

# Semantic Kernels Binarized – A Feature Descriptor for Fast and Robust Matching

Frederik Zilly<sup>1</sup>, Christian Riechert<sup>1</sup>, Peter Eisert<sup>1,2</sup>, Peter Kauff<sup>1</sup>

<sup>1</sup>Fraunhofer Institute for Telecommunications - Heinrich Hertz Institute, Berlin, Germany

{frederik.zilly | christian.rieichert | peter.eisert | peter.kauff} @hhi.fraunhofer.de

<sup>2</sup>Humboldt University, Berlin, Germany

---

## Abstract

*This paper presents a new approach for feature description used in image processing and robust image recognition algorithms such as 3D camera tracking, view reconstruction or 3D scene analysis.*

*State of the art feature detectors distinguish interest point detection and description. The former is commonly performed in scale space, while the latter is used to describe a normalized support region using histograms of gradients or similar derivatives of the grayscale image patch. This approach has proven to be very successful. However, the descriptors are usually of high dimensionality in order to achieve a high descriptiveness.*

*Against this background, we propose a binarized descriptor which has a low memory usage and good matching performance. The descriptor is composed of binarized responses resulting from a set of folding operations applied to the normalized support region. We demonstrate the real-time capabilities of the feature descriptor in a stereo matching environment.*

---

**Keywords:** Feature Descriptor, Feature Matching, 3D Scene Analysis, Stereo Matching.

## 1 Introduction

In the past years much attention has been paid to the development of robust interest point detectors and descriptors. SIFT [1] and SURF [2] are two prominent examples of such combined interest point detectors with associated descriptors. Moreover a number of dedicated region detectors have been proposed [7][8][9][10][23]. In-depth comparisons of the different detectors and descriptors have been carried out in [4] and [5].

Usually, the interest point detector searches for extremal values of a defined function (e.g. Laplacian/Difference of Gaussian or Determinant of Hessian functions) in scale space. Later on, a region around this interest point, nominated as support region, is described using a specific descriptor, while the support region might be normalized in scale and possibly in orientation (to achieve scale invariance and rotational invariance respectively). An automatic scale selection is commonly performed as proposed in [7]. Some region detectors are even invariant against affine transformations [6].

The commonly used descriptors, such as SIFT [1], SURF [2], or GLOH [4] use histograms of gradients to describe the support region. High dimensional feature vectors are used to describe the feature point. SURF uses 64 dimensions while SIFT uses 128 dimensions. Each dimension is typically represented by a floating point number or by a 1-byte integer value. This leads to a high memory usage and a fairly slow matching process.

Consequently, effort has been spent to reduce the dimensionality. PCA-SIFT proposed by [14] and GLOH [4] reduce the dimensions by performing a principal component analysis of the feature vectors. However, the description process is slowed down by this operation. Moreover, the descriptors itself remain floating point numbers or integer values which use the relatively slow L2-norm to match. On the other hand, the matching quality in terms of recall rate is much higher than simpler interest point detectors such as the Harris Corner detector [11] in combination with a cross-correlation based matching.

In the recent past, research has been conducted to build descriptors consisting of binary elements, denoted as binary strings in order to reduce the bitrate needed to transmit and to store the descriptors and to reduce the matching time, as matching can then be performed using the Hamming distance which can be efficiently computed on modern CPUs. Different descriptor binarization strategies exist. Some approaches first compute a higher dimensional descriptor, which will later be binarized by hashing the real-valued interest point descriptors [19], by performing a transform coding [20] or by direct binarization of the descriptor components [22]. A good overview of state of the art strategies for reducing the descriptor bitrate is given in [21] where a descriptor designed for efficient compression is being proposed.

On the other hand, an even further speed up can be achieved when the description process leads directly to a binarized descriptor. A prominent example for this method is the BRIEF descriptor [18]. A number of intensity comparisons (e.g. 256 in the case of BRIEF-256) is conducted which results in a 256-bit descriptor. This approach is nearest to the method proposed in this paper. Our proposed descriptor will therefore be compared to the BRIEF-256 descriptor.

The basic idea of the descriptor proposed in this paper is that a number of convolutions with a set of folding kernels is performed on the support region. The kernels consist of basic geometric structures, namely edges, ridges, corners, blobs, and saddles. As all these kernels have a dedicated meaning in computer vision, we call them therefore *semantic kernels*. As

the filter responses are binarized, we call the method *semantic kernels binarized*, or SKB.

Most of the kernels are computed for different orientations. In total, 16 kernels are evaluated at 16 positions around the interest point resulting in 256 kernel responses.

The filter response to the kernels is now binarized. We describe three different binarization strategies. In the easiest way, we need only 1 bit per filter response, in a more complex scenario we use 2 bits per filter response. Depending on the method chosen, all responses above a certain threshold  $t$  are set to 1, all others are set to 0, while the number of bits set to 1 can be normalized. This can be performed by choosing the threshold  $t$  such that a predefined number of bits are above the threshold.

The descriptor has consequently a dimension of 256 (512 when using 2 bits per filter response), but a size of only 32 (64) bytes. As comparison, SIFT uses 128 floating point numbers, resulting in a descriptor size of  $4 \times 128 = 512$  bytes.

The matching process can now be performed very efficiently. Indeed only a binary *AND* operation needs to be performed in order to calculate the scalar product between two feature vectors. Subsequently the population count of the bits set to 1 is evaluated. Modern CPUs support the SSE4.2 instruction set [17]. This contains dedicated functions to evaluate the number of bits set of a 64 bit number, i.e. the population count. Only 4 (8) SSE executions are required to evaluate the scalar product of two possibly matching interest points. The matching operation only needs to calculate the Hamming distance which can be evaluated very efficiently on modern CPUs.

The main contribution of this paper is the SKB descriptor which is described in section 3. However, to demonstrate its capability for real-time applications in a stereo matching environment, we briefly describe the complete processing chain which includes a fast interest point detector described in section 2. Section 4 describes different matching strategies. In section 5 results are presented, i.e. comparisons with other descriptors and a proof of concept of the real-time capability of the feature descriptor in combination with a fast interest point detector in a stereo matching environment.

## 2 Interest Point Detector

In this section, we briefly describe the interest point detector used in our real-time implementation. The interest point detector is a variant of SUSurE which has been proposed in [23].

We use a blob detector to detect interest points in scale space. Blobs are regions whose grayscale value near the center is higher (or lower) than the surrounding pixels. The Laplacian of Gaussian function, or similar functions such as the Mexican hat function can be used to detect blobs. In the case of SUSurE, a binarized kernel is used to detect extremal values of the Laplacian of Gaussian function. The detector uses only a single filter operation performed on an integral image in the box filter variant of SUSurE.

## 2.1 Filter Response Evaluation

The kernel is particularly efficient to evaluate. While [13] showed that a Difference of Gaussians can be used to approximate the Laplacian of Gaussian, SUSure's approach is based on an even further going simplification. Figure 1 shows the kernel used for the convolution. The innermost square has positive weights, surrounded by a square with negative weights. The black and the white region have the same surface. The outer region (grey) does not need to be considered any more. The weights have been set to zero.

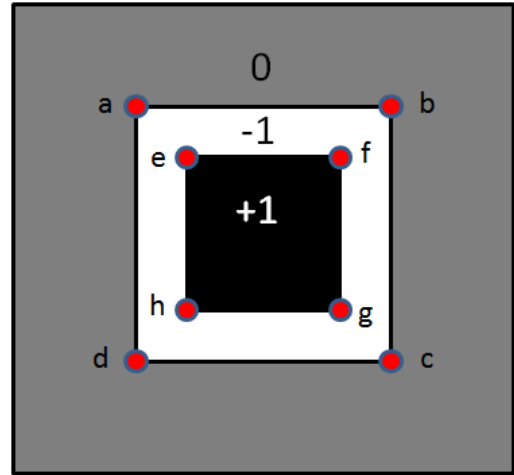


Figure 1: Kernel used for searching extremal values of the Laplacian of Gaussian in scale space.

An integral image is a very efficient way to evaluate box filter responses [15]. Consequently an integral image is calculated as a first pre-processing step. Using integral images, one can easily evaluate the sum of (rectangular) image regions by simple pixel fetch.

In our case, we need to evaluate only two lobes in order to evaluate the filter response. We evaluate only a single digital filter. For each lobe, the four corners need to be evaluated. In total, only eight pixel fetches are needed to evaluate the complete filter response.

The filter response  $r$  of our blob filter is:

$$r = (e + g + d + b) - (a + c + f + h). \quad (1)$$

Please note that the convolution operation is constant in time over all scales.

## 2.2 Building the Scale Space Pyramid

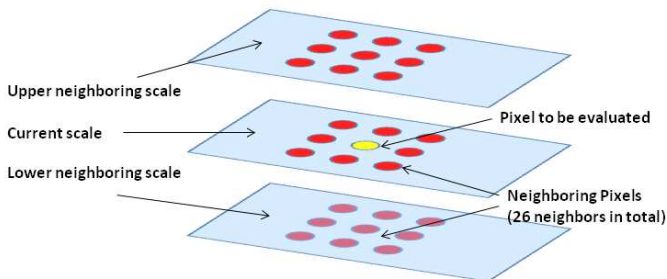
The filter responses are stored in a filter response pyramid. The pyramid consists of  $n$  octaves. Each octave consists of  $m$  intervals, i.e.  $m$  different scales exist per octave. Within the first octave, all pixels in the integral image are evaluated. For the subsequent octave, a 1:2 subsampling in  $x$ - and  $y$ -direction is applied, in the next octave a 1:4 subsampling etc. This saves processing time and memory usage for the filter response images. Within an octave, the sampling remains constant for all intervals.

### 2.3 Search for Local Extrema

After the filter responses have been evaluated, one searches for local extrema in the filter response images. All pixels in the convoluted images above a threshold  $t$  are examined.

*A neighborhood of 26 surrounding pixels is evaluated – 8 neighbors in the current scale, and 2x9 neighbors in the adjacent scales.*

Figure 2 illustrates this process.



**Figure 2: Non-maximum suppression and local search for extremal values.**

The search for local extrema can be efficiently implemented as proposed in [16].

### 2.4 Interest Point Localization

The local maxima which are above a threshold  $t$  are now candidates for interest points. In the higher octaves, the sampling of the filter responses is coarser than in the lower octaves, with respect to the integral image. Therefore, an additional sampling will be performed in our real-time implementation for the filter response around candidate pixels in higher octaves.

We will now need to interpolate the subpixel location of the interest point in scale and position. The basic mechanism is to build a Taylor polynomial of second degree [12].

The interest point candidates are not numerically stable if they do not correspond to a well defined blob, but to a ridge-like structure. If the subpixel interpolation position has a distance of more than  $\frac{1}{2}$  pixel from the original integer pixel position, the interest point candidate is discarded.

An explicit edge rejection (as performed within the DoG search used by SIFT [1]) is not performed here, but it can easily be introduced if required. However, our experience is that it is not necessary.

After the subpixel interpolation, the interest point has now a scale and position, which can later be used by the descriptor to define a suitable support region. Please note that we do not describe a rotational invariant detector here. Nevertheless, one can easily add support for rotational invariance by extracting the main gradient direction and “turn” the interest point (and later the support region) such that the direction points northwards. However, for many applications such as stereo matching with nearly rectified cameras, rotational invariance is not necessary.

## 3 The SKB-Descriptor

The major contribution of this paper is a new robust descriptor. The basic idea is to perform a set of convolution operations on the support region of the interest point. In total 16 different kernel responses are evaluated on 16 positions within the support region. The 16 kernels represent basic geometric structures which correspond to corners, edges, ridges, blobs, and saddles. This explains the naming *semantic kernels*.

This results in 256 dimensions of the descriptor. However, the filter responses can be binarized. Consequently, despite the high descriptor dimension, only 256 bit, i.e. 32 bytes are needed. This is particularly important, if an important number of interest point descriptors needs to be stored or transmitted over a network with limited bandwidth.

In addition, the matching can be performed very efficiently, as binary *AND* operations can be performed in order to evaluate the scalar product of two descriptors. Details will be provided in section 4.

### 3.1 Defining the Support Region

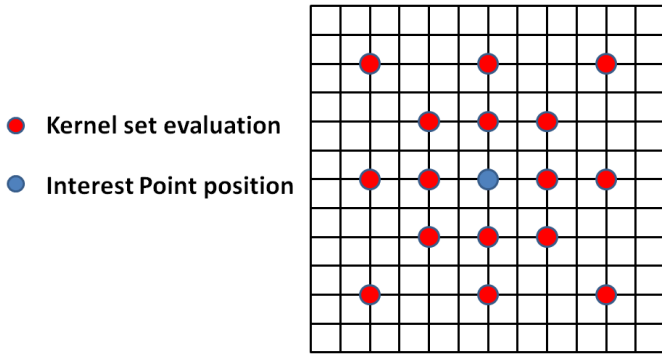
A first step of the description process is to interpolate the support region based on the subpixel position and scale of the interest point. If required, one can achieve rotational invariance by “turning” the support region such that the main gradient direction points northwards.

We propose two variants of the descriptor:

- Type A: support region of 12x12 pixel and overlapping kernel set evaluation regions, optimized for interest points with complex gradient structure in the center of the support region (e.g. corner detectors)
- Type B: support region of 16x16 pixel with equidistant and non-overlapping kernel set evaluation regions, optimized for interest points with uniform gradient structure, or (as for blob detectors) for interest points where most of the gradients occur at the border of the support region.

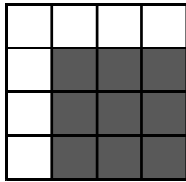
#### Type A: Overlapping Kernel Set Evaluation Regions

We will now describe the process of interest point description in detail. Let us assume, that an interest point has been detected and a normalized support region of 12x12 pixel around the interest point has been computed.



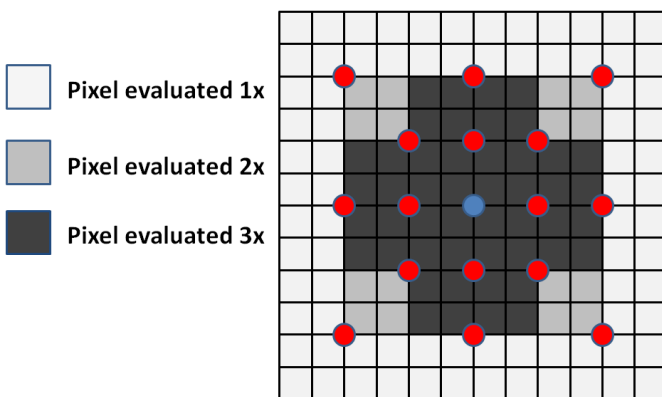
**Figure 3:** All kernels are evaluated at the positions around the red circles. The blue circle indicates the position of the interest point. Please note that the kernels have a size of 4x4 pixels, and that their respective evaluation regions overlap.

Around each of the red dots in Figure 3, a number of 16 kernels will be evaluated. An example is shown in Figure 4. All 16 kernels are shown in Figure 7 to Figure 10.



**Figure 4:** Example of a corner-like kernel of size 4x4.

Apparently, the regions of evaluation overlap as the kernels have a size of 4x4 and some evaluation points are only two pixels apart. Some pixels contribute to one, two or, three evaluation points. Figure 5 illustrates this behavior. The inner region will be evaluated three times, the outer pixels only once. This approximates a Gaussian giving more statistical weight to the center pixels.



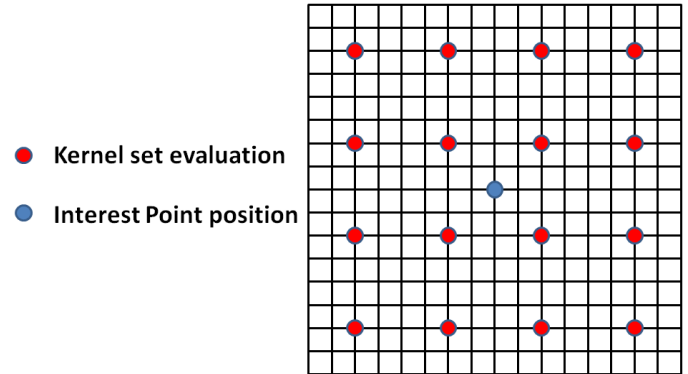
**Figure 5:** Overlap of the kernel set evaluation regions. The inner region will be evaluated three times, the outer pixels only once. This approximates a Gaussian giving more statistical weight to the center pixels.

This type of description is suitable when the most important gradients are near the interest point. This is the case for region

detectors or interest point detectors which search for corners, saddles, etc.

### Type B: Uniform Kernel Set Evaluation Regions

In the case of blob detectors, one should use a support region of 16x16 pixel. The sampling is now uniform and non-overlapping. Figure 6 illustrates this type of support region. All 16 kernels are evaluated at the red dot positions. The kernels have a size of 4x4, thus, the evaluated regions do not overlap. In total, 256 filter responses are calculated.



**Figure 6:** Support region suitable for blob detectors.

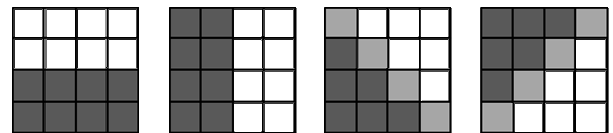
*Equidistant and non overlapping regions for kernel set evaluation.*

Blobs do not have a complex gradient structure at the center. The important gradients lie at the border of the support region. This type of support region is more suitable when the descriptor is combined with a blob-based interest point (or region) detector. Just like in the *type A* version, 16 kernels are evaluated at 16 positions resulting in 256 filter responses.

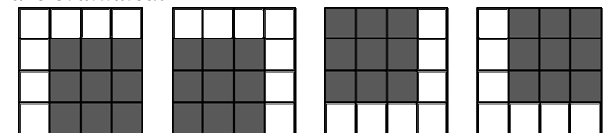
### 3.2 The Kernels

The support regions are convoluted using 16 different kernels. All kernels represent a named geometric structure justifying the terminology of *semantic kernels*. Many kernels exist for different orientations. For instance, four main directions are evaluated among the edge-like kernels.

All kernels are illustrated in Figure 7 to Figure 10.



**Figure 7:** Edge-like kernels. Four different main directions are evaluated.



**Figure 8:** Corner-like kernels.

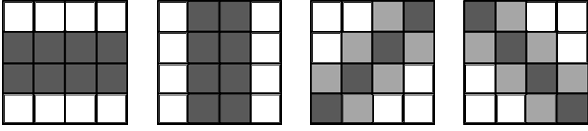


Figure 9: Ridge-like kernels

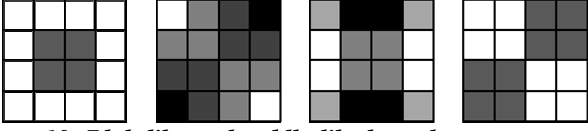


Figure 10: Blob-like and saddle-like kernels

### 3.3 Filter response Binarization

The filter responses resulting from the convolution with the semantic kernels can be binarized. There are three versions to perform this task.

#### Variant A: 256 bits

In the easiest way only the signum of the filter response is used. In that case, the descriptor has a dimension of 256 bits. It is not normalized, i.e. the number of bits which are set can differ from descriptor to descriptor. Matching can later be performed by evaluating the Hamming distance between two descriptors.

#### Variant B and C: 512 bits

Variant B and C use 2 bits per filter response  $r$  to describe a tri-state situation. Three cases are distinguished: positive answer of high amplitude, positive or negative answer of low amplitude, and negative answer of high amplitude. The resulting bit pair  $b$  depends on the filter response  $r$  and the threshold  $t$ :

$$b = \begin{cases} 01: & r > t \\ 00: & -t \leq r \leq t \\ 10: & r < -t \end{cases} \quad (2)$$

In variant B, the threshold  $t$  is fixed. This makes the computation faster, but also impairs the robustness against contrast changes. The descriptor is not normalized. The matching needs to evaluate the Hamming distance.

In variant C, the threshold  $t$  is self-normalized. The aim is to normalize the number of bits set. We define a unit length  $u$  to normalize the descriptor. The threshold  $t$  is now chosen such that for each descriptor, exactly  $u$  bits are set. Later matching can then be performed by evaluating the scalar product.

As we have 512 bits, representing a tri-state, the unit length  $u$  is calculated as following:

$$u = \left\lfloor \frac{512}{3} \right\rfloor = 171. \quad (3)$$

## 4 Matching

### 4.1 Matching Norm

One of the main advantages of the proposed descriptor is the fast matching ability. The descriptors are binary numbers. Consequently the scalar product which is needed for a correlation matrix can be computed very efficiently as well as the Hamming distance between two descriptors.

#### Scalar Product

To evaluate the scalar product  $s$ , a binary **AND** is performed between the feature vectors  $a$  and  $b$ . Subsequently, the number of bits set is evaluated, namely the population count:

$$s = \text{pop\_count}(a \wedge b). \quad (4)$$

The scalar product can be computed if the length of the feature vector is normalized as it is the case for binarization variant C.

#### Hamming Distance

The Hamming distance  $h$  between the feature vectors  $a$  and  $b$  is calculated using the binary **XOR** and subsequent evaluation of the population count:

$$h = \text{pop\_count}(a \vee b). \quad (5)$$

The Hamming distance can be computed when the length of the feature vector is not normalized as it is the case for binarization variant A and B.

Modern CPUs support SSE4.2 [17] which has dedicated instructions to calculate the *population count*. The CPU instruction *pop\_cnt64* evaluates the population count of a 64 bit number. Only eight executions of this instruction are needed for the 512 dimensional feature vector (variant C) and only 4 executions are needed for a 256 dimensional feature vector (variants A and B).

### 4.2 Additional Matching Constraints

The best matching feature vector (variant B and C: lowest Hamming Distance, variant A: highest scalar product) is compared to the second best. To achieve better matching results, we demand that scalar product of the best match is  $s$  times higher than the second best match. If the Hamming distance is evaluated, we demand that the lowest distance is  $s$  times lower than the second lowest distance.

## 5 Results and Conclusion

We have conducted a set of comparison tests. In a first step we've compared the three variants of the SKB descriptor using the evaluation framework and dataset used in [4]<sup>1</sup>.

<sup>1</sup> The data set is available at <http://www.robots.ox.ac.uk/~vgg/research/affine>.

Subsequently, we compared the fastest SKB variant with the BRIEF-256 descriptor [18] to compare the recall rate and the processing time of the description process.

In a last test we evaluated the runtime of a stereo matching application and its detection quality in a real-time environment.

### SKB vs. GLOH, SIFT and Cross Correlation

In a first test we compared the three variants of the SKB descriptor with GLOH, SIFT, and Cross Correlation using the test framework provided in [4].

As region detector we chose the Hessian-Laplace detector which is part of the region detectors provided in the test framework. We chose four images from the data set which do not require rotational invariance as we intended to use the descriptor for matching feature points in nearly rectified stereo images. Figures 11 to 14 show the results of this performance evaluation. For all test images, image 1 has been matched with the respective image 6, which shows the strongest artifacts, i.e. Blur for the *Bikes* and *Trees* images, Jpeg compression for the *UBC* image, and illumination change for the *Leuven* image.

The descriptor shows a very high precision when the threshold is chosen to reproduce a recall rate around 50% of the maximum recall rate. For applications such as stereo matching and calibration purposes, it is important to have a low outlier rate while a reasonable number of feature points is sufficient for this task. In other words, our aim is not to maximize the recall for high 1-precision values, but to maximize the precision at a given recall rate. For our tests, we chose 50% of the maximum recall rate as target value.

One can observe that the three variants A, B, and C show strong results at low thresholds for the nearest neighbour matching. In that scenario, the precision is high compared to other descriptors (*Bikes*, *UBC*, *Trees*). However, in the test image *Leuven*, which tests for strong illumination changes, other descriptors show a better performance.

### SKB vs. BRIEF

We compared the fastest SKB variant (SKB-A) with the OpenCV 2.2 implementation of BRIEF-256 as described in [18]. Both descriptors use 256 Bit and can be matched using a Hamming distance matcher. We performed a test where the recall rate was calculated for both descriptors using images provided in [4]. For different image pairs, the SURF detector [2] was used to detect interest points for the first image. Subsequently, this interest point was transferred using a ground truth homography into the second image. This approach is also performed in [18]. The homographies and images are part of the test framework provided in [4].

In the next step, the interest points were described using SKB-A and BRIEF-256. In a last step, the feature points were matched using a nearest neighbor matcher. Due to the ground truth homography, one can examine the rate of correct and false matches.

For each image set, image 1 is subsequently matched against the images 2, 3, 4, 5, and 6. The image degradation (blur,

illumination change, Jpeg compression) increases within each set. Consequently, the recall rate for the comparison between image 1 and 2 is higher than the comparison between image 1 and 6. The result is illustrated in figure 16. As one can see, the recall rate is higher for SKB in all cases.

We compared the runtime of SKB-A, SIFT, SURF, and BRIEF-256 descriptors when describing 8372 feature points. For SIFT, SURF and BRIEF the OpenCV 2.2 implementation was used using a single CPU at 3.33 GHz. The matching process was not compared, as it is identical between BRIEF-256 and SKB-A.

<i>Descriptor</i>	<i>Runtime in ms</i>
SIFT	2959
SURF	457
BRIEF-256	42
SKB-A (incl. Integral Image)	24
SKB-A	17

**Table 1: Runtime of different descriptors. 8375 feature points needed to be described. SKB-A performs the fastest even when the calculation of the integral image is included. Without counting the time for the integral image, it runs around 2.5 times faster than BRIEF-256.**

Table 1 shows the result of the runtime comparison. SKB-A performs the fastest with 24 ms for 8372 feature points, including the time required for the computation of the integral image. If the integral image is already available (e.g. already computed by the interest point detector) SKB-A needs only 17 ms.

### Real-Time Stereo Matching

In a third test, we used the SKB descriptor for a real-time stereo matching application. The video resolution was 960x540 pixels, the frame-rate was 25 Hz, i.e. 40 ms were available for the complete detection, description, and matching process.

A single screenshot is shown in Figure 15. The full video sequence is available as supplementary data. Around 3000 interest points were described per frame and camera, resulting in 600 consistent matches in average. The matching was performed using a left-right consistency check (i.e. matching twice). Afterwards, a RANSAC algorithm was used to estimate the fundamental matrix and to subsequently eliminate outliers according to [3]. Table 1 shows the time consumption of the real-time matching application running on an Intel Xeon X5680 CPU at 3.33 GHz. The CPU has 6 cores plus 6 cores via hyper threading. The matching was performed using all CPUs, while the interest point detection and description was performed on 2 cores (one per camera). The interest point description was performed on every second pixel, but the full resolution was used for the pixel position interpolation.

<i>Processing Step</i>	<i>Runtime in ms</i>
Integral Image	2
Binarized LoG	12
SKB Descriptors	3
Matching	12
RANSAC	8
<b>Total</b>	<b>37</b>

**Table 2: Runtime of the different image processing steps. At 25 Hz, an image pair needs to be processed within 40 ms.**

Table 2 shows that the real-time requirement was met within the stereo matching application. The description process and the matching needed 12 ms each. The description could be speeded up when using more than 2 CPUs for the process. However, even in this configuration, the stereo matching application ran fast enough for 25 fps.

### Conclusion

We have proposed a new feature descriptor which showed strong results at low outlier rates. Its suitability in terms of robustness and speed were demonstrated in a comparison with the BRIEF descriptor as well as the SIFT and GLOH descriptor. Moreover the SKB was successfully used within a real-time stereo matching environment, underlying its suitability for real-time image processing applications.

### References

- [1] D. Lowe, Distinctive image features from scale invariant keypoints. In *IJCV*, 60(2):91-110, 2004.
- [2] Herbert Bay, Andreas Ess, Tinne Tuytelaars, Luc Van Gool, SURF: Speeded Up Robust Features, *Computer Vision and Image Understanding (CVIU)*, Vol. 110, No. 3, pp. 346-359, 2008.
- [3] Hartley, R. and Zisserman, A. 2000. *Multiple view geometry in computer vision*, Cambridge University Press: Cambridge, UK.
- [4] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *PAMI*, 27(10):1615-1630, 2005.
- [5] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir and L. Van Gool, A comparison of affine region detectors. In *IJCV* 65(1/2):43-72, 2005.
- [6] K. Mikolajczyk and C. Schmid, Scale and Affine invariant interest point detectors. In *IJCV*, 60(1):63-86, 2004.
- [7] T. Lindeberg, "Feature Detection with Automatic Scale Selection," *Int'l J. Computer Vision*, vol. 30, no. 2, pp. 79-116, 1998.
- [8] J. Matas, O. Chum, M. Urban, and T. Pajdla, Robust wide baseline stereo from maximally stable extremal regions. In *BMVC*, p. 384-393, 2002.
- [9] T. Tuytelaars and L. Van Gool, Matching widely separated views based on affine invariant regions. In *IJCV*, 59(1):61-85, 2004.
- [10] T. Kadir, A. Zisserman, and M. Brady, An affine invariant salient region detector. In *ECCV*, p. 404-416, 2004.
- [11] C. Harris and M. Stephens. A combined corner and edge detector. In *Proceedings of the Alvey Vision Conference*, pages 147-151, 1988.
- [12] M. Brown and D. Lowe. Invariant features from interest point groups. In *BMVC*, 2002.
- [13] D. Lowe. Object recognition from local scale-invariant features. In *ICCV*, 1999.
- [14] Y. Ke and R. Sukthankar. PCA-SIFT: A more distinctive representation for local image descriptors. In *CVPR* (2), pages 506-513, 2004.
- [15] P.A. Viola and M.J. Jones. Rapid object detection using a boosted cascade of simple features. In *CVPR* (1), pages 511 - 518, 2001.
- [16] A. Neubeck and L. Van Gool. Efficient non-maximum suppression. In *ICPR*, 2006.
- [17] Intel Corporation, Intel SSE4 Programming Reference. Reference Number: D91561-001, April 2007.
- [18] M. Calonder, V. Lepetit, and P. Fua. BRIEF: Binary Robust Independent Elementary Features. In *Proceedings of the European Conference on Computer Vision*, 2010.
- [19] J. Ventura, T. Höllerer. Fast and Scalable Keypoint Recognition and Image Retrieval Using Binary Codes. In *IEEE Workshop on Motion and Video Computing (WMVC)*, 2011.
- [20] J. Brandt, Transform coding for fast approximate nearest neighbor search in high dimensions. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010, pp. 1815-1822.
- [21] V. Chandrasekhar, G. Takacs, D. M. Chen, S. S. Tsai, R. Grzeszczuk, Y. Reznik, and B. Girod. Compressed histogram of gradients: A low bitrate descriptor. In *Int. J. Comput. Vis. (Special Issue on Mobile Vision)*, May 2011, pp. 1-16.
- [22] M. Stommel. Binarising SIFT-Descriptors to Reduce the Curse of Dimensionality in Histogram-Based Object Recognition. In *International Journal of Signal Processing, Image Processing and Pattern Recognition* Vol. 3, No. 1, March, 2010.
- [23] Ebrahimi, M., & Mayol-Cuevas, W. (2009). SUSurE: Speeded up surround extrema feature detector and descriptor for realtime applications. In *Workshop on feature detectors and descriptors: the state of the art and beyond. IEEE conference on computer vision and pattern recognition (CVPR)*, 2009.

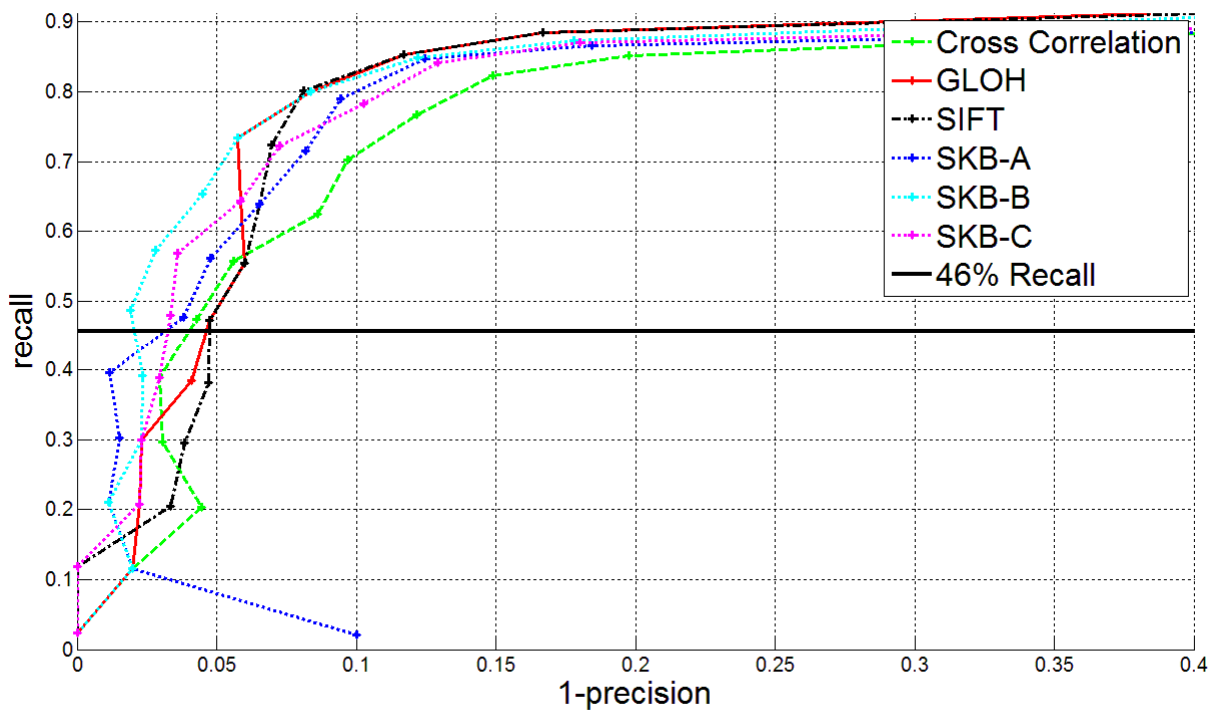


Figure 11: Bikes (Blur). The three variants SKB-A, SKB-B, and SKB-C have a similar performance. They show a particularly good performance at low thresholds compared to SIFT, GLOH, and Cross Correlation. At a recall rate of 46% which is half of the maximum recall rate, SKB-B has the highest precision, i.e. the fewest outliers.

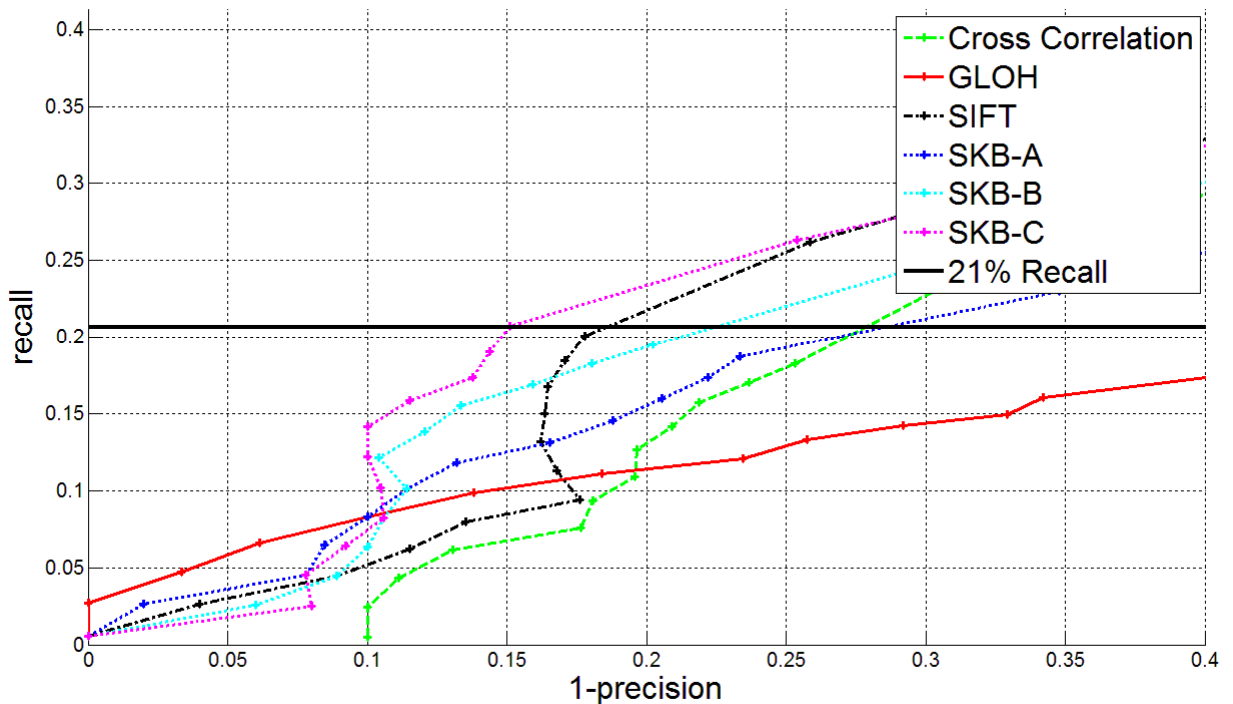


Figure 12: Trees (Blur). The three variants SKB-A, SKB-B, and SKB-C have a similar performance. They show a particularly good performance at low thresholds compared to SIFT, GLOH, and Cross Correlation. At a recall rate of 21% which is half of the maximum recall rate, SKB-C has the highest precision, i.e. the fewest outliers.

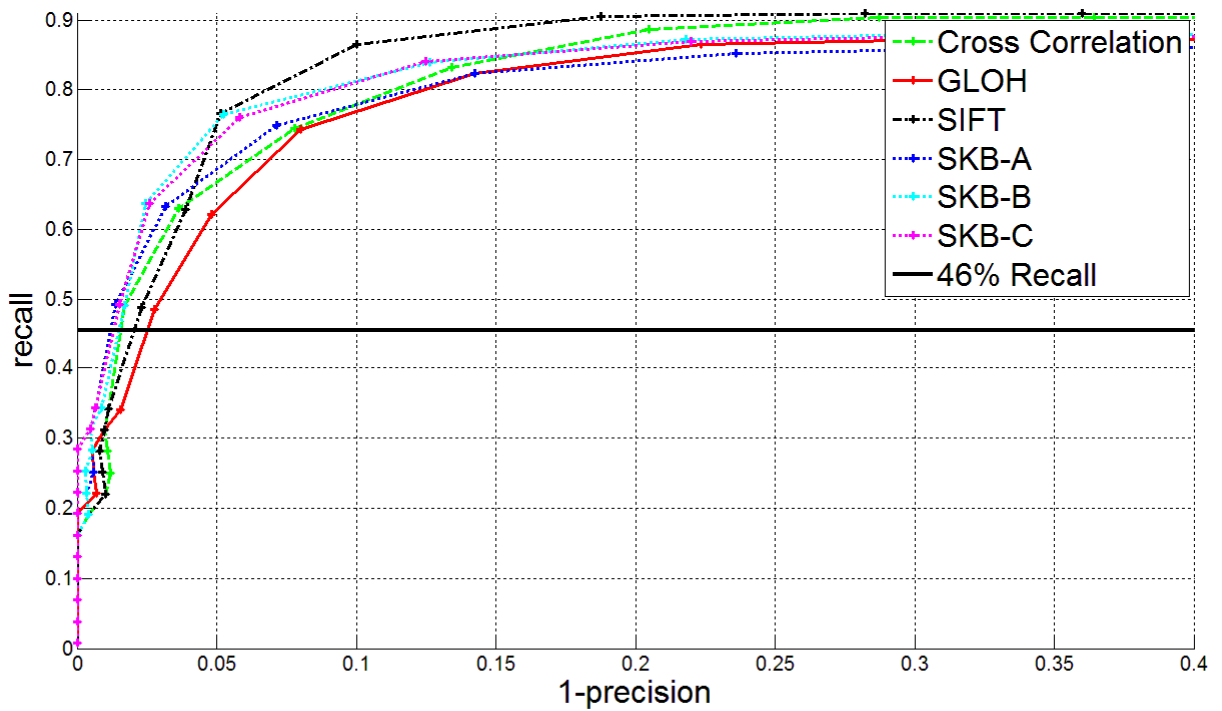


Figure 13: UBC (Jpeg compression artifacts). The three variants SKB-A, SKB-B, and SKB-C have a similar performance. They show a particularly good performance at low thresholds compared to SIFT, GLOH, and Cross Correlation. At a recall rate of 46% which is half of the maximum recall rate, SKB-A has the highest precision, i.e. the fewest outliers.

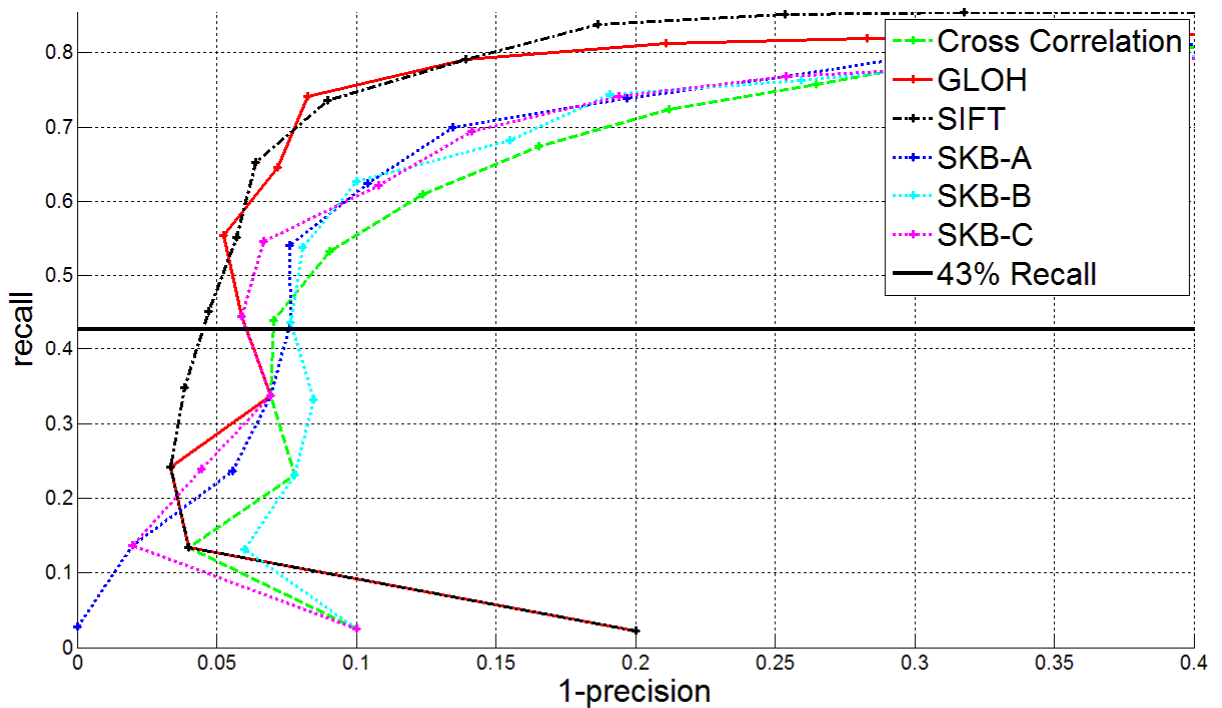


Figure 14: Leuven (Illumination change). The three variants SKB-A, SKB-B, and SKB-C have a similar performance. They perform better than Cross Correlation, but worse than SIFT and GLOH. At a recall rate of 43% which is half of the maximum recall rate, SIFT has the highest precision, i.e. the fewest outliers. GLOH and SKB-C have the same precision at that recall rate.



Figure 15: Screenshot of the real-time stereo matching application. One can see that the scene contains objects in different distances to the stereo camera. The matched feature points are coloured according to their horizontal disparity, brown for near objects, blue for far objects in blue. Apparently, no outliers are visible. The full sequence is available as supplementary data. The original resolution used for the matching was 960x540 pixel.

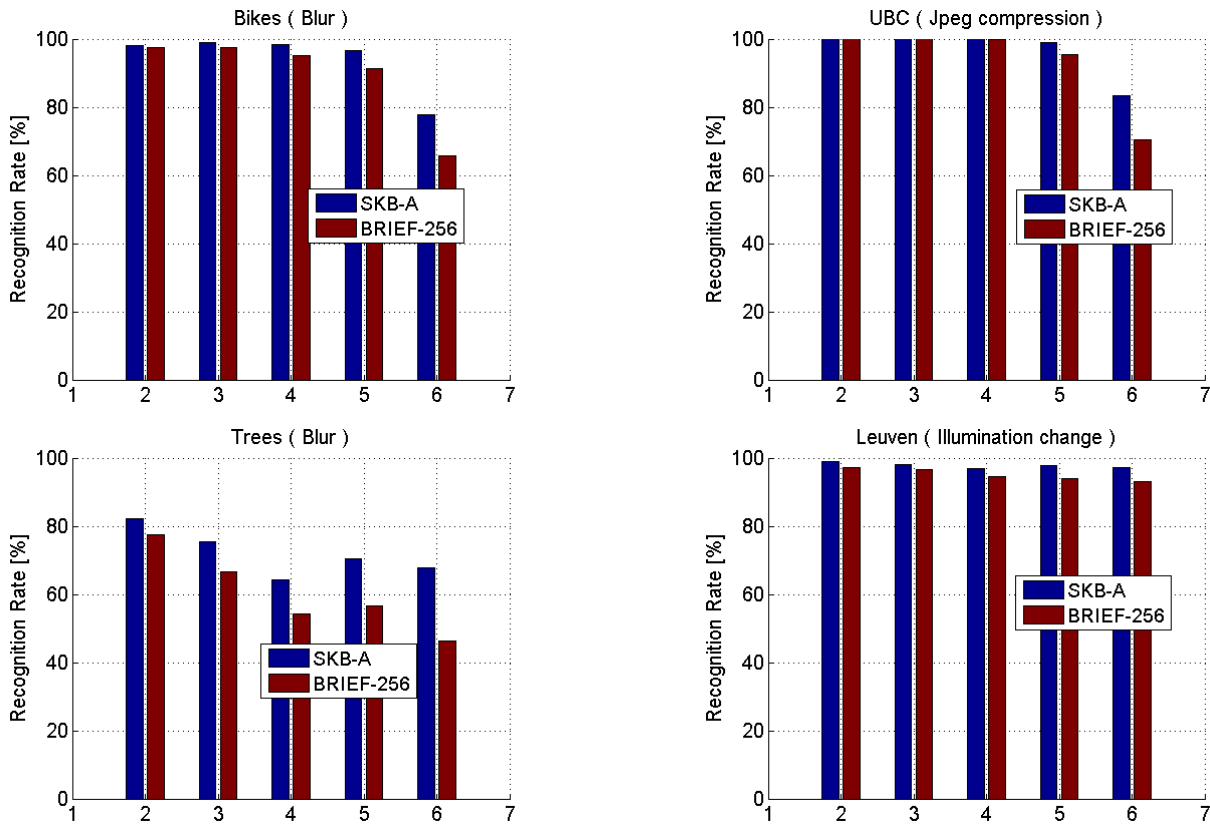


Figure 16: The SKB-A descriptor is compared to the BRIEF-256 descriptor using four image series. Each of the image series consists of 6 images, resulting in 5 pairs, i.e. image 1 is matched against the images 2 to 6. SKB-A shows a higher recall rate for all images.