IMAGE MOSAICING

MPA PROJECT

Group:12

Aman Raj Vanawat 16ucs030

Divyesh Maheshwari 16ucs063

Abstract

Mosaicing is one of the techniques of image processing which is useful for tiling digital images. Mosaicing is blending together of several arbitrarily shaped images to form one large radiometrically balanced image so that the boundaries between the original images are not seen. We can combine information from many images into one image for better understanding what they represent. Though there exist some complex cameras to increase the field of view, it would be good if we could use the normal cameras to do this. Image mosaicing is a technique which enables to combine together many small images into a one large image, from which more information can be collected easily.

Every Image have some objects which are also the feature of the image. Each of these features have some direction and position where it can be viewed more clearly. The current available simple mosaic algorithm which only stitch the input images. We want an algorithm in which the features of the mosaiced object are being viewed and by this we can get more information from the image.

In this paper we have implemented the algorithm by which we can shows these features in best possible view. This process used some existing mosaicing algorithm for generating novel views from limited number of views.

Introduction

The Practice of image mosaicing have been since long before the age of modern computers. Back then, the acquired the images were manually pieced together. The need for mosaicing continued to increase later in history as satellites started sending pictures back to earth. Improvements in computer technology became a natural motivation to develop a computational technique and to solve related problems. It is widely used in daily life by stitching pictures into panoramas or a large picture which can display the whole scenes vividly. For example, it can be used in virtual travel on the internet, building virtual environments in games and processing personal pictures.

After development of advanced computation, we have developed many algorithms that ease our process of image mosaicing. For generating a comprehensible mosaic, we take a referential image and make other images align to the referential image and then paste together. This Depends on the path of camera so after aligning with other images may produce Distortion (for e.g. elongation, contraction, viewing at an angle, etc.) in them.

Conventional method of mosaicing only some parts of the scene can be viewed with minimal distortion. Later this algorithm developed to reduce this distortion and we can see the scene with minimal Distortion. In Image Mosaicing the image is divided into (usually equal sized) rectangular sections, each of which is replaced by another photograph that matches the target photo. This Project is based on the developed version of this algorithm that helps to stitch this image into a best view image.

Background

Image Mosaicing

Definition

” Image mosaicing” is a technique for constructing a large seamless panoramic image from many source images. A Video can show us the full scene but not at the same time because the field of the camera is less than that of human eye. Image mosaicing help us to combines these scenes so that we can acquire more information from this composition of the scenes. Image mosaicing is the main focuses of the researches in the recent years. There has been variety of changes in the classical algorithms that enhance the image resolution and the field of view.

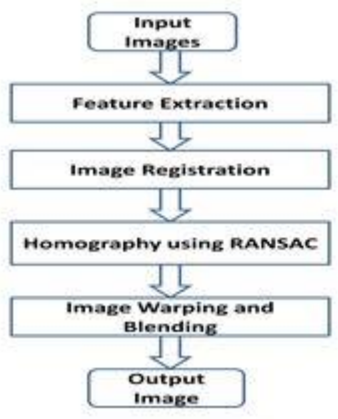
As mentioned earlier, image mosaicing involves the individual images to be aligned to some reference image. The relation between the coordinates of the reference image and the image to be aligned is governed by a property called the homography. The following are the major issues in image mosaicing

Image alignment: Finding the homographies between the given images with respect to some reference image.

Image registration: Pasting the aligned images so that overlapping regions between them are merged.

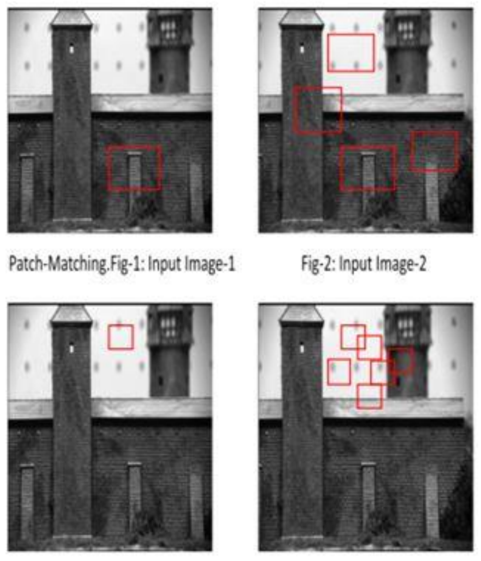
Image blending: Smooth out or rectify the intensity differences between the two stitched images.

Image Mosaicing Model



Feature Extraction

Feature Extraction is the first step of Image Mosaicing Process. In this we take the common element in the input images that to be matched. For images to be matched they are taken inside an image patch. These image patches are groups of pixels in images. Patch matching is done for the input images**.**


Patch-Matching.Fig-1: Input Image-1
Fig-2: Input Image-2
Patch-Matching.Fig-3: Input Image-1
Fig-4: Input Image-2

Let’s analyze the above figure for feature Extraction. In fig 1 and fig 2 there is a best patch match as there is a common patch in both the images. When we consider fig 3 and fig 4 there is some bad patch match as there is similar patches in fig 4 for the given patch in fig 3. So, the exact matching is difficult as there are slightly equal intensities.

Corner features are the best features to give quantitively measurement in Image Mosaicing. Corners are the best feature to match. The corner features are stable in course of change in viewing angles. The other most important features of corner are that it shows abrupt change in the intensities in the neighborhood pixels. For example

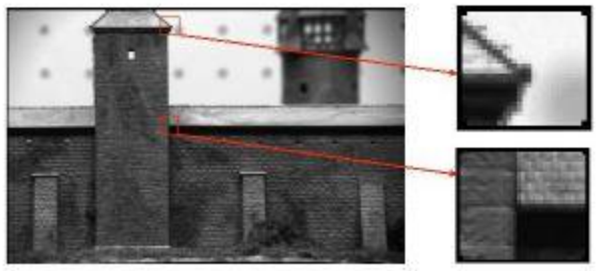


Fig-5: Corners (Junction of Contours.)

Corners can be detected in the image by various algorithms. Some of them are:
1.Harris Corner Detection Algorithm
2.SIFT Algorithm
3.Fast Algorithm
4.SURF Algorithm

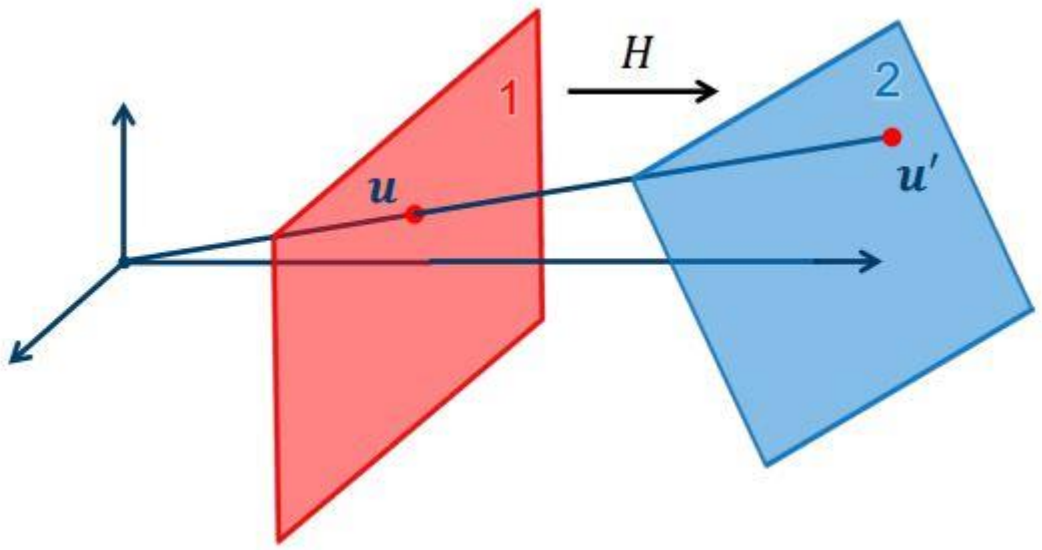
Image Registration

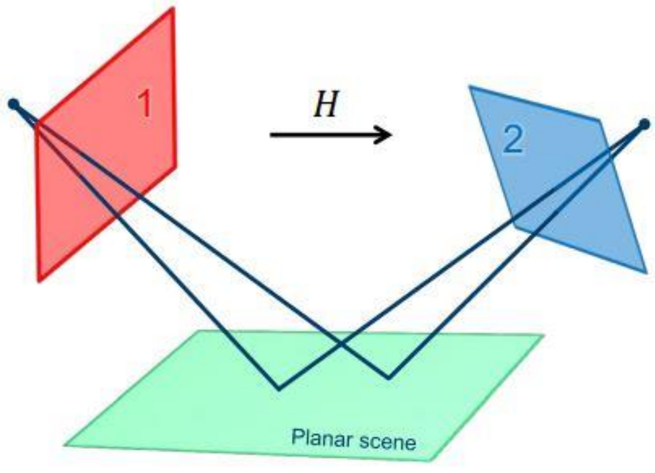
Image registration refers to the geometric alignment of a set of images. The different sets of data may consist of two or more digital images taken of a single scene from different sensors at different time or from different viewpoints. In image registration the geometric correspondence between the images is established so that they may be transformed, compared and analyzed in a common reference frame. This is of practical importance in many fields, including remote sensing, computer vision, medical imaging.

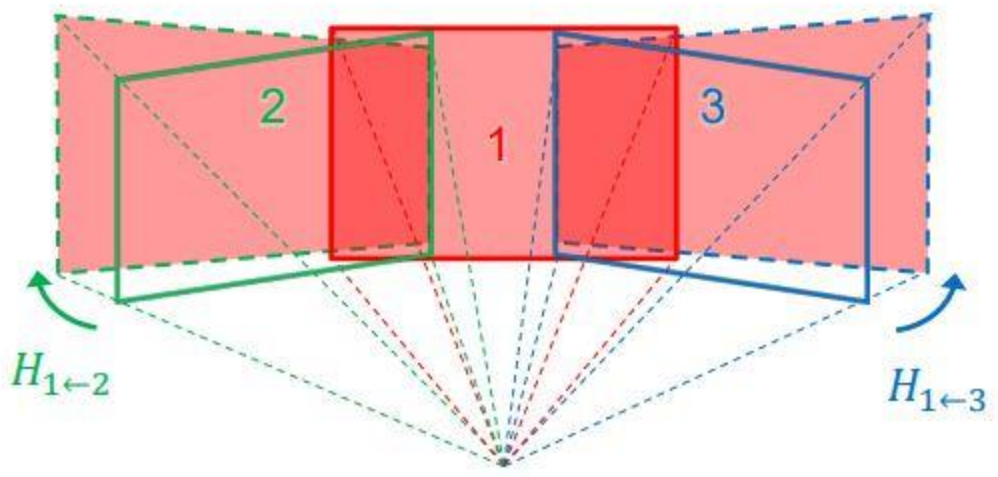
It can be divided into following classes
1.Image that uses Pixel value Directly. E.g. Correlation Method
2.Algorithm that uses Frequency Domain. E.g. Fast Fourier Transform based.
3.Algorithm that uses low level feature like edge and corners. E.g. Feature Based Method.
4.Algorithms that use high-level features such as identified parts of image objects, relations between image features. E.g. Graph-theoretic methods.

Computing Homography

Calculating Homography between two images is the third step for Image Mosaicing. By using Homography undesired corner which do not belong to the overlapping area are removed.
 It can be done by
1.RANSAC Algorithm
2.Basic Homography Estimation







RANSAC Algorithm

This Algorithm was first developed by Fischler and Bolles. RANSAC is an abbreviation for “Random Sample Consensus”. It is used to estimate parameters of a mathematical model from a set of observed data which contain outliers. It is an iterative and non-deterministic algorithm such that it produces a reasonable result with certain probabilities and probability increase as the iteration increases. Given a fitting problem with parameters considering the following assumptions.

1. Parameters can be estimated from N data items.

2. Available data items are totally M.

3. The probability of a randomly selected data item being part of a good model is P_g.

4.The probability that the algorithm will exit without finding a good fit if one exists is P_{fail}.

Algorithm

1. Selects N data items at random.

2. Estimates parameter x.

3. Finds how many data items (of M) fit the model with parameter vector x within a user given tolerance. Call this K.

4. If K is big enough, accept fit and exit with success.

5. Repeat 1.4 L times.

6. Fail if you get here.

How big K has to be depended on what percentage of the data we think belongs to the structure being fit and how many structures we have in the image. If there are multiple structures than, after a successful fit, remove the fit data and redo RANSAC.

We can find L by the following formulae:
P_{fail} = Probability of L consecutive failures.
P_{fail} = (Probability that a given trial is a failure) L.
P_{fail} = (1 - Probability that a given trial is a success)L.
P_{fail} = (1-(Probability that a random data item fits the model)N)L.
$P_{fail} = (1-(P_g)^N)^L$
$L = log(Pfail)/\ log(1 - (Pg)N)$

Basic Homography Estimation

In basic Homography estimation we require.

1.A basic homography estimation method for 4-point correspondence.

2.A way to determine the inlier set of point correspondence for a given homography.

$$\begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix}\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} u' \\ v' \\ 1 \end{bmatrix} \Leftrightarrow \begin{Bmatrix} uh_1 + vh_2 + h_3 = u' \\ uh_4 + vh_5 + h_6 = v' \\ uh_7 + vh_8 + h_9 = 1 \end{Bmatrix} \Leftrightarrow \begin{bmatrix} 0 & 0 & 0 & -u & -v & -1 & v'u & v'v & v' \\ u & v & 1 & 0 & 0 & 0 & -u'u & -u'v & -u' \\ -v'u & -v'v & -v' & u'u & u'v & u' & 0 & 0 & 0 \end{bmatrix}\begin{bmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \\ h_5 \\ h_6 \\ h_7 \\ h_8 \\ h_9 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \Leftrightarrow A\mathbf{h} = \mathbf{0}$$

The third row(highlighted row) is the linear combination of first and second row.

$Row_3 = -u' . Row_1 - v' . Row_2$

Hence every 9 unknown variable can be found by two equation of U_i <-> U_i'.

$$\begin{bmatrix} 0 & 0 & 0 & -u_1 & -v_1 & -1 & v_1'u_1 & v_1'v_1 & v_1' \\ u_1 & v_1 & 1 & 0 & 0 & 0 & -u_1'u_1 & -u_1'v_1 & -u_1' \\ 0 & 0 & 0 & -u_2 & -v_2 & -1 & v_2'u_2 & v_2'v_2 & v_2' \\ u_2 & v_2 & 1 & 0 & 0 & 0 & -u_2'u_2 & -u_2'v_2 & -u_2' \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix} \begin{bmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \\ h_5 \\ h_6 \\ h_7 \\ h_8 \\ h_9 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \vdots \end{bmatrix}$$

$$A\mathbf{h} = \mathbf{0}$$

Since h is homogenous,for determining h up to scale the matrix A need to have the rank 8.It is sufficient with 4 point correspondence where no 3 points are collinear.We can calculate non trival solution of Ah=0 by SVD.

$$\text{svd}(A) = USV^{T.}$$

After applying SVD the solution is given by the right singular vector which is a last column of V i.e $h=V_9$.

Image Warping and Blending

When we stitch the image there might be some distortion as well as color mismatch. Image Warping is the process that helps in digitally processing the photo to remove Distortion.It can be used to creative purpose as well.

The final process is to blend the pixel colors in the overlapped region to avoid the seams.One method is to use the Histogram Equalization in the resultant image.Second method is to use the weighted average of the color values of both the joined images.We generally use alpha factor often called the alpha channel having the value α =1 and gradually decrease as we got to the corners where value of alpha becomes 0. . Where at least two photos overlap occurs in an output mosaic we will use the alpha values as follows to compute the colour at a pixel in there, suppose there are two photos, I1, I2, overlapping in the output photo; each pixel (x,y) in photo Ii is represented as Ii(x, y) = (αiR, αiG, αiB, αj) where (R,G,B) are the color values at the pixel. We will compute the pixel value of (x, y) in the stitched output photo as [(α1R, α1G, α1B, α1) + (α2R, α2G, α2B, α2)]/(α1+α2).

Algorithm Implemented by us

Step 1: Choose four features points between two images U_i & U_i'.

Step 2: Represent the points by suitable Descriptors.

Step 3: Estimate the Homography H such that $U_i = HU_i'$.

Step 4: Apply the Inverse Homography to the corners to find the x and y offset.

Step 5: Create a Blank Image of size First image combined with Second image with offset.

Step 6: Stitch the first Image as it is in the Blank Image.

Step 7: Apply the homography to the final image and pick the corresponding color value of the coordinates from the second image and assign to it.

Project Code(MATLAB)

```
%Main Function
% code for stitching images
close all;
clear all;
% stitching image 0 and 5
% loading of images
img1=imread('0.jpg');
img2=imread('5.jpg');
% corresponding points of images
a=[1003,916,963,1505,1722,1536,1735,1115];
b=[1130,236,948,712,1881,1045,2065,617];
% call mosaicing function
result=mosaicing(img1,img2,a,b);
imwrite(result,'mosaicing05img.jpg');
% output image
figure,imshow(result);
% stitching image 0,5 and 4
img2 = imread('4.jpg');
img1 = imread('mosaicing05img.jpg');
b=[95,576,22,1105,804,1022,1196,1082];
a=[1628,955,1575,1753,2333,1535,2627,1579];
result=mosaicing(img1,img2,a,b);
imwrite(result,'mosaicing054img.jpg');
figure,imshow(result);
% stitching image 0,5,4 and 2
img1 = imread('mosaicing054img.jpg');
img2 = imread('2.jpg');
a= [2579,928,2629,1580,2957,956,2976,1178];
b= [349,583,157,1339,1105,701,1035,1025];
result=mosaicing(img1,img2,a,b);
imwrite(result,'mosaicing0542img.jpg');
figure,imshow(result);
% stitching image 0,5,4,2 and 3
img1 = imread('mosaicing0542img.jpg');
img2 = imread('3.jpg');
a= [2572,905,2879,950,2591,1496,2935,1616];
b= [127,129,719,73,85,827,733,1011];
result=mosaicing(img1,img2,a,b);
imwrite(result,'mosaicing05423img.jpg');
figure,imshow(result);
% stitching image 0,5,4,2,3 and 1
img1 = imread('mosaicing05423img.jpg');
img2 = imread('1.jpg');
a= [2255,1347,2591,1495,2879,949,2976,1178];
b= [899,1449,1539,1259,1667,325,1993,473];
result=mosaicing(img1,img2,a,b);
imwrite(result,'mosaicing054231img.jpg');
figure,imshow(result);
```

```
% Mosaicing.m
function [ temp ] = mosaicing( img1,img2,a,b )
% general function
%Convert image to double precision
img1=im2double(img1);
img2=im2double(img2);
% call homography function to calculate homography matrix
H=homography(a(1),a(2),a(3),a(4),a(5),a(6),a(7),a(8),b(1),b(2),b(3),b(4),b(5),b(6),b(7),b(
8));
% call projected points function
[x1,y1,x2,y2,x3,y3,x4,y4]=projected_points(img2,H);
% corners points of both images
corners=[x1 y1;x2 y2;x3 y3;x4 y4;1 1;1 ,size(img1,1);size(img1,2),
1;size(img1,2),size(img1,1)];
% call min_max_offset function
[max_height,max_width,x_offset,y_offset]=max_min_offset(corners);
% create new image with max height and max width
temp = zeros(max_height,max_width);
temp=im2double(temp);
% call image1_copy function
temp=image1_copy(temp,img1,x_offset,y_offset);
% iterate each coordinate of new image and apply homography
for i=1:size(temp,1)
for j=1:size(temp,2)
% projected points after applying homography
projected_point= [j, i, 1]*H;
a1=projected_point(1,1)/projected_point(1,3); % points in image 2
b1=projected_point(1,2)/projected_point(1,3);
a1=round(a1); % rounding values
b1=round(b1);
if(a1>=1 && b1>=1 && a1<=size(img2,2) && b1<=size(img2,1))
temp(i+y_offset,j+x_offset,1)=img2(b1,a1,1); %red channel
temp(i+y_offset,j+x_offset,2)=img2(b1,a1,2); % green channel
temp(i+y_offset,j+x_offset,3)=img2(b1,a1,3); % blue channel
end
end
end
end
```

```
%homography(1).m
function H=homography(x1,y1,x2,y2,x3,y3,x4,y4 , xp1,yp1,xp2,yp2,xp3,yp3,xp4,yp4)
%This function will find the homography betweeb 4 points using svd


A=[
-x1  -y1  -1   0    0    0   x1*xp1   y1*xp1   xp1;
 0    0    0 -x1   -y1  -1   x1*yp1   y1*yp1   yp1;
-x2  -y2  -1   0    0    0   x2*xp2   y2*xp2   xp2;
 0    0    0 -x2   -y2  -1   x2*yp2   y2*yp2   yp2;
-x3  -y3  -1   0    0    0   x3*xp3   y3*xp3   xp3;
 0    0    0 -x3   -y3  -1   x3*yp3   y3*yp3   yp3;
-x4  -y4   -1  0    0    0   x4*xp4   y4*xp4   xp4;
 0    0    0  -x4  -y4  -1   x4*yp4   y4*yp4   yp4];

[~,~,V] = svd(A);


H=V(:,end);
% normalizing homography matrix
H = reshape(H,3,3);
for i=1:3
    for j=1:3
        H(i,j)=H(i,j)/H(3,3);
    end
end
end
```

```
% projected_points.m
function [ x1,y1,x2,y2,x3,y3,x4,y4 ] = projected_points( img2,H )
% function to apply inverse homography on corners of image 2
IH=inv(H);
trans_points=[1 1 1] * IH;
x1=trans_points(1,1)/trans_points(1,3);
y1=trans_points(1,2)/trans_points(1,3);

trans_points=[1,size(img2,1), 1] * IH;
x2=trans_points(1,1)/trans_points(1,3);
y2=trans_points(1,2)/trans_points(1,3);

trans_points=[size(img2,2),1, 1] * IH;
x3=trans_points(1,1)/trans_points(1,3);
y3=trans_points(1,2)/trans_points(1,3);

trans_points=[size(img2,2), size(img2,1), 1] * IH;
x4=trans_points(1,1)/trans_points(1,3);
y4=trans_points(1,2)/trans_points(1,3);
end
```

```
% max_min_offset.m
function [max_height,max_width,x_offset,y_offset ] = max_min_offset( corners )
% function to calculate min , max , x offset , y offset values of x and y
MAX_X=0;
for i=1:8
if MAX_X<= corners(i,1)      % max x calculation
MAX_X=corners(i,1);
end
end

MAX_Y=0;
for i=1:8
if MAX_Y<= corners(i,2)      % max y calculation
MAX_Y=corners(i,2);
end
end

MIN_X=0;
for i=1:8
if MIN_X>= corners(i,1)      % min x calculation
MIN_X=corners(i,1);
end
end

MIN_Y=0;
for i=1:8
if MIN_Y>= corners(i,2)      % min y calculation
MIN_Y=corners(i,2);
end
end

max_height=round(MAX_Y-MIN_Y);     % maximum height of new image
max_width=round(MAX_X-MIN_X);      % maximum width of new image
x_offset=round(abs(MIN_X));        % x offset
y_offset=round(abs(MIN_Y));        % y offset

end
```

```
% image1_copy.m
function [ temp ] = image1_copy( temp,img1,x_offset,y_offset )
% function to copy image 1 to canvas taking care of offsets
for i=1:size(img1,1)
    for j=1:size(img1,2)

        temp(i+y_offset,j+x_offset,1)=img1(i,j,1);  % copy intensity values
        temp(i+y_offset,j+x_offset,2)=img1(i,j,2);
        temp(i+y_offset,j+x_offset,3)=img1(i,j,3);

    end
 end

figure, imshow(temp);
end
```

Input Images

All images are of size **1164x2496.**

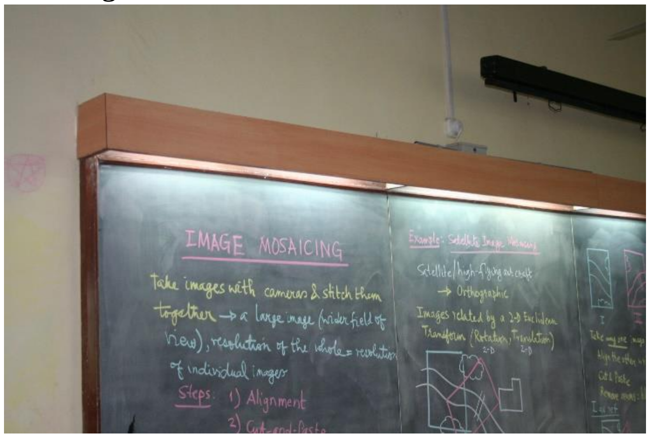


Image 1

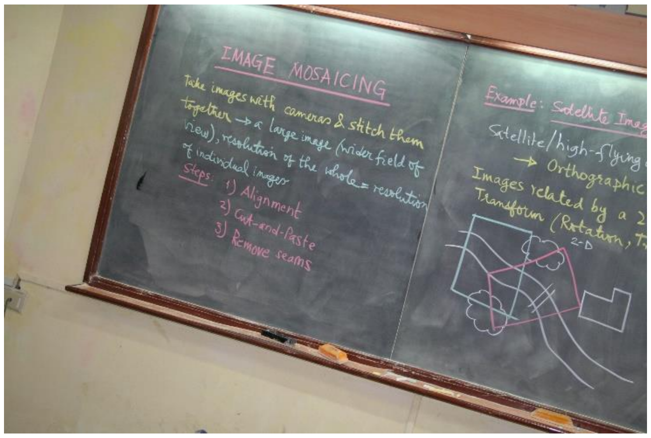


Image 2

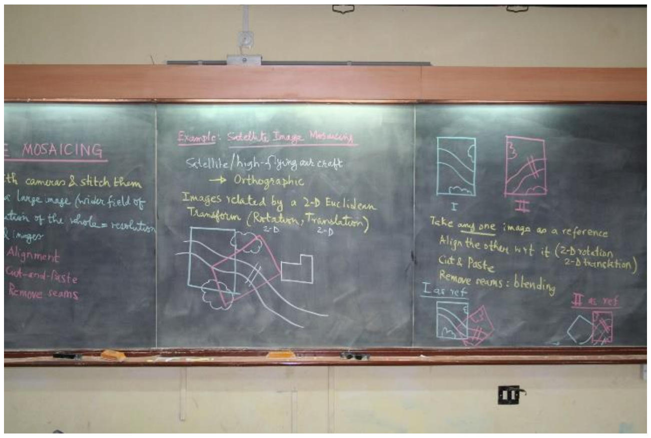


Image 3

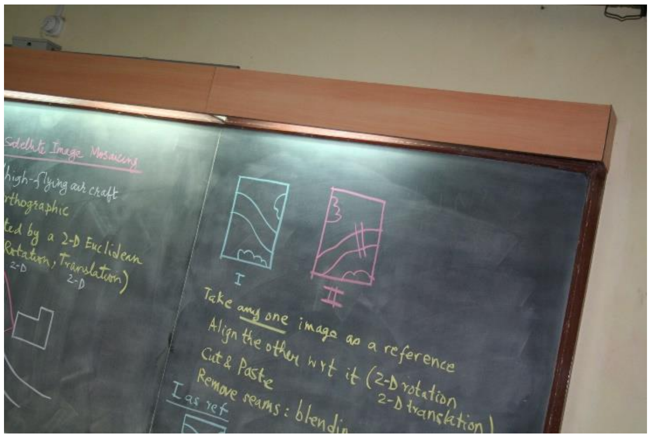


Image 4

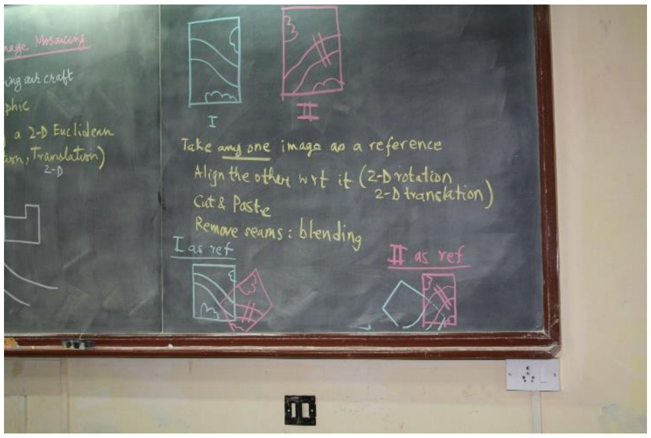
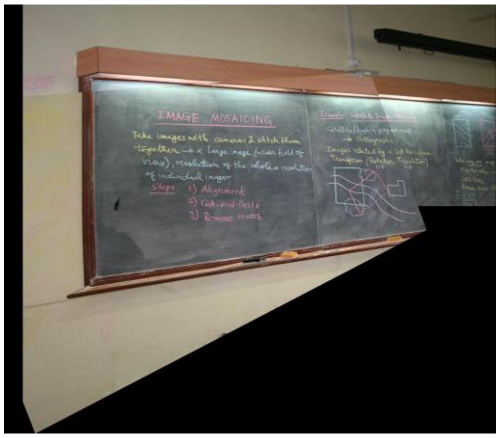
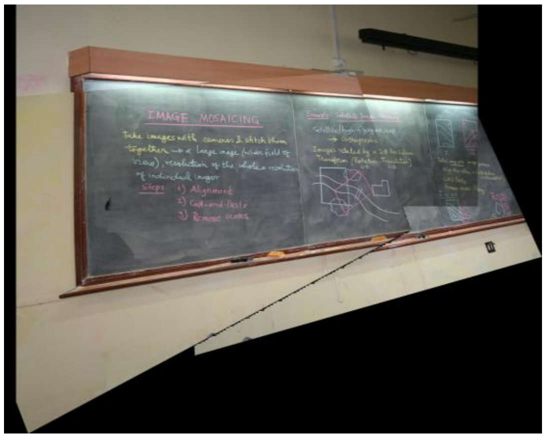

Take any one image as a reference
Align the other wrt it (2-D rotation
2-D translation)
Cut & Paste
Remove seams : blending
I as ref
II as ref

Image 5

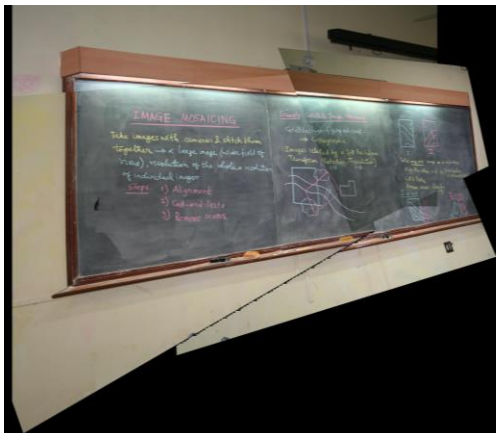
Output Images

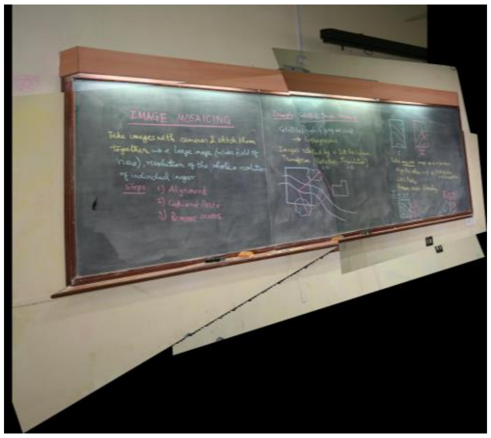


Output = Image 1 + Image 2



Output 1 = Output + Image 3





Output 2 = Output 1 + image 4   Output 3 = Output 2 + image 5

Observation

- Mosaicing of multiple images cannot be achieved by repeatedly warping new images to one reference image. Hence, after mosaicing 4 images to the reference image, the image alignment doesn't look good anymore.

- The methods work fine for all types of documents but they consume time.

- It may fail if the sequence is missed.

Applications

- We can create high resolution image without using complex cameras and combine images for better representation of images.
- It can also be helpful in Video processing and mark an impact.
- It is used in Image Stabilization and Resolution enhancement.

Assumptions

- The images to be mosaiced have a large correspondence between them, i.e., the inter-frame motion between them is small.

- The lighting conditions are almost the same in the input images.

- All the input images are assumed to be of equal size and captured from the same camera.

Conclusion

Image Mosaicing is widely used in creating a best view image by number of limited view images. Image mosaicing is also used in today's smartphones also known as panorama and are working very effectively. In this paper we only implemented a basic mosaicing algorithm that works well in 2D frame. We have explored all the steps required for image mosaicing and their introduction. Various algorithm is listed that can make our algorithm more effective.

Reference

1. M. A. Fischler, R. C. Bolles. Random Sample Consensus*: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography*. Comm. of the ACM, Vol 24, pp 381-395, 1981.
2. Hima Bindu: *Generating a Best View Mosaic from limited number of views.*
3. Dushyant Vaghela, Prof Kapildev Naina: *A Review of Image Mosaicing Techniques.*
4. http://vision.stanford.edu/ birch/klt
5. R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University