The Color-BRIEF Feature Descriptor

Michal Kottman*
Faculty of Informatics and Information Technologies, STU Bratislava

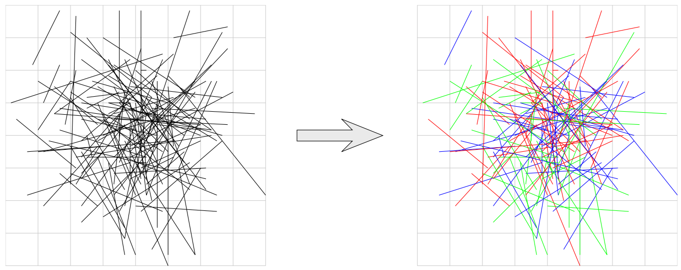

Figure 1: The proposed Color-BRIEF descriptor

CR Categories: I.4.7 [Image Processing and Computer Vision]: Feature Measurement—Feature Representation I.4.7 [Image Processing and Computer Vision]: Feature Measurement—Invariants

Keywords: augmented reality, feature description, color descriptor

1 Introduction

Feature description and matching algorithms have become important areas in the field of computer vision, with many applications in computer vision, like tracking, recognition, scene reconstruction (either 3D or panoramic), object detection and recognition and many others. Having a fast, easy to compute and small feature descriptor is essential for applications in augmented reality, especially in mobile applications, where computational power and storage space is sparse.

The main purpose of feature descriptors is to establish local correspondences between two or more images – i.e. to find the same feature in different images. This information can be used in various ways, for example to reconstruct the 3D scene from the camera view. It can further be used to augment the scene with additional information, like navigation marks, overlays and various 3D objects.

Since the appearance of the SIFT [Lowe 2004] descriptor, there is ongoing research for feature descriptors with similar or better recognition rates as SIFT. The main disadvantage of the SIFT descriptor is that it is large to store - it is a 128 dimensional vector - and relatively slow to compute. There are several techniques to manage the size of SIFT descriptors [Chandrasekhar et al. 2010], but they do not generally solve the problem of slow computation.

The SURF [Bay et al. 2006] descriptor tries to solve the computational problem by using integral images and Haar wavelets to speed up calculation. This results in much better performance, however, the resulting descriptor still has 64 floating point numbers, making storage and matching challenging on embedded and small devices, like mobile phones.

An interesting alternative to these descriptors has been presented in [Lepetit and Fua 2006; Ozuysal et al. 2010]. While SIFT calculates orientation histograms from the image, and SURF uses Haar wavelet responses, the authors use a simple comparison of the intensity of two pixels as an elementary feature. Their research has resulted in the BRIEF descriptor [Calonder et al. 2010], which is described in the Section 2. It's main advantages are the speed of computation and small resulting descriptor size.

The original BRIEF descriptor only uses gray-scale/intensity information for feature description. We describe an extended Color-BRIEF descriptor, which is based on the BRIEF descriptor and takes into consideration the RGB color information from the image. It is described in Section 3. In the evaluation in Section 4 we show that our descriptor achieves up to 2× better recognition rates, mainly in situations where the original descriptor loses performance.

2 The BRIEF Descriptor

We first describe the BRIEF descriptor. As was mentioned earlier, it uses simple pixel comparisons for feature description. The descriptor is calculated for a small patch that surrounds a keypoint. A method for selection of keypoints is not a part of the algorithm, unlike SIFT and SURF, so keypoints are acquired by another algorithm, like the Fast-Hessian of SURF, or the FAST [Rosten and Drummond 2006] algorithm.

Given a patch $\mathbf{p}$ of size $S \times S$ centered on a keypoint, a test τ between pixel intensities at two positions $\mathbf{x}, \mathbf{y}$ relative to the center of the patch is given as

$$\tau(\mathbf{p},\mathbf{x},\mathbf{y}) = \begin{cases} 1 & \text{if } \mathbf{p}(\mathbf{x}) < \mathbf{p}(\mathbf{y}) \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

The result of this test is a single bit. Given n such tests for different values of $\mathbf{x}, \mathbf{y}$, it is possible to construct a bitstring of length n from the results of these tests on a patch $\mathbf{p}$ as

*e-mail: kottman@fiit.stuba.sk

Figure 2: An example of matching pairs of keypoints between two images using out proposed Color-BRIEF descriptor.

$$f_n(\mathbf{p}) = \sum_{i=1}^{n} 2^{i-1} \tau(\mathbf{p}, \mathbf{x}_i, \mathbf{y}_i) \quad (2)$$

This resulting bitstring $f_n(\mathbf{p})$ is the BRIEF descriptor for patch $\mathbf{p}$. The positions $\mathbf{x}_i, \mathbf{y}_i$ are selected randomly from a Gaussian distribution surrounding the keypoint, and are fixed at compilation time.

To increase the invariance to noise, the image may be first preprocessed by a Gaussian filter. Alternatively, a response of a small kernel surrounding the compared pixel is calculated from an integral image, as implemented in the OpenCV library [Bradski 2000]. This decreases the influence of noise, since the response of a larger area than a single pixel is considered, and is faster to calculate than applying a Gaussian filter.

There are two advantages to using bitstring as a descriptor instead of using floating point numbers, like in SIFT and SURF. The first advantage is speed. It is very easy and fast to calculate the BRIEF descriptor from an image patch. In order to match two descriptors, the Hamming distance is used to match BRIEF descriptors, as opposed to Euclidean distance in floating point descriptors. The distance of descriptors a, b is easily calculated by counting the bits set to 1 in $a \,\mathbf{xor}\, b$, since equal bits in the **xor** operation result in 0, and different in 1. Since newer CPUs support the POPCNT instruction, which calculates the number of bits set to 1 in a register, the Hamming distance can be computed very effectively.

The second advantage of the descriptor is size. The authors have shown that in order to achieve recognition rate comparable with SURF on the same set of keypoints, only about 256 comparisons (bits) are needed, compared with 2048 bits of SURF descriptor (64×32, assuming 32-bit floats). Smaller descriptors are easier to store and transmit, and also faster to match, because smaller items have to be taken into account. An example of such matching is shown in Figure 2.

The disadvantage is that the BRIEF descriptor is not invariant to rotation and scale, however many applications do not require rotational invariance, and scale-invariance can be achieved by computing the descriptor on image pyramids constructed from the image.

3 Assuming Color - the Color-BRIEF Descriptor

The authors of the BRIEF descriptor only deal with intensity information in a gray scale image. However, color can be used to increase the photometric invariance and discriminative power of descriptors [van de Sande et al. 2010]. We therefore propose a novel color-enhanced variation of the descriptor - the Color-BRIEF descriptor.

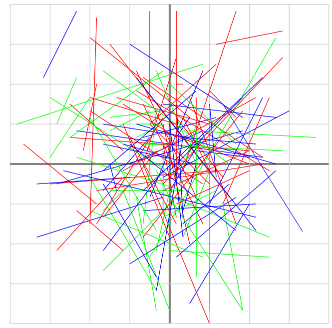

Figure 3: Visualization of comparisons used in our Color-BRIEF descriptor. The end points of lines represent the positions of two pixels, whose intensities are compared. The color of the line represents the color channel used for comparison. We use a patch of size 48×48 pixels.

We consider images in the RGB color model, which is standard for image acquisition hardware. In order to account for the color information, we extend the comparison pairs that are fixed at compile time and tests as follows:

$$(\mathbf{x}_i, \mathbf{y}_i) \rightarrow (\mathbf{x}_i, \mathbf{y}_i, \mathbf{c}_i) \quad (3)$$

$$\tau(\mathbf{p}, \mathbf{x}, \mathbf{y}, \mathbf{c}) = \begin{cases} 1 & \text{if}\, \mathbf{p_c}(\mathbf{x}) < \mathbf{p_c}(\mathbf{y}) \\ 0 & \text{otherwise} \end{cases} \quad (4)$$

Here, $\mathbf{c}_i \in \{R, G, B\}$, and it is selected randomly from a uniform distribution when generating the pairs. $\mathbf{p_c}$ is the intensity of a pixel at a given location relative to the center of the patch in a given color channel. The selection of $\mathbf{c}_i$ is fixed and stored together with $\mathbf{x}_i, \mathbf{y}_i$.

The distribution of the comparisons we use in further evaluation is visualized in Figure 3. The comparisons are centered at the keypoint location and they are performed in a 48×48 patch surrounding it.

3.1 Time and Space Complexity

The time required to calculate our descriptor is the same as for the original BRIEF descriptor, since at every comparison, only a single channel is inspected. Thus, 256-bit Color-BRIEF descriptor does the same amount of comparisons as 256-bit BRIEF descriptor. The preparation time increases, because the integral image used for smoothing has to be computed $3\times$, once for every color channel. However, since the preparation time forms only a part of the process, and the comparison calculation time remains the same, the overall speed of the descriptor is roughly $2\times$ slower, as will be shown Section 4.

The descriptor size that we use remains the same, because as we show in Section 4, it is not necessary to increase the descriptor size to increase recognition rates. Therefore our descriptor enjoys both desired qualities - fast computation, and small descriptor size.

3.2 Invariant Color Properties

Because of the use of simple pixel intensity comparisons, the BRIEF descriptor is invariant to linear changes in the lightning of the scene, because the relative intensity of the pixels does not change. With the added RGB color information, there are several changes that have to be taken into consideration [van de Sande et al. 2010]. If (R,G,B) are the values of a pixel in a reference image, and (R_2,G_2,B_2) are the values of the corresponding pixel in another image, the following changes (or their combination) can occur:

- intensity change - $(R_2,G_2,B_2) = (\alpha R, \alpha G, \alpha B)$
- intensity shift - $(R_2,G_2,B_2) = (R+o_1, G+o_1, B+o_1)$
- color change - $(R_2,G_2,B_2) = (\alpha R, \beta G, \gamma B)$
- color shift - $(R_2,G_2,B_2) = (R+o_1, G+o_2, B+o_3)$

Because the comparisons between two pixel intensities are always done in a single selected channel, their relative values keep their order under every change listed - therefore the corresponding bit is the same in both images (considering that there are only lightning and color changes). By exploiting the nature of the simple features, Color-BRIEF is a color-invariant descriptor.

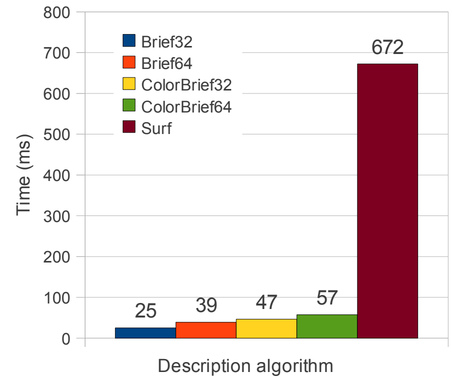

Figure 4: The time required to compute the descriptors for all keypoints detected in a single image. The time is taken averaged from all pictures in the evaluated dataset.

4 Evaluation

4.1 Evaluated Datasets and Algorithms

To evaluate the inclusion of color information we use the image sequences from a publicly available dataset [Mikolajczyk and Schmid 2005]. The same dataset was used for evaluation of the BRIEF descriptor. It consists of sequences of images of the same subject with varying conditions (listed by their name in dataset):

- *Wall* and *Graffiti* - perspective change
- *Leuven* - lightness change
- *Ubc* - JPEG compression change
- *Trees* and *Bikes* - increasing blur

The scale/rotational sequences Boat and Bark were not used, because BRIEF does not provide such invariance as it is defined.

The sequences consist of 6 images, in increasing order of difficulty. The later images are always compared with the first image, and for each such pair a ground-truth homography is provided, to help evaluate the matching of the feature description algorithm.

Because the BRIEF descriptor does not define a feature detection algorithm, we use keypoints provided by the Fast-Hessian detector defined by the SURF algorithm. To obtain comparable results, the same set of keypoints was used by every algorithm tested. The following algorithms are compared:

- SURF - as implemented in the OpenCV library
- BRIEF-32, BRIEF-64 - a 32-byte (256-bit)/512-bit BRIEF descriptor implemented in OpenCV
- ColorBRIEF-32, ColorBRIEF-64 - 256-bit/512-bit Color-BRIEF, own implementation

4.2 Evaluation Methodology

The descriptors are compared according to the recognition rate. Unlike the original paper, which used simple matching, we use the left-right consistency check when matching descriptors, and define the evaluation method as follows:

1. Use the Fast-Hessian detector to determine two sets of keypoints - k_1 in the first image and k_2 in the second image.
2. Use the homography provided in the dataset to calculate the projection of k_1 to the other image: r.
3. Calculate the descriptors d_1 for keypoints k_1 in the first image and d_2 for keypoints k_2 in second image.
4. Find the nearest neighbor (Euclidean for SURF / Hamming for the others) among d_2 for every descriptor d_1, and the also in reverse. Only consider those pairs, that both match their nearest neighbours (left-right consistency check). $N =$ number of found matches.
5. For every match, calculate the distance of the matched keypoint from k_2 with the position calculated using ground-truth homography. Let n_{ok} be the number of keypoints, that lie within 4 pixels of the calculated position.
6. The recognition rate is defined as n_{ok}/N.

Since we use the same methodology for all tested algorithms, it is possible to compare the relative differences between algorithms by comparing their recognition rates. Along with recognition rate, we also calculate the time required to compute the descriptors (not for the matching).

4.3 Results

The results of the evaluation are shown in Figure 5. Each image represents a sequence from the dataset, where the columns represent a pair of images from the sequence, with increasing difficulty.

As can be seen from the graphs, except for the Graffiti sequence, the BRIEF variants generally outperform the SURF algorithm. The Graffiti sequence is hard because it requires rotational invariance, which the BRIEF descriptor does not support.

The results also show that out color variant, ColorBRIEF, generally performs better than the original BRIEF descriptor, mainly for image pairs further in the sequence, where the addition of color information helps to better differentiate the descriptors. On the last images in the Wall and Graffiti sequence, it has twice the recognition rate of the original descriptor, for both sizes (32 and 64).

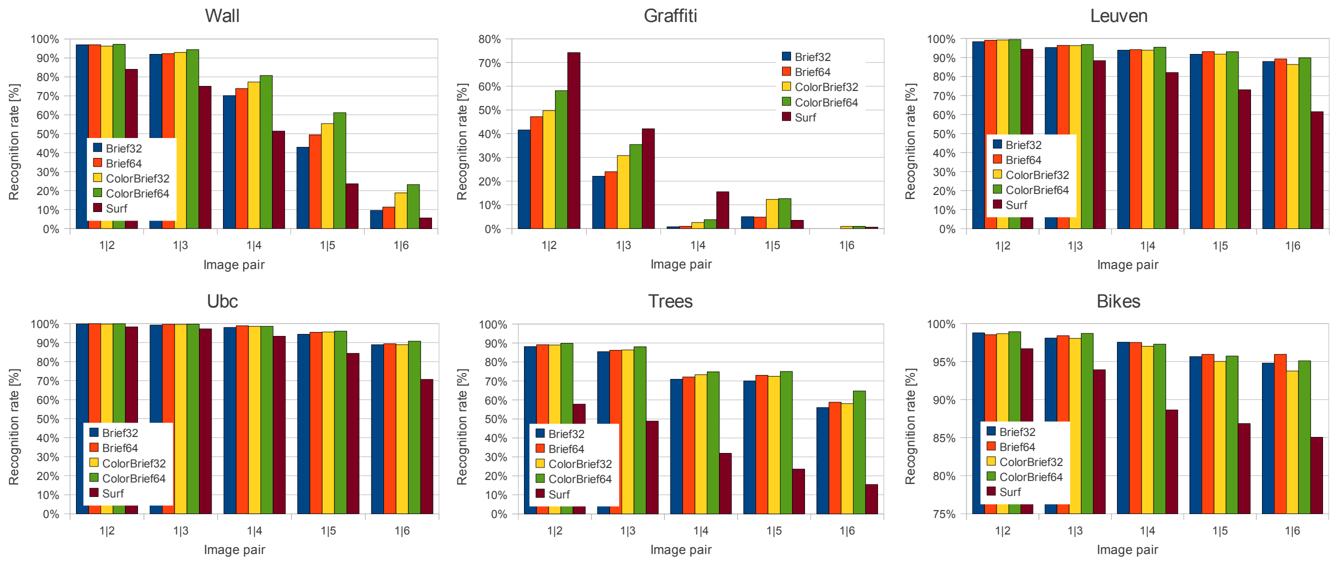

Figure 5: Results of evaluation of the selected description algorithms. Each graph represents the results on a sequence of the dataset. Each column represents a pair of compared images in the sequence, where further right means more complicated situation.

The computational requirements are shown in Figure 4. The time required to calculate our ColorBRIEF descriptor takes roughly 2× the time of the original descriptor. The additional time is needed because the integral image used to suppress noise has to be computed for every color channel, as was already explained in Section 3.1.

However, it still performs marginally better than the SURF algorithm. Since the amount of descriptors found per image is relatively high, we did not include the SIFT descriptor in our evaluation, since it's computation times were in order of minutes (not milliseconds).

5 Conclusion

The Color-BRIEF descriptor is based on the BRIEF descriptor and is extended to take RGB color information into account. We provided a description of the algorithm along with it's main properties.

It proves to be a very good descriptor in situation where high rotational and scale invariance is not required. The main advantages of this descriptor is it's small size and fast computation, making it a good candidate for high performance descriptor matching and mobile applications.

The results of our evaluation on a publicly available dataset show that in many situations it performs better than the original descriptor. It's recognition rates are also comparable and many times better than of the SURF algorithm, while being on average 10× faster.

Acknowledgements. This work was supported by the grant KEGA 244-022STU-4/2010.

References

BAY, H., TUYTELAARS, T., AND VAN GOOL, L. 2006. Surf: Speeded up robust features. *Computer Vision - ECCV 2006 3951*, 404–417.

BRADSKI, G. 2000. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*.

CALONDER, M., LEPETIT, V., STRECHA, C., AND FUA, P. 2010. BRIEF: Binary Robust Independent Elementary Features. *Computer Vision - ECCV 2010*, 778–792.

CHANDRASEKHAR, V., MAKAR, M., TAKACS, G., CHEN, D., TSAI, S., CHEUNG, N., GRZESZCZUK, R., REZNIK, Y., AND GIROD, B. 2010. Survey of SIFT Compression Schemes. *The Second International Workshop on Mobile Multimedia Processing*, 1–8.

LEPETIT, V., AND FUA, P. 2006. Keypoint recognition using randomized trees. *IEEE transactions on pattern analysis and machine intelligence 28*, 9 (Sept.), 1465–79.

LOWE, D. 2004. Distinctive image features from scale-invariant keypoints. *International journal of computer vision 60*, 2, 91–110.

MIKOLAJCZYK, K., AND SCHMID, C. 2005. Performance evaluation of local descriptors. *IEEE transactions on pattern analysis and machine intelligence 27*, 10 (Oct.), 1615–30.

OZUYSAL, M., CALONDER, M., LEPETIT, V., AND FUA, P. 2010. Fast keypoint recognition using random ferns. *IEEE transactions on pattern analysis and machine intelligence 32*, 3 (Mar.), 448–61.

ROSTEN, E., AND DRUMMOND, T. 2006. Machine learning for high-speed corner detection. *Computer Vision - ECCV 2006*, 1–14.

VAN DE SANDE, K. E. A., GEVERS, T., AND SNOEK, C. G. M. 2010. Evaluating color descriptors for object and scene recognition. *IEEE transactions on pattern analysis and machine intelligence 32*, 9 (Sept.), 1582–96.