CMSC733 Project 2
Motion Detection

Vikas Chandrakant Raykar

May 11, 2004

1 Part A

1.1 Step 0:Camera calibration

The camera was calibrated using the camera Calibration toolbox by Jean-Yves Bouguet using the calibration images show in Figure 1. The calibration results were:

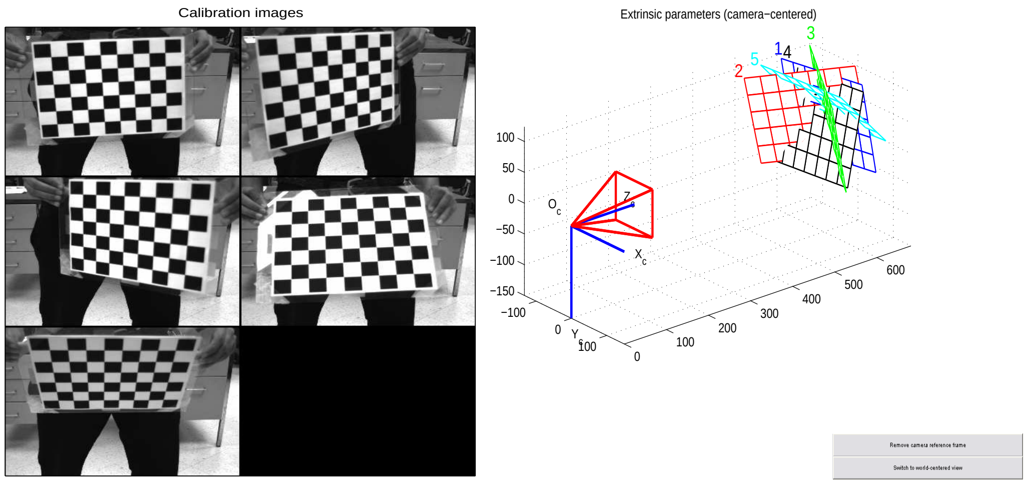
```
Calibration results (with uncertainties):

Focal Length:  fc = [ 928.06270   926.31470 ]  [ 5.81192 5.87795]
Principal point:cc =[ 327.09625   236.40833 ]  [ 9.11568 7.05710]
Skew:     alpha_c = [ 0.00000 ]  [ 0.00000  ]

Distortion:    kc = [-0.11397 0.47305   0.00163 0.00121  0.00000 ]
 [ 0.04314 0.49295 0.00240 0.00279 0.00000 ]

Pixel error:  err = [ 0.14054   0.15150 ]

Note: The numerical errors are approximately three times the
standard deviations (for reference).

The camera intrinsic matrix K

   928.062699831228                 0   327.096253964371
                  0  926.314702000249   236.408331969381
                  0                 0   1.00000000000000
```

1.2 Step 1:Stereo Matching

Figure 2 shows two images from the sequences overlaid above each other. 200 matches as found by the algorithm in Project 1 are plotted. Note that there are a lot of mismatches and RANSAC has to be used to get the inliers.

1.3 Step 2:Fundamental Matrix Computation

The fundamental matrix was computed using the robust RANSAC algorithm. The normalized 8 point algorithm was used to compute the fundamental matrix using a minimum of 8 point correspondences. For the RANSAC algorithm the approximate Sampson's error was used as the



(a) (b)

Figure 1: (a) Calibration Images (b) Extrinsic geometry

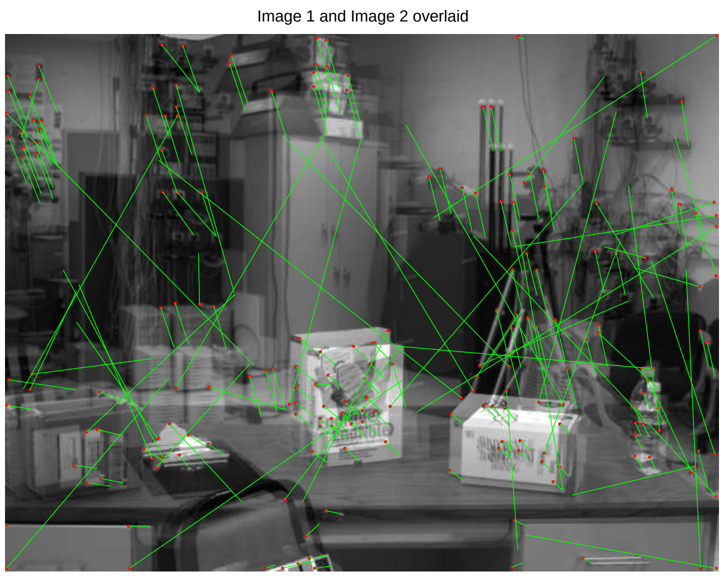


Figure 2: The two images overlaid and the corresponding 200 stereo matches as obtained by the algorithm.

distance metric. Figure 3 shows the original correspondences and the set of inliers as computed using the RANSAC algorithm. Figure 4 shows a few points and the corresponding epipolar lines.

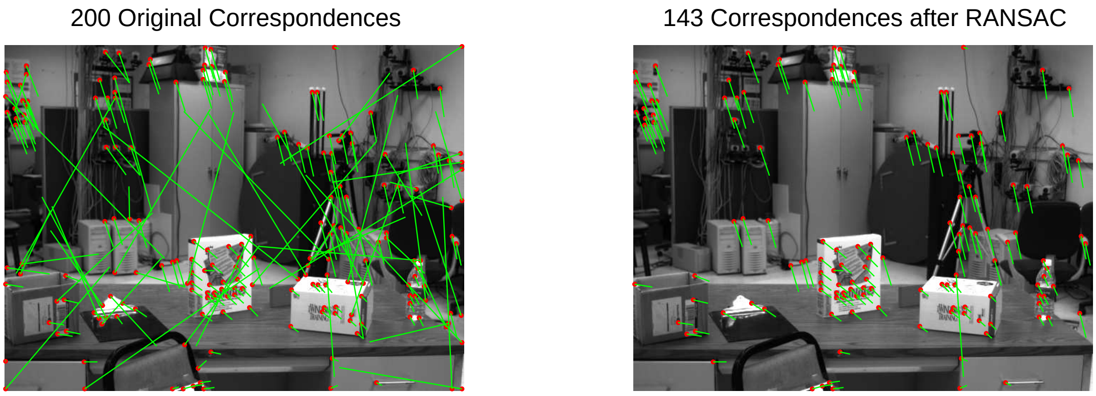


Figure 3: The original correspondences and the set of inliers as computed using the RANSAC algorithm.

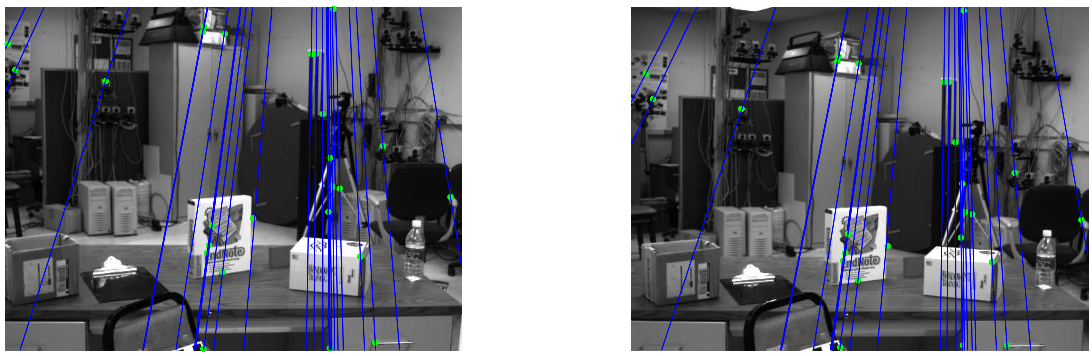


Figure 4: The epipolar lines in the two images.

The fundamental matrix as computed by RANSAC is

```
F =

   -0.0000    0.0000    0.0031
   -0.0000   -0.0000    0.0017
   -0.0029   -0.0017   -0.1321
```

1.4 Step 3:The Essential matrix

If the calibration matrix is K then the essential matrix E is given by $E = K^T F K$.

```
---------------------------------------------------
Essential Matrix E
---------------------------------------------------
   -0.0029    3.1435    3.7096
```

```
   -3.3279   -0.0087    0.4195
   -3.5384   -0.4370   -0.0587
```

1.5 Step 4: Relative Orientation

Given the essential matrix then there are two possible rotation matrices and a two translation vector(opposite direction). If $E = UDV^T$ (SVD) then thw two possible factorization of $E = SR$ are $S = UZU^T$ and $R = UWV^T$ or $R = UW^TV^T$. (The algorithm given in Hartley and Zisserman). Note that $S = [t]_x$. So there are four possible choices for the camera projection matrix. Only for one solution the reconstructed point will lie in front of both cameras.

```
---------------------------------------------------
Retrieving the camera matrices
---------------------------------------------------

R1 =

    0.9999   -0.0056    0.0083
    0.0052    0.9986    0.0519
   -0.0086   -0.0518    0.9986


R2 =

   -0.9845   -0.1355    0.1113
   -0.1259    0.1042   -0.9866
    0.1221   -0.9853   -0.1197


t1 =

    0.0928
   -0.7253
    0.6821


t2 =

   -0.0928
    0.7253
   -0.6821


P =

     1     0     0     0
     0     1     0     0
     0     0     1     0


P_dash_1 =
```

```
    0.9999   -0.0056    0.0083    0.0928
    0.0052    0.9986    0.0519   -0.7253
   -0.0086   -0.0518    0.9986    0.6821


P_dash_2 =

    0.9999   -0.0056    0.0083   -0.0928
    0.0052    0.9986    0.0519    0.7253
   -0.0086   -0.0518    0.9986   -0.6821


P_dash_3 =

   -0.9845   -0.1355    0.1113    0.0928
   -0.1259    0.1042   -0.9866   -0.7253
    0.1221   -0.9853   -0.1197    0.6821


P_dash_4 =

   -0.9845   -0.1355    0.1113   -0.0928
   -0.1259    0.1042   -0.9866    0.7253
    0.1221   -0.9853   -0.1197   -0.6821
```

2 Part B

In this part we have to detect independent motion in a given stereo video. we first calibrate the stereo video. After calibration we ca do a stereo rectification. From this we get a disparity map. For rectified stereo depth is inversely proportional to the disparity. From the video we can compute the the optical flow. Given the optical flow and depth we can estimate the camera translation and rotation. Independently moving points give a larger error.

2.1 Camera calibration and Rectification

The cameras were calibrated using the calibration toolbox. Given a pair of stereo images, rectification determines a transformation of each image plane such that pairs of conjugate epipolar lines become collinear and parallel to one of the image axes. The important advantage of rectification is that computing stereo correspondences is reduced to a 1-D search problem along the horizontal or vertical raster lines of the rectified images. Figure 6(a) shows the original stereo images overlaid above each other. Figure 6(b) shows the rectified images. rectification was done using the routine provided in the calibration toolbox. It can be seen that for the rectified images the disparity is only in the y direction.

2.2 Disparity Map

Once the images are rectified we can obtain a dense stereo correspondence by searching for a corresponding point along the vertical line. I implemented a simple code which uses normalized cross correlation to search for corresponding point. Figure **??** shows the disparity map for the images shown before. Once the disparity is known, since the images are rectified the depth is

Figure 5: The original and the rectified images

inversely proportional to the disparity. More specifically $Z = fb/d$ where Z is the depth, f is the focal length, b is the baseline and d is the disparity.

2.3 Optical Flow

Using images from one camera we can compute the optical flow for each point. I implemented the iterative algorithm given in the booth Robot vision [1]. Figure 7 shows the optical flow obtained. Figure 8 shows the optical flow blow up around Abhijeets head.

2.4 Camera Motion Estimation

So upto now at each point in the image we have both the optical flow and the depth information. Using the equation relating the depth, optical flow and the image position and having a minimum of three points we can solve for the the 6 camera parameters(tx, ty, tz, α, β and γ). I used the RANSAC to estimate the motion parameters. In this way even if there are objects moving independently because we use RANSAC we estimate the right motion parameters. Once we have estimated the motion parameters of the camera we ca build an error map for each pixel in the image. Points belonging to independently moving objects will have a very high error. Figure 9(a) shows the error in estimating the motion parameters. once we have the error map we can threshold it to detect independently moving objects. Figure 9(b) shows thresholded Abhijeet. I ran out of memory when running the code in MATLAB. So I worked with down sampled images. Figure 10, Figure 11 Figure 12 shows the results for three different frames.

[1]Chap 12 Motion Field and Optical Flow in Robot Vision by B.K.P.Horn

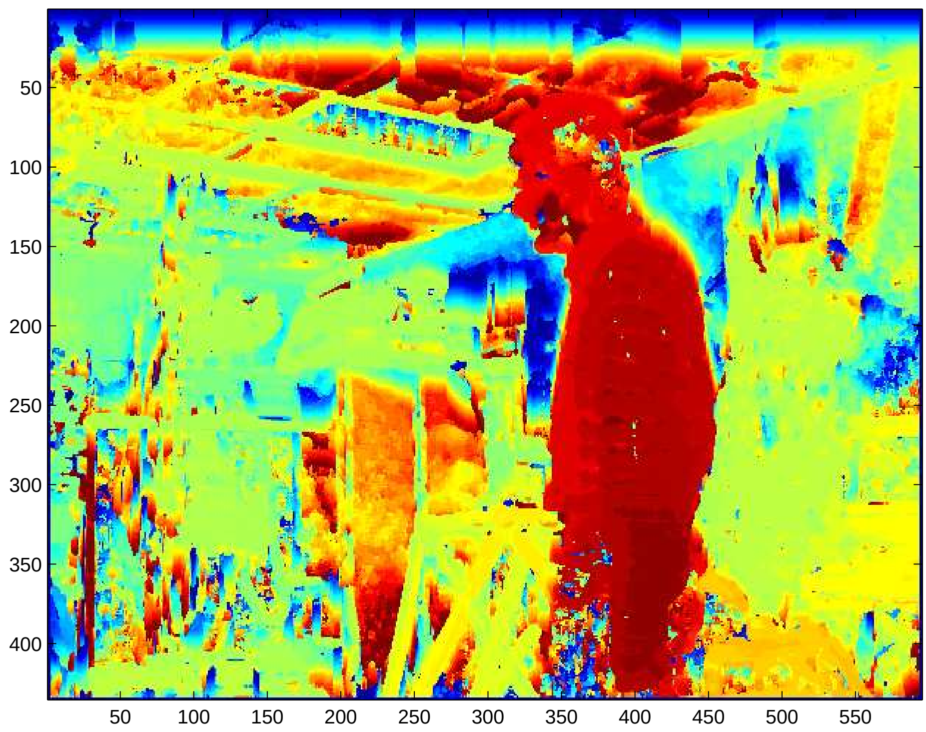


Figure 6: The disparity map for the right and the left images.

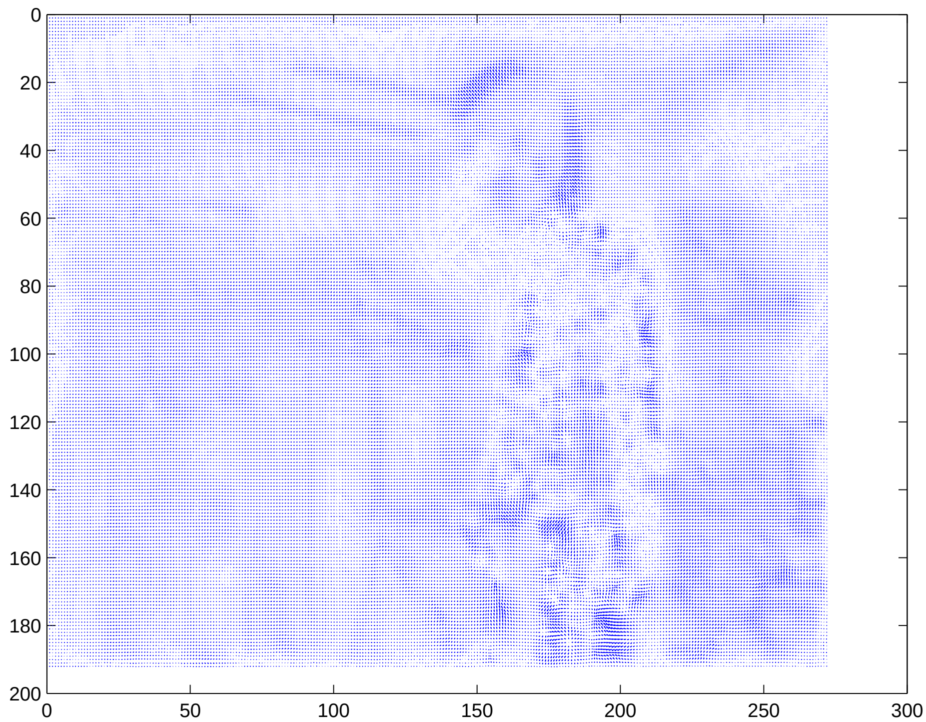


Figure 7: The optical flow for a frame in the first camera.

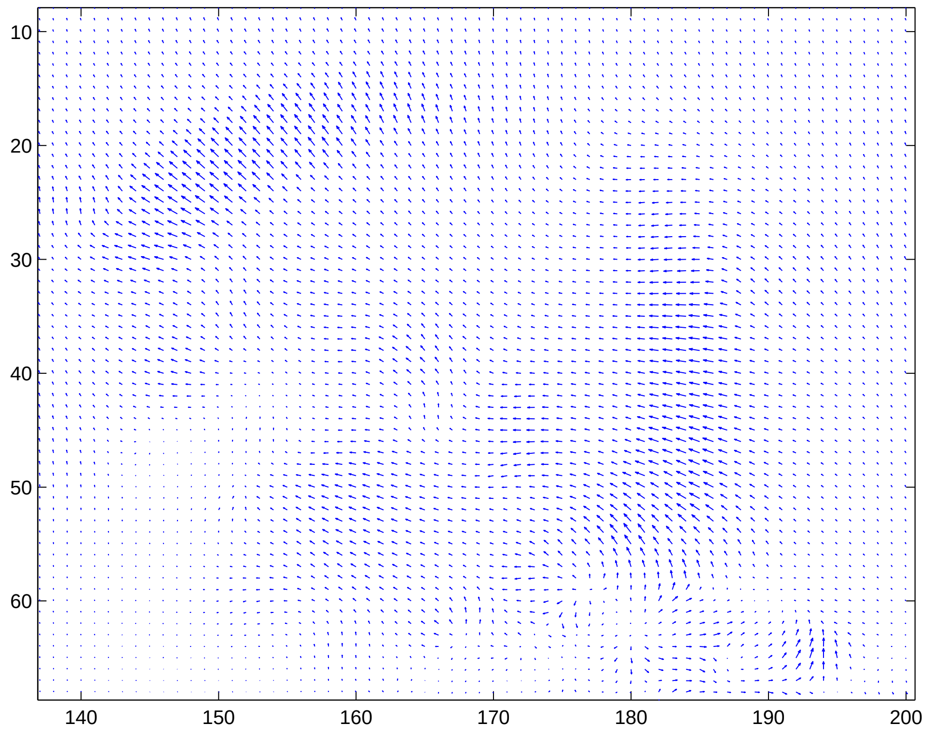


Figure 8: Blow up of the optical flow around Abhijeets head.

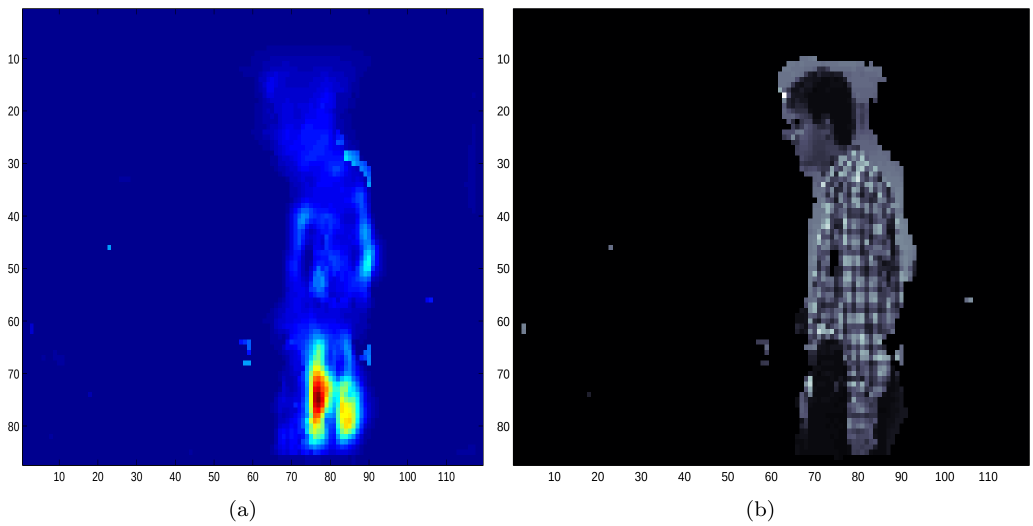


Figure 9: (a) Error in estimating the motion parameters. (b) Independently moving objects detected using thresholded error.

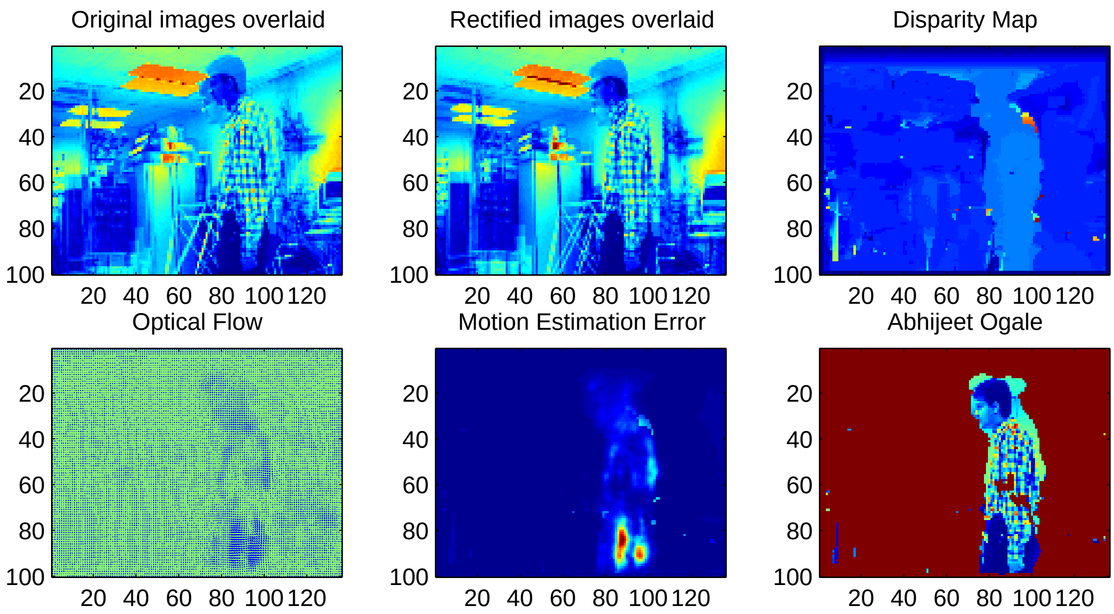


Figure 10: Result for frame 2.

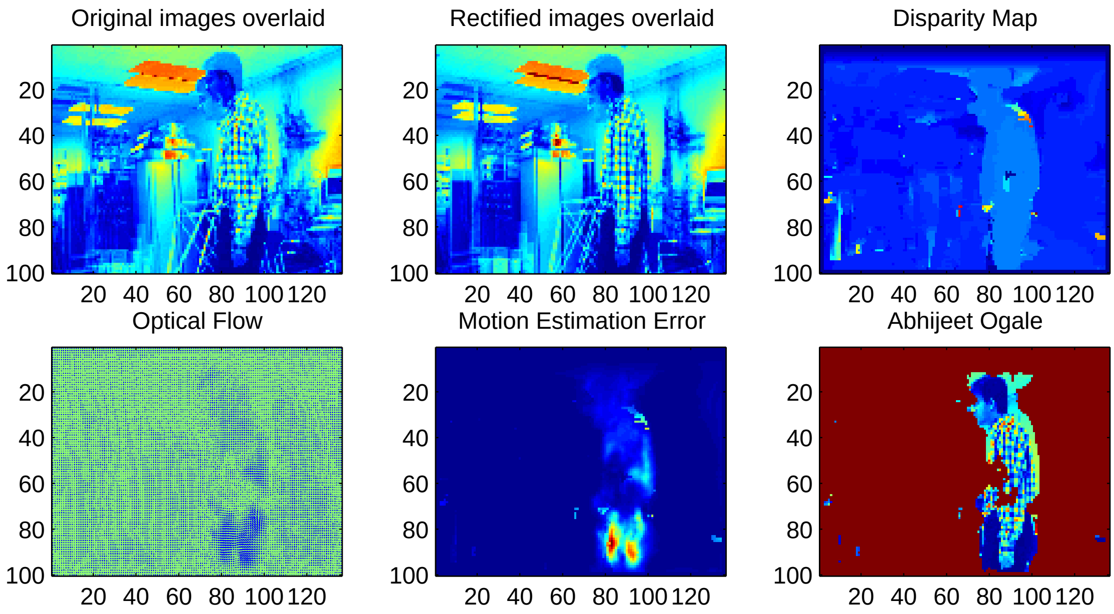


Figure 11: Result for frame 3.

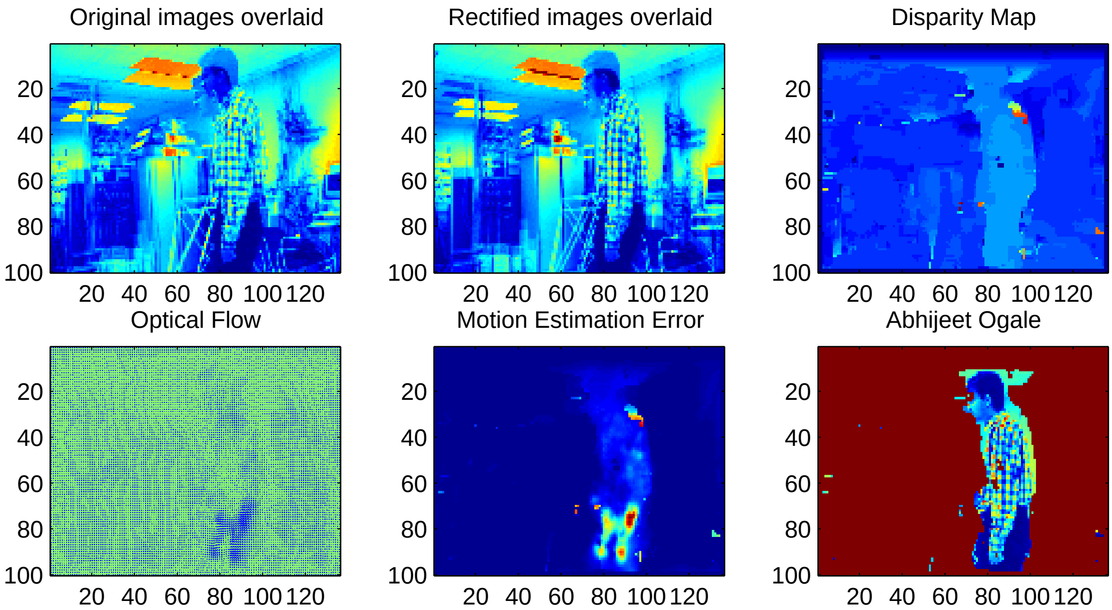


Figure 12: Result for frame 4.