MITSUBISHI ELECTRIC RESEARCH LABORATORIES
http://www.merl.com

Parameter Learning for Improving Binary Descriptor Matching

Sankaran, B.; Ramalingam, S.; Taguchi, Y.

TR2016-132 October 2016

Abstract

Binary descriptors allow fast detection and matching algorithms in computer vision problems. Though binary descriptors can be computed at almost two orders of magnitude faster than traditional gradient based descriptors, they suffer from poor matching accuracy in challenging conditions. In this paper we propose three improvements for binary descriptors in their computation and matching that enhance their performance in comparison to traditional binary and nonbinary descriptors without compromising their speed. This is achieved by learning some weights and threshold parameters that allow customized matching under some variations such as lighting and viewpoint. Our suggested improvements can be easily applied to any binary descriptor. We demonstrate our approach on the ORB (Oriented FAST and Rotated BRIEF) descriptor and compare its performance with the traditional ORB and SIFT descriptors on a wide variety of datasets. In all instances, our enhancements outperform standard ORB and is comparable to SIFT.

IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)

This work may not be copied or reproduced in whole or in part for any commercial purpose. Permission to copy in whole or in part without payment of fee is granted for nonprofit educational and research purposes provided that all such whole or partial copies include the following: a notice that such copying is by permission of Mitsubishi Electric Research Laboratories, Inc.; an acknowledgment of the authors and individual contributions to the work; and all applicable portions of the copyright notice. Copying, reproduction, or republishing for any other purpose shall require a license with payment of fee to Mitsubishi Electric Research Laboratories, Inc. All rights reserved.

Copyright © Mitsubishi Electric Research Laboratories, Inc., 2016
201 Broadway, Cambridge, Massachusetts 02139

Parameter Learning for Improving Binary Descriptor Matching

Bharath Sankaran, Srikumar Ramalingam, and Yuichi Taguchi

Abstract— **Binary descriptors allow fast detection and matching algorithms in computer vision problems. Though binary descriptors can be computed at almost two orders of magnitude faster than traditional gradient based descriptors, they suffer from poor matching accuracy in challenging conditions. In this paper we propose three improvements for binary descriptors in their computation and matching that enhance their performance in comparison to traditional binary and non-binary descriptors without compromising their speed. This is achieved by learning some weights and threshold parameters that allow customized matching under some variations such as lighting and viewpoint. Our suggested improvements can be easily applied to any binary descriptor. We demonstrate our approach on the ORB (Oriented FAST and Rotated BRIEF) descriptor and compare its performance with the traditional ORB and SIFT descriptors on a wide variety of datasets. In all instances, our enhancements outperform standard ORB and is comparable to SIFT.**

I. Introduction

Feature matching is a key component in several vision tasks such as object detection, object recognition, and structure-from-motion. State-of-the-art approaches to these problems rely on robustly matching descriptors that are costly to compute and match. This led to the advent of binary descriptors that are fast to compute and match. These are particularly crucial in mapping and localization tasks for autonomous navigation, where computational speed is critical. The computational speed and efficiency of binary descriptors are attributed to the following properties:

- Binary descriptors are computed by pairwise pixel intensity comparisons in a given image patch. Pixel comparisons are faster to compute than gradient operations, which are used in gradient based descriptors such as SIFT and SURF.
- Binary descriptors are matched using Hamming distance metrics that are faster to compute than the L2 metric used for gradient based descriptors.

ORB [1] is one of the binary descriptors that is two orders of magnitude faster than SIFT [2], without losing much on the performance with respect to keypoint matching. We propose several extensions to improve the performance of ORB descriptor by learning a few parameters, without making the descriptor computation any slower. Our approach is readily extensible to any binary descriptor, since all binary descriptors only vary by the way the pairwise pixels are sampled from a given image patch. For instance BRIEF [3] descriptor uses random pairs. In BRISK [4] the sampling pattern is hand crafted, whereas in ORB [1] and FREAK [5] we use pairs that are learned from data. In order to explain our approach effectively we provide a brief introduction to the computation of the ORB descriptor below.

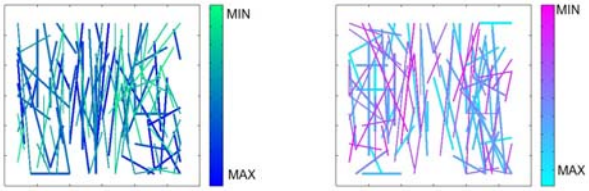

Fig. 1. The figure shows the relative weights assigned to various binary comparisons of pixel intensities (shown as end points of line segments) used in generating the binary feature vectors. Out of the 256 binary comparisons, only the top 50 and bottom 50 weights are shown for visualization. On the left, we show the learned weights for a dataset with changes in view point and rotation. On the right, we show the learned weights for a dataset with changes in lighting and view point.

A. The Basic ORB Computation and Matching

Let us consider the problem of matching two keypoints $k_1(x_1, y_1)$ and $k_2(x_2, y_2)$. Consider a small patch of dimension $p \times p$ centered at these keypoints. ORB considers 256 pairs of pixels $(p_i, q_i), i = \{1, ..., 256\}$ in the patch and performs simple binary tests. An example of pairs selected for such binary tests are shown in Figure 1. These 256 different pairs are chosen based on a greedy algorithm that looks for highly informative pairs learned from PASCAL data set [6]. Let $I(p)$ denote the intensity value of the pixel p. The binary test performed on the intensity values is

$$b_i = \begin{cases} 1 & \text{if } I(p_i) > I(q_i), \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

The entire 256×1 feature vector $f_1 \in \mathbb{B}^{256}$ for the patch at keypoint k_1 is given by $f_1 = [b_1, ..., b_{256}]^T$. Two keypoints k_1 and k_2 are matched by looking at the Hamming distance $\mathcal{H}(f_1, f_2)$ between the feature vectors constructed at these keypoints.

$$\mathcal{H} = \sum_{i=1}^{256} |f_1(i) - f_2(i)| \quad (2)$$

II. Related Work

Fast similarity search has garnered significant attention in the recent years to enable real-time applications in various kinds of data like image, video, and audio. In image matching in particular, the advent of a wide variety of binary descriptors [3], [1], [4], [5] has led to substantial gains in matching speeds for real-time applications without a huge compromise on performance. Though binary descriptors perform similar to traditional descriptors in easy matching cases, they are

This work was supported by Mitsubishi Electric Research Labs (MERL)

B. Sankaran is with the Computational Learning and Motor Control Lab, University of Southern California, Los Angeles, CA 90089, bsankara@usc.edu

S. Ramalingam and Y. Taguchi are with Mitsubishi Electric Research Labs, Cambridge, MA 02139, ramalingam@merl.com,taguchi@merl.com

sensitive to challenging conditions such as lighting, view point, and scale variations.

In order to improve the matching accuracy of binary descriptors there have been earlier approaches that have proposed the idea of learning weights. The general idea tries to improve matching accuracy by learning weights such that the Hamming distance of correct matches is lower in comparison to wrong matches. Fan et al. [7] demonstrate a lookup table approach to compute fast weighted Hamming distances that demonstrate equivalent matching speeds in comparison to the standard Hamming distance. Similarly, weighted Hamming approaches have been used for feature ranking. For instance, Zhang et al. [8] introduce a dynamic bit level weighting method for ranking binary codes to reduce the number of instances that receive the same Hamming distance. This approach to binary code ranking, though more discriminative than a regular weighted Hamming matcher, is not computationally efficient. In our approach to weighted matching of binary descriptors we avoid compromising on matching speed by performing a forward and reverse consistency check, i.e., a source to target and target to source match. We later prune these matches to get the final set of accurate matches.

Comparing single pixels in binary descriptors causes the representation to be sensitive to noise and minor image distortions. Patch based approaches have been proposed in order to be robust. For instance LATCH [9] is a novel binary descriptor that focuses on comparing mini-patches in order to increase the spatial support of binary tests. To construct the descriptor they use triplets of mini-patches instead of pairs. The set of triplet of patches are learned from data around a given keypoint. The LATCH descriptor uses a predefined set of 512 triplets where similarity between patches are measured with sum of squared differences (SSD). In contrast, in our approach we let the threshold parameter handle the variation in noise and image distortion that the binary descriptor is susceptible to.

Most methods that focus on improving descriptor matching accuracy have primarily focused on learning better similarity metrics. Apart from weighted matching of descriptors, there are also approaches that focuses on improving the expressiveness of the descriptor. Zagoruyko and Komodakis [10] train Convolutional Neural Networks (CNN) for matching image patches. Their network is a 2 channel CNN with the two top layers consisting of single channel fully connected layers. This architecture was tested in a siamese, pseudo-siamese, and non-siamese framework. This approach to image matching was shown to outperform conventional descriptor based image matching approaches. Though these approaches have higher matching accuracy, they have lower computational and matching speeds in comparison to binary feature matching approaches.

We also show in Section IV that our learned parameters can generalize to other datasets and still outperform the current state of the art approaches.

III. Problem Formulation

A. Proposed Extensions

Based on the basic binary descriptor computation and matching framework, we propose three extensions to the descriptor computation and performance. We demonstrate these extensions over the standard ORB descriptor, while using the same pairs as the standard ORB descriptor. In the first extension we propose to use a weighted Hamming distance matcher instead of the one that considers uniform weights for all the 256 different binary tests. We learn a weight vector $\mathbf{w} = [w_1, ..., w_{256}]^T$ and match two keypoints using weighted Hamming distance:

$$\mathcal{H}_w = \sum_{i=1}^{256} w_i |f_1(i) - f_2(i)| \tag{3}$$

These weights can be learned as shown in Section III-B.

The binary tests are usually performed after smoothing the images. Despite this, the binary tests are sensitive to lighting and viewpoint variations. We propose to use a threshold vector $\mathcal{T} \in \mathbb{R}$ in the binary tests as follows:

$$b_i = \begin{cases} 1 & \text{if } I(p_i) - I(q_i) > \mathcal{T}, \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

Learning this threshold is more involved than learning weights. We explain this second extension in Section III-C. Then as the third extension, we propose to learn both the weights and threshold, which is explained in Section III-D.

B. Learning The Weights

Given some training data $\mathcal{D} = \{x_i, y_i\}, i = \{1, ..., n\}$, we would like to learn some weights so that the weighted Hamming distance for correct matches is smaller than the distance for the incorrect matches. Here, x_i and y_i are 256×1 binary vectors for n correct keypoint matches. We formulate the problem of learning the weights using the standard max-margin network learning [11] in the following manner:

$$\begin{aligned} \min_{\mathbf{w},b,\epsilon} \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{n}\epsilon_i \\ \text{s.t.} \\ \mathcal{H}_w(x_i, y_i) + b \leq -1 + \epsilon_i \\ \mathcal{H}_w(x_i, y_j) + b \geq 1 - \epsilon_i, \forall j \neq i \\ \epsilon_i \geq 0 \end{aligned}$$

Here ϵ_i is the slack variable and C is the soft margin parameter in standard max-margin network learning algorithms. $\mathbf{w}$ is the set of weights that we learn and b is the bias term. To learn the weights we use two negative cases of $\mathcal{H}_w(x_i, y_i)$ for every positive case of $\mathcal{H}_w(x_i, y_i)$. The positive case of $\mathcal{H}_w(x_i, y_i)$ is the correct match between source and target descriptor, x_i and y_i. This is given from ground truth data. The two negative cases used for learning are the target descriptors which have the smallest and second smallest Hamming distance to the source descriptor, where $j \neq i$.

C. Learning The Threshold

The optimization problem for threshold learning can be formulated as follows. Given some training data $\mathcal{D} = \{d_{i1}, d_{i2}\}, i = \{1, ..., n\}$, we would like to learn a threshold $\mathcal{T} \in \mathbb{R}$. Here, d_{i1} an d_{i2} refer to 256×2 matrices storing the intensity values for 256 pairs of pixels used for building the binary descriptors at two different matching keypoints. We formulate the learning problem as shown below:

$$\begin{aligned}
&\min_{\mathcal{T},b,\epsilon} \sum_{i=1}^{n} \epsilon_i \\
&\text{s.t.} \\
&\mathcal{H}(x_i, y_i) + b \leq -1 + \epsilon_i \\
&\mathcal{H}(x_i, y_j) + b \geq 1 - \epsilon_i, \forall j \neq i \\
x_i(k) = &\mathop{\arg\min}_{x_i(k)\in\{0,1\}} x_i(k)(d_{i1}(k,1) - d_{i1}(k,2) - \mathcal{T}) \\
y_i(k) = &\mathop{\arg\min}_{y_i(k)\in\{0,1\}} y_i(k)(d_{i2}(k,1) - d_{i2}(k,2) - \mathcal{T}) \\
&\mathcal{T} \geq -256.0 \\
&\mathcal{T} \leq 256.0
\end{aligned}$$

The threshold $\mathcal{T}$ takes only integer values, because the error does not change for any intermediate real values. We can do a brute-force search for different threshold values.

D. Combined Weight and Threshold Learning

To combine both the weight and threshold learning we formulate the optimization as follows. Given some training data $\mathcal{D} = \{d_{i1}, d_{i2}\}, i = \{1, ..., n\}$, we would like to learn the weight vector $\mathbf{w} \in \mathbb{R}^{256}$ and the threshold $\mathcal{T} \in \mathbb{R}$. Here, d_{i1} an d_{i2} refer to 256×2 matrices storing the intensity values for 256 pairs of pixels used for building the binary descriptors at two different matching keypoints. We formulate the learning problem as shown below:

$$\begin{aligned}
&\min_{\mathbf{w},b,\mathcal{T},\epsilon} \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{n} \epsilon_i \\
&\text{s.t.} \\
&\mathcal{H}_w(x_i, y_i) + b \leq -1 + \epsilon_i \\
&\mathcal{H}_w(x_i, y_j) + b \geq 1 - \epsilon_i, \forall j \neq i \\
x_i(k) = &\mathop{\arg\min}_{x_i(k)\in\{0,1\}} x_i(k)(d_{i1}(k,1) - d_{i1}(k,2) - \mathcal{T}) \\
y_i(k) = &\mathop{\arg\min}_{y_i(k)\in\{0,1\}} y_i(k)(d_{i2}(k,1) - d_{i2}(k,2) - \mathcal{T}) \\
&\mathcal{T} \geq -256.0 \\
&\mathcal{T} \leq 256.0
\end{aligned}$$

The above problem is non-convex and it is difficult to get an optimal solution. We can fix the threshold $\mathcal{T}$ to different integer values and this makes the optimization problem convex, similar to the weight learning method explained in section III-B. The constraint involving $\arg\min$ leads to the non-convexity. Exploiting the integer nature of the threshold values, it can be learned via a brute force search.

IV. Experiments and Observations

We evaluated the descriptor matching accuracy of our method and other approaches on four different datasets: the oxford affine covariant regions dataset[1] first introduced in [12], the AMOS (Archive of Many Outdoor Scenes) dataset[2] [13], the KITTI stereo dataset[3] [14], and the CAVE dataset[4] [15]. We evaluated our approach by comparing the RANSAC refined inlier ratio with other descriptors in the Oxford, AMOS and CAVE datasets. For oxford and AMOS datasets we also evaluate our approach with models trained on other datasets like Cornell Multiview dataset[5] [16] and the KITTI stereo dataset. For CAVE we only evaluate SIFT and ORB against a model trained on the KITTI stereo dataset. Finally for KITTI, we only compare our approach against the standard ORB.

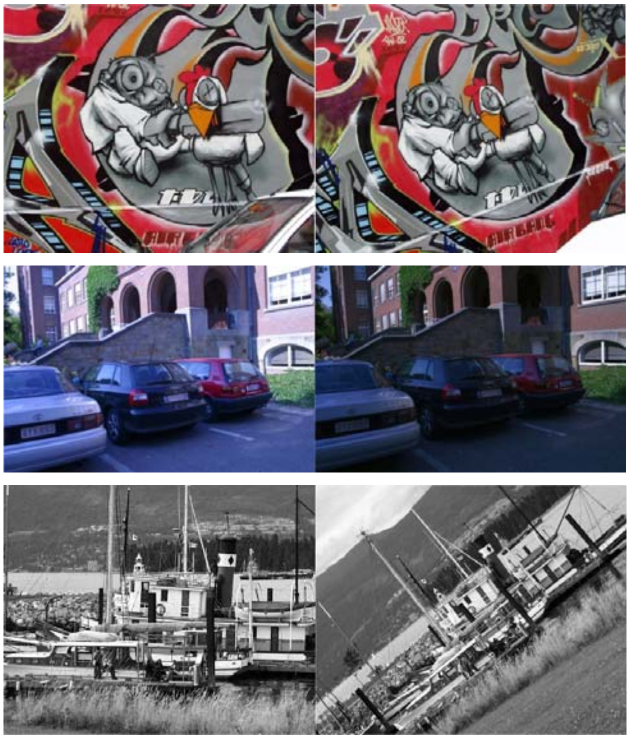

Fig. 2. **Oxford dataset**. This dataset consists of several groups, where each group consist of images taken under certain variations such as scale, lighting, viewpoint, etc. The top row shows images from the graffiti subgroup which has both scale and viewpoint changes.The middle row shows images from the light subgroup that has images with lighting variations. The last row shows images from the boat subset that contains images with scale, viewpoint and rotation variations.

A. Datasets

1) Oxford dataset: The oxford dataset has images with scale, viewpoint and lighting variations. A few examples from this dataset are shown in Figure 2. In the oxford dataset, the ground truth homographies between a single source image and multiple target images are provided along with the dataset. We use this information to find all matching keypoint pairs in source and target images across the dataset.

[1] http://www.robots.ox.ac.uk/~vgg/data/data-aff.html

[2] http://amos.cse.wustl.edu/dataset

[3] http://www.cvlibs.net/datasets/kitti/eval_stereo_flow.php

[4] http://www.cs.columbia.edu/CAVE/databases/multispectral/

[5] http://www.cs.cornell.edu/projects/p2f/

From these keypoint pairs, we extract the threshold optimized descriptor differences and use this information to train our max-margin model to learn the weights as discussed in Section III-D. During the testing phase, we independently detect 500 keypoints in source and target images (without using the ground truth homography). There is no special reason for chosing 500 keypoints in our experiments, except that feature detectors like ORB are very efficient in extracting 500 keypoints from VGA images in many real-time applications. We match these keypoints using our algorithm and report the RANSAC refined inlier ratio. To compare with other methods, we compare our results with other descriptors like SIFT and vanilla ORB. We also train models on other datasets like KITTI and Cornell and test them on the oxford dataset.

2) The AMOS dataset: The AMOS dataset is a publicly available archive of outdoor scenes taken from fixed cameras over multiple days. From this dataset we assembled an illumination variant dataset with images taken from the same camera position over multiple times of the day over multiple days of the month. An example of such a sequence of a single day is shown in Figure 3. We use models trained on the oxford lighting data subset, the KITTI dataset and the Cornell 3D dataset to evaluate our approach on the AMOS dataset.

Similar to the oxford evaluation, for AMOS we independently detect 500 keypoints in the source and target image and match them using SIFT, ORB and our approach. We compute the RANSAC refined inlier ratio for comparison with other approaches like SIFT and ORB.

3) The KITTI dataset: The KITTI dataset is an autonomous driving platform dataset [14] that was introduced as a standard benchmark for computer vision problems like stereo, optical flow, visual odometry, 3D object detection and 3D tracking. A sample stereo pair from the dataset is show in Figure 4. The KITTI dataset provides a standard train and test set. Ground truth disparity for all stereo pairs are provided with the dataset. In the training phase we extract threshold optimized descriptors from matching keypoint pairs in stereo images using the disparity information provided with the dataset. For evaluation we compare the RANSAC refined inlier ratio for stereo matches computed using ORB and our approach.

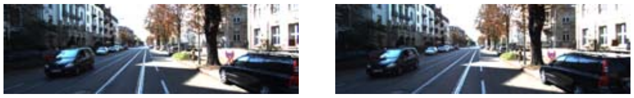

Fig. 4. **KITTI dataset**. The left and right image of a stereo pair are shown in the sample images above.

4) CAVE dataset: The CAVE dataset contains multispectral images that are used to emulate a GAP camera. The images are a wide variety of real-world materials and objects. Examples are shown in Figure 5. We only evaluate some subsets of the CAVE dataset where ORB keypoints could be detected in both source and target images.

5) The Cornell multiview dataset: The Cornell dataset is a multiview city scale dataset that contains images taken from different viewpoints, different locations and different cameras. The dataset was first introduced in [16]. The training data comes with ground truth bundler [17] data that can be used to get the pixel location of 3D points seen by multiple cameras. We use this information to learn a model on points extracted from multiple image/pixel pair combinations across the dataset. In our experiments we specifically train a model on the Dubrovnik dataset. We use the model trained on this dataset to evaluate matching on the Oxford and AMOS dataset.

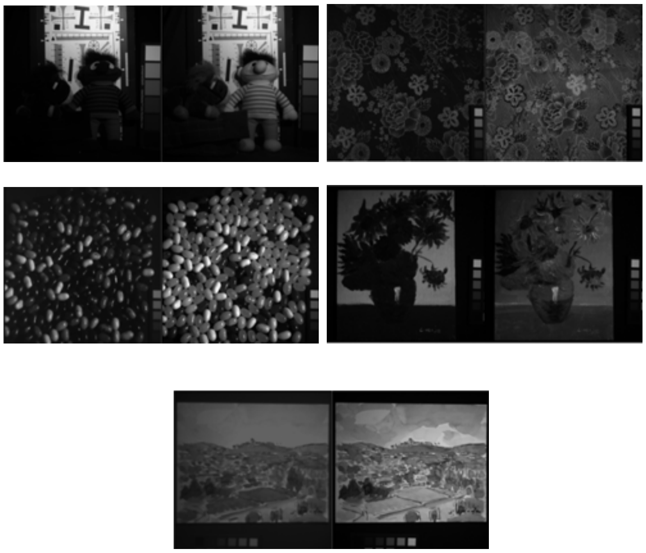

Fig. 5. **CAVE dataset**. Images from the multispectral CAVE dataset are arranged according to the following subsets from the top to the bottom row: chart and stuffed toy (CST), cloth (CL), jelly beans (JB), oil painting (OP), and water color (WC).

B. Matching results with threshold learning and parameter learning - Oxford dataset

In this evaluation we detect the same number of keypoints in SIFT, the threshold and the non-threshold version of the ORB detector. The inlier percentage is the mean inlier percentage across all target images from the Oxford dataset. Each subset of the Oxford dataset has 5 target images. $\mathcal{H}_w(x_i, y_j)$ is ORB matched with the weighted Hamming distance without any threshold optimizations. $\mathcal{H}_\tau(x_i, y_j)$ is the threshold optimized ORB, matched with a vanilla Hamming distance matcher. $\mathcal{H}_{w\tau}(x_i, y_j)$ is the model that was trained on the threshold optimized ORB with the weighted Hamming distance matcher. $\mathcal{H}_{w\tau}(x_i, y_j)$ trained on data from the Oxford dataset, the Cornell dataset and the KITTI dataset are denoted as Oxford $\mathcal{H}_{w\tau}(x_i, y_j)$, Cornell $\mathcal{H}_{w\tau}(x_i, y_j)$ and KITTI $\mathcal{H}_{w\tau}(x_i, y_j)$ respectively. The best results are **boldfaced** and the second best results are **boldfaced** in blue color. We show the accuracy in Table I. The weighted Hamming distance provides better accuracy compared to naive Hamming distance. In general, the weighted Hamming distance along with the threshold provides the best accuracy. The Cornell and KITTI datasets are larger than the Oxford dataset. The weights and thresholds learned using these datasets also provide good accuracy on the oxford dataset.

From our experiments we can see that our learned model outperforms the vanilla ORB descriptor in all datasets and

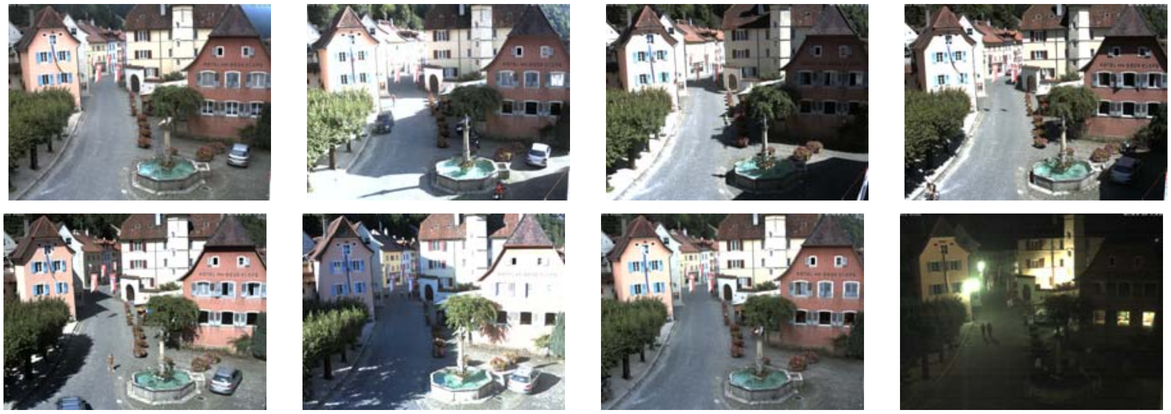

Fig. 3. **AMOS dataset**. The images show the progression of an entire day from a single view point. The leftmost image in the first row is the first image taken at the start of the day. The rightmost image in the second row is the last image taken at night.

TABLE I

OXFORD DATASET EVALUATION RESULTS

	Graffiti	Boat	Light
SIFT	**66.2**%	**58.52**%	71.56%
ORB	55.43%	47.28%	66.83%
ORB ($\mathcal{H}_w(x_i,y_j)$)	61.14%	52.93%	76.91%
ORB ($\mathcal{H}_\tau(x_i,y_j)$)	55.43%	47.28%	66.83%
ORB (Oxford $\mathcal{H}_{w\tau}(x_i,y_j)$)	59.55%	53.64%	**82.89**%
ORB (Cornell $\mathcal{H}_{w\tau}(x_i,y_j)$)	**63.31**%	54.24%	77.57%
ORB (KITTI $\mathcal{H}_{w\tau}(x_i,y_j)$)	60.77%	**55.10**%	**82.04**%

performs comparably (and sometimes much better) than the SIFT descriptor.

C. Matching results with threshold learning and parameter learning - AMOS Lighting dataset

Similar to the Oxford evaluation, we independently detect 500 keypoints in the source and target images and match them. We evaluate our approach, i.e., learning the threshold optimized weighted Hamming matcher $\mathcal{H}_{w\tau}(x_i,y_j)$ against SIFT and ORB. We trained three independent models trained on the Oxford dataset, Cornell dataset and KITTI dataset respectively. These are the same models used in the Oxford evaluation. The inlier percentage is the mean inlier percentage across all target images from the AMOS dataset. The AMOS dataset has 20 target images. As we can once again see, our approach outperforms both SIFT and ORB. We show the results of our approach in Table II. The images are captured from the same camera position. By learning weights and thesholds from large datasets such as Cornell and KITTI, we observe an improvement in the accuracy of the matching algorithm.

TABLE II

AMOS DATASET EVALUATION RESULTS

	AMOS dataset
SIFT	45.01%
ORB	45.03%
Oxford $\mathcal{H}_{w\tau}(x_i,y_j)$	45.86%
Cornell $\mathcal{H}_{w\tau}(x_i,y_j)$	46.29%
Kitti $\mathcal{H}_{w\tau}(x_i,y_j)$	**46.97**%

D. Matching results with threshold learning and parameter learning - Kitti Stereo dataset

For the KITTI evaluation we independently detect 1500 keypoints on source and target images and match them. We evaluate our approach against the vanilla ORB descriptor and matcher. We perform the evaluation on the test set provided along with the dataset.The result of the model evaluation for descriptor matching is shown in Table III below. The inlier percentage is the mean inlier percentage across all stereo pairs from the KITTI test set. The KITTI test set has 200 stereo pairs. We observe that by learning weights and thesholds, we achieve an increase in the accuracy of the matching.

TABLE III

KITTI STEREO DATASET EVALUATION RESULTS

	Stereo dataset ($\mathcal{H}_{w\tau}(x_i,y_j)$)
ORB	56.21%
Kitti $\mathcal{H}_{w\tau}(x_i,y_j)$	**59.68**%

As it can be noted our approach outperforms the regular ORB descriptor. In our evaluation we also noticed that our threshold optimized approach, does comparably to ORB on the simple cases and considerably much better on the harder cases.

E. Matching results with threshold learning and parameter learning - CAVE Multispectral dataset

Similar to the Oxford evaluation, we independently detect 500 keypoints in the source and target images and match them. We evaluate our threshold optimized weighted Hamming matcher $\mathcal{H}_{w\tau}(x_i,y_j)$ against SIFT and ORB. We use the model trained on the KITTI dataset for our evaluation. The inlier percentage is the mean inlier percentage across all target images from the CAVE dataset. The CAVE dataset has 30 target images per subset. The best results are **boldfaced** and the second best results are **boldfaced** in blue color.

We show the results in Table IV. In the multispectral dataset we observe that the model trained on the KITTI

TABLE IV

CAVE DATASET EVALUATION RESULTS

	CST	CL	OP	JB	WC
SIFT	94.51%	**90.71**%	**72.34**%	**84.73**	**84.75**%
ORB	**96.99**%	80.45%	71.83	75.06%	**77.90**%
$\mathcal{H}_{w\tau}(x_i,y_j)$	**99.32**%	**81.47**%	**81.08**%	**75.96**%	77.18%

dataset either significantly outperforms both SIFT and ORB or matches the performance of vanilla ORB.

F. Discussion

Through our experiments we made the following observations about the characteristics of our threshold optimized weighted Hamming matcher.

- Our matcher had fewer false positives as compared to the traditional ORB descriptor and matcher. This led to a lower number of overall matches but a higher number of accurate matches. Hence the RANSAC refined inlier ratio was higher.
- Our results show that the weights can be easily learned on a large publicly available dataset to allow better generalization.

Finally, since our approach is applicable to any binary descriptor, our approach can also be applied other binary descriptors like FREAK, BRISK and BRIEF.

V. IMPLEMENTATION DETAILS

We implemented our algorithms using OpenCV. The weights for the weighted Hamming distance matcher were learned with a linear SVM using the LIBSVM library [18].

We exploit the integer nature of the threshold values by using brute force search to solve for the threshold values. The threshold adjusted descriptors are computed using a speed optimized implementation similar to the OpenCV ORB implementation.

VI. CONCLUSION AND FUTURE WORK

We have demonstrated an approach to learn weights and thresholds to improve descriptor matching performance for binary descriptors. We demonstrated our approach on the ORB descriptor, but our method is readily applicable to other binary descriptors. For the threshold optimization, we present a search algorithm by exploiting the fact that we only need to consider the integer threshold values. In the future, we plan to develop an automatic algorithm to optimize and solve the non-convex threshold learning problem. The threshold based Hamming distance without the weights runs at the same speed as traditional ORB, since there are no added computations other than bit comparisons. The weighted Hamming distance (under it's current non-optimized implementation) is at least 15x slower than the regular Hamming distance. However, there are approaches to make it as fast as regular Hamming distance using a lookup table. Given that our current implementation is in OpenCV, we intend to make our code available.

REFERENCES

[1] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to sift or surf," in *International Conference on Computer Vision (ICCV)*, 2011.

[2] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision (IJCV)*, vol. 60, no. 2, pp. 91–110, Nov. 2004.

[3] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "BRIEF: Binary robust independent elementary features," in *European Conference on Computer Vision (ECCV)*. Berlin, Heidelberg: Springer-Verlag, 2010, pp. 778–792.

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

[5] A. Alahi, R. Ortiz, and P. Vandergheynst, "FREAK: Fast retina keypoint," in *IEEE Conference on Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2012, pp. 510–517.

[6] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results," http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

[7] B. Fan, Q. Kong, X. Yuan, Z. Wang, and C. Pan, "Learning weighted hamming distance for binary descriptors." in *ICASSP*. IEEE, 2013, pp. 2395–2399.

[8] L. Zhang, Y. Zhang, J. Tang, K. Lu, and Q. Tian, "Binary code ranking with weighted hamming distance," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2013, pp. 1586–1593.

[9] G. Levi and T. Hassner, "LATCH: learned arrangements of three patch codes," in *Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2016.

[10] S. Zagoruyko and N. Komodakis, "Learning to compare image patches via convolutional neural networks," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[11] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun, "Large margin methods for structured and interdependent output variables," *Journal of Machine Learning Research (JMLR)*, 2005.

[12] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. V. Gool, "A comparison of affine region detectors," *International Journal on Computer Vision (IJCV)*, vol. 65, no. 1-2, pp. 43–72, Nov. 2005.

[13] N. Jacobs, N. Roman, and R. Pless, "Consistent temporal variations in many outdoor scenes," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2007, pp. 1–6.

[14] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite," in *IEEE Conference on Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[15] F. Yasuma, T. Mitsunaga, D. Iso, and S. Nayar, "Generalized Assorted Pixel Camera: Post-Capture Control of Resolution, Dynamic Range and Spectrum," Tech. Rep., Nov 2008.

[16] Y. Li, N. Snavely, and D. Huttenlocher, "Location recognition using prioritized feature matching," in *European Conference on Computer Vision (ECCV)*, 2010, pp. 791–804.

[17] N. Snavely, S. M. Seitz, and R. Szeliski, "Photo tourism: Exploring photo collections in 3d," *ACM Transactions on Graphics*, vol. 25, no. 3, pp. 835–846, Jul. 2006.

[18] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, pp. 27:1–27:27, 2011.

Acknowledgments: We thank Jay Thornton, Arvind Raghunathan, Gim Hee Lee, Alan Sullivan, anonymous reviewers, and the area chair for useful feedback and support.