Efficient and Effective Matching of Image Sequences Under Substantial Appearance Changes Exploiting GPS Priors

Olga Vysotska Tayyab Naseer Luciano Spinello Wolfram Burgard Cyrill Stachniss

***Abstract*— The ability to localize a robot is an important capability and matching of observations under substantial changes is a prerequisite for robust long-term operation. This paper investigates the problem of efficiently coping with seasonal changes in image data. We present an extension of a recent approach [15] to visual image matching using sequence information. Our extension allows for exploiting GPS priors in the matching process to overcome the main computational bottleneck of the previous method and to handle loops within the image sequences. We present an experimental evaluation using real world data containing substantial seasonal changes and show that our approach outperforms the previous method in case a noisy GPS pose prior is available.**

I. Introduction

The ability to identify that a robot is at a previously visited place is an important element of localization in partially known environments. Dealing with large changes of appearance in the scene is a challenging problem but offers the potential to improve autonomous long-term navigation. This paper addresses the problem of visual localization using image sequences and a rough position estimate, for example from a simple GPS receiver. A series of robust approaches to visual localization have been proposed in the past including FAB-MAP2 [7], SeqSLAM [14], SP-ACP [16], and the work of Naseer *et al.* [15]. Some of these methods have been shown to robustly recognize previously seen locations even under a wide spectrum of visual changes including dynamic objects, different illumination, and varying weather conditions.

The contribution of this paper is an extension to our previous work [15], addressing one of its key limitation when it comes to long-term operation. The approach relies on a dense image matching matrix with the size of the number of images in the database times the length of the query sequence. This is a bottleneck when dealing with large scale datasets. In this paper, we present a technique that avoids building up the full matching matrix by exploiting an uncertain GPS prior. We achieve this by proposing a modified version of the data association graph, which is used for identifying the sequences of matched images. As a result of that, only a small fraction of the computationally expensive image comparisons needs to be conducted. As a by product of the new graph topology, we can effectively deal with loops in the image sequences without requiring to formulate and solve a network flow problem.

Olga Vysotska and Cyrill Stachniss are with Institute for Geodesy and Geoinformation, University of Bonn, Germany while Tayyab Naseer, Luciano Spinello, and Wolfram Burgard are with Institute of Computer Science, University of Freiburg, Germany. This work has partly been supported by the European Commission under the grant numbers FP7-610603-EUROPA2 and ERC-AG-PE7-267686-LifeNav.

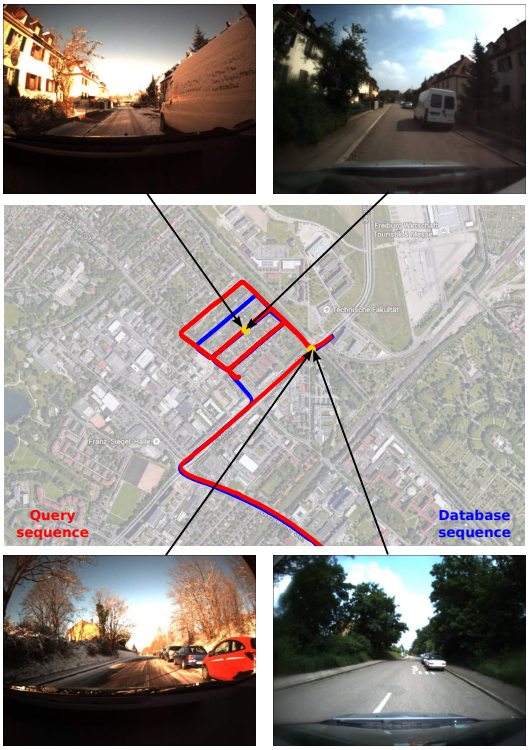

Fig. 1: Examples of typical image pairs taken at the same places in winter and summer. This paper presents an approach for effectively and efficiently addressing the problem of matching such images. Source of aerial image: Google Maps.

The approach presented in this paper to match image sequences may also support other mapping and navigation approaches, for example, the experience-based navigation framework by Churchill and Newman [6]. A possible combination of our work with experience-based navigation could enhance the ability to associate new images to an existing place by exploiting sequence information—even if individual images cannot be matched. In addition to that, we believe that our sequence-based approach can also support visual change detection in the environment—by explicitly identifying parts of the dataset which can only be matched based on sequences.

II. Related Work

Localization is a frequently investigated problem in robotics, computer vision, photogrammetry and other fields, see [9] for a survey article. Different approaches have been

proposed for visual robot localization [7], [8], [1], [3]. Dealing with substantial variations in the visual input has been recognized as a major obstacle for persistent autonomous navigation and this task has been addressed by different researchers [11], [7]. In this paper, we extend our previous work [15] for across season localization along routes to effectively exploit GPS priors and to more efficiently find matching sequences.

The majority of visual place recognition systems exploit features such as SURF [2] and SIFT [13]. Such feature-based approaches can deal with rotations and scale changes and show a great performance if the environment appearance does not change dramatically. They, however, perform rather poor under extreme perceptual changes. Across season matching using SIFT and SURF has been investigated by Valgren and Lilienthal [18] by combining features and geometric constraints. We found that SIFT and SURF features do not match robustly over seasonal changes and that systems relying on these features are often prone to errors in such settings.

SeqSLAM [14], which also aims at matching image sequences under strong seasonal changes, also computes an image-by-image matching matrix that stores dissimilarity scores between all images in a query and database sequence. This is similar to [15] and in this paper, we aim at eliminating the computational bottleneck to build a full matching matrix as this introduces a substantial computational complexity. SeqSLAM computes a straight-line path through the full matching matrix and select the path with the smallest sum of dissimilarity scores across image pairs to determine the matching route.

An interesting approach has recently been proposed by Neubert *et al.* [16]. Their method aims at predicting the change in appearance, building on top of a vocabulary. For this vocabulary, they predict the change of the visual word over different seasons. This learning phase requires an accurate image alignment. The recent approach by Johns and Young [12] builds a statistic on the co-occurrence of features under different conditions. This approach relies on the ability to stable and discriminative features over different seasons. Finding such discriminative and stable features under the strong changes is however a challenge and—from our point of view—not solved yet. To avoid addressing the problems of finding features that are robust under extreme perceptual differences, Churchill and Newman [5], [6] store different appearances for each place. These so-called experiences enable them to localize in previously learned experiences and associate a new data to places.

Biber and Duckett [4] address the problem of dealing with changes in the environments by representing maps at five different time scales. Each map is maintained and updated using the sensor data modeling short-term and long-term memories. This also enables for modeling and handling variations in the map. In contrast to that, Stachniss and Burgard [17] model different instances of typical world states using clustering. To achieve a visual localization in a long term autonomy setup, Furgale and Barfoot [10] propose a teach and repeat system that is based on a stereo setup. The approach exploits local submaps and enables a robot to navigate over long trajectories but their method does not address large perceptual changes with respect to the taught path.

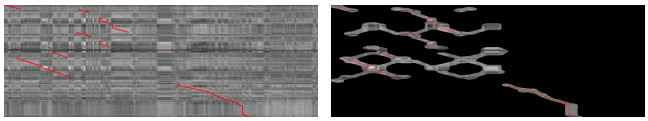

Fig. 2: Examples of matching matrices with pairs depicted in red. Left: dense matrix. Right: sparse matrix based on a GPS prior. Black pixels correspond to elements that do not need to be computed.

Our approach extends our previous work [15] by introducing the possibility of exploiting noisy GPS priors and by offering substantial savings in the computational time as the full cost matrix does not need to be computed. In addition to that, the proposed data association graph structure allows us for handling place revisits in a single pass of topological sorting and thus eliminating the need to setup and solve a network flow problem as in [15].

III. Image Matching Exploiting Sequence Information

This section briefly summarizes our previous work [15] on which this paper builds upon. The key idea of that method is to build up a dense data association graph where the nodes correspond to possible data associations of image pairs or to non-matching events (so called hidden nodes). Thus, the data association graph that relates images from two images sequences, potentially retrieved in different seasons. The set of possible paths through the data association graph thus presents the possible sequences of image matches.

To build up the data association graph, the approach needs to compute the dense matching matrix $\mathbf{C}$. The matching matrix $\mathbf{C}$ has a size of the query set times the size of the database. An element c_{ij} models the matching cost of two images and is computed using the cosine distance of a dense grid of HOG descriptors in each image.

The problem of finding path hypotheses is then addressed by means of network flows in the association graph. The possible flows in the graph represent multiple vehicle route hypotheses. By exploiting the specific structure of our graph, we could show that the network flow problem can be solved efficiently.

This approach works well to match image sequences and it was shown that it outperforms methods such a SeqSLAM or FABMAP2, given substantial changes in the appearance of the scene. The main computational bottleneck, however, is the need to build up the dense matching matrix $\mathbf{C}$ so that a quadratic number of comparisons (in the length of the smaller sequence) is needed. The contribution of this paper is to address this bottleneck by exploiting noisy GPS priors. Using such a prior allows to significantly reduce the number of image comparisons that needs to be performed, see Figure 2. In order to incorporate this information, we

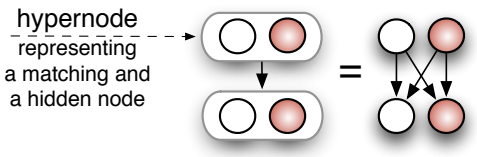

Fig. 3: A hypernode $X_{ij} = (x_{ij}, \breve{x}_{ij})$ is simplified illustration of a matching node x_{ij} and a hidden node $\breve{x}_{ij}$.

propose a new data association graph structure. Besides the fact that the number of image comparisons is reduced, it also allows us to find loops in the image sequences without the need to compute a network flow.

IV. Efficient Matching Using Pose Priors

We build upon the data association graph structure of our previous work and modify it. In contrast to [15], our modification, which requires a rough GPS prior, eliminates the need of formulating the localization problems as a network flow problem—instead a single search through the graph solves it for us. To be able to perform matching between the *database* sequence and the *query* sequence we define the image sequences to be an ordered sets of images $\mathcal{D} = (d_1, ..., d_D)$ with $D = |\mathcal{D}|$ and $\mathcal{Q} = (q_1, ..., q_Q)$ with $Q = |\mathcal{Q}|$ respectively.

Our main data structure is a data association graph $G = (X, E)$, where X is a set of nodes and E is a set of edges. The set of nodes X consists of four type of nodes: the *start state* x^s, the *goal state* x^t, the matching nodes x_{ij} and the hidden nodes $\breve{x}_{ij}$. A matching node x_{ij} represents the fact that the images i and j match, whereas a hidden node models the fact that no matching between both images could be found. For every matching node, there exists a corresponding hidden node. To simplify the notation, we group both nodes together and call this a hypernode $X_{ij} = (x_{ij}, \breve{x}_{ij})$, i.e., it represents the matching node x_{ij} and hidden node $\breve{x}_{ij}$. Visiting a matching node comes at a cost, which is proportional to the similarity of the images given the global HOG descriptor. Then, the localization problem can be described as a search for the shortest path through the graph.

Throughout this paper, *we assume that a rough GPS prior is available*—or any other global pose prior. Given this pose prior, we can avoid initiating the majority of matching and hidden nodes—a pair of matching and hidden nodes is only needed if the distance between the sensor locations was less than the prior, for example, $d_{GPS} < 500\,m$. As a result of that, only a fraction of the matching matrix $\mathbf{C}$ needs to be computed, which substantially limits the number of image comparisons that have to be conducted.

A. Edges

The set of edges E specifies, in which way the nodes can be traversed. In our representation, we use three types of edges E^s, E^t, E^X, which are identical to those of [15]. In addition to that, we define two further types of edges (E^d and E^q) to appropriately exploit the pose priors, so that $E = \{E^s, E^t, E^X, E^d, E^q\}$. The construction of the graph starts with edges E^s that connect the start node x^s with a set of matching and hidden nodes, defined as

$$E^s = \{(x^s, x_{fj}), (x^s, \breve{x}_{fj})\}_{j \in N(f)}. \tag{1}$$

In Eq. (1), f refers to the index of the first image in the query sequence for which a database image exists that has been taken in a distance smaller than d_{GPS} from f. The term $N(i)$ is defined as

$$N(i) = \{j \mid j \in \mathcal{D} \wedge \mathrm{dist}(i, j) < d_{GPS}\}, \tag{2}$$

where $\mathrm{dist}(i, j)$ is distance between the location at which the images with index i and j have been taken according the GPS prior.

The next set of edges is E^X. It models the connection between the hypernodes as:

$$E^X = \{(X_{ij}, X_{(i+1)k})\} \quad \begin{array}{l} i=1,...,Q, \\ j \in N(i), \\ k=j,...,(j+K) \text{ with } k \in N(i+1) \end{array} \tag{3}$$

where a connection between two hypernodes X_{ij} and $X_{i'j'}$ is defined as

$$\begin{aligned} (X_{ij}, X_{i'j'}) = \{&(x_{ij}, x_{i'j'}), (\breve{x}_{ij}, \breve{x}_{i'j'}), \\ &(x_{ij}, \breve{x}_{i'j'}), (\breve{x}_{ij}, x_{i'j'})\}, \end{aligned} \tag{4}$$

i.e., the edge is created from a matching node x_{ij} to a matching node $x_{i'j'}$ and as well as to a hidden node $\breve{x}_{i'j'}$. Analogously, the edges from the hidden node $\breve{x}_{ij}$ are build, see also Figure 3 for an illustration.

These edges model the potential transition from one image in the database sequence to another, when the transition between subsequent images in query sequence occurs. The value of K specifies the possible paths that are exiting from a node. Values for $K > 1$ allow for matching sequences recorded at different vehicle speeds or in case of different camera frame rates, see [15] for details.

The set of edges, E^t, connects the GPS neighbours of the last query image l for which $N(l) \neq \emptyset$ to the goal state x^t:

$$E^t = \{(x_{lj}, x^t), (\breve{x}_{lj}, x^t)\}_{j \in N(l)} \tag{5}$$

Traversing such an edge corresponds to the end of the matching process.

The presented graph structure considers the neighborhoods $N(i)$ and encodes the pose prior constraints. Exploiting GPS information yields a serious reduction in the number of image comparisons that have to be performed. In analogy to the dense matching matrix $\mathbf{C}$, this corresponds to a *sparse matching matrix* $\mathbf{C}'$ in which all not specified matches take infinity. Thus, the set E^X in this formulation can be seen as a set of edges that connects the consecutive rows of $\mathbf{C}'$ as this corresponds to the temporal order of the images in the query image sequence. The proposed structure, however, can lead to multiple separated components in $\mathbf{C}'$, see also Figure 4. These components in the matrix lead to an unconnected data association graph. Thus, the current graph topology may prevent to find the shortest path from the start to the goal.

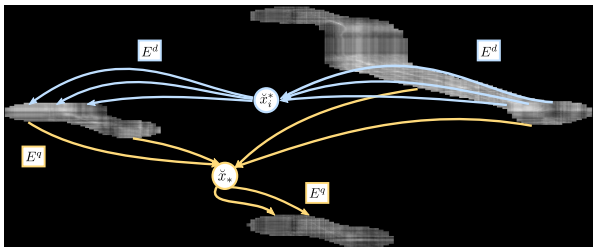

Fig. 4: Illustration of a sparse matching matrix C' and the process of connecting the separate components using E^d and E^q.

In order to compute the matching sequence as a shortest path problem, we need to connect the individual components so that every node has at least one parent node and one child node. For this, we use the new sets of edges E^d and E^q to connect nodes among components. Two situations an occur in this context.

First, a component can appear if the vehicle visited a place more than once while recording the database images. This leads to nodes that have no parent. To reconnect such components, we introduce a new hidden node $\breve{x}_i^*$ and we re-connect the components via $\breve{x}_i^*$ using edge set E^d defined as:

$$E^d = \{(X_{(i-1)k}, \breve{x}_i^*), (\breve{x}_i^*, X_{ij})\}_{k \in N(i-1)} \quad \forall X_{ij} \text{ with } \mathrm{par}(X_{ij}) = \emptyset, \tag{6}$$

where $\mathrm{par(X)}$ is the set of parents of X, i.e. all nodes that have an outgoing edges to X. The newly introduced node $\breve{x}_i^*$ exists once per line and connects all the components, which represent the same place in a real world. See Figure 4 for an illustration. Note that in such a situation there are nodes X in a component that do not have any outgoing edges, i.e., $child(X^F) = \emptyset$. We refer to nodes without children as X^F.

Second, a component can appear if the vehicle visits an area that has not been mapped in the database and then returns to a known place. This situation can easily be detected if an image i from the query set has no neighbors within the range of the pose prior, i.e. $N(i) = \emptyset$. At the point in time when query images are again close to database images, we connect them through a new hidden node $\breve{x}_*$ and the edge set E^q defined as:

$$E^q = \{(X_{i'j'}, \breve{x}_*), (\breve{x}_*, X_{ij})\}_{X_{i'j'} \in X^F} \quad \forall X_{ij} \text{ with } \mathrm{par}(X_{ij}) = \emptyset. \tag{7}$$

Afterwards all the elements from the set X^F are removed as they now have a child node. The node $\breve{x}_*$ exists for every query break, i.e. subsequences, where query dataset deviates from the area mapped in database.

B. Edge Cost

So far, we defined the vertices and edges of the data association graph but have not specified the cost associated to an edge. As finding the best matching sequence will be approached using a shortest path planner, the cost are associated to the ability to match two images.

The cost for the edges in the sets E^s and E^t are set to 0 as they are used to initiate/terminate the sequence. The weights for the edges in the set E^X as well as in the set E^d are distinguished by the type of the nodes the edges are connecting. The cost associated to an edge that connects two matching nodes $(x_{i'j'}, x_{ij})$ or that connects a hidden node with a matching node $(\breve{x}_{i'j'}, x_{ij})$ is given by the inverse matching cost $w_{ij} = \frac{1}{c'_{ij}}$, where c'_{ij} is the entry in C'. The cost of an edge that connects two hidden nodes $(\breve{x}_{i'j'}, \breve{x}_{ij})$ or a matching node with a hidden node $(x_{i'j'}, \breve{x}_{ij})$ is specified by a constant $\breve{w} = W$. This parameter can be seen as the cost of rejecting a match of two images. We determine this parameter experimentally by using a precision-recall evaluations, for more details see [15].

The edges E^d enable the graph search to treat multiple images from the same places in database alike. Thus the costs for edges from any node to x_i^* is zero, all other cost are identical to the cost in E^X.

For the edges in E^q, we use the following cost

$$w_{ij} = \begin{cases} \breve{w}(i - i' - 1) & x_{ij} \text{ is hidden node} \\ \breve{w}(i - i' - 2) + \frac{1}{c'_{ij}} & \text{otherwise} \end{cases} \tag{8}$$

with the indices i, j, i', j' as in Eq. (7). This definition of the cost replicates the cost that we would generate if using the *dense* matching matrix and moving between the components through hidden nodes. Thus, the cost is proportional to the distance in rows between the components times the cost of traversing a hidden node.

C. Normalization of the Edge Cost

As mentioned in [14], it is important to normalize the matching cost in the matrix $\mathbf{C}$ and thus $\mathbf{C'}$. We apply the normalization used in the implementation by Naseer *et al.* [15], which normalizes the cost values by the mean of cost values over each column. As we do not compute the full cost matrix $\mathbf{C}$ due to the exploitation of the pose prior, we cannot compute the same normalization. We therefore approximate it using samples. In more detail, we sample fixed number of additional image pairs (in our implementation we use 30 additional image pairs) from the same column and use this for obtaining an approximation of the normalization constant. Let μ_1 be the mean of the cost values in the column under consideration of matrix C' and μ_2 the mean of the sampled pairs along this column. Thus, we approximate the normalization constant $\mu = \frac{\mu_1 n + \mu_2 (Q-n)}{Q}$, where n is the number cost values contributing to the computation of μ_1.

D. Complexity

For the complexity analysis, we assume that the covered areas are substantially larger than the GPS range, so that for all query images, only a bounded number of elements from the database is within the d_{GPS}. The highest complexity is the task of finding the images in the database that have been taken near a given location according to the pose prior. We achieve this through a kd-tree, yielding a logarithmic complexity in the size of the database. Overall, this results

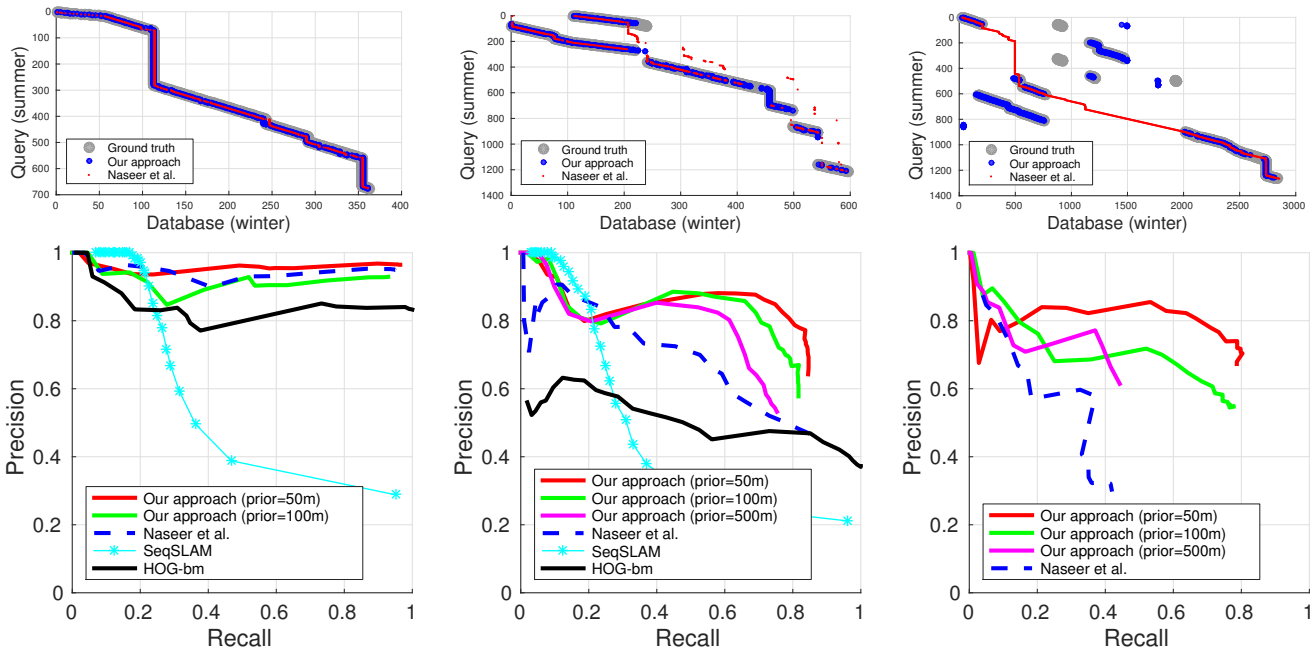

Fig. 5: Comparisons to other methods. The images in the first row show the matches, including ground truth and the plots in the second row show the precision recall plots. *First column:* Comparison of our approach the method Naseer *et al.* [15], openSeqSLAM and a heuristic that always selects the best match in $\mathbf{C}$ on a dataset that consists of a sequence of 3 km. *Second column:* Comparison between the same approaches on a dataset containing a loop in a query sequence. *Third column:* Comparison to the method of Naseer *et al.* on a third dataset containing several loops in database as well as in query.

in $\mathcal{O}(Q \log D)$. Due to the directed acyclic graph structure, computing the shortest path via topological sorting yields a complexity of $\mathcal{O}(|X| + |E|)$.

V. Experiments

The evaluation is designed to illustrate the performance of our approach and to support the three main claims made in this paper. These three claims are: (i) we can exploit GPS pose priors to substantially reduce the computational load of the image matching process, (ii) we can naturally handle loops without the need of using network flow algorithms, (iii) we can either improve the matching results or perform comparably to our previous work.

All our experiments have been conducted using real world data, recorded in summer and winter. The data has been obtained with a bumblebee camera mounted in a regular car. The algorithm works with image of resolution 1024×768, no cropping, undistortion or other preprocessing is done. Examples for matching image pairs from the datasets can be seen in Figure 6. We evaluate the performance of our algorithm by precision-recall curves, which are computed based on manually labeled ground truth image matches. We calculate precision as $\frac{\mathrm{TP}}{\mathrm{TP+FP}}$ and recall as $\frac{\mathrm{TP}}{\mathrm{TP+FN}}$. A match is considered as a *true positive* (TP) if the found match and the manually provided match differs by up to three images within the sequence. If an image pair is not within the specified boundaries than it is considered as a *false positive* (FP). All the ground truth pairs that were not found by algorithm are considered as *false negatives* (FN). To obtain the precision-recall curves, we vary the parameter $\breve{w}$ from small to large values. If $\breve{w}$ takes a values that is smaller than the smallest element in the matrix, all potential matches will be rejected. With increasing values for $\breve{w}$ more and more potential matches will be accepted.

Fig. 6: Four example image pairs from the database and query set illustrating the perceptual change over the seasons.

The first experiment is designed to show that our approach performs comparable to our previous approach [15], which constructs the full matching matrix. As the precision-recall plots as well as the matching performance with respect to the (manually labeled) ground truth information suggests, we perform equally or even better than the original method, see Figure 5. The images in the first row of that figure show the ground truth matches, as well as the performance of our approach w.r.t. [15]. In the first dataset (see first column of Figure 5), both approaches show a similar performance as also the precision recall plots in the second row suggest.

The second experiment contains a loop, i.e., the robot revisited places stored in the database twice (approx. the first 200 images). As can be seen from second column of Figure 5, our approach—although solving the data associa-

TABLE I: Number of descriptor comparisons needed to build the cost matrix is shown in the first rows. The second rows show the overall computation time (1st value) and the time needed to find the matching sequence through the given graph (2nd value).

	Dataset				
	1	2	3	4	5
Q	79	676	1,213	1,266	1,428
D	943	361	596	3,601	1,476
[15]	74,498	244,037	722,948	4,558,866	2,107,728
	100s / 0.7s	277s / 2s	798s / 6s	4,843s / 55s	2,305s / 19s
GPS	74,498	134,791	298,432	2,620,748	1,102,689
500m	100s / 0.7s	155s / 1s	325s / 2s	2,734s / 23s	1,283s / 10s
GPS	50,643	38,334	72,788	621,369	106,672
100m	47s / 0.2s	52s / 0.23s	107s / 0.5s	660s / 4s	138s / 0.6s
GPS	38,313	26,841	45,457	288,312	76,171
50m	33s / 0.1s	42s / 0.1s	70s / 0.2s	316s / 1s	100s / 0.25s

tion problem using topological sorting—can handle the loops better than the network flow solution in [15]. This is also visible from the precision recall plots shown in the second row. They also illustrate that we outperform openSeqSLAM and a best match strategy based on the HOG descriptors ignoring the sequence information (called "HOG-bm").

Finally, we used a third dataset, a more challenging one, since it contains multiple loops in database and in query sequences. Here, the car was driving in circles around perceptually similar blocks. The trajectory is also illustrated in Figure 1. Similarly to before, the usage of a GPS prior enables us to better match the corresponding parts of the query to database trajectory. In the third column of Figure 5, we observe in the precision recall plot that our method with $d_{GPS} = 50\,\text{m}$ gives the best results. This is due to the fact that distance between the parallel streets in the dataset is smaller then $100\,\text{m}$ and using $d_{GPS} = 50\,\text{m}$ leads to clearly disconnected components in cost matrix.

The next experiment is designed to illustrate that exploiting the GPS prior can lead to a substantial reduction in the computational requirements. Table I summarizes the timing results of evaluating the algorithms on datasets of different size and complexity. The table shows the total number of global image comparisons for each approach (all values in the first rows). In addition to that, the table lists (in the second rows) the overall runtime of the algorithm as well as the time needed to compute the path through the matching matrix—the latter ones are the timings reported in [15].

As can be seen from the table, the datasets of smaller size tend to show a smaller gain in terms of the overall reduction of image comparisons and thus nodes in the data association graph compared to larger datasets. This is due to the fact that smaller datasets typically cover more near-by places and thus yield a denser matching matrix. In general, we can say the larger the area that the dataset covers, the bigger the gain of our method. For the spatially distributed datasets, the factor of the computations savings is between 9 and 27 as a substantially smaller number of nodes needs to be created.

Most of the computation time is spent on the comparison of the image descriptors. The computation of the global HOG descriptor per image, described in [15], takes around $23\,\text{ms}$ and matching two descriptor takes around $6\,\text{ms}$—but this has to be done often. The GPS priors help to reduce the overall number of comparisons and thus lead to a substantial reduction of the computation times.

VI. Conclusion

We proposed an approach to visual image matching under substantial appearance changes by exploiting sequence information. We extended our recent approach [15] so that it can exploit noisy GPS pose priors and at the same time substantially reduce the number of required image comparisons and yields a substantial speed-up of the approach. In addition to that, our approach can naturally handle loops in the input image sequences. We implemented and tested our approach using real world image and GPS data acquired in summer and in winter. Our comparisons suggest that our approach can increase the matching performance while reducing the computation time and in this way outperforms the existing methods under consideration.

References

[1] M. Agrawal and K. Konolige. Frameslam: From bundle adjustment to real-time visual mapping. *IEEE Trans. on Robotics*, 24(5), 2008.

[2] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Speeded-up robust features (SURF). *Comput. Vis. Image Underst.*, 110(3):346–359, 2008.

[3] M. Bennewitz, C. Stachniss, W. Burgard, and S. Behnke. Metric localization with scale-invariant visual features using a single perspective camera. In *European Robotics Symposium*, pages 143–157, 2006.

[4] P. Biber and T. Duckett. Dynamic maps for long-term operation of mobile service robots. In *Proc. of Robotics: Science and Systems*, pages 17–24. The MIT Press, 2005.

[5] W. Churchill and P. Newman. Practice makes perfect? managing and leveraging visual experiences for lifelong navigation. In *Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2012.

[6] W. Churchill and P. Newman. Experience-based Navigation for Long-term Localisation. *Int. Journal of Robotics Research*, 2013.

[7] M. Cummins and P. Newman. Highly scalable appearance-only SLAM - FAB-MAP 2.0. In *Proc. of Robotics: Science and Systems*, 2009.

[8] A.J. Davison, I.D. Reid, N.D. Molton, and O. Stasse. Monoslam: Real-time single camera slam. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29:2007, 2007.

[9] J. Fuentes-Pacheco, J. Ruiz-Ascencio, and J.M. Rendón-Mancha. Visual simultaneous localization and mapping: a survey. *Artificial Intelligence Review*, pages 1–27, 2012.

[10] P.T. Furgale and T.D. Barfoot. Visual teach and repeat for long-range rover autonomy. *Int. J. Field Robotics*, 27:534–560, 2010.

[11] A.J. Glover, W.P. Maddern, M. Milford, and G.F. Wyeth. FAB-MAP + RatSLAM: Appearance-based slam for multiple times of day. In *Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 3507–3512, 2010.

[12] E. Johns and G.-Z. Yang. Feature co-occurrence maps: Appearance-based localisation throughout the day. In *Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2013.

[13] D.G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2):91–110, 2004.

[14] M. Milford and G.F. Wyeth. Seqslam: Visual route-based navigation for sunny summer days and stormy winter nights. In *Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2012.

[15] T. Naseer, L. Spinello, W. Burgard, and C. Stachniss. Robust visual robot localization across seasons using network flows. In *Proc. of the AAAI Conference on Artificial Intelligence*, 2014.

[16] P. Neubert, N. Sunderhauf, and P. Protzel. Appearance change prediction for long-term navigation across seasons. In *Proc. of the European Conference on Mobile Robotics (ECMR)*, 2013.

[17] C. Stachniss and W. Burgard. Mobile robot mapping and localization in non-static environments. In *Proc. of the AAAI Conference on Artificial Intelligence*, pages 1324–1329, 2005.

[18] C. Valgren and A.J. Lilienthal. SIFT, SURF & Seasons: Appearance-based long-term localization in outdoor environments. *Robotics and Autonomous Systems*, 85(2):149–156, 2010.