ISSN: 2277-3754
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 1, Issue 3, March 2012

A Review: Semantic Template Matching using Color- Texture features

Rajeev K. Singh, Uday Pratap Singh, Sunil Phulre

Abstract*— Finding corresponding features between images is a prerequisite for many image related applications such as image retrieval, motion detection, shape reconstruction, etc. This paper gives an introduction into the most prominent approaches of the different algorithms which are involved in the process of generating matches between two images. In case of feature matching the first step is to detect features in both images with a feature detector. Ideally, the detected features represent the same image content across both images. A feature descriptor is used to represent the detected features in a distinctive way. Correspondences are defined between the two sets of features using a matching strategy that compares features based on a similarity measurement. The three parts of the detect–describe–match (DDM) framework determine the performance of image matching. The first step is the basis of this framework. Unstable and variant features increase the difficulties of the next two steps. Researchers mostly focus on the first step for invariant feature extraction and have proposed many excellent detectors.

***Key Index*-Detecting local Features, Interest points, Describing regions, image matching**

I. INTRODUCTION

Image matching is a fundamental aspect of many problems in computer vision, including object or scene recognition, solving for 3D structure from multiple images, stereo correspondence, and motion tracking. The distinction between different matching primitives is probably the most prominent difference between the various matching algorithms. The primitives fall into two broad categories: either windows composed of grey values or features extracted in each image a priori are used in the actual matching step. The resulting algorithm usually called: Area Based Matching (ABM) and Feature Based Matching (FBM) respectively [1]. Image Matching exemplifies two fundamental problems of image matching.

- Ambiguous solutions may occur, if image matching is tackled using only local information.
- Good approximations are needed.

In most of the image matching algorithms, parameters like translation, rotation, scaling and perspective invariance must often be taken into account [1]. Moreover, the input to different sensors at the same time or input to same sensor at different times will always result in occluded output, and matching an occluded image is always been a challenging task[2]. For quick and reliable object recognition, the kind of feature to be used for matching and criterion for best matching must be known. For this, the comparative study is done which results with one optimal method i.e. feature based matching. Number of parameters has been discussed which helps in resulting with Feature Extraction.

In image matching, key region or point of interest is often used as the local feature due to its stable performance in detection and description [3]. A region feature is usually derived from a circle or ellipse with certain location and radius and is effective and efficient, compared with other types of features such as edges and contours. Therefore, region features are extensively used in real applications. Generally speaking, the framework of a region feature based image matching consists of three steps.

II. DETECTING INTEREST POINT

Interesting points are extracted from images and the region of interest is the associated circular (or elliptical) region around the interesting point. Generally, researchers use corner (Harris [5], SUSAN [6], CSS [7], etc.) or center of silent region (SIFT [8], SURF [9], DoH [10], HLSIFD [11], etc.) as the interesting point since they are stable and easy to locate and describe. The radius of the region is determined by a priori setting (Harris corner) or the region scale (scale invariant features). The total number of features detected is the minimum number of the features extracted from the matched images.

A. Corners as Interest Points

Many applications require relating two or more images in order to extract information from them. For example, if two successive images taken from a moving camera can be related, it is possible to extract information regarding the depth of objects in the environment and the speed of the camera. The brute force method of comparing every pixel in the two images is computationally prohibitive for the majority of applications. Intuitively, one can image relating two images by matching only locations in the image that are in some way interesting. Such points are referred to as interest points and are located using an interest point detector. Finding a relationship between images is then performed using only these points. This drastically reduces the required computation time.

Many different interest point detectors have been proposed with a wide range of definitions for what points in an image are interesting. Some detectors find points of high local symmetry; others find areas of highly varying texture, while others locate corner points. Corner points are interesting as they are formed from two or more edges and edges usually define the boundary between two different objects or parts of the same object.

ISSN: 2277-3754
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 1, Issue 3, March 2012

B. Requirements of a Corner Detector

It is desirable for a corner detector to satisfy a number of criteria [4]:

1. All "true corners" should be detected.
2. No "false corners" should be detected.
3. Corner points should be well localized.
4. Detector should have a high repeatability rate (good stability).
5. Detector should be robust with respect to noise.
6. Detector should be computationally efficient.

The detection of all true corners with no false corners is application (interpretation) dependent since there is no well defined definition of a grayscale corner. However, in many images the corners are intuitively clear and such images can be used to evaluate the performance of different corner detectors.

Localization refers to how accurately the position of a corner is found. This is critical in applications requiring the precise alignment of multiple images. In figure (1) -the reported position of the corner is illustrated with a red circle. The corner detector on the right has good localization whereas the corner detector on the left has poor localization. Although good localization is desirable for all applications, it is not critical for all applications (for example, an object detection algorithm may simply require the approximate location of all the object's corners).

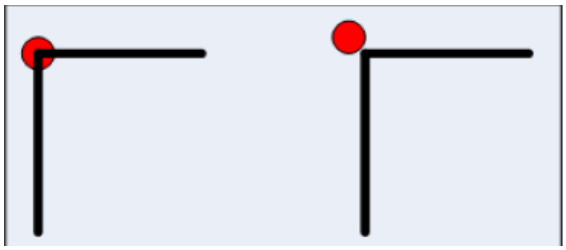

Fig. 1. Illustration of good and poor localization, respectively

The image in figure (2) contains many corner types (L-Junction, Y-Junction, T-Junction, Arrow-Junction, and X-Junction as depicted in figure (3)) and is widely used to evaluate how a corner detector responds to each of these corner types.

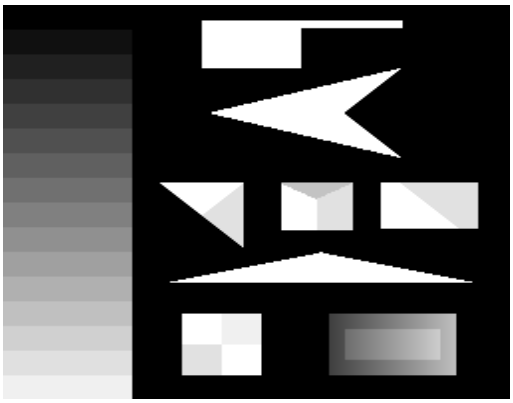

Fig. 2. A Test image containing different corner types

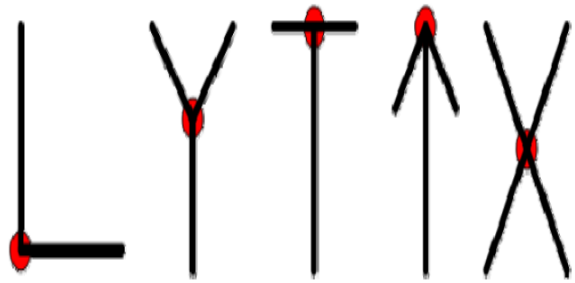

Fig. 3. Example of L-junction, Y-junction, T-junction, Arrow-junction, and X-junction corner types

C. Corner Detection

There are several corner detection algorithms but some following corner detection algorithms are considered here:

1. Moravec (1977)[22]
2. Harris/Plessey (1988)[5]
3. Trajkovic and Hedley (4-Neighbours) (1998)[23]
4. Trajkovic and Hedley (8-Neighbours) (1998)[23]

All of these algorithms follow the same general steps for detecting corners Figure 6- shows a flowchart of these steps:

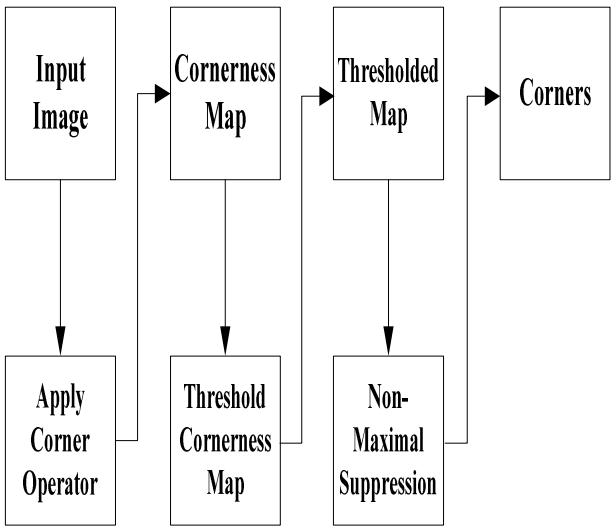

Fig. 4. Flowchart for interest point corner detectors

(i) Apply Corner Operator

This step takes as input the image and typically a few parameters required by the corner operator. For each pixel in the input image, the corner operator is applied to obtain a corner ness measure for this pixel. The corner ness measure is simply a number indicating the degree to which the corner operator believes this pixel is a corner. Interest point corner detection algorithms differ on how the corner operator makes this measurement, but all algorithms consider only pixels within a small window centered on the pixel a measurement is being made for. The output of this step is a corner ness map. Since for each pixel in the input image the corner operator is applied to obtain a corner ness measure, the corner ness map has the same dimensions as the input image and can by thought of as a processed version of the input image.

ISSN: 2277-3754
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 1, Issue 3, March 2012

(ii) Threshold Corner ness Map

Interest point corner detectors define corners as local maximum in the corner ness map. However, at this point the corner ness map will contain many local maximum that have a relatively small corner ness measure and are not true corners. To avoid reporting these points as corners, the corner ness map is typically threshold. All values in the corner ness map below the threshold are set to zero. Choosing the threshold is application dependent and often requires trial and error experimentation. The threshold must be set high enough to remove local maximum that are not true corners, but low enough to retain local maximum at true corners. In practice there is rarely a threshold value that will remove all false corners and retain all true corners so a trade-off must be made based on the requirements of the application.

(iii) Non-maximal Suppression

The threshold corner ness map contains only nonzero values around the local maximums that need to be marked as corner points. To locate the local maxima, non-maximal suppression is applied. For each point in the threshold corner ness map, non-maximal suppression sets the corner ness measure for this point to zero if its corner ness measure is not larger than the corner ness measure of all points within a certain distance. After non-maximal suppression is applied, the corners are simply the non-zero points remaining in the corner ness map.

D. Corner Detection method

In literature, there is several Corner detection methods are proposed but some corner detection methods are discussed in this section-

The Curvature Scale Space (CSS) [4] Operator detects corners by directly looking for local maxima of absolute curvature. That is, it detects corners using the intuitive notion of locating when the contour of an object makes a sharp turn. The following is an outline of the steps involved in the CSS corner detector:

1. Extract the edge contours from the input image using any good edge detector (e.g. Canny).
2. Fill small gaps in edge contours. When the gap forms a T-junction, mark it as a T-corner.
3. Compute curvature on the edge contours at a high scale.
4. The corner points are defined as the maxima of absolute curvature that are above a threshold value.
5. Track the corners through multiple lower scales to improve localization.
6. Compare T-corners to the other corners found using the CSS procedure and remove very close corners.

Experimental results indicate approximately 80% of the time is spent performing edge detection. Applications requiring faster corner detection can trade off other performance criteria for speed by using a simpler corner detector. Steps 2 and 6 concern themselves with T-junctions in order to avoid assigning two corners to pixels which are close together. As illustrated in Figure 4, as the contour is being followed it will pass by a T-junction twice and thus come across the same corner twice.

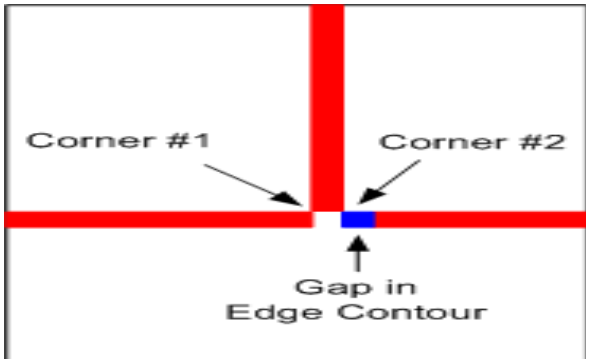

Fig. 5. CSS operator detecting the same corner twice at a T-junction

SUSAN (Smallest Univalue Segment Assimilating Nucleus):A novel corner detector algorithm based on brightness comparisons within a circular mask is proposed by Smith and Brady. SUSAN (Smallest Univalue Segment Assimilating Nucleus) [6] assumes that within a relatively small circular region pixel belonging to a given object will have relatively uniform brightness. The algorithm computes the number of pixels with similar brightness to the pixel at the center of the mask (the nucleus of the mask). These pixels are called the USAN (Univalue Segment Assimilating Nucleus) of the mask. Corners are detected by applying the mask to all pixels in the image and then finding the local minima in this new USAN map. Figure 5 illustrates this circular mask applied to different positions of a black rectangle with the USAN shown in red. Notice that the USAN becomes smaller as it approaches an edge and this reduction is stronger at corners. SUSAN can thus be used for both line and edge detection. This corner detector is robust to noise (note that no spatial derivatives are computed), fast to compute, but only has an average repeatability rate.

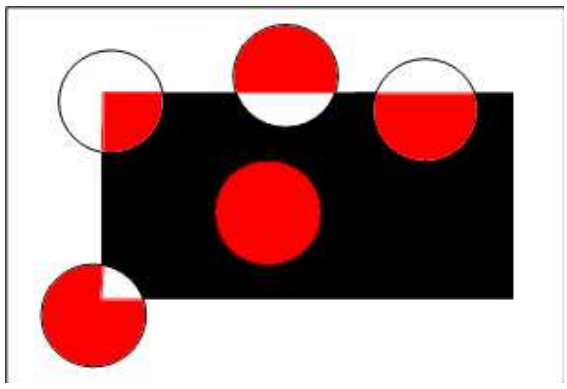

Fig.6. USAN for different circular masks on a uniform rectangle

Harris Corner Detector: The Harris corner detector, proposed by Harris and Stephens [5], is based on the second moment matrix, also called the auto-correlation matrix. This matrix describes the gradient distribution in a local neighborhood of a point.

ISSN: 2277-3754
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 1, Issue 3, March 2012

$$\mu(x,\sigma_I,\sigma_D) = \sigma_D^2 g(x,\sigma_I) * \begin{pmatrix} I_x^2(x,\sigma_D) & I_x(x,\sigma_D)I_y(x,\sigma_D) \\ I_x(x,\sigma_D)I_y(x,\sigma_D) & I_y^2(x,\sigma_D) \end{pmatrix}$$

$$I_x(x,\sigma_D) = \frac{\partial}{\partial_x} g(\sigma_D) * I(x)$$

$$g(\sigma) = \frac{1}{2\pi\sigma^2} e^{\frac{-x^2+y^2}{2\sigma^2}}$$

The local image derivatives are computed with Gaussian kernels of the differentiation scale σ_D. The derivatives are then averaged in the neighborhood of the point by smoothing with a Gaussian window of the integration scale σ_I.

The eigenvalues of this matrix represent the signal changes in two orthogonal directions in a neighborhood around the point. Corners can be found at locations in the image for which the image signal varies significantly in both directions, hence, for which both eigenvalues λ_1, λ_2 are large. Different measurements for corner ness were proposed and we compare the following from Harris [5], Tomasi [12] and Noble [13]:

$$C_{Harris} = \det(\mu) - h.trace^2(\mu)$$

$$C_{Tomasi} = \min(\lambda_1,\lambda_2)$$

$$C_{Noble} = \frac{\det(\mu)}{\text{trace}(\mu) + \grave{o}}$$

In the next step, local maxima of the corner ness function are extracted within a neighborhood of window size w and small corners are rejected using a threshold. This threshold is defined as the highest corner ness found in the image multiplied with a constant factor q. The Harris corner detector is not invariant to scale changes. Mikolajczyk and Schmid [14] proposed the Harris-Laplace detector that combines the Harris detector with automatic scale selection [15] to obtain a scale invariant detector. At first, a scale-space representation for the corner function is built for scales

$$\sigma_n = \xi^n \sigma_0$$

Where ξ is the scale factor between successive levels and σ_0 is the initial scale. The matrix $\mu(x; \sigma_n)$ is computed with integration scale $\sigma_I = \sigma_n$ and differentiation scale $\sigma_D = s\sigma_n$, where s is a constant factor. At each level initial interest points are extracted the same way as the Harris detector using the parameters w and q. Local maxima selection is also done including the upper and lower scale using a window size w_s. To determine scale and location of the interest points, it is verified for each of the initial points whether the scale-normalized Laplacian operator attains a maximum at the scale of the point, that is, the response is lower for the finer and the coarser scale. The scale-normalized Laplacian operator is computed as follows:

$$\Delta_n(x,\sigma_n) = \sigma_n^2 |L_{xx}(x,\sigma_n) + L_{yy}(x,\sigma_n)|$$

Where Lxx; Lyy are second partial derivatives with respect to x and y, respectively. Points for which the Laplacian attains no extreme or the response is below a threshold are rejected. This threshold is defined as the highest response multiplied with a constant factor qs.

Center-Surround Extremas (CenSurE): The Harris-Laplace and Hessian-Laplace detectors described earlier require computing the Hessian/Harris measure at all locations and all scales, and additionally calculating the Laplacian at all scales where there are peaks in the corner detector.

Obviously, this strategy is computationally expensive. The Center-Surround Extremas (CenSurE) detector [16] adopts this strategy; however, using approximations it shows real-time potential. At first, a simplified center-surround filter is computed at all locations and scales and then local extrema in a neighborhood are found. In a final step, these extrema are filtered by computing the Harris measure and eliminating those with a weak corner response.

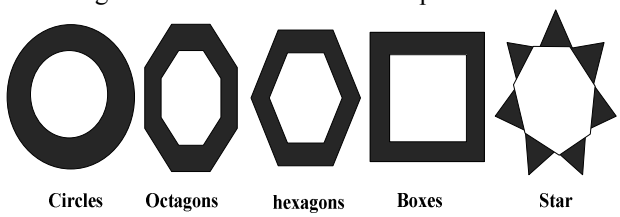

Fig.7. Center-Surround bi-level filters approximating the Laplacian

While the SIFT-Detector approximates the Laplacian with DoG, CenSurE uses even simpler approximations: Bi-level center-surround filters, that is, they multiply the image value by either 1 or -1. The tradeoff is to design the filter in such a way that it is both faithful to the Laplacian and fast to compute with integral images. Figure 6 shows a bi-level Laplacian of Gaussian and some examples of approximations that can be used in conjunction with integral images. The octagons, hexagons and boxes were proposed by Agrawal et al. [16]. We use the star-version, which is also referred to as the Star Key-point Detector [17]. The circle shape is approximated with rotated squares. For fast computation a modified version of integral images is used that can handle slanted integral images, controlled by a parameterα:

$$I_\alpha(x,y) = \sum_{j=0}^{y} \sum_{i=0}^{x+\alpha(y-j)} I(i,j)$$

After the computation of the responses up to size s is done, non-maxima suppression is performed in a $w \times w \times w$ neighborhood. Thereby, the magnitude of the responses is taken as an indication of the strength of the feature for being stable. Furthermore, weak features are discarded using a threshold tr for the response. To filter out features that lie along an edge or line, the second moment matrix of the response function at the particular scale is used:

$$H = \begin{pmatrix} L\sum_{x}^{2} \square & \Sigma L_x L_y \\ \Sigma L_x L_y & L\sum_{y}^{2} \square \end{pmatrix}$$

Where Lx, Ly are the derivatives of the response function along x and y within a circle around the feature point with radius dependent on the scale. Feature points for which the following inequality holds are discarded:

$$trace^2(H) \geq t_1.\det(H)$$

ISSN: 2277-3754
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 1, Issue 3, March 2012

This is also done for the scale of the feature point with other parameter ts.

III. DESCRIBING FEATURE

Color, structure, and texture are widely used to describe images in the recent literature. Descriptors with edge orientation information (SIFT and HOG) are also very popular since they are more robust to scale, blur, and rotation.

A. SIFT-Descriptor

The SIFT-Descriptor, analysis the gradient magnitudes and orientations to form a feature vector [7]. Therefore, the gradient magnitudes and orientations are sampled around the interest point location using the scale of the interest point to select the level of Gaussian blur for the image. The magnitudes are weighted by a Gaussian window with σ equal to one half the width of the descriptor window. These samples are then accumulated into orientation histograms (with eight bins) summarizing the contents over 4×4 sub-regions. The feature vector contains the values of all orientation histograms entries. With a descriptor window size of 16×16 samples leading to 16 sub-regions the resulting feature vector have 16.8 = 128 elements. The construction is schematically visualized in figure (7).

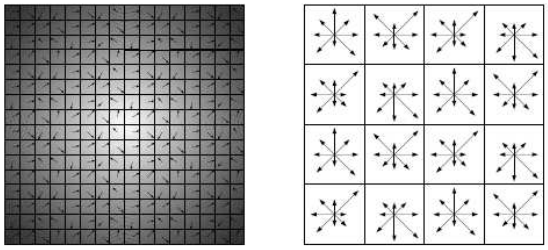

Fig. 8. Schematic construction of SIFT-Descriptor

To achieve orientation invariance, the coordinates of the descriptor window and the gradient orientations are rotated relative to the interest point orientation. Finally, the feature vector is modified to reduce illumination effects. First it is normalized to unit length in order to enhance invariance to affine changes in illumination. To reduce the effects of non-linear illumination a threshold is applied such that no value in the feature vector is larger than 0.2 and the vector is again normalized.

B. SURF-Descriptor

The SURF detector-descriptor scheme was proposed by Bay et al. [18]. The SURF Descriptor describes how the pixel intensities are distributed within a scale dependent neighborhood of an interest point. To achieve scale invariant results all calculations are based on measurements relative to the detected scaleσ.To extract the SURF-Descriptor, the first step is to construct a square window of size 20σ around the interest point oriented along the dominant direction, such that all subsequent calculations are relative to this direction. The window is divided into 16 regular sub-regions and within each of these sub-regions Haar wavelets of size 2σ are calculated for 25 regularly distributed sample points. Then for each sub-region the four values are computed,

$$\sum d_x, \sum d_y, \sum |d_x|, \sum |d_y|$$

Where dx and dy refer to the Haar wavelet responses in horizontal and vertical directions in relation to the dominant orientation. This leads to an overall vector of length 4 * 16 = 64. The resulting SURF-Descriptor is invariant to rotation, scale, brightness and, after reduction to unit length, to contrast.

The original SURF-Descriptor [18] weights the Haar responses with a Gaussian centered at the feature point the same way as the SIFT-Descriptor does (see figure (7) left). This implementation instead uses the modified descriptor proposed by Agrawal et al. [16]. To avoid boundary effects in which the descriptor abruptly changes as a sample shifts smoothly from being within one histogram to another or from one orientation to another, each boundary in the descriptor has a padding of 2σ. Therefore, the region size is increased from 20σ to 24σ and the Haar wavelet responses are computed for 9 × 9 sub-regions. For each sub-region the values are weighted with a Gaussian with σ = 2:5 centered on the center of the sub-region and another Gaussian weighting with σ= 1:5 centered at the feature point is applied on the descriptors of the sub-regions (see figure (7) right). The feature vector is then normalized like the original SURF-Descriptor. The overlap allows each sub-region to work on a larger area so samples that get shifted around are more likely to still leave a signature in the correct sub-region vectors. Likewise, the sub-region Gaussian weighting means that samples near borders that get shifted out of a sub-region have less impact on the sub-region descriptor vector.

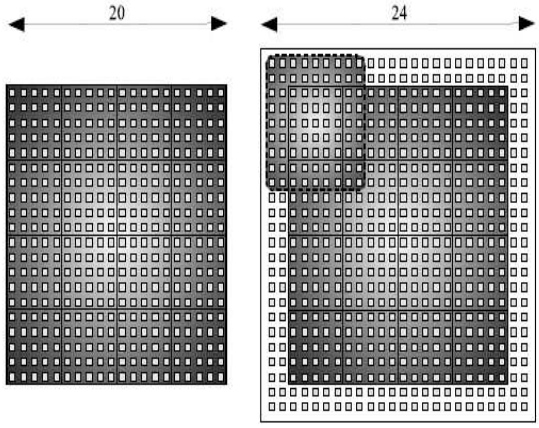

Fig. 9. Schematic construction of the original SURF- Descriptor (left) and the modified version (right).

IV. MATCHING FEATURE

In an image matching framework, key-points are detected from database and query images. Then, feature descriptors are calculated for every key-point. Matching between two descriptors is usually evaluated using the L2 distance. SIFT is generally regarded as one of the best feature extraction algorithms for being robust against many image deformations [15]. Some methods are discussed in this section.

ISSN: 2277-3754
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 1, Issue 3, March 2012

A. *Image Matching Based on Morphology Description Matrix and Genetic algorithm*

This algorithm [20] is basically based in combining genetic algorithm with morphology description matrix is presented to speed up image matching, and the precision and robust of matching for the nonergodic search characteristic of genetic algorithm is utilized, the global optimization is improved rapidly.

The image preprocessing is mainly to detect image edge and get binary image. The scale of filters is a difficult problem for edge detection. The bigger is the scale, the better is the effect of edge detection; but the more inaccurate is the edge location. In general case, we use global searching strategy to match images, but a main drawback of which is heavy computation and can not meet the real-time request. This paper presents a new fast and effective search strategy which combines the similarity degree of morphology description matrix and genetic algorithm. In order to realize image matching by genetic algorithm, the fitness function must be reasonably designed. Considering that the satisfying conditions of fitness function are single value, continuous, nonnegative and maximization, the matching fitness function in this paper is defined as:

$$f = \frac{1}{1 + (S_{ij}^1, S_{ij}^2)}$$

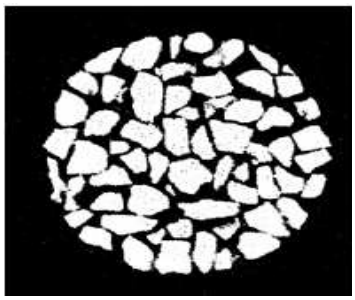

Fig. 10. Query image

Fig. 11. Shape features retrieval results

Fig 12. Shape features retrieval results

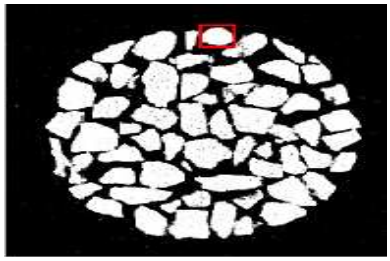

Fig 13. Shape features retrieval results

The experimental results show that the algorithm improves both the matching accuracy and matching speed.

B. *Optimized Sift Image Matching Algorithm*

This algorithm [21] can dispose of matching problem with translation, rotation and affine distortion between images and to a certain extent is with more stable feature matching ability of images which are shot from random different angles. But its algorithm is complicated and computation time is long. Method of precision orientation key-pots SIFT-based and image matching algorithm are analyzed, Euclidean distance is replaced by the linear combination of city-block distance and chessboard distance in computing process in this algorithm. Basically two concepts are used:

Detection of scale-space extrema is the first stage of key-point detection. It is implemented efficiently by using a difference-of-Gaussian function to identify potential interest points that are invariant to scale and orientation. Another is Analysis and Optimization of Key-point Orientation accurately- at each candidate location, a detailed model is fit to determine location and scale. Key-points are selected based on measures of their stability. Located key-points accurately at the location and scale of the central sample point uses the Taylor expansion (up to the quadratic terms) of the scale-space function $D(x, y, \sigma)$, shifted so that the origin is at the sample point $D(x_0, y_0, \sigma_0)$.

$$D(x, y, \sigma) = D(x_0, y_0, \sigma_0) + \frac{\partial D^T}{\partial X} X + \frac{1}{2} X^T \frac{\partial^2 D}{\partial X^2} X$$

The results are shown of two images matching, which screened in the different zoom scale, lighting conditions and rotate angle.

C. *Robust Image Matching Algorithm*

RIMA [2] is low level feature based method, in which edge points or low level feature points are extracted from digital images (using any suitable edge extraction scheme), converted to binary images, which are distance transformed, and then distance transform is used for matching. The distance transform of template is superimposed on the distance transform of the model and values are subtracted pixel wise and matching is found as per the metric. RIMA propose few matching measures: ranked highest numbers of zeros, range, minimum average and RMS value. RIMA is fast, robust, capable of matching the occluded images, independent of rotation, scale and perspective invariance. After conducted various experiments, but only one result is

ISSN: 2277-3754
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 1, Issue 3, March 2012

shown here. The images of different sizes having 8 bit gray level were used. All the images (reference/template) were applied soble operator for edge detection. After finding the DT image, templates were matched to the reference as per the matching measures.

Figure (14) is a 487x204, 8 bit gray level reference image. Figure 7 is a 108x38, 8 bit gray level template image. Template is rotated and perturbed with noise. Figure (15) and figure (17) are edge images of reference and template images, respectively. These edge images are then converted to 3-4 DT images, applied as input to RIMA, and translation parameters are found as per matching measure. Matching measure three was used in this example. It took about 168 seconds for producing the match including about 2-3 minutes DT computation time. For real time matching, DT of reference can be computed off shelf, and more that 50% computation time is saved.

Fig. 14. Gray Level reference Image

Fig. 15. Edge Reference Image

Fig. 16. Gray Level Templates

Fig. 17. Edge Template

V. CONCLUSION

In this survey, the aim has been to investigate and discuss different traditional and popular image matching Algorithms and discussed image matching frameworks- Detect-Describe- Matching (DDM) in details. Fundamental properties and methodologies of different techniques have been highlighted. The limitation and probable remedy is discussed in short. Although numbers of techniques are available, each technique works on specific concept hence it is important which image matching techniques should be used as per application domain. With this survey we try to discuss use of image matching that is a central problem in the pattern recognition, image analysis and computer vision. The exact position of objects, units and features related to image must be known.

REFERENCES

[1] Geetanjali Babbar, Punam Bajaj, Anu Chawla& Monika Gogna, "Comparative Study of Image Matching Algorithms." International Journal of Information Technology and Knowledge Management, Vol. 2, No. (2), pp. 337-339, 2010.

[2] Abdul Ghafoor, Rao Naveed Iqbal, and Shoab Khan, "Robust image matching algorithm", EC-VIP-MC 2003,4th EURASIP Conference focused on Video/Image Processing and Multimedia Communications, 2-5 July 2003, Zagreb, Croatia.

[3] Yinan Yu, Kaiqi Huang, Wei Chen, "A Novel Algorithm for View and Illumination Invariant Image Matching". IEEE TRANSACTIONS ON IMAGE PROCESSING, Vol.. 21, NO.(1), 2012.

[4] Farzin Mokhtarian and Riku Suomela. "Curvature Scale Space Based Image Corner Detection".

[5] C. Harris and M.J. Stephens, "A combined corner and edge detector". In Proceedings of the Alvey Vision Conference, pp. 147–152, 1988.

[6] S.M. Smith and M. Brady, "SUSAN – A New Approach to Low Level Image Processing". International Journal of Computer Vision, Vol. 23(1), pp. 45-78, 1997.

[7] D.G. Lowe," Distinctive image features from scale-invariant key points". International Journal of Computer Vision, Vol.60 (2): pp.91–110, 2004.

[8] F. Mokhtarian and R. Suomela, "Robust image corner detection through curvature scale space," IEEE Trans. Pattern Anal. Mach. Intell., Vol. 20, No. (12), pp. 1376–1381, 1998.

[9] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool, "Speeded-up robust features (SURF)" Computer. Vis. Image Understand., vol. 110, no. (3), pp. 346–359, 2008.

[10] T. Lindeberg, "Scale-Space Theory in Computer Vision". Norwell, MA: Kluwer, 1994.

[11] Y. Yu, K. Huang, and T. Tan, "A Harris-like scale invariant feature detector," in Proc. Asian Conf. Computer. Vis., pp. 586–595, 2009.

[12] C. Tomasi and T. Kanade. "Shape and motion from image streams: a factorization method - part 3 detection and tracking of point features". Technical Report CMU-CS-91-132, Computer Science Department, Pittsburgh, PA, April 1991.

[13] J.A. Noble, "Finding corners". Image Vision and Computing, Vol.6, pp. 121–128, 1988.

[14] K. Mikolajczyk and C. Schmid, "Indexing based on scale invariant interest points". In Proceedings of the IEEE Computer Society International Conference on Computer Vision, pp. 525–531, 2001.

[15] T. Lindeberg, "Feature detection with automatic scale selection". International Journal of Computer Vision, Vol. 30(2), pp. 79–116, 1998.

ISSN: 2277-3754
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 1, Issue 3, March 2012

[16] M. Agrawal, K. Konolige, and M.R. Blas, "Censure: Center surrounds extremas for real-time feature detection and matching". In Proceedings of the European Conference on Computer Vision, pp. 102–115, 2008.

[17] G. Bradski, T. Darrell, I. Essa, J. Malik, P. Perona, S. Sclaroff, and C. Tomasi. OpenCV: Open Computer Vision Library, 2010.

[18] H. Bay, A. Ess, T. Tuytelaars, and L. van Gool, "Speeded-up robust features (surf"). Computer Vision and Image Understanding, pp. 346–359, 2008.

[19] K. Mikolajczyk and C. Schmid, "Performance evaluation of local descriptors", IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 27, pp. 1615–1630, 2005.

[20] Wenli Yang, Zhiyuan Zeng "Application Research of Image Matching Based on Morphology Description Matrix and Genetic algorithm", International Conference on Computational Intelligence and Natural Computing Vol. 1, pp. 159-161, 2009.

[21] Xiaohua Wang Weiping Fu, Xiaohua Wang." Optimized SIFT Image Matching Algorithm". Proceedings of the IEEE International Conference on Automation and Logistics Qingdao, pp. 843 - 847 China September 2008.

[22] H. P. Moravec. "Towards Automatic Visual Obstacle Avoidance". Proc. 5th International Joint Conference on Artificial Intelligence, pp. 584, 1977.

[23] M. Trajkovic and M. Hedley. "Fast Corner Detection". Image and Vision Computing, Vol. 16(2), pp. 75-87, 1998.