Exercise 3 report, title "Optical flow", Ntambaazi Tonny_CIMET : ...,

Table of Contents

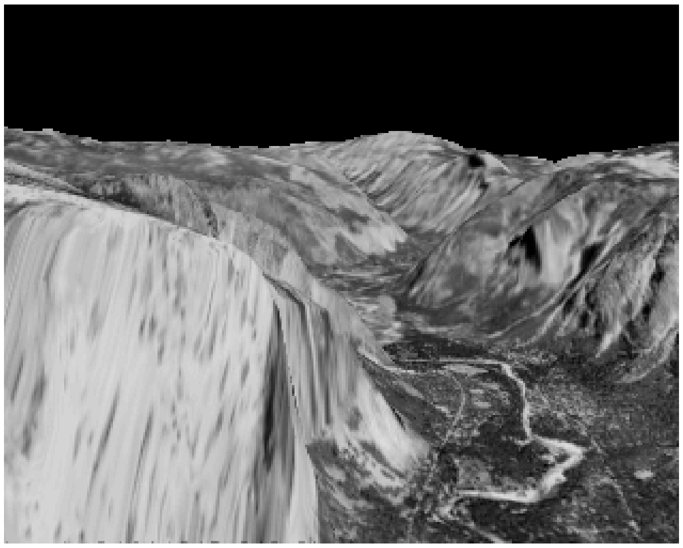
0/ Test image computation

```
clear;close all;clc
% N=256;
% tx=0.43789;ty=0.87230;
% X0=N/2+1;Y0=N/2+1;
%
% [X,Y]=meshgrid(1:N,1:N);
% I1=exp(-((X-X0).^2+(Y-Y0).^2)/(2*sigm^2));
% figure(1);imshow(I1,[]); axis on;title('gaussian function
 1');colorbar;hold on ; plot(X0,Y0,'+')
% I2=exp(-((X-X0-tx).^2+(Y-Y0-ty).^2)/(2*sigm^2));%
+randn(size(I1))*sig
% figure(2);imshow(I2,[]); axis on;title('gaussian function
 2');colorbar;hold on ;plot(X0,Y0,'+');plot(X0+tx,Y0+ty,'r+')
```

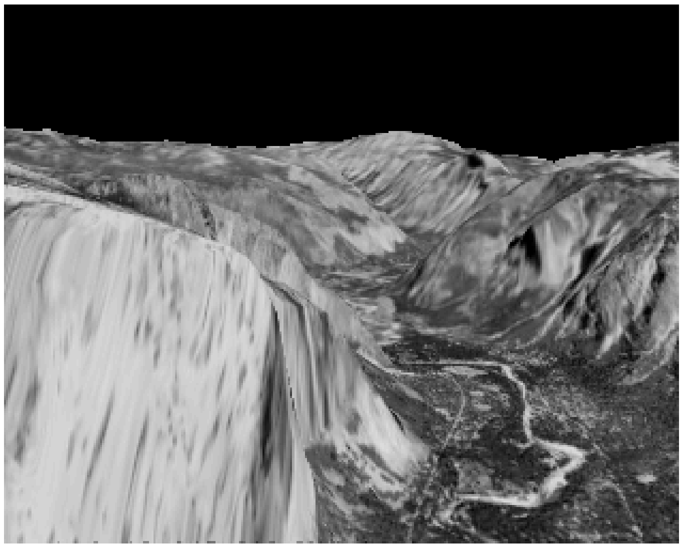
1. Simulate and display the two images: Compute the gradient images Ix and Iy of I1.

```
images_1 = imread('yos_img_10.pgm');
figure; imshow(images_1); title('First Image');
images_2 = imread('yos_img_11.pgm');
figure; imshow(images_2); title('Second Image');
window_size = 15;
Alpha = 0.5;
```

First Image



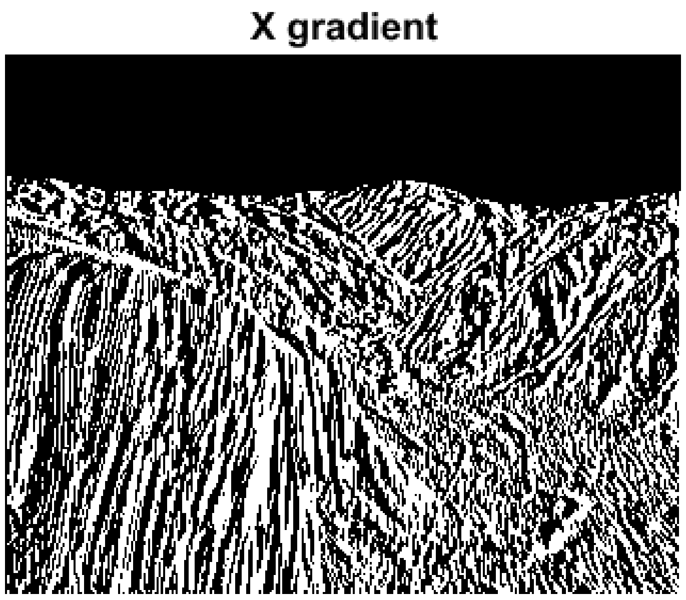
Second Image

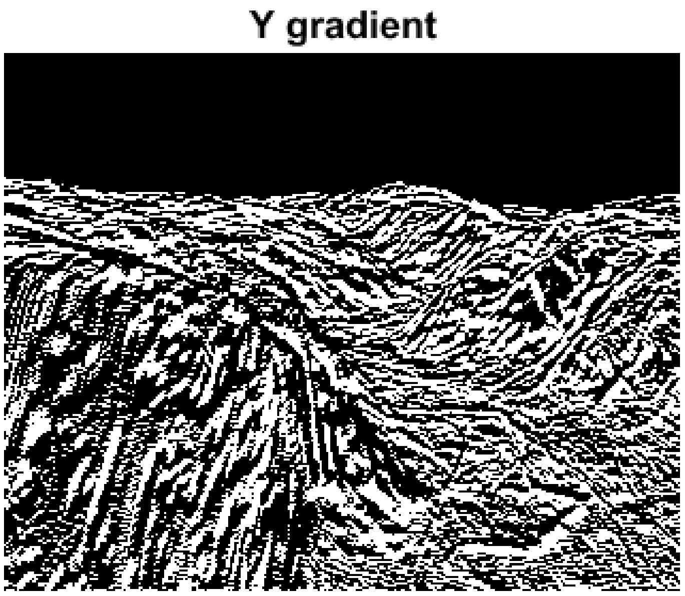


1. Compute the gradient images Ix and Iy of I1.

```
Ix = conv2(images_1, Alpha * [-1 1; -1 1]) + conv2(images_2, Alpha *
 [-1 1; -1 1]);
figure; imshow(Ix); title('X gradient');
Iy = conv2(images_1, Alpha * [-1 -1; 1 1]) + conv2(images_2, Alpha *
 [-1 -1; 1 1]);
figure; imshow(Iy); title('Y gradient');
```

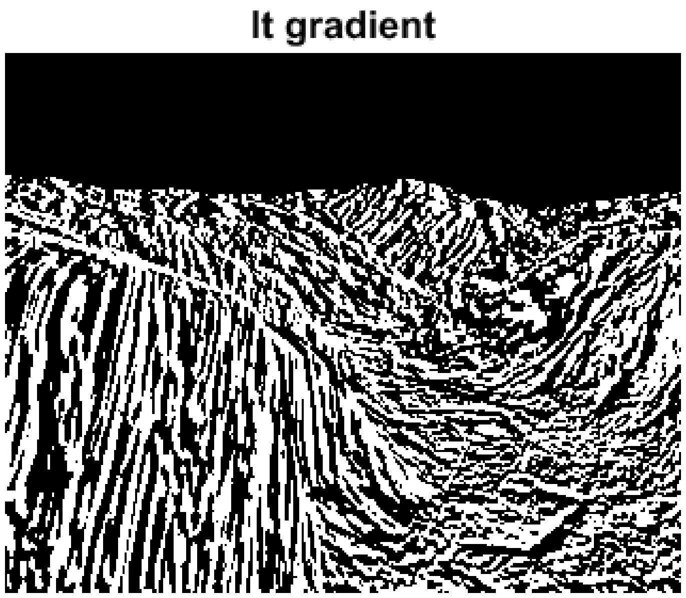
```
Warning: CONV2 on values of class UINT8 is obsolete.
         Use CONV2(DOUBLE(A),DOUBLE(B)) or CONV2(SINGLE(A),SINGLE(B))
 instead.
Warning: CONV2 on values of class UINT8 is obsolete.
         Use CONV2(DOUBLE(A),DOUBLE(B)) or CONV2(SINGLE(A),SINGLE(B))
 instead.
Warning: CONV2 on values of class UINT8 is obsolete.
         Use CONV2(DOUBLE(A),DOUBLE(B)) or CONV2(SINGLE(A),SINGLE(B))
 instead.
Warning: CONV2 on values of class UINT8 is obsolete.
         Use CONV2(DOUBLE(A),DOUBLE(B)) or CONV2(SINGLE(A),SINGLE(B))
 instead.
```





2. Compute the gradient image It

```
It = conv2(images_1, Alpha * ones(2)) + conv2(images_2, -Alpha *
 ones(2));
figure; imshow(It); title('It gradient');

Warning: CONV2 on values of class UINT8 is obsolete.
         Use CONV2(DOUBLE(A),DOUBLE(B)) or CONV2(SINGLE(A),SINGLE(B))
 instead.
Warning: CONV2 on values of class UINT8 is obsolete.
         Use CONV2(DOUBLE(A),DOUBLE(B)) or CONV2(SINGLE(A),SINGLE(B))
 instead.
```



3. Considering a neighborhood of n pixels, write the problem in a matrix form.

```
window_center = floor(window_size / 2);
image_size = size(images_1);
u = zeros(image_size);
v = zeros(image_size);
for i = window_center + 1:image_size(1) - window_center
  for j = window_center + 1:image_size(2) - window_center
    % Get values for current window
    Ix_window = Ix(i - window_center:i + window_center, j -
 window_center:j + window_center);
    Iy_window = Iy(i - window_center:i + window_center, j -
 window_center:j + window_center);
    It_window = It(i - window_center:i + window_center, j -
 window_center:j + window_center);

    Ix_window = Ix_window';
    Iy_window = Iy_window';
    It_window = It_window';

    % Computing the matrix for all pixels i.e part 3
    mat = [Ix_window(:) Iy_window(:)];

    % compute velocity
    U = pinv(mat' * mat) * mat' * -It_window(:);
```

```
        u(i, j) = U(1);
        v(i, j) = U(2);
    end
end
```

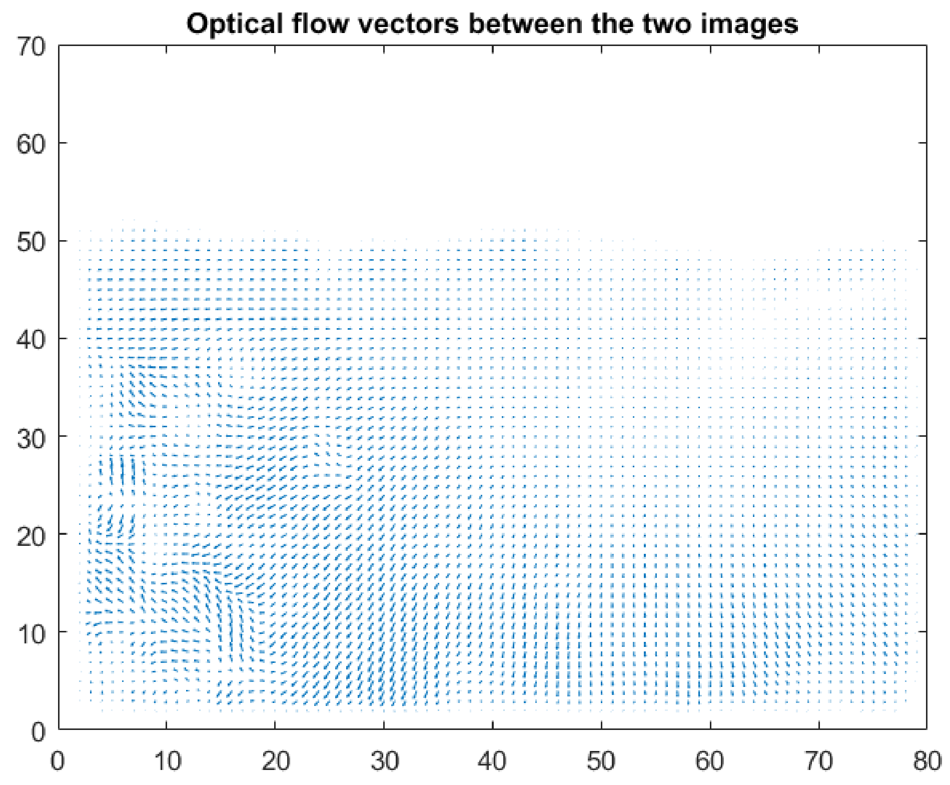
4. Theoretically, what is the minimum number of pixel-neighbors to consider in order to solve this problem.

Theoretically, there are only 2 unknown values for Vx, and Vy, therfore, for computation of the optical flow we have to use only 2 pixels for computation of the optical flow

5. Implement a program computing the optical flow in one point in the image and that compute the error of the shift estimation.

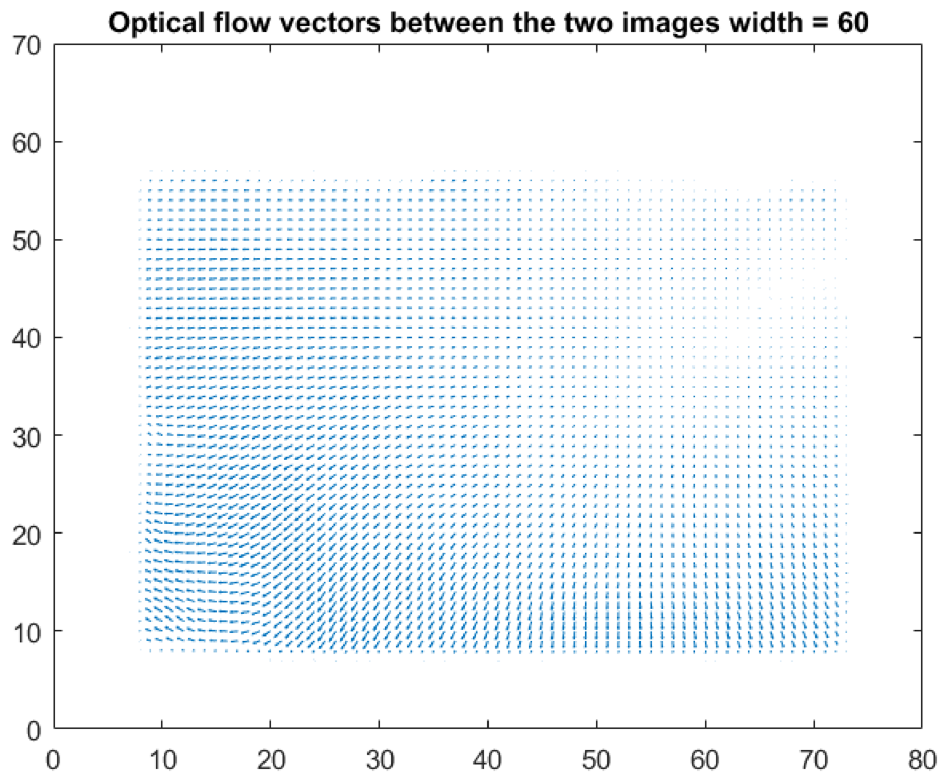
Display the result

```
figure
axis equal
quiver(impyramid(impyramid(medfilt2(flipud(u), [5
 5]), 'reduce'), 'reduce'), -impyramid(impyramid(medfilt2(flipud(v),
 [5 5]), 'reduce'), 'reduce'));
title ('Optical flow vectors between the two images')
```



6 The effect of increasing neighbourhood width

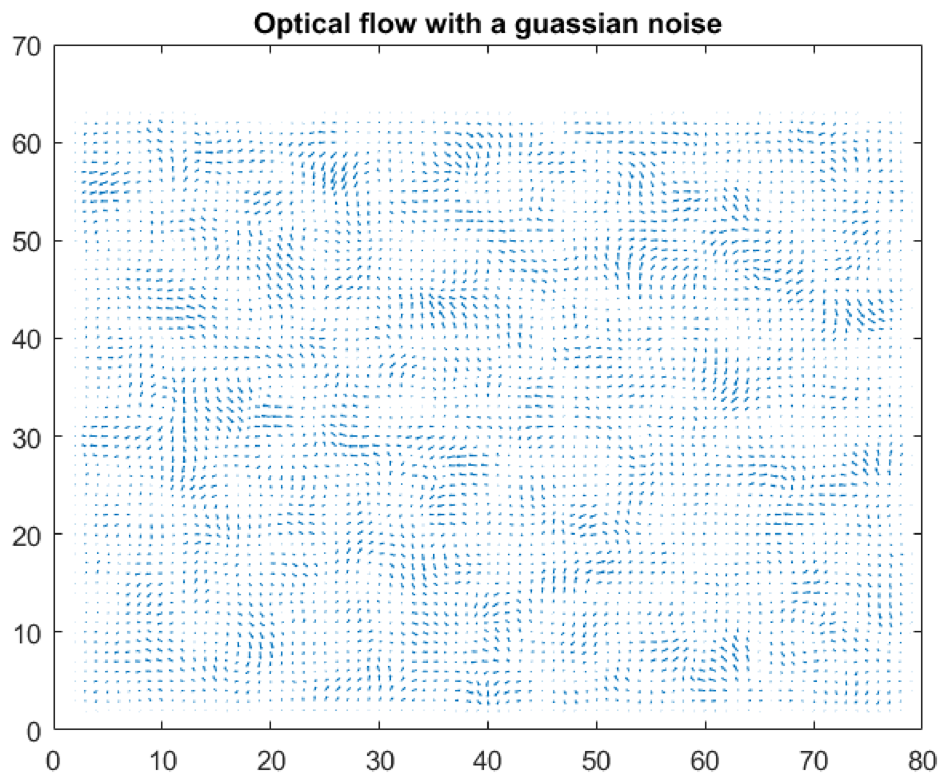
```
window_size = 60;
window_center = floor(window_size / 2);
image_size = size(images_1);
u = zeros(image_size);
v = zeros(image_size);
for i = window_center + 1:image_size(1) - window_center
  for j = window_center + 1:image_size(2) - window_center
    % Get values for current window
    Ix_window = Ix(i - window_center:i + window_center, j -
 window_center:j + window_center);
    Iy_window = Iy(i - window_center:i + window_center, j -
 window_center:j + window_center);
    It_window = It(i - window_center:i + window_center, j -
 window_center:j + window_center);

    Ix_window = Ix_window';
    Iy_window = Iy_window';
    It_window = It_window';

    % Computing the matrix for all pixels i.e part 3
    mat = [Ix_window(:) Iy_window(:)];

    % compute velocity
    U = pinv(mat' * mat) * mat' * -It_window(:);

    u(i, j) = U(1);
    v(i, j) = U(2);
  end
end

figure
axis equal
quiver(impyramid(impyramid(medfilt2(flipud(u), [5
 5]), 'reduce'), 'reduce'), -impyramid(impyramid(medfilt2(flipud(v),
 [5 5]), 'reduce'), 'reduce'));
title ('Optical flow vectors between the two images width = 60')

%Increasing the width of the neighbourhood causes a shift of the
%displacement field to the inside of the search window thus reducing
 the
%size of the displacement field.  a  small  region considered for
%calculating the optical flow causes large motions get out  of the
 local
%window which is why a larger displacement field is observed for
 smaller
%neighbours than larger neighboring sizes:
```



Effect of noise on optical flow

```
window_size = 15;
Alpha = 0.5;

% adding noise to the images
images_1 = imnoise (images_1/255, 'gaussian', 0.2);
images_2 = imnoise (images_2/255, 'gaussian', 0.2);


Ix = conv2(images_1, Alpha * [-1 1; -1 1]) + conv2(images_2, Alpha *
 [-1 1; -1 1]);
Iy = conv2(images_1, Alpha * [-1 -1; 1 1]) + conv2(images_2, Alpha *
 [-1 -1; 1 1]);
It = conv2(images_1, Alpha * ones(2)) + conv2(images_2, -Alpha *
 ones(2));

window_center = floor(window_size / 2);
image_size = size(images_1);
u = zeros(image_size);
v = zeros(image_size);
for i = window_center + 1:image_size(1) - window_center
  for j = window_center + 1:image_size(2) - window_center
    % Get values for current window
    Ix_window = Ix(i - window_center:i + window_center, j -
 window_center:j + window_center);
    Iy_window = Iy(i - window_center:i + window_center, j -
 window_center:j + window_center);
    It_window = It(i - window_center:i + window_center, j -
 window_center:j + window_center);

    Ix_window = Ix_window';
    Iy_window = Iy_window';
    It_window = It_window';

    % Computing the matrix for all pixels i.e part 3
    mat = [Ix_window(:) Iy_window(:)];

    % compute velocity
    U = pinv(mat' * mat) * mat' * -It_window(:);

    u(i, j) = U(1);
    v(i, j) = U(2);
  end
end

figure
axis equal
quiver(impyramid(impyramid(medfilt2(flipud(u), [5
 5]), 'reduce'), 'reduce'), -impyramid(impyramid(medfilt2(flipud(v),
 [5 5]), 'reduce'), 'reduce'));
title ('Optical flow with a guassian noise')
```

```
% Adding noise to the images used for testing optical flow causes a
% distortion with in the displacement field of the optical flow as
 observed in the figure
% showing the effect of guassian noise to original images. In this
% case, there will be a change in the image intensity thus causing a
 change
% in the velocities of the two images which results into a different
% diplacement field than the original displacement field.

Warning: CONV2 on values of class UINT8 is obsolete.
         Use CONV2(DOUBLE(A),DOUBLE(B)) or CONV2(SINGLE(A),SINGLE(B))
 instead.
Warning: CONV2 on values of class UINT8 is obsolete.
         Use CONV2(DOUBLE(A),DOUBLE(B)) or CONV2(SINGLE(A),SINGLE(B))
 instead.
Warning: CONV2 on values of class UINT8 is obsolete.
         Use CONV2(DOUBLE(A),DOUBLE(B)) or CONV2(SINGLE(A),SINGLE(B))
 instead.
Warning: CONV2 on values of class UINT8 is obsolete.
         Use CONV2(DOUBLE(A),DOUBLE(B)) or CONV2(SINGLE(A),SINGLE(B))
 instead.
Warning: CONV2 on values of class UINT8 is obsolete.
         Use CONV2(DOUBLE(A),DOUBLE(B)) or CONV2(SINGLE(A),SINGLE(B))
 instead.
Warning: CONV2 on values of class UINT8 is obsolete.
         Use CONV2(DOUBLE(A),DOUBLE(B)) or CONV2(SINGLE(A),SINGLE(B))
 instead.
```

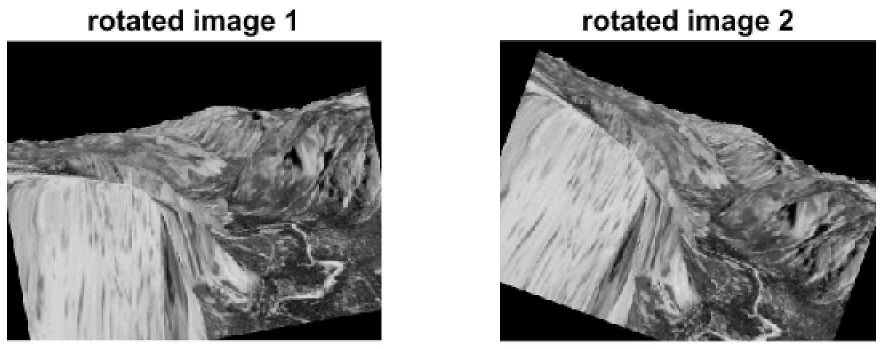


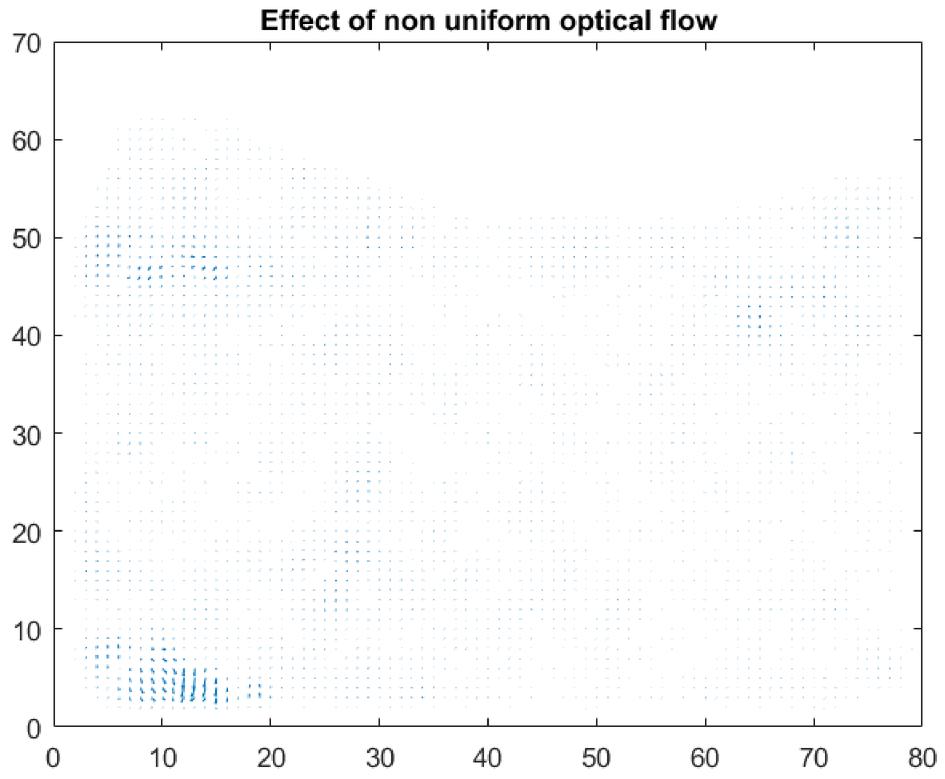
Effect of non uniform optical flow

```
% loading images
images_1 = imread('yos_img_10.pgm');
images_2 = imread('yos_img_11.pgm');

% rotating images
images_1 = imrotate(images_1,10,'bilinear','crop');
images_2 = imrotate(images_2,-20,'bilinear','crop');

%Comparing the images
figure; subplot(1,2,1); title('comparing rotated images');
imshow(images_1); title('rotated image 1');
 subplot(1,2,2); imshow(images_2); title('rotated image 2');


Ix = conv2(images_1, Alpha * [-1 1; -1 1]) + conv2(images_2, Alpha *
 [-1 1; -1 1]);
Iy = conv2(images_1, Alpha * [-1 -1; 1 1]) + conv2(images_2, Alpha *
 [-1 -1; 1 1]);
It = conv2(images_1, Alpha * ones(2)) + conv2(images_2, -Alpha *
 ones(2));

window_center = floor(window_size / 2);
image_size = size(images_1);
```

```
u = zeros(image_size);
v = zeros(image_size);
for i = window_center + 1:image_size(1) - window_center
  for j = window_center + 1:image_size(2) - window_center
    % Get values for current window
    Ix_window = Ix(i - window_center:i + window_center, j -
 window_center:j + window_center);
    Iy_window = Iy(i - window_center:i + window_center, j -
 window_center:j + window_center);
    It_window = It(i - window_center:i + window_center, j -
 window_center:j + window_center);

    Ix_window = Ix_window';
    Iy_window = Iy_window';
    It_window = It_window';

    % Computing the matrix for all pixels i.e part 3
    mat = [Ix_window(:) Iy_window(:)];

    % compute velocity
    U = pinv(mat' * mat) * mat' * -It_window(:);

    u(i, j) = U(1);
    v(i, j) = U(2);
  end
end

figure
axis equal
quiver(impyramid(impyramid(medfilt2(flipud(u), [5
 5]), 'reduce'), 'reduce'), -impyramid(impyramid(medfilt2(flipud(v),
 [5 5]), 'reduce'), 'reduce'));
title ('Effect of non uniform optical flow')

% Non uniform optical flow causes a dramatic channge in the intensity
 of
% the image thus reducing the correlation of the displacement pixels
 that
% are computed on obtaining the optical flow. Here there is also a
% reduction on the stability of the observed displacement field
 observed in
% the resulting plot

Warning: CONV2 on values of class UINT8 is obsolete.
         Use CONV2(DOUBLE(A),DOUBLE(B)) or CONV2(SINGLE(A),SINGLE(B))
 instead.
Warning: CONV2 on values of class UINT8 is obsolete.
         Use CONV2(DOUBLE(A),DOUBLE(B)) or CONV2(SINGLE(A),SINGLE(B))
 instead.
Warning: CONV2 on values of class UINT8 is obsolete.
         Use CONV2(DOUBLE(A),DOUBLE(B)) or CONV2(SINGLE(A),SINGLE(B))
 instead.
Warning: CONV2 on values of class UINT8 is obsolete.
```

```
         Use CONV2(DOUBLE(A),DOUBLE(B)) or CONV2(SINGLE(A),SINGLE(B))
 instead.
Warning: CONV2 on values of class UINT8 is obsolete.
         Use CONV2(DOUBLE(A),DOUBLE(B)) or CONV2(SINGLE(A),SINGLE(B))
 instead.
Warning: CONV2 on values of class UINT8 is obsolete.
         Use CONV2(DOUBLE(A),DOUBLE(B)) or CONV2(SINGLE(A),SINGLE(B))
 instead.
```





Published with MATLAB® R2016a