IEICE TRANS. INF. & SYST., VOL.E93–D, NO.2 FEBRUARY 2010

PAPER

LSH-RANSAC: Incremental Matching of Large-Size Maps

Kanji TANAKA[†a)], ***Member***, **Ken-ichi SAEKI**[†], **Mamoru MINAMI**[†], ***and*** **Takeshi UEDA**[††], ***Nonmembers***

SUMMARY This paper presents a novel approach for robot localization using landmark maps. With recent progress in SLAM researches, it has become crucial for a robot to obtain and use large-size maps that are incrementally built by other mapper robots. Our localization approach successfully works with such incremental and large-size maps. In literature, RANSAC map-matching has been a promising approach for large-size maps. We extend the RANSAC map-matching so as to deal with incremental maps. We combine the incremental RANSAC with an incremental LSH database and develop a hybrid of the position-based and the appearance-based approaches. A series of experiments using radish dataset show promising results.

key words: *mobile robot, self-localization, incremental map-matching, RANSAC, LSH*

1. Introduction

Self-localization using landmark maps is an important task for mobile robot environments [1]-[3]. With recent progress in SLAM researches, it has become possible for a robot to obtain and use large-size maps that are incrementally built by other mapper robots [4]. For instance, such an online information sharing network has attracted much interest in recent years, in the context of networked robots, sensor networks as well as robot GIS. As a result, it has become crucial for a robot to estimate the self-position in real-time with respect to such *incremental maps*. A challenge is self-localization using *large-size maps*, where the increasing the number of self-position hypotheses as well as perceptual aliasing becomes a serious problem [1], [5], [6].

In this paper, we study the self-localization problem from the perspective of large-size and incremental maps. Self-localization using large-size maps has been studied in literature. However, most of them rely on a *batch assumption* that a complete map is a priori given (e.g. familiar environments) or fixed during the self-localization task (e.g. static environments). In such cases, it is straightforward to build the map structure in a batch manner using typical optimization techniques. On the other hand, we consider more general environments where the map may be built or incrementally updated even during the self-localization task. In such cases, the batch assumption is no longer relevant. To deal with the problem, we present a novel scheme for self-localization as well as map representation that successfully works with large-size and incremental maps.

Approaches to self-localization are broadly classified into appearance-based [6]–[8], position-based [1], [3], [9] and combinations of both [10]–[12]. Appearance-based approaches efficiently prune the robot pose hypotheses by using the appearance of observed landmarks as a cue. They evaluate the likelihood of a robot pose hypothesis by comparing appearance of an actually observed landmark and the appearance of a landmark in the map that should be observed at the hypothesized viewpoint. Position-based approaches verify each hypothesis by using the geometric constraints among observed landmarks as a cue. They evaluete the likelihood of a robot pose hypothesis by comparing the position-relationship of actually observed landmarks and prediction from the map. Both approaches play important roles in dealing with large numbers of hypotheses and constraints in large-size problems.

Appearance-based methods typically employ high dimensional descriptors of local features (e.g. SIFT [13], spin image [14], shape context [15]) to represent the appearance of landmark. For instance, 120-dim shape features from laser scanners will be considered in our experiments. Due to the high dimensionality, we have to employ efficient databases for information retrieval, for example, PCA with kd-tree [11], visual vocabulary with inverted file and words statistics [8], vocabulary with ANN and feature selection [6]. Unfortunately, most of existing databases are based on batch structures and thus require expensive preprocessing. To overcome, our map structures are based on hash tables. Unlike other structures such as trees, a hash table is essentially an incremental structure. More formally, a new element can be inserted by an incremental manner with an additional space/time cost bounded by a constant. In our previous work [16], we have developed incremental maps using an incremental locality sensitive hashing (iLSH) database [17].

Position-based methods seek a reliable coordinate transformation from the local map built by the map user robot to the global map built by mapper robots. When the size of the global map is large, the map is usually represented as a set of submaps. As a difficulty, the computational complexity grows exponentially with the number of landmarks in the maps. A standard solution to the difficulty is RANSAC map-matching. In [1], successful map-matching using large scale maps have been achieved

Manuscript received June 10, 2009.
Manuscript revised September 24, 2009.

[†]The authors are with the Faculty of Engineering, University of Fukui, Fukui-shi, 910–8507 Japan.
[††]The author is with the Faculty of Engineering, Kyushu University, Fukuoka-shi, 812–8581 Japan.
a) E-mail: tnkknj@u-fukui.ac.jp

DOI: 10.1587/transinf.E93.D.326

with purely position-based RANSAC by Neira, Tardós and Castellanos [1]. In [18], map-matching has been robust even when the ratio of outlier observations is high. Unfortunately, they are essentially batch algorithms and thus take as input fixed maps. To overcome, we pose the real-time localization problem as *incremental map-matching* problem. We have introduced the notation of incremental RANSAC (iRANSAC) in [19] and further developed with an efficient batch database in [20].

In this paper, we combine the advantages of the above two methods, iLSH and iRANSAC, and introduce a novel scheme called LSH-RANSAC [21]. This paper proposes a method for multi-robot large-size mapping problem using LSH-RANSAC. The method combines traditional methods: LSH, RANSAC and SLAM algorithms. The primary contribution of this paper is the development of a localization system that is fully incremental and scales to large-size environments. We will conduct experiments on the proposed scheme in large-size environments with over 10 mapper robots by using radish dataset [22].

Self-localization has been a central research area in robotics. It is difficult here to cover all the literature. Much effort has been focused on the accuracy issue of robot localization. One popular approach is to generate and track hypotheses of the robot pose over time in a pose tracking framework [3], [5], [10]–[12], [16], [23]. In particular, particle filter has received considerable attentions in recent years [10], [11], [23]. In [16], we also developed an efficient particle filter algorithm. Unfortunately, their computational cost grows linear to the size of pose space. The computational efficiency is an important issue of on-going researches [21].

In object recognition, the combination of LSH and RANSAC also has been recently studied for large-size problems [24]–[26]. There are important applications including image retrieval [24], epipolar geometry [25] and shape database [26]. Only a few use LSH as an incremental database [27]. Our contribution is incremental extension of LSH-RANSAC as well as its application to SLAM and localization problems.

In place recognition, visual dictionaries are used to interpret a high-dimensional feature to a 1D visual word [6]. Most of the vocabularies are conditioned on a specific feature type, parameter setting as well as training environments. Such conditional vocabularies increase the risk of overfitting/overgeneralization. Incremental building of a vocabulary is not a trivial problem [28], [29]. Our approach is rather similar with approaches in [30], [31] for object recognition, in a sense that we directly discretize the feature space without any dictionary.

In SLAM, localization using an incremental map is studied in the context of loop closure [32]. Most of them rely on prior knowledge (typically in the form of probability distribution) obtained from usual pose-tracking in SLAM. In such revisiting problems, the robot pose is often represented simply as a 1D space of visited place ID instead of the full pose space (e.g. 3D) considered in this paper.

In computer vision, RANSAC is a standard algorithm for robust estimation [33]. To speed up for large problems, a series of randomized RANSAC algorithms (rRANSAC) as well as the probabilistic variants and the batch optimization have been studied in recent years [25], [26], [34]–[36]. The objective of most rRANSAC (e.g. $T_{d,d}$ test [35]) is similar with original RANSAC, in a sense that they iterate the hypothesize-and-test until they find some reliable hypotheses. On the other hand, preemptive RANSAC (pRANSAC) proposed by Nister [36] deals with real-time applications where the computation time is always limited and typically constant. This property is appealing for the real-time problem of robot localization. Our iRANSAC is an incremental extension of the pRANSAC scheme.

2. Map-Matching Strategies

This section explains our basic strategies for map-matching in the RANSAC formulation. Let L denote a local map. The map is built by the target robot it-self with its motion measurements (e.g. odometry) and perceptual measurements (e.g. laser scan) using map-building techniques such as FastSLAM [37] as well as scan matching [38]. Let G denote a global map. The global map is a set of R metrical submaps $G = \{G_1, \cdots, G_R\}$ built by R mapper robots and shared via some information sharing network (typically wireless networks [4]). The maps represent the configuration of landmarks or features in the individual robot coordinates. The objective of map-matching is to identify which submap G_i the local map L corresponds to and to find a reliable coordinate-transformation ψ (i.e. translation and rotation) from the local map L to the global map G_i by which the two maps maximally overlap.

Figure 1 illustrates the notation of map-matching problem. In this figure, there are the target robot as well as several mapper robots exploring their own environments. Some of the robots may be located near to each other while some others far apart. The target and the other mapper robots respectively maintain so-called local map and global submaps. The position-relationship among the local map as well as global submaps are initially unknown.

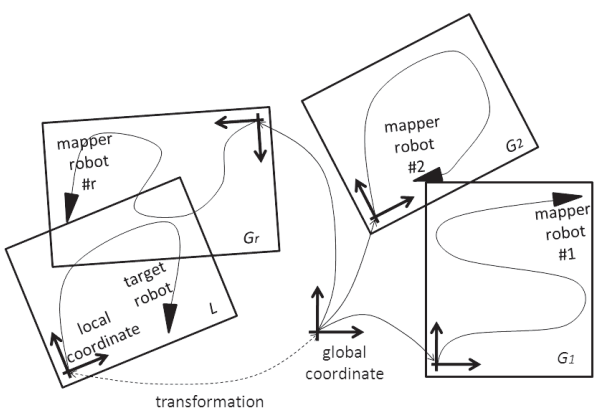


Fig. 1 Map matching problem.

2.1 RANSAC

The general RANSAC algorithm is described as follows. N denotes the number of datapoints. $U = \{x_i\}$ denote a set of N datapoints. $f_m : S \rightarrow P$ denotes a model function that computes the model parameters P from a sample set S of datapoints. $c(p, x)$ denotes the score function for a single datapoint x with model parameters p, and takes 1 if x is an inlier to the model or 0 otherwise. The objective is to find a "best" model parameter p^*, which should maximize the score function, with its associated cost value C^*. The RANSAC algorithm iterates the following steps until it finds some reliable hypotheses.

1. Select randomly a small subset $S_k \subset U$ of datapoints and compute a model parameter hypothesis $p_k = f_m(S_k)$.
2. Compute score $C_k = \sum_{x \in U} c(p_k, x)$.
3. If $C_k > C^*$ then $C^* \leftarrow C_k$, $p^* \leftarrow p_k$.

2.2 RANSAC in Map-Matching

It is straightforward to apply the above RANSAC algorithm to a *batch* map-matching problem. In the setting, we can interpret the above parameters as follows.

- N: the number of features on the local map L.
- x: a feature on the local map L.
- p: a transformation from the local map L to the global map G.
- $c(p, x)$: the score function. $c(p, x)$ takes 1 if a feature point x on the transformed map L^p matches a feature point on the global map G or 0 otherwise. Here, L^p is the local map rotated and translated according to the transformation p.
- f_m : the model function. f_m outputs a transformation candidate from the local map to a global map based on some local matches.

The function f_m can be viewed as a map retrieval function. Given a local feature as a query, it searches similar features from global submaps. For each retrieved feature, it outputs a transformation from the local feature to the retrieved feature.

2.3 Preemptive RANSAC Scheme

The random sampling in naive RANSAC generates a large number of useless hypotheses. The preemptive RANSAC (pRANSAC) scheme aims to avoid excessive scoring of such useless hypotheses. It maintains the history $H = \{(p_j, x_j, C_j)\}$ of every feature-hypothesis pair (p_j, x_j) scored so far and its corresponding scoring result $C_j = c(p_j, x_j)$. It employs two user-defined rules, order rule and preference rule. The order rule

$$(p, x) = f_o(H) \tag{1}$$

decides which pair should be scored next according to the history H of scoring results. The preference rule

$$p^* = f_p(H) \tag{2}$$

decides which hypothesis is best according to H. Suppose we have been given a set of features

$$U = \{x_i\} \tag{3}$$

and already generated a set of hypotheses

$$V = \{p_j\}. \tag{4}$$

pRANSAC iterates the following steps until the computation time is exhausted:

1. Select a feature-hypothesis pair $(p_k, x_k) = f_o(H)$ and compute the score $c(p_k, x_k)$,
2. Update the history H incorporating the scoring results.

If necessary, it outputs the best hypothesis $p^* = f_p(H)$.

2.4 Preemptive Breadth-First Rule

The performance of pRANSAC depends strongly on the order rule $f_o(H)$. The preemptive breadth-first order rule proposed by Nister has some appealing properties [36]. Firstly, its computation time is bounded by a constant proportional to the size of hypothesis set. Secondly, it compares the hypotheses against each other, rather than against some absolute quality measure (which is often sub-optimal or unavailable). Thirdly, its stability is analyzed using inlier-outlier model [36].

To save the computation time, it limits the number of hypotheses to be considered ("active hypotheses") using a decreasing preemption function

$$f_b(i). \tag{5}$$

As a preparation, it randomly permutes the sequence of feature ids $1, \cdots, N$ as well as the sequence of hypotheses ids $1, \cdots, M$ and initializes the score $C_j = 0$ for every hypothesis $j (1 \le j \le M)$. After the preparation, it performs the following steps for each iteration i until the computation time is exhausted:

1. Compute the scores $C_j \leftarrow C_j + (p_j, x_i)$ for each hypothesis $j (1 \le j \le f_b(i))$,
2. Reorder the hypothesis ids $1, \cdots, M$ so that the range $1, \cdots, f_b(i)$ contains the best $f_b(i)$ active hypotheses according to the accumulated score C_j.

The preemption function $f_b(i)$ is in the form:

$$f_b(i) = \lfloor M 2^{-\lfloor \frac{i}{B} \rfloor} \rfloor \tag{6}$$

where $\lfloor \cdot \rfloor$ denotes downward truncation and B is a preset constant called "block size" [36]. Note that the size of hypothesis set reduces to the half every B^{th} iteration. This results in an approximately $O(BM)$ time cost.

3. Incremental Map-Matching Algorithm

We now turn to more general case of *incremental* map-matching tasks. The main process of incremental map-matching is initialized only once at the beginning of the localization task at the start viewpoint. After that, it continues to search for answers using latest maps. There are two types of events by which the maps are modified. The first event is arrival of new features into the global map. In the event, the new features are added to the corresponding submap, then each of the new features is inserted into the appearance database. Section 3.1 will describe the details of the appearance database. The second event is arrival of new features into the local map. In the event, the new features are added to the feature set, and new hypotheses are generated by retrieving the appearance database using each of the new feature as a query, then added to the hypothesis set. Section 3.2 will describe the details of the map-matching algorithm.

Figure 2 illustrates the incremental map-matching system. This system is incremental in such a sense that the target and the mapper robots are incrementally updating the local and the global submaps. Our method queries the appearance database to search relevant transformation hypotheses as well as matches the geometric constraints among landmarks to verify each hypothesis.

3.1 Appearance Database

The appearance database employs an approximate near neighbor technique called locality sensitive hashing (LSH) [39]. LSH uses a continuous representation of datapoint, and searches points that are near from the query point in l_2 space. More formally, the E^2LSH addresses an (R, cR)-Near Neighbor problem, where the goal is to report a point within distance cR from a query q, if there is a database point within distance R from q [40].

Unlike other structures such as trees, a structure based on hash tables has strong advantages in an incremental setting. The database element can be frequently added to or deleted from hash tables. This strongly motivates us to use LSH as a basis of the incremental database. In the preprocessing of E2LSH, L hash functions with K dimensions are probabilistically generated using a p-stable distribution, in our case, the normal distribution. When a new feature arrives in global maps, the database is updated in the following procedure:

- Memorizes the real-world location of the feature,
- Hashes the feature using the E^2LSH function and accesses the corresponding bins,
- Associates the real-world location of the feature to the accessed bins.

Loosely following [41], we do not memorize the high-dimensional features themselves (but only the hash codes) and also not use them in the retrieval task. This is a simple modification but quite effectively reduces the space and the time costs of a database [39]. In [16], we have applied this technique to Monte Carlo localization and reported its performance. With this modification, the above process for updating the database (i.e. learning and dimensionality reduction) can be performed in an incremental manner.

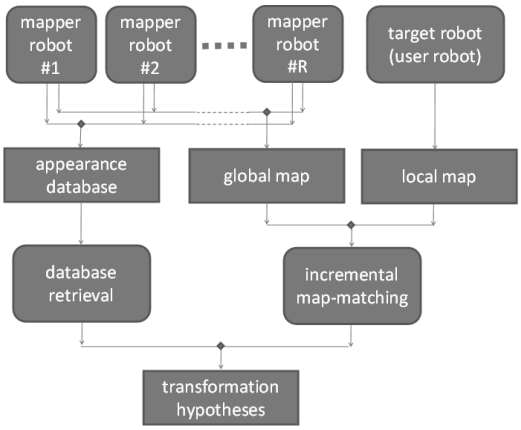


Fig. 2 Incremental map-matching system.

We also adapt the *hash maps* [16]. In conventional systems, the real-world location of features are commonly memorized on grid maps. In large-size environments, naive implementation of a grid map consumes a significant amount of memory, in proportional to the number of grid cells. To overcome, we adapt universal hash (UH) tables and map the real-world space (e.g. 3D space) onto 1D hash-code space. The size of hash table should be determined so that the probability of collision is sufficiently low, taking into account the spatial density of features. In consequence, the space cost of such an approximated grid map can be practically low, for example, 3% of the original grid map in our experimental settings.

3.2 Map-Matching Algorithm

The iRANSAC algorithm is summarized as follows. At the beginning of the task, it initializes the feature and the hypothesis sets to empty sets $U \leftarrow \phi$, $V \leftarrow \phi$. During the task, it iterates the following steps:

1) When new features U^{new} arrive, modify the feature set
$$U \leftarrow U^{new} \cup U, \tag{7}$$
2) When new hypotheses V^{new} arrive, modify the hypothesis set
$$V \leftarrow V^{new} \cup V, \tag{8}$$
3) Perform the steps 1,2 of pRANSAC.

The modified breadth-first rule performs the following steps for each iteration i:

1) Perform the steps 1,2 of the iRANSAC algorithm,
2) Perform the steps 1,2 of the original breadth-first rule.

The preemption function $f_b(i)$ is in the form:

$$f_b(i) = \begin{cases} m(i-1)/2 & (i \bmod B = 0) \\ m(i-1) & (i \bmod B \neq 0) \end{cases}, \quad (9)$$

where $m(i)$ denotes the number of hypotheses at the end of i-th iteration.

We have to slightly modify the preference rule. In the incremental setting, the number N_j of scored can be very different among individual hypotheses. It tends to be smaller for younger hypotheses than older ones. For fair comparison, we use a normalized measure to compare among the hypotheses

$$C_j/N_j \quad (10)$$

in place of the conventional measure C_j. In addition, we do not consider a young hypothesis as a candidate of the best hypothesis when its score is smaller than a threshold C_o.

The modified breadth-first rule degenerates to the original breadth-first rule in a special case where the hypothesis set is fixed. It is known that the performance of rRANSAC (including pRANSAC) is maximized when a randomly permuted sequence is used [36]. However, it is no longer possible to permute the hypothesis set beforehand in the incremental setting. The process of random permutation of a set U (or V) is implemented as an $O(|U^{new}|)$ cost process of inserting each element in U^{new} at a random point in U given U has been already permuted. The actual performance of our scheme will be evaluated in the experimental section.

4. Experiments

We evaluate our approach through robot localization experiments using radish dataset [22]. For each dataset, there are sequences of motion and perception measurements. Each motion measurement indicates the robot's movements from one viewpoint to the next and represented in a forward-sideward-rotate (FSR) format in our system. Each perception measurement is a single scan by the front laser scanner and represented by a set of 180 datapoints in a robot centric coordinate. As appearance feature, shape feature is usually used in the case of laser datapoints [24]. In particular, we use a shape feature called generalized shape context (GSC) [15] which is found to be stable and useful for robot localization applications in our previous experiments [16], [20]. We use a simple scan matching algorithm as a technique for SLAM i.e. local mapping and pose-tracking. The number of new features and new hypotheses are set to $|U^{new}| = 10$ and $|V^{new}| = 1,000$. The block size used for preemption is simply set as $B = |U^{new}|$. The parameters for LSH are set empirically to $K = 30$ and $L = 20$. We later investigate the sensitivity of our system against these parameters.

We consider a typical scenario where R mapper robots are exploring R different buildings. The buildings shown in the maps in Fig. 3 (a)-(c) are three different environments where a series of localization tasks take place. The solid and the dotted curves respectively are the trajectories of the target robot and the mapper robots. Shown in Fig. 3 (d) are the

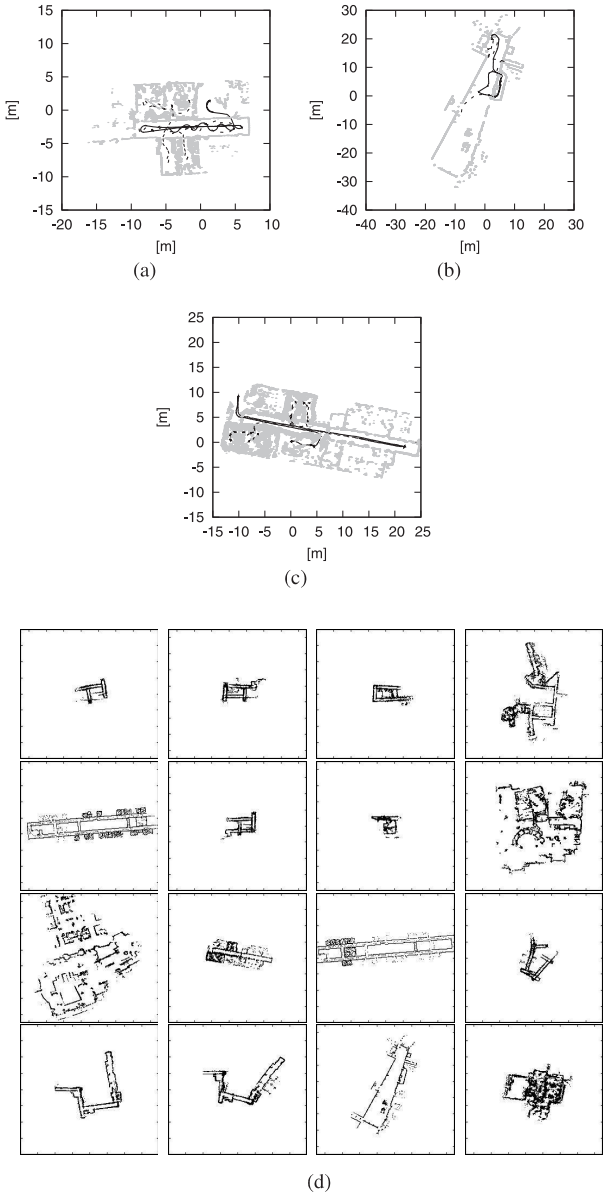


Fig. 3 Maps generated and used in experiments.

16 buildings where the 16 mapper robots are exploring in the case of Fig. 3 (a). Among these buildings, the building at third column and second row is the one where the target robot locates. No a priori knowledge is given on the buildings as well as on which buildings the individual robots locate. Each submap G_r $(1 \leq r \leq R)$ is initialized to an empty map at the beginning of the map building task. Then, it is incrementally built by the corresponding mapper robot during the task. Every time a novel scan arrives, a set of appearance features are extracted from the scan points. Then, the grid maps as well as the appearance database are updated with the extracted features. Figure 4 (a) shows the time consumed by this updating process per feature for different size global maps for three different environments "albert", "fr079" and "fr101". It can be seen that the required time is independent

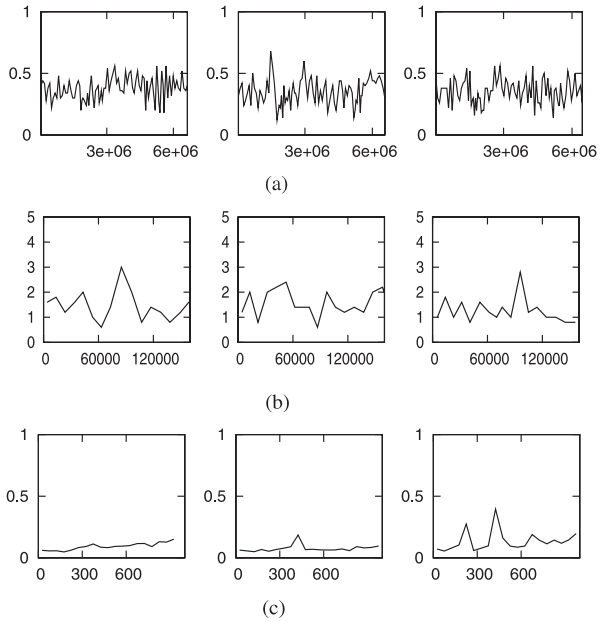


Fig. 4 Time costs [msec] for (a) database updating (feature ID vs. cost per feature), (b) database retrieval (feature ID vs. cost per feature) and (c) RANSAC map-matching (viewpoint ID vs. cost per feature-hypothesis pair) for three different environments, "albert", "fr079" and "fr101" from left to right panels.

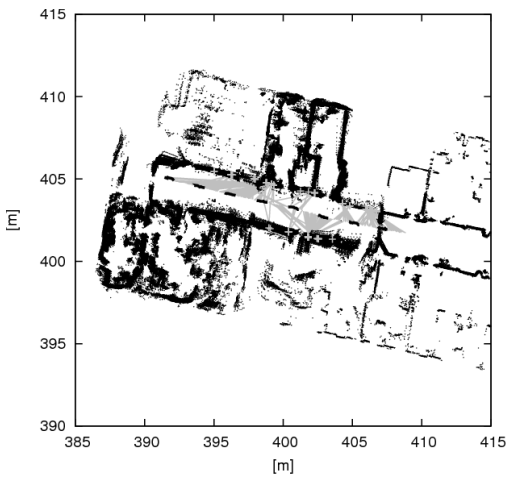


Fig. 5 Estimated trajectory.

of the map size and bounded by a constant. As discussed in Sect. 3.1, the naive implementation of a submap consumes a significant amount of space. In our case, 2.1×10^7 cells in the 3D $xy\theta$ pose space and the global map consumes around $3.4{\times}10^8$ cells in total. With the modified implementation using UH tables, the space costs are reduced down to 1×10^7 bins. The size of UH table is less than 3% of the original grid map. The LSH table consumed by the LSH database is at most 7.2×10^6 bins in this experiment. It could be said that our structure is fully incremental and scales to large-size maps.

Figures 5, 6 illustrate localization tasks. In Fig. 5, the dots are datapoints obtained from the laser scanner and

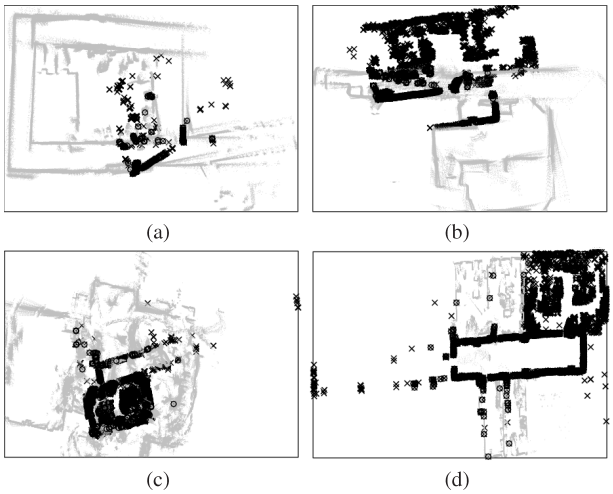


Fig. 6 Evolution of map-matching as we get more viewpoints from the target robot.

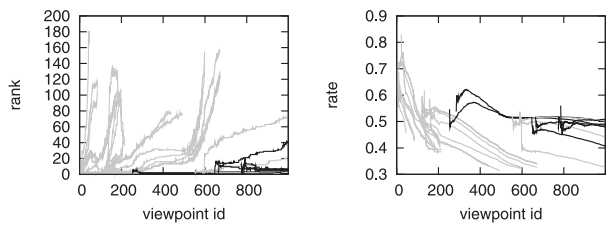


Fig. 7 Scoring results. The black and gray curves are correct and wrong hypotheses.

the thick dashed lines are the robot trajectory. In Fig. 6, the gray dots are datapoints in the global submap while the black 'o' and 'x' respectively are inlier and outlier points in the local map. At the beginning of the localization task, the local map is initialized to an empty map. Every time a novel scan arrives, the local map is modified incorporating the scan points. Then, a set of appearance features are extracted from the scan points. Then, the appearance database is retrieved using each feature as a query. Then, new hypotheses are generated based on the retrieval results. The time consumed by the retrieval process is actually bounded by a constant as explained in the previous section and as shown in Fig. 4 (b). The results shown in Fig. 6 (a),(b) and (c) include two typical cases where the map-matching is not yet successful. The first case is that the local map is not yet informative. The second case is that most part of the local map is originated from *unknown* region that is not described in the global map. As a result, all the matching results are caused by false matches, for example, the local map is matched with a wrong building. Figure 6 (d) shows that the matching finally becomes successful. In the case, the local map now grows informative and reliable. Sufficiently large part of the map is originated from known region in the global map. Figure 4 (c) shows that the time consumed by the map-matching process is also bounded by a constant. The time cost is independent of the size of map database.

Figure 7 illustrates scoring results. In the figure, "rank"

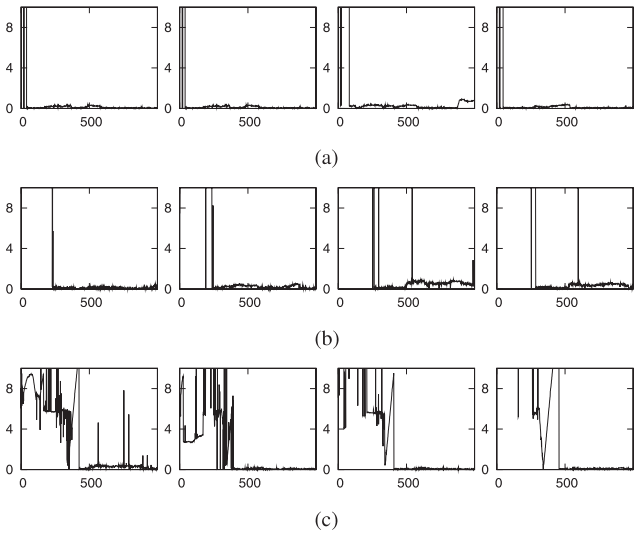


Fig. 8 Localization errors [m] (y-axis) when the amount of measurements or iterations (x-axis) increases, for (a) "fr101", (b) "albert" and (c) "fr079". The panels from left to right correspond to 1, 4, 9 and 16 submaps.

is a relative measure that indicates the height of score of one hypothesis against others. For the sake of clarity, only those hypotheses that are top-ranked (i.e. assigned highest rank) at least once are shown in the figure. The scoring by iRANSAC is performed in a preemptive manner similar with other rRANSAC schemes such as [26] and [36]. Good hypotheses supported by many datapoints tend to be efficiently detected. Bad hypotheses contaminated by noises tend to be quickly weeded out. Once a good hypothesis is high-ranked it tends to stay in the high-rank group for a long period of time. A key difference from typical rRANSAC schemes is that the rate (or the ranking) of a hypothesis tend to decrease (or increase) as the robot navigates. This is because of that old hypotheses tend to be inconsistent with new datapoints due to the accumulation of errors in the odometry as well as in the local map. In consequence, old and new hypotheses are compared against each other in a preemptive manner and the top-ranked hypothesis supported by many recent datapoints tends to be output as a best hypothesis at each viewpoint.

Figure 8 reports the localization performance in several experiments. A series of localization tasks are conducted in 12 different scenarios, for three different local maps (corresponding to "albert", "fr079", "fr101") shown in Fig. 3 (a)-(c), and for four different size global maps respectively composed of 1, 4, 9 and 16 submaps. The target robot is assumed to be located in the same building as one of the mapper robots. A pair of non-overlapping measurement sequences can be created from a single dataset with a procedure described in [16] and respectively assigned to the target robot and the mapper robot located in the same building. In all the cases of Fig. 8 (a)-(c), it can be seen that localization errors finally become small, less than 1 m. The global map size 16 corresponds to over 3.5×10^5 appearance features. In the Figure 8, it can be seen that the localization task is successful even for 16 submaps environments. This is a large-size map compared with previously published works on fully incremental systems. It is noted that the proposed incremental scheme is successful even for such a large-size map. We also conducted additional experiments to investigate the parameter sensitivity and the scalability of our scheme. Figure 9 summarizes the sensitivity against LSH parameters. It can be seen that our system is stable for a wide range of parameters. We also conducted experiments with much larger global maps using synthesized datasets, and found that our scheme is successful as much as 40 submaps (1.2×10^6 features) environments. From above results, it is concluded that the proposed scheme is fully incremental and scales to large-size problems.

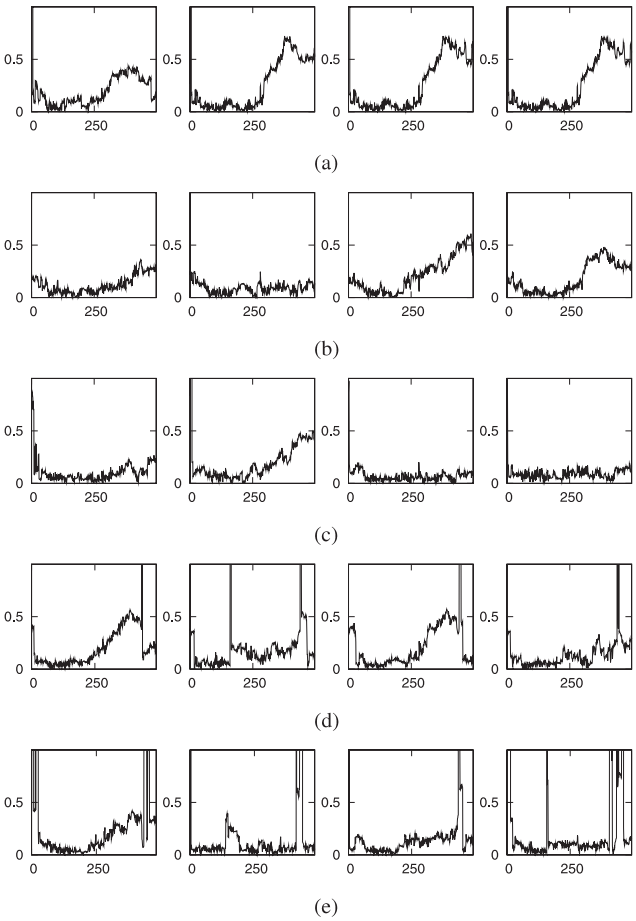


Fig. 9 Comparison of localization performance for (a) K=10, (b) K=20, (c) K=30, (d) K=40 and (e) K=50 for L=5, L=10, L=20 and L=30 from left to right panels.

5. Conclusion

We have studied the problem of robot localization from the perspective of incremental map-matching. In contrast to most map-matching approaches, our approach does not assume fixed maps but considers more general cases where the maps are incrementally modified during the task. We based our system on LSH appearance database and

RANSAC map-matching and developed an incremental extension of the LSH-RANSAC scheme. In experiments, we have demonstrated that the system is efficient even with large-size maps that are incrementally built by other mapper robots. In future, we plan to apply the proposed scheme to some other platforms including robot GIS as well as vision-guided mobile robots.

Acknowledgments

This work was partially supported by the Japan Ministry of Education, Science, Sports and Culture, Grant in-Aid for Young Scientists (B), 17700200 (2005-2006) and 19700192 (2007-2008), and by Suzuki Foundation Research Grant (2006), and by Electro Mechanic Technology Advancing Foundation (2007-2008).

References

[1] J. Neira, J.D. Tardos, and J.A. Castellanos, "Linear time vehicle relocation in slam," IEEE Int. Conf. Robotics and Automation, vol.1, pp.427–433, 2003.

[2] K. Ho and P. Newman, "Multiple map intersection detection using visual appearance," 3rd International Conference on Computational Intelligence, Robotics and Autonomous Systems, 2005.

[3] F. Dellaert, D. Fox, W. Burgard, and S. Thrun, "Monte carlo localization for mobile robots," IEEE International Conference on Robotics and Automation (ICRA), 1999.

[4] L.A.A. Andersson and J. Nygards, "C-sam: Multi-robot slam using square root information smoothing," IEEE Int. Conf. Robotics and Automation (ICRA), pp.2798–2805, 2008.

[5] R. Kummerle, R. Triebel, P. Pfaff, and W. Burgard, "Monte carlo localization in outdoor terrains using multilevel surface maps," Journal of Field Robotics, vol.25, pp.346–359, 2008.

[6] G. Schindler, M. Brown, and R. Szeliski, "City-scale location recognition," IEEE Conference on Computer Vision and Pattern Recognition, pp.1–7, 2007.

[7] A.C. Murillo, J.J. Guerrero, and C. Sagues, "Surf features for efficient robot localization with omnidirectional images," IEEE International Conference on Robotics and Automation, pp.3901–3907, 2007.

[8] J. Wang, H. Zha, and R. Cipolla, "Coarse-to-fine vision-based localization by indexing scale-invariant features," IEEE Trans. Syst. Man Cybern., vol.36, no.2, pp.413–422, 2006.

[9] L.M. Paz, P. Piniés, J. Neira, and J.D. Tardós, "Global localization in slam in bilinear time," Proc. 2005 IEEE/RSJ Int. Conf. Intelligent Robots and Systems, pp.655–661, 2005.

[10] A. Gil, O. Reinoso, A. Vicente, C. Fernandez, and L. Paya, "Monte carlo localization using sift features," Lect. Notes Comput. Sci. (LNCS), vol.1, no.3523, pp.623–630, 2005.

[11] N.A, Vlassis, B. Terwijn, and B.J.A. Kroese, "Auxiliary particle filter robot localization from high-dimensional sensor observations," Proc. IEEE Int. Conf. Robotics and Automation (ICRA), 2002.

[12] J. Wolf, W. Burgard, and H. Burkhardt, "Robust vision-based localization by combining an image retrieval system with monte carlo localization," IEEE Trans. Robotics, vol.21, no.2, pp.208–216, 2005.

[13] J.J. Foo and R. Sinha, "Pruning sift for scalable near-duplicate image matching," Proc. Int. Conf. ACM, pp.63–71, 2007.

[14] A. Johnson, Spin-Images: A Representation for 3-D Surface Matching, PhD thesis, Carnegie Mellon University, 1997.

[15] G. Mori, S. Belongie, and J. Malik, "Shape contexts enable efficient retrieval of similar shapes," Proc. Int. Conf. IEEE Comput. Vis. Pattern Recognit., vol.1, pp.723–730, 2001.

[16] K. Tanaka and K. Eiji, "A scalable algorithm for monte carlo localization using an incremental e2lsh-database of high dimensional features," IEEE Int. Conf. Robotics and Automation, pp.2784–2791, 2008.

[17] A. Gionis, P. Indyk, and R. Motwani, "Similarity search in high dimensions via hashing," Proc. 25th Very Large Database (VLDB) Conference, 1999.

[18] B. Yamauchi and P. Langley, "Place recognition in dynamic environments," J. Robotic Systems, vol.14, pp.107–110, 1997.

[19] K. Tanaka and E. Kondo, "Incremental ransac for online vehicle relocation in large dynamic environments," Proc. IEEE Int. Conf. Robotics and Automation (ICRA), pp.1025–1030, 2006.

[20] T. Ueda and K. Tanaka, "On the scalability of robot localization using high-dimensional features," IAPR International conference on pattern recognition, 2008.

[21] K. Saeki, K. Tanaka, and T. Ueda, "Lsh-ransac: An incremental scheme for scalable localization," IEEE Int. Conf. Robotics and Automation, 2009.

[22] A. Howard and N. Roy, The robotics data set repository (radish), 2003.

[23] F. Linaker and M. Ishikawa, "Real-time appearance-based monte carlo localization," Robotics and Autonomous Systems, vol.54, no.3, pp.205–220, 2006.

[24] B. Matei, Y. Shan, H.S. Sawhney, Y. Tan, R. Kumar, D. Huber, and M. Hebert, "Rapid object indexing using locality sensitive hashing and joint 3d-signature space estimation," IEEE Trans. Pattern Anal. Mach. Intell., vol.28, no.7, pp.1111–1126, 2006.

[25] G. Liran and S. Ilan, "Balanced exploration and exploitation model search for efficient epipolar geometry estimation," IEEE Trans. Pattern Anal. Mach. Intell., vol.30, no.7, pp.1230–1242, 2008.

[26] O. Chum and J. Matas, "Optimal randomized ransac," IEEE Trans. Pattern Anal. Mach. Intell., vol.30, no.8, pp.1472–1482, 2008.

[27] M. Yang, J. Yuan, and W. Ying, "Spatial Selection for Attentional Visual Tracking," Proc. Int. Conf. IEEE Comput. Vis. Pattern Recognit., vol.1, pp.1–8, 2007.

[28] T. Yeh, J. Lee, and T. Darrell, "Adaptive vocabulary forests br dynamic indexing and category learning," IEEE International Conference on Computer Vision, 2007.

[29] Z. Zhu, T. Oskiper, S. Samarasekera, R. Kumar, and H.S. Sawhney, "Ten-fold improvement in visual odometry using landmark matching," IEEE International Conference on Computer Vision, 2007.

[30] T. Tuytelaars, C. Schmid, and L.K.U. Leuven, "Vector quantizing feature space with a regular lattice," Proc. Int. Conf. IEEE Comput. Vis. Pattern Recognit., vol.1, pp.1–8, 2007.

[31] F. Moosmann, E. Nowak, and F. Jurie, "Randomized clustering forests for image classification," IEEE Trans. Pattern Anal. Mach. Intell., vol.30, no.9, pp.1632–1646, 2008.

[32] I. Ulrich and I. Nourbakhsh, "Appearance-based place recognition for topological localization," IEEE International Conference on Robotics and Automation, 2000.

[33] 25 years of ransac, Proc. IEEE Int. Workshop in Conjunction with CVPR '06, 2006.

[34] O. Chum and J. Matas, "Matching with prosac progressive sample consensus," IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp.220–226, 2005.

[35] O. Chum and J. Matas, "Randomized ransac with $t_{d,d}$ test," Proc. British Machine Vision Conference, pp.448–457, 2002.

[36] D. Nister, "Preemptive ransac for live structure and motion estimation," Machine Vision and Applications, vol.16, pp.321–329, 2005.

[37] M. Montemerlo, FastSLAM: A Factored Solution to the Simultaneous Localization and Mapping Problem with Unknown Data Association, PhD thesis, Carnegie Mellon University, 2003.

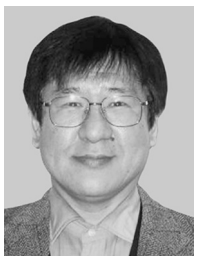
[38] J. Nieto, T. Bailey, and E. Nebot, "Recursive scan-matching slam," Robotics and Autonomous Systems, vol.55, pp.39–49, 2007.

[39] A. Andoni, M. Datar, N. Immorlica, P. Indyk, and V. Mirrokni, "Locality-sensitive hashing scheme based on p-stable distributions," Nearest Neighbor Methods in Learning and Vision: Theory and

Practice, 2006.
[40] M. Datar, N. Immorlica, P. Indyk, and V.S. Mirrokni, "Locality sensitive hashing scheme based on p-stable distributions," Proc. Symposium on Computational Geometry, pp.253–262, 2004.
[41] D. Nister and H. Stewenius, "Scalable recognition with a vocabulary tree," IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp.2161–2168, 2006.

Kanji Tanaka received the MS and Ph.D. degrees in the graduate school of information science and electrical engineering, Kyushu university, in 1997 and 2000, respectively. He became a research assistant in the school of systems information science in 2000, Future University-Hakodate. He became a research associate at the department of intelligent machinery and systems, Kyushu university, in 2002. Since 2008, he has been an associate professor at the graduate school of engineering, university of Fukui. His current research interests are intelligent robotics, machine learning and pattern recognition. He is a member of IEEE, SICE, JSME and RSJ.

Ken-ichi Saeki received the BS degree in engineering from university of Fukui in 2008. He is currently working toward the MS degree in the graduate school of engineering, university of Fukui. His research interest include mobile robotics and computer vision.



Mamoru Minami completed the M.E. program in aeronautical engineering at University of Osaka Prefecture in 1981, and the course in natural science in 1993 at the Graduate School of Kanazawa University, receiving a D.Eng. degree. He became an associate professor in 1994 in the Department of Mechanical Engineering, and a professor in 2002 in the Department of Human and Artificial Intelligence Systems, Faculty of Engineering, University of Fukui. He is engaged in research on the control of mobile manipulators, dynamic image recognition, visual servoing, and force control. He is member of the Japan Society of Mechanical Engineers, the Robotics Society of Japan, SICE, and IEEE.

Takeshi Ueda received the BS degree in engineering from Kyushu university in 2007. He is currently working toward the MS degree in the graduate school of engineering, Kyushu university. His research interest include mobile robotics and machine learning.