Appendix MATLAB Codes

```
%
% ---- EE392J Final Project ----
%
% All.m: MC Noise Reduction for B&W Films, major file
%
%
% ZHU Xiaoqing
% March, 2002
%
% -----------------------------------------
%

% ---  Choose from different video
%start=378;
%finish=415;

%start=7810;   %index of frame to start
%finish=7840;   %number of frames to be processed
start=18;   %index of frame to start
finish=75;   %number of frames to be processed

%InFileName='america frontier.avi';
InFileName='ggb.avi';
%OutFileName='am_out.avi';
%OutFileName='ggb_out.avi';

% --------   initialize variables    --------

FrameIndex=start:finish;
InSequence=aviread(InFileName,FrameIndex);   %read in
FileInfo=aviinfo(InFileName);
FrameWidth=FileInfo.Width;
FrameHeight=FileInfo.Height;
BetterSequence=MedianSequence(InSequence,5);
OutSequence=BetterSequence;

MeCounter=0;
BlockSize=8;
Threshold=5;

SectionNumber=8;
SectionHeight=FrameHeight/SectionNumber;
SectionWidth=FrameWidth/SectionNumber;

%--------  process frame by frame  -------
CurrentFrameA=ReadSequence(InSequence,1);
CurrentFrameB=ReadSequence(BetterSequence,1);
ErrAvg(1)=mean(mean(abs(CurrentFrameA-CurrentFrameB)))+eps;
NextFrameA=ReadSequence(InSequence,2);
NextFrameB=ReadSequence(BetterSequence,2);
ErrAvg(2)=mean(mean(abs(NextFrameA-NextFrameB)))+eps;

ForwardMVF=zeros(FrameHeight,FrameWidth,2);       %LastFrame => CurrentFrame
```

```
BackwardMVF=zeros(FrameHeight,FrameWidth,2);      %NextFrame => CurrentFrame
for i=2:length(FrameIndex)-1
   FrameIndex(i)                              % indicate frame index in processing

   LastFrameA=CurrentFrameA;
   CurrentFrameA=NextFrameA;
   NextFrameA=ReadSequence(InSequence,i+1);
   LastFrameB=CurrentFrameB;
   CurrentFrameB=NextFrameB;
   NextFrameB=ReadSequence(BetterSequence,i+1);
   ErrAvg(i+1)=mean(mean(abs(NextFrameA-NextFrameB)))+eps;

   BetterFrame=CurrentFrameB;
   ErrFrame=CurrentFrameA-CurrentFrameB;
   MskFrame=abs(ErrFrame)>15;
   EdgFrame=edge(ErrFrame,'sobel');
   MCMarker=0;

   LastFrameMC=CurrentFrameB;
   NextFrameMC=CurrentFrameB;


   % -- Sectionwise processing ----
   for indexI=1:SectionHeight:FrameHeight
      for indexJ=1:SectionWidth:FrameWidth
         RangeI=indexI:indexI+SectionHeight-1;
         RangeJ=indexJ:indexJ+SectionWidth-1;

         LastSectionA=LastFrameA(RangeI,RangeJ);
         CurrentSectionA=CurrentFrameA(RangeI,RangeJ);
         NextSectionA=NextFrameA(RangeI,RangeJ);
         LastSectionB=LastFrameB(RangeI,RangeJ);
         CurrentSectionB=CurrentFrameB(RangeI,RangeJ);
         NextSectionB=NextFrameB(RangeI,RangeJ);

         % -- Motion/Noise Detection --
         MotionMarker=0;
         BlotchMarker=0;
         ErrSection=ErrFrame(RangeI,RangeJ);
         EdgSection=EdgFrame(RangeI,RangeJ);
         MskSection=abs(ErrSection)>15;
         Err=mean(mean(abs(ErrSection)))+eps;        % MAE
         Msk=mean(mean(MskSection));              %percentage of 'real' change, excluding small noise
         Edg=mean(mean(EdgSection));
         if Err/ErrAvg(i)>1.1 & Msk>.02        % detect change
            ErrLA=mean(mean(abs(LastSectionA-CurrentSectionA)))+eps;
            ErrNA=mean(mean(abs(NextSectionA-CurrentSectionA)))+eps;
            ErrLNA=mean(mean(abs(LastSectionA-NextSectionA)))+eps;

            ErrLB=mean(mean(abs(LastSectionB-CurrentSectionB)))+eps;
            ErrNB=mean(mean(abs(NextSectionB-CurrentSectionB)))+eps;
            ErrLNB=mean(mean(abs(LastSectionB-NextSectionB)))+eps;

            if abs(ErrLA*ErrLNB/ErrLNA/ErrLB-1)<.3 & abs(ErrNA*ErrLNB/ErrLNA/ErrNB-1)<.3
               MotionMarker=1;
            else
```

```
          BlotchMarker=1;
        end
      end
      %----------------------------
      if MotionMarker
        if MCMarker==0
          WeightMap=1./(abs(ErrFrame)+1);
          MCMarker=1;
        end


[NewSection,MVF]=WeightedMC(LastFrameA,CurrentFrameA,WeightMap,RangeI,RangeJ,BlockSize);
          ErrNew=mean(mean(abs(NewSection-CurrentSectionB)));
          if ErrNew/ErrLA<.9
            LastFrameMC(RangeI,RangeJ)=NewSection;
          end


[NewSection,MVF]=WeightedMC(NextFrameA,CurrentFrameA,WeightMap,RangeI,RangeJ,BlockSize);
          ErrNew=mean(mean(abs(NewSection-CurrentSectionA)));
          if ErrNew/ErrNA<.9
            NextFrameMC(RangeI,RangeJ)=NewSection;
          end
         MeCounter=MeCounter+1
      end
    end
  end
  % -- Improve by motion-compensated filtering --
  if MCMarker==1
    BetterFrame=(LastFrameMC+NextFrameMC)/2;
   % EdgyMask=edge(LastFrameMC-CurrentFrameA,'sobel') | edge(NextFrameMC-
CurrentFrameA,'sobel') ;
   % BetterFrame(EdgyMask)=CurrentFrameA(EdgyMask);
    BetterFrame=JointDenoise(CurrentFrameB,CurrentFrameA,BetterFrame);      % filtering again to get
the post-processing result
    OutSequence=WriteSequence(OutSequence,BetterFrame,i);
  end
end
%movie(OutSequence,10);
movie2avi(OutSequence,OutFileName);
```

```
%
% ---- EE392J Final Project ----
%
% ReadSequence.m: read in one frame from sequence
%
%
% ZHU Xiaoqing
% March, 2002
%
% -----------------------------------------
%
```

```
function Frame=ReadSequence(InSequence,Index);
% read in one image from Sequence
TempFrame=frame2im(InSequence(Index));
Frame=double(TempFrame(:,:,1));
```

```
%
% ---- EE392J Final Project ----
%
% WriteSequence.m: write out one frame to sequence
%
%
% ZHU Xiaoqing
% March, 2002
%
% ------------------------------------------
%

function OutSequence=WriteSequence(OutSequence,Frame,Index)
% write frame to sequence
TempFrame=frame2im(OutSequence(1)); %ensure data structure is right
for j=1:3
   TempFrame(:,:,j)=Frame;
end
OutSequence(Index)=im2frame(TempFrame);
```

```
%
% ---- EE392J Final Project ----
%
% MedianSequence.m: Temporal median filtering of sequence
%
%
% ZHU Xiaoqing
% March, 2002
%
% ------------------------------------------
%

function OutSequence=MedianSequence(InSequence,WindowDepth)
%
%--------   initialize variables    --------

FrameLength=length(InSequence);
WindowSpan=floor(WindowDepth/2);
OutSequence=InSequence;
for j=1:WindowSpan
   InSequence=[InSequence(1) InSequence InSequence(end)];
end
% ----  pre-shift, initialize window  ----
InSequence=[InSequence(1) InSequence];
for j=1:WindowDepth
   FrameWindow(:,:,j)=ReadSequence(InSequence,j);
```

```
end

[FrameHeight,FrameWidth,WindowDepth]=size(FrameWindow);

% ---- Median filtering ----
for i=1:FrameLength
   FrameWindow(:,:,1:end-1)=FrameWindow(:,:,2:end);
   FrameWindow(:,:,end)=ReadSequence(InSequence,i+WindowDepth);
   BetterFrame=median(FrameWindow,3);
   OutSequence=WriteSequence(OutSequence,BetterFrame,i);
End
```

```
%
% ---- EE392J Final Project ----
%
% WeightedMC: % block-based motion estimation for sections with weighted MAD criterion
%
%
% ZHU Xiaoqing
% March, 2002
%
% ------------------------------------------
%
function [NewSection, MVF] = WeightedMC (LastFrame, CurrentFrame, WeightMap,RangeI, RangeJ,
BlockSize )
%
%


[FrameHeight,FrameWidth]=size(CurrentFrame);
NewSection=LastFrame(RangeI,RangeJ);
[SectionHeight,SectionWidth]=size(NewSection);
MVF=zeros(SectionHeight,SectionWidth,2);
rt=.9;                    %threshold for rejecting spuriousmotion
BlockSpan=min([RangeI(1) RangeJ(1) FrameHeight-RangeI(end) FrameWidth-RangeJ(end)
BlockSize*3/2-1])-BlockSize;
if BlockSpan>=2
   for i=1:SectionHeight
      for j=1:SectionWidth
         BlockI=RangeI(i)-BlockSpan:RangeI(i)+BlockSpan;
         BlockJ=RangeJ(j)-BlockSpan:RangeJ(j)+BlockSpan;
         CurrentBlock=CurrentFrame(BlockI,BlockJ);
         PredBlock=LastFrame(BlockI,BlockJ);
         WeightBlock=WeightMap(BlockI,BlockJ);
         Min=sum(sum(abs(PredBlock-CurrentBlock).*WeightBlock));
         SAD0=Min+eps;
         MVi=0;MVj=0;
         for vi=-BlockSize+1:BlockSize
            for vj=-BlockSize+1:BlockSize
               PredBlock=LastFrame(BlockI-vi,BlockJ-vj);
               SAD=sum(sum(abs(PredBlock-CurrentBlock).*WeightBlock));
               if SAD/SAD0<rt & SAD<Min
                  MVi=vi;
                  MVj=vj;
                  Min=SAD;    %update
               end
            end
         end
         MVF(i,j,1)=MVi;
         MVF(i,j,2)=MVj;
         NewSection(i,j)=LastFrame(RangeI(i)-MVi,RangeJ(j)-MVj);
      end
   end
end
```

```
%
% ---- EE392J Final Project ----
%
% JointDenoise.m: %remove noise by spatio-temopral filtering
%
%
% ZHU Xiaoqing
% March, 2002
%
% -------------------------------------------
function Better=JointDenoise(Current,Last,Next)
%
 [Ni,Nj]=size(Current);
Temp(:,:,1)=Current;
Temp(:,:,2)=Last;
Temp(:,:,3)=Next;
Better=median(Temp,3);
```

```
%
% ---- EE392J Final Project ----
%
% GenSyn.m: generate synthetic scene for evaluation
%
% ZHU Xiaoqing
% March, 2002
%
% -------------------------------------------

Image=imread('lena.bmp');
NoisyImage=Image;
SequenceLength=100;
NoiseMarker=zeros(1,SequenceLength);
for FrameIndex=1:SequenceLength
   if rand>.7
      NoisyImage=imnoise(Image, 'speckle');    %add noise to image;
      NoiseMarker(FrameIndex)=1;      % mark this frame
   else
      NoisyImage=Image;
   end

   for j=1:3
```

```
    TempFrame(:,:,j)=Image;
    end
    StaticSequence(FrameIndex)=im2frame(TempFrame);

    for j=1:3
    TempFrame(:,:,j)=NoisyImage;
    end
    NoisySequence(FrameIndex)=im2frame(TempFrame);
end
% movie(StaticSequence,5);
```

```
%
% ---- EE392J Final Project ----
%
% quality.m: evaluate quality of processed synthetic sequence
%
% ZHU Xiaoqing
% March, 2002
%
% -----------------------------------------
%quality.m
InImage=double(Image);
for i=1:length(OutSequence)
    TempFrame=frame2im(OutSequence(i));
    OutImage=double(TempFrame(:,:,1));
    Err(i)=sum(sum((OutImage-InImage).^2))/FrameHeight/FrameWidth+eps;
    psnr(i)=10*log10(255^2/Err(i));
end
```

```
%
% ---- EE392J Final Project ----
%
% Test.m: test the effect of 5-tap temporal median filtering
%
% ZHU Xiaoqing
% March, 2002
%
% -----------------------------------------
% sandwich structure of frame processing
% try blockwise motion detection + pixelwise MV estimation
%
%--------   initialize variables    --------

GenSyn; % generate synthetic data
FrameIndex=1:SequenceLength;
InSequence=NoisySequence;
OutSequence=InSequence(3:end-2);
InSequence=[InSequence(1),InSequence];

for j=1:5
```

```
    TempFrame=frame2im(InSequence(j));
    FrameWindow(:,:,j)=double(TempFrame(:,:,1));
end

[FrameHeight,FrameWidth,WindowDepth]=size(FrameWindow);

for i=1:length(OutSequence)
    TempFrame=frame2im(InSequence(i+4));
    FrameWindow(:,:,1:4)=FrameWindow(:,:,2:5);
    FrameWindow(:,:,5)=double(TempFrame(:,:,1));
    BetterFrame=median(FrameWindow,3);
    for j=1:3
        TempFrame(:,:,j)=BetterFrame;
    end
    OutSequence(i)=im2frame(TempFrame);
end
InSequence=InSequence(3:end-2);
Quality
```