Efficient Decentralized Visual Place Recognition Using a Distributed Inverted Index

Titus Cieslewski and Davide Scaramuzza

***Abstract*—State-of-the-art systems that do place recognition in a group of n robots either rely on a centralized solution, where each robot's map is sent to a central server, or a decentralized solution, where the map is either sent to all other robots, or robots within a communication range. Both approaches have their drawbacks: centralized systems rely on a central entity which handles all the computational load and cannot be deployed in large, remote areas; decentralized systems either exchange n times more data or preclude matches between robots that visit the same place at different times while never being close enough to communicate directly. We propose a novel decentralized approach which requires a similar amount of data exchange as a centralized system, without precluding any matches. The core idea is that the candidate selection in visual bag-of-words can be distributed by pre-assigning words of the vocabulary to different robots. The result of this candidate selection is then used to choose a single robot to which the full query is sent. We validate our approach on real data and discuss its merit in different network models. To the best of our knowledge, this is the first work to use a distributed inverted index in multi-robot place recognition.**

***Index Terms*—Distributed Robot Systems, Networked Robots, SLAM**

I. Introduction

MANY robotic applications can benefit from parallel deployment of multiple robots. In a search and rescue mission for example, the search area can be subdivided, so that each robot has less space to cover, resulting in quicker task completion. In order for the robots to efficiently collaborate, they need to know where they are with respect to each other and to the search area. In unstructured, GPS-denied environments, a robust way to achieve this is to have each robot create a map of the environment and then let it localize itself in the maps created by other robots. In such a scenario, if two robots A and B want to find out their relative poses, robot A can send its map data M_A to robot B (or vice versa). Robot B can then perform place recognition between M_A and M_B. When extending this problem to $n > 2$ robots, roboticists typically use one of the following approaches:

Centralized: Each robot sends its map to a central server, which performs all place recognition computations [1]–[5].

Manuscript received: September, 10, 2016; Revised November, 30, 2016; Accepted December, 24, 2016.

This paper was recommended for publication by Editor Antonio Bicchi upon evaluation of the Associate Editor and Reviewers' comments. This work was supported by Armasuisse (project number 043-12) and the National Centre of Competence in Research Robotics (NCCR) through the Swiss National Science Foundation.

The authors are with the Robotics and Perception Group, University of Zurich, Switzerland—http://rpg.ifi.uzh.ch.

Digital Object Identifier (DOI): see top of this page.

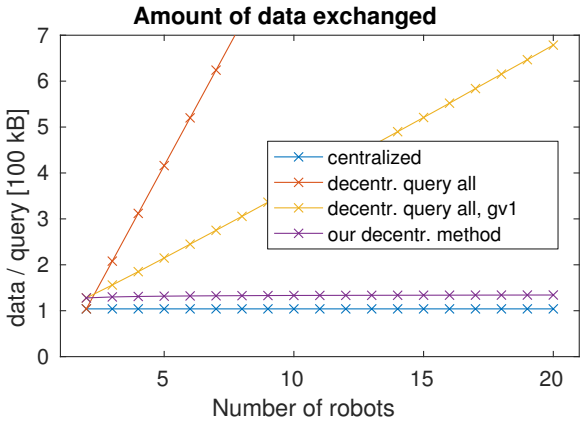


Fig. 1. Our method significantly reduces the amount of data exchanged for place recognition in a decentralized multi-robot system. This is achieved by distributing the candidate selection of bag of words place recognition. *decentr. query all, gv1* stands for a query-all approach where geometric verification is performed only at one robot.

Decentralized, query all: Each robot sends its map to every other robot. Then, each robot can run place recognition between its own map and the maps of all other robots [6].

Decentralized, query in range: Same, but maps are only shared with robots in a certain communication range [7]–[9].

In this paper we show that, in terms of the amount of data exchanged as a function of robot count, current decentralized place recognition schemes are less scalable than their centralized counterparts. We then propose a novel decentralized scheme specifically designed for feature-based visual approaches, which scales similarly to a centralized method (see Fig. 1). This comes at the cost of only mildly affected recall. The proposed method is inspired by work on distributing visual queries in server clusters [10], [11]. The gist is that we distribute the task of candidate frame selection into n queries of size $\frac{C}{n}$, where C is the amount of data that would otherwise be sent to every other robot. We then send the full query for geometric verification only to the one robot that has observed the best candidate. The motivation for using a decentralized system instead of a centralized one is that the latter has a computational bottleneck at the central station, which should always stay operational. Furthermore, centralized systems cannot be deployed in remote, large areas because of limited communication range.

II. Related Work

The presented decentralized place recognition algorithm is based on bag-of-words visual place recognition. This method has been pioneered by [12], and we make use of further improvements as presented by [13], [14]. An essential part

of our system is the use of an inverted index, a technique borrowed from document retrieval [15]. There, the inverted index is a key-value map where the keys are words and the value for each word is a list of documents containing it.

The fact that inverted indices can be distributed has already been discovered in document retrieval literature in the early 90s [16], [17]. Back then, the main focus was on distributing document retrieval to multiple processors on the same machine. With the advent of distributed hash tables [18], [19] and faster Internet connections in the early 2000s, however, researchers started to propose distributed document retrieval over peer-to-peer networks [20], [21], all pivoting around distributed inverted indices.

Only a small number of these can be effectively applied to visual place recognition, as there are significant differences between document and place retrieval. In visual place recognition, the query size is typically equal to the document size. Furthermore, each query is simultaneously inserted as a new document into the database. Lastly, while document retrieval is typically about obtaining all documents relevant to a query, we will contend with at most a single match per query in our scenario.

Some work exists in distributed image retrieval: Yan et al. [22] have developed distributed image search in camera sensor networks. Their query approach, however, is decentralized, query all. Aly et al. [23] proposed to distribute kd-tree-based image retrieval. However, the hierarchical nature of kd-trees requires that approach to pass through a central server. Ji et al. [10], [11] made the step of going from distributing images among servers to distributing the vocabulary, which is essential to effectively reduce the amount of data exchange. Doing this work in the context of image queries in a local network, however, their focus is more on overall query speed than it is on the amount of transmitted data. Thus, they do not consider that it is not scalable, in terms of bandwidth, to return partial scores for all images — which we address in section IV-C. Furthermore, we exploit geometric verification, which allows us to simplify the distributed query without affecting precision. Finally, while their approach distributes the workload, they use a central server that coordinates each query. In contrast, our approach is perfectly decentralized as soon as every robot knows the address of every other robot.

To the best of our knowledge, we are the first ones to apply a distributed inverted index in multi-robot place recognition.

III. Amount of data exchanged: scalability

Regarding place recognition systems, we will from here on use the term *add-querying* a place, meaning *to simultaneously add a place to the database and query it*. We assume that in a multi-robot system, places (in our case visual keyframes) observed by each robot are add-queried regularly. The expected total amount of data exchanged in the system due to place recognition within a unit of time is then:

$$d_{tot} = n \cdot \mathbf{E}[f] \cdot d_q, \tag{1}$$

where n is the number of robots in the system, $\mathbf{E}[f]$ the expected frequency at which each robot add-queries a place, and d_q the data that needs to be transmitted for each add-query. For now, if the same data is transmitted to two different robots, we count it twice, and we consider direct connections between robots. See Section VII for a discussion of multi-cast and multi-hop networks. We assume that both n and $\mathbf{E}[f]$ are factors of d_{tot} independent of the approaches considered hereafter, so we characterize their scalability solely by d_q.

In centralized place recognition, d_q is independent of n, or $\mathcal{O}(1)$, since each add-query is sent only to the central server. In decentralized, query-all place recognition, each add-query is sent to $n-1$ robots, and d_q assumes a complexity of $\mathcal{O}(n)$. In the *query in range* variant, the amount of robots to which an add-query is sent varies, but the complexity is $\mathcal{O}(n)$ in the worst case. It could be limited to $\mathcal{O}(1)$ by only sending queries to the closest k robots, but that would preclude localization with respect to robots outside of that clique. Limited-communication approaches that accumulate add-queries or propagate them through the network (such as [7]) reduce to a d_q complexity of $\mathcal{O}(n)$, since in the end each place gets shared with every robot. Thus, previous decentralized schemes, which have a d_q complexity of $\mathcal{O}(n)$ are inherently less scalable than the centralized scheme with its complexity of $\mathcal{O}(1)$.

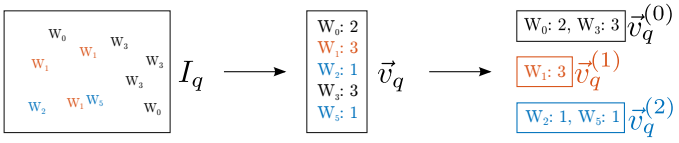
In contrast, we propose a decentralized scheme whose d_q is close to the centralized case. This is achieved by splitting the add-query into two queries: first, a distributed query in which messages of $\mathcal{O}(\frac{1}{n})$ size are sent to every other robot [1] second, a targeted query where the full place information is sent to one robot only. The recipient of the targeted query is determined by the outcome of the distributed query.

IV. Methodology

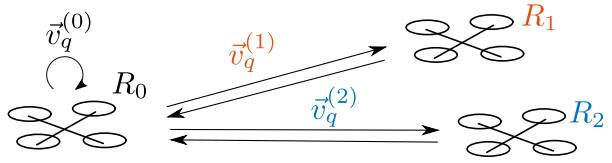
For simplicity, we consider place recognition algorithms which identify places with images. This reduces visual place recognition to matching of images. State-of-the-art visual place recognition is typically performed in two steps: 1) The full set of previously seen images $\{I_i\}$ is reduced to a set of candidates using visual appearance-based matching or other criteria. 2) These candidates are verified for geometrical consistency with the query image I_q. Only places passing this step are considered to be a match. A popular choice for the first step is comparing visually salient features, typically image patches around keypoints. Since image patches are high-dimensional and matching them is expensive, they are reduced to lower-dimensional features. This reduction can be achieved using PCA or variations thereof [24], [25], quantization of the high-dimensional space, e.g. using k-means [12], or more recently, convolutional neural networks [26].

We focus on approaches which quantize the feature space, since they are most amenable to distribution, as will become clear soon. When quantizing the feature space, each cell is identified by what is called a *word* W_w in a *vocabulary* $\{W\}$. A frame I can now be represented as a *bag-of-words* vector

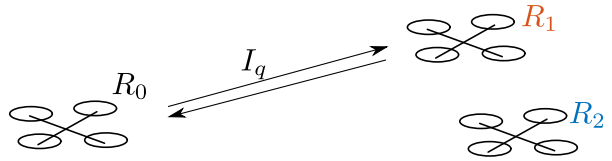
[1] Strictly speaking, the complexity is still $\mathcal{O}(n)$ due to overhead and responses from sending data to n other robots. However, since the amount of the bulk of the data is reduced by a factor of n, the resulting total amount of data sent per query is much closer to the centralized case, see Fig. 1.;



(a) The query frame I_q contains several features which are mapped to visual words W_i. I_q can thus be represented by a bag-of-words vector $\vec{v}_q$. Responsibility for visual words is distributed among the robots and $\vec{v}_q$ is subdivided accordingly.



(b) R_{r_q} sends to each robot R_r the portion of $\vec{v}_q$ which it is responsible for. Each robot R_r then returns a *partial response*.



(c) From the partial responses, R_{r_q} derives which robot is the most likely to have seen a place matching I_q. Only to that robot it sends the full I_q to perform standard place recognition with geometric verification.

Fig. 2. The essential steps of the proposed method. Illustrated for $n = 3$ and where the querying robot R_{r_q} is R_0.

$\vec{v} \in \mathbb{R}^{|\{W\}|}$, where the w-th coefficient of $\vec{v}$ is the TF-IDF value of W_w in I. The TF-IDF (term frequency - inverse document frequency) value is the frequency of this word in I normalized by its overall frequency. It reflects that overall less frequent words carry more information when matching frames containing them [15]. Now, a given frame I_i can be matched to the query frame I_q using the score

$$s(i,q) = 1 - \frac{1}{2}\left|\frac{\vec{v_i}}{|\vec{v_i}|} - \frac{\vec{v_q}}{|\vec{v_q}|}\right|_1, \tag{2}$$

where $|\cdot|_1$ stands for the ℓ_1 norm. If $s(i,q)$ is close to 1, $\vec{v_i}$ and $\vec{v_q}$ are similar, in which case I_i is passed to the geometric verification step. If the vectors $\vec{v}$ are pre-normalized, s can be re-written as a sum for each w:

$$\begin{aligned} 2 \cdot s(i,q) &= 2 - |\vec{v_i} - \vec{v_q}|_1 = |\vec{v_i}|_1 + |\vec{v_q}|_1 - |\vec{v_i} - \vec{v_q}|_1 \\ &= \sum_w v_{i,w} + v_{q,w} - |v_{i,w} - v_{q,w}| \doteq \sum_w s_w(i,q) \end{aligned} \tag{3}$$

An important property of this is that a given summand s_w is zero if either $v_{i,w}$ or $v_{q,w}$ is zero. This allows fast candidate selection using inverted indices. We exploit the sum form of the score to distribute the candidate selection.

A. Distributing candidate selection

Regarding key-value lookup in a peer-to-peer network, distributed hash tables show that significantly less data can be exchanged by deterministically assigning keys to peers, and having these peers store the corresponding values. If peers that want to perform a query can locally compute which peer is responsible for storing the sought value, then there is no need to send the query to other peers. This cannot be directly applied to place recognition since, to the best of our knowledge, there is no deterministic mapping of observations of the same place to a unique string. In contrast, words of a pre-calculated vocabulary can be assigned to specific peers. We therefore distribute the evaluation of scores in (3) by assigning each word W_w to a robot $R_{h(w)}$ with $h(w) = w \mod n$. Each bag-of-words vector $\vec{v_i}$ can then be subdivided into n partial vectors $\vec{v}_i^{(r)}$, which only contain the coefficients $w : h(w) = r$ (see Fig. IV). Note that this scheme does not support dynamic sets of robots natively. Adding robots can be handled by not distributing queries to them. For orderly departure of robots, however, we cannot conceive a method that would not transmit a lot of data. The special case of node failure is analyzed in a similar system by [10].

Now, each robot can calculate a partial score

$$s_r(i,q) = \sum_{w:h(w)=r} s_w(i,q) \tag{4}$$

for every potential match using only $\vec{v}_q^{(r)}$ and $\vec{v}_i^{(r)}$. Since in multi-robot place recognition the potential matches for the q-th query is the set of all previous queries, robot R_r only needs to receive $\vec{v}_q^{(r)}$ at the q-th query to calculate the partial scores $s_r(i,q)$ for all candidates. It can then add $\vec{v}_q^{(r)}$ to $\{\vec{v}_i^{(r)}\}$ for future queries. Thus, the first part of our place recognition query consists of sending $\vec{v}_q^{(r)}$ to each robot R_r (see Fig. IV). For now, let's assume that robot R_r responds with a map $\{i \to (r_i, s_r(i,q))\}$. We address the scalability of these responses in section IV-C. Having received $\{i \to (r_i, s_r(i,q))\}$ from each robot R_r, the querying robot can calculate the score for all candidate frames:

$$\{i \to (r_i, s(i,q) = \sum_r s_r(i,q))\} \tag{5}$$

B. Asking the right peer for geometric verification

State-of-the-art methods such as DBoW2 [14] or its implementation in ORB-SLAM [27] pass multiple candidates to geometric verification. To reduce data exchange, we send the geometric verification query to only one robot R_{i^*} (Fig. IV), potentially sacrificing recall. Note that this one robot may still evaluate multiple candidate frames. We let it perform local candidate selection between I_q and all the frames that it has previously observed itself. R_{i^*} is chosen from (5) with:

$$i^* = \arg\max_i \sum_r s_r(i,q) \tag{6}$$

C. Scalability with the map size - minimal partial responses

As hinted at the end of section IV-A, having each peer respond with $\{i \to (r_i, s_r(i,q))\forall i\}$ is not scalable, as the response size would grow linearly with the amount of frames observed by all robots. We have found a simple way to make the response size scalable: we make each robot return only the one frame with the highest partial score $s_r(i,q)$. While the underlying assumption

$$i^* = \arg\max_i \sum_r \begin{cases} s_r(i,q) & \text{if } i = \arg\max_{i'} s_r(i',q) \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

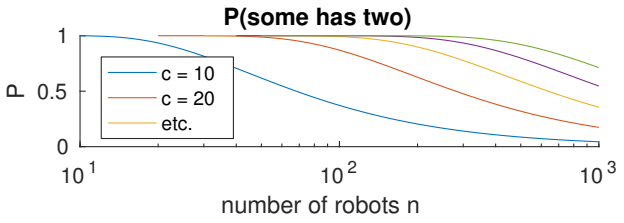


Fig. 3. Probability that out of n robots, at least one is assigned at least two out of c words. Being responsible for multiple words increases the odds of contributing to a successful distributed candidate selection even if only the single best partial score is returned.

might not hold for general vectors, we have found that it works well enough for visual bag-of-words vectors and the range of the robot count n that we have performed our experiments with. We identify some mechanisms that help with this: as has been previously shown [28], a small part of observed words contributes a big part towards the score. In the extreme case where only one word is responsible for the entire score, it is trivial show that (7) holds. In practice however, there are up to a couple dozen words that contribute to most of the score. For higher relevant word counts, another mechanism is in place: in order for an outlier to be considered the correct match by our evaluation (7), it needs to consistently have better partial scores $s_r(i, q)$ than the true positive. This is unlikely unless it observes a very similar scene. In contrast, if the outliers do not have consistently better partial scores, it should be sufficient for the true positive to have the best partial score at only two robots — if all of the top contributing words contribute similarly, the true positive will be the maximum in (7). Finally, for matches with a larger amount of contributing words, it is more likely that some robots will be responsible for several of them, which decreases the odds that an outlier outperforms the true positive at that robot while at the same time increasing the weight of the partial score by that robot. For instance, the probability that there is at least one of n robots responsible for two of the c contributing words is 1 minus the probability that no robot is assigned two or more words, or $1 - \prod_{j=1}^{c} \frac{n+1-j}{n}$ (see Fig. 3). Evidently for as little as twenty contributing words, groups of up to 100 robots will almost certainly have at least one robot that is responsible for two words. A detailed study of mathematically derived guarantees for these mechanisms is outside of the scope of this paper. Instead, we provide a detailed case analysis for $n = 30$ in Fig. 4.

V. Experiments

In order to evaluate our system, we use the data from KITTI 00 [29] and Málaga 10 [30] as if it were recorded by a group of robots: we first split each dataset into 20 non-overlapping sequential parts (Fig. 5), run the visual odometry (VO) thread of ORB-SLAM [27] on each part, and save each resulting map together with all timestamps that capture at what time what portion of the map has been created. We then load all maps and run the ORB-SLAM [27] version of DBoW2 [14] between all possible pairs of maps, after having performed bundle adjustment. We save the resulting matches as ground-truth reference, as we evaluate precision and recall

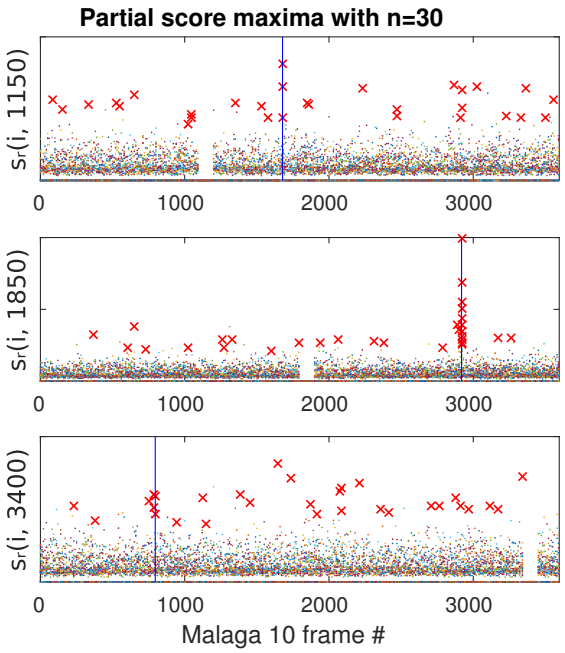
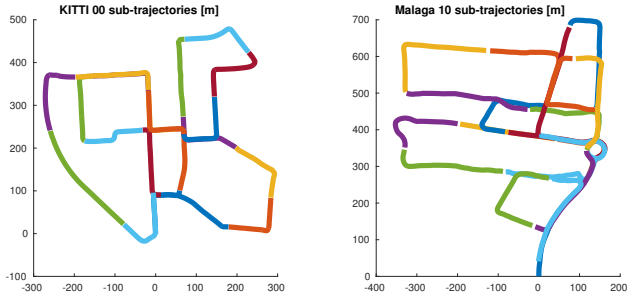
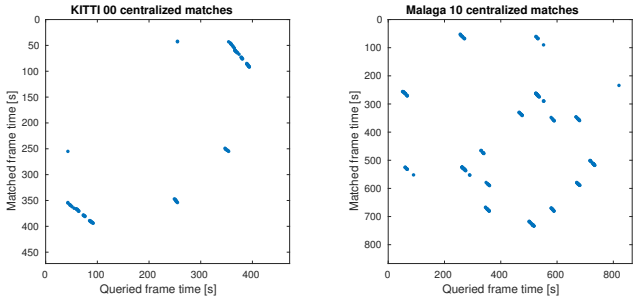


Fig. 4. A close look at three instances of partial maxima-based candidate selection (7). For three query frames with a unique ground truth match (blue line), dots represent the partial scores $s_r(i, q)$ for all robots r (color) and all other frames i (x-axis) except the ones inside a radius of 50 key-frames (gaps). Red crosses represent the best partial candidate $(i, s_r(i, q))$ at each robot. In the first two cases, the true positive accumulates enough partial scores to outperform the outliers, which do not manage to outperform the true positive consistently across robots. In the third case, however, an outlier at around $i \sim 2000$ successfully accumulates two maximum partial scores, which is enough to outperform the weak true positive. Note how the true positive suffers from having neighboring frames which also have maximum scores at some robot. This suggests that clustering partial maxima that are close in time, or using a lower frame rate for place recognition could improve recall.



(a) KITTI 00 (b) Málaga 10

Fig. 5. Sub-trajectories assigned to different robot processes.



(a) KITTI 00 (b) Málaga 10

Fig. 6. Confusion matrices for centralized evaluation on the used datasets.

relative to this centralized place recognition. See Fig. 6 for the resulting confusion matrices. We run 10 trials of our system for each of $n \in [2, 20]$, where the maps are chosen such that there is always at least one ground truth match. This avoids meaningless recall values for small n.

All experiments are performed on a single i7 machine and run in real time. This is enabled by pre-calculating the VO, bundle adjustment (required for good results in geometric verification) and all descriptor-to-word mappings. Each robot is implemented as an independent process and networking is done using ZeroMQ and Google Protocol Buffers. We use the pre-trained visual vocabulary that is provided with ORB-SLAM.

A. Data exchange evaluation

For every place recognition query we log both the amount of data exchanged for the distributed candidate selection query and responses d_c, and the amount of data exchanged for the geometrical verification query and response d_g. For each evaluated robot count n we average $(d_c + d_g)$ and report them as data exchanged per query by our method. We compare this to d_g, our model traffic for centralized place recognition and $(n-1) \cdot d_g$, our model traffic for classic decentralized place recognition. Since we observe that $d_g > d_c$, we also propose an alternative query-all approach where first only $\vec{v}_q$ is sent to all robots for candidate selection based on their individual maps. Similarly to our approach, the querying robot then only selects the one robot with the best candidate and sends to it the full information required for performing geometric verification. We model the data exchange for this with $(n-1) \cdot d_c + d_g$.

B. Accuracy evaluation

In order to evaluate the impact of our approach adjusted for the accuracy of the underlying (centralized) place recognition algorithm, we evaluate precision and recall relative to the matches of that centralized algorithm. Absolute precision and recall for the centralized algorithm are reported in [14]. Relative precision and recall are then calculated using the binary classification values described in Table I. Since the reference place recognition can match backwards in time, we expand the matches returned by our method by all their inverse matches; that is, if our method matches I_A to I_B, a match from I_B to I_A is added to the set of matches for evaluation. Furthermore, matches where the query or match frame is less than $2s$ off a corresponding ground truth match are also considered true positives.

Since both the centralized and the decentralized method use the same geometric verification method, one can expect that relative precision is 1. We have verified that this is indeed the case, therefore we only report relative recall. Very occasionally there are single false positives among ~ 100 true positives.

C. Computation time and memory evaluation

Finally, we evaluate the computational cost of our approach by measuring the CPU times for five subroutines: t_{lp}, the local

	tp	**fp**	**fn**
If $q(I_q) \neq \emptyset$ and $q(I_q) \subset q'(I_q)$	1	0	0
If $q(I_q) \cap q'(I_q) \neq \emptyset$ yet $q(I_q) \not\subset q'(I_q)$	1	1	0
If $q(I_q) \neq \emptyset$ and $q(I_q) \cap q'(I_q) = \emptyset$	0	1	0
If $q'(I_q) \neq q(I_q) = \emptyset$	0	0	1
If $q'(I_q) = q(I_q) = \emptyset$	0	0	0

TABLE I

RELATIVE TRUE POSITIVE, FALSE POSITIVE AND FALSE NEGATIVE EVALUATION FOR EACH QUERY FRAME I_q. HERE, $q(I_q)$ IS THE SET OF MATCHED FRAMES RETURNED BY OUR ALGORITHM, WHILE $q'(I_q)$ IS THE SET OF MATCHED FRAMES RETURNED BY THE CENTRALIZED VERSION.

processing time without processing related to the distributed candidate selection. This contains mostly addition of the query frame to the local database and reference score calculation, a DBoW trick to adaptively select a threshold for candidate rejection [14]. We use this reference score only for the full query that is sent to the robot selected by the distributed candidate selection query. The local processing time dedicated to the distributed candidate selection is measured separately as t_{ld} and contains mostly splitting up of the bag-of-words vector $\vec{v}_q$ and the evaluation of the partial scores (4) of self-assigned words. The handling of distributed candidate selection queries from other robots is measured as t_{rd}. Finally, full query handling is measured with candidate selection time t_{rcs} and geometric verification time t_{rgv}.

We are able to reliably measure these times with up to 16 robot processes, beyond which we encounter anomalies which we attribute to context switching, file system waits and other low-level causes. Such anomalies also occur spuriously with fewer robot processes, which is why we use the median and quartiles to accumulate these measurements.

With these measurements, we can estimate the total computation cost per query with $t_{lp} + t_{ld} + (n-1)t_{rd} + t_{rcs} + t_{rgv}$. We compare this with a decentralized, query-all approach whose time is modeled with $t_{lp} + (n-1)(t_{rcs} + t_{rgv})$, as well as a decentralized, query-all approach with only one geometric verification modeled with $t_{lp} + (n-1)t_{rcs} + t_{rgv}$. We do not compare with a centralized approach, since there, all the computational and memory load is focused on a single machine.

In terms of memory requirements, our approach should be somewhat more expensive than the query-all approach. Since for the geometrical verification query we maintain the same data structures as the query-all approach, our memory requirements are strictly larger by the data structure necessary for distributed candidate selection. The accumulation of this data should be exactly the same as the inverted index data that is used for local candidate selection at each robot, except that it is re-arranged between the robots, and each frame reference also contains a reference to the robot that has seen it. In particular, if all robots query the same amount of frames, the size of the data structure should not depend on the robot count n. We verify this using a profiling tool.

VI. RESULTS

The amount of data exchanged per query is reported in Fig. 1. Unsurprisingly, our approach requires less bandwidth than both decentralized, query all approaches starting with as little as 3 robots. Furthermore we see that $d_c < d_g$. A

Distributed query payload size $= 2000\cdot$	
Word index and score	$4 + 4$
Geometric verification query payload size $= 2000\cdot$	
Short integer keypoint coordinates	$2 \cdot 2$
Single precisiton keypoint uncertainty	4
ORB descriptor	32
Single precistion landmark 3d positions	$3 \cdot 4$

TABLE II

PAYLOADS, IN BYTES, OF THE DISTRIBUTED CANDIDATE SELECTION QUERIES (CORRESPONDS TO d_c) AND THE TARGETED GEOMETRIC VERIFICATION QUERY (CORRESPONDS TO d_g). THE FACTOR 2000 STEMS FROM ORB-SLAM, WHICH TRACKS 2000 KEYPOINTS AT EVERY FRAME.

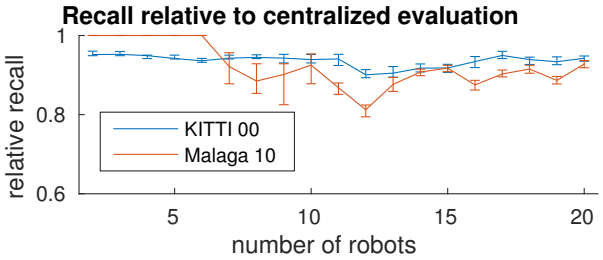


Fig. 7. Recall relative to centralized place recognition. Mean, minimum and maximum are reported for 10 trials each.

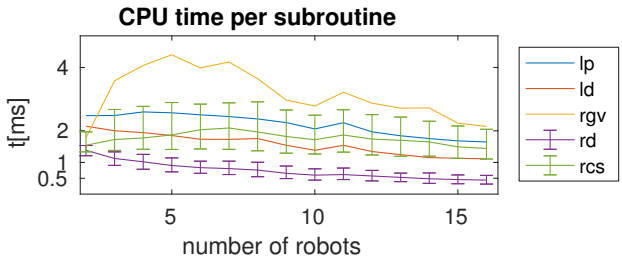


Fig. 8. Subroutine durations as a function of robot count. *lp*: local processing, *ld*: local DII query, *rgv*: remote geometric verification, *rd*: remote DII query, *rcs*: remote candidate selection.

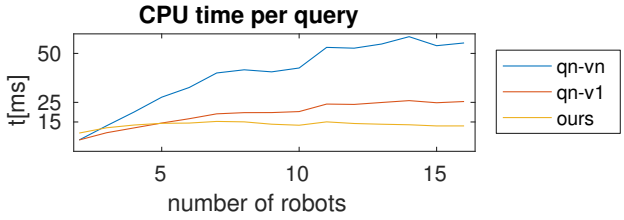


Fig. 9. Estimation of per-query total duration based on subroutine durations (see Fig. 8). *qn-vn*: query all with geometric verification at each robot, *qn-v1*: query all with geometric verification at only best robot.

detailed listing of the data transmitted at each query, minus overhead and responses, is provided in Table II. We are using the geometric verification method from ORB-SLAM [27], which requires both the original ORB descriptors of keypoints for accurate keypoint matching in presence of ambiguities as well as the 3D positions of corresponding landmarks for scale drift-aware 3D-3D RANSAC. Thus, d_g could be reduced by omitting the landmark 3D positions and using 2D-3D RANSAC. We speculate that d_g could be further reduced by not sending ORB descriptors, but only node IDs at a certain level of the vocabulary tree, and using a RANSAC that can take 1-to-n matches into account. Finally, as annunciated at the end of section III, both d_c and d_g can be reduced by only sending a subset of most relevant keypoints. Neither of these reductions would change the fact that our approach is more scalable than classical decentralized place recognition. Moreover, since $d_c < d_g$ should be the case even if d_g is strongly reduced, our approach is always worthwhile with more than three robots — at least considering the amount of data that needs to be exchanged.

From an accuracy perspective, Fig. 7 shows that for groups of up to 20 robots, recall is only mildly affected, being typically above 0.9, and at worst 0.8 times as high as with centralized place recognition. Interestingly, the biggest amount of robots does not result in the worst recall. We have found that the groups of robots that exhibit lower recall happen to traverse portions of the map (Fig. 5) that are hard to match — note that adding a robot to the system also adds a new sub-trajectory, which has new potential matches with respect to the rest of the map. As we add more robots with trajectories that are easy to match to a group with low recall, the overall true positives increase faster than the false negatives, and recall increases.

The computation times per subroutine, as evaluated on KITTI 00, are reported in Fig. 8. The local processing time t_{lp} is relatively large, and profiling with `valgrind` reveals that most of it can be attributed to the calculation of the reference score, which in our case involves intricate data structure traversal. Both the local and remote distributed candidate selection decrease with the amount of robots. This is on one hand well explained by the fact that the size of the partial vectors $\vec{v}_q^{(r)}$ which are processed per robot decreases as the robot count increases. On the other hand, it might seem surprising, since at the same time, the total amount of frames grows with the amount of robots. This, however, just indicates the efficiency of the inverted index approach. The candidate selection from a full query at a single robot's map t_{rcs} fluctuates, which we attribute to heuristics applied in DBoW. In fact, profiling in one instance reveals that ℓ_1 score evaluation takes around 40% of the time. However, heuristics are applied for candidate filtering based on common word count before score evaluation. Similar heuristics are applied for geometric verification time t_{rgv}, which exhibits even stronger fluctuations. The estimated overall execution time is reported in Fig. 9. The sampled memory footprint of the distributed inverted index per robot in the KITTI 00 dataset fluctuates between roughly 7 and 10 MB out of 100 MB overall memory use and is not correlated with the robot count. We explain these fluctuations with uneven word distribution and pre-allocation of `std::vector`, which we use as mapee in the inverted index.

VII. DISCUSSION: SCALABILITY IN REAL NETWORKS

So far, we have seen that our approach requires one order less data to be exchanged than previous decentralized visual place recognition approaches that also do not preclude matches. In this section, we re-consider all the data exchanged per unit of time (1) and discuss how the scalability with respect to the amount of exchanged data translates to scalability in real networks. Seeing the main application of our approach in mobile robotics, we only consider wireless data transmission. The following discussion is based on a comprehensive survey

of wireless sensor networks [31]. Considering the large amount of available protocols and considerations listed in that survey, a complete discussion is well outside of the scope of this paper. Instead, we discuss two points that we consider most relevant to a prospective deployment of our system. To compare our approach to a decentralized query-all approach we only analyze the communication necessary for candidate selection. This is the only communication in our approach which is not strictly a subset of the communication in the former. Note that the communication needed for centralized place recognition is strictly a subset of all considered decentralized approaches, and so we will not consider it in this comparison. In the considered scenario, all robots make a query simultaneously. We define $d^{t\to r}$ the data transmitted from robot t to robot r, $b(d)$ the byte size of d, $t(d)$ the time required to transmit d over a direct connection and p_l the probability of packet loss in a direct connection. Subscript o denotes the use of our approach while subscript a denotes the use of a query-all approach. We use the following simplified model:

$$b(d^{t\to i}) = b(d^{t\to j}) \quad \forall t, i, j \in \{r\} \tag{8}$$

$$b(d_a^{t\to i}) = n \cdot b(d_o^{t\to i}) \doteq b^\star \tag{9}$$

$$t(d^{i\to j}) = B \cdot \frac{1}{1-p_l} b(d^{i\to j}), \tag{10}$$

where B is the bandwidth that is assumed equal for all connections and i and j in (10) are within communication range. (10) reflects that a packet might need to be re-transmitted once with a probability of p_l, twice with a probability of p_l^2, etc.

Channels, multi-casting and packet loss A given wireless channel can only have one active transmitter at a given time, while it may have multiple concurrent receivers. Hardware designed for a given frequency typically only has access to a limited amount of channels. If the amount of robots in a group is smaller than the amount of available channels, each robot could receive on a separate channel and switch the transmitter channel depending on which other robot it would like to talk to. Assuming the robots are all in communication range and assuming that they could coordinate with negligible cost that only one robot talks to a given robot at the same time, each robot r would simply need to sequentially transmit $d^{r\to i}\ \forall i \neq r$ in parallel with the other robots and so the total time for candidate selection $t = (n-1)\cdot t(d^{t\to i})$. Applying (8)-(10), we obtain $t_a = n \cdot t_o$. Note that as long as enough channels can be provided, the duration of candidate selection in our approach is independent of the size of the robot group!

If only $c < n$ channels can be provided, several robots need to listen to the same channel. With that, the data thus additionally needs to be addressed, and not all robots can receive data at the same time. In the worst case, if $c = 1$, we can show that with the above transmission approach $t = (n-1)\cdot n \cdot t(d^{t\to i})$. $t_a = n \cdot t_o$ still holds, but the scalability with respect to the number of robots is now limited. Note that with the query-all approach we could exploit that the queries originating from one robot are all the same. Since all robots are listening to the same channel, the querying robot could thus address its data to all robots (multi-cast), and if $p_l = 0$ we can show that $t_a = t_o$, and our approach has no advantage over the query-all approach besides the use of less CPU time.

However, our approach still has merit in the presence of packet loss. Assuming that the packet loss dice are rolled for each receiver independently, (10) needs to be adapted for multi-cast. Now, a packet needs to be retransmitted with probability $(1-(1-p_l)^n)$, and thus

$$t(d^{i\to\{r\neq i\}}) = B \cdot \frac{1}{(1-p_l)^n} b(d^{i\to\{r\neq i\}}) \tag{11}$$

and our approach remains faster with $t_o = (1-p_l)^{n-1} t_a$. Reliability of multi-cast is a well-known issue in wireless networks [32].

Multi-hop networks We have previously assumed that all robots are within communication range of each other. However, n-to-n communication can be achieved with more relaxed constrains using multi-hop networks. To that end, data exchanged between two robots that are not directly connected needs to be routed according to the available topology [33] and the robots need to be controlled in a way that maintains overall connectivity [34]. Due to the sheer amount of possible topologies in a multi-hop network, we cannot derive and compare t_o and t_a within the scope of this paper. We have already shown that $t_o \leq t_a$ for the case where all robots can directly communicate. It is simple to show that for the remaining cases the amount of data that needs to be routed is smaller with our approach than with a query-all approach. If not all robots can communicate directly, there is at least one robot a such that from the remaining robots one can form two non-empty robot sets B and C, such that all communication from a to C needs to go through at least one robot in B. This can be shown by contradiction: if no such robot a exists, this means that for every robot, no sets B and C fulfilling the above condition can be constructed. In other words, for every robot a' there exists no other robot c' such that the communication between a' and c' would need to be routed through at least one other robot b'. In that case, however, all robots can communicate directly, which contradicts the initial premise.

In our application, the robots in B are cumulatively responsible for forwarding $d^{a\to C} := d^{a\to j}\ \forall j \in C$. With the query-all approach, $d_a^{a\to C}$ is the full query originating in a, which needs to be sent to all robots, and thus $b_a(d^{a\to C}) = b^\star$. With our approach, $d_o^{a\to C}$ is only the portion of the $\frac{b^\star}{n}$-sized queries originating in a that are addressed to C, and thus $b_o(d^{a\to C}) = \frac{|C|}{n} b^\star$, which is strictly smaller than $b_a(d^{a\to C})$. We have thus shown that the amount of data that needs to be forwarded with our approach is less than with a query-all approach. We assume that in a multi-hop network, data forwarding can only increase the overall time needed to transmit the same data as in a fully connected work, and that this increase should correlate positively with a combination of $b(d^{a\to C})$ for different a, C. Since $b_o(d^{a\to C}) < b_a(d^{a\to C})\ \forall\ a, C$ we conclude that our approach should also be worthwhile in a multi-hop network.

VIII. Conclusion

In this paper, we have proposed a method that makes decentralized place recognition competitive compared to centralized place recognition in terms of the amount of data transmitted per robot count n, while only mildly affecting

recall. This is opposed to previous decentralized place recognition approaches, which, while generally maintaining perfect relative recall, typically use $n-1$ times more bandwidth than centralized place recognition. Unlike centralized place recognition, our decentralized method evenly distributes the workload among the robots involved. At the same time, it is deployable anywhere as long as the robots can communicate with each other. We have validated our approach for groups of up to 20 robots on the public datasets KITTI and Málaga and discussed what it would mean to deploy it in wireless networks.

REFERENCES

[1] D. Zou and P. Tan, "Coslam: Collaborative visual slam in dynamic environments," *IEEE transactions on pattern analysis and machine intelligence*, 2013.

[2] C. Forster, S. Lynen, L. Kneip, and D. Scaramuzza, "Collaborative monocular slam with multiple micro aerial vehicles," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2013.

[3] L. Riazuelo, J. Civera, and J. Montiel, "C2tam: A cloud framework for cooperative tracking and mapping," *Robotics and Autonomous Systems*, 2014.

[4] J. G. Morrison, D. Gálvez-López, and G. Sibley, "Moarslam: Multiple operator augmented rslam," in *Distributed Autonomous Robotic Systems*, 2016.

[5] M. Gadd and P. Newman, "Checkout my map: Version control for fleetwide visual localisation," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2016.

[6] V. Indelman, E. Nelson, J. Dong, N. Michael, and F. Dellaert, "Incremental distributed inference from arbitrary poses and unknown data association: Using collaborating robots to establish a common reference," *IEEE Control Systems*, 2016.

[7] A. Franchi, L. Freda, G. Oriolo, and M. Vendittelli, "The sensor-based random graph method for cooperative robot exploration," *IEEE/ASME Transactions on Mechatronics*, 2009.

[8] E. Montijano, R. Aragues, and C. Sagüés, "Distributed data association in robotic networks with cameras and limited communications," *IEEE Transactions on Robotics*, 2013.

[9] A. Cunningham, V. Indelman, and F. Dellaert, "Ddf-sam 2.0: Consistent distributed smoothing and mapping," in *IEEE Int. Conf. on Robotics and Automation*, 2013.

[10] R. Ji, L.-Y. Duan, J. Chen, L. Xie, H. Yao, and W. Gao, "Learning to distribute vocabulary indexing for scalable visual search," *IEEE Transactions on Multimedia*, 2013.

[11] X. Lin, Y. Shen, L. Cai, and R. Ji, "The distributed system for inverted multi-index visual retrieval," *Neurocomputing*, 2016.

[12] J. Sivic and A. Zisserman, "Video google: A text retrieval approach to object matching in videos," in *Int. Conf. on Computer Vision*, 2003.

[13] D. Nister and H. Stewenius, "Scalable recognition with a vocabulary tree," in *IEEE Conf. on Computer Vision and Pattern Recognition*, 2006.

[14] D. Galvez-Lopez and J. D. Tardos, "Bags of Binary Words for Fast Place Recognition in Image Sequences," *IEEE Transactions on Robotics*, 2012.

[15] R. Baeza-Yates, B. Ribeiro-Neto, *et al.*, *Modern information retrieval*. ACM press New York, 1999.

[16] A. Tomasic and H. Garcia-Molina, "Performance of inverted indices in distributed text document retrieval systems," 1992.

[17] B.-S. Jeong and E. Omiecinski, "Inverted file partitioning schemes in multiple disk systems," *IEEE Transactions on Parallel and Distributed Systems*, 1995.

[18] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for internet applications," *ACM SIGCOMM Computer Communication Review*, 2001.

[19] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, *A scalable content-addressable network*. ACM, 2001.

[20] S. Melink, S. Raghavan, B. Yang, and H. Garcia-Molina, "Building a distributed full-text index for the web," *ACM Transactions on Information Systems*, 2001.

[21] P. Reynolds and A. Vahdat, "Efficient peer-to-peer keyword searching," in *ACM/IFIP/USENIX Int. Conf. on Middleware*, 2003.

[22] T. Yan, D. Ganesan, and R. Manmatha, "Distributed image search in camera sensor networks," in *Proceedings of the 6th ACM Conf. on Embedded Network Sensor Systems*, 2008.

[23] M. Aly, M. Munich, and P. Perona, "Distributed kd-trees for retrieval from very large image collections," in *Proceedings of the British Machine Vision Conf. (BMVC)*, 2011.

[24] M. Bosse and R. Zlot, "Keypoint design and evaluation for place recognition in 2d lidar maps," *Robotics and Autonomous Systems*, 2009.

[25] S. Lynen, M. Bosse, P. Furgale, and R. Siegwart, "Placeless place-recognition," in *Int. Conf. on 3D Vision*, 2014.

[26] N. Sünderhauf, S. Shirazi, F. Dayoub, B. Upcroft, and M. Milford, "On the performance of convnet features for place recognition," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2015.

[27] R. Mur-Artal, J. Montiel, and J. D. Tardós, "Orb-slam: a versatile and accurate monocular slam system," *IEEE Transactions on Robotics*, 2015.

[28] P. Turcot and D. Lowe, "Better matching with fewer features: The selection of useful features in large database recognition problems," in *ICCV Workshop on Emergent Issues in Large Amounts of Visual Data (WS-LAVD)*, 2009.

[29] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *The Int. Journal of Robotics Research*, 2013.

[30] J.-L. Blanco, F.-A. Moreno, and J. González-Jiménez, "The málaga urban dataset: High-rate stereo and lidars in a realistic urban scenario," *The Int. Journal of Robotics Research*, 2014.

[31] J. Yick, B. Mukherjee, and D. Ghosal, "Wireless sensor network survey," *Computer networks*, 2008.

[32] D. Koutsonikolas, Y. C. Hu, and C.-C. Wang, "Pacifier: High-throughput, reliable multicast without crying babies in wireless mesh networks," *IEEE/ACM Transactions on Networking (TON)*, 2012.

[33] G. Di Caro and M. Dorigo, "AntNet: Distributed stigmergetic control for communications networks," *Journal of Artificial Intelligence Research*, 1998.

[34] N. Michael, M. M. Zavlanos, V. Kumar, and G. J. Pappas, "Maintaining connectivity in mobile robot networks," in *Experimental Robotics*, 2009.