Noname manuscript No.
(will be inserted by the editor)

Anytime merging of appearance-based maps

Gorkem Erinc · Stefano Carpin

the date of receipt and acceptance should be inserted later

Abstract We consider the problem of merging together multiple appearance-based maps independently built by a team of robots jointly exploring an indoor environment. Due to the lack of accepted metrics to evaluate the quality of merged appearance-based maps, we propose to use algebraic connectivity for this purpose, and we discuss why this is an appropriate measure. Next, we introduce *QuickConnect*, an anytime algorithm aiming to maximize the given metric and we show how it can merge couple of maps, as well as multiple maps at the same time. The proposed algorithm has been implemented and tested on a fully functioning robotic system building appearance-based maps using a bag of words approach. QuickConnect outperforms alternative methods and features a convenient tradeoff between accuracy and speed.

1 Introduction

In this paper we consider the problem of merging multiple appearance-based maps independently created by teams of robots exploring the same unknown environment. Spatial awareness is one of the fundamental abilities for mobile robots, and the ability to fuse multiple maps into a single spatial model greatly enhances the utility of multi-robot systems. Numerous robotic tasks are solved more efficiently when a team of robots are used instead of a single robot. Robot teams are more robust to failures and can solve a task in less time. Moreover, teams of robots can physically operate in different areas and then collect spatially distributed information. The problem of combining together data produced by different sources then naturally emerges, and in this paper we specifically target the issue of merging spatial models (maps) independently computed by robots operating in a shared environment. After multiple maps have been combined together and the result has been shared, each of the robots that contributed one of the partial maps is then equipped with a more comprehensive spatial model.

Gorkem Erinc · Stefano Carpin
School of Engineering
University of California, Merced
5200 North Lake Rd.
Merced, CA 95343
E-mail: gerinc,scarpin@ucmerced.edu

The merged map can then enable robots to individually perform tasks they would otherwise not be able to accomplish, like for example safely navigating to a part of the environment they have not explored. In this paper we indeed see map merging as a process aiming to enable safe navigation into regions explored by other robots, but the concept has evidently a much broader applicability. For this reason, it is important for the merging process to be computationally efficient, so that robot can immediately exploit merged maps for navigation without having to wait for the completion of an off-line merging process.

The problem of autonomous map building has received enormous attention in the last two decades, and led to the development of diverse spatial models, like occupancy grids, feature-based maps, topological maps, and, more recently, appearance-based maps. In this work we focus on 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 of an appearance-based map.

Fig. 1: A simple appearance-based map with edges inserted between sufficiently similar images.

While map building has attracted plenty of research, map merging has emerged only recently and fewer results are available. Map merging is here defined as the process of combining multiple maps that have been independently built by different robots. This is different from cooperative mapping, where multiple robots concurrently and continuously contribute their data to a single map (see e.g., [55]). In our previous research we have proposed algorithms to combine multiple occupancy grid maps [6, 8, 9]. Other solutions for merging occupancy grid maps were proposed in [20, 21, 28]. The problem of combining topological maps has been addressed in [16] and [25].

Additionally, various methods have been proposed for merging feature-based maps [4, 5, 12, 30, 42]. However, to the best of our knowledge, the problem of combining multiple appearance-based maps has not been considered, and this paper presents the first solution to this challenge. Given that appearance-based maps are represented by graphs, our method aims to identify edges connecting vertices belonging to different maps. Stated differently, it aims to find pairs of similar images in disjoint maps, where the concept of similarity will be defined in the following.

A problem inherently related to map merging (and more in general to mapping) is the evaluation of results. Two maps can be merged in countless ways, so the problem of quantitatively measuring the quality of a merged map immediately arises. Unsurprisingly, this is an open problem and no well-accepted metric exists. In this manuscript we therefore put forward a criterion based on algebraic connectivity specifically developed for appearance-based maps. The contributions we offer in this paper are the following.

- We propose to use algebraic connectivity as a metric to measure the quality of multiple appearance-based maps merged together and we show how some of its well known properties capture desirable properties for merged maps.
- Starting from the proposed metric, we develop an anytime algorithm that quickly identifies edges leading to the largest gain in terms of algebraic connectivity. Given that the problem of determining edges leading to the largest increase in algebraic connectivity is known to be NP-hard [38], the algorithm we propose necessarily produces a suboptimal solution. The algorithm is anytime inasmuch as it iteratively discovers and adds new edges, and it outputs a partial, valid solution if stopped before completing its computation.
- We offer an end-to-end experimental validation of the algorithm we propose. The system we developed constructs appearance-based maps in real time and then merges them. Experimental findings reveal that our proposed method quickly determines the most convenient edges to add, and, according to an anytime paradigm, eventually determines all edges that could be added.

The remainder of the paper is organized as follows. We address related work in Section 2. Next, in Section 3 the framework used to build appearance-based maps is presented. Section 4 discusses why algebraic connectivity is a good measure for the quality of merged maps, and Section 5 illustrates the algorithm we used to build appearance-based maps in real-time. The proposed map merging method is introduced in Section 6 for the case when two maps have be combined. Section 6.2 discusses how our algorithm can be extended when multiple maps have to be combined. Finally, in Section 7, we present an experimental evaluation of the algorithm and conclude the paper with final remarks and future work in Section 8.

2 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 [48–50], omnidirectional cameras [7, 10, 54], and monocular cameras [11, 19, 46] have been proposed. Among them, systems based on monocular cameras provide an inexpensive solution using off-the-shelf sensors.

The paper by Kosnar et al. [29] focuses on monocular cameras and considers the problem of visual topological mapping for outdoor environments where color based

segmentation is used for path identification. Vertices in the graph are defined as cross sections, and vertex matching is realized by comparing the number of outgoing edges and their azimuths. With these assumptions this method is designed for very specific environments and is not suitable for generic scenarios. Similarly, Victor et al. [47] build topological maps only targeting corridor-like indoor environments. Since their visual navigation algorithm is based on vanishing points, the method is unsuited for environments with large open spaces. Many of the aforementioned systems embed metric information extracted either from odometry or multi-view geometry when multiple cameras are used. Fraundorfer et al. [19], targeting systems where there is no odometry available or odometry is very difficult to estimate as in human motion, present 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 [51] demonstrates an efficient image search algorithm by using a Bag of Words (BoW) framework where images are categorized by the set of words they contain and their frequencies. Similar to [41], 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.

Indoor localization based on the BoW idea is presented in [31], where each feature is treated as a single word in the dictionary. Focusing on the localization aspect, the paper does not provide any mapping algorithm. Similarly, in [36, 53] localization to one of the images acquired in the learning stage is implemented by comparing features from the current image to the features in the database using an approximate nearest neighbor search algorithm. Kang et al. [26] 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.

A hierarchical appearance-based topological mapping algorithm is proposed by Zivkovic et al. [7, 56–59]. The higher level topological map is constructed by using a graph partitioning method to cluster vertices in the appearance graph. However, the algorithm requires the number of clusters to be known in advance. Besides, vertices are permanently assigned to their clusters which limits the algorithm's adaptivity to the changes in the environment. An alternative approach is demonstrated in [1, 2, 18, 40] 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 requirements, offline training methods are adopted and their efficiency is shown [43] for very large (1M+) image collections.

Among these approaches for appearance-based mapping and localization, only a few discuss how the system could be extended to accommodate multi robots and how

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

maps built by different robots could be merged. One of the few systems implementing visual localization and mapping for multi-robot systems is presented by Hajjdiab and Laganiere [22]. Each robot is equipped with a monocular camera and starts from an unknown location and incrementally builds a local visual map of the environment with the ability to localize itself in the map. In the case of an overlap between any two robots, images from both maps are stitched together using inter-image homography and the resulting joint map is utilized by both robots. However, the resulting map is not suitable for our purposes because we rely on an epipolar based visual servoing approach for navigation [13, 37]. Another study published by Ferreira et al. [16] addresses the problem of merging image-based maps. In particular, they propose a two-step approach to find overlaps in view sequences and stitch them together into a generic topological map. Tentative overlaps are first discovered from the similarity matrix built by pairwise comparisons of all images, and identified overlaps are merged together after they are verified based on the idea that matched similar vertices should also have similar neighbors. One disadvantage of the method is that the 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. Ho and Newman [24] propose a system in which an algorithm to identify matching subsequent images between two maps is implemented. They also process the similarity matrix and apply a modified version of the Smith-Waterman algorithm [52] to find local alignments. The proposed method similarly does not scale well with respect to the number of images due to the dependency on the creation of a full similarity matrix.
This article builds upon and extends our former papers [13] and [15]. In [13] we only developed a strategy to navigate an appearance-based map using epipolar geometry, but we did not consider the map merging problem, as we focused on just one map. In [15] we presented an initial version of the merging algorithm limited to the case of pairwise matching and we illustrated its performance on a limited set of test cases. This manuscript presents results from more datasets, introduces an algorithm to efficiently merge together multiple maps (Section 6.2), and builds upon an improved version of the map building stage.

3 Appearance-based maps and map merging

In formerly published work there exist no unified definition of appearance-based map. The one we embrace in this paper is aligned with most literature but includes also some ad-hoc features aiming to enable autonomous navigation via image-based visual servoing. We define an appearance-based map as an undirected weighted graph $M = (V, E, w)$ in which every vertex[2] $v \in V$ represents an image acquired by a camera at a certain position in the workspace. The correspondence between images and vertices is one to one, i.e., every vertex is associated with one and only one image, and vice versa. An edge $e_{ij} \in E$ connects vertices $v_i, v_j \in V$ whenever the associated images are sufficiently similar according to a given similarity metric. Different metrics can be used and edges are therefore parametric with respect to the used metric. Let $S : V \times V \to \mathbb{R}$ the similarity function used. An edge is added between v_i and v_j whenever $S(v_i, v_j) > T_f$, where T_f is a metric dependent threshold. If $e_{ij} \in E$ we set $w(e_{ij}) = S(v_i, v_j)$,

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

i.e., the weight of an edge is given by the similarity between the vertices it connects. In Section 5 we describe the image similarity metric we use in our implementation.

The basic version of the map merging problem we consider in this paper is the following. Given two appearance-based maps $M_1 = (V_1, E_1, w_1)$ and $M_2 = (V_2, E_2, w_2)$, determine a merged map $M_m = (V_m, E_m, w_m)$ with $V_m = V_1 \cup V_2$, and $E_m = E_1 \cup E_2 \cup E_c$, where $E_c \subseteq V_1 \times V_2$ is the set of edges connecting images in V_1 with images in V_2. Evidently, w_m are the weights induced by the image similarity function S. We require that every edge in the merged map still satisfies the condition $S(v_i, v_j) > T_f$ introduced above. This is trivially true for edges coming from E_1 and E_2, but must also hold for edges in E_c discovered during the merging process. From our point of view the very purpose of map merging is to enable robots to navigate and reach areas discovered by other robots (for example, using the image-based methods described in [13, 37]). Therefore, our emphasis is on finding edges in E_c connecting the two maps and enabling navigation. From the definition it is evident that the merged map M_m retains all images originally included in M_1 and M_2 as well as their edges. The value of merging is given by the novel edges discovered in E_c that link the two graphs together. Given the definition of E_c and its constraints, it is immediately clear that one could evaluate image similarity for every couple of images in V_1, V_2 and add edges whenever their similarity exceeds T_f. This brute force approach will discover the largest possible set, E_c^*, and will be further discussed in a later section. However, brute force search is clearly time consuming and we are interested in faster alternatives that will output sets of edges that are subsets of E_c^*.

The problem statement considers only the case where two maps are merged, but the idea can naturally be extended when more than two maps need to be combined. In the following the discussion mainly focuses on pairwise merging, and the case of multiple maps merged together is presented in Section 6.2.

We conclude this section noting that the graph structure does not include any metric information. In the next section we then propose a method to measure merging quality that relies exclusively on graph properties.

4 Measuring the quality of merged maps

The definition of mapping metrics is an open question still drawing significant interest [35]. The answer remains elusive for almost all types of maps, including metric and topological models, and the research community has not yet settled on a widely accepted comparison method. With this motivation, we proposed a set of task-based performance evaluation criteria in [14] to measure the quality of appearance-based maps independently from the algorithm used to build them.

In this paper we are concerned with a slightly different problem, i.e., assessing the quality of merged maps. Referring to the problem stated in the previous section, for a given instance of the map merging problem two different algorithms may produce two sets of connecting edges, say $E_c' \subset E_c^*$ and $E_c'' \subset E_c^*$. We are therefore interested in a criterion to establish in a quantitative way which one is better. We maintain that the main attribute to be considered in appearance-based map merging is how much the merged maps are intertwined. In the following we refer to this property as *entanglement*, but this term does not refer to the graph entanglement property some times used in graph theory. Informally speaking, entanglement is the amount of effort

needed to split the merged map back into two separate maps. Given two solutions to the merging problem, we prefer the one with higher entanglement.

The concept of entanglement can be formalized in different ways. Edge connectivity $e(G)$ is a commonly used metric to measure the connectivity of graphs and is defined as the minimum number of edges to be removed to get two separate components. In a map merging scenario where vertices are preserved and interconnecting edges are added this metric offers poor results. For example, for a well connected graph with just a single vertex of degree 1, $e(G) = 1$. Even tough inserting edges will improve the graph's overall connectivity (and then utility in terms of navigation), its edge connectivity will not change until one of the inserted edges is connected to the vertex with degree 1. Based on this and similar observations, we maintain that *algebraic connectivity* is a better measure to assess the quality of map merging in the appearance-based domain. Introduced in the seminal work by Fiedler [17], algebraic connectivity is a spectral property of the graph widely used to measure robustness and connectivity. Algebraic connectivity carries more information about the structure of the graph and is a more useful measure than edge connectivity. In the following we therefore recall its formal definition and display some examples substantiating our preference over edge connectivity. We emphasize that our choice for this metric is based on some of its well known properties, and we are not aiming at unveiling new properties for this well studied mathematical concept.

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* and is usually indicated as $\alpha(G)$. Various properties of algebraic connectivity have been discovered [17]. For example $\alpha(G) > 0$ if and only if G is connected. It is also known that for non-complete graphs, algebraic connectivity defines a lower bound on both the vertex and edge connectivity. Additionally, algebraic connectivity is bounded from below by a monotonic function of the minimum degree $\delta(G)$ of the graph (see Section 6 for details). More importantly, $\alpha(G)$ 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 $\alpha(G_1) \leq \alpha(G_2)$. Therefore, the more edges the merging algorithm inserts, the more connected the graph, and the higher algebraic connectivity will be. Therefore, E_c^* is optimal when algebraic connectivity is the considered metric. 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. Figure 2 displays some examples of simple graphs and contrasts their algebraic connectivity with edge connectivity, a graph property that one could consider as an alternative to algebraic connectivity. These simple topologies show that edge connectivity is not very discriminative because very different topologies like chain, star, and tree all share the same value for $e(G)$ whereas algebraic connectivity provides a better distinction.

Figure 3 instead shows how algebraic connectivity varies when different edges are added between two star shaped graphs being merged. This simple example aims at showing how algebraic connectivity changes when edges are added between different vertices of the graphs being connected. Graphs associated with the appearance-based maps considered in this paper include thousands of vertices and edges and are evidently much more complex. However, similar effects in terms of variations of algebraic connectivity are observed when edges are added between vertices that are peripheral or more internal to the graph. These changes in algebraic connectivity justify the trends that will be later on observed in Figure 8 when discussing the experimental results.

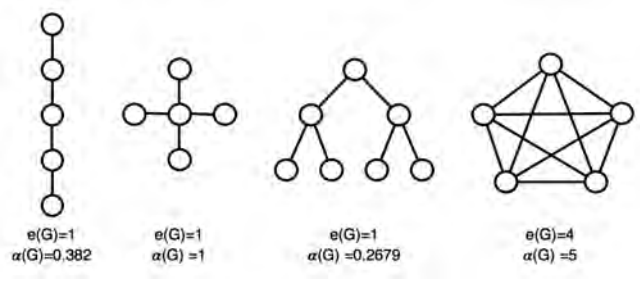


Fig. 2: The figure shows edge connectivity $e(G)$ and algebraic connectivity $\alpha(G)$ for different elementary graph topologies.

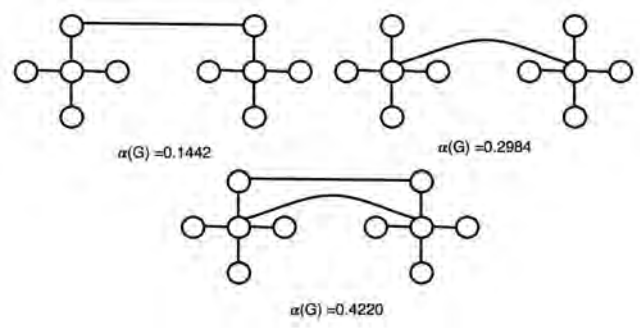


Fig. 3: Variations in algebraic connectivity when different edges are added between two star shaped graphs being merged.

It therefore appears that algebraic connectivity is a promising criterion to evaluate the performance of merging algorithms and we embrace it for the remaining of this paper.

5 Building appearance-based maps

In this section we describe the system we developed for real-time building of appearance-based maps. Our goal is not to create an algorithm to generate appearance-based maps, but to merge them. Therefore in this section we purposefully combine approaches formerly proposed, and we are aware that more sophisticated solutions could be implemented. However, map building is a pre-requisite for map merging, so for sake of completeness we here describe the system we use. The system has three parts, i.e., training, localization, and map update. Figure 4 shows its block diagram, and we describe it in the following.

5.1 Data structures and training

Our method is based on the BoW approach developed by Sivic and Zisserman [51] for searching similar images. The dictionary is built extracting SIFT features [32] from a

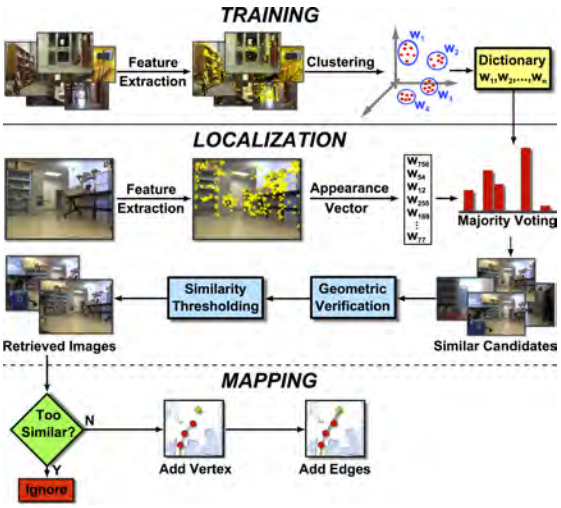


Fig. 4: Overview of how dictionary learning, map building, and localization interact.

set of training images and quantizing them using k-means clustering. The dictionary is built extracting more than 20 million SIFT features from random images obtained from online image repositories [3, 23, 33, 44, 45]. After various tests we determined that 50k clusters, also known as words, offer a good compromise between speed and accuracy, so we fixed the number of clusters to this value. This computationally expensive offline training procedure is only applied once and the resulting dictionary is preloaded to all robots performing appearance-based localization and mapping. Robots can then encode their visual perception with words from the same dictionary. This aspect plays an important role in the map merging algorithm, as described in the next section. It is also worth noting that no images from the environment used to test our algorithm were included in the training set. With the idea that words that are too common among the training images will not help differentiating query images, a stop list with the top 5% of the most common words is created. The dictionary includes also an *inverted index* that is incrementally updated when new images are added to the map. The inverted index associates every word to the list of images in which the word is seen, together with the x, y coordinates in the image plane.

5.2 Localization

Given an appearance-based map $M = (V, E, w)$ and a query image I_q, we want to find the most similar image in V. This step is needed for both map building and map merging. The algorithm should either return the most similar image, or determine that no sufficiently similar image exists in the map. Localization proceeds as follows. Starting from I_q we extract the set of SIFT features F_q. For every feature $f_i^q \in F_q$, the closest word in the dictionary is determined. Matching is performed using the L_2 norm, and given that we opted for a SIFT implementation with 128 features we use

an approximate[3] nearest neighbor solution [39]. After every feature has been matched to the closest word, an *appearance vector* v_q is built. The appearance vector v_q records every word in the dictionary matched to the features extracted from I_q. Then, by using v_q and the inverted index each word that is not in the stop list and appeared in the query image casts a vote for all images in the map that also contain this word. When normalized, the histogram of accumulated votes defines a probability distribution over all images in the map. As a result of this voting procedure, all images sharing one or more features with I_q are identified as candidate matches. The number of votes indicating the number of common features defines the formerly introduced similarity metric S . Among all candidate images, we discard those with less than a fixed number T_{min} of matches. We set this threshold to 15, i.e., the minimum number of matches required to robustly navigate between two images using the navigation algorithm we presented in [13]. Note that, for what concerns navigation, we have experimentally determined that this majority voting based approach outperforms other methods [13].

It is worth noting that T_{min} not only plays a role in defining when an edge should be added between two vertices, but it will also influence the behavior of the merging algorithm, as explained in section 6.1. However, according to our experience, the choice of its value should be mainly driven by the requirement of providing enough features to ensure robust navigation between two images using a visual servoing algorithm, like [13] or [37]. The T_{min} value we outlined above reflects the characteristics of our test environment, of the cameras we used, and of the robotic platform used to navigate in the environment. Evidently, in a different scenario the user should adjust the value based on preliminary experiments aimed at identifying the sensitivity to T_{min}.

The number of feature matches computed through this voting scheme could still be affected by outliers due to quantization effects in word clustering and to the approximation in finding the nearest neighbor. Therefore, as final step a robust estimation of the multi-view geometry that links these images is performed by utilizing the RANSAC algorithm presented in [34]. Feature matches supporting the computed fundamental matrix are also tested for spatial consistency as described in [51]. Based on the idea that matching regions in compared images should have a similar spatial agreement, the algorithm eliminates feature matches not complying with the spatial layout of the neighboring matches in the query and target images. If the number of remaining feature matches still exceeds T_{min}, the image is considered a match. Among all candidate images that pass the geometric verification test, the image with the largest number of matches is chosen to be the most similar and considered as the most likely location in the map. If not enough matches are left, the algorithm terminates indicating no similar image was found.

5.3 Mapping

Mapping is the process of iteratively acquiring new images and updating the appearance-based map, if needed. Given a new image I_n, the process starts by running the formerly described localization algorithm. Localization returns either a set of similar images, or an empty set. If no image is returned, then I_n is inserted into the map because the

[3] Kd-trees provide no speedup over exhaustive search for spaces with 10 or more dimensions for exact solutions, therefore striving for real time performance we decided for an approximate solution.

robot has discovered a new location that has no connection with any previous image. If localization returns a set of similar images, then image similarity is used to determine whether I_n should be inserted. If I_s is one of the returned images and $S(I_n, I_s) > T_{max}$, then I_s is considered too similar to I_n and it will not be inserted because it is not sufficiently informative. By rejecting the insertion of similar images into the map, we prevent the map from growing too much (in terms of images) when the robot revisits the same area numerous times. Otherwise, the image is added to the map and edges are added between the new vertex and all similar images identified by the localization algorithm. If the robot maps an environment with highly repetitive visual patterns, it is possible that in this stage edges will be added between images that are similar but far apart. This aliasing problem affects also the merging process described later on and is inherent to any algorithm establishing similarity relying only on visual information. However its effects are mitigated by discarding images with more than T_{max} matches. In fact, in the results presented in this paper this problem hardly occurred when the robot mapped one of the buildings in our university.

Figure 5 shows some snapshots of the process we described. The system can build appearance-based maps in real-time. A non-optimized C++ implementation[4] of mapping runs in real-time on a 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 [13].

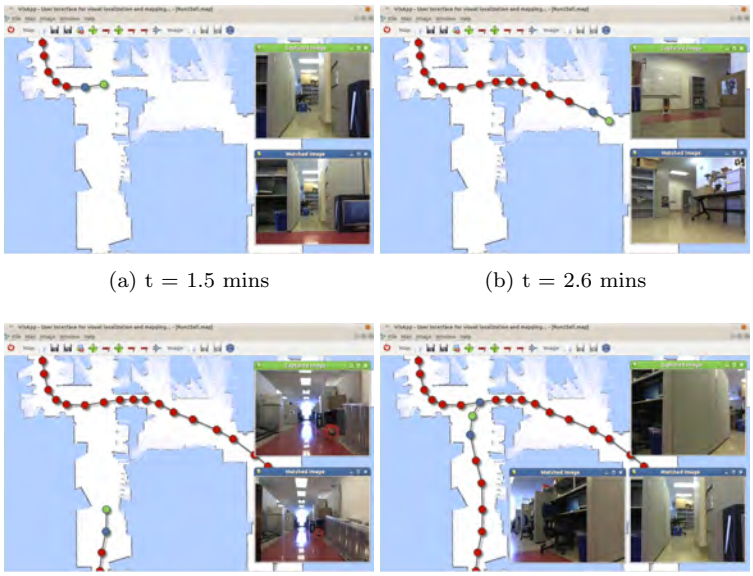
6 Merging Appearance-based Maps

6.1 Merging Two Maps

Let $M_1 = (V_1, E_1, w_1)$ and $M_2 = (V_2, E_2, w_2)$ be two appearance-based maps independently created by two robots running the algorithm described in Section 5. Without loss of generality let $|V_1| \geq |V_2|$. As explained in Section 3, our focus is on determining the set E_c, i.e., edges connecting vertices in V_1 with vertices in V_2. The motivation behind this idea is that these connections are needed to take advantage from maps created by other robots. In particular, with no connectivity between its own map and the new one, a robot will not be able to use image-based visual servoing to reach locations discovered by the other robot. In the following we present a centralized solution, i.e. we assume all computation is performed by one robot, but the proposed method can be easily distributed to share the workload.

To put our algorithm in perspective, it is useful to first consider a brute-force solution. Edges in E_c can be determined by brute-force by repeatedly calling the localization algorithm to localize every vertex in $v \in V_2$ inside map M_1. Localization returns the set of images whose similarity with v exceeds the given threshold T_{min}. This approach therefore identifies all edges that can be added between M_1 and M_2. Since algebraic connectivity is a monotonic function of the set of edges, this algorithm is optimal with respect to the algebraic connectivity metric formerly introduced. However, its performance is evidently unsatisfactory from the point of view of required time. Nevertheless, it provides a useful yardstick to evaluate the performance of the solution we propose.

[4] All code and dataset is freely available on our website.



(a) t = 1.5 mins (b) t = 2.6 mins

(c) t = 8.3 mins (d) t = 9.2 mins

Fig. 5: The figure shows some snapshots taken while the robot builds an appearance-based map using the BoW method. The last captured image is displayed at the top right corner of the GUI, while matched images are shown at the bottom right corner. Vertices corresponding to query and matched images are shown in green and blue, respectively. Note that the occupancy grid map overlaid with images is shown for display purposes only and not used by the robot.

A fundamental aspect to consider is that algebraic connectivity will be maximized only after adding all possible edges in E_c. Therefore, if the goal is to optimize this performance metric there is not much one can do to outperform the brute force approach. But a key observation to develop a more efficient method is that most edges make only marginal contributions to algebraic connectivity, while just a few yield large gains [27]. Our goal is therefore to develop an *anytime* algorithm that quickly determines a subset of E_c yielding a large increase in algebraic connectivity and eventually reaching the optimal value returned by the brute-force method. The algorithm *QuickConnect* that we describe in the following aims to test and insert edges yielding large gains early in the process, and to postpone performing computationally expensive multi-view geometry and spatial consistency tests for edges giving only marginal increments. The algorithm is therefore *anytime* in the sense that even when it is stopped before it has processed all its input, it still produces a valid solution. Moreover, if it is allowed to run to completion, it determines the optimal solution and the quality of the solution it produces is a monotonic function of the time spent. The problem of identifying edges yielding the largest increase in algebraic connectivity is NP-hard [38] and the method we propose is therefore necessarily suboptimal.

Algorithm 1 sketches the pseudocode for QuickConnect. The algorithm consists of two phases, i.e., *exploration* (Line 1-10) and *refinement* (Line 11-13). Exploration works towards quickly identifying the most similar images from both maps and add at

most one edge per vertex. The goal is to create only the essential connections between most similar vertices, and postpone the validation and insertion of the remaining edges. After all similar vertices have at most one edge inserted, refinement starts and all edges not processed during exploration are considered. According to this idea, QuickConnect will eventually discover all possible connections, but ideally the most relevant ones will be identified early on during the exploration process.

Algorithm 1 QuickConnect($M_1 = (V_1, E_1, w_1)$,$M_2 = (V_2, E_2, w_2)$)

```
1: W ← InitializeQueue(M2)
2: R, L ← null
3: Ec ← ∅
4: repeat
5:     d ← W.pop_front()
6:     P ← getVotes(M1,M2,d)
7:     E ← update(R,P)
8:     if !E.empty() then
9:         L ← processEdges(E,W,L,Ec)
10: until W.empty()
11: L ← sort(L)
12: for all e in L do
13:     insertEdge(e,Ec)
14: return Ec
```

Exploration is based on the incremental construction of a similarity matrix R where r_{ij} stores a score (vote) between image $v_i \in V_1$ and $v_j \in V_2$. The score is the number of common features between v_i and v_j and R is incrementally built as follows. A priority queue W is initialized with all dictionary words appearing at least once among images in map M_2 (Algorithm 1, Line 1). All words are initially assigned the same priority. Then, all words in W are dequeued. For every word d a vote vector P is computed. 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 (Line 6), and the similarity matrix R is then updated with the integration of new votes (Line 7). While updating R a list of candidate edges E is also created. A candidate edge between v_i and v_j is added to E as soon as r_{ij} exceeds T_{min}. Every candidate edge is then processed by the algorithm processEdges described in Algorithm 2.

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

```
1: for all e in E do
2:     if !isProcessed(e.source) OR !isProcessed(e.target) then
3:         insertEdge(e,Ec)
4:         setProcessed(e.source, true)
5:         setProcessed(e.target, true)
6:         Q1 ← getWords(e.source)
7:         Q2 ← getWords(e.target)
8:         W.reorder(Q1 ∪ Q2)
9:     else
10:        L.push_back(e)
11: return L
```

Algorithm processEdges inserts a new edge in E_c only if it originates from a vertex in M_2 from which no new edge has been discovered yet (Algorithm 2, Line 2). If the edge is added, then its source vertex is suspended from inserting any more edges until the end of exploration phase (Line 4). All edges not inserted because connected to an already processed vertex are inserted in a waiting list L that will be processed during the refinement stage (Line 10). In order to improve the performance of exploration, we also alter the priority of queue W based on the following *locality* observation. A word seen by an image is likely to be visible by its adjacent images. Therefore, by casting votes from this word we can identify new edges originating from these neighbor vertices. To implement this idea, once a similar image is found we increase the priorities of the words it contains (Line 8) 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.

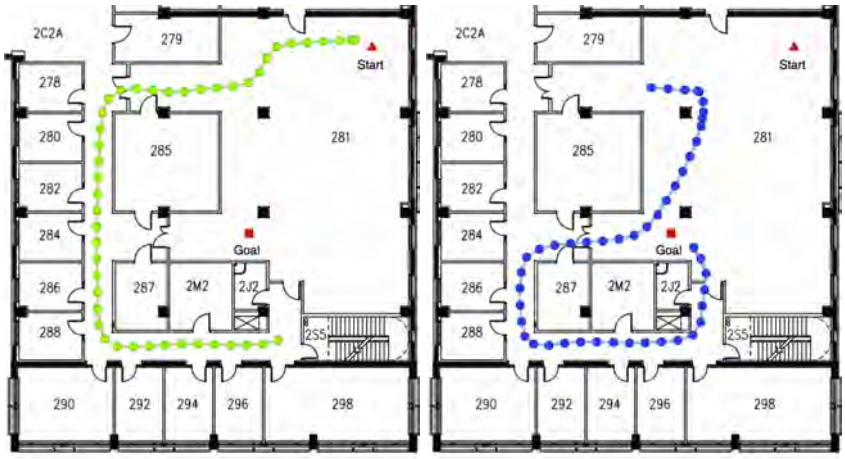
When all the words are processed and the similarity matrix is completely filled, the refinement stage starts (Algorithm 1 Line 11 - 13). Exploiting the fact that the minimum vertex degree provides a lower bound on algebraic connectivity[5], we aim to improve the minimum vertex degree (and then algebraic 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 6 shows an example of result for the map merging process implemented by the algorithm just described. Figures 6a and 6b display two maps independently built by two robots exploring different parts of the Science and Engineering building at UC Merced. Note that vertices of the graph have been correctly placed in the blueprint of the building because the robots used for this experiment are also equipped with a laser range finder, and we run a localization algorithm together with the known map to accurately determine where pictures are taken. This information is however used only for display purposes and for performance analysis, but is not available to the merging algorithm. Figure 6c shows the resulting merged map. In this case too, the merged graph M_m is overlaid to the blueprint to evidence the coverage of the merged map, but this is just for display purposes. The reader will notice that if the maps are used for navigation via visual servoing the merged map is much more useful than the individual ones because thanks to its discovered edges (see detail in the bottom one) a robot will be able to go from the start to the goal location, whereas none of the individual maps would allow that.

6.2 Merging Multiple Maps

In the previous section we addressed the problem of merging two maps. The presented method can be easily extended to solve the problem of merging N appearance-based maps. The map merging problem description presented in Section 3 can be generalized as follows. Given a set of N appearance maps $M = \{M_1, ..., M_N\}$ where $M_i = (V_i, E_i, w_i)$, compute a merged map $M_m = (V_m, E_m, w_m)$ with $V_m = \bigcup V_i$, and $E_m = \bigcup E_i \cup E_c$, where $E_c = \bigcup E_c^{ij}$ with $1 \leq i, j \leq N$, and $E_c^{ij} \subseteq V_i \times V_j$ is the set of edges connecting images in V_i with images in V_j. Note that, as in the previous de-

[5] It is known that $2\delta(G) - n + 2 \leq \alpha(G)$, where n is the number of vertices [17].



(a) First map

(b) Second map

(c) Merged maps

Fig. 6: The top two panels show two appearance-based maps independently built by two robots exploring different parts of the same environment. The last figure shows the merged map and the detail shows some of the edges (red) discovered during the merging process and linking the two maps together.

scription, the merged map includes all vertices from all maps and we focus on creating new edges between images from different maps.

One approach to merging multiple maps is to merge two of them, merge the resulting map with a third, and so on, until all input maps are merged together. This *sequential pairwise merging* method merges a pair of maps at a time and creates in-between merged maps after each step. However, the intermediate solutions provided by this approach do not capture the connectivity of all maps until the very last step. In other words, during all but the last step there is at least one map not involved in the merging process, and because of this map the number of connected components

within M_m is greater than one. Hence, the algorithm is not strictly anytime because algebraic connectivity remains zero until merging with the last map starts.

To overcome this limitation, and restore the desirable anytime feature, we introduce *QuickConnect-simultaneous*, an efficient parallel merging algorithm that overcomes the disadvantages of sequential merging approach. In the exploration phase, the algorithm incrementally constructs N_R similarity matrices in parallel, where $N_R = N(N-1)$, i.e., one similarity matrix for each pair of maps. For each word being processed, the algorithm first identifies images that contain this word from the first pair of maps and casts votes into their similarity matrix as described in Algorithm 1. Then, the same procedure is repeated for the remaining map pairs and the corresponding similarity matrices are filled with the votes for the same word. When all map pairs cast their votes into their similarity matrices, this word is labeled as processed and the next one in the priority queue is selected. A candidate edge is identified whenever the score of an image pair exceeds T_{min} in any of the similarity matrices, and the edge is inserted according to the processEdges routine described in Algorithm 2. Thanks to this parallel voting schema, from the very beginning edges between all map pairs are created simultaneously. All the remaining identified but not inserted edges from all map pairs are sorted by their vertex degree in increasing order and inserted during the refinement phase starting with the one with the minimum degree.

7 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 couples of vertices to try adding edges between them. 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 from an anytime standpoint, the algorithm should be tested on a large set of highly diverse maps. On the other hand, having only a limited number of datasets available, we compare our algorithm against a randomized brute-force method, *BruteUniform*, 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, as outlined in section 6.1.

To test these algorithms, various appearance-based maps with a number of vertices ranging from a few hundreds to several thousands are built by a P3AT mobile robot equipped with a monocular camera using the map building algorithm described in Section 5. Various combinations of these maps are merged using these algorithms and a representative sample of these results are shown in Figure 7. The first 5 columns correspond to map pairs built in one side of the engineering building at UC Merced (research labs), whereas the remaining columns correspond to maps built on a different side of the building featuring a significantly different layout (administrative wing). The

goal is to evaluate the algorithm on realistic and heterogeneous datasets. For different map pairs we present the quality of merging (i.e., algebraic connectivity) after 10% of the time required by the brute-force approach to complete its processing. The rationale behind this evaluation criterion is to embrace an anytime viewpoint and assess how different algorithms will perform when stopped early. The results for the randomized algorithms (randomized brute-force and *DegreeMin*) are the average of 20 runs. The chart shows that *QuickConnect* 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.

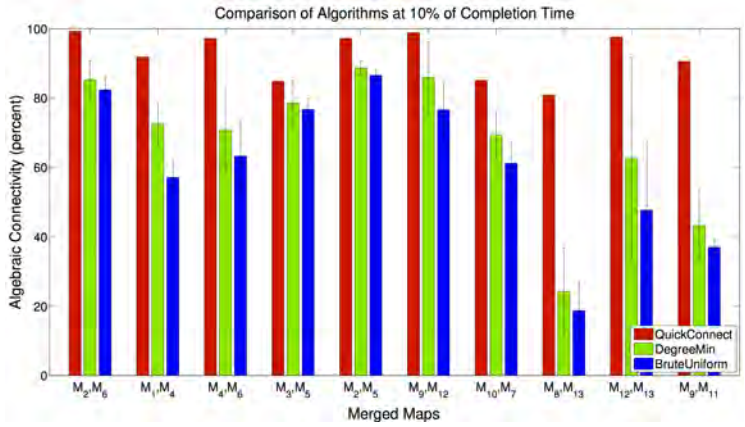
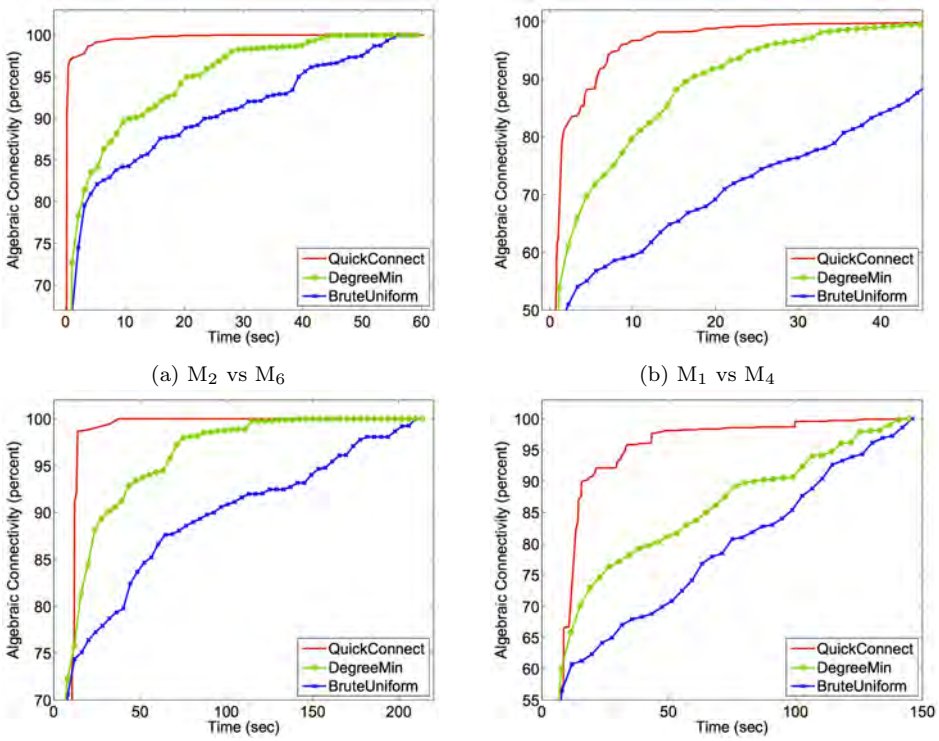


Fig. 7: The chart contrasts algebraic connectivity when merging is stopped at 10% of the time required by brute-force algorithm to complete its exhaustive search. Results are shown for a representative selection of merged maps and three different algorithms are compared: QuickConnect, DegreeMin, and BruteUniform. Error bars representing one standard deviation are shown for randomized algorithms. Note that since algebraic connectivity varies for different map couples, we display a normalized value corresponding to the percent of the maximum value.

The temporal evolution of the merging process for a subset of representative map pairs from Figure 7 is shown in Figure 8. Note that all algorithms eventually reach the same maximum algebraic connectivity by adding all possible edges. However, QuickConnect achieves a steeper quality gain with its selective insertion process during the exploration phase. For instance, during the merging of the map pair shown in Figure 8a 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%. Similar trends are observed in the rest of the dataset as can be seen in Figure 8. These charts experimentally support the claim that the exploration stage successfully identifies the small portion of edges that substantially increase algebraic connectivity.



(a) M_2 vs M_6 (b) M_1 vs M_4

(c) M_9 vs M_{12} (d) M_{10} vs M_7

Fig. 8: Normalized algebraic connectivity of four pairs of merged maps as a function of time for the three algorithms are presented.

Finally, in figure 9 we show a further analysis of the changes in algebraic connectivity for the last map merging example shown in Figure 8. The figure analyzes the spikes in algebraic connectivity observed during the exploration stage of QuickConnect. For each of the three major increases we show the two images associated with vertices being connected by the new edge.

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 10, 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 could 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.

When multiple maps created by teams of robots need to to be merged, it is important to maintain the same performance displayed by the pairwise matching algorithm. Thus, we evaluated the performance of the generalized version of the proposed merging algorithm, QuickConnect-simultaneous, on the same datasets. For comparison purposes we also devised modified versions of previously presented methods: *BruteUniform-*

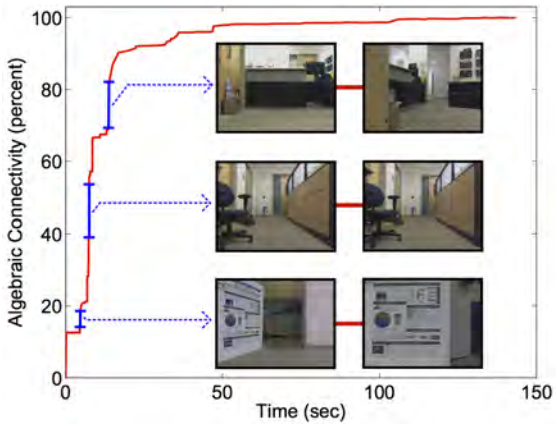
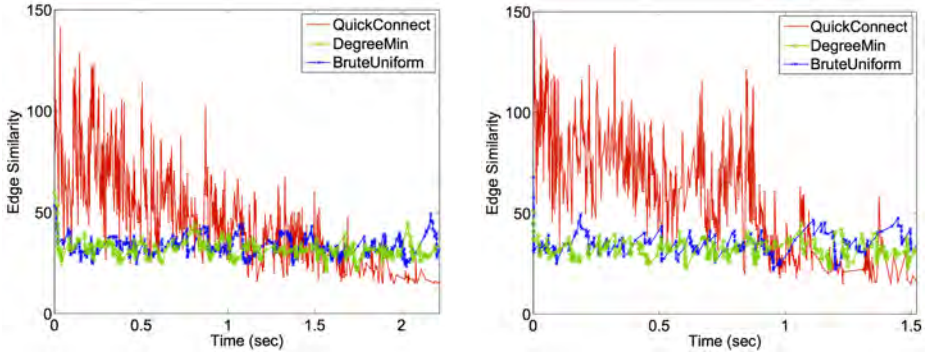


Fig. 9: Images associated with the vertices being connected by edges causing the largest increases in algebraic connectivity during the explore stage of QuickConnect.



(a) M_2 vs M_6 (b) M_1 vs M_4

Fig. 10: Weights, i.e., similarity of connecting images, of the identified edges for two of the representative merging processes are plotted for three algorithms for the duration of the exploration phase of *QuickConnect*.

simultaneous and *DegreeMin-simultaneous* which randomly sample vertices from the set of all vertices, V_m, uniformly and with a mass distribution inversely proportional to their vertex degree, respectively. Additionally, the standard sequential pairwise merging approach described in Section 6.2 is applied to all pairwise merging algorithms resulting in three new merging methods: *QuickConnect-pairwise*, *BruteUniform-pairwise*, and *DegreeMin-pairwise*. However, as mentioned in Section 6.2, in sequential pairwise merging the algebraic connectivity remains equal to zero until the very last map is merged. Therefore, presenting the quality of merging captured by each algorithm within 10% of total time as in Figure 7 would negatively affect sequential pairwise merging methods. To counter this fact, for each algorithm the time required to reach 90% of the overall quality is recorded and the results are shown in Figure 11. Again, for randomized methods the results are the averages of 20 runs.

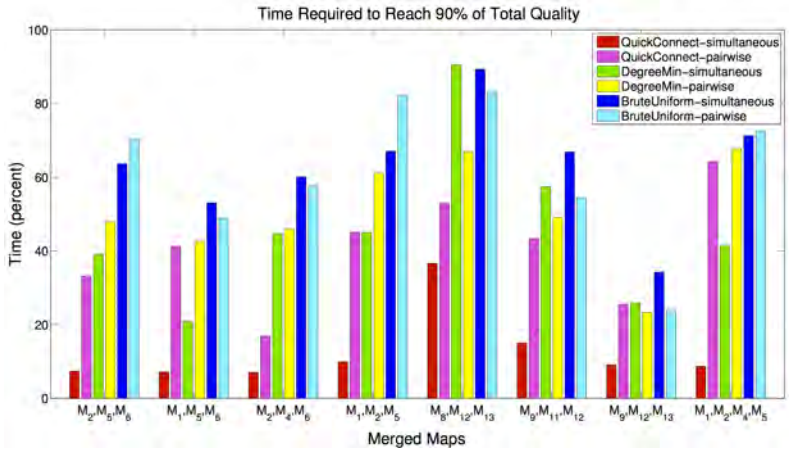


Fig. 11: A representative selection of maps are merged together using simultaneous and pairwise merging algorithms. The cart shows the normalized time required to reach 90% of the total quality.

As shown in Figure 11, QuickConnect-simultaneous, taking advantage of its parallel voting schema, creates edges between vertices from different map pairs, and therefore, outperforms all other algorithms for all merged maps. The ranking for the rest of the algorithms is not consistent. QuickConnect-pairwise in general achieves higher performance than its closest competitor DegreeMin-simultaneous with the exception of second and last group in Figure 11. However, its performance depends on the ratio of the duration of last merging step to the overall merging time, and cannot be generalized since until the last step the algebraic connectivity stays at zero due to the map being disconnected. Therefore, the performance of sequential pairwise merging algorithms suffers more when the number of maps to be merged increases, as in the last group where 4 maps are merged together.

8 Conclusion and Future Work

We have studied the problem of merging appearance-based maps, an underexplored topic that is very relevant in the area of multi-robot systems. Because of the lack of clearly accepted criteria to evaluate the quality of a merging algorithm operating on appearance-based maps, we have put forward algebraic connectivity and illustrated why it is a good criterion to assess the value of a map merging algorithm. The algorithm we proposed, *QuickConnect*, is motivated by this metric and features an anytime behavior by discovering important edges early on during the process. Our claims have been validated by an end-to-end implementation including real time map generation and merging. We are aware that more advanced algorithms creating appearance-based maps have been and are being developed, but this is not the focus of this work, as we position ourselves as *consumers* of maps after they have been generated. Finally, in this paper we have also explored how pairwise merging can be extended to handle the case when multiple maps need to be combined together. In this case too, QuickConnect yields the best 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.

Acknowledgments

A shorter version of this paper appeared at ICRA 2012 [15].

References

1. A. Angeli, S. Doncieux, J.-A. Meyer, and D. Filliat. Real-time visual loop-closure detection. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1842–1847, 2008.
2. A. Angeli, S. Doncieux, J.-A. Meyer, and D. Filliat. Visual topological slam and global localization. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 4300–4305, 2009.
3. 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, 2008.
4. R. Aragüés, J. Cortés, and C. Sagüés. Distributed consensus algorithms for merging feature-based maps with limited communication. *Robotics and Autonomous Systems*, 59(3-4):163–180, 2011.
5. R. Aragüés, J. Cortés, and C. Sagüés. Distributed consensus on robot networks for dynamically merging feature-based maps. *IEEE Transactions on Robotics*, 28(4):840– 854, 2012.
6. A. Birk and S. Carpin. Merging occupancy grids from multiple robots. *Proceedings of the IEEE*, 94(7):1384–1397, 2006.
7. O. Booij, B. Terwijn, Z. Zivkovic, and B. Krose. Navigation using an appearance based topological map. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3927–3932, 2007.
8. S. Carpin. Fast and accurate map merging for multi-robot systems. *Autonomous Robots*, 25(3):305–316, 2008.
9. S. Carpin. Merging maps via Hough transform. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1878–1883, 2008.
10. M. Cummins and P. Newman. Highly scalable appearance-only slam fab-map 2.0. In *Robotics Science and Systems*, 2005.
11. A. Danesi, D. Fontanelli, and A. Bicchi. Visual servoing on image maps. In *Proceedings of the IEEE International Symposium on Experimental Robotics*, pages 277–286, 2006.
12. G. Dedeoglu and G. S. Sukhatme. Landmark-based matching algorithm for cooperative mapping by autonomous robots. In *Distributed Autonomous Robotic Systems 4*, pages 251–260. Springer, 2000.
13. G. Erinc and S. Carpin. Image-based mapping and navigation with heterogeneous robots. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5807–5814, 2009.
14. G. Erinc and S. Carpin. Evaluation criteria for appearance-based maps. In *Proceedings of Performance Metrics for Intelligent Systems*, 2010.

15. G. Erinc and S. Carpin. Anytime merging of appearance based maps. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1656–1662, 2012.
16. F. Ferreira, J. Dias, and V. Santos. Merging topological maps for localisation in large environments. In *Proceedings of the International Conference on Climbing and Walking Robots and the Support Technologies for Mobile Machines*, 2008.
17. M. Fiedler. Algebraic connectivity of graphs. *Czech. Math. J.*, 23(98):298–305, 1973.
18. D. Filliat. Interactive learning of visual topological navigation. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 248–254, 2008.
19. F. Fraundorfer, C. Engels, and D. Nister. Topological mapping, localization and navigation using image collections. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3872–3877, 2007.
20. S. Saeedi Gharahbolagh, L. Paull, M. Trentini, M. Seto, and H. Li. Efficient map merging using a probabilistic generalized voronoi diagram. In *Proceedings of the IEEE/RJS International Conference on Intelligent Robots and Systems*, pages 4419–4424, 2012.
21. S. Saeedi Gharahbolagh, L. Paull, M. Trentini, M. Seto, and H. Li. Map merging using Hough peak matching. In *Proceedings of the IEEE/RSJ International Conference on Intelliegent Robots and Systems*, pages 4683–4688, 2012.
22. H. Hajjdiab and R. Laganiere. Vision-based multi-robot simultaneous localization and mapping. In *Proceedings of Canadian Conference on Computer and Robot Vision*, pages 155–162, 2004.
23. V. Hedau, D. Hoiem, and D. Forsyth. Recovering the spatial layout of cluttered rooms. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1849–1856, 2009.
24. K.L. Ho and P. Newman. Multiple map intersection detection using visual appearance. In *International Conference on Computational Intelligence, Robotics and Autonomous Systems*, 2005.
25. W.H. Huang and K.R. Beevers. Topological map merging. *International Journal of Robotics Research*, 24(8):601–613, 2005.
26. 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 Workshop on Egocentric Vision*, 2009.
27. S. Kirkland. Algebraic connectivity for vertex-deleted subgraphs, and a notion of vertex centrality. *Discrete Mathematics*, 310(4):911–921, 2010.
28. 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.
29. K. Kosnar, T. Krajnik, and L. Preucil. Visual topological mapping. In *Proceedings of European Robotics Symposium*, volume 44, pages 333–342, 2008.
30. H.-C. Lee and B.-H. Lee. Improved feature map merging using virtual supporting lines for multi-robot systems. *Advanced Robotics*, 25(13-14):1675–1696, 2011.
31. Z. Lin, S. Kim, and I. S. Kweon. Recognition-based indoor topological navigation using robust invariant features. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2309–2314, 2005.
32. D.G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

33. J. Luo, A. Pronobis, B. Caputo, and P. Jensfelt. The kth-idol2 database. Technical report, KTH Royal Institute of Technology, CVAP/CAS, 2006.
34. Y. Ma, S. Soatto, J. Kosecka, and S.S. Sastry. *An Invitation to 3-D Vision: From Images to Geometric Models*. SpringerVerlag, 2003.
35. R. Madhavan, C. Scrapper, and A. Kleiner. Special issue on "characterizing mobile robot localization and mapping". *Autonomous Robots*, 2009.
36. L. Maohai, H. Bingrong, and L. Ronghua. Novel method for monocular vision based mobile robot localization. In *Proceedings of International Conference on Computational Intelligence and Security*, volume 2, pages 949–954, 2006.
37. G. L. Mariottini, G. Oriolo, and D. Prattichizzo. Image-based visual servoing for nonholonomic mobile robots using epipolar geometry. *IEEE Transactions on Robotics*, 23(1):87–100, 2007.
38. D. Mosk-Aoyama. Maximum algebraic connectivity augmentation is np-hard. *Operations Research Letters*, 36(6):677–679, 2008.
39. M. Muja and D.G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *International Conference on Computer Vision Theory and Applications*, pages 331–340, 2009.
40. T. Nicosevici and R. Garca. On-line visual vocabularies for robot navigation and mapping. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 205–212, 2009.
41. D. Nister and H. Stewenius. Scalable recognition with a vocabulary tree. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2161–2168, 2006.
42. N. Ergin Özkucur and H. Levent Akin. Cooperative multi-robot map merging using fast-SLAM. In J. Baltes et al., editor, *Robocup 2009: Robot Soccer World Cup XIII*, pages 449–460. Springer, 2010.
43. J. Philbin and A. Zisserman. Object mining using a matching graph on very large image collections. In *Proceedings of Indian Conference on Computer Vision, Graphics & Image Processing*, pages 738–745, 2008.
44. A. Pronobis and B. Caputo. The kth-indecs database. Technical report, KTH Royal Institute of Technology, CVAP/CAS, 2005.
45. A. Quattoni and A.Torralba. Recognizing indoor scenes. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 413–420, 2009.
46. P.E. Rybski, S.J. Roumeliotis, M. Gini, and N. Papanikolopoulos. Appearance-based mapping using minimalistic sensor models. *Autonomous Robots*, 24:229–246, 2008.
47. J. Santos-Victor, R. Vassallo, and H. Schneebeli. Topological maps for visual navigation. In *Proceedings of International Conference on Computer Vision Systems*, pages 21–36, 1999.
48. S. Se, D. Lowe, and J. Little. Local and global localization for mobile robots using visual landmarks. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, volume 1, pages 414–420, 2001.
49. S. Se, D. Lowe, and J. Little. Vision-based mobile robot localization and mapping using scale-invariant features. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 2, pages 2051–2058, 2001.
50. S. Se, D. Lowe, and J. Little. Mobile robot localization and mapping with uncertainty using scale-invariantvisual landmarks. *International Journal of Robotics Research*, 21(8):735–758, 2002.
51. 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, volume 2, pages 1470–1477, 2003.
52. T.F. Smith and M.S. Waterman. Identification of common molecular subsequences. *Journal of Molecular Biology*, 147(1):195–197, 1981.
53. T.T.H. Tran and E. Marchand. Real-time keypoints matching: application to visual servoing. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3787–3792, 2007.
54. C. Valgren, A. Lilienthal, and T. Duckett. Incremental topological mapping using omnidirectional vision. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, pages 3441–3447, 2006.
55. X. S. Zhou and S. I. Roumeliotis. Multi-robot SLAM with unknown initial correspondence: The robot rendezvous case. In *Proceedings of the IEEE/RSJ International Conference on Intelliegent Robots and Systems*, pages 1785–1792, 2006.
56. Z. Zivkovic, B. Bakker, and B. Krose. Hierarchical map building using visual landmarks and geometric constraints. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2480–2485, 2005.
57. Z. Zivkovic, B. Bakker, and B. Krose. Hierarchical map building and planning based on graph partitioning. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 803–809, 2006.
58. Z. Zivkovic, O. Booij, and B. Krose. From images to rooms. *Robotics and Autonomous Systems*, 55(5):411–418, 2007.
59. Z. Zivkovic, O. Booij, B. Krose, E.A. Topp, and H.I. Christensen. From sensors to human spatial concepts: An annotated data set. *IEEE Transactions on Robotics*, 24(2):501–505, 2008.