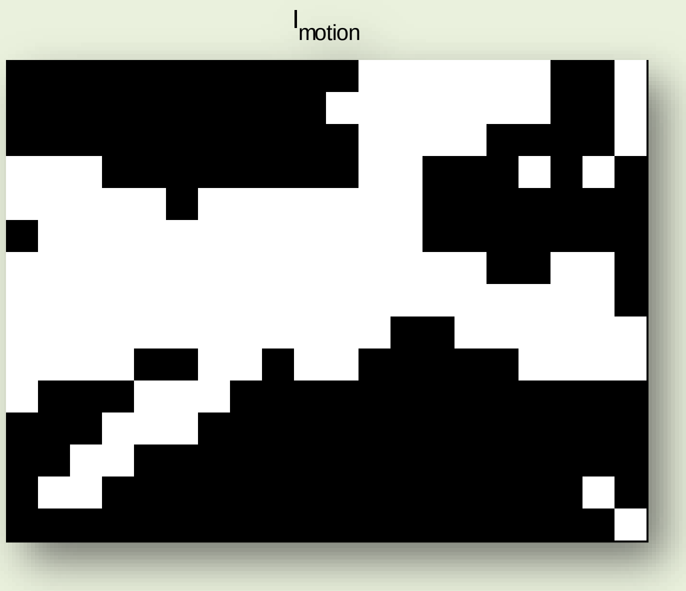


Figure 4: Motion Image

The motion blocks are represented by white colour. The threshold value of 50/255 seems to be appropriate as it covers most motion area in the original image.

3. Task 3 – Computing Motion Vectors

3.1.Image Information

- Image Size (sizey, sizex) = (241, 321)
- Block Size = 16 x 16
- Number of Blocks in Y-direction = 15
- Number of Blocks in X-direction = 20

3.2. Histogram Plot

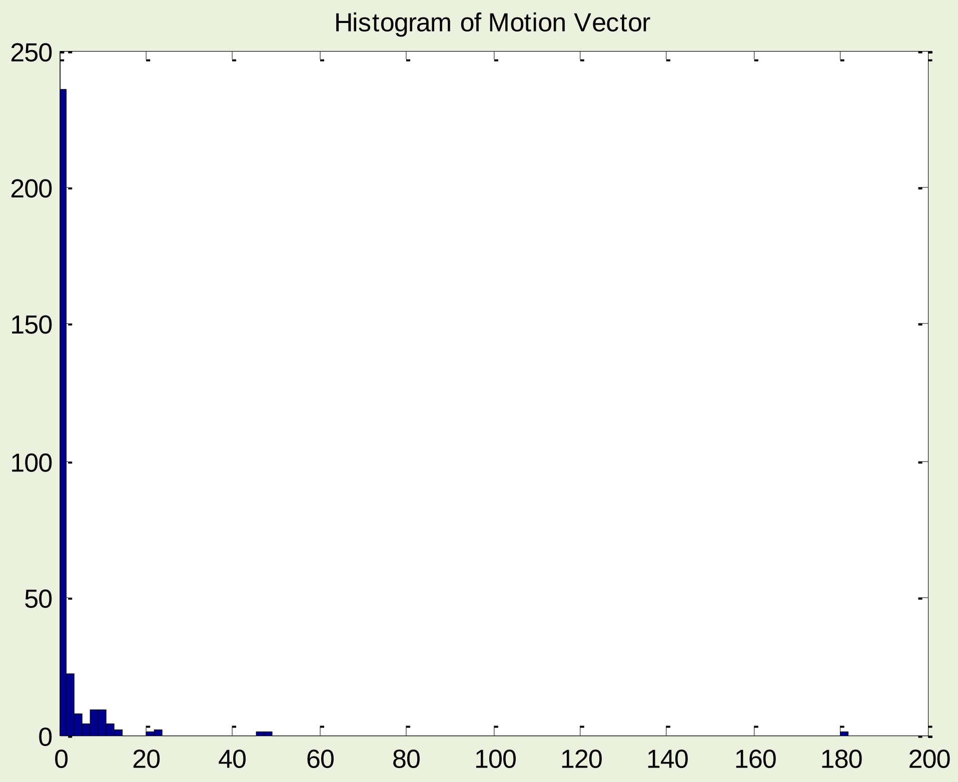


Figure 5: Histogram of Motion Vector Magnitudes

Mean of Displacements = 2.4289

Variance of Displacements = 134.5188

Standard Deviation of Displacements = 11.5982

3.3. Searching Strategy: Global vs. Local

It is evident from the histogram of Figure 5 that most of the motion vectors have magnitude in the range of 0 and 17. A very small number of motion vectors can be seen after this range. This is due to the fact that correlation occurs between the consecutive frames which imply that blocks are closer to each other and haven't moved too far away. The video compression will be much faster if the local searching region is confined to 17 pixels around the motion block instead of the whole image. The resulting image will be as good as the image obtained using full search with very minor difference which is almost negligible. However, we may not be able to find the best match block in all cases as search is constrained to a limited region. So, one should be careful in choosing the right size of local searching in order to save the computational cost while maintaining the same image quality.

4. Task 4 – INTER Processing Mode

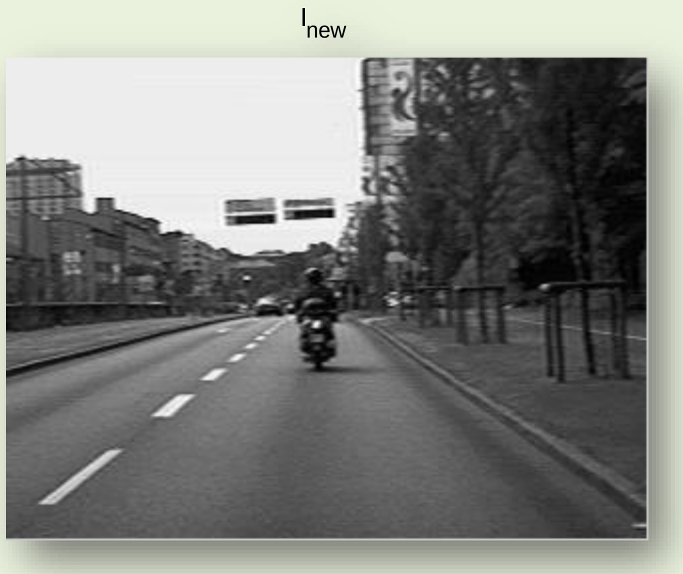


Figure 6: Original Image

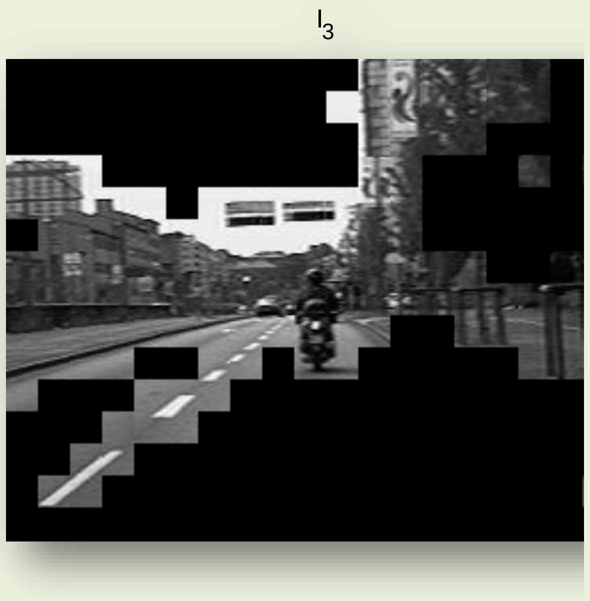


Figure 7: I_3 Motion Parts

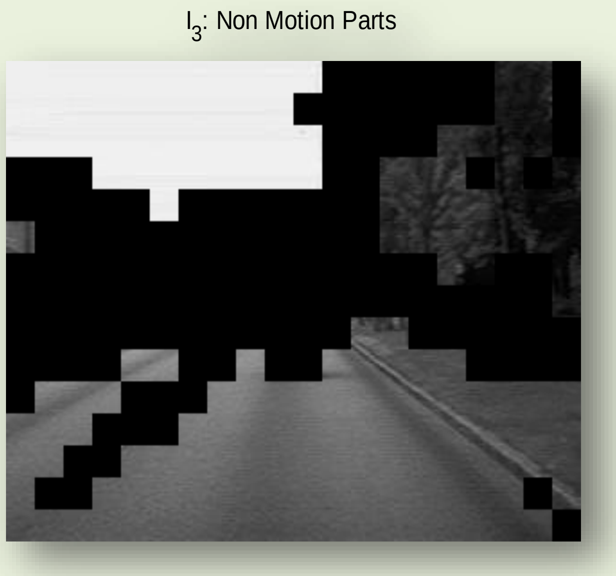


Figure 8: I_3 Non-Motion Parts

5. Task 5 – INTRA Processing Mode

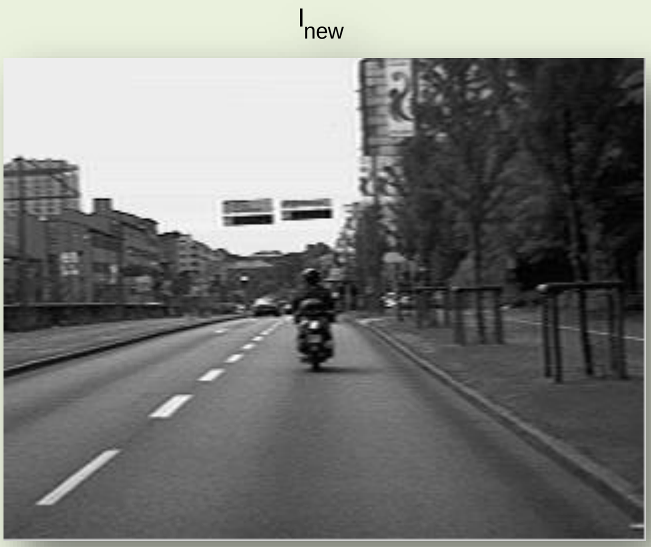
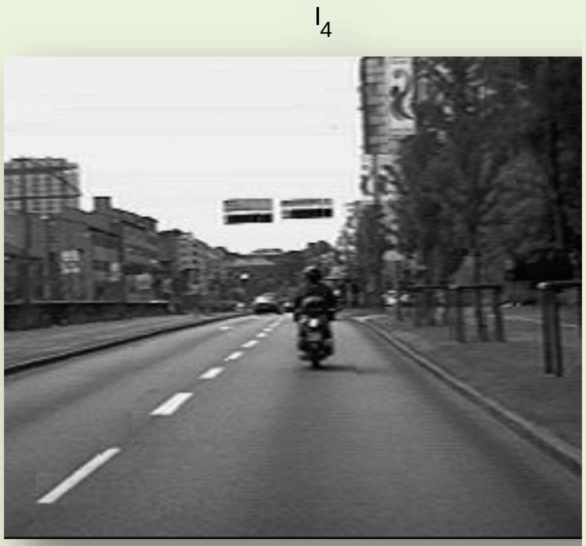


Figure 9: Original Image



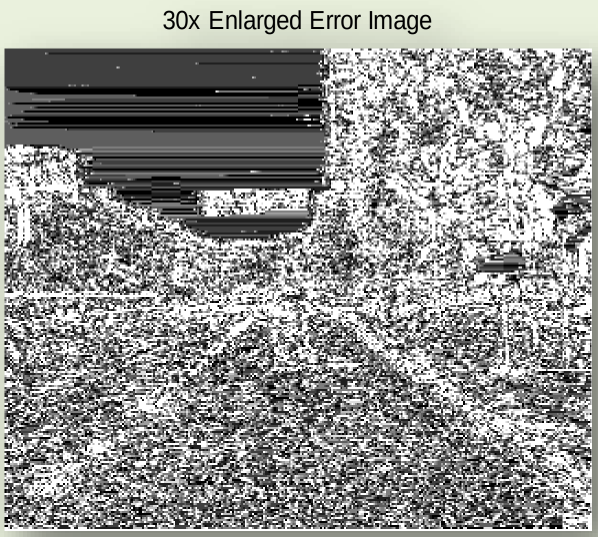


Figure 10: I_4 Reconstructed Image

Figure 11: 30x Enlarged Error Image

6. Peak Signal-to-Noise Ratio (PSNR)

Peak Signal-to-Noise Ratio of the reconstructed image I_4 in Figure 10 for a threshold value of $\frac{50}{255} = 0.196$ was calculated to be;

$$PSNR = 22.0091$$

7. Summary & Comments

The purpose of this lab was to get deep insight on lossy video compression. It utilizes a simple and a common technique referred as Block-Matching for implementation of this kind of compression. In the first task reading avi video files and playing *movie* in Matlab was learnt. It also includes learning on how to display and save image frames, and convert image formats in Matlab. In task 2 implementation of video compression using block-matching motion compensation was done. Two different algorithms namely **"Mean Square Error (MSE)"** and **"Mean Absolute Error (MAE)"** criterion are frequently used for the calculation of movement of motion blocks. In this lab, "Mean Absolute Error (MAE)" criterion, being the simpler and easy one in computations was implemented. Later on a comparison was done regarding the change in searching strategy based on ***global*** and ***local*** search. Global search means looking for a matching block while searching the whole image. On the contrary, local search refers to finding the matching block in a small area around the motion block specified by the magnitude of the motion vectors. As mentioned earlier in task 3, it is evident from the histogram of Figure 5 that most of the motion vectors have magnitude in the range of 0 and 17. A very small number of motion vectors were seen after this range. The video compression would be much faster if the local searching region is confined to 17 pixels around the motion block instead of the whole image. Afterwards in task 5 re-construction of a new frame using non-motion and motion blocks from the old image was done. Later on the quality of the reconstructed image was assessed by calculating the PSNR. Some important observations were made in this lab which is summarized in the points below.

a. Block Matching

For slowly changing frames correlation is comparatively higher and block matching works very well. But if the new frame is changing rapidly then there is less correlation and block matching is not that good.

b. Threshold Value

By increasing the threshold to a smaller extent, the computational time decreases drastically and vice versa. But PSNR remains almost the same. This can be visualized from the table below. The point here is that there is a Trade-off between computational complexity and image quality. By selecting high threshold value we can reduce computational complexity and degradation in image quality will not be distinguishable by HVS (human visual system).

No./Value	Threshold Value	Peak Signal-to-Noise Ratio	Computational Time (sec)
1	40/255=0.1569	22.1052	84.0979
2	50/255=0.1961	22.0091	69.1950
3	60/255=0.2353	21.9129	53.1745
4	100/255=0.3922	21.5804	29.6305
5	150/255=0.5822	20.9466	9.7998

c. Size of Block

Increase in block size result in lower number of motion vectors and eventually lesser number of computations. Compression will be increased but a downside is of lower PSNR. The re-constructed video frame will contain errors. On the contrary if smaller block size is used

PSNR will increase and re-constructed video will contain less errors. But likewise, the downside will be of less compression and higher number of computations.

d. Searching Criterion

As mentioned earlier, global searching is slow and requires more computational power thus more cost. On the other hand local searching is confined to a specific region which can be easily calculated thus taking less time and computation power and eventually less cost.

References

[1] Irene Y. H. Gu, Laboratory Exercise 3: *Video Compression Using Block-Matching Motion Compensation,* Chalmers University, 2009

[2] http://en.wikipedia.org/wiki/Lossy_compression

Appendix – MATLAB Code

```
close all;
clear all;
clc;
%% Task1
video=aviread('Trees1.avi');    %Reading the video file
%movie(video);                  %Playing movie in matlab
FN=length(video);               %Total number of frames in video(110 frames)
CIF=[];                         %Initialization
CI=[];
I=[];
CIF=video(80:85);               %Extracting 5 consecutive image frames of video

for j=1:5
[I map]=frame2im(CIF(j));       %converting 5 video frames from video to images
I=mat2gray(rgb2gray(I));        %converting to gray scale image
CI=[CI I];
end
size(CI);

I1=CI(1:241,1:321);             %size of 1 image is [241 321]
I2=CI(1:241,322:642);           %consecutive image frames
I3=CI(1:241,643:963);
I4=CI(1:241,964:1284);
I5=CI(1:241,1285:1605);

% imwrite(I1,'I1.bmp','bmp');   %saving the images to disk
% imwrite(I2,'I2.bmp','bmp');
% imwrite(I3,'I3.bmp','bmp');
% imwrite(I4,'I4.bmp','bmp');
% imwrite(I5,'I5.bmp','bmp');

%% Task2,3,4,5
%step 2.1
Iold=I2;                        %loading 2 consecutive images
Inew=I3;
%step 2.2
Idiff=Inew-Iold;                %difference
th=50/255                       %setting threshold value
tic
[R C]=size(Idiff);
for l=1:R
    for m=1:C
        if(abs(Idiff(l,m))<=th)
            Idiff(l,m)=0;
        end
    end
end
% step 2.3
blocksize=16;
Imotionblocks = zeros(size(Idiff));%Initializing for motion block
Istaticblocks=zeros(size(Iold));%Initializing static blocks with zeros
Imap=zeros(size(Iold));         %Initializing Reconcstucted image of the same size with zeros
BR=floor(R/blocksize);          %dividing the image into blocks of 16*16
BC=floor(C/blocksize);

for j=1:BR
   for i=1:BC                   %checking for motion and static blocks
      temps=sum(sum(Idiff((j-1)*blocksize+1:j*blocksize, (i-1)*blocksize+1: i*blocksize)));
      if temps==0
          Istaticblocks((j-1)*blocksize+1:j*blocksize, (i-1)*blocksize+1:
i*blocksize)=Iold((j-1)*blocksize+1:j*blocksize, (i-1)*blocksize+1: i*blocksize);
          staticblocks((j-1)*blocksize+1:j*blocksize,(i-1)*blocksize+1:i*blocksize)=Iold((j-
1)*blocksize+1:j*blocksize,(i-1)*blocksize+1:i*blocksize);%just for plotting purpose
      else
         Imotionblocks((j-1)*blocksize+1:j*blocksize, (i-1)*blocksize+1:
i*blocksize)=ones(blocksize,blocksize);
         Inewblock=Inew((j-1)*blocksize+1:j*blocksize,(i-1)*blocksize+1:i*blocksize);
           for yold=1:R-blocksize+1
                for xold=1:C-blocksize+1
                    Ioldblock=Iold(yold:yold+15,xold:xold+15);
                    MAE(yold,xold)=sum(sum(abs(Inewblock-Ioldblock)));
                end
           end
       [x,y]=find(MAE==min(min(MAE)));%finding index value corresponding to minimum MAE
       %Finding best motion vector for block(L,M)
       MV(j,i,1)=(j-1)*blocksize+1-x; %dy
       MV(j,i,2)=(i-1)*blocksize+1-y; %dx
% Step 3.2

 r(j,i)=sqrt(MV(j,i,1)^2+MV(j,i,2)^2); %Magnitude(displacement)of the motion vector
 %figure;hist(r,max(r));title('Histogram of Motion vectore');
 Imap((j-1)*blocksize+1:j*blocksize,(i-1)*blocksize+1:i*blocksize)=Iold(x:x+blocksize-
1,y:y+blocksize-1);    %INTER processing(TASK4)
 Istaticblocks((j-1)*blocksize+1:j*blocksize,(i-
1)*blocksize+1:i*blocksize)=Iold(x:x+blocksize-1,y:y+blocksize-1);   %INTRA processing(TASK5)
      end
   end
end
Ifinal=Istaticblocks                %reconstructed image
Errorimage=abs(Inew-Ifinal);        %computing error image
%Plots
figure; imshow(Iold); title('Origanl Old image');
figure; imshow(Inew); title('Origanl New image');
figure; imshow(Idiff);title('Difference Image');%Images to include in report and threhold
value=50/255=0.1961
figure; imshow(Imotionblocks); title('Imotion Image');
figure;imshow(Imap);title('INTER processed image');
figure;imshow(staticblocks);title('Image with only non motion blocks');
figure;imshow(Ifinal);title('Reconstructed image');
figure;imshow(30*Errorimage);title('30 times Enlarged Error Image');
r1=reshape(r,1,[]);
figure;hist(r1,100);

mean(r1);
var(r1);
std(r1);

%% Task6
MSE=(1/(R*C))*(sum(sum(Errorimage.^2)));    % Mean square error
PSNR=10*log10(1/MSE)                        % PSNR
toc
```