ROBUST SYNTHETIC BASIS FEATURE DESCRIPTOR

Lindsey Raven, Dah-Jye Lee, and Alok Desai

Department of Electrical and Computer Engineering
Brigham Young University
Provo, Utah, USA

ABSTRACT

Feature detection and matching is an important step in many object detection and tracking algorithms. This paper discusses methods to improve upon our previous work on the SYnthetic BAsis feature descriptor (SYBA) algorithm, which describes and compares image features in an efficient and discrete manner. SYBA utilizes synthetic basis images overlaid on a feature region of interest (FRI) to generate binary numbers that uniquely describe the feature contained within the image region. These binary numbers are then used to compare against feature values in subsequent images for matching. However, in a non-ideal environment the accuracy of the feature matching suffers due to variations in image scale, and rotation. This paper introduces a new version of SYBA which processes FRI's such that the descriptions generated with SYBA for feature matching are rotation and scale invariant.

Index Terms— Feature Descriptor, Feature Matching, Synthetic Basis Image.

1. INTRODUCTION

Image correspondence can be found by matching basic components, called features, between two images. Once the correspondence between two images has been found, it can then be applied towards many applications such as image object recognition, object tracking, and motion tracking. In short, finding image correspondence has become a key step in many computer vision applications. Because finding image correspondence is crucial, many image feature detector and descriptor algorithms have been developed.

For most algorithms, the act of finding image correspondence can be broken down into three fundamental steps: feature detection, description, and matching. Features are detected and then described in a unique manner such that the descriptions can be used to match features between two images. Modern algorithms must also consider variations of features between images. Some methods of feature variation can include changes in feature scale, angle, blurring, and lighting. Thus, to maximize the number of correctly matched features, algorithms must be invariant to these variations. Some well-known robust feature description and matching algorithms to date include the Scale Invariant Feature Transform (SIFT) [1] and the Speeded-Up Robust Features (SURF) [2].

To achieve feature scale and blurring invariance, SIFT generates multiple copies of the captured image at different scales and blurriness. SIFT utilizes the difference of Laplacian of Gaussians [3] to detect the feature locations in the images. For each detected corner, SIFT computes the gradients of the feature region of interest (FRI) to generate a feature descriptor vector. The dominant gradient of the FRI is then subtracted from the gradient description to make the description rotation invariant. Differences in gradient vectors between descriptors are compared for final matching. SURF uses the same methodology as SIFT to achieve robustness. However, SURF utilizes Gaussian kernels to generate Gaussian blurred images which reduces the number of generated images compared to SIFT. Using integral images, the Gaussian images are also generated quicker compared to SIFT. SURF uses a similar methodology as SIFT by computing the gradients of the FRI to generate a descriptor vector.

Both SIFT and SURF perform well on intensity images with rotation and scaling variations, but their complexity and robustness require extensive computations and storage. This can make these algorithms undesirable for embedded applications which have limited storage, or certain real-time applications. To counteract these limitations, some binary feature descriptor algorithms have been developed with compact size and lower computational requirements. Some examples include the Binary Robust Independent Elementary Features (BRIEF) [4] and the Binary Robust Invariant Scalable Key-points (BRISK) [5] algorithms. BRIEF comprises of a binary descriptor which contains the results of simple image intensity comparisons at random pre-determined pixel locations. BRISK computes brightness comparisons at pixel locations determined from using a configurable circular sampling pattern. These comparisons are then used to form a descriptor. Since both BRIEF and BRISK utilize random sampling, less bits are needed to describe the whole image. This reduced count also results in less overall computations and an increase in overall execution speed. However, despite improvements upon these algorithms such as ORB (rBRIEF) [6], most of these binary descriptor algorithms suffer from image variations detailed prior and can have less accurate matches and a reduced match count compared to SIFT and SURF.

To increase the number of accurate matched features for binary description algorithms, the SYnthetic BAsis (SYBA) [7] descriptor algorithm was created. SYBA requires less computational complexity using synthetic basis images and is still able to provide accurately matched feature points. In our previous work, we have made in depth comparisons between SYBA and other feature descriptor algorithms mentioned prior demonstrating its improvements upon feature matching precision [7]. However, comparable to the other binary feature descriptor algorithms, the number of matched points SYBA generates, as well as feature point matching accuracy, suffers under large amounts of image variation. Proposed in this paper is an improved version of SYBA, robust-SYBA (rSYBA), which maintains the reduced storage space and complexity compared to SIFT and SURF while increasing the number of matched feature points and maintaining feature point matching precision.

The SYBA descriptor and its limitations are outlined in Section 2. Section 3 introduces the rSYBA algorithm. Experimental results are presented in Section 4. Conclusions are shown in Section 5.

2. SYBA DESCRIPTOR

SYBA focusses on using synthetic basis images (SBIs) overlaid over an FRI to generate binary numbers that uniquely describe the feature contained within the image region. SYBA compares and matches these values to find image correspondence.

978-1-5090-2175-8/17/$31.00 ©2017 IEEE

2.1. Compressed sensing theory

The concept behind compressed sensing theory is that you can achieve an accurate representation of a signal or image through sub-sampling. The basis of this idea was proved by Emmanuel Candes, Terence Tao, and David Donoho in 2004 [8]. The contents of this research demonstrate that a signal may be reconstructed with fewer samples than Nyquist-Shannon sampling theorem [9]. This concept can be applied to describing a FRI: by sub-sampling the FRI using synthetic basis functions, as introduced by Anderson in [10], we can obtain an accurate descriptor value of that region. This concept is the basis of SYBA.

2.2. SYBA description

To start, the FRI is binarized based on the average intensity of the region. This helps to make the algorithm lighting invariant. The pixels contained in the FRI can then be described by overlaying an SBI over the region. The SBI acts as the synthetic basis function described in [10]. An SBI is a binary image the same size as the region being described. The number of black pixels in the SBI is equivalent to half the region size plus one (assuming odd region dimensions). The location of the black and white pixels are randomy generated. The synthetic basis image is then overlaid on the FRI to generate a "similarity" value. Wherever a black pixel on the synthetic basis image coincides with the binarized FRI causes the value to be incremented. The number of corresponding locations is tallied to generate a descriptor value corresponding to the SBI used. The descriptor values for each SBI are then combined to form the overall FRI descriptor value. A visual of how the descriptor value is generated can be seen in Fig 1.

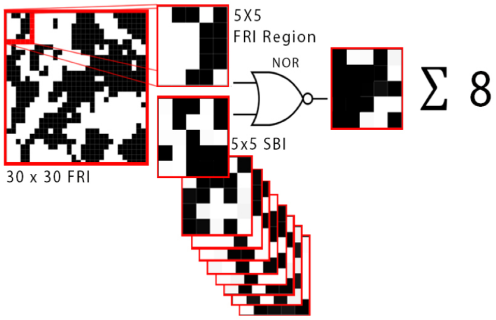

Fig. 1. SYBA Similarity Measure: A 30×30 FRI is divided into 36 5×5 sub regions. Nine 5×5 SBIs are subjected to NOR logic to find coinciding black pixels. The sum is taken of the output to find the descriptor value, in this example the sum is 8.

The number of SBI's required to accurately describe an FRI, as shown in [10], is:

$$M = C\left(K \ln \frac{N}{K}\right) \tag{1}$$

where N is the size of the region. K is equivalent to the number of white pixels. C represents rounding up to the nearest integer, and M is the number of random patterns required to accurately locate all the black pixels.

In the example shown in Figure 1. there are thirty-six, 5×5 sub-regions in a 30×30 FRI. Nine SBIs are needed to accurately describe a 5×5 sub-region using (1), with 4 bits being needed to represent the hit count per SBI (due to 13 being the max hit count, where a hit represents overlapping black pixels between the two images). Thus, to accurately describe the 30×30 region 4×9×36 bits are needed, which equates to 1296 bits per FRI. This count is less than the number of bits required to describe the gradients used in SIFT and SURF which requires 256 bytes [7]. The example shown in Fig. 1 also demonstrates the ideal FRI and SBI dimensions to produce the best matching accuracy [7].

2.3. SYBA feature matching

Euclidean distances are often used as comparison metrics but require complicated operations such as multiplication and square root. Therefore, we use the L1 norm is used to evaluate how well two features match. The L1 norm is computed by,

$$d = \sum_{i=1,j=1}^{n} |x_i - y_j|, \tag{2}$$

where x_i is the descriptor of a feature point in the first image, y_j is the descriptor of a feature point in the second image, and d is the L1 norm. The value n represents the total number of regions used for feature description. For a 5×5 grid and a 30×30 FRI, n = 36. The smallest value of d represents the least difference in descriptor values and the best feature match. Here is an example of the computation of d as shown in [7].

$$\begin{array}{r} 5\ 4\ 6\ 6\ 6\ 4\ 5\ 6\ 7 \ldots 2\ 5\ 0\ 0\ 0\ 0\ 0\ 1\ 1 \\ 5\ 3\ 7\ 6\ 6\ 4\ 5\ 5\ 7 \ldots 1\ 5\ 0\ 0\ 1\ 0\ 0\ 1\ 1 \\ \hline \sum(0\ 1\ 1\ 0\ 0\ 0\ 0\ 1\ 0 \ldots 1\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0) = 5 \end{array}$$

At the feature matching stage, each descriptor in the first image goes through this computation to compare with all the descriptors in the second image. Matches are then found at the end of this step.

Of the L1 values computed against the features in the second image, the min is found resulting in the best matched feature for the feature in the first image. However, for the pair to be uniquely matched the computed best match for the corresponding feature in the second image must also match back to the feature in the first image. In other words, the features must be each-others' best match. Otherwise, no match is made for the feature. This results in more accurate matches than those found with BRIEF, BRISK, and ORB (rBRIEF) as shown in [7]. The matched features are also ranked based on the difference in L1 norm values between the best match and the second-best match against the second image. The larger the distance the more likely the match was a unique match. Threshold can be put into place to reject matched pairs with a small difference in L1 norm matching distances between the second-best matches. This results in less matched pairs but higher matched precision.

2.4. SYBA limitations

While SYBA results in less space and complexity when describing features while maintaining accurately matched features, the number of matched features is less than that of SIFT and SURF. This is because SYBA suffers from variations between images. Should the image rotate or scale differently between image one and two, the FRI that the synthetic basis images overlays over would be fundamentally different, resulting in high L1 norms. To increase the number of matches, the FRI that the synthetic basis images overlay over must be normalized such that there are no differences in angle or scale of the feature contained therein.

3. Robust SYBA Implementation

3.1. Algorithm flow

Typically, SYBA is used in conjunction with a feature detector algorithm which will give the FRI locations contained in the image.

Given the locations, SYBA extracts a 30x30 region around the feature, generating a FRI which is then used to create the descriptor value. rSYBA uses the same flow as SYBA, however instead the FRI is pre-processed such that the regions used to generate FRI are scaled and rotated. The flow can be shown in Fig 2.

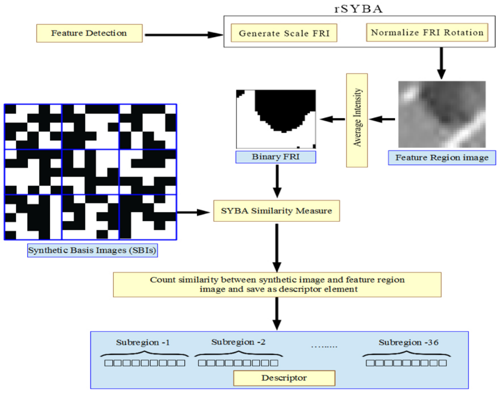

Fig. 2. SYBA algorithm flow including rSYBA logic

3.2. Scale invariance

To achieve scale invariance, similar methods to SIFT and SURF are employed. The input image is re-scaled to different scale factors while the FRI dimensions and location within the image remain constant. This will result in several FRIs corresponding to the same feature point location. A descriptor value is generated for each of the generated FRIs. Matching then selects the best match from the scaled features. A visual of this process can be seen in Fig. 3. Most modern cameras operate at 60~120 frames per second. The scale difference that can occur between two consecutive frames is usually small, thus for rSYBA the scale factors for re-scaling range from .8 to 1.2 with .1 scaling intervals. This results in 5 FRIs being generated for a single detected FRI. Only three of them are shown in Fig. 3. Depending on the application, the range of the scale factor and intervals can be adjusted for optimization.

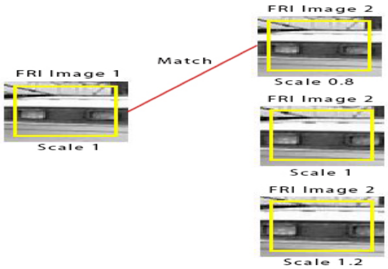

Fig. 3. Illustration of how generating scaled FRIs achieves scale invariance

3.3. Rotation invariance

SIFT and SURF both utilize image gradients and orientations to describe the FRI's. Because of this, they can calculate the dominant gradient orientation of the region and normalize the descriptor value by subtracting that dominant orientation from the descriptor. This makes the description rotation invariant.

For rSYBA, the dominant gradient orientation for the FRI is computed using similar methods to SIFT [1]. A histogram is generated with each element corresponding to a range of gradient orientations. The range for the elements can vary depending on the application; in general use cases a range of 10 degrees is sufficient. The gradient and orientation is then computed for each pixel value contained in the FRI. The corresponding orientation element in the histogram is then incremented an amount proportional to the gradient magnitude. After computing the gradient orientation, and corresponding histogram for the FRI, the max of the histogram gives the dominant gradient orientation for the FRI.

Using this orientation, the FRI is back-rotated by the orientation found. Surrounding pixels are included around the FRI such that when the region is rotated no information is lost. Rotation invariance is achieved through this method as the matching FRI's are rotated to approximately the same normalized orientation.

4. Results

4.1. Precision calculation methods

Once the matched pairs between two consecutive images have been found, the results are input into a RAndom SAmple Consensus (RANSAC) [11] to generate a homography matrix *H*. The homography matrix can be used to find:

$$\boldsymbol{p_2} = \boldsymbol{H} * \boldsymbol{p_1}, \tag{3}$$

where p2 is the matched coordinate in the second image, p1 is the matched coordinate in the first image, and H is the homography matrix. If a homography is accurate, p2 should coincide with the matched point found in the second image. To determine whether a match is a correct match, the homography matrix is used to compute p2. If p2 is not within a set error range of the matched coordinate in the second image, then the matched pair is an inaccurate match. For these calculations, the error range is within 5 square pixels of the matched coordinate in the second image. This method is used to find the good matches found by SYBA, rSYBA, and mainstream algorithms.

The matching precision of each algorithm is computed to be the number of good matches divided by the number of matches found. Recall is computed by taking the number of good matches and dividing it by the number of possible matches. Recall is difficult to compute as there is no ground truth, however since the features being input into each algorithm are the same, the ground truth is the same, thus the total number of good matches is used as an accurate comparison since all algorithms use the same common denominator of total possible feature matches.

4.2. Overall results

To highlight the matching results of SYBA and rSYBA under different levels of image variations, the BYU Rotation and Scale datasets were created. For the scaling dataset, the image is scaled from .8 to 1.2 scale factor with .1 scaling intervals. For the rotation dataset, the image is rotated from 5 degrees to 15 degrees (maintaining original image dimensions through rotation). Example images contained in the set are shown in Fig. 4.

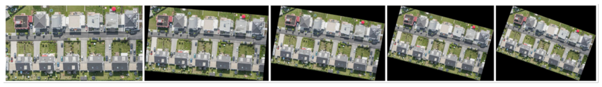

Fig. 4. First five images contained in the BYU Rotation dataset

Results were taken by matching the un-altered baseline image to each image in the dataset. Using the precision calculations detailed prior, we compute the matching precision for each corresponding number of good matches. In short, we find the

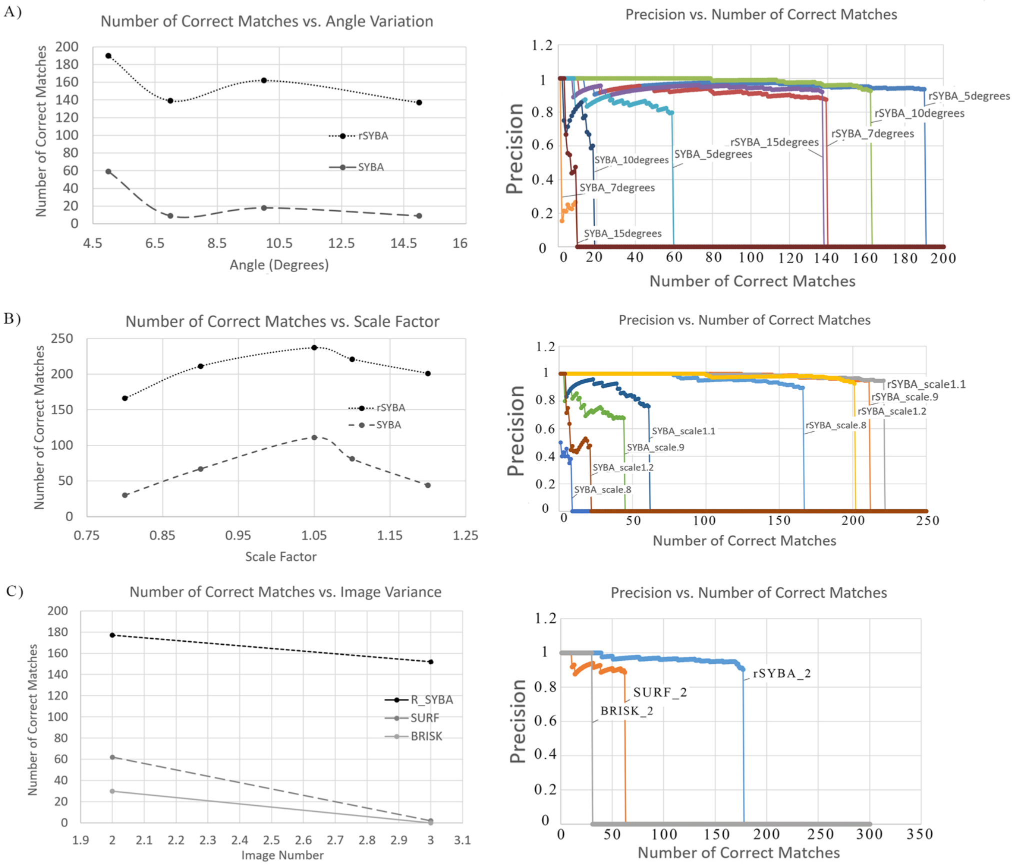

Fig. 5. a) Rotation Dataset results b) Scale Dataset results c) Oxford "Boat" Dataset Results, image 1 comparing image 2, for number of correct matches count comparing against image 3 is also included

number of matches needed to produce a certain quantity of good matches, and divide the count of good matches by the number of matches found to calculate precision. This way precision can be compared irrespective of the number of good matches the algorithm produces. The total number of good matches produced by each algorithm corresponding to each amount of image variation is also calculated. For these results, 300 features were input to SYBA and rSYBA for each image.

Results were also taken using the Oxford Affine Features dataset, which is one of the more widely used datasets for comparing results. The dataset contains sets of 6 images, with increasing variation in each consecutive image. For this research, we selected the "boat" dataset in the oxford dataset which contains images with zoom and rotation variance. We used the same methods detailed with the BYU Scaling and Rotation datasets for comparison. To demonstrate rSYBA's overall performance we compared results with Matlab implementation of BRIEF and SURF [14] using image 1 versus image 2 and 3 of the dataset. We didn't compare with more images contained in the dataset because no algorithm produced enough matches to give sufficient data for a meaningful comparison.

Scaling parameters for rSYBA could be adjusted to produce matches, however this required unrealistic scaling bounds which increased overall computation time drastically. Precision drops to zero when the algorithm could no longer match any features per its matching algorithm. Overall, results can be visualized in Fig. 5.

5. Conclusion

rSYBA produces higher precision rates and number of correct matches compared to SYBA as evidenced by the BYU Rotation and Scaling dataset results shown in Figs. 5 (a) and (b). rSYBA also competes with mainstream algorithms as shown by the Oxford dataset results in Fig. 5 (c). Despite these benefits, the argument can be made that the space benefits made with SYBA are counteracted by the number of increased generated features in rSYBA. With the advancement in Graphics Processing Unit (GPU) technologies for embedded vision applications such as NVIDIA's Jetson module, most embedded space applications can be implemented on a GPU with a significant amount of cache and main memory. By utilizing a GPU, the increase in comparisons, and thus execution time, between features due to the increase in feature count becomes negligible. This is because of the parallelization capabilities of the GPU. Another implementation option is Field Programmable Gate Arrays (FPGAs).

With these findings, rSYBA will be applied towards applications with varying degrees of image variation. SYBA will be applied to applications such as [12] and [13] to see if better results can be found. Further research will be conducted in implementing rSYBA on a GPU platform.

References

[1] D. G. Lowe, “Distinctive image features from scale-invariant keypoints,” Int. J. Comput. Vision, vol. 60, no. 2, 2004, pp. 91–110.

[2] H. Bay, T. Tuytelaars, and L. Van Gool, “SURF: speeded up robust features,” Comput. Vision-ECCV, Berlin, Heidelberg: Springer, 2006, pp. 404–417.

[3] Harris, C., and M. Stephens. "A Combined Corner and Edge Detector." *Procedings of the Alvey Vision Conference 1988* (1988): n. pag. Web.

[4] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, “BRIEF: binary robust independent elementary features,” Comput. Vision-ECCV, 2010, pp. 778-792.

[5] S. Leutenegger, M. Chli, and R. Siegwart, “BRISK: binary robust invariant scalable keypoints,” IEEE Int. Conf. Comput. Vis., 2011, pp. 2548 -2555.

[6] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, “ORB: an efficient alternative to SIFT or SURF,” IEEE Int. Conf. Comput. Vis., 2011, pp. 2564-2571.

[7] A. Desai, D. J. Lee, and D. Ventura, “An efficient feature descriptor based on synthetic basis functions and uniqueness matching strategy,” Comput. Vis. Image Und., vol. 142, 2016, pp. 37–49.

[8] E. J. Candès, J. K. Romberg, and T. Tao, “Stable signal recovery from incomplete and inaccurate measurements,” Commun. Pur. Appl. Math, vol. 59, no. 8, 2006, pp. 1207–1223.

[9] D. L. Donoho, “Compressed sensing,” IEEE T. Inform. Theory, vol. 52, no. 4, 2006, pp. 1289–1306.

[10] H. Anderson, “Both lazy and efficient: compressed sensing and applications,” Sandia National Laboratories, Albuquerque, NM, 2013, pp. 2013-7521.

[11] Fischler, Martin A., and Robert C. Bolles. "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography." *Communications of the ACM* 24.6 (1981): 381-95. Web.

[12] A. Desai, D.J. Lee, and S.H. Mody, “Automatic Motion Classification for Advanced Driver Assistance Systems,” Lecture Notes in Computer Science (LNCS), International Symposium on Visual Computing (ISVC), Part II, LNCS 9475, p. 819-829, Las Vegas, NV, U.S.A., December 14-16, 2015.

[13] A. Desai, D.J. Lee, and M. Zhang, “Using Accurate Feature Matching for Unmanned Aerial Vehicle Ground Object Tracking,” Lecture Notes in Computer Science (LNCS), International Symposium on Visual Computing (ISVC), Part I, LNCS 8887, p. 435–444, Las Vegas, NV, U.S.A., December 8-10, 2014.

[14] Matlab and Computer Vision System Toolbox Releas 2016b, The Matchworks, Inc., Nantick, Massachussetts, United States.