Lazy Sequences Matching Under Substantial Appearance Changes (Short Paper)

Olga Vysotska Cyrill Stachniss

***Abstract*— The ability to localize in changing environments is essential for robust long-term navigation. Robots operating over extended periods of time must be able to handle substantial appearance changes. In this paper, we investigate the problem of efficiently coping with seasonal changes in online localization. We propose an online lazy data association approach for matching streams of incoming images to a reference image sequence. We propose a search heuristic to quickly find matches between the current image sequence and the database. We present an experimental evaluation using real world data containing substantial seasonal changes and show that our approach can efficiently match sequences by requiring comparably small number of image comparisons.**

I. Introduction

The ability to identify a previously visited place is an important element of robot localization. Handling large appearance changes such as those depicted in Fig. 1 is a challenging problem. Dealing with substantial variations in the visual input is key for persistent autonomous navigation and this task has been addressed by different researchers [4], [5]. The majority of visual place recognition systems exploit features such as SURF or SIFT. 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.

Several approaches for aligning image sequences have been proposed in recent years. SeqSLAM [8], for example, computes a matching matrix that stores dissimilarity scores between all images in a query and database sequence. It computes a straight-line path through the full matching matrix and selects the path with the smallest sum of dissimilarity scores to determine the matching route. Related to that, Naseer *et al*. [9] focus on offline sequence matching using a network flow approach. A further interesting approach has recently been proposed by Neubert *et al.* [10]. 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.

A recent approach by Johns and Young [7] builds a statistic on the co-occurrence of features under different conditions. It relies on the ability to detect stable and discriminative features over different seasons. Finding such discriminative and stable features under the strong changes is however a challenge on its own. To avoid finding features that are robust under extreme perceptual differences, Churchill and Newman [2], [3] store different appearances for each place. These so-called experiences enable them to localize in previous sequences and associate the new data to places.

Olga Vysotska and Cyrill Stachniss are with Institute for Geodesy and Geoinformation, University of Bonn, Germany. This work has partly been supported by the European Commission under the grant numbers FP7-610603-EUROPA2.

Fig. 1: Example images of the datasets used in our experiments. Upper: Freiburg dataset (seasonal changes); Middle: Nordland dataset (seasonal changes); Bottom: dataset for VPRiCE'15 challenge (daily changes).

We propose an online image sequence matching approach that builds upon our recent work on offline matching [9], [12] and the lazy data association approach of Hähnel et al. [6]. We apply deeply learned features as proposed by Chen et al. [1] as they provide a superior matching performance than for example HOG features in our settings. We propose a heuristic that estimates the expected cost of matching images based on a statistic of the best matches found so far. Our approach can handle multiple parallel hypotheses of matching image sequences. To achieve that, we build upon the ideas of our previous work but instead of building a matching matrix, we perform the search in the data association graph in an online fashion. The graph is built incrementally and its leaf models the data association hypotheses that are currently under consideration.

II. Lazy Matching for Online Matching

This paper proposes an online algorithm that uses image sequences to perform global localization under strong appearance changes. We perform localization in the sense that we match a sequence $\mathcal{Q}$ of the images that we receive from the robots sensors with a reference or database sequence of images called $\mathcal{D}$. For every incoming image, we want to know if there is a corresponding match in the database and if so, to which image it corresponds to.

A. Data Association Graph

We build upon our previous work [9], [12] and use a directed acyclic graph $G = (X, E)$ as our main data structure for modeling the data association problem. We can model the sequential image matching task as finding a shortest path in this data association graph, see [9]. In our work, we build up the graph on the fly and only need to perform an image comparison if our search algorithm expands the corresponding node of the graph. The key idea of the data association graph can be explained as follows. Each node in the graph represents a potential match between two images. We aim at finding the best combination of matching images by searching a path through this graph where the cost of visiting a node depends on the similarity of both images. The graph consists of the following elements.

a) Nodes: We have two types of nodes in X: the root or start node x^s and matching nodes. A matching node x_{ij} models a match of the image $i \in \mathcal{Q}$ is with the image $j \in \mathcal{D}$. The more similar two images are, the more likely is the fact that they may represent the same place. The similarity of an image is defined as $c_{ij} \in [0, 1]$ where 1 means both images appear identical. The similarity c_{ij} is computed by comparing the images $i \in \mathcal{Q}$ and $j \in \mathcal{D}$. As we are building up the graph online, new nodes x_{ij} need to be created as soon as a new image i is recorded. Adding a node x_{ij} to the graph, however, comes at a *computational cost* as we need to compare the images and compute c_{ij}. Thus, for building up the graph, we should avoid instantiating unnecessary nodes x_{ij}.

b) Edges: Similar to the nodes, we use two types of edges $E = \{E^s, E^X\}$ according to the types of nodes the edges connect. Set of edges E^s connects the source node x^s with the matching nodes x_{ij} corresponding to matching the query first image with any database image $j \in \mathcal{D}$, i.e.,

$$E^s = \{(x^s, x_{0j})\}_{j \in \mathcal{D}}. \tag{1}$$

The second set of edges E^X, which was also used in a similar form in [12], [9], connects the matching nodes. In this approach, we define the set E^X of edges slightly different to the one defined before as

$$E^X = \{(x_{ij}, x_{(i+1)k})\}_{k=j-K,\ldots,j+K}, \tag{2}$$

where K is a "fanout" parameter that influences the nodes that are connected between the query images i and $i + 1$. The fanout basically models that the cameras can move at different speeds through the environment or that the cameras can operate at different framerates. The nodes $x_{(i+1)k}$ are furthermore specified as $\mathrm{ch}(x_{ij})$, i.e. are the children of the node x_{ij}.

c) Weights: Each edge in E has a weight. This weight is related to the cost c_{ij} defined above. The weight of an edge $e = (x_{ij}, x_{i'j'}) \in E^X$ is inverse proportional to the similarity of the node to which this edges leads to, i.e. $w(e) = \frac{1}{c_{i'j'}}$, where $c_{i'j'}$ is a cost of matching image i' and j'.

B. Computing Image Similarity with Features from Deep Convolutional Neural Networks

As we have pointed out before computation of the matching cost c_{ij} between two images has to be performed often and thus we are interested in a fast computation. Nevertheless, the quality of the similarity function is of high importance. The larger c_{ij} for the images taken from the same place and the smaller c_{ij} for images taken from different place, the better. The more distinct such values are, the better the performance of our graph search algorithm as less nodes will need to be expanded as well.

In our previous works [9], [12], we computed the global HOG descriptor. HOG-based image comparisons were sufficient to find good solution with an exhaustive search. In the context of the lazy data association approach with a non-admissible heuristic, we experience problems to find matching sequences reliable without expanding the whole graph. Therefore, we changed the image descriptors in this work to the deeply learned features from the pre-trained image recognizer and feature extractor OverFeat as proposed by Sermanet *et. al* [11]. OverFeat is built using a deep convolutional neural network trained on the ImageNet dataset. We used the results of 10^{th} layer as it was reported by Chen *et. al* [1] to give the best results in their work on place recognition tasks. For each image the descriptor of size $512 \times 18 \times 24$ was extracted. Using Overfeat features instead of HOG directly improves the performance of our algorithm and makes the lazy approach possible.

C. Image Sequence Matching by Graph Search

The sequence of matching images between $\mathcal{Q}$ and $\mathcal{D}$ can be computed by a shortest path search from the start node x^s to any node x_{l*}, with * referring to any index in $\mathcal{D}$, where l is the most recent image in $\mathcal{Q}$. Every node that is a part of the shortest path corresponds to a selected data association.

The computationally most demanding process for building up and searching in such a data association graph is instantiating nodes as a large number of possible matches may be created. For online localization, we are interested in keeping the computational efforts small and avoiding creating too many nodes. To address this issue, we propose the algorithm that keeps the number of image comparisons that need to be performed small and thus results in an efficient algorithm.

Our work is motivated by the ideas of lazy data associations in the context of graph-based SLAM proposed by Hähnel *et al.* [6] for constructing a graph. Hähnel *et al.* build up a data association tree and expand in each round the node with the highest log likelihood of representing a

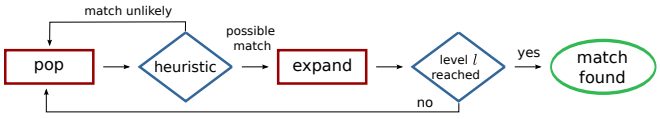

Fig. 2: Illustration of searching for a match for an input image.

match between laser range scans. This basically is similar to a greedy search in a data association tree.

In our case, we go a step further and seek to performing an *informed* search through the graph, while the graph is built up on demand. One popular way to perform an informed search is the A* algorithm in combination with a heuristic, which allows us to estimate to cost from the current node to the goal node. For our matching problem, that means we need to predict how well the images that we will receive in the future will match our database images —this is in general difficult task. Furthermore, A* requires that the heuristic is a predefined function and does not change during the search. We, however, take a different approach and try to predict the matching cost based on the images that we have received so far. This means, our heuristic is updated *during the search*, which, unfortunately, prevents the use of standard A*. Our search procedure taking into account the estimated matching cost works as follows.

Similar to A*, we use an open-list F of nodes that are still under consideration. This open-list is realized through a priority queue. In contrast to A*, the key of the priority queue for a node x_{ij} is the cost $g(x_{ij})$ of reaching x_{ij} from the source x^s. Our search and simultaneous graph construction starts with creating the source node x^s and connecting it to the matching nodes according to Eq. (1). This step requires to instantiate $|\mathcal{D}|$ nodes if no further information about the first possible match is provided.

For every incoming image q_l, we use the following procedure to update the graph as well as the matching sequence (see Fig. 2 for a brief illustration): Whenever a new image q_l is obtained, we pop a node from F. We then use our heuristic, which will be described in the remainder of this section, to estimate if the node x_{ij} is worth expanding or is unlikely to be part of the matching sequences given the cost estimate. If the node is unlikely to be part of the matching sequences, we continue with the next node in F. Otherwise, we expand the node x_{ij} by computing the matching costs for its children $\mathrm{ch}(x_{ij})$ and connecting the node x_{ij} with $\mathrm{ch}(x_{ij})$ using the edges define by E^X. If a node in $\mathrm{ch}(x_{ij})$ lies on the l depth level of the graph, then it represents the so far best match for q_l and the search terminates for this input image. Otherwise we proceed expanding nodes from F.

The above described method relies on a heuristic to estimate the sum of matching costs for reaching the l^{th} level (given that the last obtained image is q_l). The key problem here is that defining an effective *and* admissible heuristic is hard due to the small amount of background information that can be exploited to predict future image matching cost. Therefore, we take an alternative approach to come up with a heuristic that provides a good estimate of the cost. We take a statistical approach and approximate an *expected* lower

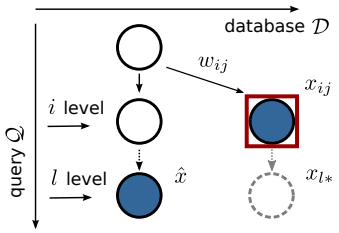

Fig. 3: Illustration for the graph expanding procedure. Blue nodes are nodes in the F. The red square indicates that the element x_{ij} will be the next one in F. The dashed grey line represent nodes and edges not computed yet.

bound for the *average* cost of the unexpanded and thus unknown nodes. We do so by using the average cost of the best path found so far as a prediction of the lower bound of the cost. Furthermore, we exploit that we know the number of images obtained so far, i.e., we know that the shortest path will have $l+1$ nodes (start node plus one matching node for each image). This allows us to formulate the expected cost $f(x_{l*})$ for a node x_{l*} as the computed cost from x^s to x_{ij} expressed through $g(x_{ij})$ plus the estimate cost as:

$$f(x_{l*}) = g(x_{ij}) + \underbrace{\alpha(l-i)\mu_{\text{cost}}(\hat{x})}_{\text{heuristic}} \tag{3}$$

where $\alpha \in (0, 1]$ is a factor to trade off the quality of the solution and the number of nodes that needs to be expanded. For $\alpha \to 0$, we obtain a greedy search behavior and for $\alpha = 1$ we may not expand enough nodes to find a good solution. The term $(l-i)$ is the number of images that should be matched to end the sequence and $\mu_{\text{cost}}(\hat{x})$ is the average cost of the best path found so far, see also Fig. 3.

III. Experiments

The evaluation is designed to illustrate the performance of our approach and to support the two main claims made in this paper. These two claims are: (i) our approach has the ability to run in an incremental fashion so that only few nodes are expanded and that online localization is possible, (ii) our heuristic is well suited to find a competitive solution in most real world situation.

Throughout our evaluation, we rely on multiple publicly available datasets, see Fig. 1. First, we use the summer-winter dataset used in [9], [12], later referred to as *Freiburg*. Second, the *Nordland dataset*, which is a four season train ride dataset from Norway. Finally, we used the datasets that have been selected for the VPRiCE Challenge 2015. The latter one consists of 4022 query and 3756 database images organized as a single sequence but being stitched together from multiple different datasets.

The first experiment is designed to show that we can achieve online performance as only a comparably small number of nodes gets expanded. For this experiment, we varied the scaling parameter α of our heuristic in Eq. (3) between 0 and 1. Zero basically leads to a greedy search, while $\alpha = 1$ approximates the expected cost by the average cost of the best path.

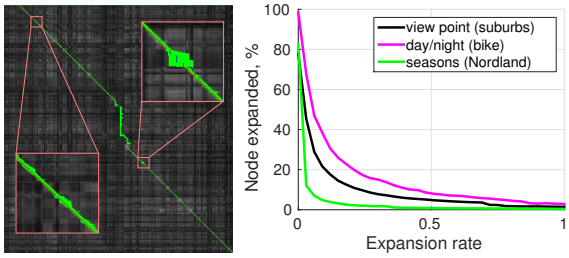

Fig. 4: Left: visualization of the graph structure for the dataset with dramatic seasonal changes (Nordland). The algorithm computes the matching costs only for the nodes marked with green. Other nodes are computed for visualization only. Right: Plot of the dependency between the expansion rate α and the number of matching cost computations, expressed in percentage from total number of nodes.

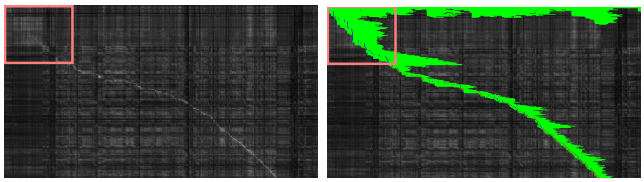

Fig. 5: Full matching matrix (left) and the nodes expanded by our algorithm (green nodes in the right image). The cost matrix is computed for visualization only. The squares highlights an area with hard to identify matches, which leads to a larger node expansion.

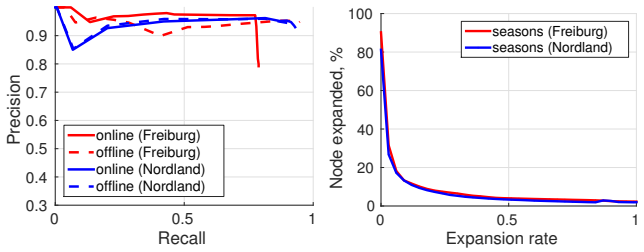

Fig. 6: Left: Comparison between our algorithm (online) and [9] (offline). Right: Dependence between the expansion parameter α and number of feature comparisons relative to the total number of comparisons $|\mathcal{Q}| \times |\mathcal{D}|$.

Fig. 4 depicts subset of the VPRiCE dataset with strong seasonal changes. In sum, our algorithm computes matches for $29,317$ image pairs out of $5,693,135$ possible matching, that the standard approaches such as [9] would expand. This yields a reduction of computation cost of 99.5%. Similar reductions can be noticed for other datasets, see right image of Fig. 4, where the larger the dataset the bigger the savings. Also the distinctiveness of the matching costs plays a role for our algorithm. As it can be seen in Fig. 5, the block of the matching matrix in the upper left corner shows no distinct matching pattern. As result, our approach expands a comparably large number of nodes, indicated by the green elements in the right image. Note that our algorithm does not need the full matching matrix, we depict it here for visualization only.

Computing the image descriptor takes the largest amount of time with approx. 500 ms. Expanding a single node, i.e., comparing two descriptors, takes 8 ms. Incremental update of the shortest path takes around 40 ms on average. As a result of that, our approach can run online with around 1 fps.

The second set of experiments is designed to show that the proposed heuristic does not degrade the matching performance. We confirm this statement by comparing our results with our previous approach using the full matching matrix using the Freiburg dataset and the Nordland dataset as ground truth information is available. The results are depicted in Fig. 6. We used $\alpha = 0.8$ and the parameter varied to obtain the precision recall plots was the non-matching cost $\breve{w}$, see [9] for details. As can be seen, our heuristic leads to comparable results for both datasets. Furthermore, the number of image comparisons that needed to be performed drops dramatically with increasing the expansion rate. Thus, we can reduce the number of matching operations while maintaining a high matching performance.

IV. Conclusion

We proposed an incremental approach to image sequence matching under substantial appearance changes for online operation. The key idea is to apply a lazy data association approach and define a heuristic for the search in the data association graph that estimates the path cost. This allows us to achieve online performance for image matching under substantial appearance changes. We implemented and tested our approach using real world data. The experiments suggest that our approach provides comparable results while it can run online and avoids expanding large portions of the data association graph.

References

[1] Z. Chen, O. Lam, A. Jacobson, and M.Milford. Convolutional neural network-based place recognition. *arXiv:1411.1509*, 2014.

[2] 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.

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

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

[5] 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.

[6] D. Hähnel, W. Burgard, B. Wegbreit, and S. Thrun. Towards lazy data association in slam. In *Proc. of the Int. Symposium of Robotics Research (ISRR)*, pages 421–431, Siena, Italy, 2003.

[7] 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.

[8] 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.

[9] 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.

[10] 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.

[11] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. In *Int. Conf. on Learning Representations (ICLR)*, 2014.

[12] O. Vysotska, T. Naseer, L. Spinello, W. Burgard, and C. Stachniss. Efficient and effective matching of image sequences under substantial appearance changes exploiting gps priors. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, 2015.