Mingxuan Wang

University of California, San Diego

December 17, 2014

ASSIGNMENT 4

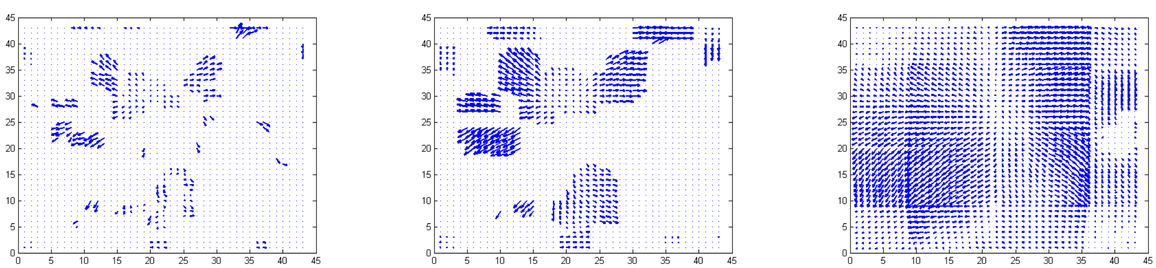
Note that I also implemented in-level iteration to refine the optical flow in following problems.

Problem 1. **Optical Flow [10 points]**

Answers:

(a) **1.1 Dense optical flow**

The output of corridor is figure 1 and 2.Window size is 15, 30 and 100. Tau is 0.07



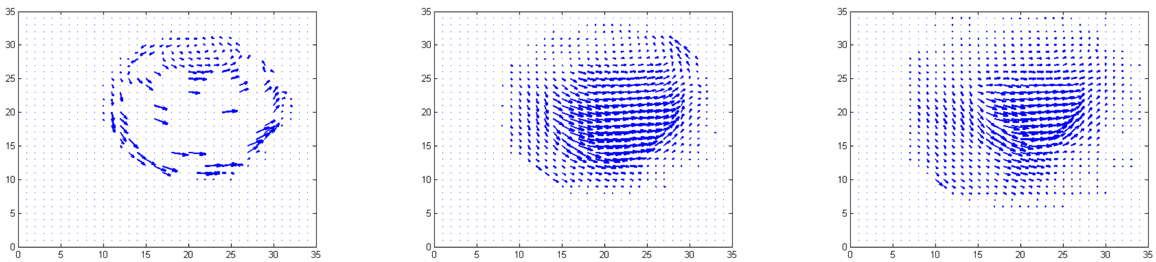
a b c

Figure 1: (a) needle map:15, (b) needle map:30, (c)needle map:100

a b c

Figure 2: (a) hit map:15, (b) hit map:30, (c)hit map:100

The output of sphere is figure 3 and 4.Window size is 10, 30 and 50. Tau is 0.01



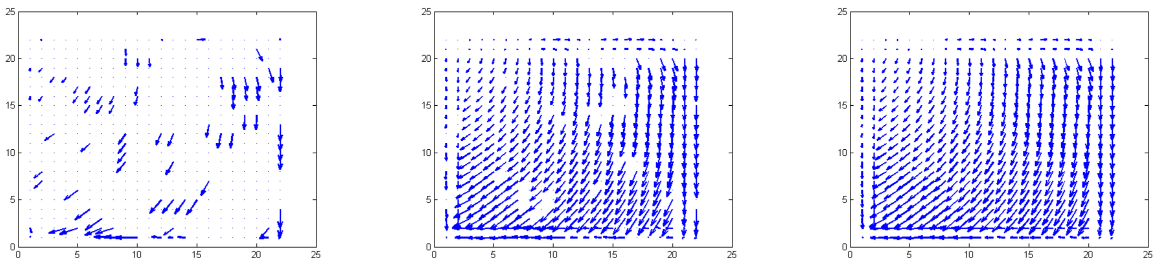
a b c

Figure 3: (a) needle map:10, (b) needle map:30, (c)needle map:50

a b c

Figure 4: (a) hit map:10, (b) hit map:30, (c)hit map:50

The output of sphere is figure 5 and 6.Window size is 10, 15 and 20. Tau is 0.04



a b c

Figure 5: (a) needle map:10, (b) needle map:15, (c)needle map:20

a b c

Figure 6: (a) hit map:10, (b) hit map:15, (c)hit map:20

Comments: For a certain tau,when the window size is small, it is hard to detect enough u and v(optical flow vectors) and correspondingly the hit map has much black(miss) area than white(hit) area. When the window size becomes larger, the hit area becomes bigger and bigger and the optical flow vectors become more and more dense.

(b) **1.2 Corner Detection**

The result of corner detection is below:

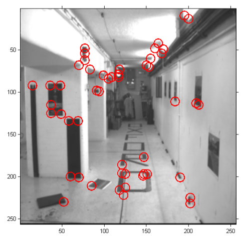


Figure 7: corner detection result: corridor

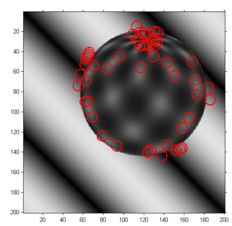


Figure 8: corner detection result: sphere

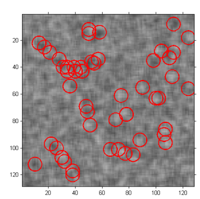


Figure 9: corner detection result: synth

(c) **1.3 Sparse Optical Flow**

The result of sparse optical flow is below:

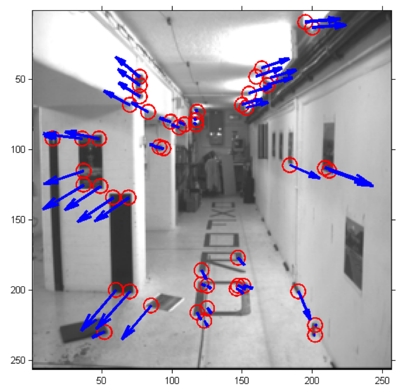


Figure 10: sparse optical flow: corridor

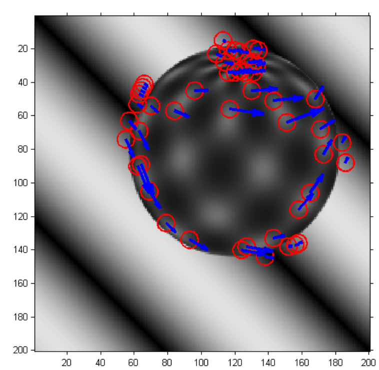


Figure 11: sparse optical flow: sphere

It is possible to mark FOE of corridor but not with other two iamges. FOE is the point where all optical flow vectors come together.So FOE only exits when the camera moves forward.

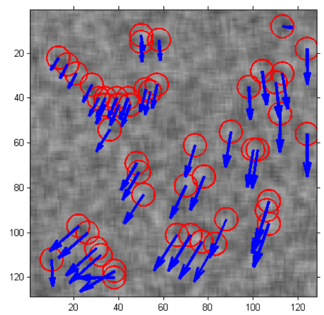


Figure 12: sparse optical flow: synth

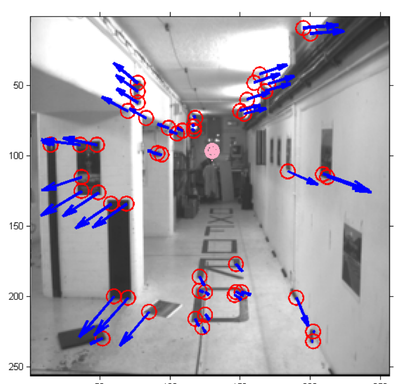
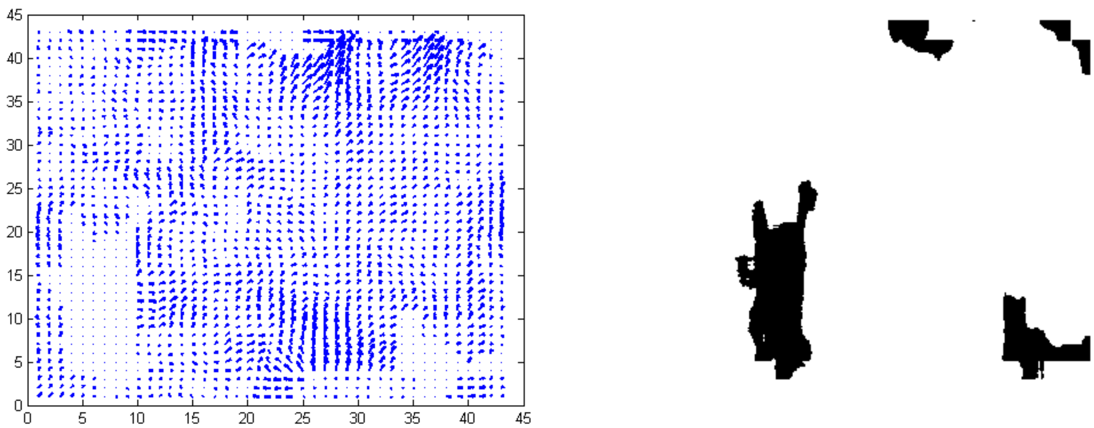


Figure 13: sparse optical flow with FOE: corridor(pink point)

(d) **1.4 My own images**

For my own images, the algorithm works fine. But it still holds the problem that when window size or tau is not good enough. Some pixel that has small eigenvalue is hard to detect optical flow.

The parameters: windowsize: 25, tau:0.09



a b

Figure 14: (a)dense: needle map, (b)dense: hit map

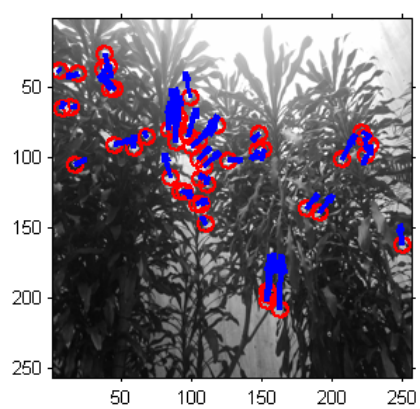
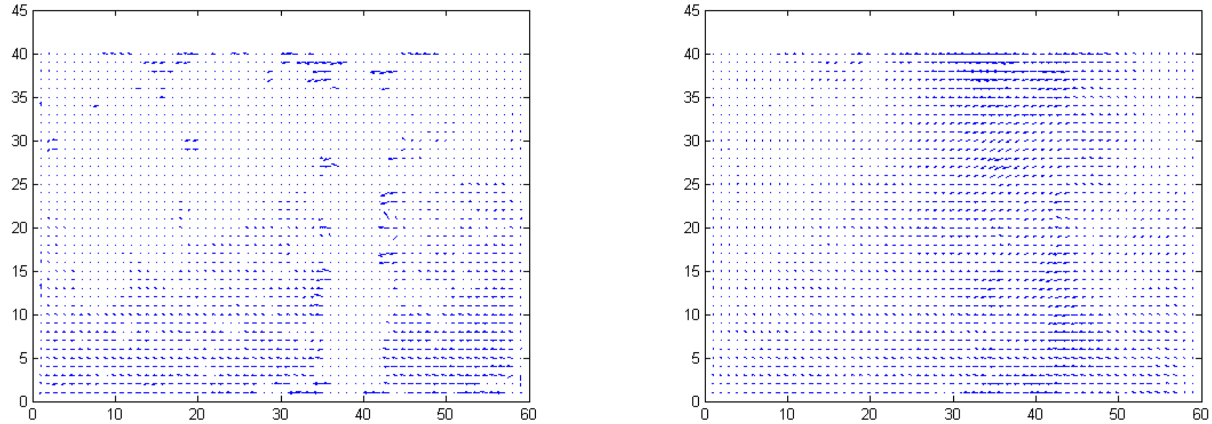


Figure 15: sparse optical flow

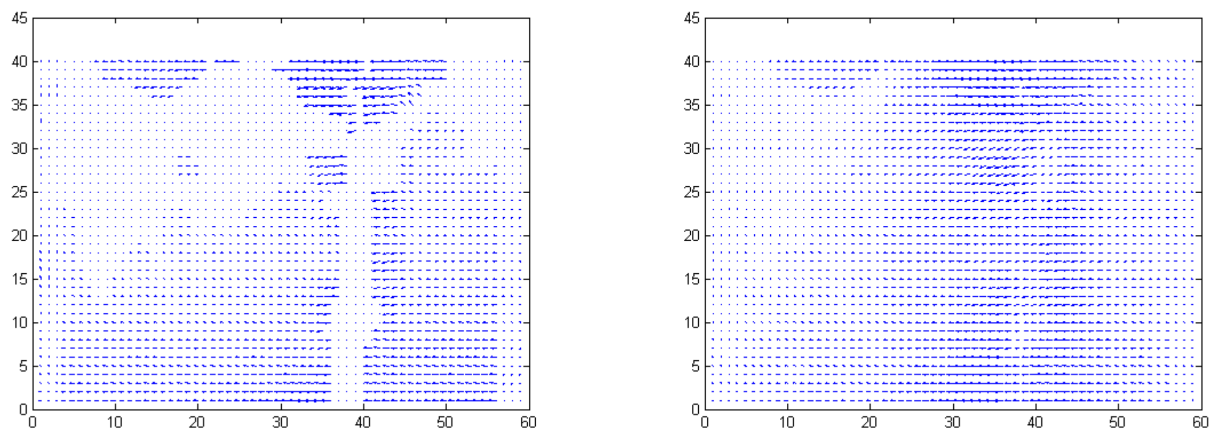
Problem 2. Iterative Coarse to Fine Optical Flow [10 points]

The result is as below:



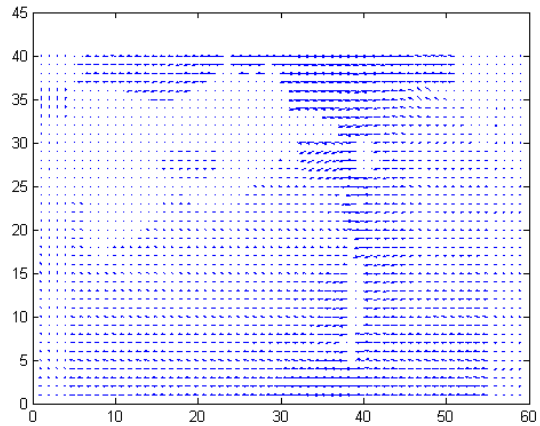
a b

Figure 16: (a)dense optical flow windowsize 5, (b)iterative optical flow windowsize:5

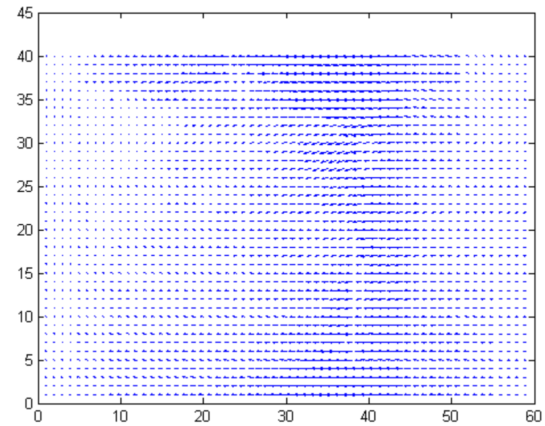


a b

Figure 17: (a)dense optical flow windowsize 15, (b)iterative optical flow windowsize:15



a



b

Figure 18: (a)dense optical flow windowsize 20, (b)iterative optical flow windowsize:20

Comments: When window size is small, the one-level dense optical flow struggle with the points with large movement distance(larger than 1 pixel) because this breaks the assumption that optical flow is based on small motion. While the iterative optical flow algorithm is better because in low resolution images, the movement distance is not that large as high resolution images. With the increasing of the window size, both dense optical flow and iterative optical flow becomes more clear. The outline of the tree is more clear. But actually iterative optical flow is not strictly better than standard dense optical flow. Because the interpolation and other calculation in the process may cause some error.

Problem 3. Background Subtraction [5 points]

The result is like:(with tau=0.4, alpha=0.06)

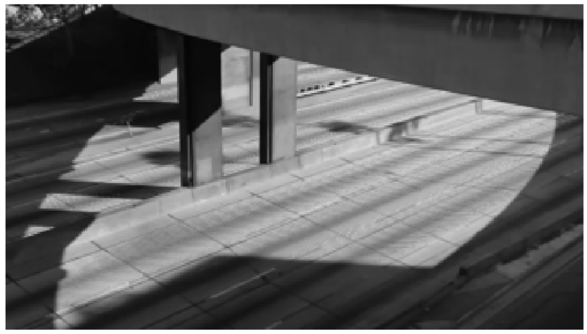


Figure 19: background substraction: highway

Figure 20: background substraction: truck

I use median as the initial value of B, and update it when I find a pixel is belong to background.

Problem 4. Motion Segmentation [5 points]

The result is like:

Figure 21: motion segmentation: image 29 and 30

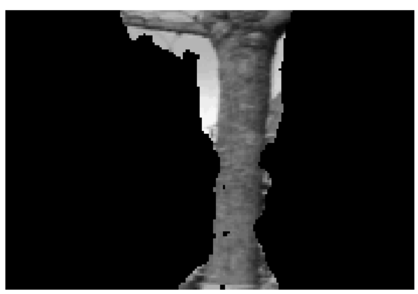


Figure 22: motion segmentation: image 35 and 36

Listing 1: optical flow function

```
function [u, v, hitMap] = opticalFlow(I1,I2,I1warp,windowSize, tau)
%% apply gaussian filter
smoothSTD = 1;
GaussFil = fspecial('gaussian',[3 3],smoothSTD);
I1 = imfilter(I1,GaussFil,'same');
I2 = imfilter(I2,GaussFil,'same');
I1warp = imfilter(I1warp,GaussFil,'same');
%% kernel derivative
kernel = 1/12*[-1,8,0,-8,1];
Dx_1 = conv2(I1,kernel, 'same');
Dx_2 = conv2(I2,kernel, 'same');
Dy_1 = conv2(I1,kernel','same');
Dy_2 = conv2(I2,kernel','same');
Ix = (Dx_1 + Dx_2) / 2;
Iy = (Dy_1 + Dy_2) / 2;
It = I2 - I1warp;

%% dense optical flow
tauhit=tau;
Ix=padarray(Ix,[floor(windowSize/2),floor(windowSize/2)],'symmetric','both');
Iy=padarray(Iy,[floor(windowSize/2),floor(windowSize/2)],'symmetric','both');
It=padarray(It,[floor(windowSize/2),floor(windowSize/2)],'symmetric','both');
height = size(Ix, 1);width = size(Ix, 2);
for i=(1+floor(windowSize/2)):(height-floor(windowSize/2))
    for j=(1+floor(windowSize/2)):(width-floor(windowSize/2))
        A=zeros(2,2);B=zeros(2,1);
        for m=i-floor(windowSize/2):i+floor(windowSize/2)
            for n=j-floor(windowSize/2):j+floor(windowSize/2)
                B(1,1)=B(1,1) + It(m,n)*Ix(m,n);
                B(2,1)=B(2,1) + It(m,n)*Iy(m,n);
                A(1,1)=A(1,1) + Ix(m,n)*Ix(m,n);
                A(1,2)=A(1,2) + Ix(m,n)*Iy(m,n);
                A(2,1)=A(2,1) + Ix(m,n)*Iy(m,n);
```

```
                A(2,2)=A(2,2) + Iy(m,n)*Iy(m,n);
            end
        end
        Ahit=A;
        [V,D]=eig(A);
        [Vhit,Dhit]=eig(Ahit);
        lamda = min(D(1,1),D(2,2));
        lamdahit = min(Dhit(1,1),Dhit(2,2));
        if lamda < tau
            u(i-floor(windowSize/2),j-floor(windowSize/2))=0;
            v(i-floor(windowSize/2),j-floor(windowSize/2))=0;
        else
        Ainv=inv(A);
        result=Ainv*(-B);
        u(i-floor(windowSize/2),j-floor(windowSize/2))=result(1,1);
        v(i-floor(windowSize/2),j-floor(windowSize/2))=result(2,1);
        end
        if lamdahit<tauhit
            b(i-floor(windowSize/2),j-floor(windowSize/2))=0;
        else
            b(i-floor(windowSize/2),j-floor(windowSize/2))=1;
        end
    end
end
hitMap=b;
end
```

Listing 2: script for dense and sparse optical flow

```
 %% load image
clear;clc;
%I1=im2double(imread('F:\252\corridor\bt.000.png'));
%I2=im2double(imread('F:\252\corridor\bt.001.png'));
%I1=im2double(imread('F:\252\synth\synth_000.png'));
%I2=im2double(imread('F:\252\synth\synth_001.png'));
```

```
% I1=im2double(rgb2gray(imread('F:\252\sphere\sphere.0.png')));
% I2=im2double(rgb2gray(imread('F:\252\sphere\sphere.1.png')));
%I1=im2double(rgb2gray(imread('F:\252\flower\00029.png')));
%I2=im2double(rgb2gray(imread('F:\252\flower\00036.png')));
I1=im2double(rgb2gray(imread('F:\252\1.jpg')));
I2=im2double(rgb2gray(imread('F:\252\2.jpg')));
I1 = imresize(I1,[256 256]);
I2 = imresize(I2,[256 256]);
% max1=max(max(I1)); max2=max(max(I2));
% I1=I1/max1;I2=I2/max2;
I1ori=I1;
height1 = size(I1, 1);width1 = size(I1, 2);
height2 = size(I2, 1);width2 = size(I2, 2);
windowSize=25;
tau=0.09;
NumIteration=4;
nCorners = 50;
windowSize2 = 7;
smoothSTD = 1;

%% corner detection
[corners1y corners1x] = CornerDetect(I1, nCorners, smoothSTD, windowSize2);
corners1 = [corners1x corners1y];

%% dense optical flow
u = zeros(size(I1));
v = zeros(size(I1));
uin = zeros(size(I1));
vin = zeros(size(I1));
for j=1:NumIteration
        I1warp = warp2(I1,uin,vin);
        [uin vin hitMap] = opticalFlow(I1,I2,I1warp,windowSize, tau);
        u = u + uin;
        v = v + vin;
```

```
end

%% sparse optical flow

for i=1:nCorners
      qu(i)=u(corners1x(i),corners1y(i));
      qv(i)=v(corners1x(i),corners1y(i));
end
%% show result

figure(1);
imshow(I1ori);
hold on;
axis on;
  for i=1:nCorners
      r = 5;
      sita=0:pi/20:2*pi;
      plot(corners1y(i)+r*cos(sita),corners1x(i)+r*sin(sita),'r','LineWidth',2);
  end
hold on;
quiver(corners1y,corners1x,qu',qv','LineWidth',3);
figure(2);
imshow(I1ori);
hold on;
axis on;
  for i=1:nCorners
      r = 5;
      sita=0:pi/20:2*pi;
      plot(corners1y(i)+r*cos(sita),corners1x(i)+r*sin(sita),'r','LineWidth',2);
  end
u=flipud(u); v=flipud(-v);
endd1=size(I1,1);
endd2=size(I1,2);
stride=6;
```

```
u1 = u(1:stride:endd1,1:stride:endd2);v1 = v(1:stride:endd1,1:stride:endd2);
figure(3);
quiver(u1, v1, 1.5, 'LineWidth',2);
figure(4);
imshow(hitMap);
```

Listing 3: script for iterative optical flow

```
 %% load image
clear;clc;
%I1=im2double(imread('F:\252\corridor\bt.000.png'));
%I2=im2double(imread('F:\252\corridor\bt.001.png'));
%I1=im2double(imread('F:\252\synth\synth_000.png'));
%I2=im2double(imread('F:\252\synth\synth_001.png'));
%I1=im2double(rgb2gray(imread('F:\252\sphere\sphere.0.png')));
%I2=im2double(rgb2gray(imread('F:\252\sphere\sphere.1.png')));
I1=im2double(rgb2gray(imread('F:\252\flower\00029.png')));
I2=im2double(rgb2gray(imread('F:\252\flower\00030.png')));
max1=max(max(I1)); max2=max(max(I2));
I1=I1/max1;I2=I2/max2;
% if mod(size(I1,1),2)~=0
%     I1=imresize(I1,[size(I1,1)+1,size(I1,2)]);
%     I2=imresize(I2,[size(I2,1)+1,size(I2,2)]);
% end
% if mod(size(I1,2),2)~=0
%     I1=imresize(I1,[size(I1,1),size(I1,2)+1]);
%     I2=imresize(I2,[size(I2,1),size(I2,2)+1]);
% end
windowSize=20;
tau=0.1;
NumPyramid = 6;
NumIteration = 10;
nCorners = 50;
windowSize2 = 7;
smoothSTD = 1;
```

```
%% make image pyramid
I1_pyramid{1}=I1;
I2_pyramid{1}=I2;

for i = 2:NumPyramid
    I1_pyramid{i} = impyramid( I1_pyramid{i-1}, 'reduce' );
    I2_pyramid{i} = impyramid( I2_pyramid{i-1}, 'reduce' );
end


%% pyramid LK optical flow

for k=NumPyramid:-1:1
    I1current=I1_pyramid{k};
    I2current=I2_pyramid{k};
    if k==NumPyramid
        u = zeros(size(I1_pyramid{k}));
        v = zeros(size(I1_pyramid{k}));
    end
    uin = zeros(size(I1_pyramid{k}));
    vin = zeros(size(I1_pyramid{k}));
     utmp = zeros(size(I1_pyramid{k}));
     vtmp = zeros(size(I1_pyramid{k}));
    % in-level iteration refinement
    I1warp=warp2(I1current,u,v);
     for j=1:NumIteration
         I1warp = warp2(I1current,utmp,vtmp);
         [uin vin hitMap] = opticalFlow(I1current,I2current,I1warp,windowSize, tau);
         uin = uin + utmp;
         vin = vin + vtmp;
     end
    u=u+uin;
    v=v+vin;
```

```
    if k~= 1
        sizerow=size(u,1); sizecol=size(u,2);
        sizerow2=2*sizerow; sizecol2=2*sizecol;
        if sizerow2~=size(I1_pyramid{k-1},1)
            sizerow2=size(I1_pyramid{k-1},1);
        end
        if sizecol2~=size(I1_pyramid{k-1},2)
            sizecol2=size(I1_pyramid{k-1},2);
        end
        u = 2 * imresize(u,[sizerow2 sizecol2],'bilinear');
        v = 2 * imresize(v,[sizerow2 sizecol2],'bilinear');
    end
end
%% sparse optical flow
%
% figure(1);
% quiver(flipud(u),flipud(-v));
figure(2);
endd1=size(I1,1);
endd2=size(I1,2);
stride=3;
u1 = u(1:stride:endd1,1:stride:endd2);v1 = v(1:stride:endd1,1:stride:endd2);
quiver(flipud(u1),flipud(-v1));
```

Listing 4: function for warp image

```
function [ I1warped] = warp2(I1,u,v)
[x y] = meshgrid(1:size(I1,2),1:size(I1,1));
I1warped = interp2(I1, x+u, y+v, 'cubic');
I1warped(isnan(I1warped)) = I1(isnan(I1warped));
end
```

Listing 5: function for background substraction

```
 % function [background] = backgroundSubtract(framesequence, tau)
% end
```

```
clear;clc;
%file_path =  'F:\252\highway\';
file_path =  'F:\252\truck\';
img_path_list = dir(strcat(file_path,'*.png'));
img_num = length(img_path_list);
if img_num > 0
          for i = 1:img_num
               image_name = img_path_list(i).name;
               tmpimage =  im2double(imread(strcat(file_path,image_name)));
               image{i} = tmpimage;
          end
end

tau=0.4;
a=0.05;
height=size(image{1},1);  width=size(image{1},2);
for i=1:img_num
     for m=1:height
          for n=1:width
               array{m}{n}(i)=image{i}(m,n);
          end
     end
end
B=zeros(height,width);
for m=1:height
     for n=1:width
          B(m,n)=median(array{m}{n});
     end
end

for i=1:img_num-1
     imgcu=image{i};
     imgne=image{i+1};
     for m=1:height
```

```
        for n=1:width
            if abs(imgcu(m,n)-imgne(m,n))>tau
                B(m,n)=(1-a)*B(m,n)+a*imgcu(m,n);
            else
                B(m,n)=B(m,n);
            end
        end
    end
end

figure(1);
imshow(B);
```

Listing 6: script for motion segmentation

```
%% load image
clear;clc;
%I1=im2double(imread('F:\252\corridor\bt.000.png'));
%I2=im2double(imread('F:\252\corridor\bt.001.png'));
%I1=im2double(imread('F:\252\synth\synth_000.png'));
%I2=im2double(imread('F:\252\synth\synth_001.png'));
%I1=im2double(rgb2gray(imread('F:\252\sphere\sphere.0.png')));
%I2=im2double(rgb2gray(imread('F:\252\sphere\sphere.1.png')));
I1=im2double(rgb2gray(imread('F:\252\flower\00035.png')));
I2=im2double(rgb2gray(imread('F:\252\flower\00036.png')));
max1=max(max(I1)); max2=max(max(I2));
I1=I1/max1;I2=I2/max2;
% if mod(size(I1,1),2)~=0
%     I1=imresize(I1,[size(I1,1)+1,size(I1,2)]);
%     I2=imresize(I2,[size(I2,1)+1,size(I2,2)]);
% end
% if mod(size(I1,2),2)~=0
%     I1=imresize(I1,[size(I1,1),size(I1,2)+1]);
%     I2=imresize(I2,[size(I2,1),size(I2,2)+1]);
% end
```

```
windowSize=10;
tau=0.005;
NumPyramid = 5;
NumIteration = 10;
windowSize2 = 7;
smoothSTD = 1;
th=9;
%% make image pyramid
I1_pyramid{1}=I1;
I2_pyramid{1}=I2;

for i = 2:NumPyramid
    I1_pyramid{i} = impyramid( I1_pyramid{i-1}, 'reduce' );
    I2_pyramid{i} = impyramid( I2_pyramid{i-1}, 'reduce' );
end


%% pyramid LK optical flow

for k=NumPyramid:-1:1
    I1current=I1_pyramid{k};
    I2current=I2_pyramid{k};
    if k==NumPyramid
        u = zeros(size(I1_pyramid{k}));
        v = zeros(size(I1_pyramid{k}));
    end
    uin = zeros(size(I1_pyramid{k}));
    vin = zeros(size(I1_pyramid{k}));
     utmp = zeros(size(I1_pyramid{k}));
     vtmp = zeros(size(I1_pyramid{k}));
    % in-level iteration refinement
    I1warp=warp2(I1current,u,v);
     for j=1:NumIteration
         I1warp = warp2(I1current,utmp,vtmp);
```

```
    [uin vin hitMap] = opticalFlow(I1current,I2current,I1warp,windowSize, tau);
          uin = uin + utmp;
          vin = vin + vtmp;
     end
    u=u+uin;
    v=v+vin;
    if k~= 1
        sizerow=size(u,1); sizecol=size(u,2);
        sizerow2=2*sizerow; sizecol2=2*sizecol;
        if sizerow2~=size(I1_pyramid{k-1},1)
            sizerow2=size(I1_pyramid{k-1},1);
        end
        if sizecol2~=size(I1_pyramid{k-1},2)
            sizecol2=size(I1_pyramid{k-1},2);
        end
        u = 2 * imresize(u,[sizerow2 sizecol2],'bilinear');
        v = 2 * imresize(v,[sizerow2 sizecol2],'bilinear');
    end
end
%% sparse optical flow

for i=1:size(I1,1)
    for j=1:size(I1,2)
        if sqrt(u(i,j)^2+v(i,j)^2)>th
            tree(i,j)=I1(i,j);
        else
            tree(i,j)=0;
        end
    end
end

figure(1);
imshow(tree);
```

Submitted by Mingxuan Wang on December 17, 2014.