MULTIMEDIA SYSTEMS

ELECTRICAL AND COMPUTER ENGINEERING
COLLEGE OF ENGINEERING
UIC

Submitted By:

Bhuvan Jain (UIN: 667704103)

PROJECT AIM

To compare the motion analysis of a digital video stream using the following methods:

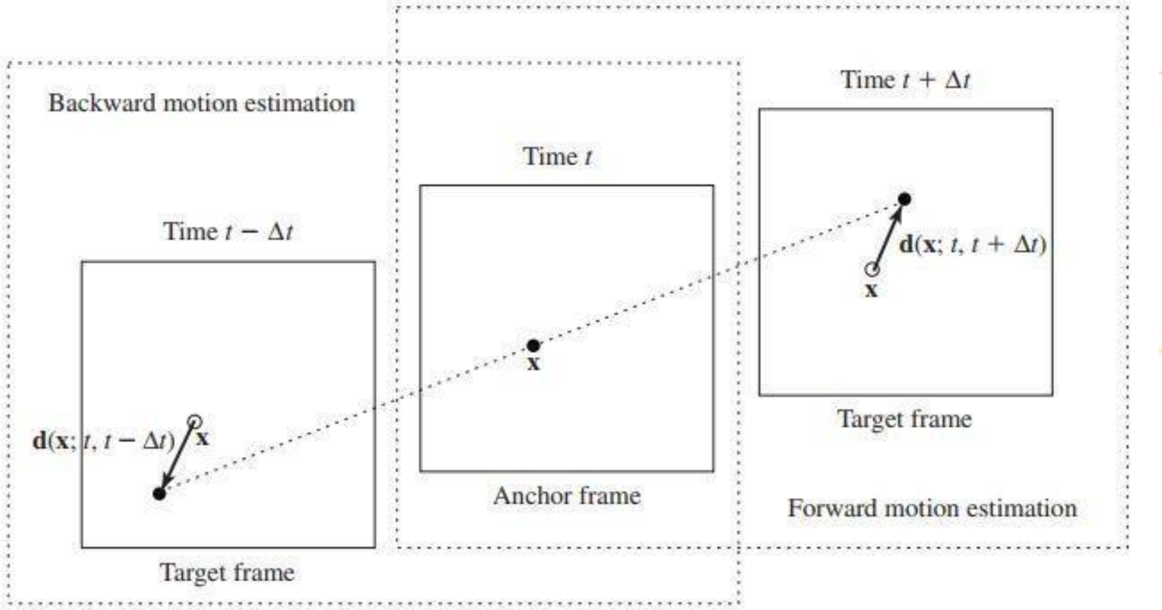
(a) **Optical Flow**.
(b) **Displaced Frame Difference.**

To perform each method for the following applications:

(1) **Pixel-based motion analysis**.
(2) **Block-based motion analysis.**
(3) **Region-based motion analysis.**
(4) **Global motion analysis.**

OVERVIEW

Forward and Backward Motion Estimation



Notations

Anchor frame: $\Psi_1(\mathbf{x})$

Target frame: $\Psi_2(\mathbf{x})$

Motion parameters: $\mathbf{a}$

Motion vector at a pixel in the anchor frame: $\mathbf{d}(\mathbf{x})$

Motion field: $\mathbf{d}(\mathbf{x};\mathbf{a})$, $\mathbf{x} \in \Lambda$

$\mathbf{w}(\mathbf{x};\mathbf{a}) = \mathbf{x} + \mathbf{d}(\mathbf{x};\mathbf{a})$, $x \in \Lambda$

Motion Estimation Criteria based on Displaced Frame Difference

The most popular criterion for motion estimation is to minimize the sum of the errors between the luminance values of every pair of corresponding points between the anchor frame Ψ_1 and the target frame Ψ_2. x in Ψ_1 is moved to **w(x;a) in Ψ_2.** Therefore, the objective function can be written as,

$$E_{\mathrm{DFD}}(\mathbf{a}) = \sum_{x\in\Lambda} \left|\psi_2(\mathbf{x}+\mathbf{d}(\mathbf{x};\mathbf{a})) - \psi_1(\mathbf{x})\right|^p \rightarrow \min$$

When p = 1, the above error is called mean absolute difference (MAD), and when p = 2, it is mean squared error (MSE).

Motion Estimation Criteria based on Optical Flow

When illumination condition is unknown, optical flow equation is unknown. Assuming illumination to be constant, optical flow equation is given by

Under "constant intensity assumption":

$$\psi(x+d_x, y+d_y, t+d_t) = \psi(x,y,t)$$

But, using Taylor's expansion :

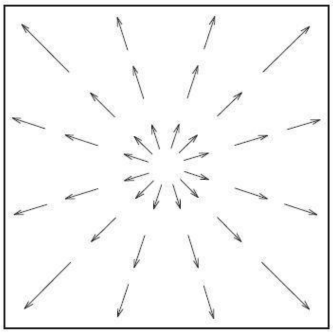
$$\psi(x+d_x, y+d_y, t+d_t) = \psi(x,y,t) + \frac{\partial\psi}{\partial x}d_x + \frac{\partial\psi}{\partial y}d_y + \frac{\partial\psi}{\partial y}d_t$$

Compare the above two, we have the optical flow equation :

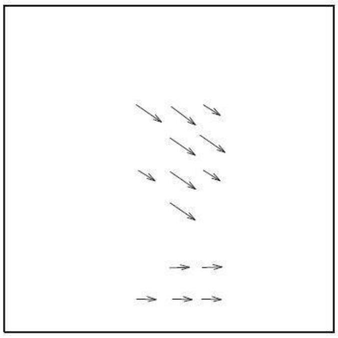
$$\frac{\partial\psi}{\partial x}d_x + \frac{\partial\psi}{\partial y}d_y + \frac{\partial\psi}{\partial t}d_t = 0 \quad \text{or} \quad \frac{\partial\psi}{\partial x}v_x + \frac{\partial\psi}{\partial y}v_y + \frac{\partial\psi}{\partial t} = 0 \ \text{or} \ \nabla\psi^T\mathbf{v} + \frac{\partial\psi}{\partial t} = 0$$

Motion Representation

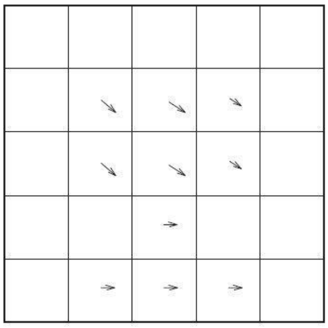
a) **Global**: Entire motion field is represented by a few global parameters.



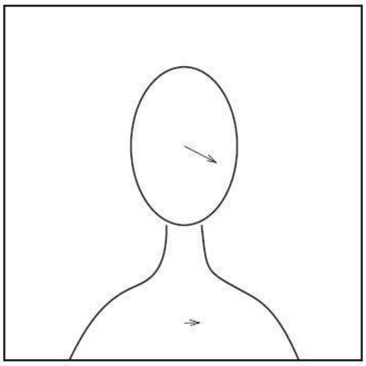
b) **Pixel based**: One MV at each pixel, with some smoothness constraint between adjacent MVs.



c) **Block based**: Entire frame is divided into blocks, and motion in each block is characterized by a few parameters.



d) **Region based**: Entire frame is divided into regions, each region corresponding to an object or sub-object with consistent motion, represented by a few parameters.



DETAILED PROJECT DESCRIPTION

Displaced Frame Difference

1. Pixel Based Motion Analysis

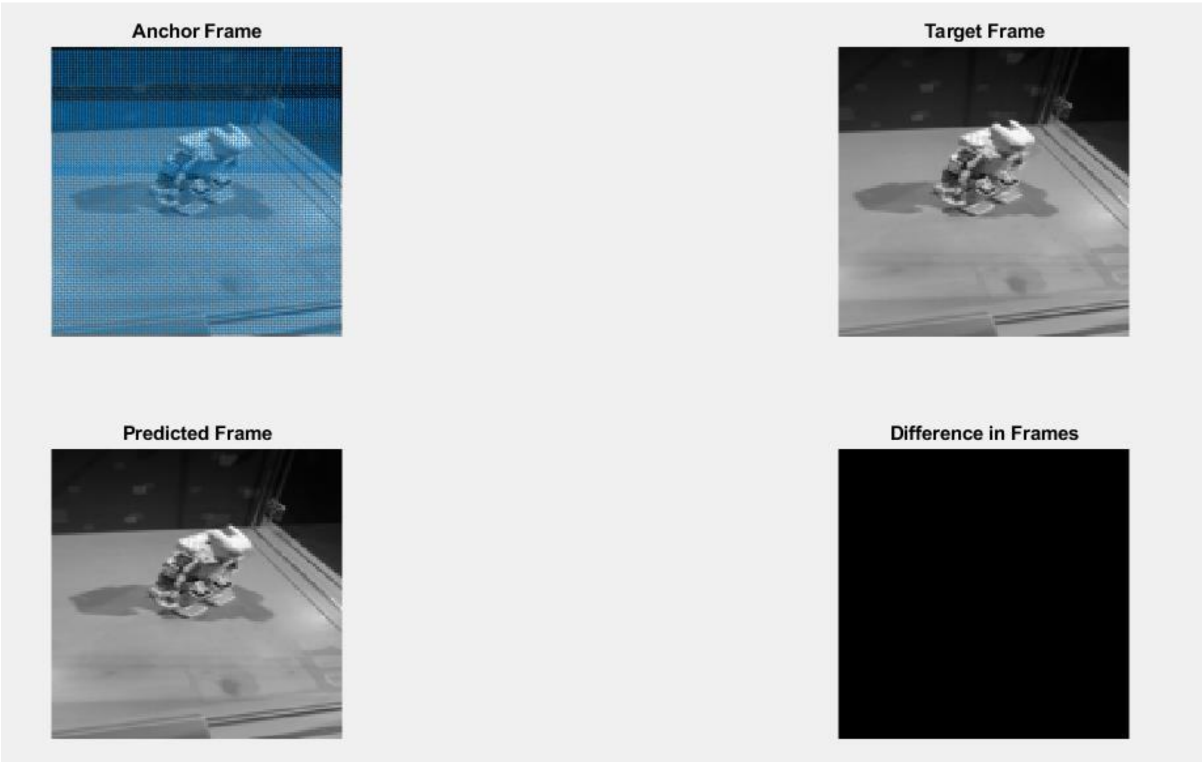
In Pixel based estimation the block size is kept equal to that of 1 pixel of an image. Using this approach we can predict the motion of each pixel of the image and hence get the best result. But it is time consuming and requires more information to be transmitted.

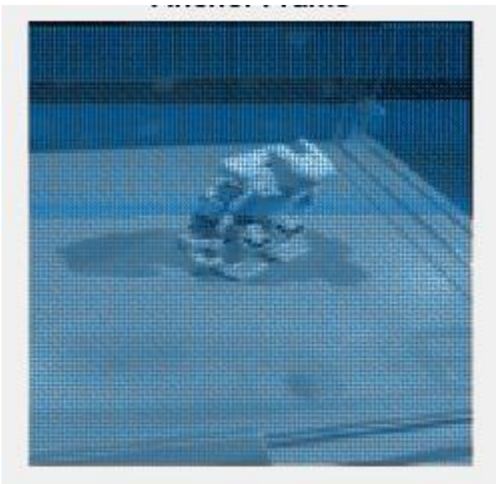
Code:

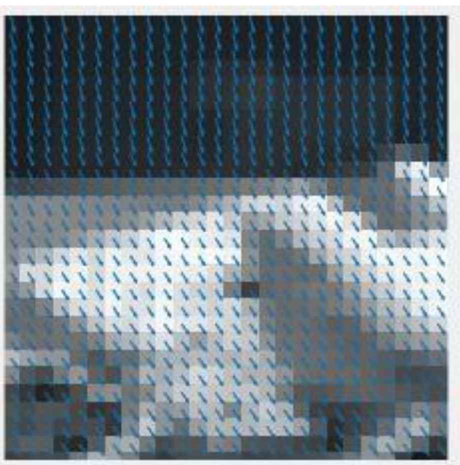
```
video= VideoReader('vidi.mp4');
num_frames = video.NumberOfFrames;

    for f=230:230     %reading 1 frames
       anchorframe = imresize(im2double(rgb2gray(read(video,f))),[128,
128]);  %anchor frame 128x128
       targetframe = imresize(im2double(rgb2gray(read(video,f+1))),[128,
128]);  %target frame 128x128

       [row,col] = size(anchorframe);
       wind = 1;      % for pixel , size 1x1;
       for m = 1:wind:row
          for n = 1:wind:col
             point(1) = m;
             point(2) = n;
             anchorblock = anchorframe(point(1):point(1)+wind-
1,point(2):point(2)+wind-1);
             min_dist = Inf;
             Block_match = 0;
             mv_x=[]; mv_y=[]; ii = 1;  % variable to store motion vector
along x-horizontal, y-vertical side

             for i = 1:row-wind
                 jj=1;
                 for j = 1:col-wind
                   targetblock = targetframe(i:i+wind-1,j:j+wind-1);
                   MSE = (sum(sum((targetblock-anchorblock).^2)))/(wind*2);
%mean square error estimation
                   if MSE < min_dist
                      Block_match(1) = i;     %finding best match block
                      Block_match(2) = j;
                      min_dist = MSE;
                   end

                   mv_x(ii,jj) =  Block_match(1); %recording the estimated
motion vector
                   mv_y(ii,jj) =  Block_match(2);
                   jj = jj + 1;
                 end
                 ii = ii + 1;

             end
             %best matching block is put in the predictive frame
             predictedframe(m:m+wind-1,n:n+wind-
1)=targetframe(Block_match(1):Block_match(1)+wind-
1,Block_match(2):Block_match(2)+wind-1);
          end
```

```
        end
        figure;
        subplot(221), imshow(anchorframe), title('Anchor Frame');
        hold on
        quiver(mv_x,mv_y);axis ij;     %plotting the motion vector
        hold off
        subplot(222), imshow(targetframe), title('Target Frame');
        subplot(223), imshow(predictedframe), title('Predicted Frame');
        subplot(224), imshow(imsubtract(anchorframe, predictedframe)),
title('Difference in Frames');

    end
```

Results:







(Motion vector representation on a frame) (Zoomed in)

2. Global motion estimation method

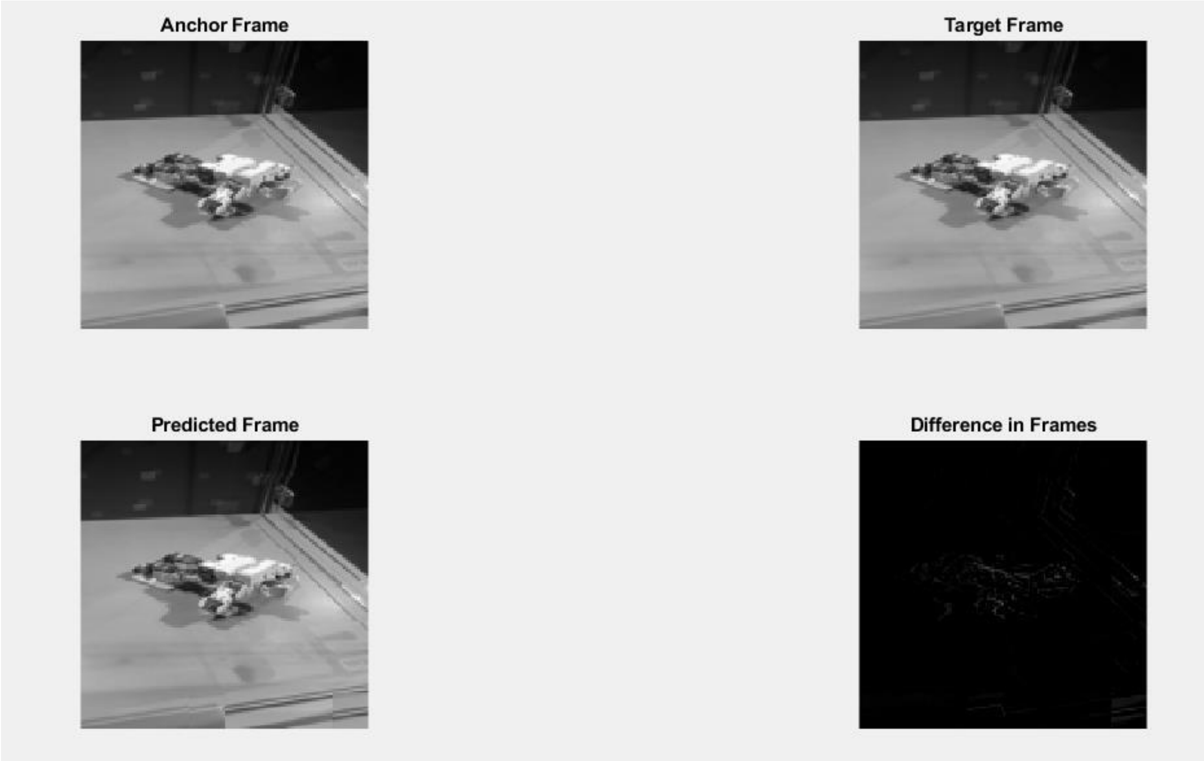
In case of global estimation, the image is taken as a whole and estimation is done using projective mapping that is affine or bilinear mapping. This case is especially suited to efficiently estimate camera motion. Indeed, camera motion such as dolly, track, boom, pan, tilt or roll, is an essential cinematic technique.

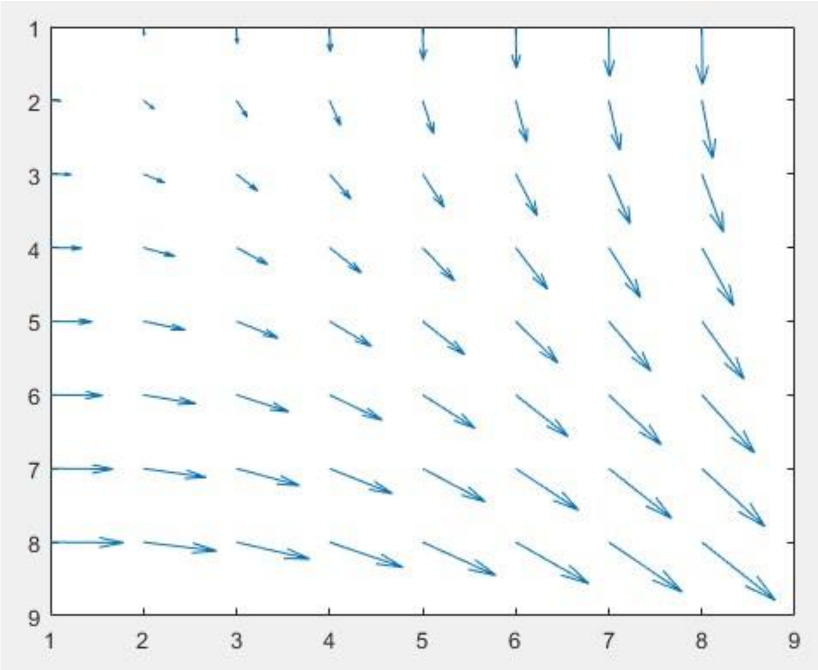
Code:

```
clear;
clc;

video = VideoReader('vidi.mp4');
num_frames = video.NumberOfFrames;
  for f=230:240    %reading 10 frames
     anchorframe = imresize(im2double(rgb2gray(read(video,f))),[128, 128]);
%anchor frame 128x128
     targetframe = imresize(im2double(rgb2gray(read(video,f+1))),[128,
128]);  %target frame 128x128

     [row,col] = size(anchorframe);
     wind = 32;  % anchor block size 32x32, estimation by less parameter;
     mv_x=[]; mv_y=[]; mm = 1;  % variable to store motion vector along x-
horizontal, y-vertical side
     for m = 1:wind:row
     nn=1;
         for n = 1:wind:col
            point(1) = m;
            point(2) = n;
            anchorblock = anchorframe(point(1):point(1)+wind-
1,point(2):point(2)+wind-1);
            min_dist = Inf;
            Block_match = 0;

            for i = 1:row-wind                  %loop for every block
               for j = 1:col-wind
                   targetblock = targetframe(i:i+wind-1,j:j+wind-1);
                   MSE = (sum(sum((targetblock-anchorblock).^2)))/(wind*2);
%mean square error estimation
                   if MSE < min_dist
                      Block_match(1) = i;       %finding best match block
                      Block_match(2) = j;
                      min_dist = MSE;
                   end
               end

            %best matching block is put in the predictive frame
            predictedframe(m:m+wind-1,n:n+wind-
1)=targetframe(Block_match(1):Block_match(1)+wind-
1,Block_match(2):Block_match(2)+wind-1);
            end
            mv_x(mm,nn) = m;       %recording the estimated motion vector
            mv_y(mm,nn) = n;
            nn = nn + 1;
        end
        mm = mm + 1;
     end
     figure;
     subplot(221), imshow(anchorframe), title('Anchor Frame');
```

```
    subplot(222), imshow(targetframe), title('Target Frame');
    subplot(223), imshow(predictedframe), title('Predicted Frame');
    subplot(224), imshow(imsubtract(anchorframe, predictedframe)),
title('Difference in Frames');
    figure; quiver(mv_x,mv_y);axis ij;     %plotting the motion vector
  end
```

Results:





(Motion vector of predicted frame using global motion estimation)

3. Block based motion estimation method

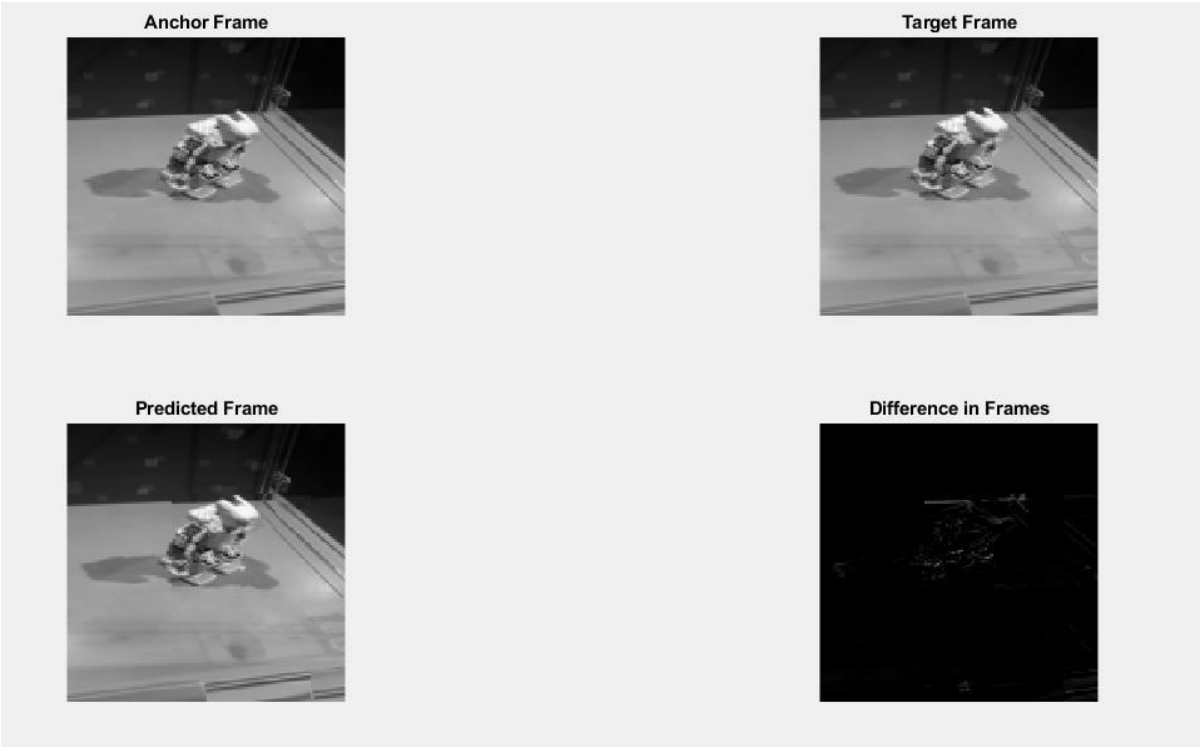
In block based coding a block of size 16x16 is taken. Using the window of this size we generate the displaced frame difference and the predicted frame. Advantage of a block based motion estimation is that it does not require additional information to represent the shape of the region.

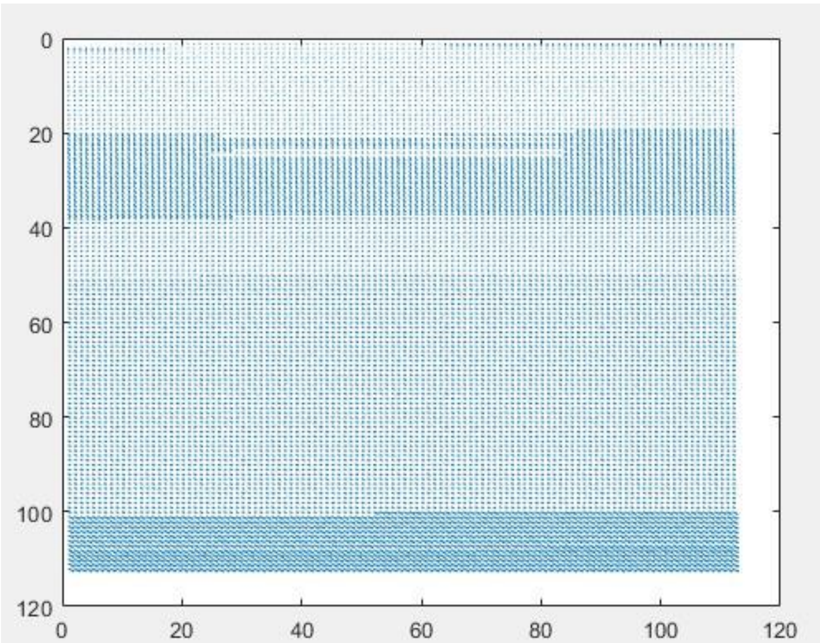
Code:

```
clear;
clc;

video= VideoReader('vidi.mp4');
num_frames = video.NumberOfFrames;

    for f=230:230     %reading 1 frame
       anchorframe = imresize(im2double(rgb2gray(read(video,f))),[128,
128]);  %anchor frame 128x128
       targetframe = imresize(im2double(rgb2gray(read(video,f+1))),[128,
128]);  %target frame 128x128

       [row,col] = size(anchorframe);
       wind = 16;     % anchor block size 16x16;
       for m = 1:wind:row
          for n = 1:wind:col
             point(1) = m;
             point(2) = n;
             anchorblock = anchorframe(point(1):point(1)+wind-
1,point(2):point(2)+wind-1);
             min_dist = Inf;
             Block_match = 0;
             mv_x=[]; mv_y=[]; ii = 1;  % variable to store motion vector
along x-horizontal, y-vertical side

             for i = 1:row-wind
                 jj=1;
                 for j = 1:col-wind
                   targetblock = targetframe(i:i+wind-1,j:j+wind-1);
                   MSE = (sum(sum((targetblock-anchorblock).^2)))/(wind*2);
%mean square error estimation
                   if MSE < min_dist
                      Block_match(1) = i;     %finding best match block
                      Block_match(2) = j;
                      min_dist = MSE;
                   end
                   mv_x(ii,jj) = Block_match(1); %recording the estimated
motion vector
                   mv_y(ii,jj) = Block_match(2);
                   jj = jj + 1;
                 end
                 ii = ii + 1;

             end
             %best matching block is put in the predictive frame
             predictedframe(m:m+wind-1,n:n+wind-
1)=targetframe(Block_match(1):Block_match(1)+wind-
1,Block_match(2):Block_match(2)+wind-1);
          end
       end
```

```
        figure;
        subplot(221), imshow(anchorframe), title('Anchor Frame');
        subplot(222), imshow(targetframe), title('Target Frame');
        subplot(223), imshow(predictedframe), title('Predicted Frame');
        subplot(224), imshow(imsubtract(anchorframe, predictedframe)),
title('Difference in Frames');
        figure; quiver(mv_x,mv_y);axis ij;     %plotting the motion vector
    end
```

Results:





(Motion vector of predicted frame using block based motion estimation)

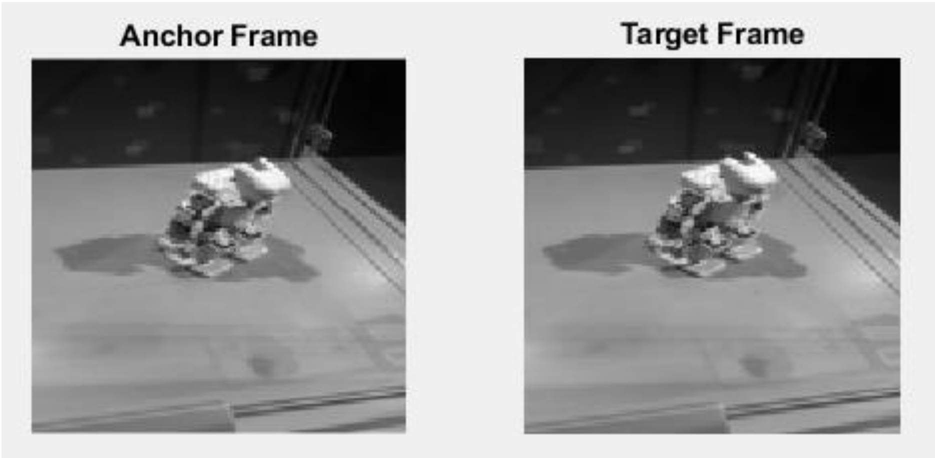
4. Regional motion estimation method

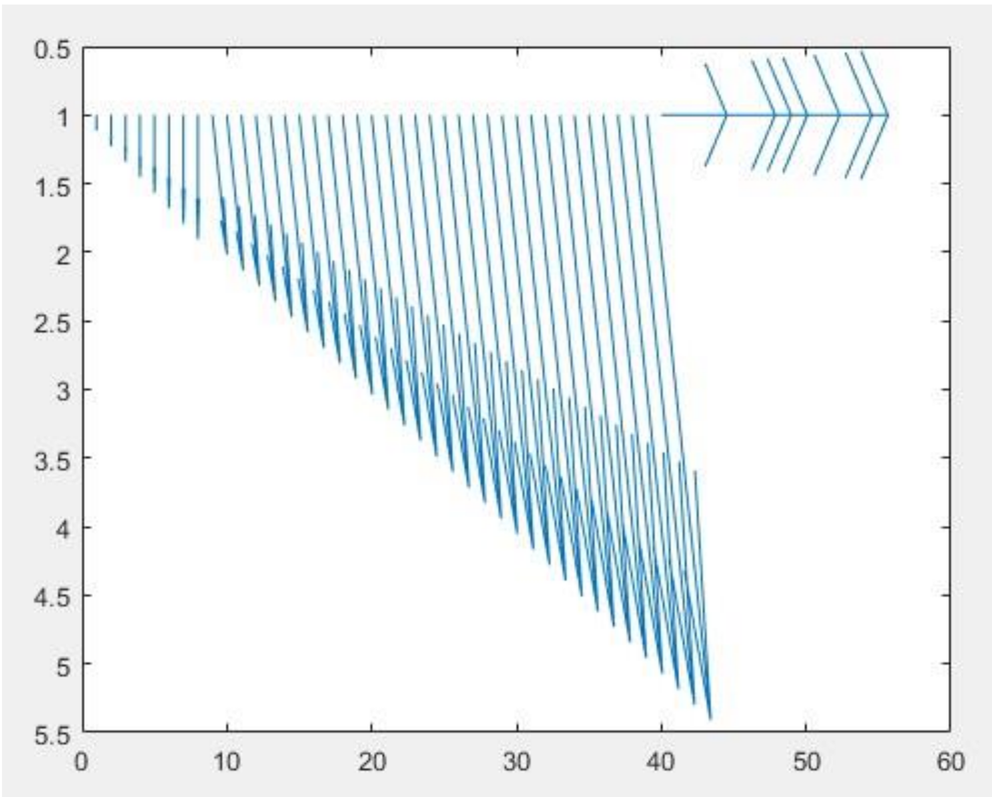
In case of regional motion estimation method, a particular region is defined by hit and trial. We then do the processing to see the motion of that particular region in the video frames.

Code:

```
video= VideoReader('vidi.mp4');
 num_frames = video.NumberOfFrames;
    for f=230:240   %reading 10 frames
       anchorframe = imresize(im2double(rgb2gray(read(video,f))),[128,
128]);  %anchor frame 128x128
       targetframe = imresize(im2double(rgb2gray(read(video,f+1))),[128,
128]);  %target frame 128x128
       [row,col] = size(anchorframe);

       figure;
       subplot(121), imshow(anchorframe), title('Anchor Frame');
       subplot(122), imshow(targetframe), title('Target Frame')
       R1 = edge(anchorframe(25:75,50:100),'Sobel'); %Region of anchor
Frame
       R2 = edge(targetframe(25:75,50:100),'Sobel'); %Region of target
Frame


       % Computing region based Displaced Frame
       DFD_region = (imabsdiff(R1,R2).^2); % p=2 for MSE
       figure;
       subplot(131); imshow(R1); title('Frame 1');
       subplot(132); imshow(R2); title('Frame 2');
       subplot(1,3,3); imshow(DFD_region); title('Predicted error');

       figure;
       mv_x=zeros(1,50);  %computing the motion vectors
       mv_y=zeros(1,50);
        for i=1:50
           for j=1:50
             if DFD_region(i,j)==1
                mv_x(j)=j;
                mv_y(i)=i;
             end
           end
        end
       quiver(mv_y,mv_x),axis ij;  %plotting motion vectors
     end
```

Results:





(Motion vector of predicted frame using regional motion estimation)

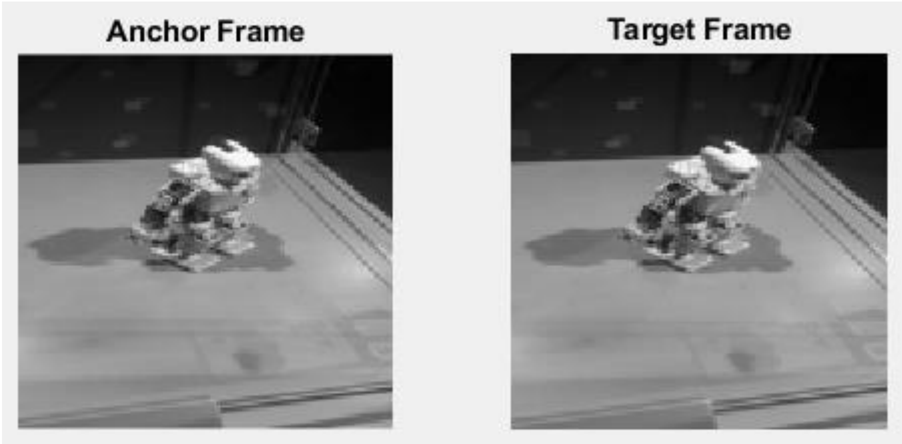
Optical Flow

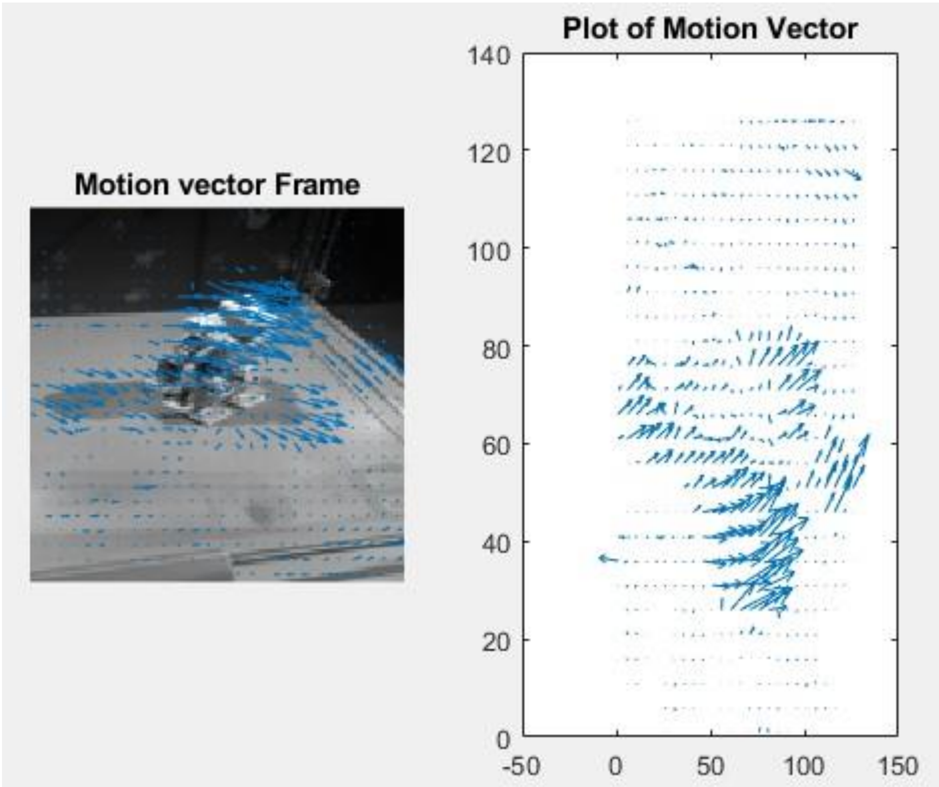
1. Pixel Based Motion Analysis

Code:

```
clear;
clc;
video = VideoReader('vidi.mp4');
% obtaining the velocity/motion vector using using Farneback method
opticFlow = opticalFlowFarneback;
     for f=230:240         %reading 10 frames
       anchorframe = imresize(im2double(rgb2gray(read(video,f))),[128,
128]);%anchor frame 128x128
      targetframe
=imresize(im2double(rgb2gray(read(video,f+1))),[128,128]);  %target frame
128x128
       %calculation motion vetor from the target frame
       flow = estimateFlow(opticFlow,targetframe);

       figure;
       subplot (121), imshow(anchorframe), title('Anchor Frame');
       subplot (122), imshow(targetframe), title('Target Frame');
       figure;
       subplot (121), imshow(targetframe), title('Motion vector Frame');
       hold on
       plot(flow,'DecimationFactor',[5 5],'ScaleFactor',10);
       hold off
       subplot (122), plot(flow,'DecimationFactor',[5 5],'ScaleFactor',10),
title('Plot of Motion Vector');
     end
```

Results:


Anchor Frame
Target Frame


Motion vector Frame
Plot of Motion Vector
140
120
100
80
60
40
20
0
-50
0
50
100
150

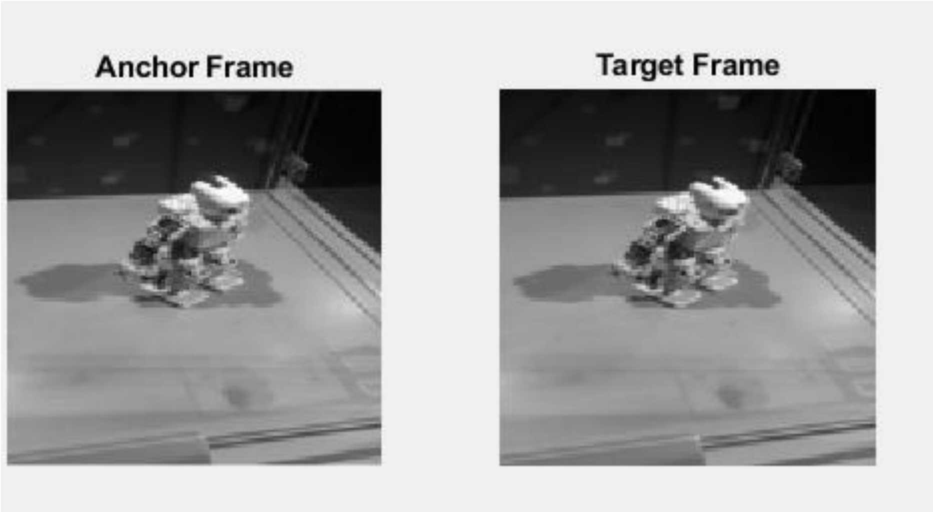
2. Global motion estimation method

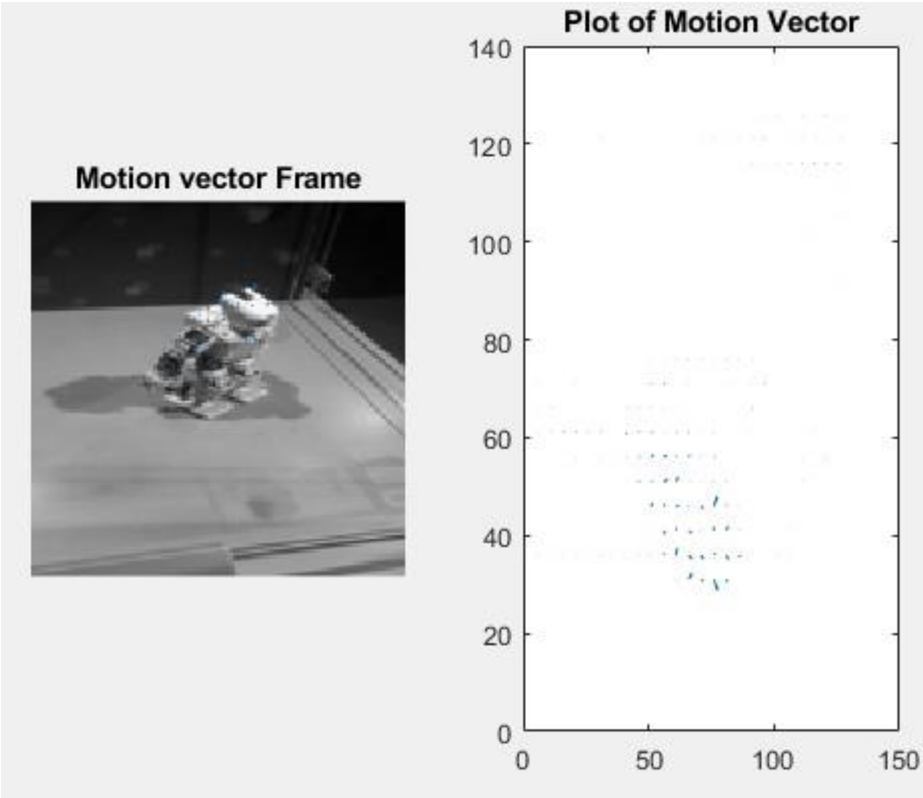
In case of global estimation, the image is taken as a whole and estimation is done using projective mapping that is affine or bilinear mapping. This case is especially suited to efficiently estimate camera motion. Indeed, camera motion such as dolly, track, boom, pan, tilt or roll, is an essential cinematic technique.

Code:

```
clear;
clc;
video = VideoReader('vidi.mp4');
% obtaining the velocity/motion vector using using Horn-Schunck method
opticFlow = opticalFlowHS;
     for f=230:240         %reading 10 frames
       anchorframe = imresize(im2double(rgb2gray(read(video,f))),[128,
128]);  %anchor frame 128x128
     targetframe = imresize(im2double(rgb2gray(read(video,f+1))),[128,
128]);  %target frame 128x128
       %calculation motion vetor from the target frame
       flow = estimateFlow(opticFlow,targetframe);

       figure;
       subplot (121), imshow(anchorframe), title('Anchor Frame');
       subplot (122), imshow(targetframe), title('Target Frame');
       figure;
       subplot (121), imshow(targetframe), title('Motion vector Frame');
       hold on
       plot(flow,'DecimationFactor',[5 5],'ScaleFactor',10);
       hold off
       subplot (122), plot(flow,'DecimationFactor',[5 5],'ScaleFactor',10),
title('Plot of Motion Vector');
     end
```

Results:


Anchor Frame
Target Frame


Motion vector Frame
Plot of Motion Vector
140
120
100
80
60
40
20
0
0
50
100
150

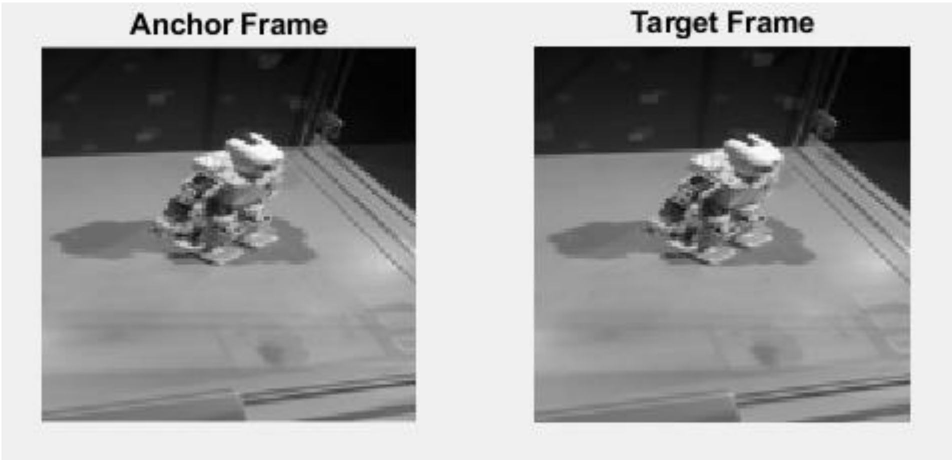
3. Block based motion estimation method

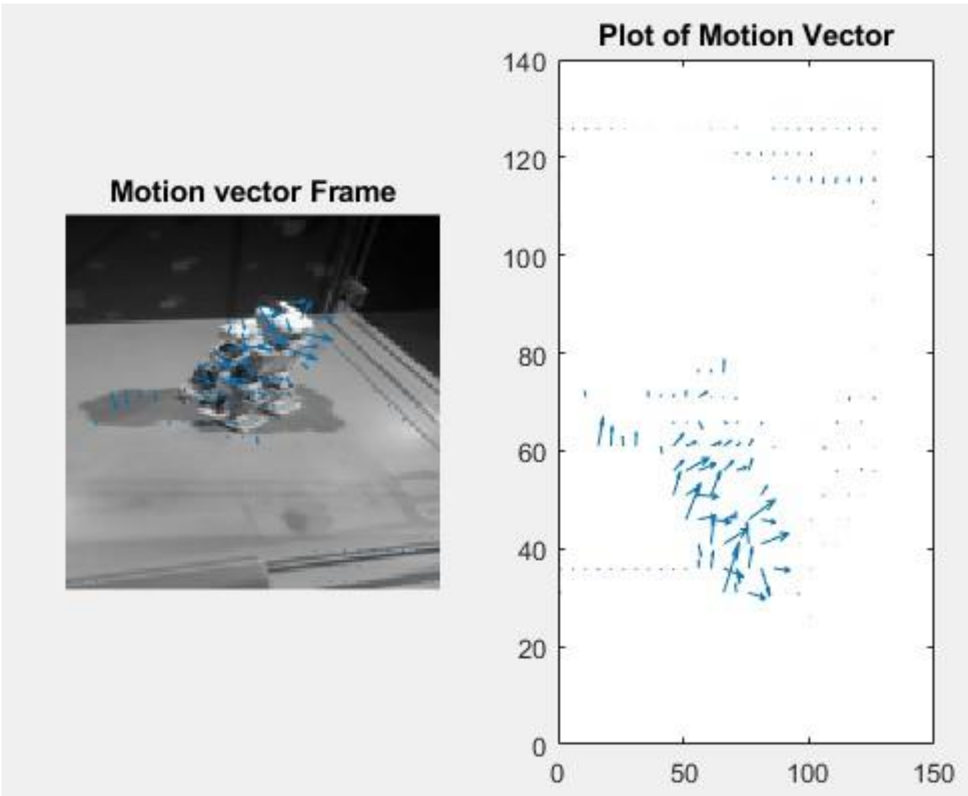
In case of optical flow equation, the velocity vector of the frames is calculated using inverse of the hessian matrix of the target frame. This velocity vector is then vector multiplied to the gradient of the target frame and added to the Mean Square Error.

Code:

```
clear;
clc;
video = VideoReader('vidi.mp4');
% obtaining the velocity/motion vector using using Lucas-Kanade method
opticFlow = opticalFlowLK('NoiseThreshold',0.0039);
     for f=230:240         %reading 10 frames
       anchorframe = imresize(im2double(rgb2gray(read(video,f))),[128,
128]);  %anchor frame 128x128
     targetframe = imresize(im2double(rgb2gray(read(video,f+1))),[128,
128]);  %target frame 128x128
       %calculation motion vetor from the target frame
       flow = estimateFlow(opticFlow,targetframe);

       figure;
       subplot (121), imshow(anchorframe), title('Anchor Frame');
       subplot (122), imshow(targetframe), title('Target Frame');
       figure;
       subplot (121), imshow(targetframe), title('Motion vector Frame');
       hold on
       plot(flow,'DecimationFactor',[5 5],'ScaleFactor',10);
       hold off
       subplot (122), plot(flow,'DecimationFactor',[5 5],'ScaleFactor',10),
title('Plot of Motion Vector');
     end
```

Results:


Anchor Frame
Target Frame


Motion vector Frame
Plot of Motion Vector
140
120
100
80
60
40
20
0
0
50
100
150

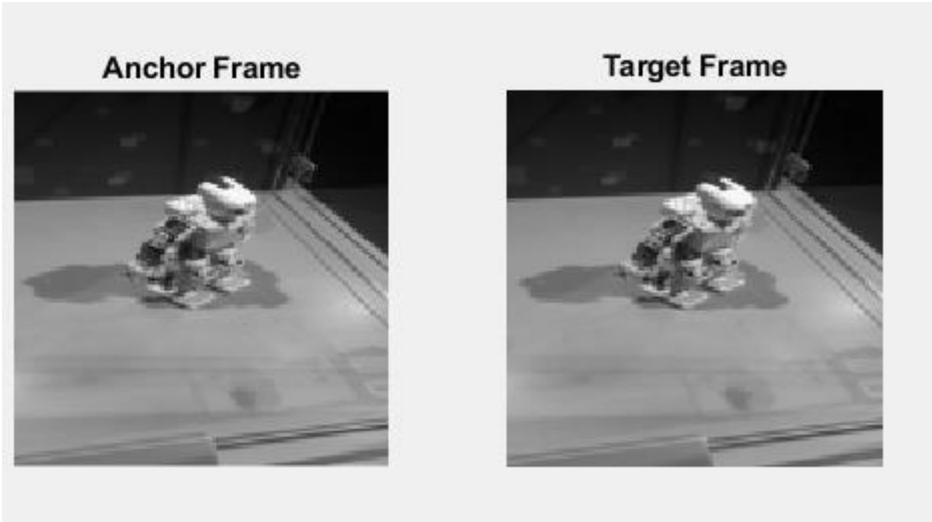
4. Regional motion estimation method

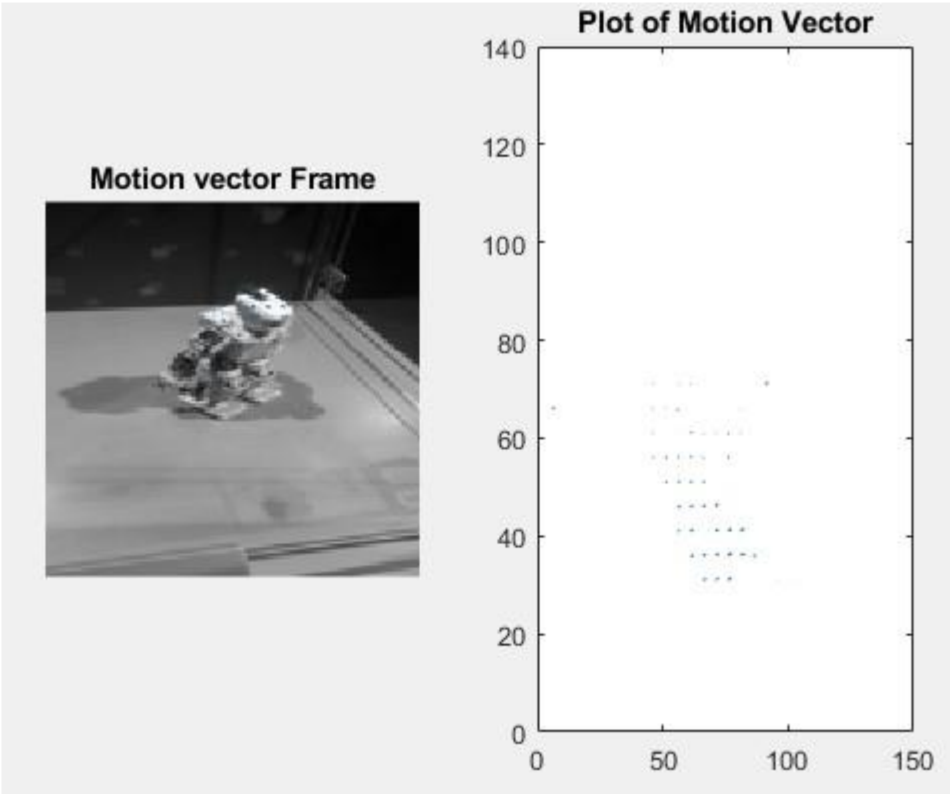
In case of regional motion estimation method, a particular region is defined by hit and trial. We then do the processing to see the motion of that particular region in the video frames.

Code:

```
clear;
clc;
video = VideoReader('vidi.mp4');
% obtaining the velocity/motion vector using using Lucas-Kanade derivative
of Gaussian method
opticFlow =  opticalFlowLKDoG;
     for f=230:240         %reading 10 frames
       anchorframe = imresize(im2double(rgb2gray(read(video,f))),[128,
128]);  %anchor frame 128x128
     targetframe = imresize(im2double(rgb2gray(read(video,f+1))),[128,
128]);  %target frame 128x128
       %calculation motion vetor from the target frame
       flow = estimateFlow(opticFlow,targetframe);

       figure;
       subplot (121), imshow(anchorframe), title('Anchor Frame');
       subplot (122), imshow(targetframe), title('Target Frame');
       figure;
       subplot (121), imshow(targetframe), title('Motion vector Frame');
       hold on
       plot(flow,'DecimationFactor',[5 5],'ScaleFactor',10);
       hold off
       subplot (122), plot(flow,'DecimationFactor',[5 5],'ScaleFactor',10),
title('Plot of Motion Vector');
     end
```

Results:


Anchor Frame
Target Frame


Motion vector Frame
Plot of Motion Vector
140
120
100
80
60
40
20
0
0
50
100
150

BIBILOGRAPHY:

1. Y. Wang, J. Ostermann, and Y.-Q. Zhang, *Video processing and communications*, Prentice Hall, 2002
2. YAO WANG's SLIDES: http://eeweb.poly.edu/~yao/EL6123/Introduction.pdf
3. MATLAB Mathworks Forum