Volume 4, Issue 3, March 2014 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Image Mosaicing using first order Moments and Pixel to Pixel Comparison

Geeta Hegde*, Shaheen Mujawar, Rekha V B, Chandrakala G
Computer science &Engg
SGBIT, Belgaum, India

Abstract—In real-life situations, the field of view of the imaging devices is usually smaller than the scene to be imaged, due to the inherent limitations of the imaging devices. Even if the whole scene is captured in a single exposure, it could result in an image having poor resolution. In such circumstances, it will never be possible to capture the scene as a single image for further processing or human inspection or secondary storage. The only solution to such a problem is to capture the scene by splitting the scene into adjacently placed multiple frames. This process warrants a post imaging operation that requires cohering of these small images to create a single image corresponding to the source scene. The process of cohering of these small / split images (which are also called as image components) has been referred as IMAGE MOSAICING in the literature. Here mosaicing of images of same size and same intensity is done based on texture moments, pixel to pixel comparison. Carrying out the process of cohering interactively on a computer or completely automating the task of cohering is an active research area in the development of Image Processing, Pattern Recognition and Computer Vision applications.

Keywords— Registration, Composition, First order moment, contrast, Mosaicing.

I. INTRODUCTION

The general task of image mosaicing is divided into two sub tasks namely, Image Registration and Image Composition. Image registration is an important task for matching two or more images. Given a set of images of a scene with overlap region, one can register them and combine them into a single image, which is considered to be a three step process. The first step in registering two images is to decide upon the feature space, which includes edges, contours, surfaces, corners and line intersections, statistical features such as moments or centroids and higher-level structural and syntactic descriptions. The second step is to select a similarity metric, which represents one of the important aspects of how the registration transformation is determined. The third step is to select the best search space and the best search strategy. Due to large computational costs associated with many matching features, it is very important to reduce the number of features for similarity measures. Alignment of images [5] may not be always proper or perfect due to errors that occur during image registration. The errors mainly result from incompatible model assumptions, presence of dynamic scenes, etc. Furthermore, in most of the cases, not all images that need to be mosaiced are exposed evenly to the camera at all due to variations in lighting conditions, automatic controls of cameras, printing or scanning devices, etc. These unwanted errors get alleviated during the compositing process. The important issue in image compositing is the problem of determining how the pixels in an overlap region should be represented. Determination of the best separation border between overlap regions has the potential to eliminate the geometric distortion.

The motivation to take of this work is because of limitations of imaging device which is not able to capture a single large scene due to its inherent limitations. The efficient working of imaging devices with real-life applications warrants either overcoming of the limitation of resolution or the limitation of field of view. These limitations are overcome by IMAGE MOSAICING. Keeping the aforesaid facts in the backdrop, it is motivated to develop simple algorithmic models that are capable of creating a single image from the given split images. In this work, new ways of finding overlap regions for mosaicing of split images are devised. Texture features are explored to find the common region between split images. Accordingly, three sets of algorithms based on (i) Texture correspondence and (ii) Mosaicing based on pixel to pixel correspondence is developed.

II. RELATED WORK

Mosaicing is also expressed as a way of creating large images from image sequences taken from a moving / still camera. Early mosaicing algorithms are constrained to camera parameters and scene geometry. The simplest mosaics are created from a set of images, which are displaced by pure image plane translations. This is typical to aerial and satellite images, in which the objects of the scene are distant from the camera, and the camera motion is modelled as a translation process parallel to the image plane. When the scene, as far as planar, and the camera motion are concerned a more general transformation for image alignment are used. A. Elibol, et.al. (2008) [1] presents a new global alignment method. It works on the mosaic frame and does not require any non-linear optimization. There is no limitation on the problem size and since its computational cost is very low, it is faster.

YIN YanSheng et.al. (2007) [2] proposed a novel method to mosaic two microscopic images with an amplification co-efficient of 1000. The two images are denoised by Gaussian model, and feature points are then extracted by using Harris corner detector. The feature points are filtered through Canny edge detector. Using the transformational parameters acquired, the two images are transformed into the universal coordinates and merged to the final mosaic image.

D.R.Ramesh Babu et.al.(2004)[3] proposed an algorithm for mosaicing of multiple split images is presented. Corresponding points selection is based on the intersection of edges & distance measured between the junction points are used for finding corresponding control points in the overlap region.

G Hemantha Kumar et.al. (2004)[4] proposed a novel and simple approach to mosaic two split images of a large document based on pixel value matching. The method compares the values of pixels in the columns of split images to identify the common or overlapping region (OR) in them, which helps in mosaicing of split images of a large document.

In summary, literature available indicates that the methods focus on domain specific applications like panoramic mosaic or video imagery. There is scope for further research in other domains such as medical, computer graphics, defence applications etc.

III. MOSAICING OF EQUAL SIZED IMAGES USING FIRST ORDER MOMENTS.

In this section, approach based on comparing the values of pixels in the columns of the split images to mosaic them in order to produce a single, large document image is presented. Mosaicing is achieved by identifying the positions of overlapping region (OR) in the split images. The OR is obtained by comparing the values of pixels in the columns of the split images. If the columns match, then the pointers j (where j is the pointer to columns in the split images) of both split images 1 and 2 are incremented by one. If there is no match, the pointer j of the split image 1 is moved to the next column, while the pointer j of the image 2 remains unaltered. This procedure is repeated till an overlapping region is found. The method works based on the assumption that the OR is present at the right and the left ends of the split images 1 and 2 respectively

A. *Mosaicing of two images of equal size*

The two images say I_1 and I_2 are mosaiced by finding the overlap region of Image I_1 in Image I_2 and then composite operation is performed to make the overlap regions coincide with each other. The Image I_1 is considered as the reference image and Image I_2 is considered as the target image. . Mosaicing of two images by finding the first order moments of two images is shown in Figure 1.

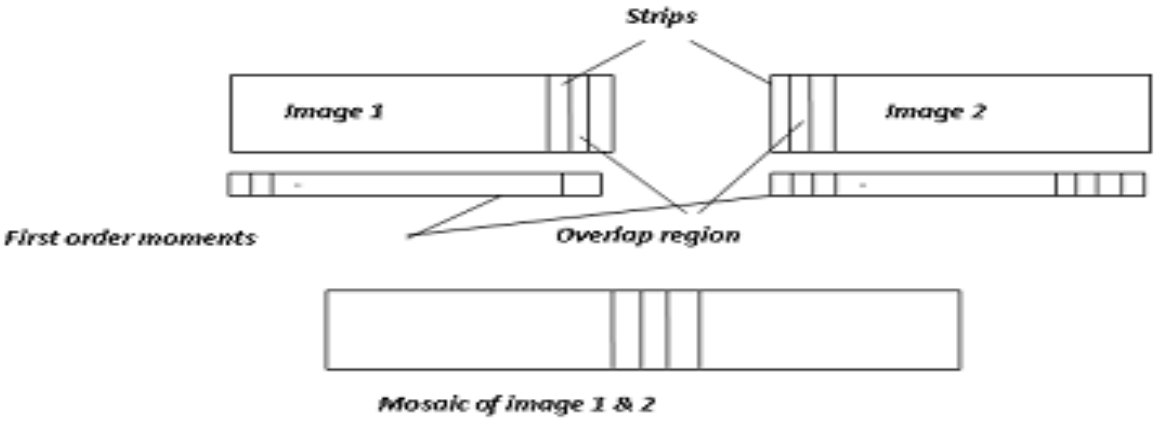


Fig 1: Block diagram for Mosaicing of two images by finding first order moments

The target image is initially divided into 'N' number of strips and for each strip the 1st order moments are computed. The moments are stored as feature vectors in the database. The strip selected from the target image is slided over the reference image from left to right and from top to bottom by one column / row pixels correspondingly to find the best overlap region. At the best match, the size of the strip is increased and matching is continued until the identical contiguous overlap region is found. Once the overlap region is found, the images are composed so as to make overlap regions of the images coincide. Initially code is written by finding the first order moments of both the images and comparing both the images. Comparison starts from moments of 1st column of 1st image with mean of 1st column of 2nd image, if match exists then column vectors of both the images are incremented by one.

If there is no match then only column vector of image 1 is incremented by one and is compared with 1st column vector of 2nd image. Each time count is incremented by one when mean of both the images match. Count value indicates the number of overlapping columns. The problem with this method is that mean value of both the images may be same even the pixel values are interchanged in the two images (Mean is same in both the images). Other problem with this method is that only some column vectors of 1st image matches with the second image but will not reach till the end of the 1st image. The developed methodology is put forth as Algorithm 1.

The algorithm 1, instead of working with one pixel at a time, an image strip of width 'W' is taken, which varies from 1 to N number of column pixels, where 'N' less than the maximum size (maximum width) of the image I_2. Features extracted from each strip of the target image are compared with the reference image I_1 to find the overlap region.

Suppose, there are M strips in the reference image (say, Image I_1) and N strips in target image(say, Image I_2), as given in Figure 3.2, then the total number of comparisons is of the order O(MN) in the worst case. For strip width (W > 1) pixels, the total number of comparisons is of the order O ((M-W)(N-W)). When the width 'W' is close to M or N, the time complexity reduces to a greater extent but at the cost of accuracy. The proposed algorithmic model is simple, robust and considers only low-level features for finding the overlap region. The algorithm assumes that the images are ortho normal and distortion free. In addition, the split images are assumed to share a maximum overlap region and the images have similar intensity values. The algorithm fails if the images are of different contrast.

1) Algorithm 1: Mosaicing of two images by finding the first order moments.

Input: Two split images (Image I1 and Image I2) with overlap region
Output: Single mosaiced image
Begin

Step 1: Divide Image I2 into 'N' strips, where N=1 to max-size of the Image I2.
Step 2: For each strip in I2, compute 1st order moments and store the values in the database as feature vectors.
Step 3: Slide the first strip of the I2 over I1 from left to right by one column pixels and top to bottom by one row pixels.
Step 4: Search for the identical and contiguous region, by comparing the feature vectors present in the database.
Step 5: List the corresponding co-ordinates of the matched strips as correspondences.
Step 6: Translate I2 over I1 so that the identified overlap regions are coincided with each other.

End

Drawback of algorithm 1 is explained by Figure 2

78	67	67	67	67
4	67	50	25	0
67	89	255	50	45
78	25	255	255	255
78	91	25	255	255

67	67.8	130.4	130.4	124.4

67	67	34	67	67
25	0	45	78	78
50	45	57	89	23
255	255	87	89	45
255	255	56	45	56

130.4	124.4	55.8	73.6	53.8

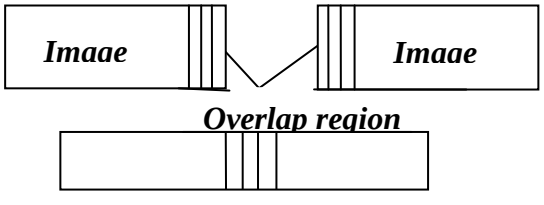
Fig 2: First-order moments to find the overlap region & its drawback.

In this case there is overlap between last two columns and 1st two columns of image1 and image2. But when scanning from left to right column 3 of image 1 and 1st column of image 2, pixel values between the 2 images match. But there is no match between 4th and 2nd columns of image 1 & 2. Hence program will give the output as there is no overlap between two images even though there is overlap. Other drawback of this algorithm is that sometimes the pixel values may match between the two columns of images but match may not be there till the end of image 1.In this case the column in image 1 should be incremented and start comparing from 1st column of image 2.

IV. MOSAICING BASED ON PIXEL TO PIXEL COMPARISON

A. *Mosaicing of two images of equal size*

Here two input images of same size and same intensity are taken and their size is found. Initially first column of 2nd image is compared with 1st column of 1st image. In 1st column from top to bottom each pixel of 1st image is compared with 2nd image. If 1st column of both image match then column vector of both image are incremented by one and again comparison is done from top to bottom. If there is no match between 1st column of 1st image and 2nd image, then 1st image column is incremented by one and is compared with 2nd image's 1st column. Each time if the whole column matches then the columns of both the images are incremented by one. The count value is incremented by one for each match between the columns of 2 images. Count value will give the overlapping region between the two images. In other case if only some of the columns of 1st image match with 2nd image columns, match is not there till the end of 1st image (till last column of 1st image), in this case the column (of 1st image) from where match started previously is stored in one variable and that variable value is incremented by one and the comparison is repeated from that column of 1st image with 1st column of 2nd image. It is shown in Figure 3



Mosaic of imaae 1 & 2

Fig 3: Block diagram for mosaicing of two images by pixel to pixel comparison of both the images.

1) Algorithm 2: Mosaicing of two images by comparing pixel values of both the images.

Input : Two split images of same size (Image I1 and Image I2) with overlap region.
Output: Single mosaiced image.
Begin

Step1: Start comparing pixel values from the column value of I1 stored in 'var'(or js1)(Initially its 1) with the 1st column of I2 from top to bottom. If match exists then go to Step 2. If match doesn't exist go to Step 5.
Step 2: Increment the row pointer of I1 & I2 and repeat comparison till the end of last row of both I1 & I2.If match exists till the last row of both the images go to Step 3.If
match doesn't exists till the end of last row of I1 & I2 go to Step 5.
Step 3: Increment the value of 'count' which holds the value of number of matching columns between I1 & I2. Store the column pointer value of I1 in 'var'.Initialize the values of row pointer of both I1 & I2 to 1 and column pointers of I1 & I2 are incremented by one.
Step 4: If last column of I1 is reached and if 'count'=0 (no match exists from that column of I1 stored in 'var' with 1st column of I2) then increment the column of I1 by incrementing the value of 'var' & initialize the column pointer of I2, row pointer of I1 & I2 to 1. Go to Step 1.
Step 5: If count≠0 continue else go to Step 6. count≠0 since only some of the columns of I1 matched with I2 but no match till last column of I1 then initialize the column pointer of I2 & row pointer of I1 & I2 to 1, initialize count value to 0 & increment the value of 'var' & assign it to column pointer of I1. Go to Step 1.
Step 6: If count=0(No match between I1 & I2 from column value of I1 stored in var), initialize the row pointer of I1 & I2, column pointer of I2 to 1,Increment the value of column pointer of I1 go to Step 1.
Step 7: If count=0 & 'var'=last column of I1, then there is no overlap between I1 & I2.End.
Step 8: If count≠0 and match exists till the last column of I1, then append I2 from the column from where there is no overlap with I1 to split image I1. Now I1 contains the
mosaic of split images I1 & I2.

End

B. *Mosaicing of two images of unequal size*

Here two input images of different size and same intensity are taken and their size is found out as shown in figure 4. Here I1 is having larger number of rows when compared with rows in image I2. Initially a part of I1 who's number of rows equal to number of rows in I2 is assigned to I4 and is compared with I2 by invoking Algorithm 2.If count value is still 0 then variable incrow which stores the row number of I1 is incremented by one & same number of rows equal to number of rows of I2 is assigned to I4 and the process is repeated. This is done till the last row of I1. Still if there is no match there is no overlap between the two images. If match is found, then append image 2 to image 1

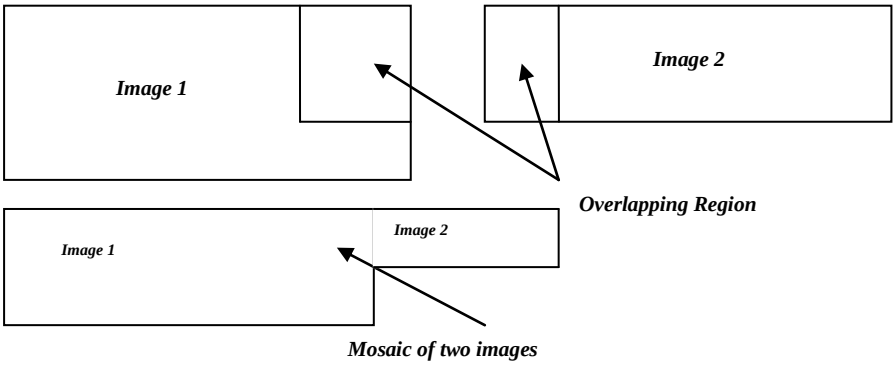


Fig. 4 Block diagram for mosaicing of 2 images of different size.

1) Algorithm 3: Mosaicing of two images of unequal size

Input: Two split images of different size (Image I1 and Image I2) with overlap region
Output: Single mosaiced image
Begin

Step1: Initially made 'incrow'=1 & assign a part of I1 whose number of rows (Row starting from incrow) equal to number of rows of I2, to I4.

Step 2: With I4(I4 instead of I1 in Algorithm 2) & I2 as input, Algorithm 2 is invoked.

Step 3: If match doesn't exist 'incrow' is incremented by one, Step 1 & Step 2 are repeated till match is found.
Step 4: Once match is found then mosaic I1 & I2 by appending I2 to I1.

End

V. RESULTS

A. *Mosaicing of equal size images*

Split Image (I_1)

Split Image (I_2)

Mosaiced Image of split Images (I_1) and (I_2)

B. *Mosaicing of two images of unequal size*

Image 1

Image 2

Mosaic of Image 1 & Image 2

Method used for mosaicing	Execution Time
Mosaicing based on FOM(First Order Moments)	0.000044 seconds
Mosaicing based on pixel to pixel comparison	0.232307 seconds

VI. CONCLUSION

In this work, new ways of finding overlap regions for mosaicing of split images are devised. Texture features are explored to find the common region between split images. Accordingly, three sets of algorithms based on (i) Texture correspondence and (ii) Mosaicing based on pixel to pixel correspondence. In case of texture-based algorithms, texture properties derived from the two split components are contrasted to locate the overlap region before accurate mosaicing is realized. First order moments are found out and are compared to find the correspondence between two images.

In case of mosaicing based on pixel to pixel comparison, each pixels of both the images are compared to find the overlap region between the two images. It overcomes the disadvantage of texture based methods, but the time required to find the overlap region is more in this method. Enhancements can be made in view of achieving high computation speed by choosing different algorithms based on image features.

REFERENCES

[1] A. Elibol, R. Garcia, O. Delaunoy, and N. Gracias., "*A New Global Alignment Method for Feature Based Image Mosaicing*", G. Bebis et al. (Eds.), Part II, LNCS 5359, pp.257–266, Springer-Verlag Berlin Heidelberg 2008

[2] YIN YanSheng,ZHAO XiuYang, TIAN XiaoFeng & LI Jia, "*Feature-point- extracting-based automatically mosaic for composite microscopic images*", Chinese Science Bulletin, Vol 52,2007.

[3] D.R.Ramesh Babu, G.Hemanth Kumar, Piyush M Kumat*, "Mosaicing of multiple Split Images Based on Dominant Junction Points*".

[4] G Hemantha Kumar, P Shivakumara, D F Guru and P Nagabhushan. "*Document Image Mosaicing: A Novel Approach*", Sadhana Vol. 29, Part 3, pp. 329–341, June 2004.

[5] Ligang Miao, Yongjuan Yue, Silong Peng. *"Error Analysis of Large-Scale Microscopic Image Mosaicing*", IEEE, 2006.

