Anytime merging of appearance based maps

Gorkem Erinc, Stefano Carpin

***Abstract*— Appearance based maps are emerging as an important class of spatial representations for mobile robots. In this paper we tackle the problem of merging together two or more appearance based maps independently built by robots operating in the same environment. Noticing the lack of well accepted metrics to measure the performance of map merging algorithms, we propose to use algebraic connectivity as a metric to assess the advantage gained by merging multiple maps. Next, based on this criterion, we propose an anytime algorithm aiming to quickly identify the more advantageous parts to merge. The system we proposed has been fully implemented and tested in indoor scenarios and shows that our algorithm achieves a convenient tradeoff between accuracy and speed.**

I. Introduction

The use of multiple cooperating robots is now mainstream, with applications found in many diverse areas. Multiple robots bring benefits like increased robustness through redundancy, decreased time requirements to solve problems, and spatial distributedness. Along with these advantages come also algorithmic challenges because of the necessity to extend single robot solutions to the multi-robot domain. Among these problems, we have in the past worked on the map merging problem, i.e. the problem of combining together various partial maps produced by numerous robots operating in the same environment [2], [4]. Other solutions were proposed in [11], [21], a sign of increasing interest in this area. Note that map merging is an incremental and online process, i.e. in general, maps are not necessarily merged together when the mission is over, but rather while they are still being built. For example, robots exchange and merge their maps when they are within communication range [14], but during the remaining part of their mission they operate independently and possibly continue to improve their spatial models individually. Past research dealing with map merging has mostly focused on metric or topological maps.

In this work we extend this line of research by considering the problem of merging appearance based maps. These spatial models are gaining importance because images offer a natural way to exchange information between robots and humans. Informally speaking, an appearance based map consists of a graph where every vertex is associated with an image, and edges are added between similar images. Figure 1 shows a simple example.

The problem of merging maps together goes hand in hand with the problem of evaluating the quality of a map. This challenge is far from being solved even for more traditional approaches (e.g. metric or topological), as witnessed by recent special issues and projects devoted to robot benchmarking [17]. In this paper we address these problems jointly, i.e. we propose a metric and introduce a merging algorithm that is then evaluated based on the metric. Our contributions are the following.

G. Erinc and S. Carpin are with the School of Engineering, University of California, Merced (USA). E-mail: {gerinc,scarpin}@ucmerced.edu

Fig. 1. A sample appearance map with edges inserted between sufficiently similar images is shown.

- We analyze the use of algebraic connectivity as a metric to determine the gain obtained by merging multiple maps together. Our findings corroborate that algebraic connectivity is a good metric for this purpose.
- We propose an *anytime* algorithm to merge two (or more) appearance based maps. The algorithm aims to quickly identify edges leading to large gains in terms of the metric outlined above. The reader should note that the problem of identifying edges leading to the largest gain is known to be NP-hard [18], and therefore one has to necessarily look for a suboptimal solution.
- The algorithm and the metric are experimentally validated by merging appearance based maps autonomously built by a mobile robot navigating indoor. Our results show that the merging technique offers a convenient tradeoff when compared with the brute-force algorithm eventually achieving the best possible merging within our framework. More precisely we observe a 10-90 ratio, i.e. by spending only 10% of the time it is possible to get about 90% of the benefit.

The remainder of the paper is organized as follows. In Section II we address related work. Next, in Section III the general structure of the framework used for appearance based maps is presented. Algebraic connectivity to measure the quality of merged maps is presented in Section IV, and the proposed map merging method is introduced in Section V. Finally, in Section VI, we present an experimental evaluation

of the algorithm and conclude the paper with final remarks and future work in Section VII.

II. Related Work

The problem of robot mapping and localization using visual sensors has generated enormous interest thanks to progress in sensor technologies and computer vision research. Solutions based on triclops camera systems [25], omnidirectional cameras [3], [5], and monocular cameras [9] have been proposed. Among them, systems based on monocular cameras provide a cheap solution by utilizing a ready to use tool to exchange information between users and robots. However, many of these systems embed metric information extracted either by means of odometry data or multiview geometry in case of multiple cameras. Fraundorfer et. al. [9], targeting systems where there is no odometry available or odometry is very difficult to estimate as in human motion, presented an image based localization and mapping solution. They build an *appearance based map*[1] using images from a monocular camera. Under this paradigm, mapping is reduced to identifying edges between similar images, whereas localization is defined as the problem of finding the image most similar to a query image. This setup takes advantage from state-of-art solutions to extensively studied information retrieval problems.

A common step in most content-based image retrieval algorithms is to discover pairs of matching images using local feature matches, and organize these results into a graph structure. Then, given a query image the graph is searched for the most similar one. The seminal paper by Sivic and Zisserman [26] demonstrates an efficient image search algorithm by using a Bag of Words (BoW) framework where images are categorized by the set of words that they contain and their frequencies. Similar to [20], the authors generate a dictionary of visual words offline by clustering descriptors from a training set. Due to its speed and accuracy, the BoW framework is extensively used for image based mapping problems. In [15] indoor localization based on the BoW idea is implemented, where each feature is treated as a single word in the dictionary. Kang et. al. [12] introduce an additional filtering level in which a small set of images returned by the standard voting process is re-evaluated and new similarity weights are learned from this small set, rather than from the whole database. An alternative approach is demonstrated in [1] which dynamically constructs the dictionary. While this dynamic approach allows images with features not represented in the training to be recognized, it can only accommodate a few thousand images for real-time performance. Hence, for systems with real-time performance requirement, offline training methods are adopted and their efficiency is shown [22] for very large (1M+) image collections.

Among these solutions for appearance based mapping and localization, only a few of them proposed ideas about how the system can be extended to accommodate multi robots and how maps built by different robots can be merged. Ho and Newman [10] propose a system in which an algorithm to identify matching subsequent images between two maps is implemented. They process the similarity matrix built by pairwise comparisons of all images and apply a modified version of Smith-Waterman algorithm [27] to find local alignments. The results of merging small maps have been shown. One disadvantage of the method is that merging procedure does not start until after the similarity matrix is built by pairwise comparisons of all images and local alignments are found. Hence, the algorithm's performance suffers as the number of images increases.

[1]From now on we will refer to image based maps in which no metric information is used as appearance based maps

III. Appearance based Maps

An appearance based map is an undirected weighted graph, $G = (V, E)$, in which each vertex[2], $v \in V$, represents an image. An edge, $e_{ij} \in E$, connects two distinct vertices v_i, v_j, whenever the associated images are sufficiently similar, according to a given similarity metric. A frequent way to define similarity between two images is to extract salient features from each image and count the number of common ones. The number of matching features is used as a measure of similarity and is then set as the weight w_{ij} of the edge e_{ij} between v_i and v_j.

Our BoW implementation uses a modified version of the image search engine by Sivic and Zisserman [26] due to its efficient matching process and scalability. A set of robust local image features (SIFT) characterizing the scene perceived is extracted from each image and quantized into visual words. Quantization is obtained by k-means clustering performed on the descriptors from a number of training images. The collection of these visual words is the dictionary for the BoW.

With the aim of testing our algorithms in indoor environments, we trained our system with over 20 million SIFT features extracted from random indoor images collected from online image repositories (e.g. [23], [16], [24]) and we experimentally verified that 50K words (clusters) perform the best for our purposes. Features from random images are used as the basis of our dictionary and the training dataset contains no images from the environment in which mapping and localization tests occur. Despite this, robots preloaded with this static dictionary perform mapping and localization tasks successfully. In addition to the list of words, the dictionary also holds an inverted index that stores the appearance of each word among all images. An appearance object contains the feature's x and y pixel coordinates in the image plane along with the pointers to the vertices the feature belongs to.

A. Localization

We will first describe the localization algorithm, which will be needed later on when solving the merging problem, assuming the map has already been built. Localization in appearance based maps is the problem of finding the most

[2]Throughout the paper, *vertex* and *image* will be used interchangeably.

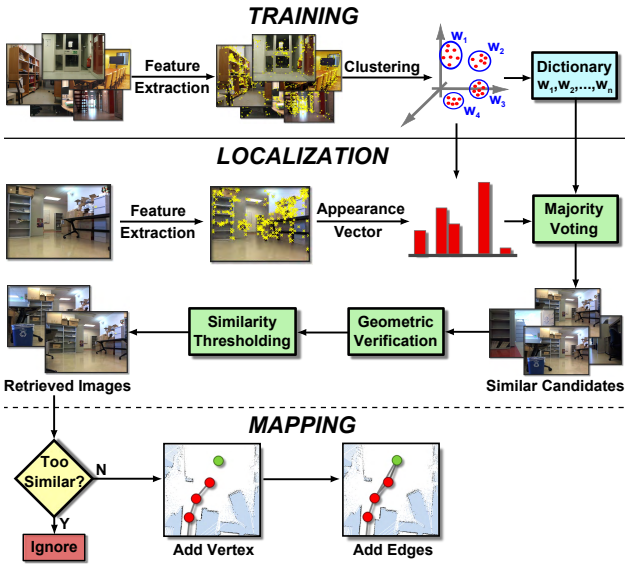

Fig. 2. Overview of dictionary learning, map building and localization procedures are presented.

similar image to a query image, I_q. Let F_q be the set of m features associated with I_q. First, each feature $f_i^q \in F_q$ is matched to the closest word with respect to L_2-norm in the dictionary. Since in our implementation SIFT features have 128 dimensions, finding the closest cluster center is a computationally expensive process. Kd-trees provide no speedup over exhaustive search for spaces with 10 or more dimensions for exact solutions. Thus, striving for real-time performance, we used the approximate nearest neighbor solution presented by Muja and Lowe [19].

With the assignment of all features to their closest words, we build a sparse *appearance vector*, v_q, which holds the appearances of each word in the query image. This vector is then used by a voting schema in order to find images with similar word frequencies. Voting works as follows. Each word appearance casts a vote for all images containing the same word. Then, similarity between the query image, I_q, and a target image, I_k, is given by

$$S_k^q = \sum_{\substack{0<i<m \\ 0<j<n}} \alpha(f_i^q, f_j^k)$$

where $F_k = \bigcup_{0<j<n} f_j^k$ denotes the extracted features from image I_k. α is the voting function defined as

$$\alpha(f_i, f_j) = \begin{cases} 1 & v_q(f_i) = v_k(f_j) \\ 0 & \text{otherwise} \end{cases}$$

where v denotes the appearance vector, and given a feature it returns the index of the associated word. Note that it has been shown that this approach outperforms other methods [6]. At the end of the voting process, the image I_m receiving the most votes is selected as the strongest candidate, where m is defined as

$$m = \underset{i}{\text{argmax}} S_i^q$$

In order to eliminate possible outliers, a robust estimation of the multi-view geometry that links these images is computed utilizing a RANSAC algorithm as described in [6]. Feature matches supporting the computed fundamental matrix are also tested for spatial consistency as described in [26]. Based on the idea that matching regions in compared images should have a similar spatial agreement, the algorithm eliminates matches not complying with the spatial layout of the neighboring matches in query and target images. If the number of remaining matches exceeds a threshold T_{min} the image is considered a match. We set this threshold to the minimum number of feature matches required to robustly navigate between two images using the navigation algorithm we presented in [6]. Figure 2 shows an overview of the method described thus far.

B. Mapping

Mapping, defined as the process of possibly adding a new image to an appearance based map, starts with localizing the new image within the existing map. The localization algorithm returns a set of similar images and their similarities in terms of number of common features. If the new image and the strongest candidate share a number of features that is larger than some threshold T_{max}, they are considered too similar and, with the idea that its insertion will not improve the quality of the map, the new image is ignored. Otherwise, a new vertex representing this image is inserted in the map and new edges between this vertex and other vertices symbolizing identified similar images are created. By rejecting the insertion of similar images into the map size of the map is prevented from increasing unboundedly in cases where the robot revisits the same area many many times. Figure 3 shows some snapshots of the process of mapping our robotics laboratory. The system we described can build appearance based maps in real-time. A non-optimized C++ implementation of mapping runs in real-time on our P3AT robot equipped with a 2GHz CPU. It takes around 0.6 seconds to process a single image of size 320×240, including image capturing, feature extraction, global localization, and map update. The framework also enables robust navigation without the need of any metric information. For the details on this navigation algorithm we refer the reader to [6].

IV. Quality Metric

In spite of their recent introduction, several appearance based mapping algorithms have been proposed, as presented in Section II. However, a formal quality metric for appearance based maps has not been adopted yet. To date, for most problems it is impossible to compare two different solutions according to a standard criterion, and it is therefore challenging to evaluate the real impact of different ideas proposed in this field. With this motivation, we proposed a set of task-based performance evaluation criteria in [7] to measure the quality of appearance based maps independently of the algorithm used to build them.

Measuring the performance of map merging algorithms, however, is different from the evaluation of a single appear-

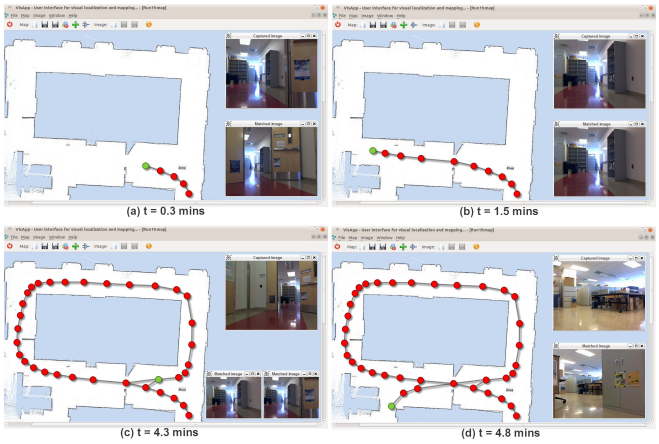

Fig. 3. The figure shows some snapshots of the GUI taken while the robot builds the appearance based map using the BoW method we described. Note that the occupancy grid map overlaid with images is shown for display purposes only, but neither the grid map nor image coordinates are used by the robot.

ance based map. The main attribute required for a merging algorithm to focus on is *entanglement*. Entanglement in this context can be defined as the amount of effort needed to split the merged map back into two maps. With this respect, the quality of the merging should be in positive correlation with the entanglement of the map. Edge connectivity is one of the commonly used metrics to measure the connectivity of graphs and is defined as the minimum number of edges which need to be removed to disconnect the graph. However, in a map merging scenario where vertices are preserved and interconnecting edges are added, this metric is inapprorpriate. For instance, for a connected graph with one vertex of degree 1, edge connectivity is computed as 1. Even tough inserting edges will improve the graph's overall connectivity, its edge connectivity will stay the same unless one of the inserted edges is connected to that specific vertex.

Based on this and similar observations, we maintain that to assess the quality of map merging in the appearance-based map domain the *algebraic connectivity* of the graph is a better measure. Introduced in the seminal work by Fiedler [8], algebraic connectivity is a spectral property of the graph widely used to measure robustness and connectivity. It carries more information about the structure of the graph and therefore, can be a more useful measure than edge connectivity. For instance, edge connectivity of all trees is equal to one, whereas the algebraic connectivity of a star is higher than that of a path.

Let $G = (V, E)$ be an undirected graph with n vertices. The Laplacian matrix $L(G)$ is defined as $L = D - A$ where A is the adjacency matrix and D is the $n \times n$ diagonal matrix of vertex degrees. The second smallest eigenvalue of L, $\lambda_2(L)$, is called *algebraic connectivity*. Various properties of algebraic connectivity have been outlined [8]. For example $\lambda_2(L) > 0$ if and only if G is connected. It is also known that algebraic connectivity defines a lower bound on both the vertex (k_V) and edge (k_E) connectivity. Additionally, algebraic connectivity is bounded from below by the minimum vertex degree and the diameter of the graph, and therefore edges that bolster connections between weakly connected vertices and connect distant vertices yield larger improvements in algebraic connectivity. More importantly, $\lambda_2(L)$ is a monotonically increasing function of the edge set, i.e. if $G_1 = (V, E_1)$ and $G_2 = (V, E_2)$ are such that $E_1 \subseteq E_2$, then $\lambda_2(L_1) \leq \lambda_2(L_2)$. Therefore, the more edges the merging algorithm inserts, the more connected the graph and the greater algebraic connectivity will be. Finally, algebraic connectivity is related to the sparsity of cuts in the graph. That is, a graph with large algebraic connectivity cannot have very sparse cuts. All these properties nicely relate to the characteristics that a quality measure for map merging algorithms should have. It therefore appears that algebraic connectivity is a promising criterion to evaluate the performance of merging algorithms and we embrace it for the remaining part of this paper.

V. Merging Appearance Based Maps

Let $m_1 = (V_1, E_1)$ and $m_2 = (V_2, E_2)$ be two appearance based maps possibly created by two robots exploring the same environment, and without loss of generality let $|V_1| \geq |V_2|$. Similarly, for simplicity we focus here on merging two maps together since merging of multiple maps can be achieved by merging them in pairs. Given m_1 and m_2, we would like to compute the merged map $m_{12} = (V_{12}, E_{12})$ where $V_{12} = V_1 \cup V_2$ and $E_{12} = E_1 \cup E_2 \cup E_{inter}$. At this point we are mainly interested in connecting these maps by inserting new interconnection edges, E_{inter}, rather than compressing maps by merging common vertices. The motivation behind this idea is that connectivity plays a key role for robots in order to utilize maps created by other robots. With no connectivity between its own map and the new one, a robot will not be able to take advantage from any vertex in the map produced by other robot(s). Therefore, in map merging the emphasis is in generating new edges between V_1 and V_2. Even though throughout the paper we are addressing the problem from a single robot's perspective, the proposed map merging method can be easily distributed so that the workload can be shared between two robots merging their maps.

The simplest solution is the brute-force approach where the largest map m_1 is kept fixed and vertices from m_2 are sequentially localized in it. Identified edges are then inserted after their geometric consistency is verified as explained in Section III-A. This algorithm can be considered optimal in the sense that it eventually identifies all edges that can be added between m_1 and m_2. However, its performance is completely unsatisfactory from the point of view of required time. Nevertheless, it provides a useful yardstick to evaluate the performance of the solution we propose.

Motivated by the observation that many of the vertices make marginal contributions to algebraic connectivity, while just a few yield large gains [13], we aim to reduce the time spent on validating and inserting non-critical edges. Therefore, we propose *QuickConnect*, an anytime appearance based map merging algorithm that adopts a more selective

edge insertion approach by trying to process first vertices giving large gains in algebraic connectivity. As mentioned in the beginning, the problem of identifying edges giving the largest increment is NP-hard, and therefore the method we propose is necessarily suboptimal.

The algorithm, whose pseudo-code shown on Algorithm 1, consists of two phases: exploration (Line 1-10) and refinement (Line 11-14). In the exploration phase the goal is to quickly search the map for the most similar images and add at most one edge per vertex. The idea is to create only the essential connections between most similar vertices, and postpone the validation of the remaining edges. Once all similar vertices have at most one edge inserted, the algorithm moves to the second stage and processes the postponed list of edges. This way, *QuickConnect* implements an anytime algorithm where most of the connectivity will be captured in the early stages of the merging process, and remaining connections are discovered only if the algorithm is left to run for extended periods of time.

Algorithm 1 QuickConnect(Map m_1, Map m_2)

```
1:  W ← InitializeQueue(m_2)
2:  S, L ← null
3:  repeat
4:      w ← W.pop_front()
5:      V ← getVotes(m_1,m_2,w)
6:      E ← update(S,V)
7:      if !E.empty() then
8:          L ← processEdges(E,W,L)
9:      end if
10: until W.empty()
11: L ← sort(L)
12: foreach e in L do
13:     insertEdge(e)
14: end for
```

The exploration phase (Algorithm 1 Line 3-10) is based on the incremental construction of the similarity matrix S, where S_{ij} gives the number of votes received by image pair $I_i \in m_1$ and $I_j \in m_2$. It starts with creating a priority queue of words, W, that contains words appeared at least once among the images from m_2 (Algorithm 1 Line 1). Initially, all words have the equal priority.

For each word in the list it computes the votes similarly to the voting schema used in the localization procedure described in Section III-A. Images in m_2 which contain this word cast a vote for all images in m_1 that also have this word in their appearance vectors (Algorithm 1 Line 5). The similarity matrix is updated with the addition of new votes (Algorithm 1 Line 6) and a list of candidate edges, E, is identified from the updated cells that reach the minimum similarity threshold T_{min}.

Insertion of edges occurs as they are incrementally revealed. As soon as a matrix entry S_{ij} identifies a new couple of vertices v_i, v_j with $v_j \in m_2$ not processed yet, the algorithm inserts a new edge after geometric verification and its source vertex is suspended from inserting any more

Algorithm 2 processEdges(List E, Queue W, List L)

```
1:  foreach e in E do
2:      if !isProcessed(e.source) then
3:          insertEdge(e)
4:          setProcessed(e.source, true)
5:          Q ← getWords(e.source)
6:          W.reorder(Q)
7:      else
8:          L.push_back(e)
9:      end if
10: end for
```

edges until the end of exploration phase (Algorithm 2 Line 3-4). All other identified edges sourcing from these suspended images are kept in a list, L, to be inserted in refinement stage (Algorithm 2 Line 8). This process is summarized in Algorithm 2.

Additionally, in order to improve the performance of exploration, we alter the direction of search using the identified similar images. The motivation is that a word seen by an image is highly likely visible by its adjacent images so by casting votes from this word we can identify new edges sourcing from these neighbor vertices. In order to implement this idea, once a similar image is found, we increase the priorities of the words it contains (Algorithm 2 Line 6) so that if they are not processed yet, they move to the top of the word priority queue and become the next in the list to be processed.

When all the words are processed and the similarity matrix is completely filled, the algorithm initiates the refinement stage (Algorithm 1 Line 11 - 14). Exploiting the fact that the minimum vertex degree provides a lower bound on algebraic connectivity, we aim to improve the minimum vertex degree (and then algebraic connectivity) early in the map merging process. Therefore, identified edge candidates are sorted with respect to the minimum degree of the two vertices it connects. Finally, edges are inserted sequentially starting from the ones connecting vertices with least number of adjacent vertices. Figure 4 shows the details of the map merging process implemented by the algorithm just described.

VI. Results

In this section we present a comparative evaluation of the proposed map merging algorithm. Brute-force merging sets the baseline for our comparisons because it eventually reaches the highest possible algebraic connectivity of the merged map. However, the trend to reach the maximum strongly depends on the sequence it follows in selecting couple of vertices to try to add edges between them. In fact, algebraic connectivity may quickly increase if the algorithm finds good vertices and edges early in the process, but it may also be the case that good attempts are made only later on in the process. Therefore, if the goal is to assess the performance of brute-force using an anytime approach, the algorithm should be tested on a large set of highly diverse maps to merge. On the other hand, having only a limited

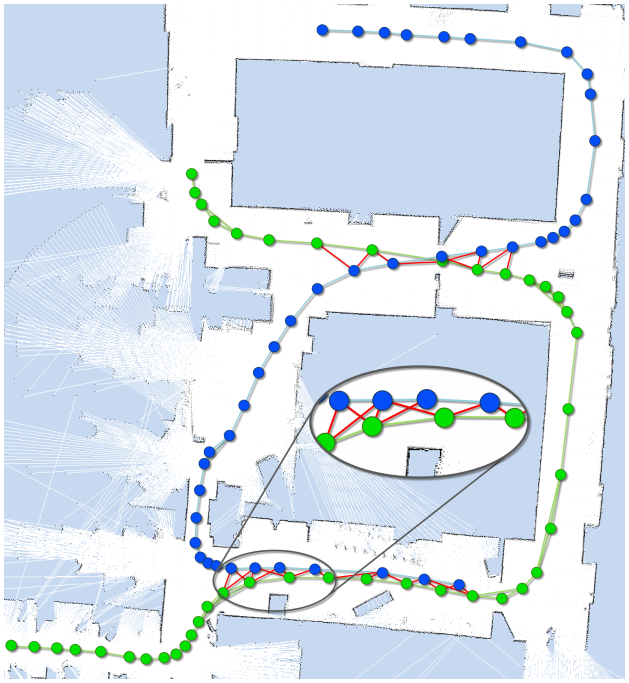

Fig. 4. Two independently built appearance based maps of our laboratory are shown as blue and green vertices. Similar images from both maps are identified and maps are connected through the edges, shown in red, between these images.

number of datasets available, we compare our algorithm against a randomized brute-force method that randomly selects couples of images from m_1 and m_2 using a uniform distribution, and we consider its average performance over repeated runs. As an additional term of comparison, we introduce *DegreeMin*, a variant of the randomized brute-force method that samples vertices from m_2 with a mass distribution inversely proportional to their degree. The rationale behind *DegreeMin* is trying to bias the search towards vertices with low degrees in order to possibly obtain large gains in algebraic connectivity.

In order to test these algorithms, various appearance based maps with a number of vertices ranging from a few hundreds to several thousands are built by our P3AT mobile robot equipped with a monocular camera using the map building algorithm described in Section III-B. Various combinations of these maps are merged using these algorithms and a representative sample of these results are shown in Figure 5. Here we present for different map pairs the quality of merging after 10% of the time required by the brute-force approach to complete merging. The results for the randomized algorithms (randomized brute-force and *DegreeMin*) are the average of 20 runs. As can be seen, *QuickSearch* outperforms the other two methods, reaching above 90% of the maximum possible algebraic connectivity within 10% of total time. It is also notable that *DegreeMin* with its heuristic that favors minimum degree vertices performs better than randomized brute-force.

The merging process of the map pair corresponding to the first column in Figure 5 is shown in Figure 6. Note that

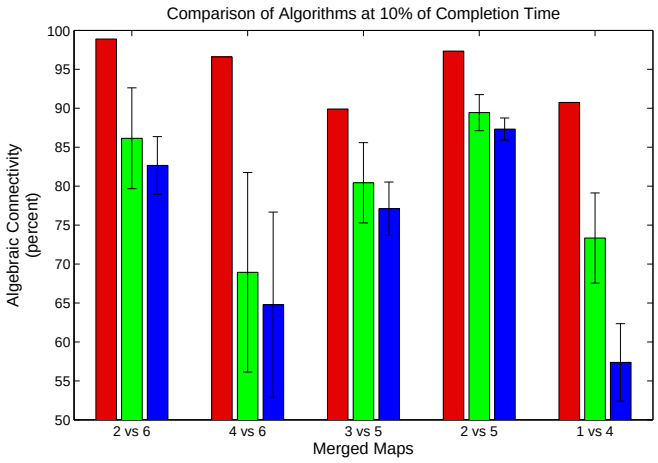

Fig. 5. Merging process is stopped at 10% of the time required for brute-force algorithm to merge maps and quality results for a representative selection of merged maps are presented for three algorithms: QuickConnect (red, left), DegreeMin (green, middle), and randomized brute-force (blue, right). Error bars representing one standard deviation is shown for randomized algorithms.

all algorithms eventually reach the same maximum algebraic connectivity by adding all possible edges. However, the exploration phase inserting only 247 edges in 1.5 seconds improves the quality up to 96% while it takes around 24 and 40 seconds and 2112 and 3989 edges to reach the same level of quality for *DegreeMin* and brute-force respectively. On the other hand, the refinement stage adds almost 15 times more edges than the exploration phase to only improve the connectivity by 4%. Hence, we can conclude that the exporation stage successfully identifies the small portion of edges that have substantial contribution to algebraic connectivity.

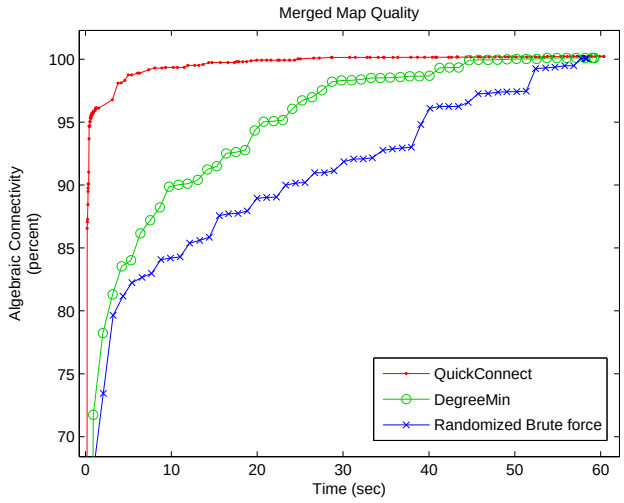

Fig. 6. Normalized algebraic connectivity as a function of time for the three algorithms presented. This chart refers to the process of merging a map, m_1, with 1356 vertices with m_2 with 610 vertices.

It is also important to note that in its exploration stage *QuickConnect* identifies the edges connecting the most similar images between two maps. This trend can be seen in Figure 7, where a closeup view of the aforementioned

merging process is shown. This characteristic of the proposed algorithm is indeed the main desired property of a compression algorithm that unifies similar vertices in both maps. Hence, this framework can easily be extended to merge maps by not only adding new edges but also unifying vertices. Nevertheless, the problem of merging images described as sets of features in appearance maps and the effects of such unification on the quality and robustness of the map in terms of algebraic connectivity warrants much further investigation.

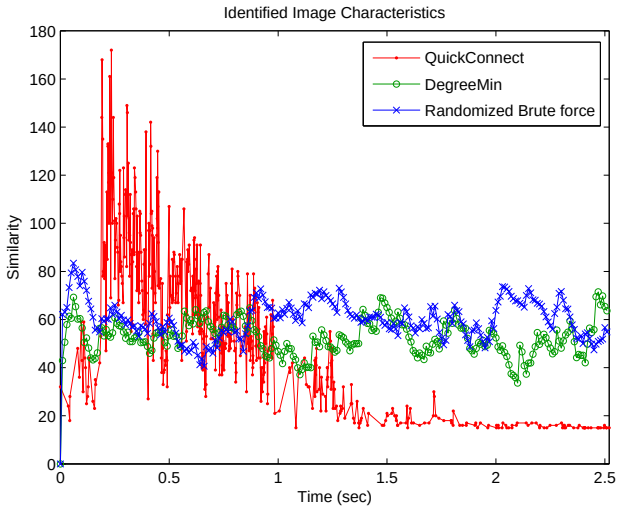

Fig. 7. Weights, i.e. similarity of connecting images, of the identified edges are plotted for three algorithms for the duration of the exploration phase of *QuickConnect*.

VII. Conclusion and Future Work

In this paper we have investigated the problem of merging appearance based maps, an underexplored topic that is very relevant in the area of multi-robot systems. Noting that there is a lack of well established criteria to evaluate the quality of a merging algorithm operating on appearance based maps, we have put forward algebraic connectivity and explained why it is a promising criterion to assess the value of a map merging algorithm. Starting from this criterion, we have developed *QuickConnect*, an anytime algorithm that discovers important vertices and edges early on during the process. The method we have proposed has been integrated into an end-to-end system that creates appearance based maps in real time using a bag of words approach. The goodness of the method we proposed has been experimentally assessed by processing various maps we produced with the aforementioned method, and we verified that even if stopped early during the process *QuickConnect* yields better results.

In the future, we plan to refine this line of research by considering merging algorithms that not only add novel edges between the maps being merged, but that also compress them by removing images that are represented in both maps.

References

[1] A. Angeli, D. Filliat, S. Doncieux, and J.-A. Meyer. Fast and incremental method for loop-closure detection using bags of visual words. *IEEE Transactions on Robotics*, 24(5):1027–1037, October 2008.

[2] A. Birk and S. Carpin. Merging occupancy grids from multiple robots. *Proceedings of the IEEE*, 94(7):1384–1397, 2006.

[3] O. Booij, B. Terwijn, Z. Zivkovic, and B. Krose. Navigation using an appearance based topological map. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, 2007.

[4] S. Carpin. Fast and accurate map merging for multi-robot systems. *Autonomous Robots*, 25(3):305–316, 2008.

[5] M. Cummins and P. Newman. Highly scalable appearance-only slam fab-map 2.0. In *Robotics Science and Systems (RSS)*, 2009.

[6] G. Erinc and S. Carpin. Image-based mapping and navigation with heterogeneous robots. In *Proceedings of the 2009 IEEE International Conference on Intelligent Robots and Systems*, pages 5807–5814, 2009.

[7] G. Erinc and S. Carpin. Evaluation criteria for appearance-based maps. In *Proceedings of PerMIS 2010*, 2010.

[8] M. Fiedler. Algebraic connectivity of graphs. *Czech. Math. J.*, 23 (98):298–305, 1973.

[9] F. Fraundorfer, C. Engels, and D. Nister. Topological mapping, localization and navigation using image collections. In *Proceedings of International Conference on Intelligent Robots and Systems (IROS)*, 2007.

[10] K. L. Ho and P. Newman. Multiple map intersection detection using visual appearance. In *International Conference on Computational Intelligence, Robotics and Autonomous Systems*, 2005.

[11] W.H. Huang and K.R. Beevers. Topological map merging. *International Journal of Robotics Research*, 24(8):601–613, 2005.

[12] H. Kang, A. A. Efros, M. Hebert, and T. Kanade. Image matching in large scale indoor environment. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR) Workshop on Egocentric Vision*, 2009.

[13] S. Kirkland. Algebraic connectivity for vertex-deleted subgraphs, and a notion of vertex centrality. *Discrete Mathematics*, 310 (4):911–921, 2010.

[14] K. Konolige, D. Fox, B. Limketkai, J. Ko, and B. Steward. Map merging for distributed robot navigation. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 212–217, 2003.

[15] Z. Lin, S. Kim, and I. S. Kweon. Recognition-based indoor topological navigation using robust invariant features. In *Proceedings of IEEE International Conference on Intelligent Robots and Systems(IROS)*, 2005.

[16] J. Luo, A. Pronobis, B. Caputo, and P. Jensfelt. The kth-idol2 database. Technical report, Kungliga Tekniska Hoegskolan, 2006.

[17] R. Madhavan, C. Scrapper, and A. Kleiner. Special issue on "characterizing mobile robot localization and mapping". *Autonomous Robots*, 2009.

[18] D. Mosk-Aoyama. Maximum algebraic connectivity augmentation is np-hard. *Operations Research Letters*, 36 (6):677–679, 2008.

[19] M. Muja and D. G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *International Conference on Computer Vision Theory and Applications (VISAPP)*, 2009.

[20] D. Nister and H. Stewenius. Scalable recognition with a vocabulary tree. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2006.

[21] N. Ergin Ozkucur and H. Levent Akin. Cooperative multi-robot map merging using fast-SLAM. In *Robocup 2009: Robot Soccer World Cup XIII*. Springer, 2010.

[22] J. Philbin and A. Zisserman. Object mining using a matching graph on very large image collections. In *ICVGIP*, 2008.

[23] A. Pronobis, B. Caputo, P. Jensfelt, and H. I. Christensen. A discriminative approach to robust visual place recognition. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2006.

[24] A. Quattoni and A.Torralba. Recognizing indoor scenes. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[25] S. Se, D. Lowe, and J. Little. Mobile robot localization and mapping with uncertainty using scale-invariantvisual landmarks. *International Journal of Robotics Research*, 21:735–758, 2002.

[26] J. Sivic and A. Zisserman. Video google: A text retrieval approach to object matching in videos. In *Proceedings of the International Conference on Computer Vision*, 2003.

[27] T. F. Smith and M. S. Waterman. Identification of common molecular subsequences. *Journal of Molecular Biology*, 147:195–197, 1981.