Incremental Topological Mapping Using Omnidirectional Vision

Christoffer Valgren, Achim Lilienthal
Centre for Applied Autonomous Sensor Systems
Örebro University
SE-701 82 Örebro, Sweden
{*christoffer.wahlgren, achim.lilienthal*}*@tech.oru.se*

Tom Duckett
Department of Computing and Informatics
University of Lincoln
Brayford Pool, Lincoln, LN6 7TS, United Kingdom
tduckett@lincoln.ac.uk

***Abstract*— This paper presents an algorithm that builds topological maps, using omnidirectional vision as the only sensor modality. Local features are extracted from images obtained in sequence, and are used both to cluster the images into nodes and to detect links between the nodes. The algorithm is incremental, reducing the computational requirements of the corresponding batch algorithm. Experimental results in a complex, indoor environment show that the algorithm produces topologically correct maps, closing loops without suffering from perceptual aliasing or false links. Robustness to lighting variations was further demonstrated by building correct maps from combined multiple datasets collected over a period of 2 months.**

I. Introduction

There are two main types of maps for robots: metric and topological. Because the metric map, which is a representation of the world in two or three dimensions, typically is used for detailed path planning and obstacle avoidance, it is required that the map has a certain accuracy. This accuracy comes at the cost of memory and processor cycles. The topological map, on the other hand, stores only distinct places and the links between these places, and is thus a more efficient representation for large-scale navigation. However, to build a correct topological map, a mapping algorithm must be able to identify places reliably (correspondence problem), while being robust to perceptual aliasing (where multiple places have similar appearance).

The long term aim of our research is on-line mapping for mobile robot navigation in both indoor and outdoor environments. Many challenges await; outdoor environments can be huge, unstructured and dynamic. Many of the existing solutions to the robotic mapping problem indoors will not be applicable outdoors [1], simply because the methods employed do not scale well to larger and more complex environments.

Topological approaches use landmarks to define the nodes (places) of the map. Clustering the sensor data into nodes is one problem that all topological mapping schemes have to solve. Existing approaches include defining the places by hand, unsupervised clustering algorithms [2], and detection of "distinctive places" [3]. If the whole set of landmark observations is available, the mapping problem can be viewed as a search to find the best topological map that fits the data. The landmarks can be artificial landmarks such as beacons, or naturally occurring features extracted by sensors such as laser range finders, sonars or cameras.

The incremental method we propose manages to accurately construct a topological map, yet the algorithm remains fairly simple. The *link likelihood* (section III-C) is determined by visual similarity of image sequences, so the map can be interpreted from a probabilistic viewpoint, while still being computationally cheap.

II. Related Work

There are many approaches to using vision for mobile robot localization and mapping. Ulrich and Nourbakhsh [4] used color histograms to calculate topological maps. Gaspar et al. [5] applied principal component analysis to condense a large data set of panoramic images into a smaller set of eigenimages that was used for localization.

Extracting local features from panoramic images has proved to be a successful approach to localization. Košecká and Li [6] showed that the descriptors of the Scale Invariant Feature Transform (SIFT) were superior to using orientation histograms as landmarks. Modified SIFT features were used for topological localization in non-stationary environments by Andreasson and Duckett [7], and Zivkovic et al. [8] combined SIFT with geometrical constraints to obtain a topological map from non-sequential images. Bradley et al. [9] showed that it was possible to perform localization over several kilometers using local features extracted from monocular images.

Closely related to the method presented in this paper is the work by Tapus and Siegwart [10], where they extract "fingerprints" from panoramic images and laser scan data. New nodes are added to the map whenever an important change in the environment (or, rather, to the fingerprint) occurs. The approach by Tapus and Siegwart differs from ours, in particular in that they rely on additional data from a laser scanner and that their approach does not consider the problem of loop closing.

III. Topological Mapping Using Vision

This section describes our proposal for topological mapping using vision. The approach utilizes local features extracted from panoramic images combined with a segmentation technique to derive a topological map from a sequence of images

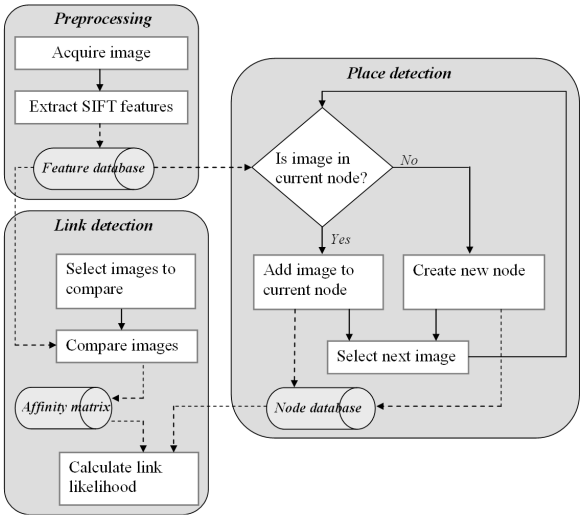

Fig. 1. Flowchart of the three processes that make up the incremental version of the algorithm.

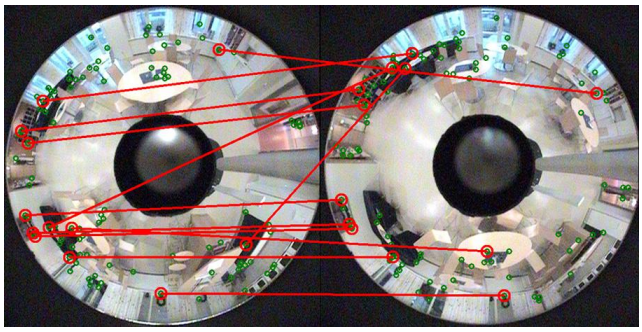

Fig. 2. Some of the features matched between two panoramic images. Small circles indicate the positions of extracted MSIFT features, large circles connected with a line indicate a match.

obtained by a mobile robot. Two versions of the algorithm are presented. The first version performs the calculations in a batch; it utilizes a greedy search to find the nodes and the links. The second version is incremental, while the computation time at each time step is approximately constant (i.e. although there are variations from time step to time step, there will not be a net increase over time). This is achieved by using a random search, guided by heuristics, through the search space.

Both approaches have been shown to give good experimental results in a medium-sized, complex indoor environment (Figure 5). Furthermore, the incremental version (Figure 1), although being much faster, shows comparable results to the batch version.

A. Matching images using local features

The images are acquired by an omnidirectional camera, which consists of a curved mirror lens mounted below a camera. Local features are extracted from the images. We use a variant of SIFT, the Scale-Invariant Feature Transform, which was first presented by David Lowe in 1999 [11]. The main characteristic of SIFT is that it uses a feature description that is invariant to scaling and rotation. It is also partially invariant to changes in illumination and camera location. In this work, we use the Modified SIFT (MSIFT) algorithm of Andreasson and Duckett [7]. The interest points are selected from the image by using the algorithm *GoodFeaturesToTrack* proposed by Shi and Tomasi [12]. The MSIFT algorithm uses the same keypoint descriptors as the SIFT algorithm, but the descriptors are only found in one resolution, because full invariance to scale and translation is not required. In fact, in this application, scale sensitivity is desirable, since we want to only recognize a feature from viewpoints within a small area or "place".

The local features extracted from one image can be matched to features from another image. Using local features for image comparison in this way has several advantages over methods that use global features for the comparison: it is less sensitive to occlusion and changing environments [13], and it is possible to directly use the number of feature matches as a measure of image similarity. There is, however, always a risk that some features will be wrongly matched. We set a threshold N_{min} for the minimum number of local feature matches before two images are said to match each other.

The feature matching algorithm calculates the Euclidean distance between each feature in image i and all the features in image j. (In the MSIFT algorithm, the keypoint descriptors are not normalized (as in the original SIFT algorithm) before the Euclidean distance is calculated.) A potential match is found if the smallest distance is smaller than 60% of the second smallest distance. Note that a feature f_i in image i may match feature f_j in image j, without f_j matching f_i. Therefore, the potential matches are considered using the following rules:

- All reciprocal matches (i.e. features mutually matching each other) are found.
- Any other match is found, where better (i.e. of higher quality, as defined by *GoodFeaturesToTrack* [12]) features are matched first.
- We only allow a feature in image i to match once to a feature in image j.

An example of feature matching is shown in Figure 2.

B. Nodes

In our approach, a *node* is a collection of images that are considered similar enough, i.e., there is a sufficient number of feature matches between the images. A node always consists of images in sequence. By using only local feature matching as the criterion for determining boundaries between nodes, we do not require any knowledge about the geometry of the environment. This gives the algorithm the potential to work well in unstructured environments. On the other hand, there is also a risk that we will misclassify an image because of noise, occlusion or perceptual aliasing. Using geometrical constraints in the algorithm might reduce the risk of such misclassification, and incorporating odometry data is indeed on the list for future work.

C. Links

To get a complete topological map, the nodes detected need to be connected to each other. Because the images are assumed to be obtained in sequence, there are *weak links* (following Lu and Milios [14]) between consecutive nodes in the sequence. In addition, a *strong link* exists between node A and node B when an image in node A matches an image in node B. We expect strong links to form between nodes in larger areas, because some features are common to some images in the nodes. Also, we expect that if we revisit a previously visited area, a link will be created between the old and new nodes corresponding to that area. Again, the feature matching is not perfect, which means that there is a risk that *false links* will be created. Working towards a probabilistic approach, we propose the concept of link likelihood, which measures the probability that two nodes share common features. Let the number of feature matches between image i and image j be denoted $C(i,j)$. The link likelihood L_{AB} is a (heuristic) function of the number of image matches between node A and B:

$$L_{AB} = 1 - k^p \frac{N_{min}^2}{\sum_{i \in A, j \in B} C^*(i,j)^2} \tag{1}$$

$$C^*(i,j) = \begin{cases} 0 & \text{if } C(i,j) < N_{min}; \\ C(i,j) & \text{if } C(i,j) \geq N_{min}. \end{cases} \tag{2}$$

where k is the number of image pairs where $C(i,j) > N_{min}$, N_{min} is the minimum number of feature matches between images for an image match, and p is a constant usually slightly less than or equal to 1. Note that $0 \leq L_{AB} \leq 1$. The function L_{AB} is non-linear, and assigns a high likelihood to links that have at least one high value of $C(i,j)$. The constant $0 \leq p \leq 1$ gives a slightly higher likelihood to links that have more than one similar value of $C(i,j)$.

It is worth noting that in the data sets used in the paper, the maximum value of $C(i,j)$ was 70 (the total number of extracted features N_{max} was 101). Even in the case of two consecutive images, the slight change in position is sufficient to reduce the number of matches substantially. In practice, it is very rare to find a value of $C(i,j)$ higher than $\frac{N_{max}}{2}$.

D. Finding the links

The link likelihood defined in (1) is based on the number of image matches between node A and B. The *affinity matrix* stores the results of the image comparisons so that the entry at position (i,j) corresponds to the value of $C(i,j)$. An example affinity matrix is shown in Figure 3. The number of *possible* image comparisons increases quadratically with the number of added images. If we were to compare every new image with all previous images, the computation required would quickly grow beyond what is feasible, for all but very small data sets. Comparing each image with every other image could be viewed as doing an exhaustive search for links in a quadratically growing search space. In terms of the affinity matrix, this means calculating a new row for every image added.

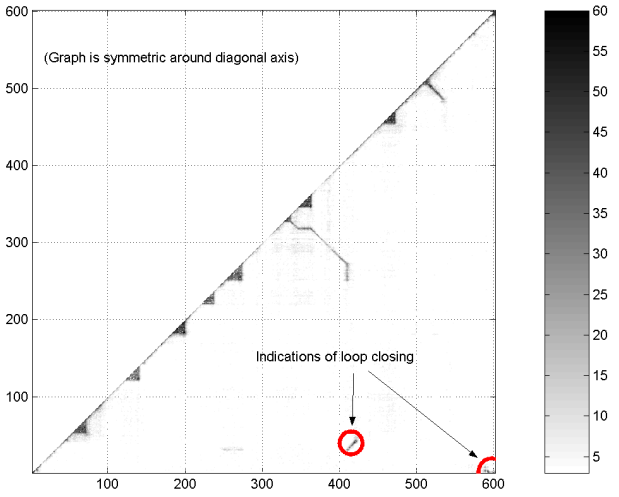

Fig. 3. Affinity matrix, illustrating the number of feature matches between images of a 602-image run.

IV. Batch version

The batch version of this algorithm uses a greedy search to find the nodes in the map. The search attempts to divide the entire image sequence into as few nodes as possible, or – equivalently – to create as large nodes as possible. The links are found by calculating the entire result matrix, i.e. performing an exhaustive search. Doing an exhaustive search is the only way to find all possible links, but there are other ways to do the search if we can accept some uncertainty in the answer. The batch version of the algorithm is seen as a benchmark for the incremental version.

V. Incremental version

In the incremental version, we add nodes and links to the map at each step, without having any knowledge of future data. A node is represented by a *node representative*. The node representative R is the image that is most similar to all other images within the node, with N_R matches:

$$N_R = \max_{i \in I}(\min_{j \in I, j \neq i} (C(i,j))) \tag{3}$$

$$R = \operatorname*{argmax}_{i \in I}(\min_{j \in I, j \neq i} (C(i,j))) \tag{4}$$

where $C(i,j)$ is the number of feature matches between images i and j, and I is the set of images within the node. For each new image S obtained, a decision function determines whether the image belongs to the current node or whether a new node should be created. If there are more than N_{min} matches between the obtained image and the node representative R, the image belongs to the current node and is added to the set I and a new node representative R is calculated. Otherwise, a new node is created. This technique guarantees that each image in the node always has at least N_{min} matches to the node representative.

In the simplest case, the calculation required in classifying a new image S is just one execution of the comparison function C.

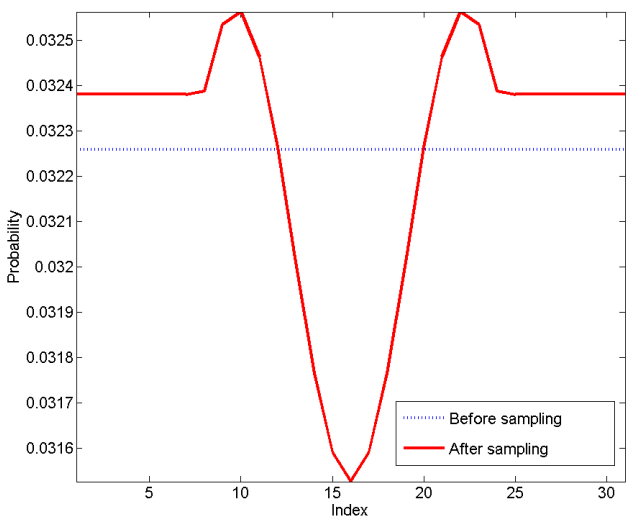

Fig. 4. Example probability density function used for selecting images, before (uniform distribution) and after sampling (original distribution multiplied by Mexican hat function), assuming a match was found at index 16.

This occurs when

- the number of matches between the new image and the node representative is higher than the number of matches recorded for the node representative; $C(S,R) > N_R$. In this case, the node representative does not change.
- the number of matches between the new image and the node representative is lower than N_{min}; $C(S,R) < N_{min}$. In this case, a new node is created.

In the case when $N_{min} \leq C(S,R) \leq N_R$, the number of calculations required depends on the size of I. However, it is always guaranteed that the number of executions of C is less than the size of I.

A. Searching the affinity matrix

The incremental version of the topological mapping algorithm calculates a fixed number n_C of feature matches $C(i,j)$ at each time step. The image indices (i,j) in the affinity matrix are selected by a search algorithm. The search is a random sampling algorithm, where the sampling distribution is updated by a heuristic function that increases the chance of investigating already established links (and improving our confidence in the calculated link likelihood). The first image index i is selected from a discrete distribution that is common for all images. The second image index j is selected from an image-specific distribution stored for image i. The initial sampling distributions are uniform. When the comparison has been performed, both the global distribution and the image-specific distributions are updated depending on the result. The heuristic function always lowers the sampling probabilities for images i and j (in order to avoid repeated evaluation). If the resulting comparison gave a value $C(i,j) < N_{min}$, the sampling probabilities for images i and j are multiplied by the function

Fig. 5. The robot in the environment used to obtain the images.

$$p_i(x,j) = \begin{cases} 1 - e^{-(\frac{x-j}{4})^2} & \text{if } |x-j| \leq 8; \\ 1 & \text{otherwise.} \end{cases} \quad (5)$$

where x denotes the (integer) image indices. If the comparison gave a sufficiently high value, $C(i,j) \geq N_{min}$, the sampling probabilities of nearby images around i and j are increased by multiplication of the distribution with a Mexican hat function

$$p_i(x,j) = \begin{cases} 1 - \left(\frac{sin(\frac{x-j+\pi}{2})+1}{2} - \frac{9}{50}\right) & \text{if } |x-j| \leq 8; \\ 1 & \text{otherwise.} \end{cases} \quad (6)$$

Figure 4 shows an example of the sampling distributions before and after a comparison in the case when $C(i,j) \geq N_{min}$.

There is a rationale behind the choice of these heuristic functions. While we wish to improve our confidence about an already existing strong link, we do not wish to "waste" an image comparison by investigating images that are really close to already compared images.

VI. Experiment

A. Experimental setup

The results presented below are based on five data sets (the largest containing 602 panoramic images) collected over a period of 2 months, using a teleoperated ActivMedia PeopleBot equipped with a RemoteReality Netvision 360 panoramic lens. The positions of the robot were estimated using data from a SICK LMS200 laser scanner, passing the laser scan and odometry data into a metric SLAM algorithm [15]. A new image was obtained every 0.5 m travelled or every 15° turned. The area covered in these experiments is an approximately 60 × 55 m indoor office area (Figure 5). 101 MSIFT features (or less, if there were not sufficiently many strong features) were extracted from each image ($N_{max} = 101$), and 15 features were required for a match between two images ($N_{min} = 15$).

B. Results

Figure 6a) shows the result from "ground truth" of the largest data set, i.e. the result from the batch version of the algorithm. Darker lines indicate links with high likelihood;

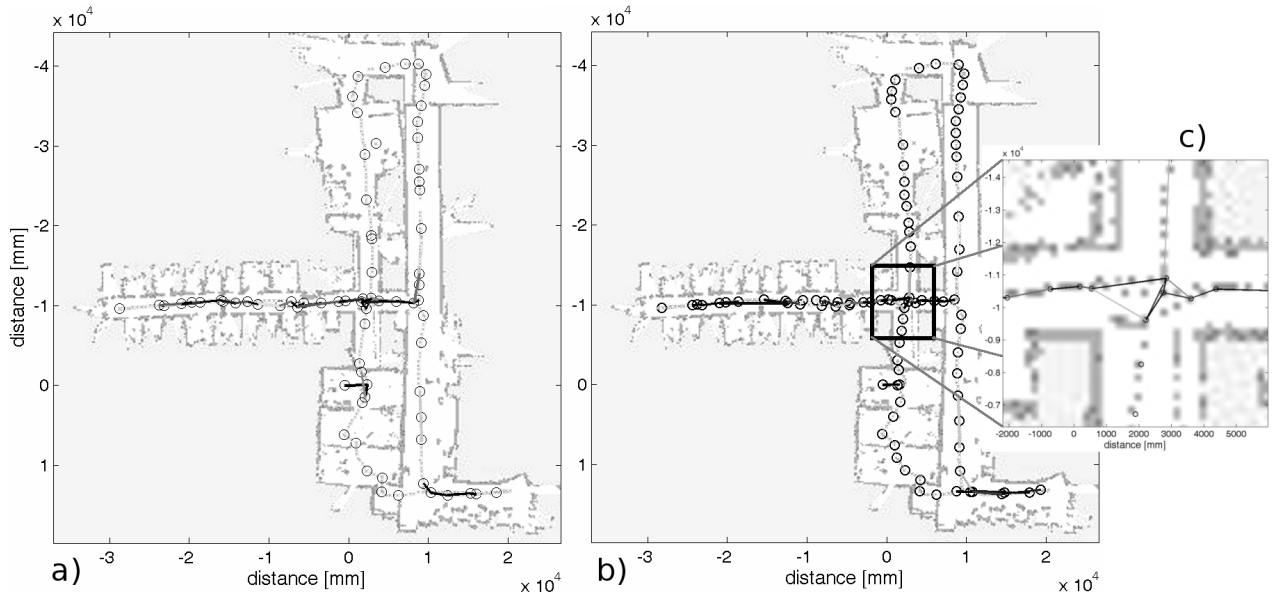

Fig. 6. a) Resulting topological map (batch version). Link likelihood is illustrated by darkness of line (darker - better). Weak links are not shown. b) Resulting topological map (incremental version). All entries in the affinity matrix were calculated. c) Zoomed in on links at crossing.

	Batch	Incremental (full result matrix)	Incremental (30 comp./iter.)	Incremental (60 comp./iter.)
Number of nodes	74	92	92	92
Number of links	41	51	13.6	21.4

Fig. 7. Results table. For the incremental version, the results were determined from 10 runs with random seeds.

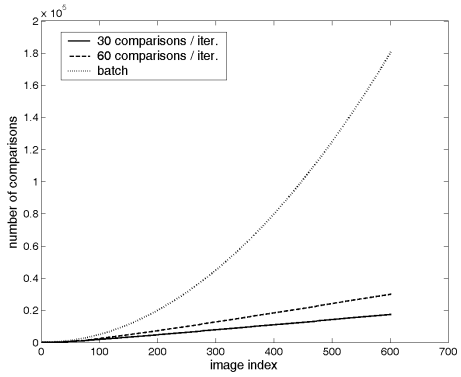

Fig. 9. Total number of comparisons vs. image index.

weak links are not shown. (The coordinates of the nodes, as well as the background map, are outputs of the laser-based SLAM-algorithm, and are used for visualization only.) Figures 6b) and 6c) show the result from the incremental version, but with the entire affinity matrix calculated. Although the number of nodes is higher in the incremental version (92 compared to 74), the maps show a qualitative resemblance. Figure 7 shows the result when the fixed number of comparisons per iteration n_C is set to 30 and 60, respectively.

C. Evaluation of the incremental algorithm

The total number of entries in the affinity matrix is approximately 1.8×10^5 for the data set with 602 images. For values of n_C of 30 and 60, we see that we calculate approximately one tenth or one fifth of the total number of comparisons (Figure 9). Still, this algorithm manages to find, on average, one quarter and two fifths respectively of the total number of links. This shows that it is possible to produce correct maps and yet avoid combinatorial explosion.

D. Additional data sets

The incremental algorithm was also applied to three smaller datasets, calculating the entire result matrix for all three maps (Figure 8). For all data sets, the loops were closed properly and there were no false links. Further, the incremental algorithm was applied to a large combined dataset, consisting of four smaller datasets obtained at different times with varying lighting conditions (Figure 10). Correct strong links are found, which shows that the algorithm has good potential for handling dynamic environments.

VII. Conclusion and Future Work

The approach suggested in this paper illustrates that it is possible to utilize vision only to achieve correct topological maps. It also shows that it is possible to calculate these maps incrementally, and thus *without* performing a full exhaustive search of the affinity matrix. Furthermore, the link likelihood can be viewed as a confidence measure, giving the map some probabilistic properties.

The algorithm has been tested with four different data sets acquired by a mobile robot in an indoor, complex environment, yielding good topological maps with loop closes, and avoiding false links completely. The approach does not eliminate the need for multi-hypothesis tracking; on the contrary, our approach is intended to be incorporated into such a framework

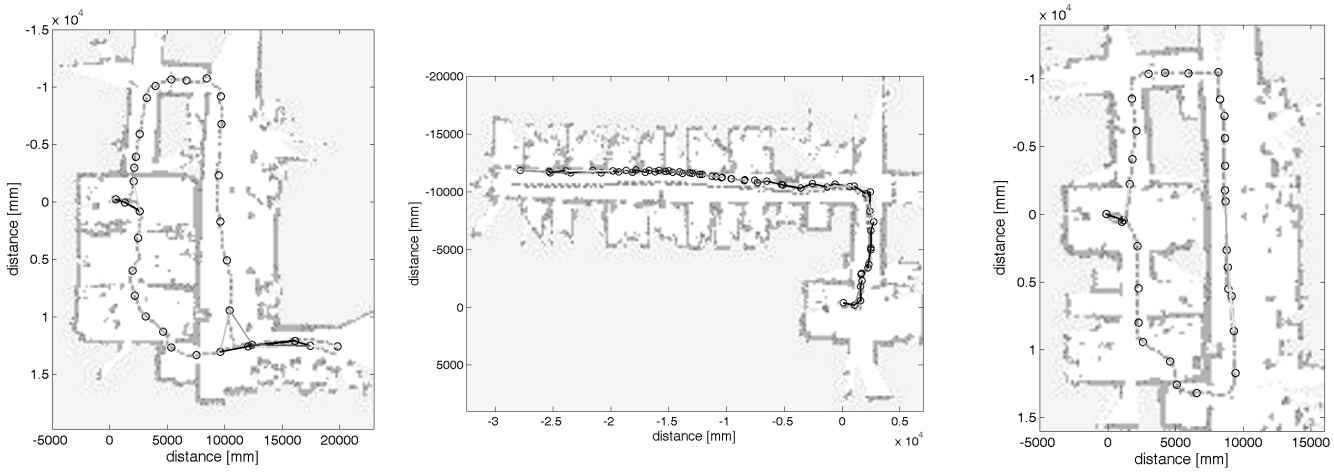

Fig. 8. Resulting map for three other runs with 316, 328 and 280 images (from left to right). The result was 31, 60, 30 nodes and 11, 63, 9 links respectively. All entries in the affinity matrices were calculated. Weak links are not shown.

(e.g. [16]). However, because the method has been shown to be quite insensitive to perceptual aliasing, the computational explosion associated with multi-hypothesis algorithms could possibly be mitigated to a great extent.

For this approach to work well, the environment cannot have unlimited size. If the enviroment is limited, and there is functionality for merging nodes, the size of the affinity matrix will eventually stabilize around a specific size. Over time, the number of accumulated comparisons will "catch up" with the size of the search space. An additional requirement is that the environment does not contain too many features that can be seen everywhere. An example of such a global feature is a distinct horizon feature outdoors that is visible from all locations within the data set. Any such feature needs to be removed; otherwise the resulting map may contain just a few very large nodes.

Future work will include:

- implementation of node split/merge functionality,
- taking geometric constraints (e.g. from odometry) into consideration.

The approach presented can potentially handle both large and unstructured environments. To be able to handle dynamic environments even better, we could track features from multiple images over time, and store only those features that are stable. The feature database could thus change over time, allowing robust relocalization in dynamic environments.

ACKNOWLEDGEMENT

This work is partially supported by The Swedish Defence Material Administration.

REFERENCES

[1] S. Thrun. Robotic mapping: A survey. Survey CMU-CS-02-111, School of Computer Science, February 2002.

[2] U. Nehmzow. "Meaning" through clustering by self-organisation of spatial and temporal information. In C. Nehaniv, editor, *Computation for Metaphors, Analogy and Agents*. Springer Verlag, 1999. Lecture Notes in Artificial Intelligence 1562.

[3] B. Kuipers. The spatial semantic hierarchy. *Artifical Intelligence*, 119:191–233, 2000.

[4] I. Ulrich and I. Nourbakhsh. Appearance-based place recognition for topological localization. In *Proc. IEEE Int. Conf. Robotics and Automation*, 2000.

[5] J. Gaspar, N. Winter, and J. Santos-Victor. Vision-based navigation and environmental representations with an omni-directional camera. *IEEE Trans. Robotics and Automation*, 16:890–898, 2000.

[6] J. Kosecka and F. Li. Vision based topological Markov localization. In *Proc. IEEE Int. Conf. Conference on Robotics and Automation*, 2004.

[7] H. Andreasson and T. Duckett. Topological localization for mobile robots using omni-directional vision and local features. In *Proc. IAV 2004, the 5th IFAC Symposium on Intelligent Autonomous Vehicles*, Lisbon, Portugal, 2004.

[8] Z. Zivkovic, B. Bakker, and B. Kröse. Hierarchical map building using visual landmarks and geometric constraints. In *Proc. IEEE Int. Conf. Intelligent Robots and Systems*, 2005.

[9] D.M. Bradley, R. Patel, N. Vandapel, and S.M. Thayer. Real-time image-based topological localization in large outdoor environments. In *Proc. IEEE Int. Conf. Intelligent Robotics and Systems*, August 2005.

[10] A. Tapus and R. Siegwart. Incremental robot mapping with fingerprints of places. In *Proc. IEEE Int. Conf. Intelligent Robots and Systems*, 2005.

[11] D.G. Lowe. Object recognition from local scale-invariant features. In *Proc. Int. Conf. Computer Vision ICCV*, 1999.

[12] J. Shi and C. Tomasi. Good features to track. In *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, 1994.

[13] H. Andreasson, A. Treptow, and T. Duckett. Localization for mobile robots using panoramic vision, local features and particle filter. In *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA 2005)*, Barcelona, Spain, 2005.

[14] F. Lu and E. Milios. Globally consistent range scan alignment for environment mapping. *Autonomous Robots*, 4:333–349, 1997.

[15] U. Frese, P. Larsson, and T. Duckett. A multilevel relaxation algorithm for simultaneous localisation and mapping. *IEEE Trans. on Robotics*, 21(2):196–207, April 2005.

[16] A. Ranganathan, E. Menegatti, and F. Dellaert. Bayesian inference in the space of topological maps. *IEEE Trans. on Robotics*, 22(1):92–107, February 2006.

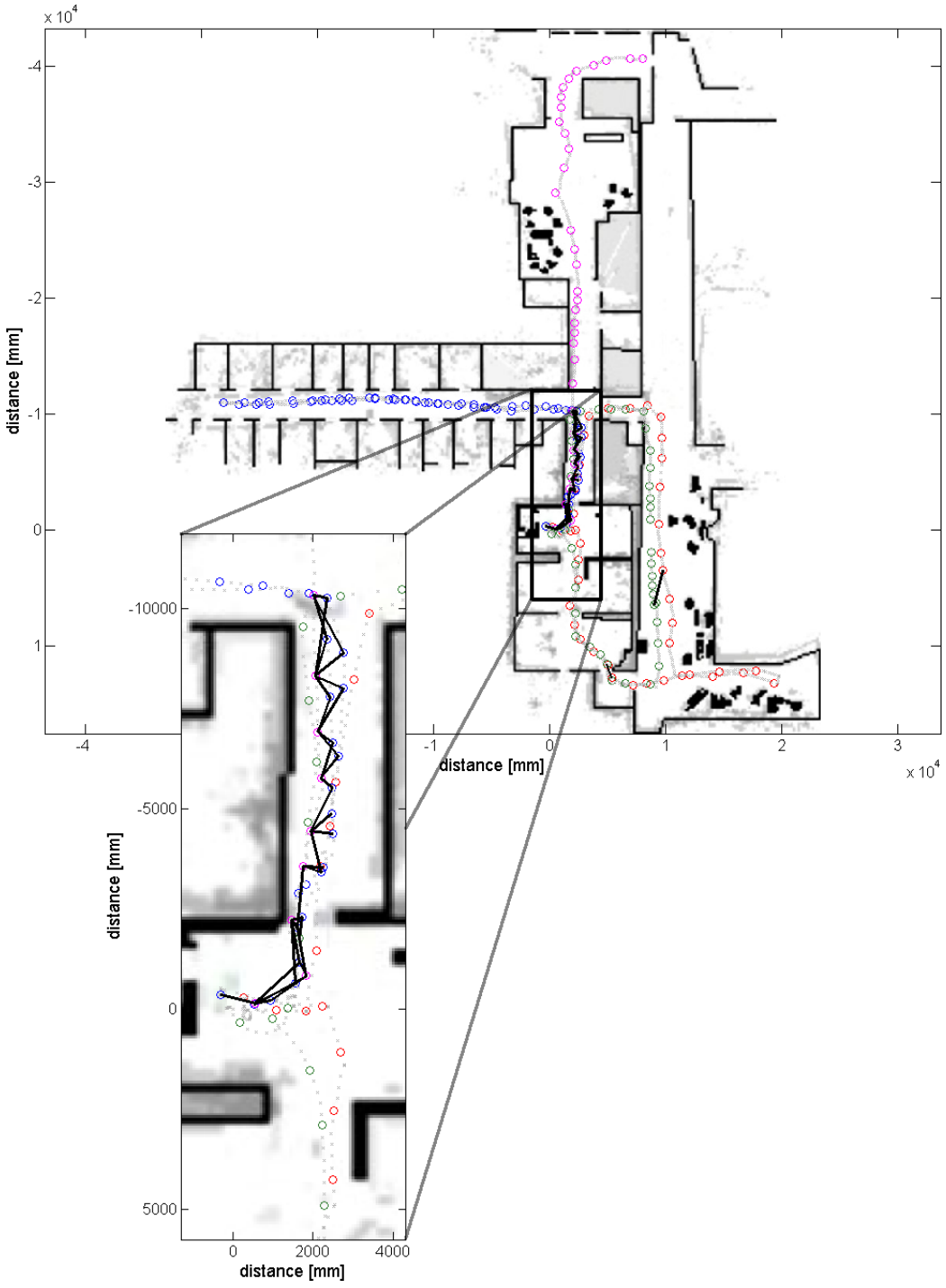

Fig. 10. Topological map calculated by combining four different data sets collected in a real populated environment on different days under different lighting conditions. Incremental algorithm, entire affinity matrix was calculated. The different SLAM-maps have been merged into a common map. Because of inconsistencies between runs, the image positions have been slightly adjusted for visualization purposes. The inset figure highlights the links found between the different datasets in one area of the environment where the respective submaps overlap. All strong links are shown in black.