Chapter 5

Multiview Image Registration

5.1 Introduction

Human brain mosaics the split images of a large object that are automatically captured through eyes. Each eye functions as a camera lens. But, it is impossible to cover very large area with the help of an eye than a pair of eyes. Keeping this in mind, one can infer that two eyes capture the two split images of a large object which are later fused into a single complete large image. Similarly, even in the real world the concept of multiview registration is essential because it may not be possible to capture a large document with a given camera in a single exposure. The field of view (FOV) of the commercial camera is much smaller than that of humans. Here the aim is to attain high field of view without compromising the image quality. It is the process of generating a single, large, integrated image by combining the visual clues from multiple images. One of the reasons that multiview image registration is an extremely challenging problem is the large degree of variability of the input data. The images that are to be registered may contain visual information belonging to very different domains and can undergo many geometric and photometric distortions such as scaling, rotations, projective transformations, non rigid perturbations of the scene structure, temporal variations, and photometric changes due to different acquisition modalities and lighting conditions[8].

The problem of multiview image registration is mainly a combination of the process of estimating a transformation that maps the points from one image to the other and the process of combining the registered images into a composite image, in such a way that no seams are visible in mosaic, at the image boundaries. Image when joined leaves the some sort of distorted region behind; these regions must be dealt with. Some of the region obtained the pure black part, this part indicate either the part of the region which is not there in the image or the part that is distorted due to applied algorithms. All these things are eradicated by image blending. This feature is the most
important in the implementation of registration for multi view images. The other important feature is to detect the overlapped region in the series of given images. The detection is done through the use of counters and synthesis is performed on the extracted overlapped region. The series of images may or may not be in the same alignment as per desired output image, thus mainly scaling, rotation and image transformation must be done to the series of images to get the proper aligned image.

5.2 Multiview Image Registration Using Euclidean Warping and Image Blending

5.2.1 Image Warping

Image warping is a transformation which maps all positions in one image plane to positions in a second plane. Image warping is in essence a transformation that changes the spatial configuration of an image. Warping is a pair of two-dimensional functions, \( u(x, y) \) and \( v(x, y) \), which map a position \( (x, y) \) in one image, where \( x \) denotes column number and \( y \) denotes row number, to position \( (u, v) \) in another image, see figure 1[46].

![Figure 5.1 Notation used for image warping](attachment:figure51.png)
Euclidean Warp

The Euclidean warp is also called Euclidean similarity transform involving four parameters

\[ P = [s \, \theta \, t_x \, t_y] \]

which denotes scale, rotation and translation respectively.

Let us consider an image function \( f \) defined over a \((x',y')\) coordinate system, undergoes geometric distortion to produce an image \( g \) defined over an \((x,y)\) coordinate system. This transformation may be expressed as

\[
(x',y') = T \{ (x,y) \}
\]  

(5.1)

Where \( T \) is \((x,y)\) rotated by \( \theta \), scaled by \( s \) and translated by \( t_x \) in \( x \) direction and \( t_y \) in \( y \) direction.

\[
\begin{bmatrix}
  x' \\
  y' \\
  1
\end{bmatrix} =
\begin{bmatrix}
  s \cos \theta & -s \sin \theta & t_x \\
  s \sin \theta & s \cos \theta & t_y \\
  0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
  x \\
  y \\
  1
\end{bmatrix}
\]  

(5.2)

Let \( p = (x,y, 1) \) denote a position in the original image, \( I \) and \( p' = (x',y',1) \) denote the corresponding position in the warped image \( I' \). So transformation for one pixel may be expressed as

\[
p' = T * p
\]

(5.3)

where \( T \) is transformation matrix.

Similarly for image of \( n \) pixels

\[
P = \begin{bmatrix}
  x_1 & x_2 & \ldots & x_n \\
  y_1 & y_2 & \ldots & y_n \\
  1 & 1 & \ldots & 1
\end{bmatrix}
\]

(5.4)

So transformation may be expressed as

\[
P' = T * P
\]

(5.5)

There are two common methods for image warping for re sampling.
1. Forward Transform
2. Backward Transform

In forward warping the input image is scanned line by line and the pixels are transformed to the output image. Their positions are given by the result of linear transformation; however this technique is troublesome since it results in images with holes due to non overlapping region of the mapping. It can happen for some transformation that different points from the input are mapped to the same point in the output image, however all of them can have different values, therefore for this method, we need to store these values in accumulator for further interpolation, because of mentioned problem with forward mapping, back ward mapping is used.

In backward mapping, output image is scanned pixel by pixel and the corresponding pixel position in the input image is computed, the points from the output space are mapped to the input space, and then based on the nearest neighbors of a mapped point, its value is determined by pixel value interpolation. Even if the two points are mapped to the same input position, it does not pose a problem.

So backward mapping from output image to input image is described as

$$ P = T^{-1} * P' $$

(5.6)

Here, the transformation between images is not known. Approximate estimation of the transformation required between two images to be registered. The estimated transformation is done by providing the points of correspondence in each of the images.

### 5.2.2 Resampling Using Bilinear Interpolation

In bilinear interpolation, the intensity at a point is determined from the weighted sum of intensities at four pixels closest to it. Therefore, given location \((X,Y)\) and assuming \(u\) is the integer part of \(X\) and \(v\) is the integer part of \(Y\), the intensity at \((X,Y)\) is estimated from the intensities at \((u, v)\), \((u +1,v)\), \((u, v +1)\), \((u +1,v +1)\). This resampling involves first finding the intensity at \((X, v)\) from the linear interpolation of intensities at \((u, v)\) and \((u+1,v)\). Let this intensity be \(I(X, v)\). Then,
the intensity at \((X, v + 1)\) is determined from the linear interpolation of intensities at \((u, v + 1)\) and 
\((u+1,v + 1)\). Let this intensity be \(I(X, v + 1)\). Then, the intensity at \((X, Y)\) is computed from the
linear interpolation of intensities at \((X, v)\) and \((X, v + 1)\) as shown in figure 5.2. This can be
summarized as

\[
I(X,Y) = Wu,v I(u,v) + Wu+1,v I(u+1,v) + Wu,v+1 I(u,v+1) + Wu+1,v+1 I(u+1,v+1) 
\]  

(5.7)

Where

\[
W_{u,v} = (u+1-x)(v+1-y)
\]

\[
W_{u+1,v} = (x-u)(v+1-y)
\]

\[
W_{u,v+1} = (u+1-x)(y-v)
\]

\[
W_{u+1,v+1} = (x-u)(y-u)
\]

and \(I(u, v)\), \(I(u+1,v)\), \(I(u, v+1)\) and \(I(u+1,v+1)\) are intensities at \((u, v)\), \((u+1,v)\), \((u, v + 1)\), and
\((u + 1,v + 1)\), respectively.

**Figure 5.2** Estimating the intensity at \((x, y)\)
Algorithm 5.1 Multiview Image Registration Using Euclidean Warping

**Inputs:** Multiview Images(Two or more )

**Output:** Registered Image

1. Load two input images (color/grayscale).

2. Define matching points in both images for correspondance. While defining the points of correspondence, two corresponding points are selected in exactly the same order in both images. Failure to comply with this assumption will result in unpredictable warping.

3. Estimate transformation parameters (scaling,rotation,translation in x direction(tx),translation in y direction(ty)) using Euclidean Warping and construct the transformation matrix using eqn. (5.2)

4. Warp incoming corners to determine the size of the output image (in to out). Using Transformation matrix we can determine the warped (output) image size using corner detection of the second image.

5. Do backwards transform (from out to in). Backward warping is also performed for the proper alignment of input and output pixels.

6. Re-sample pixel values with bilinear interpolation as explained in 5.2.2 for warped image.

7. Find the Offset from the control points and according to offset copy original image into the warped image.

8. Show the result.

9. End
5.3 Multi View Image Registration Based on Affine Warp, Levenberg-Marquardt (LM) Optimization.

5.3.1 Affine Warp

The Affine warp is also called Affine transform involving four parameters \( P = [s \theta \ tx \ ty \ s_1 \ s_2] \), which denotes scale, rotation and translation (in x and y direction) and shearing (in x and y direction) respectively.

In Euclidean transformation, as described in section 5.2, lines transform to lines, planes to planes, circles to circles. All that changes are the position and orientation, hence it preserve length and orientation. With affine transformations, lines transform to lines but circle become ellipses, affine transformation is combined effect of translation, rotation, scaling and shear.

Let us consider an image function \( f \) defined over a \((x', y', w')\) coordinate system, undergoes geometric distortion to produced an image \( g \) defined over an \((x, y, w)\) coordinate system. This transformation may be expressed as

\[
(x', y', w') = T \{(x, y, w)\} \quad (5.8)
\]

Using homogeneous coordinates, we can describe the transformation matrix

\[
\begin{bmatrix}
  x' \\
  y' \\
  w'
\end{bmatrix} =
\begin{bmatrix}
  m_0 & m_1 & m_2 \\
  m_3 & m_4 & m_5 \\
  m_6 & m_7 & m_8
\end{bmatrix}
\begin{bmatrix}
  x \\
  y \\
  w
\end{bmatrix} \quad (5.9)
\]

Where

\[
x' = \frac{m_0 x + m_1 y + m_2}{m_6 x + m_7 y + 1} \quad y' = \frac{m_3 x + m_4 y + m_5}{m_6 x + m_7 y + 1} \quad D = m_6 x + m_7 y + 1 \quad (5.10)
\]

For affine transformation an image function \( f \) defined over a \((x', y')\) coordinate system undergoes geometric distortion to produced an image \( g \) defined over an \((x, y)\) coordinate system. This transformation may be expressed as
\[(x',y') = T(x,y)\]  

(5.11)

Where T is \((x,y)\) rotated by \(\theta\), scaled by \(s\) and translated by \(t_x\) in \(x\) direction, \(t_y\) in \(y\) direction, sheared by \(s_1\) in \(x\) direction and \(s_2\) in \(y\) direction.

\[
\begin{bmatrix}
  x' \\
  y' \\
  1
\end{bmatrix} =
\begin{bmatrix}
  s \cos \theta & -s \sin \theta & t_x \\
  s \sin \theta & s \cos \theta & t_y \\
  0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
  x \\
  y \\
  1
\end{bmatrix}
\]

(5.12)

So in generalized form

\[
\begin{bmatrix}
  x' \\
  y' \\
  1
\end{bmatrix} =
\begin{bmatrix}
  m_0 & m_1 & m_2 \\
  m_3 & m_4 & m_5 \\
  0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
  x \\
  y \\
  1
\end{bmatrix}
\]

(5.13)

Suppose \(p = (x, y, 1)\) denote a position in the original image \(I\) and \(p' = (x', y', 1)\) denote the corresponding position in the warped image \(I'\). So transformation for one pixel may be expressed as shown in equation (5.3)

### 5.3.2 Levenberg-Marquardt (LM) Optimization

Direct search methods like NM simplex method (section 3.4) require many function evaluations to converge to the minimum point. Gradient based methods exploit derivative information of the function and are usually faster search methods [53]. Different Gradient based methods are available in which Cauchy ‘s method works well when the initial point is far away from the minimum point and Newton’s method works well when the initial point is near the minimum point. In any given problem it is usually not known whether the chosen initial point is away from the minimum or close to the minimum, but wherever be the minimum point, a method can be devised to take advantage of both these methods. In Marquardt method, Cauchy’s method is initially followed, there after Newton’s method is adopted. The transition from Cauchy’s method to Newton’s method is adaptive and depends on the history of the obtained intermediate solutions [53, 54]. The problem for which the LM algorithm provides a solution is called Nonlinear Least
Squares Minimization. In the algorithm of registration, this optimization method is used to minimize the intensity difference in overlapped region of the images to be registered and according to that transformation matrix of equation 5.12 is updated.

Suppose \( I(x,y) \) and \( I'(x',y') \) are images to be registered, for all corresponding pair of pixels (overlapping region) in \( I(x,y) \) and \( I'(x',y') \), the sum of the squared intensity error is given by

\[
E = \sum [I'(x', y') - I(x, y)]^2 = \sum e^2
\]

(5.14)

To perform minimization, LM algorithm requires computation of the partial derivatives of \( e_i \) with respect to the unknown motion parameters \( \{m_0 \ldots m_7\} \), also known as Jacobian matrix \( J \), which is matrix of all first-order partial derivatives of a vector with respect to another vector.

\[
J = \begin{bmatrix}
\frac{\partial e_1}{\partial m_0} & \frac{\partial e_1}{\partial m_1} & \frac{\partial e_1}{\partial m_2} & \frac{\partial e_1}{\partial m_3} & \frac{\partial e_1}{\partial m_4} & \frac{\partial e_1}{\partial m_5} & \frac{\partial e_1}{\partial m_6} & \frac{\partial e_1}{\partial m_7} \\
\frac{\partial e_2}{\partial m_0} & \frac{\partial e_2}{\partial m_1} & \frac{\partial e_2}{\partial m_2} & \frac{\partial e_2}{\partial m_3} & \frac{\partial e_2}{\partial m_4} & \frac{\partial e_2}{\partial m_5} & \frac{\partial e_2}{\partial m_6} & \frac{\partial e_2}{\partial m_7} \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
\frac{\partial e_n}{\partial m_0} & \frac{\partial e_n}{\partial m_1} & \frac{\partial e_n}{\partial m_2} & \frac{\partial e_n}{\partial m_3} & \frac{\partial e_n}{\partial m_4} & \frac{\partial e_n}{\partial m_5} & \frac{\partial e_n}{\partial m_6} & \frac{\partial e_n}{\partial m_7} \\
\frac{\partial e_1}{\partial m_0} & \frac{\partial e_1}{\partial m_1} & \frac{\partial e_1}{\partial m_2} & \frac{\partial e_1}{\partial m_3} & \frac{\partial e_1}{\partial m_4} & \frac{\partial e_1}{\partial m_5} & \frac{\partial e_1}{\partial m_6} & \frac{\partial e_1}{\partial m_7}
\end{bmatrix}
\]

(5.15)

Where \( e_n \) is the intensity error (equation 5.14) at corresponding pairs of pixels \( n \) inside both images \( I(x,y) \) and \( I'(x',y') \).

So partial derivative of \( e_n \) with respect to \( m_k \) is given by

\[
\frac{\partial e}{\partial m_k} = \frac{\partial I'}{\partial x'} \frac{\partial e}{\partial m_k} + \frac{\partial I'}{\partial y'} \frac{\partial e}{\partial m_k} 
\]

(5.16)

With respect to motion parameter \( m_0 \) to \( m_7 \), finding partial derivative of \( e \)

\[
\frac{\partial e}{\partial m_0} = \frac{x}{D} \frac{\partial I'}{\partial x'} 
\]

(5.17)

\[
\frac{\partial e}{\partial m_1} = \frac{y}{D} \frac{\partial I'}{\partial x'} 
\]

(5.18)
\[
\frac{\partial e}{\partial m_2} = \frac{1}{D} \frac{\partial l'}{\partial x'} \\
\frac{\partial e}{\partial m_3} = \frac{x}{D} \frac{\partial l'}{\partial y'} \\
\frac{\partial e}{\partial m_4} = \frac{y}{D} \frac{\partial l'}{\partial y'} \\
\frac{\partial e}{\partial m_5} = \frac{1}{D} \frac{\partial l'}{\partial y'} \\
\frac{\partial e}{\partial m_6} = -\frac{x}{D} \left( x' \frac{\partial l'}{\partial x'} + y' \frac{\partial l'}{\partial y'} \right) \\
\frac{\partial e}{\partial m_7} = -\frac{y}{D} \left( x' \frac{\partial l'}{\partial x'} + y' \frac{\partial l'}{\partial y'} \right)
\]

Where D is the denominator in equation 5.10 and \( \left( \frac{\partial l'}{\partial x'}, \frac{\partial l'}{\partial y'} \right) \) is the image intensity gradient of \( I' \)

From these partial derivatives, the Levenberg-Marquardt algorithm computes an approximate Hessian matrix \( A \) and the weighted gradient vector \( B \). Hessian matrix is derivative of jacobian matrix (equation 5.15). Hessian matrix \( A \) is given by

\[
A_{kl} = \sum_n \frac{\partial e}{\partial m_k} \frac{\partial e}{\partial m_l}
\]

So

\[
A = \begin{bmatrix}
\sum_n \frac{\partial e_n^2}{\partial m_0 \partial m_1} & \sum_n \frac{\partial e_n^2}{\partial m_0 \partial m_2} & \cdots & \sum_n \frac{\partial e_n^2}{\partial m_0 \partial m_7} \\
\sum_n \frac{\partial e_n^2}{\partial m_1 \partial m_0} & \sum_n \frac{\partial e_n^2}{\partial m_1 \partial m_2} & \cdots & \sum_n \frac{\partial e_n^2}{\partial m_1 \partial m_7} \\
\vdots & \vdots & \ddots & \vdots \\
\sum_n \frac{\partial e_n^2}{\partial m_7 \partial m_0} & \sum_n \frac{\partial e_n^2}{\partial m_7 \partial m_1} & \cdots & \sum_n \frac{\partial e_n^2}{\partial m_7 \partial m_7}
\end{bmatrix}
\]

\[(5.26)\]
Chapter 5 Multiview Image Registration

And gradient vector $B$ is given by

$$B_k = -\sum e_n \frac{\partial e}{\partial m_k} \quad (5.27)$$

So

$$B = \begin{bmatrix}
    \sum e_n \frac{\partial e_0}{\partial m_0} \\
    \sum e_n \frac{\partial e_1}{\partial m_1} \\
    \sum e_n \frac{\partial e_2}{\partial m_2} \\
    \sum e_n \frac{\partial e_3}{\partial m_3} \\
    \sum e_n \frac{\partial e_4}{\partial m_4} \\
    \sum e_n \frac{\partial e_5}{\partial m_5} \\
    \sum e_n \frac{\partial e_6}{\partial m_6} \\
    \sum e_n \frac{\partial e_7}{\partial m_7}
\end{bmatrix} \quad (5.28)$$

and then updates the motion parameter estimate $m$ by an amount

$$\Delta m = (A + \lambda I_d)^{-1} B \quad (5.29)$$

Where $\lambda$ is a time-varying stabilization parameter.

Now solve the system by

$$(A + \lambda I_d) \Delta m = B \quad (5.30)$$

The motion parameters estimation is given by

$$m(t+1) = m(t) + \Delta m \quad (5.31)$$

Error in equation (5.14) is checked, if it is incremented, then compute new $\Delta m$ by changing the value of $\lambda$ and continue iterating until the error is below a threshold or a fixed number of steps
has been completed. The advantage of using Levenberg-Marquardt over straightforward gradient descent is that it converges in less iteration [55].

Figure 5.3  Levenberg-Marquadt Algorithm for multiview image registration

5.3.3 Image Blending Using Distance Weight Factor

The purpose of blending is to provide a smooth transition between images and eliminate small residues of misalignments resulting from parallax or imperfect registrations. In the overlapped area the image blending algorithm calculates the contribution of the images to be registered at every pixel. The advantage of using blending method is to minimize the effects of intensity variations, improving visual quality of the composite image and making the edges invisible. In our
implementation, weight based blending is used, in which least distance from the pixel to the boundary of each image is used as shown in figure 5.3

The blended image consists of pixels

\[ N(x, y) = \alpha I(x, y) + (1 - \alpha) I'(x', y') \]  \hspace{1cm} (5.32)

where, \( I(x, y) \) and \( I'(x', y') \) are image pixel in overlapping region and \( N(x, y) \) is the new composite image pixel, \( \alpha \) is the weight factor which is calculated as a distance from image edge.

\[ \alpha = \frac{d_2}{d_1 + d_2} \]  \hspace{1cm} (5.33)

Thus blending function refines the overlapping region by using the least distance from the pixel to the boundary of each image.

**Figure 5.4** Distance based image blending
**Algorithm 5.2**  Multiview Image Registration Using Affine Warp and LM Optimization

**Inputs:** Multiview Images(Two or more )

**Output:** Registered Image

1. Load Input images to be registered.

2. Define matching points in both images for correspondance.

3. Estimate transformation parameters (scaling,rotation,translation in x direction(tx),translation in y direction(ty),shearing in x direction and shearing in y direction) using Euclidean Warping using Euclidean Warping and construct the transformation matrix using equation (5.12).

4. Apply Levenberg Marquardt algorithm to minimize the intensity difference in overlapped region of the images to be registered and according to that transformation matrix of equation (5.12) is updated.

LM algorithm along with sampling consists of following steps.

(i) For each pixel $i$ at location $(x_i, y_i)$, Compute its corresponding position in the other image using equation (5.10).

(ii) Compute the error in intensity between the corresponding pixels (in overlapped region).

$$e = I'(x', y') - I(x, y)$$  \hspace{1cm} (5.34)

and also compute the sum of the sum of the squared intensity error using equation (5.14).

(iii) Compute the intensity gradient $\frac{\partial I'}{\partial x'}, \frac{\partial I'}{\partial y'}$ using bilinear intensity interpolation on $I'$.

The gradient of an image measures how it is changing. Bilinear interpolation is used when we need to know values at random position on a regular 2D grid as an image. In this approach the four nearest neighborhood is used to estimate the intensity at a given location [15].

Suppose (i,j) is center point in 3 X 3 region (as shown in figure 5.3) of an overlapping area of an image $I'$,
Then intensity gradient in x direction is given by

$$\frac{\partial I'}{\partial x'} = \frac{(i_1 - i_2) + (i_3 - i_4)}{2}$$  \hspace{1cm} (5.35)

Intensity gradient in y direction is given by

$$\frac{\partial I'}{\partial y'} = \frac{(i_3 - i_1) + (i_4 - i_2)}{2}$$  \hspace{1cm} (5.36)

Where  

- $i_1 = I'(i-1,j+1)$
- $i_2 = I'(i-1,j-1)$
- $i_3 = I'(i+1,j+1)$
- $i_4 = I'(i+1,j-1)$

(iv) Compute $\frac{\partial e}{\partial m_k}$ where k=1 to 8 using equation (5.17) to (5.24)

(v) Compute an approximate Hessian matrix $A$ using equation (5.26) and the weighted gradient vector $B$ using equation (5.27)

(vi) Solve the system of equation (4.31) $$(A + \lambda I_d) \Delta m = B$$ and update the motion estimate using equation (5.31)

(vii) Apply new transformation matrix to find correspondence between images to be registered

Using equation (5.13)

(viii) Check that the error in equation (5.14) has decreased; if not, increment $\lambda$, and compute a new $\Delta m$. 

(ix) Continue iterating until the error is below a threshold or a fixed number of steps have been completed.

5. Apply backward mapping as stated in section 5.2.1 and blend two images (or more than two).

6. To reduce visible artifacts blend overlapped region by finding weight-distance parameter using equation (5.33) and find new intensity value using equation (5.32).

7. Show the result.

8. End
5.4 Results

Proposed algorithms are implemented on different multiview images and results are compared for euclidean warp and affine warp with LM optimization.

Image 1

Figure 5.5  Input images showing control point selection for Euclidean warp based method. This control point for correspondence must be matched to registered image accurately. From these selected points transformation parameters are estimated using equation (5.2).

Figure 5.6  Euclidean Warp
Chapter 5 Multiview Image Registration

Figure 5.7  Registration of two images using Euclidean warp

Figure 5.8  Control point selection for registered image and third input image

Figure 5.9  Registration of three images using Euclidean warp
Figure 5.10  Registration of two and three images using affine warp and LM optimization

Image 2
Figure 5.12 Registration two (a) and of three images (b) using Euclidean warp

Minor difference in selection point can cause seems in resultant images, which is shown in figure 5.12.
Figure 5.13  Registration of two and three images using affine warp and LM optimization.

Figure 5.14  Input images to be registered

Figure 5.15  Registration of two images using euclidean warp
Figure 5.16 Registration of two images using affine warp and LM optimization

Image 4

(a) (b)

(c)

Figure 5.17 Input images (a-b) and output registered Image (c) using Euclidean warp
Figure 5.18  Registered Image using affine warp and LM optimization

Figure 5.19  Input Images to be registered

Figure 5.20  Registration of two images using Euclidean warp
Figure 5.21 Registration of three images using Euclidean warp

Without LM optimization, seems are visible because of some minor mistakes in selection points results in misregistration.

Figure 5.22 Registration of two(a) and three(b) images using affine warp and LM optimization
Image 6

Figure 5.23  Four Input Images to be registered

Figure 5.24  Registration of two (a), three (b) and four(c) images using affine warp and LM optimization
5.5 Discussion and Concluding Remark

From the above result, we can see that affine warp with LM optimization gives the best result. The most important limitation of the first method using Euclidean warp is the transformation matrix (equation 4.2) which is derived from selected initial point, is not known whether it is away from the minimum or close to the minimum. This transformation matrix is used directly to find warp for reference image. So here matching points must be selected very carefully with great accuracy, any minor change in the selection of those points will lead to distortion in the output image that we can see from the above results and second limitation is that the reference
points chosen are only static objects, those whose position remain stationary with respect to images to be registered. Euclidean warp takes only translation, rotation and scaling only, shearing of the correspondence points are not considered but in second algorithm using affine warp also takes the effect of shearing along with rotation, translation and scaling. Transformation matrix using affine parameters is optimized to see the minimum error by using all nearest neighbor pixel, so here though minor changes are there in selection of reference points will be neglected. Another major difference between both approaches is in computational efficiency. Euclidean warp is taking less time as directly formed transformation matrix is used (In proposed algorithm, it is taking average of 35 seconds, derived from different size images) but LM optimization algorithm is taking more time compared to previous (average of 70 seconds). Our experimental results suggest that second approach are competitive in quality with the best currently available technique but main limitation of this algorithm is, it is not automatic so we have to give control point selection and there must be some overlapping region in images to be registered.