Lategahn, Henning ; Beck, Johannes ; Kitt, Bernd ; Stiller, Christoph: How to Learn an Illumination Robust Image Feature for Place Recognition. In: IEEE Intelligent Vehicles Symposium. Gold Coast, Australia, June 2013, pp. 285–291

How to Learn an Illumination Robust Image Feature for Place Recognition

Henning Lategahn, Johannes Beck, Bernd Kitt, Christoph Stiller
Institute of Measurement and Control
Karlsruhe Institute of Technology
Karlsruhe, Germany
{henning.lategahn, johannes.beck, bernd.kitt, stiller}@kit.edu

***Abstract*— Place recognition for loop closure detection lies at the heart of every Simultaneous Localization and Mapping (SLAM) method. Recently methods that use cameras and describe the entire image by one holistic feature vector have experienced a resurgence. Despite the success of these methods, it remains unclear how a descriptor should be constructed for this particular purpose. The problem of choosing the right descriptor becomes even more pronounced in the context of life long mapping. The appearance of a place may vary considerably under different illumination conditions and over the course of a day. None of the handcrafted descriptors published in literature are particularly designed for this purpose.**

Herein, we propose to use a set of elementary building blocks from which millions of different descriptors can be constructed automatically. Moreover, we present an evaluation function which evaluates the performance of a given image descriptor for place recognition under severe lighting changes. Finally we present an algorithm to efficiently search the space of descriptors to find the best suited one.

Evaluating the trained descriptor on a test set shows a clear superiority over its hand crafted counter parts like BRIEF and U-SURF. Finally we show how loop closures can be reliably detected using the automatically learned descriptor. Two overlapping image sequences from two different days and times are merged into one pose graph. The resulting merged pose graph is optimized and does not contain a single false link while at the same time all true loop closures were detected correctly.

The descriptor and the place recognizer source code is published with datasets on http://www.mrt.kit.edu/libDird.php.

I. Introduction

Most modern approaches represent the SLAM problem as a graph optimization problem ([12]). Thereby a tractable level of complexity is reached. Lately, some algorithms have been demonstrated to work robustly on large scales. However, detecting previously visited places to build the system of constraints (the graph) lies at the heart of every of these methods and is commonly referred to as the loop closure problem. A plethora of methods have been proposed to address this problem using cameras. Many of these methods are difficult to implement or require previous training (e.g. creating code books etc. [9]).

Just recently a new branch of algorithms have been proposed. The entire image of a sequence is represented by one holistic descriptor vector (e.g. [19]). Image similarity is thereafter computed by vector distance. However, it remains unclear which visual descriptor is best suited to represent places. This question becomes especially pronounced since a flood of image descriptors already exist and new ones are published frequently. Nevertheless, none of them are particularly designed to be robust or invariant to changes in lighting conditions occurring during different times of the day. Moreover, these descriptors are constructed with a broad applicability in mind even though one would expect a specialized descriptor to outperform a general purpose one.

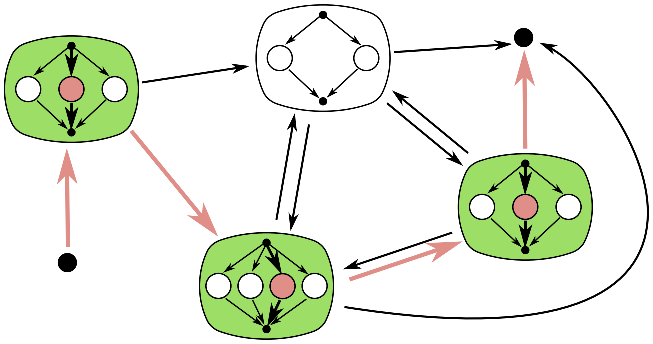

Fig. 1: A simplified example graph of processing steps. Large nodes (green) are processing steps and small nodes inside (red) are parameters. A path through the graph corresponds to exactly one descriptor.

Herein we propose a set of elementary algorithmic building blocks from which millions of different image descriptors can be constructed. A sequence of any length of these blocks results in one descriptor each. Common algorithms like BRIEF ([7]), LBP ([16]) and almost any other descriptor can be constructed by one of these sequences of blocks. This set of blocks enables us to automatically produce a multitude of image descriptors. We strive at finding a single descriptor among several millions that is best suited to represent places under illumination variations. Our goal is robust place recognition for loop closure detection using a simple holistic approach.

We resort to a fitness function which evaluates a given descriptor for the given task. For this purpose we compute the area under the Precision-Recall (PR) curve. Thereto, a set of image couples showing either the same or different places is compiled and taken for training. These pairs of images are randomly drawn from three sequences of the same route recorded on different times of the day. By recording sequences in the morning, at noon and in the evening the training set exhibits a large degree of lighting variations. Our goal is to automatically find a descriptor which is robust against these image variations. To this end we use a meta heuristic (evolution strategies) to evolve a set of descriptors, mutate these to create children and finally select only the best performing candidates to form the next generation. Thereafter iterations are continued by creating new children.

We trained a descriptor by the aforementioned method. The best descriptor was then tested on a disjoint test set and compared to state of the art image descriptors like U-SURF and BRIEF. Experiments show that the trained descriptor significantly outperforms its off the shelf counter parts. Finally we recorded two image sequences on different days, detected loop closures by the trained descriptor and optimized the visual odometry induced pose graph. Thereby we obtained one joint pose graph. The resulting graph shows no false loop closures.

II. Related Work

The presented work is related to methods on place recognition and loop closure detection [9], [1], [2], [19], [15] and research on descriptor evaluation and learning [21], [6], [17], [8], [18], [5], [22], [10], [14].

FabMap presented in [9] by Cummnis and colleagues has emerged into the work horse of loop closure detection. A bag of word model on a specifically trained code book is used. Our method for loop closure detection is rather simplistic. The entire image is down sampled, tiled and a single feature vector is computed after concatenating single tile features. No previous training is required, no code-books need to be stored and description and retrieval time is extremely fast.

Badino et al. [1], [2] use SURF like features to describe poses of a previously recored trajectory (the map). During a second traversal online imagery is processed, features are extracted and the nearest pose of the map is estimated. To this end a histogram filter is fed with these image features and velocity information to smooth the localization estimate. Experimental results show a high robustness.

Sünderhauf and co-workers present a holistic image feature based on the BRIEF descriptor in [19]. Each image is partitioned into tiles each of which is described by BRIEF. All single tile feature vectors are concatenated to represent the image. Loop closures are detected by computing vector differences. The pose graph back-end is robustified as in [20]. We herein follow their line in describing images holistically.

Milford has pushed this idea further as presented in [15]. Panoramic images are compared for similarity by the mean absolute intensity difference with topometric localization in mind. However, image resolution and bit depth are impressively reduced while still being discriminative enough. One crucial ingredient seems to be the dynamic time warping of the pair wise image difference matrix. Experiments on double round trip trajectories of up to 70km are presented.

The work closest to our descriptor learning framework is the work of Brown and co-workers [21], [6]. A set of blocks are presented which are combined and whose parameters are optimized by Powell's method. Blocks include smoothing, non-linear transforms, pooling and normalization. A large training set of different views of the same points is created by bundle adjustment. The order of blocks however is fixed and only the parameters are optimized. Our methods spans a much greater space of descriptors with more processing blocks. Furthermore, the order of blocks is optimized as well.

Parameters of SIFT and HOG features are optimized by the method of [18]. A set of patches are automatically extracted from street level imagery similarly to [6]. The parameter space of these hand crafted methods is thereafter searched for an optimum with car classification in mind. Experiments show a substantial improvement of the trained parameter set over the default set. Furthermore, it is shown how classification accuracy can benefit from application specific parameters.

Philibin et al. [17] and Carneiro [8] both present a method for feature learning. Their work focuses on learning a distance function for image retrieval. We refrained from learning a mere distance transform but rather optimized the entire descriptor.

Descriptors used for image retrieval largely depend on spatial pooling and vector coding. Boureau and colleague [5] systematically investigated the effects of proper pooling and coding choices. The importance of appropriate choices for these steps could be highlighted.

Using a filter operation followed by estimating the filter distribution has been investigated in our previous work on texture description in [14].

A large number of work has been conducted on evaluating different image detectors and descriptors. We exemplary cite the recent work of Gauglitz and co-workers [10]. A vast test set of images for feature tracking was created and commonly used descriptors are evaluated in terms of matching performance. However no alteration or automatic feature construction is proposed.

III. Descriptor Building Blocks

We present the scheme from which a broad class of different image descriptors can be constructed in the following section. First we present a simple example by constructing one sample descriptor. It serves as a mere example to introduce our approach. Thereafter the general method is elucidated and each block is presented in detail.

We consider a gray level image I and a pixel position i, j which shall be described by the descriptor. Now, a simple descriptor may convolve the image I with a set of Sobel filters to compute the filter responses $R_{\mathrm{hor}} = I * F_{\mathrm{hor}}$ and $R_{\mathrm{ver}} = I * F_{\mathrm{ver}}$ with $*$ denoting convolution. The descriptor may output a two dimensional feature vector $f_1(i, j)$ of the filter responses at the given position i.e. $f_1(i, j) =$

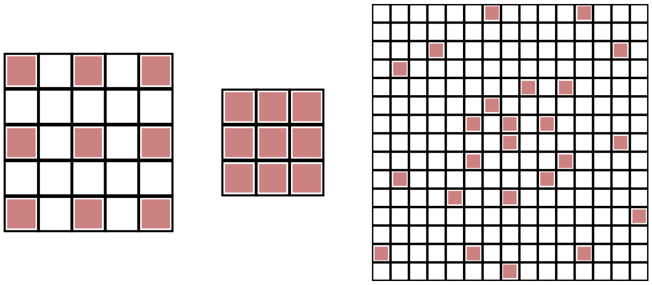

Fig. 2: A given descriptor can be extended by applying it at different offsets of pixel positions and concatenation. Several such repetition masks are shown.

$$\begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & -1 & \cdots \\ 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & \cdots \\ 0 & -1 & 0 & 0 & 0 & 0 & 1 & 0 & \cdots \\ 1 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & \cdots \\ & \vdots & & & & \vdots & & & \end{bmatrix} \begin{bmatrix} -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 \\ & & \vdots & & & & \vdots & & \end{bmatrix}$$

Fig. 3: A given descriptor can be extended by applying a fixed matrix A to it. Two example matrices are shown. See text for details.

$(R_{\text{hor}}(i,j), R_{\text{ver}}(i,j))^T$. To extend such a simplistic method one could apply the given descriptor several times at predefined positions around i, j and concatenate these into a larger vector. The resulting feature vector would then be $f_2(i,j) = (f_1(i+i_1, j+j_1), \ldots, f_1(i+i_N, j+j_N))^T$ for N offsets of pixel position $(i_1, j_1), \ldots, (i_N, j_N)$. We stress that the descriptor which computes f_2 uses the descriptor which computes f_1 as a black box. It simply applies the descriptor f_1 several times at different pixel positions around i, j and concatenates its output to form f_2. Following this line of extending any given descriptor by one additional step we can drive this idea further. Creating a third descriptor which computes f_3 given f_2 can simply be computed by matrix multiplication $f_3 = A \cdot f_2$ for a fixed matrix A. The descriptor to compute f_3 again uses the descriptor to compute f_2 without knowledge of how f_2 was constructed. Furthermore, a fourth descriptor may use the descriptor to compute f_3 and apply it several times in the vicinity of the pixel position i, j and compute a histogram of f_3-values which in turn forms f_4. This chain of elementary steps can be extended almost arbitrarily.

The steps for this example descriptor are: Sobel filtering, repetition, matrix multiplication and finally histogram computation. The main point is that any step can be altered independently of the other steps. Sobel filters may be replaced by low and high pass filters while keeping the consecutive steps fixed as before. The repetition pattern may change, the matrix A can change or histogram bins can be altered. Moreover, steps may be applied more than once for one descriptor. Nothing keeps one from applying yet another repetition to f_4 to form f_5.

In the following each of these building blocks are presented in more detail. Each block contains a set of parameters which are detailed as well. A repetition block for instance may contain many different repetition patterns (pixel position offsets), many matrices A are sensible for matrix multiplication etc. Every descriptor always starts with a filter operation. All other steps can be combined arbitrarily and in any order.

Filter Banks: The following filter banks are contained in our scheme: Sobel Filters (no parameters), Gaussian blurring (several kernel widths σ as parameters), Derivative in horizontal and vertical direction (step size for differentiation as parameters), Haar Features (cascade depth as parameter), Rank Transform (no parameters) and Census Transform (no parameters). Moreover, an empty filter (identity) is used to be able to reproduce the original image without filtering.

Repetition: A repetition is represented by a set of offsets of pixel position. Several patterns of repetition quite close to the pixel which shall be described are stored as parameters. The number of offsets for these patterns are between nine and sixteen. Moreover, a set of BRIEF like repetition patterns for different numbers of offsets are stored. These offsets follow Gaussian distributions hence exhibiting a closer density towards the center. The number of offsets ranges from 20 to approximately 300 for these patterns. Finally a small set of repetitions are stored as parameters which are further away from the center and form an equidistantly spaced grid. The number of offsets is between four and sixteen. Some of these repetition patterns are exemplary shown in Figure 2.

Linear Transform: As explained in the aforementioned example descriptor a matrix multiplication may be appended to a sequence of construction steps. Two sets of predefined matrices are stored. Set one contains $2N \times N$ matrices with exactly one -1 and one 1 in each row such that every column contains exactly one non-zero entry. The second set contains $N \times N - 1$ matrices with one column of 1s and rows containing exactly one additional -1 such that every column has at most one -1. One example each is shown in Figure 3.

Non-linear Transform: One Transform that computes polar from Cartesian coordinates and one transform that scales the feature vector to unit length are parameters for non-linear transformation. Moreover we use a sign quantization transform which replaces all negative elements of a vector by 0s and non-negative elements by 1s.

Histogram: A descriptor computing feature vectors f can be extended by applying it to a set of B independent test images to obtain feature vectors $f_1, \ldots, f_B$. These feature vectors thereafter serve as bin centers for a histogram and are kept fixed. The extended descriptor to compute f' finally computes f at many pixel positions around the point to be described. The counter for the nearest bin center is incremented. The final histogram (of dimension B) is then output as f'. The number of bin centers and the area from which the histogram shall be filled serve as parameters for this building block.

Dimensionality Reduction: The dimension of a feature vector may be reduced by multiplying the vector by a fixed matrix A. For dimension reduction we use random projection matrices ([4]). The degree by which the vector is reduced is

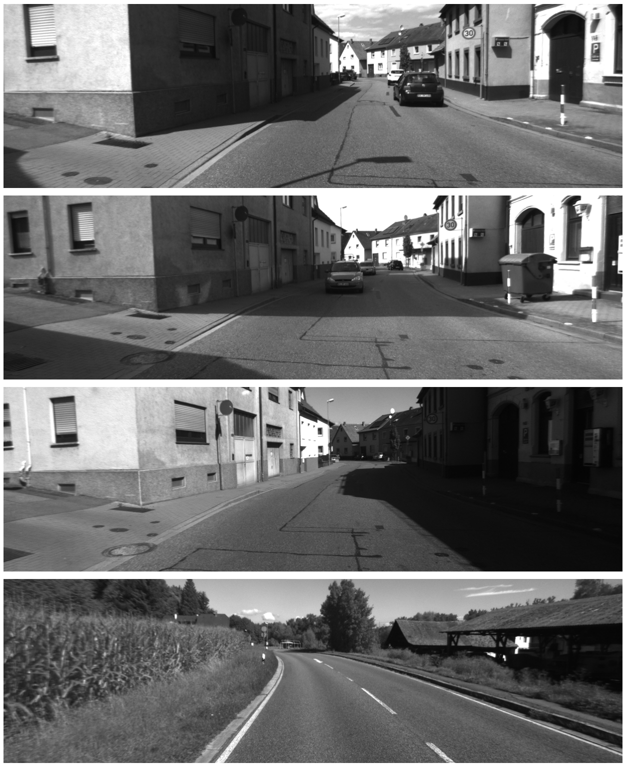

Fig. 4: Some sample images from the training set. The first three show the same place during morning, noon and evening. Note the severe changes in illumination and cast shadows.

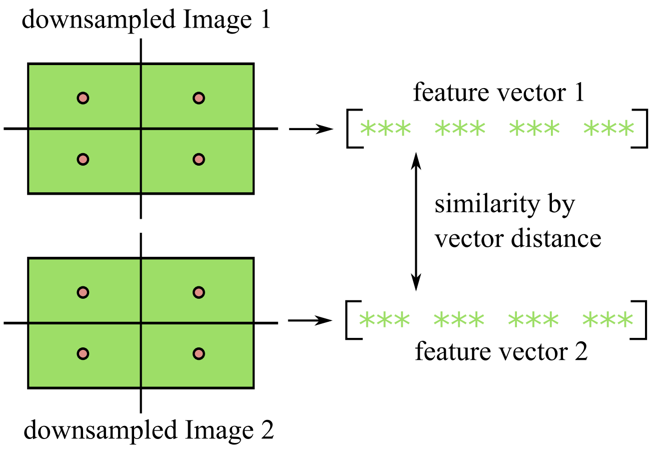

Fig. 5: An image is partitioned into tiles and each tile is described by a feature vector. The holistic image feature is formed by concatenation. Image similarity is computed by vector difference.

the set of parameters for this step. It ranges from 10% to 90%.

Summation: Much like histograms a descriptor to compute f can be extended by applying it within a region and summing all such vectors. This step does not contain parameters. The descriptor construction scheme can be summarized by a graph. The nodes of the graph represent the building blocks as described above. Each node contains a fixed set of parameters. Every path through the graph corresponds to exactly one descriptor. The graph is depicted in Figure 1. Using this graph/path notation it is easy to randomly create a large set of descriptors by sampling paths through the graph. We have constrained the maximum length of a path (number of steps) to six. But even then the total number of different descriptors which can be constructed reaches almost a billion due to the combinatorial explosion. Hence it is difficult to find the best performing descriptor for a particular computer vision problem.

We exemplary show how the BRIEF and Local Binary Pattern (LBP, [16]) descriptor can be constructed using the proposed blocks. For BRIEF an initial filter operation using a Gaussian smoothing is used. Thereafter a repetition with offset indicies as depicted on the right of Figure 2 is applied. The offset positions are Gaussian distributed (see [7]). A linear transform with multiplication matrix of Figure 3 is applied to the thus computed vector. A sign quantization finally yields exactly the descriptor presented in [7]. The LBP descriptor can be assembled using the identity filter (corresponding to no filtering) followed by a repetition with the pattern depicted in the middle of Figure 2. This yields a 9-vector so far. The vector can be multiplied (linear transform) by the right matrix of Figure 3 again followed by sign quantization. Thereafter the sign quantized vector is multiplied by the following matrix

$$A = \begin{pmatrix} 2^0 & 2^1 & \dots & 2^7 \end{pmatrix} \qquad (1)$$

Computing a histogram over these LBP values finally leads the LBP descriptor.

IV. Searching the Descriptor Space

In the next section we show how we seek a good performing descriptor. First we describe the fitness function which evaluates the performance of a given descriptor. Thereafter a meta heuristic to find a good performing descriptor among the plenty is presented.

A. *Fitness Function*

We strive to find an image descriptor that is able recognize a place from an image even under varying lighting conditions. This problem occurs frequently in visual SLAM loop closure detection. To this end each image of a sequence is down sampled and partitioned into $M \times M$ equally sized tiles. The center pixel of each image tile is described by an image descriptor and feature vectors are concatenated to form a holistic image descriptor ([19]). In all experiments we use a 4×4 tiling which was empirically found to be optimal. Place similarity is thereafter computed by vector difference. Figure 5 exemplary illustrates this method.

To evaluate the performance of a given descriptor we have traveled a route of approximately 10km on three different times of the day. We recorded image sequences and formed pairs of images showing the same place. The matching was computed from high precision GPS measurements. We refer to these image pairs as positive pairs. Furthermore, we randomly created negative pairs of images that do not show the same place. Example images are shown in Figure 4. Using this training set as ground truth a PR curve can be computed by varying a classification threshold of norms of vector differences. That means we strive to find a descriptor that best separates the positive from the negative pairs. By including images recorded in the morning, at noon and evening the change that naturally occurs during the course of a day is well captured in our training set.

B. Evolution Strategies

Since the space of descriptors is not a vector space (a descriptor cannot be expressed by a fixed size numerical vector) we resort to evolution strategies ([3]). An initial population of L descriptors is created by randomly sampling paths through the graph of building blocks. Thereafter M children are created from the parent population by mutation. All parents and children are evaluated by the fitness function presented in section IV-A and the L best performing descriptors are selected into the next generation. The remaining descriptors are discarded. Mutation and selection steps are continued until performance doesn't improve further.
The crucial step of many optimization methods of this kind is the mutation operator. Three different types of mutation are implemented in our framework.

- A given descriptor may be extended by an extra step (not necessarily at the end of the path but also in between).
- A descriptor may be reduced by one step.
- A parameter of any step may be altered.

Figure 6 shows these three mutations for a descriptor.
The evaluation of one descriptor is rather time consuming. The descriptor needs to be applied to many thousand images. To address this complexity problem we have furthermore developed a method to adaptively refine the training set. Very easy image pairs do not contribute much to the training success and should therefore be discarded in favor of more challenging pairs. Therefore the training set is thinned out after a fixed number of descriptor generations (typically four). Easy to classify image pairs are detected by the frequency they are correctly classified by different descriptors. Thereby we are able to double the number of descriptors that can be evaluated per time and speed up the training accordingly.

V. Experiments

We ran the proposed descriptor learning framework with the fitness function of Section IV-A for approximately 100 hours. We ran the evolution strategies with several instances in parallel with slightly varying parameter sets with one to four parents and four to sixteen children per generation. We observe that the overall fitness function improves drastically over the first few generations and convergence slows down thereafter. The overall best performing descriptor on the test set reaches an area under the PR curve (AUC) of 0.82. Since the paths through the graph of building blocks is quite verbose the learned descriptor can be interpreted. First the image is filtered by a Haar Filters (horizontal, vertical, diagonal) of depth four. Thereafter, the vectors of filter responses for each pixel is scaled to unit norm and summed over a set of offset positions. Finally this feature vector is evaluated on nine offset positions around the pixel in question and concatenated (repetition). The resulting dimension is 216. In the sequel we refer to this descriptor as DIRD (Dird is an Illumination Robust Descriptor).

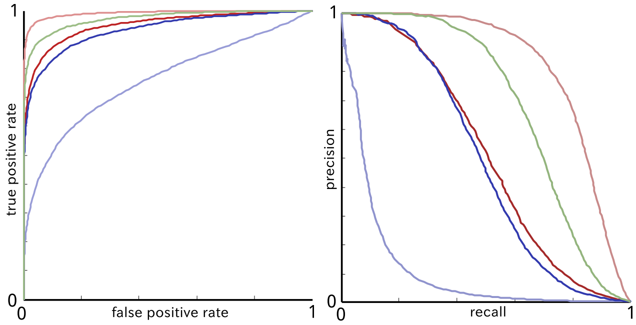

Fig. 7: ROC and PR curves for the test set. From bad to good: simple gray value descriptor, SURF, BRIEF, U-SURF and the learned descriptor DIRD. See also Figure 8.

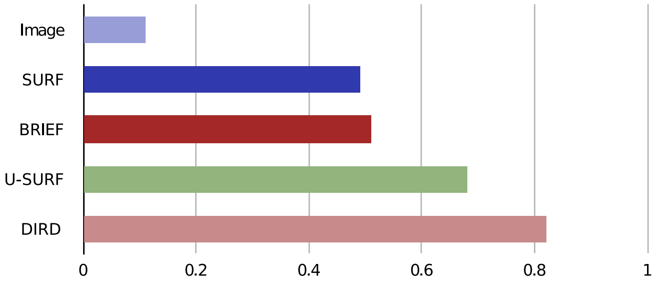

Fig. 8: Area under the PR curve for a simple gray value descriptor (Image), SURF, BRIEF, U-SURF and the learned descriptor DIRD. For the curves see Figure 7.

DIRD was then tested on a test set of image pairs showing the same and different places. Test and training sets are strictly disjoint and taken from different traversals. The performance was then compared to BRIEF, SURF, U-SURF and a simple gray value descriptor. Results are depicted in Figures 7 and 8. DIRD clearly outperforms its general purpose counter parts. The AUC of DIRD on the test set is 0.82 (same as on the training set). Using the gray values of the image as a feature achieves an AUC of 0.11, SURF of 0.49, BRIEF of 0.51 and U-SURF of 0.68. Results are shown in Figure 8. Thus the specialized descriptor DIRD outperforms the second best U-SURF by a fair margin. ROC and PR curves for these descriptors are depicted in Figure 7.

Since the descriptor was sought with place recognition for loop closure detection in mind we have also tested DIRD in this set up. Thereto we have traveled approximately 11km through an inner city scenario with several loops, recorded stereo imagery and computed a visual odometry

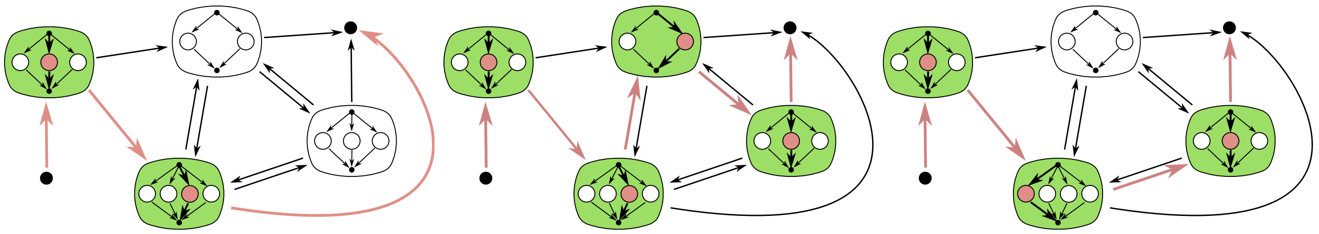

Fig. 6: The path trough the graph of Figure 1 can be mutated into any of these paths in one mutation step.

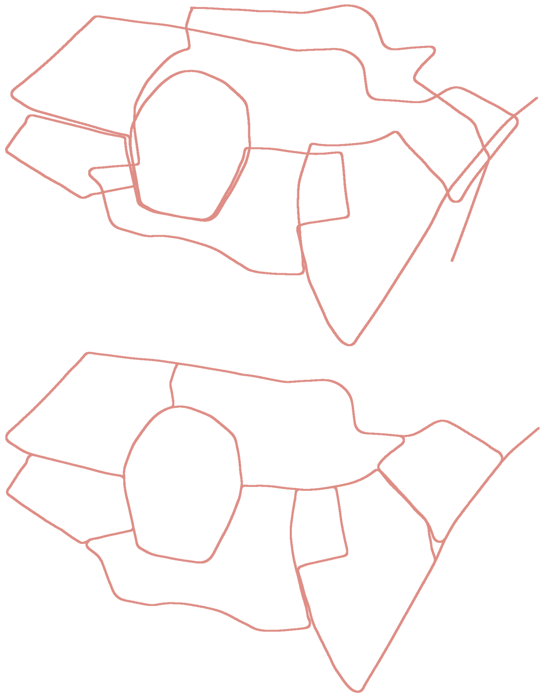

Fig. 9: Top: Pose graph of integrated visual odometry. Beginning of trajectory is the top right. Drift is clearly visible. Bottom: Pose optimized graph after loop closures have been successfully detected using the automatically learned descriptor DIRD.

([11]) induced pose graph from the data. Then we have detected loops by segment matching (see appendix) using DIRD and optimized the pose graph using the g2o library ([13]). The results before and after optimization are shown in Figure 9. The inevitable drift is clearly resolved. No false loop closures are introduced by our method.

Finally we have recorded a new trajectory on a different day and time that overlaps with the first one. This second trajectory was then fully automatically merged into the first graph. Places shared between both trajectories as well as all self loops are reliably detected. The optimized graph is depicted in Figure 10. Note that we refrained from using a robust back-end ([20]) for the optimization of the graph since it is essentially superfluous. No false loop closures are introduced by our method.

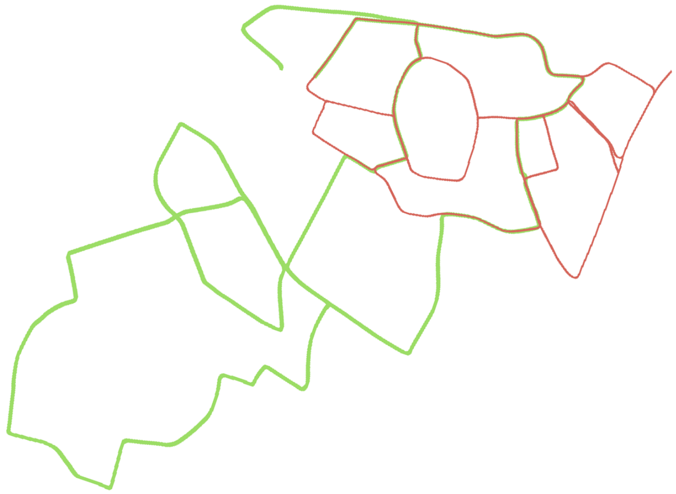

Fig. 10: Two traversals from different days and time which overlap are automatically merged by the proposed place recognizer.

VI. Conclusion and Future Research

Herein we have presented a set of elementary algorithmic building blocks from which a broad class of different descriptors can be constructed automatically. These building blocks span a large space of such methods including many established descriptors already known from literature. Moreover, we have presented a search technique to find good performing descriptors given a specific fitness function. A fitness function to evaluate a given descriptor in the case of place recognition and loop closure detection using holistic features under varying illumination conditions has been introduced. Such a problem specific image descriptor has been trained and evaluated on an independent test set. Experiments show a substantial improvement of this descriptor over its handcrafted counter parts like U-SURF and BRIEF. We conclude that it is well worth investigating efforts into automatic descriptor learning to yield problem specific methods which outperform their general purpose counter parts.

Evaluating the proposed descriptor learning framework on feature matching and other computer vision domains is ongoing research. Furthermore, it seems exciting ground for future work to add more elaborate dimensionality reduction methods and more involved sign-quantization algorithms.

Finally we are curious how far the loop closure detection method presented herein can be pushed.

Appendix

We propose to match segments (subsequences) of a sequence for robust loop closure detection. Milford uses dynamic time warping in [15] from which this method draws some inspirations. Let M be a similarity matrix of size $N \times N$ for a sequence of N poses. $M(i,j) \in [0,1]$ denotes the similarity between poses i and j. The similarity is computed from the distance of image feature vector f_i and vector f_j of the respective poses (cp. Figure 5). Distances are translated into similarities by a logistic function. We strive to translate matrix M into matrix M' which contains only non-zero entries for very likely loop closures.

The main idea is that we expect several consecutive poses to match for a true loop closure. If for instance pose i corresponds to pose j then we expect a high similarity score for i' and j' if these are in the direct vicinity of poses i and j. Formally, we expect a sequence of K indices $\mathcal{I}(i,j) = ((i-K, j_K), \ldots, (i-1, j_1), (i,j))$ with $0 < j_{k+1} - j_k < 4$ such that all $M(i-k, j_k)$ have a high similarity score. To this end we define

$$M'(i,j) \quad = \quad \max_{\mathcal{I}(i,j)} \left\{ \sum_{k=1}^{K} M(i-k, j_k) \right\} \tag{2}$$

with $\mathcal{I}(i,j)$ being a sequence of K indices ending at (i,j) as defined above. M' can be computed efficiently by dynamic programming. In our example we use $K = 20$ meaning we expect to match subsequences of length at least 20. Finally, we apply a non-maxima suppression to M'. A loop closure hypothesis is generated by thresholding M'.

A typical off-diagonal part of M is shown in Figure 11 with the maximizing sequence of indices $\mathcal{I}(i,j)$ drawn in green. The pair of poses i and j is marked with a green triangle. The source code for segment matching can be downloaded[1].

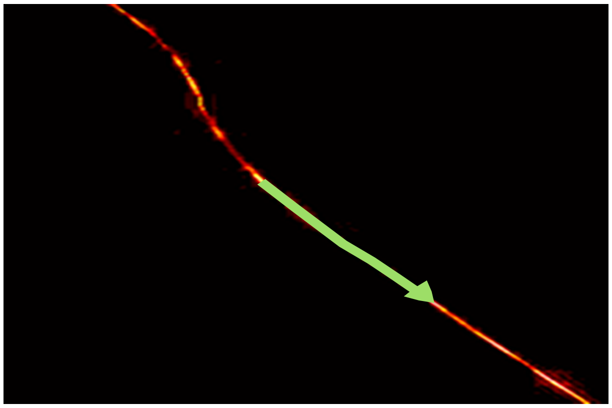

Fig. 11: An off-diagonal fraction of the loop closure matrix which stores pair wise similarities between poses. A segment (subsequence) with high consecutive similarity scores has been found. It is depicted by the green path.

References

[1] H. Badino, D. Huber, and T. Kanade. Visual topometric localization. In *Intelligent Vehicles Symposium (IV), 2011 IEEE*, pages 794–799. IEEE, 2011.

[2] H. Badino, D. Huber, and T. Kanade. Real-time topometric localization. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 1635–1642. IEEE, 2012.

[3] H.G. Beyer. *The theory of evolution strategies*. Springer, 2001.

[4] E. Bingham and H. Mannila. Random projection in dimensionality reduction: applications to image and text data. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 245–250. ACM, 2001.

[5] Y.L. Boureau, F. Bach, Y. LeCun, and J. Ponce. Learning mid-level features for recognition. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2559–2566. IEEE, 2010.

[6] M. Brown, G. Hua, and S. Winder. Discriminative learning of local image descriptors. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 33(1):43–57, 2011.

[7] M. Calonder, V. Lepetit, C. Strecha, and P. Fua. Brief: Binary robust independent elementary features. *Computer Vision–ECCV 2010*, pages 778–792, 2010.

[8] G. Carneiro. The automatic design of feature spaces for local image descriptors using an ensemble of non-linear feature extractors. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 3509–3516. IEEE, 2010.

[9] M. Cummins and P. Newman. Fab-map: Probabilistic localization and mapping in the space of appearance. *The International Journal of Robotics Research*, 27(6):647–665, 2008.

[10] S. Gauglitz, T. Höllerer, and M. Turk. Evaluation of interest point detectors and feature descriptors for visual tracking. *International journal of computer vision*, pages 1–26, 2011.

[11] A. Geiger, J. Ziegler, and C. Stiller. Stereoscan: Dense 3d reconstruction in real-time. In *Intelligent Vehicles Symposium (IV), 2011 IEEE*, pages 963–968. IEEE, 2011.

[12] G. Grisetti, R. Kümmerle, C. Stachniss, and W. Burgard. A tutorial on graph-based slam. *Intelligent Transportation Systems Magazine, IEEE*, 2(4):31–43, 2010.

[13] R. Kummerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard. G2o: A general framework for graph optimization. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 3607–3613. IEEE, 2011.

[14] H. Lategahn, S. Gross, T. Stehle, and T. Aach. Texture classification by modeling joint distributions of local patterns with gaussian mixtures. *Image Processing, IEEE Transactions on*, 19(6):1548–1557, 2010.

[15] M. Milford. Visual route recognition with a handful of bits. In *Proceedings of Robotics Science and Systems Conference 2012*. University of Sydney, 2012.

[16] T. Ojala, M. Pietikainen, and T. Maenpaa. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(7):971–987, 2002.

[17] J. Philbin, M. Isard, J. Sivic, and A. Zisserman. Descriptor learning for efficient retrieval. *Computer Vision–ECCV 2010*, pages 677–691, 2010.

[18] D. Stavens and S. Thrun. Unsupervised learning of invariant features using video. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 1649–1656. IEEE, 2010.

[19] N. Sunderhauf and P. Protzel. Brief-gist-closing the loop by simple means. In *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pages 1234–1241. IEEE, 2011.

[20] N. Sunderhauf and P. Protzel. Towards a robust back-end for pose graph slam. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 1254–1261. IEEE, 2012.

[21] S. Winder, G. Hua, and M. Brown. Picking the best daisy. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 178–185. IEEE, 2009.

[22] J. Wu and J.M. Rehg. Centrist: A visual descriptor for scene categorization. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 33(8):1489–1501, 2011.

[1] http://www.mrt.kit.edu/libDird.php