TailoredBRIEF: Online Per-Feature Descriptor Customization

Andrew Richardson and Edwin Olson

***Abstract*— Image feature descriptors composed of a series of binary intensity comparisons yield substantial memory and runtime improvements over conventional descriptors, but are sensitive to viewpoint changes in ways that vary per feature. We propose a method to improve the matching performance of such descriptors by specifically reasoning about the reliability of test results on a feature-by-feature basis. We demonstrate an intuitive method to learn improved descriptor structures for individual features. Further, these learned results can be efficiently applied during matching with little increase in runtime. We provide an evaluation using a standard, ground-truthed, multi-image dataset.**

I. Introduction

Feature descriptors generated by a sequence of two-pixel intensity comparisons are capable of representing image features tersely and quickly. These so-called *binary* or *Boolean string* descriptors, which store a comparison's outcome in a single bit, require only a few bytes per feature (e.g. 8-64B, often 32B), greatly reducing memory footprint and network transfer bandwidth. In addition, computing and matching these descriptors requires much less runtime than well-known alternatives like SIFT and SURF, with comparable matching accuracy [1], [2].

One such method, BRIEF [1], is notable due to its intuitive formulation and ability to leverage vector instructions. But unlike one of BRIEF's predecessors, Randomized Trees [3], [4], BRIEF's comparisons are fixed and do not adapt to the image content of individual image features. This makes BRIEF sensitive to viewpoint change, as the intensities shifting under the fixed sampling pattern can cause the test outcomes, and thus the computed descriptor, to change. This results in increased overlap between descriptor error distributions for true and false correspondences, which ultimately leads to a higher false match rate.

Two of BRIEF's primary advantages are the small memory footprint and fast runtime. These are suited to the demands of real-time vision applications, with constrained bandwidth and high-FPS runtime targets. However, they also set a high bar for improvements – increases in descriptor robustness must come with minimal side effects. How to do this given the unique formulation of BRIEF – the Boolean string representation and use of `XOR`/`POPCNT` – is not especially clear.

This work was supported by the Ford Motor Company via the Ford-UM Alliance under award N015392, and by the Office of Naval Research award N00014-13-1-0217.

A. Richardson is affiliated with the Research and Innovation Center, Ford Motor Company, Dearborn, MI 48121, USA `arich163@ford.com`

E. Olson is affiliated with the Department of Computer Science and Engineering, University of Michigan, Ann Arbor, MI 48109, USA `ebolson@umich.edu`, `http://april.eecs.umich.edu`

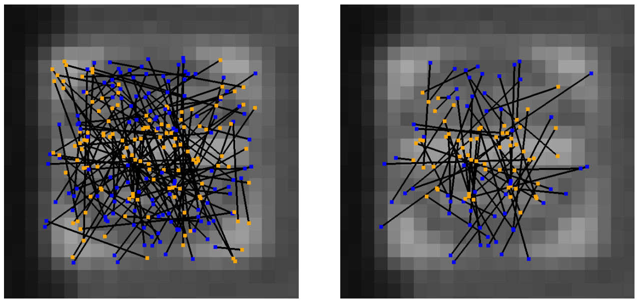


Fig. 1: Illustration of BRIEF and TailoredBRIEF intensity tests for the image patch shown. Learning is used to select the subset of tests, (b), from the full set, (a), that produce repeatable results under simulated viewpoint change. The surviving tests for this image feature tend to compare the patch center to the patch perimeter.

This paper proposes to effectively learn a unique descriptor *structure* for every feature in an image, and to do this at runtime. The key is in doing this efficiently and in a way that is congruous with the suggested implementation of BRIEF. For that, we propose *tailoring* for Boolean string descriptors, and argue that this approach is suitable for real time systems.

Unlike most approaches for feature descriptor matching, the proposed method uses an asymmetric division of labor between *reference* features and *query* features. We define a reference feature to be one in a keyframe or map, and a query feature to be those matched against reference features. Appropriate applications include Visual Odometry, SLAM, and Localization. In addition, we propose an asymmetric descriptor representation for reference and query features, as an application like Visual Odometry can afford a larger memory footprint or bandwidth requirement for the occasional reference feature.

Despite this asymmetry, the runtime for matching with the proposed method is basically unchanged, and the precision and recall improved. We achieve this by simulating the effect of viewpoint change on reference feature descriptors and defining an appropriate weighting vector to suppress unreliable intensity tests. Figure 1 shows an illustration of the result, where the focus of the intensity tests has shifted from random pixel pairs to center-perimeter comparisons for the specific feature in question.

The remainder of this paper begins with prior work and an explanation of BRIEF descriptor extraction in Sections II and III. Experiments highlighting the sensitivity of intensity tests to viewpoint change follow in Section IV. The method is detailed in Section V and evaluated in Section VI.

The contributions of this work are:

- A demonstration of the viewpoint sensitivity of intensity comparison descriptors
- A method for computing descriptor weighting vectors through synthetic observations of an image patch
- An efficient distance function for weighted descriptors
- Evaluation of matching performance under typical appearance changes and comparison to prior art

II. Related work

This work is based on BRIEF [1], which stems from a line of research from Lepetit, Fua, et al on intensity-test based feature matching [3], [4], [5], [6], [7], [1], [8].

Lepetit and Fua investigated Randomized Trees as a way to reformulate descriptor-based feature matching as a classification problem [3], [4]. They consider a class to be all views of the same scene feature, and perform offline training to learn a decision tree composed of ternary intensity tests. Synthetic views are generated to increase the available training data, and multiple randomized trees are grown because of the claimed intractability of building an optimal tree. Even using randomized trees, they describe a learning time around 15 minutes in 2005 [3].

Özuysal, Calonder, Lepetit, and Fua addressed scalability of this style of classification with Randomized Ferns [5], [6]. The primary difference between randomized trees and ferns is that while a randomized tree may perform different tests given the outcome of the previous test, a fern converted to a tree always performs the same test at a given depth. Additionally, Özuysal et al consider combining the class-membership probabilities estimated by each tree or fern in a Naive Bayes manner, instead of simply averaging, and observe a 10-20% increase in feature recognition rates. Ultimately, joint probabilities are considered between tests in the same fern, and independence assumed between ferns. This allows some control over the size of the required joint probability tables, which grow exponentially in the size of the fern.

Calonder et al introduced BRIEF [1], [8], which is formulated as a typical feature descriptor and does away with the joint probability estimates. Additionally, they justify the design by pointing out that the Hamming Distance between two descriptors can be computed with `XOR` and `POPCNT` instructions, greatly improving feature matching speed.

Offline learning was applied to binary string descriptors in BinBoost [9]. An AdaBoost-like classifier is applied to determine each bit using image gradient orientations within subregions of an image patch. Compared to BRIEF, BinBoost descriptor matching requires less computation time, as fewer bits may be used. However, the use of strong classifiers for each bit does increase the descriptor extraction time, which may not be desirable for a high-rate, real-time vision system.

Others have also contributed in this space. Rublee et al described ORB [2], combining the FAST feature detector [10], an orientation estimator, and a method to greedily choose test positions. In particular, they choose tests that are less correlated under rotation than a typical set of BRIEF tests. Leutenegger et al described BRISK [11], which uses the FAST score as a saliency measure to achieve invariance to scale and a radially-symmetric sampling pattern.

In contrast to these approaches, we propose TailoredBRIEF, an extension to BRIEF that allows per-feature customization of the tests used to robustly describe a feature. TailoredBRIEF focuses on the descriptor and matching aspects of the problem, and does not attempt to construct a full detector/descriptor system like ORB and BRISK. Finally, unlike the offline training used for Randomized Trees, Randomized Ferns, and BinBoost, TailoredBRIEF is intended for online learning as new features are detected for the first time, which is typical for a robot exploring an unknown environment.

III. Descriptor extraction

The BRIEF descriptor summarizes local appearance through intensity tests between pairs of pixels surrounding an image feature. The Boolean outputs of the tests are stored bit-packed to minimize memory usage. Further, stored in this way, the `XOR` instruction can be used with `POPCNT` to compute the number of bit errors between two BRIEF descriptors, also known as the Hamming Distance. This results in a very terse descriptor that can be matched much faster than alternatives like SURF [1], [12].

Before computing a BRIEF descriptor, a set of test points must be defined. Calonder et al explored the effect on feature matching recall with test points drawn from a number of parameterized random distributions [1]. We use a Gaussian distribution, following from their results. Once defined, the same test points are repeatedly used. However, for scale-invariance, the relative test points' positions may be resized according to a feature's scale. Finally, while the number of tests is arbitrary, it is typically chosen to be a multiple of the `POPCNT` operand length.

Algorithm 1 illustrates how a BRIEF descriptor is computed for a given feature. For each pair of scaled test points, both image intensities are looked up relative to the feature's position. If the second intensity is greater, the appropriate bit in the descriptor is set. When computing the error between two descriptors, each corresponding section from the two descriptors is loaded, differenced using the `XOR` instruction, and counted with `POPCNT` to determine the Hamming Distance.

IV. Matching errors due to viewpoint change

Image feature matching by computing nearest-neighbors in a descriptor space can be confounded by image feature appearance changes. These appearance changes result from sensor noise, changes in the environment, such as variation in lighting intensity or direction, and changes in the camera viewpoint. For descriptors composed of two-point intensity comparisons, viewpoint changes shift the test points across the image patch, resulting in different outcomes for some intensity comparisons. These instabilities increase the matching

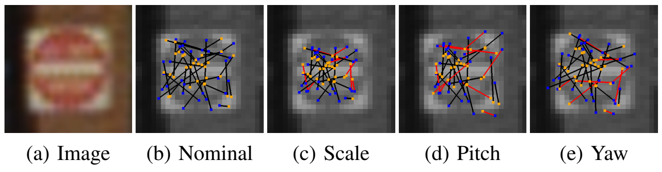
Algorithm 1 BRIEF_EXTRACT (tests, im, x, y, scale)

```
bits = zeros(length(tests))
for all test $\in$ tests do
    a = im(x + scale*test.x1, y + scale*test.y1)
    b = im(x + scale*test.x2, y + scale*test.y2)
    if a $<$ b then
        bits[test.index] = 1
    end if
end for
return bits
```

error for a true correspondence, which leads to false matches when the true error is too great.

We can explore this relationship by simulating BRIEF descriptor extraction under viewpoint changes. Figure 2 shows an image patch centered about a road sign, which contains sharp transitions, flat regions, and small text. As we sweep over changes to scale and both in-plane and out-of-plane rotations, we transform the BRIEF test positions and compute the intensity comparison results in the original image. This corresponds to applying the inverse transformation to the image before descriptor extraction, but is simpler, as the full set of transformed test positions can be cached.

The descriptors computed under simulated viewpoint change are then compared to the original descriptor. Figure 2 shows plots of the matching error, the Hamming Distance, for two of the simulated viewpoint changes. As we move along the axes away from the nominal scale or orientation, the true matching errors reach just under 33% of the total number of bits, even for small viewpoint changes.



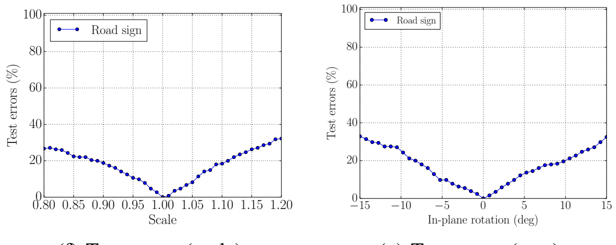


Fig. 2: Intensity tests on a road sign patch, (a), under simulated viewpoint changes. (b – e) Viewpoint change examples show the transformed positions of the nominal intensity tests. The test outcomes are denoted with colored endpoints and lines. Orange endpoints denote higher intensities. Red lines denote tests that differ from the nominal viewpoint. (f – g) Sweeping over viewpoint change parameters, we measure test errors just under 33% under small viewpoint changes (20% scale increase and $+15°$ in-plane rotation)

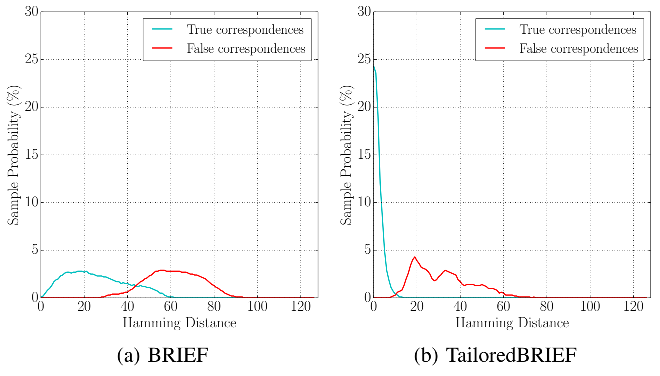


Fig. 3: Descriptor matching error distributions for example patches with and without the proposed method.

Often, we are interested in the matching error distributions for true and false correspondences. When these distributions have minimal overlap, we expect to compute a true correspondence with high probability. Drawing from independent, uniform random distributions for the viewpoint change variables over a fixed range, we can estimate the true-correspondence error distribution. To compute the false-correspondence distribution, we can compare all sampled descriptors from one class, or image patch, to those sampled from all other classes.

Figure 3 shows these sample distributions for a typical BRIEF descriptor and the proposed method. A small library of seven input patches was used, computing descriptors for each patch under 200 simulated viewpoint changes. All true and false correspondences were then compared to estimate the sample probability distributions. Without learning which tests are robust, the true and false probability distributions overlap. For some combination of patches and viewpoint changes, selecting the minimum Hamming Distance pair will result in a false correspondence. Further, the true correspondence mode is clearly non-zero, between 10% and 20% of the 128 bits used. In contrast, by applying the per-feature learned weights (proposed), the distribution overlap is decreased substantially and the true correspondence mode shifts to an error of zero bits. This suggests that feature matching precision will increase as a result.

V. Method: Tailoring BRIEF

Through online learning, we desire to improve the accuracy of descriptor-based feature matching. We achieve this primarily by considering the effect of appearance changes on the consistency of an individual feature descriptor. Like Lepetit et al, we simulate the outcome of these appearance changes to generate training data [3], which is in turn used to generate a Boolean-weighting vector that we refer to as a descriptor *mask*. This mask is used repeatedly in the inner loop of the matching process, when the matching error is computed for a specific pair of features.

Key to leveraging this technique is an efficient implementation. The design and efficiency of the BRIEF descriptor make it especially well-suited to this task. This is in part because certain intermediate results can be reused during training. It is also because the Boolean nature of the image tests lend themselves to efficient descriptor mask training, as well as application during feature matching.

The remainder of this section discusses the specific approach to learning a descriptor mask, and the application of these masks during matching.

A. Formulation

In the context of feature matching, not all intensity comparisons are equally reliable. While it is possible to learn a better set of test points, as shown by Rublee et al to reduce average test correlation [2], for any specific test, some image patch exists which will produce different results under a small perturbation. If tests were learned for individual image patch instances, this effect could be minimized.

One can imagine utilizing a different set of BRIEF test points for each of n features in a reference image. During matching, n descriptors would be extracted for each of m image patches in the query image, for a total of $n \times m$ unique descriptors and $n \times m$ matching errors. For $n = m$, which is common, the number of descriptors extracted grows quadratically in the number of features, instead of linearly as with a single, fixed set of test points. This presents an obvious challenge for real-time systems which target 30-60 Hz operation.

We propose instead to extract a single descriptor for each image feature and learn a vector of weights for each test. If we assume that the tests are independent and produce errors according to a Bernoulli distribution, we can estimate p_i for each test i by sampling viewpoint change parameters and warping the image patch or test points appropriately. We would then compute the probability of a true match as a function of the test errors and Bernoulli probabilities. However, this would negate a key property of BRIEF, as the extra mathematical operations would significantly increase the number of operations required to compute the error between two descriptors. Instead, we propose to learn a Boolean weighting vector and apply it with AND to the error vector, suppressing noisy tests. In this way, we can select the subset of tests that are reliable for a particular image patch. Despite discarding a potentially large number of tests, we show that the matching performance actually increases.

This Boolean weighting vector can be applied efficiently during matching. The Boolean weights are stored bit-packed as in the BRIEF descriptor. Algorithm 2 illustrates the matching error function with descriptors for two features, and one descriptor's learned mask. Here 'blocks' is the number of 64-bit blocks of bits in descriptors a and b, and the mask. For example, blocks is 4 for a 256-bit descriptor.

Algorithm 2 MASKED_DISTANCE (blocks, mask, a, b)

```
dist = 0
for all bi ∈ range(0, blocks) do
    dist += POPCNT(mask[bi] and (a[bi] xor b[bi]))
end for
return dist
```

In practice, masks could be learned for both sets of features, instead of one as shown. However, for many systems, this is unnecessary. For systems like Visual Odometry and Visual SLAM, reference features are added only periodically. We can take advantage of this asymmetry by performing extra processing on the reference features without affecting descriptor extraction time for the query features. Additionally, an important property of this formulation is that memory usage increases only for reference features, which require twice the memory, while the memory for query features is unchanged.

B. Mask learning via viewpoint simulation

Training data from which to compute a descriptor mask is gathered by sampling viewpoint changes from uniform distributions in scale and 3-axis rotation. The full transformation is shown in Equation 1, where $\mathbf{R}$ represents a 3D rotation matrix generated from in-plane and out-of-plane rotation terms sampled from zero-mean distributions. The original test point coordinates x and y in the range $[-0.5, 0.5]$ are rotated. The result is then projected as if at unit distance by a camera with focal length s, where s is sampled from a distribution with mean 1.

$$\mathbf{x_p} = \begin{bmatrix} x_p \\ y_p \\ w \end{bmatrix} = \begin{bmatrix} s & 0 & 0 \\ 0 & s & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \left(\mathbf{R} \cdot \begin{bmatrix} x \\ y \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \right) \tag{1}$$

Note that while we sample only over a small number of viewpoint change parameters, other terms such as additive noise could be readily integrated.

The transformed test point coordinates $\mathbf{x_p}$ are computed once and stored. To learn the descriptor masks, we do the following for each feature:

1) Compute all transformed descriptors using Algorithm 1
2) Comparing the original descriptor to each of the transformed descriptors, compute the number of errors for each test i
3) Estimate the sample probability $p_i(error)$ for each test
4) If $p_i(error)$ is greater than a small threshold, reject test i by setting its weight to zero

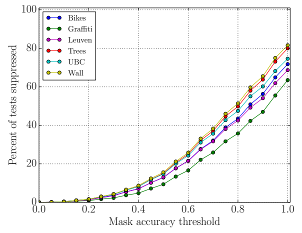
This is described further in Algorithm 3.

VI. Evaluation

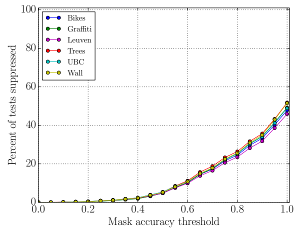
Matching performance for BRIEF and TailoredBRIEF was evaluated using sets of still images related by ground truth homography matrices. We use a portion of the standard affine region detector datasets from Mikolajczyk et al [13], as in the evaluation of BRIEF [1]. These datasets include the Bikes, Graffiti, Leuven, Trees, UBC, and Wall datasets, which possess varying degrees of viewpoint change (Graffiti, Wall), blur (Bikes, Trees), lighting change (Leuven), and image

Algorithm 3 LEARN_MASK (tests, im, x, y, scale, thresh)

```
desc = brief_extract(tests, im, x, y, scale)
mask = ones(length(tests))
for all test ∈ tests do
    transformed = transform_test(test)
    error = 0
    for all sample_test ∈ transformed do
        bit = compare(im, x, y, scale, sample_test)
        if bit ≠ desc[test.index] then
            error++
        end if
    end for
    if error > thresh then
        mask[test.index] = 0
    end if
end for
return mask
```



(a) Without blur



(b) With 7-pixel box filter

Fig. 4: Percentage of TailoredBRIEF test results that are masked-out as a function of the sample accuracy threshold.

compression (UBC). Omitted datasets possessed rotation to the degree that an orientation estimator is required, which is not considered in the scope of this work.

A. Evaluation of masks

Figure 4 shows the percentage of intensity tests that are rejected as a function of the sample accuracy threshold. These results were computed using 800 difference-of-box Center Surround Extrema (CenSurE) features detected on the first (reference) image from each dataset [14]. CenSurE features were used following from Calonder et al's report of improved recognition over SURF [1]. We used 50 randomly-sampled viewpoint changes in the ranges specified by Table I.

Blurring before learning the descriptor mask makes a notable difference on the repeatability of tests, decreasing the number of masked tests for the Wall dataset by 30% when the mask accuracy threshold is set to 1. Despite suppressing a substantial number of tests, approximately half of the tests are 100% repeatable over the range of simulated viewpoints, and 76.7% of tests are repeatable at least 75% of the time.

B. Computation time

Runtimes for various stages of BRIEF and TailoredBRIEF were computed using an Intel i7-4900MQ 2.8GHz processor and are shown in Table II. The total times are reported averaged over 100 trials using 812 image features detected on the first image in Wall. Box filtering is optional but, if enabled, is needed for both the descriptor extraction and mask learning steps.

Parameter	Min	Max
Scale	0.8	1.25
Roll	-12°	12°
Pitch	-12°	12°
Yaw	-6°	6°

TABLE I: Ranges for randomly-sampled viewpoint change parameters. Roll and pitch in this case are out-of-plane rotations, and yaw an in-plane rotation

BRIEF and TailoredBRIEF require the same amount of time for descriptor extraction, which is negligible in comparison to the box filter. Matching takes 0.97 ms longer for TailoredBRIEF due to the inclusion of a descriptor mask. The amount of time required to learn a descriptor mask depends on the number of viewpoint samples used: 16.07 ms for 10 samples, 37.94 ms for 25, and 149.28 ms for 100. If learning masks on keyframes using 25 viewpoint samples, masks are ready for use with just over a one-frame delay at 30 FPS, which should be sufficient for most applications. With a small number of samples, masks could be learned for every frame, if desired.

Computations times for BGM and BinBoost are also included. Compared to BRIEF and TailoredBRIEF, descriptor computation times are over two orders of magnitude larger when using the reference implementations for BGM and BinBoost. Feature matching recall does improve as a result, as shown in Section VI-C. However, for a 30-FPS vision system, these runtimes exceed the available time per update.

C. Precision-Recall Evaluation

Due to the prevalence of detection and matching parameters, we use precision-recall plots in our evaluation. The homography ground truth is used to classify feature matches with a radial data association threshold of 5 pixels, irrespective of feature scale. Care is taken to estimate the recall denominator by computing the number of features with geometric neighbors when mapped through the homography. This is done to avoid underestimating recall when detector

Step	Time (ms)			
	BRIEF	TailoredBRIEF	BGM	BinBoost
Box filter	4.24	4.24	4.24	4.24
Compute descriptor	0.26	0.26	367.5	205.8
Match features	1.89	2.86	3.21	2.14
Total	6.39	7.36	374.9	212.2

TABLE II: Runtimes for stages of description and matching. BRIEF and TailoredBRIEF have identical runtimes for filtering and descriptor extraction, but the matching runtime is higher by 0.97 ms for TailoredBRIEF due to the integration of descriptor masks. Both methods are well under the time constraint for a typical 30 FPS or 60 FPS runtime budget.

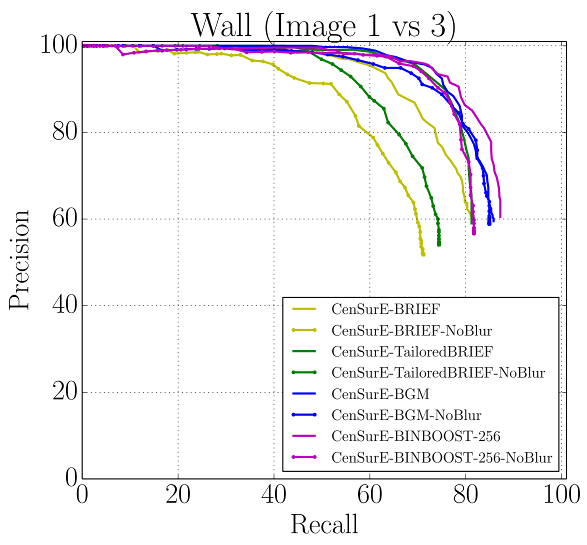


Fig. 5: Precision-recall curves for the Wall dataset, image 1 vs 3. Approximately 800 features were extracted from each image using the CenSurE detector and descriptor specified, matched with nearest neighbor and a sweep over the matching error threshold. BRIEF and TailoredBRIEF were evaluated with and without a 7-pixel box filter to reduce high frequency content.

repeatability is poor. This is important to put the magnitude of repeatability improvements in perspective. Calonder et al refer to this as the *recognition rate* [1].

Parameters were sampled in the ranges specified by Table I, again detecting 800 CenSurE features. We used 256 intensity tests and a box filter width of 7, where applicable. The number of viewpoint change samples was set to 25. This was chosen to balance the runtime/performance trade off shown in Section VI-B. The threshold used in Algorithm 3 was set to 0.1 (up to 10% test errors permitted).

Figure 5 shows the precision-recall curves for Wall, image 1 vs 3. The matching error for TailoredBRIEF was computed as discussed in Section V-A, and features were matched by computing the nearest neighbor descriptor in the query image for each feature in the reference image.

Results for two versions of the BRIEF and TailoredBRIEF (proposed) descriptors are shown. Calonder et al reported use of a box filter to blur an image before extracting descriptors. This was justified due BRIEF's sensitivity to high frequency content, and we see that the version without the box filter, denoted 'CenSurE-BRIEF-NoBlur', does yield a lower precision and recall. The results are similar for TailoredBRIEF; however, it is clear that the blur does not suppress all sensitivity to viewpoint change and that the benefits of blurring and descriptor mask learning are not exclusive.

TailoredBRIEF shows a large increase in performance over BRIEF, including a 10.3% increase in recall from 65.3% at 90% precision. At the same precision level, TailoredBRIEF captured 29.6% of the remaining true matches between the two frames. Without image blurs, we see a 5.8-18.2% increase in recall between 90% and 98% precision, respectively. Systems targeting high framerates could reduce the total CPU load by replacing the every-frame blurring operation needed for the 'Censure-BRIEF' level of performance with per-feature descriptor learning on the occasional keyframe.

Results are also shown for the gradient-based boosted image descriptors BGM [15] and BinBoost [9]. Unlike TailoredBRIEF, BGM and BinBoost are trained offline to improve matching accuracy and employ gradient-based weak learners. Like BRIEF and TailoredBRIEF, the Precision-Recall performance is improved through modest image blurring, as shown in Figure 5. In this example, the Precision-Recall curves for TailoredBRIEF, BGM, and BinBoost correspond until a recall of approximately 75%, after which the recall of BGM and BinBoost exceeds TailoredBRIEF.

The full set of Precision-Recall curves are shown in Figure 6, using all datasets and query images. The largest increase in recall occur for the Wall, Bikes, and Trees datasets, which is expected due to the out-of-plane rotation (Wall) and blur (Bikes, Trees) effects captured in training TailoredBRIEF. Similar performance results for datasets whose effects aren't simulated in training TailoredBRIEF, including lighting change (Leuven) and image compression (UBC). The Graffiti dataset, as well as Bark and Boat (not included), involve large in-plane image rotations that require an orientation-aware descriptor to improve performance.

For datasets whose effects are captured in training (Wall, Bikes, Trees), we consider the increases in recall by column. For query image 2, where the magnitude of out-of-plane rotation or blur is small, recall is already in the 60-80% range and we see improvements of about 5%, more in the case of Wall. For query images 3 and 4, the recall often improves by 10%.

The boosting-based BGM and BinBoost descriptors show greater recall in many instances, such as on Wall, Bikes, Trees, and UBC, particularly for precision values below 80%. Precision is reduced on Leuven as compared to BRIEF and TailoredBRIEF, and increased on Graffiti, which involves large viewpoint change.

The difference in Precision and Recall performance is summarized in Table III. Due to the focus on low-runtime methods, areas are not included for BGM and BinBoost. The area under the Precision-Recall curve was computed for both BRIEF and TailoredBRIEF, which used the 7-pixel box filter. By this metric, TailoredBRIEF's performance exceeds BRIEF's for most tests, and exceeds or matches BRIEF in all but one case.

VII. Summary

TailoredBRIEF is a method to improve feature matching performance for intensity test feature descriptors, which are sensitive to viewpoint change even after filtering. Through viewpoint simulation, a descriptor mask is learned on a per-feature basis by characterizing the robustness of each intensity test. This descriptor mask is easy to integrate during feature matching, results in only a slight increase to runtime,

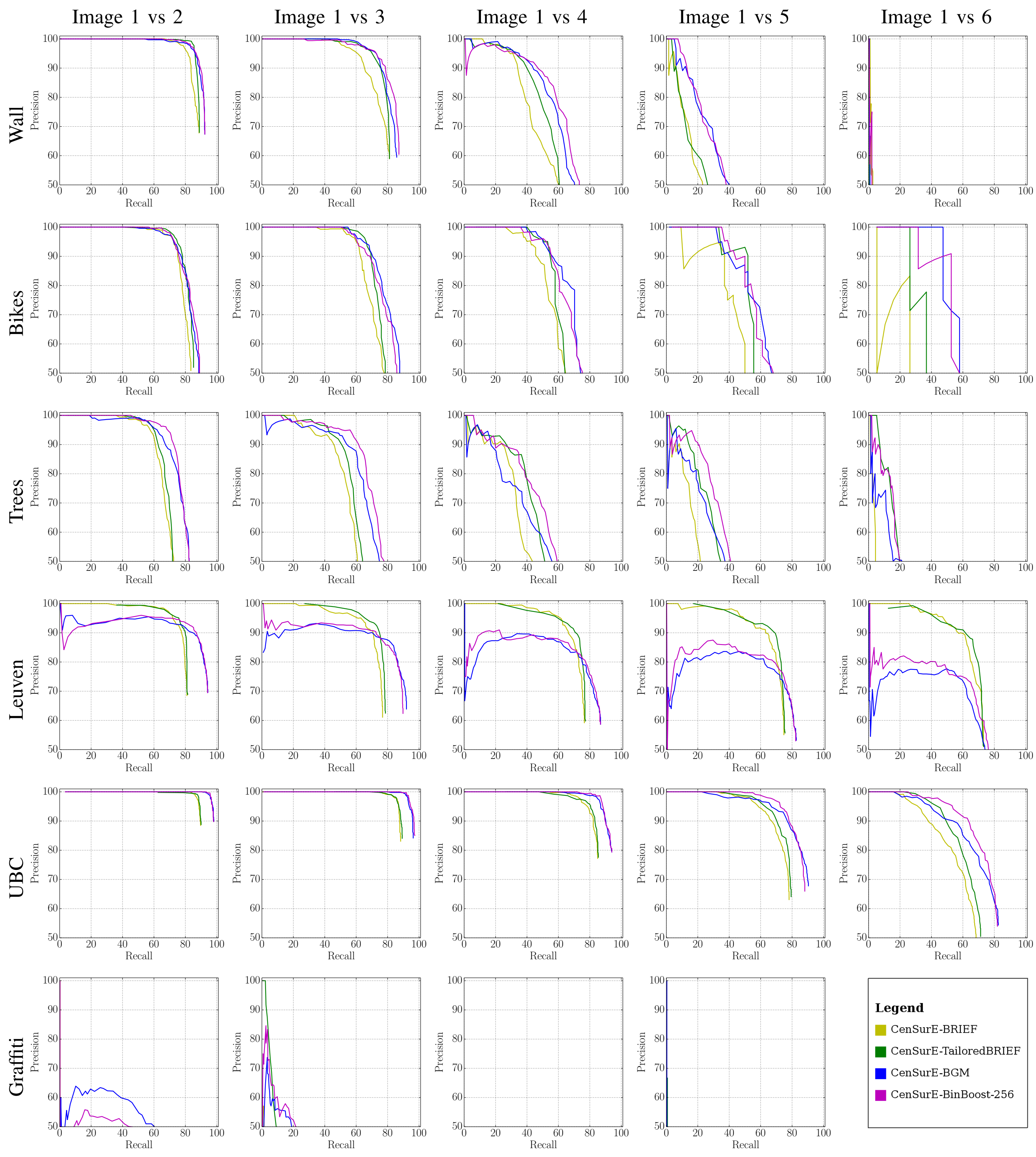


Fig. 6: Precision-recall curves for all datasets and images. Plots are organized by dataset (row) and query image number (column). Increasing query image number (left to right) typically denotes greater change in the image content relative to the reference image. Performance improvements over BRIEF is greatest for the Wall, Bikes, and Trees datasets, which primarily involve out-of-plane rotation (Wall) and blur (Bikes, Trees). Performance is generally the same for the Leuven and UBC datasets, which involve effects not simulated in training for TailoredBRIEF, including lighting change (Leuven) and significant image compression (UBC). Performance for all methods is weakest on the Graffiti dataset (similar for Bark, Boat), whose in-plane rotation requires an orientation-aware descriptor.

Dataset	**Method**	**Area Under the Precision-Recall Curve**				
		1 vs 2	1 vs 3	1 vs 4	1 vs 5	1 vs 6
Wall	BRIEF	0.873	0.771	0.529	0.227	0.037
	Proposed	**0.884**	**0.793**	**0.555**	**0.238**	0.030
Bikes	BRIEF	0.808	0.735	0.627	0.506	0.279
	Proposed	**0.830**	**0.764**	**0.645**	**0.576**	**0.347**
Trees	BRIEF	0.693	0.572	0.407	0.227	0.082
	Proposed	0.696	**0.597**	**0.460**	**0.319**	**0.188**
Leuven	BRIEF	0.796	0.741	0.736	0.708	0.685
	Proposed	0.801	**0.767**	0.745	**0.721**	0.689
UBC	BRIEF	0.895	0.880	0.840	0.751	0.607
	Proposed	0.897	**0.890**	0.844	**0.769**	**0.653**
Graffiti	BRIEF	**0.066**	0.085	0.000	0.017	0.000
	Proposed	0.056	**0.141**	0.000	0.016	0.000

TABLE III: Area under the Precision-Recall curves for all image pairs in Figure 6. Results shown correspond to BRIEF and the proposed method, TailoredBRIEF, with a 7-pixel box filter. The areas have a maximum achievable value of 1 and are arranged in columns sorted by image pair (e.g. 1 vs 2). The greater of the two areas is shown in bold for each combination of dataset and image pair for differences of 1% or more. TailoredBRIEF yields the greater area measure in 19 of 30 trials, while BRIEF yields the greater area measure in only 1 of 30 trials. See Figure 6 for additional discussion of the performance on these datasets.

and increases the overall feature matching accuracy as shown on a standard dataset.

References

[1] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "BRIEF: Binary robust independent elementary features," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2010, pp. 778–792.

[2] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," in *International Conference on Computer Vision (ICCV)*. IEEE, 2011, pp. 2564–2571.

[3] V. Lepetit, P. Lagger, and P. Fua, "Randomized trees for real-time keypoint recognition," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 2. IEEE, 2005, pp. 775–781.

[4] V. Lepetit and P. Fua, "Keypoint recognition using randomized trees," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 28, no. 9, pp. 1465–1479, 2006.

[5] M. Ozuysal, P. Fua, and V. Lepetit, "Fast keypoint recognition in ten lines of code," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2007, pp. 1–8.

[6] M. Ozuysal, M. Calonder, V. Lepetit, and P. Fua, "Fast keypoint recognition using random ferns," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 32, no. 3, pp. 448–461, 2010.

[7] M. Calonder, V. Lepetit, and P. Fua, "Keypoint signatures for fast learning and recognition," in *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer, 2008, pp. 58–71.

[8] M. Calonder, V. Lepetit, M. Ozuysal, T. Trzcinski, C. Strecha, and P. Fua, "BRIEF: Computing a Local Binary Descriptor Very Fast," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 7, pp. 1281–1298, 2012.

[9] V. L. T. Trzcinski, M. Christoudias and P. Fua, "Boosting Binary Keypoint Descriptors," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013.

[10] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 430–443, 2006.

[11] S. Leutenegger, M. Chli, and R. Y. Siegwart, "BRISK: Binary robust invariant scalable keypoints," in *International Conference on Computer Vision (ICCV)*. IEEE, 2011, pp. 2548–2555.

[12] H. Bay, T. Tuytelaars, and L. Van Gool, "SURF: Speeded up robust features," *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 404–417, 2006.

[13] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. Gool, "A comparison of affine region detectors," *International Journal of Computer Vision (IJCV)*, vol. 65, no. 1, pp. 43–72, 2005.

[14] M. Agrawal, K. Konolige, and M. R. Blas, "CenSurE: Center surround extremas for realtime feature detection and matching," in *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer, 2008, pp. 102–115.

[15] V. L. T. Trzcinski, M. Christoudias and P. Fua, "Learning Image Descriptors with the Boosting-Trick," in *Advances in Neural Information Processing Systems (NIPS)*, 2012.