TILDE: A Temporally Invariant Learned DEtector

Yannick Verdie\textsuperscript{1,*} Kwang Moo Yi\textsuperscript{1,*} Pascal Fua\textsuperscript{1} Vincent Lepetit\textsuperscript{2}
\textsuperscript{1}Computer Vision Laboratory, École Polytechnique Fédérale de Lausanne (EPFL)
\textsuperscript{2}Institute for Computer Graphics and Vision, Graz University of Technology
\{yannick.verdie, kwang.yi, pascal.fua\}@epfl.ch, lepetit@icg.tugraz.at

Abstract

We introduce a learning-based approach to detect repeatable keypoints under drastic imaging changes of weather and lighting conditions to which state-of-the-art keypoint detectors are surprisingly sensitive. We first identify good keypoint candidates in multiple training images taken from the same viewpoint. We then train a regressor to predict a score map whose maxima are those points so that they can be found by simple non-maximum suppression.

As there are no standard datasets to test the influence of these kinds of changes, we created our own, which we will make publicly available. We will show that our method significantly outperforms the state-of-the-art methods in such challenging conditions, while still achieving state-of-the-art performance on untrained standard datasets.

1. Introduction

Keypoint detection and matching is an essential tool to address many Computer Vision problems such as image retrieval, object tracking, and image registration. Since the introduction of the Moravec, Förstner, and Harris corner detectors [27, 11, 15] in the 1980s, many others have been proposed [41, 10, 31]. Some exhibit excellent repeatability when the scale and viewpoint change or the images are blurred [26]. However, their reliability degrades significantly when the images are acquired outdoors at different times of day and in different weathers or seasons, as shown in Fig. 1. This is a severe handicap when attempting to match images taken in fair and foul weather, in the morning and evening, in winter and summer, even with illumination invariant descriptors [13, 39, 14, 43].

In this paper, we propose an approach to learn a keypoint detector that extracts keypoints which are stable under such challenging conditions and allow matching in situations as difficult as the one depicted by Fig. 1. To this end, we first introduce a simple but effective method to identify potentially stable points in training images. We then use them to train a regressor that produces a score map whose values are local maxima at these locations. By running it on new images, we can extract keypoints with simple non-maximum suppression. Our approach is inspired by a recently proposed algorithm [34] that relies on regression to extract cen-
terlines from images of linear structures. Using this idea for our purposes has required us to develop a new kind of regressor that is robust to complex appearance variation so that it can efficiently and reliably process the input images.

As in the successful application of Machine Learning to descriptors [5, 40] and edge detection [8], learning methods have also been used before in the context of keypoint detection [30, 37] to reduce the number of operations required when finding the same keypoints as handcrafted methods. However, in spite of an extensive literature search, we have only found one method [38] that attempts to improve the repeatability of keypoints by learning. This method focuses on learning a classifier to filter out initially detected keypoints but achieved limited improvement. This may be because their method was based on pure classification and also because it is non-trivial to find good keypoints to be learned by a classifier in the first place.

Probably as a consequence, there is currently no standard benchmark dataset designed to test the robustness of keypoint detectors to these kinds of temporal changes. We therefore created our own from images from the Archive of Many Outdoor Scenes (AMOS) [18] and our own panoramic images to validate our approach. We will use our dataset in addition to the standard Oxford [26] and EF [44] datasets to demonstrate that our approach significantly outperforms state-of-the-art methods in terms of repeatability. In the hope of spurring further research on this important topic, we will make it publicly available along with our code.

In summary, our contribution is threefold:

- We introduce a “Temporally Invariant Learned DDetector” (TILDE), a new regression-based approach to extracting feature points that are repeatable under drastic illumination changes causes by changes in weather, season, and time of day.

- We propose an effective method to generate the required training set of “good keypoints to learn.”

- We created a new benchmark dataset for evaluation of feature point detectors on outdoor images captured at different times and seasons.

In the remainder of this paper, we first discuss related work, give an overview of our approach, and then detail our regression-based approach. We finally present the comparison of our approach to state-of-the-art keypoint detectors.

2. Related Work

Handcrafted Keypoint Detectors An extraordinary large amount of work has been dedicated to developing ever more effective feature point detectors. Even though the methods that appeared in the 1980s [27, 11, 15] are still in wide use, many new ones have been developed since.

Learned Keypoint Detectors Although work on keypoint detectors were mainly focused on handcrafted methods, some learning based methods have already been proposed [30, 38, 16, 28]. With FAST, [30] introduced Machine Learning techniques to learn a fast corner detector. However, learning in their case was only aimed toward the speed up of the keypoint extraction process. Repeatability is also considered in the extended version FAST-ER [31], but it did not play a significant role. [38] trained the Wald-Boost classifier [36] to learn keypoints with high repeatability on a pre-aligned training set, and then filter out an initial set of keypoints according to the score of the classifier. Their method, called TaSK, is probably the most related to our method in the sense that they use pre-aligned images to build the training set. However, the performance of their method is limited by the initial keypoint detector used.

Recently, [16] proposed to learn a classifier which detects matchable keypoints for Structure-from-Motion (SfM) applications. They collect matchable keypoints by observing which keypoints are retained throughout the SfM pipeline and learn these keypoints. Although their method shows significant speed-up, they remain limited by the quality of the initial keypoint detector. [28] learns convolutional filters through random sampling and looking for the filter that gives the smallest pose estimation error when applied to stereo visual odometry. Unfortunately, their method is restricted to linear filters, which are limited in terms of flexibility, and it is not clear how their method can be applied to other tasks than stereo visual odometry.

We propose a generic scheme for learning keypoint detectors, and a novel efficient regressor specified for this task. We will compare it to state-of-the-art handcrafted methods as well as TaSK, as it is the closest method from the literature, on several datasets.
3. Learning a Robust Keypoint Detector

In this section, we first outline our regression-based approach briefly and then explain how we build the required training set. We will formalize our algorithm and describe the regressor in more details in the following section.

3.1. Overview of our Approach

Let us first assume that we have a set of training images of the same scene captured from the same point of view but at different seasons and different times of the day, such as the set of Fig. 2(a). Let us further assume that we have identified in these images a set of locations that we think can be found consistently over the different imaging conditions. We propose a practical way of doing this in Section 3.2 below. Let us call positive samples the image patches centered at these locations in each training image. The patches far away from these locations are negative samples.

To learn to find these locations in a new input image, we propose to train a regressor to return a value for each patch of a given size of the input image. These values should have a peaked shape similar to the one shown in Fig. 2(b) on the positive samples, and we also encourage the regressor to produce a score that is as small as possible for the negative samples. As shown in Fig. 2(c), we can then extract keypoints by looking for local maxima of the values returned by the regressor, and discard the image locations with low values by simple thresholding. Moreover, our regressor is also trained to return similar values for the same locations over the stack of images. This way, the regressor returns consistent values even when the illumination conditions vary.

3.2. Creating the Training Set

As shown in Fig. 3, to create our dataset of positive and negative samples, we first collected series of images from outdoor webcams captured at different times of day and different seasons. We identified several suitable webcams from the AMOS dataset [18]—webcams that remained fixed over long periods of time, protected from the rain, etc. We also used panoramic images captured by a camera located on the top of a building.

To collect a training set of positive samples, we first detect keypoints independently in each image of this dataset. We use SIFT [23], but other detectors could be considered as well. We then iterate over the detected keypoints, starting with the keypoints with the smallest scale. If a keypoint is detected at about the same location in most of the images from the same webcam, its location is likely to be a good candidate to learn.

In practice we consider that two keypoints are at about the same location if their distance is smaller than the scale estimated by SIFT and we keep the best 100 repeated locations. The set of positive samples is then made of the patches from all the images, including the ones where the keypoint was not detected, and centered on the average location of the detections.

This simple strategy offers several advantages: we keep only the most repeatable keypoints for training, discarding the ones that were detected only infrequently. We also introduce as positive samples the patches where a highly repeatable keypoint was missed. This way, we can focus on the keypoints that can be detected reliably under different conditions, and correct the mistakes of the original detector.

To create the set of negative samples, we simply extract patches at locations that are far away from the keypoints used to create the set of positive samples.

4. An Efficient Piece-wise Linear Regressor

In this section, we first introduce the form of our regressor, which is made to be applied to every patch from an image efficiently, then we describe the different terms of the proposed objective function to train for detecting keypoints.
reliably, and finally we explain how we optimize the parameters of our regressor to minimize this objective function.

### 4.1. A Piece-wise Linear Regressor

Our regressor is a piece-wise linear function expressed using Generalized Hinging Hyperplanes (GHH) [4, 42]:

\[
F(x; \omega) = \sum_{n=1}^{N} \delta_n \max_{m=1}^{M} w_{nm}^T x ,
\]

where \(x\) is a vector made of image features extracted from an image patch, \(\omega\) is the vector of parameters of the regressor and can be decomposed into \(\omega = [w_{11}^T, \ldots, w_{MN}^T, \delta_1, \ldots, \delta_N]^T\). The \(w_{nm}\) vectors can be seen as linear filters. The parameters \(\delta_n\) are constrained to be either -1 or +1. \(N\) and \(M\) are meta-parameters which control the complexity of the GHH. As image features we use the three components of the LUV color space and the image gradients—horizontal and vertical gradients and the gradient magnitude—computed at each pixel of the \(x\) patches.

[42] showed that any continuous piecewise-linear function can be expressed in the form of Eq. (1). It is well suited to our keypoint detector learning problem, since applying the regressor to each location of the image involves only simple image convolutions and pixel-wise maximum operators, while regression trees require random access to the image and the nodes, and CNNs involve higher-order convolutions for most of the layers. Moreover, we will show that this formulation also facilitates the integration of different constraints, including constraints between the responses for neighbor locations, which are useful to improve the performance of the keypoint extraction.

Instead of simply aiming to predict the score computed from the distance to the closest keypoint in a way similar to what was done in [34], we argue that it is also important to distinguish the image locations that are close to keypoints from those that are far away. The values returned by the regressor for image locations close to keypoints should have a local maximum at the keypoint locations, while the actual values for the locations far from the keypoints are irrelevant as long as they are small enough to discard them by simple thresholding. We therefore first introduce a classification-like term that enforces the separation between these two different types of image locations. We also rely on a term that enforces the response to have a local maximum at the keypoint locations, and a term that regularizes the responses of the regressor over time. To summarize, the objective function \(\mathcal{L}\) we minimize over the parameters \(\omega\) of our regressor can be written as the sum of three terms:

\[
\min_{\omega} \mathcal{L}_c(\omega) + \mathcal{L}_s(\omega) + \mathcal{L}_t(\omega) .
\]

### 4.2. Objective Function

In this subsection we describe in detail the three terms of the objective function introduced in Eq. (2). The individual influences of each term are evaluated empirically and discussed in Section 5.4.

**Classification-Like Loss \(\mathcal{L}_c\)** As explained above, this term is useful to separate well the image locations that are close to keypoints from the ones that are far away. It relies on a max-margin loss, as in traditional SVM [7]. In particular, we define it as:

\[
\mathcal{L}_c(\omega) = \gamma_c \|\omega\|_2^2 + \frac{1}{K} \sum_{i=1}^{K} \max (0, 1 - y_i F(x_i; \omega))^2 ,
\]

where \(\gamma_c\) is a meta-parameter, \(y_i \in \{-1, +1\}\) is the label for the sample \(x_i\), and \(K\) is the number of training data.

**Shape Regularizer Loss \(\mathcal{L}_s\)** To have local maxima at the keypoint locations, we enforce the response of the regressor to have a specific shape at these locations. For each positive sample \(i\), we force the response shape by defining a loss term related to the desired response shape \(h\), similar to the
one used in [34] and shown in Fig. 2(b):
\[
    h(x, y) = e^{\alpha(1 - \frac{\sqrt{x^2+y^2}}{D})} - 1 ,
\]
where \(x, y\) are pixel coordinates with respect to the center of the patch, and \(\alpha, \beta\) meta-parameters influencing the sharpness of the shape.

However, we want to enforce only the general shape and not the scale of the responses to not interfere with the classification-like term \(L_c\). We therefore introduce an additional term defined as:
\[
    L_s(\omega) = \frac{\gamma_s}{K_p} \sum_{i \mid y_i = \pm 1} \sum_n \| w_{n\eta_i(n)} \ast x_i - \langle w_{n\eta_i(n)}^T x_i \rangle h \|^2_2 ,
\]
where \(\ast\) denotes the convolution product, \(K_p\) is the number of positive samples; \(\gamma_s\) is a meta-parameter for weighting the term that will be estimated by cross-validation. \(\eta_i(n) = \arg \max_m \langle w_{nm}^T x_i \rangle\) is used to enforce the shape constraints only on the filters that contribute to the regressor response of the \(\max\) operator.

It turns out that it is more convenient to perform the optimization of this term in the Fourier domain. If we denote the 2D Fourier transform of \(w_{nm}, x_i,\) and \(h\) as \(\tilde{w}_{nm}, \tilde{x}_i,\) and \(\tilde{h},\) respectively, then by applying Parseval’s theorem and the Convolution theorem, Eq. (5) becomes \(^2\)
\[
    L_s(\omega) = \frac{\gamma_s}{K_p} \sum_{i \mid y_i = \pm 1} \sum_n \| \tilde{w}_{nm}^T \tilde{s}_i^T \tilde{s}_i \tilde{w}_{nm} \|^2_2 ,
\]
where
\[
    \tilde{s}_i = \langle \text{diag} (\tilde{x}_i) - \tilde{x}_i \tilde{h} \rangle^T .
\]
This way of enforcing the shape of the responses is a generalization of the approach of [29] to any type of shape. In practice, we approximate \(\tilde{s}_i\) with the mean over all positive training samples for efficient learning. We also use Parseval’s theorem and the feature mapping proposed in Ashraf et al.’s work [2] for easy calculation \(^2\).

**Temporal Regularizer Loss \(L_t\)** To enforce the repeatability of the regressor over time, we force the regressor to have similar responses at the same locations over the stack of training images. This is simply done by adding a term \(L_t\) defined as:
\[
    L_t(\omega) = \frac{\gamma_t}{K} \sum_{i=1}^K \sum_{j \in \mathcal{N}_i} (\textbf{F}(x_i; \omega) - \textbf{F}(x_j; \omega))^2 ,
\]
where \(\mathcal{N}_i\) is the set of samples at the same image locations as \(x_i\), but from the other training images of the stack. \(\gamma_t\) is again a meta-parameter to weight this term.

---

\(^2\) See Appendix in the supplemental material for derivation
5. Results

In this section we first describe our experimental setup and present both quantitative and qualitative results on our Webcam dataset and the standard Oxford and EF dataset.

5.1. Experimental Setup

We compare our approach to TaSK, SIFT, SFOP, WADE, FAST-9, SIFER, SURF, LCF, MSER, and EdgeFoci\(^3\). In the following, our full method will be denoted TILDE-P. TILDE-P24 denotes the same method, after approximation of the piece-wise linear regressor using 24 separable filters.

To evaluate our regressor itself, we also compared it against two other regressors. The first regressor, denoted TILDE-GB, is based on boosted regression trees and is an adaptation of the one used in [34] for centerline detection to keypoint detection, with the same parameters used for implementation as in the original work. The second regressor we tried, denoted TILDE-CNN, is a Convolutional Neural Network, with an architecture similar to the LeNet-5 network [20] but with an additional convolution layer and a max-pooling layer. The first, third, and fifth layers are convolutional layers; the first layer has a resolution of \(28 \times 28\) and filters of size \(5 \times 5\), the third layer has 10 feature maps of size \(12 \times 12\) and filters of size \(5 \times 5\), and the fifth layer 50 feature maps of size \(4 \times 4\), and filters of size \(3 \times 3\). The second, fourth, and sixth layers are max-pooling layers of size \(2 \times 2\). The seventh layer is a layer of 500 neurons fully connected to the previous layer, which is followed by the eighth layer which is a fully-connected layer with a sigmoid activation function, followed by the final output layer. For the output layer we use the \(l_2\) regression cost function.

---

\(^3\)See the supplementary material for implementation details.

5.2. Quantitative Results

We thoroughly evaluated the performance of our approach using the same repeatability measure as [31], on our Webcam dataset, and the Oxford and EF datasets. The repeatability is defined as the number of keypoints consistently detected across two aligned images. As in [31] we consider keypoints that are less than 5 pixels apart when projected to the same image as repeated. However, the repeatability measure has two caveats: First, a keypoint close to several projections can be counted several times. Moreover, with a large enough number of keypoints, even simple random sampling can achieve high repeatability as the density of the keypoints becomes high.

We therefore make this measure more representative of the performance with two modifications: First, we allow a keypoint to be associated only with its nearest neighbor, in other words, a keypoint cannot be used more than once when evaluating repeatability. Second, we restrict the number of keypoints to a small given number, so that picking the keypoints at random locations would results with a repeatability score of only 2%, reported as **Repeatability (2%)** in the experiments.

We also include results using the standard repeatability score, 1000 keypoints per image, and a fixed scale of 10 for our methods, which we refer to as *Oxford Stand.* and *EF Stand.*, for comparison with previous papers, such as [26, 44]. Table 1 shows a summary of the quantitative results.

5.2.1 Repeatability on our Webcam Dataset

Fig. 5 gives the repeatability scores for our Webcam dataset. Fig. 5-top shows the results of our method when trained on each sequence and tested on the same sequence, with
the set of images divided into disjoint train, validation, and test sets. Fig. 5-bottom shows the results when we apply our detector trained on one sequence to all other unseen sequences from the Webcam dataset. We significantly outperform state-of-the-art methods when using a detector trained specifically to each sequence. Moreover, while the gap is reduced when we test on un-seen sequences, we still outperform all compared methods by a significant margin, showing the generalization capability of our method.

5.2.2 Repeatability on Oxford and EF Datasets

In Fig. 8 we also evaluate our method on Oxford and EF datasets. Oxford dataset is simpler in the sense that it does not exhibit the drastic changes of the Webcam dataset but it is a reference for the evaluation of keypoint detectors. EF dataset on the other hand exhibits drastic illumination changes and is very challenging. It is therefore interesting to evaluate our approach on these datasets.

Instead of learning a new keypoint detector on this dataset, we apply the detector learned using the Chamomix sequence from the Webcam dataset. Our method still achieves state-of-the-art performance. We even significantly outperform state-of-the-art methods in the case of the Bikes, Trees, Leuven and Rushmore images, which are outdoor scenes. Note that we also obtain good results for Boat which has large scale changes, although we currently do not consider scale in learning and detecting. Repeatability score shown here is lower than what was reported in previous works [26, 31] as we consider a smaller number of keypoints. As mentioned before, considering a large number of keypoints artificially improves the repeatability score.

5.3. Qualitative Results

We also give in Fig. 9 some qualitative results on the task of matching challenging pairs of images captured at different days under different weather conditions. Our matching pipeline is as follow: we first extract keypoints in both images using the different methods we want to compare, compute the keypoints descriptors, and compute the homography between the two images using RANSAC. Since the goal of this comparison is to evaluate keypoints not descriptors, we use the SIFT descriptor for all methods. Note that we also tried using other descriptors [3, 32, 6, 1, 21] but due to the drastic difference between the matched images, only SIFT descriptors with ground truth orientation and scale worked. We compare our method with the SIFT [23], SURF [3], and FAST-9 [31] detectors, using the same number of keypoints (300) for all methods. Our method allows to retrieve the correct transformations between the images even under such drastic changes of the scene appearance.

5.4. Effects of the Three Loss Terms

Fig. 6 gives the results of the evaluation of the influence of each loss term of Eq. (2) by evaluating the performance of our detector without each term. We will refer to our method when using only the classification loss as TILDE-Pc, when using both classification loss and the temporal regularization as TILDE-Pt, and when using the classification loss and the shape regularization as TILDE-Ps.

Table 1: Repeatability performance of our best regressors. The best results are in bold. Our approach provides the highest repeatability, when using our piece-wise linear regressor. Note that on Oxford and EF datasets the performance are slightly better when using smaller number of separable filters to approximate the original ones, probably because the approximated filters tend to be smoother.

<table>
<thead>
<tr>
<th>#keypoints</th>
<th>Webcam (2%)</th>
<th>Oxford (2%)</th>
<th>EF (2%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>TILDE-GB</td>
<td>33.3</td>
<td>54.5</td>
<td>32.8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>43.1</td>
<td>16.2</td>
</tr>
<tr>
<td>TILDE-CNN</td>
<td>36.8</td>
<td>51.8</td>
<td>49.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>43.2</td>
<td>27.6</td>
</tr>
<tr>
<td>TILDE-P24</td>
<td>40.7</td>
<td>58.7</td>
<td>59.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>46.3</td>
<td>33.0</td>
</tr>
<tr>
<td>TILDE-P</td>
<td>48.3</td>
<td>58.1</td>
<td>55.9</td>
</tr>
<tr>
<td></td>
<td></td>
<td>45.1</td>
<td>31.6</td>
</tr>
<tr>
<td>FAST-9</td>
<td>26.4</td>
<td>53.8</td>
<td>47.9</td>
</tr>
<tr>
<td></td>
<td></td>
<td>39.0</td>
<td>28.0</td>
</tr>
<tr>
<td>SFOP</td>
<td>22.9</td>
<td>51.3</td>
<td>39.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>42.2</td>
<td>21.2</td>
</tr>
<tr>
<td>SIFER</td>
<td>25.7</td>
<td>45.1</td>
<td>40.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>27.4</td>
<td>17.6</td>
</tr>
<tr>
<td>SIFT</td>
<td>20.7</td>
<td>46.5</td>
<td>43.6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>32.2</td>
<td>23.0</td>
</tr>
<tr>
<td>SURF</td>
<td>29.9</td>
<td>56.9</td>
<td>57.6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>43.6</td>
<td>28.7</td>
</tr>
<tr>
<td>TaSK</td>
<td>14.5</td>
<td>25.7</td>
<td>15.7</td>
</tr>
<tr>
<td></td>
<td></td>
<td>22.8</td>
<td>10.0</td>
</tr>
<tr>
<td>WADE</td>
<td>27.5</td>
<td>44.3</td>
<td>51.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>25.6</td>
<td>28.6</td>
</tr>
<tr>
<td>MSER</td>
<td>22.3</td>
<td>51.5</td>
<td>35.9</td>
</tr>
<tr>
<td></td>
<td></td>
<td>38.9</td>
<td>23.9</td>
</tr>
<tr>
<td>LCF</td>
<td>30.9</td>
<td>55.0</td>
<td>40.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>41.6</td>
<td>23.1</td>
</tr>
<tr>
<td>EdgeFoci</td>
<td>30.0</td>
<td>54.9</td>
<td>47.5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>46.2</td>
<td>31.0</td>
</tr>
</tbody>
</table>
Figure 8: Repeatability (2%) score on the Oxford and EF datasets. Our methods are trained on the Chamonix sequence from the Webcam dataset and tested on Oxford and EF datasets.

Figure 9: Qualitative results on several images from different sequences. From top to bottom: Courbevoie, Frankfurt, and StLouis. (a) Pairs of images to be matched, with ground truth transformation, transformations obtained with (b) the SIFT detector, (c) the SURF detector, (d) the FAST-9 detector, and (e) our TILDE detector.

We achieve the best performance when all three terms are used together. Note that the shape regularization enhances the repeatability on Oxford and EF, two completely unseen datasets, whereas the temporal regularization helps when we test on images which are similar to the training set.

5.5. Computation Times

Fig. 7 gives the computation time of SIFT and each variant of our method. TILDE-P24 is not very far from SIFT. Note that our method is highly parallelizable, while our current implementation does not benefit from any parallelization. We therefore believe that our method can be significantly sped up with a better implementation.

6. Conclusion

We have introduced a learning scheme to detect keypoints reliably under drastic changes of weather and lighting conditions. We proposed an effective method for generating the training set to learn regressors. We learned three regressors, which among them, the piece-wise linear regressor showed best result. We evaluated our regressors on our new outdoor keypoint benchmark dataset. Our regressors significant outperforms the current state-of-the-art on our new benchmark dataset and also achieve state-of-the-art performances on Oxford and EF datasets, demonstrating their generalisation capability.

An interesting future research direction is to extend our method to scale space. For example, the strategy applied in [21] to FAST can be directly applied to our method.

Acknowledgement

This work was supported in part by the EU FP7 project MAGELLAN under the grant number ICT-FP7-611526 and in part by the EU project EDUSAFE.
References

[34] A. Sironi, V. Lepetit, and P. Fua. Multiscale Centerline Detection by Learning a Scale-Space Distance Transform. In Conference on Computer Vision and Pattern Recognition, 2014. 1, 4, 5, 6


