

Cloud-based map alignment strategies for multi-robot FastSLAM 2.0

Shimaa S Ali , Abdallah Hammad and Adly S Tag Eldien

Abstract

The cooperative simultaneous localization and mapping problem has acquired growing attention over the years. Even though mapping of very large environments is theoretically quicker than a single robot simultaneous localization and mapping, it has several additional challenges such as the map alignment and the merging processes, network latency, administering various coordinate systems and assuring synchronized and updated data from all robots and also it demands massive computation. This article proposes an efficient architecture for cloud-based cooperative simultaneous localization and mapping to parallelize its complex steps via the multiprocessor (computing nodes) and free the robots from all of the computation efforts. Furthermore, this work improves the map alignment part using hybrid combination strategies, random sample consensus, and inter-robot observations to exploit fully their advantages. The results show that the proposed approach increases mapping performance with less response time.

Keywords

Cloud, random sample consensus, simultaneous localization and mapping, Hadoop, Map/Reduce

Date received: 17 February 2018; accepted: 21 December 2018

Handling Editor: Antonio Puliafito

Introduction

Nowadays, cooperative multi-robot systems (CMRSs)¹ play an important role in implementing complex tasks such as mapping, exploration, rescue, and search operations. Constructing a team of simple robots may be faster and more reliable than a single complex robot system. There are many challenges in the field of multi-agent systems: communication techniques, decision-making, a planning mechanism, and integration of partial maps from the different robots (combining the information gathered by multiple robots). One of the major requirements for achieving completely autonomous mobile robots in an unknown environment is to implement simultaneous localization and mapping (SLAM),² which enables the robots to build a map of their surroundings and, at the same time, localize themselves in it. Mobile robots can cooperate with each other to solve the SLAM problem—known as cooperative SLAM. This article focuses on the issues of cooperative SLAM, particularly the map-merging part.

Map merging is critical for maximizing efficiency and accuracy of the multi-robot SLAM, which has been receiving further attention in the literature. Every robot builds a local map of its surrounding, but the important question is how to combine these local maps into a single global one. The merging process demands calculation of a transformation matrix for converting all the local maps to the same global reference frame—referred to as map alignment. There are several approaches to align the different maps such as data association and inter-robot observation methods. Nossair et al.³ shows that these methods do not work efficiently; therefore,

Department of Electrical Engineering, Faculty of Engineering at Shoubra, Benha University, Benha, Egypt

Corresponding author:

Shimaa S Ali, Department of Electrical Engineering, Faculty of Engineering at Shoubra, Benha University, 108 Shoubra Street, Benha, Cairo 11629, Egypt.
Email: shimaa.salama@feng.bu.edu.eg



Creative Commons CC BY: This article is distributed under the terms of the Creative Commons Attribution 4.0 License

(<http://www.creativecommons.org/licenses/by/4.0/>) which permits any use, reproduction and distribution of the work without

further permission provided the original work is attributed as specified on the SAGE and Open Access pages (<https://us.sagepub.com/en-us/nam/open-access-at-sage>).

they have the constraints in cases of different states of merging process. As a result, this article proposes an innovative approach for performing cooperative SLAM, maximizing efficiency of the mapping, and minimizing the costs. Therefore, according to the case of current merging, the algorithm decides if either of the two methods (data association or inter-robot observations) is more robust to execute alignment tasks. Due to a massive exchange of data between the robots, which requires increasing processing, the traditional processing may no longer be practical (effective). As a consequence, this article proposes to distribute all of the computation load of cooperative SLAM to systems running in the cloud⁴—called CCSLAM. The addition of this technology provides further reduction of the execution time of tasks of multi-robots with low cost by parallelizing the intensive computations via the cloud-computing servers.

The structure of this article is organized as follows. Section “Related work” discusses an overview of related work. Section “Background” discusses the two alignment methods. In section “Proposed method—cooperative SLAM as service in the cloud,” the proposed method is described in detail. Experimental results are presented in section “Experimental results,” which demonstrates the efficiency and accuracy of the presented approach. The last section concludes this article.

Related work

Cooperative SLAM is one of the most researched subjects in the robotics field. Wanick et al.⁵ used the particle filter algorithm⁶ for cooperatively solving the SLAM problem by two small mobile robots via only one remote computer. However, their works do not show one of the biggest challenges of multi-robot SLAM, that is, combining the information collected by multiple robots. Yan et al.⁷ proposes issues of multiple mobile robot systems (MMRSs) such as communication technique, a planning mechanism, and a decision structure. Benavidez et al.⁸ implement the visual simultaneous localization and mapping (VSLAM) algorithm for single robot running in the cloud-distributed processing, thus this algorithm is divided into two stages. The first stage is to make a landmark database. The second stage is to utilize the landmark database to determine the pose of the robot. Kamburugamuve et al.⁹ implements a particle filtering-based SLAM algorithm using the Internet of things (IoT) Cloud framework.¹⁰

Arumugam et al.¹¹ proposed a cloud framework called Distributed Agents with Collective Intelligence (DAvinCi). Thus, Hadoop framework is used, which consists eight Intel quad-core server nodes for service robots in large environments. Riazuelo et al.¹² present

a real-time performance of cooperative SLAM using two RGBD cameras via a two-nodes cloud. Thus, the mobile robots can build their local maps and integrate them together. If an overlap is detected, then the duplicated points between these maps are deleted. The contribution of the proposed approach is that it allows the combination of the partial landmark-based maps in multi-robot systems according to the state of the environment until the case of disjoint partial maps. In addition, the incoming map of the other robot is merged simultaneously with another map. If the landmarks are previously observed, the other robot’s estimation is considered as evidence of the best map merging with lowest amount of error.

Mohanarajah et al.¹³ present a platform for cooperative building and map-merging processes from two robots via the cloud framework. The merging task is executed using the random sample consensus (RANSAC) algorithm while the matching key-frames of the two maps are found. After map merging, the robots are re-localized in the updated map. The essential contributions of this article are as follows:

1. A novel solution of cooperative SLAM based on hybrid integration of two map alignment techniques to merge the local maps with highest quality and efficiency.
2. Parallel mapping and merging mechanism are implemented via multiple nodes.
3. Multi Map/Reduce jobs are used to build large environments by four inexpensive robots as a service in the proposed cloud platform; the robots are freed from all intensive computational processes.

Background

In order to create a global map from the partial maps, the first step is a calculation of the coordinate transformation for a pair of the correspondence points for one reference frame, which consists of three alignment parameters: translation in x , y (d_x , d_y), and rotation θ . The transformation can be computed using the estimated points and their z-signatures using the RANSAC algorithm¹⁴ or inter robot–robot observations.¹⁵ After calculation of the transformation matrix T as in equation (1),¹⁶ the maps are transformed using equation (2)¹⁷ for the same reference of the robot

$$T = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & d_x \\ \sin(\theta) & \cos(\theta) & d_y \\ 0 & 0 & 1 \end{bmatrix} \quad (1)$$

$$\begin{bmatrix} x'_i \\ y'_i \\ 1 \end{bmatrix} = T \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} \quad (2)$$

Alignment using RANSAC

RANSAC is one of the most common approaches to align landmark-based maps using data association, which depends on creating a list of possible matches between the landmarks of the partial map and their correspondences in another partial map. The two pairs of correspondences are used to compute the alignment parameters (d_x, d_y, θ) with the following equations¹⁸

$$d_x = x_i - x'_i \cos \theta - y'_i \sin \theta \quad (3)$$

$$d_y = y_i - y'_i \cos \theta + x'_i \sin \theta \quad (4)$$

$$d_2 = d_x^2 + d_y^2 \quad (5)$$

$$\theta = \arctan \frac{BC - AD}{AC + BD} \quad (6)$$

Alignment using robot–robot observation

This method depends on exchanging the necessary information between the robots for merging their partial maps. It does not rely on the overlap between their local map. It instead relies on the rendezvous case to measure the relative positions of two robots. One of the robots sends this information, which includes its local map, its position, and its observation (ρ_1, θ_1) to another robot. When the message reaches the other robot, the transformation matrix is computed between the coordinate frames as shown in the following equations¹⁴

$$d_x = x_1 + \rho \cos(\varphi_1 + \theta) - (x_2 \cos \Theta - y_2 \sin \Theta) \quad (7)$$

$$d_y = y_1 + \rho \cos(\varphi_1 + \theta) - (x_2 \sin \Theta + y_2 \cos \Theta) \quad (8)$$

$$\Theta = \varphi_1 + \theta - \varphi_2 \quad (9)$$

where x_1 and y_1 are a pose of one of the robots, x_2 and y_2 are a pose of the other robot, φ_1 and φ_2 are the final orientations of both robots, $\rho = (\rho_1 + \rho_2)/2$, and $\theta = \pi + \theta_1 - \theta_2$.

Map merging

As mentioned previously, the map fusion is performed in two phases: map alignment and map merging. After converting each entity in the incoming partial map by applying the transformation matrix as in equation (2), the second phase starts to merge one of the partial maps into a single global map. The Euclidean distance is used to detect the corresponding landmarks; thus if the landmark is a new one, it is simply added to the global map, and if it is known, evaluation of the other robot is used as the best prediction for a position of the landmark. The common landmarks for the other robot are integrated as follows

$$\Sigma_{merged} = \Sigma_1 - \Sigma_1 [\Sigma_1 + \Sigma_2]^{-1} \Sigma_1 \quad (10)$$

$$P_{merged} = P_1 + \Sigma_1 [\Sigma_1 + \Sigma_2]^{-1} (P_2 - P_1) \quad (11)$$

where Σ_i is the covariance matrix, and P_i is the position of the i th landmark. P_{merged} is the coordinate of landmark i in the global map, and Σ_{merged} is its covariance matrix.

Hadoop Map/Reduce

Developers face several problems while running their applications, such as heavy input data and complex computations. To solve these issues, an automatic parallelization and distribution of computations across hundreds and thousands of CPUs is required, which is achieved by Map/Reduce programming model. Hadoop Map/Reduce is a software framework, which allows programmers to exploit huge amounts of resources easily for distributing and processing computational tasks in parallel on a large number of computing clusters. Map/Reduce model provides multiple functions—Map and Reduce—to process the big data by several nodes at the same time to increase execution speed. The Map/Reduce is widespread because it is very easy and straightforward to execute.

Figure 1 shows how Map/Reduce operations are actually carried out in Hadoop.^{19,20} While the client writes a program to make a job, the JobClient will transmit a request to the JobTracker to get a JobID. The JobClient will store the configuration files and JAR files—containing a description of the job—to the Hadoop distributed file system (HDFS). According to the job scheduler, JobTracker can manage several requests in a queue. The JobTracker restores the input data from HDFS to determine the number of the Map tasks. The Reduce tasks are decided by the parameters of the configuration files. The Mapping and Reducing refers to the functional programming languages. The TaskTracker will make a new TaskRunner for executing the Map task and then, the TaskRunner run the map() function inside a Java virtual machine (JVM). While all the Map tasks have been finished, the JobTracker will tell a certain Reduce TaskTracker to implement the reduce() function in other JVMs after downloading the output results from the Map TaskTracker. The Map task picks the input data and generates a set of intermediate (key, value) pairs, which are sorted and partitioned. Then, reducers take the output of the map to produce the final results and will send them to the HDFS. One of the important features of the Map/Reduce model is that the computational process is permanently transported to nodes, which have data.

The main goal of Hadoop is to process massive amount of data sets by splitting these data into smaller chunks. HDFS breaks down huge files into large blocks with size of 16–128 megabyte (MB) optimal for the

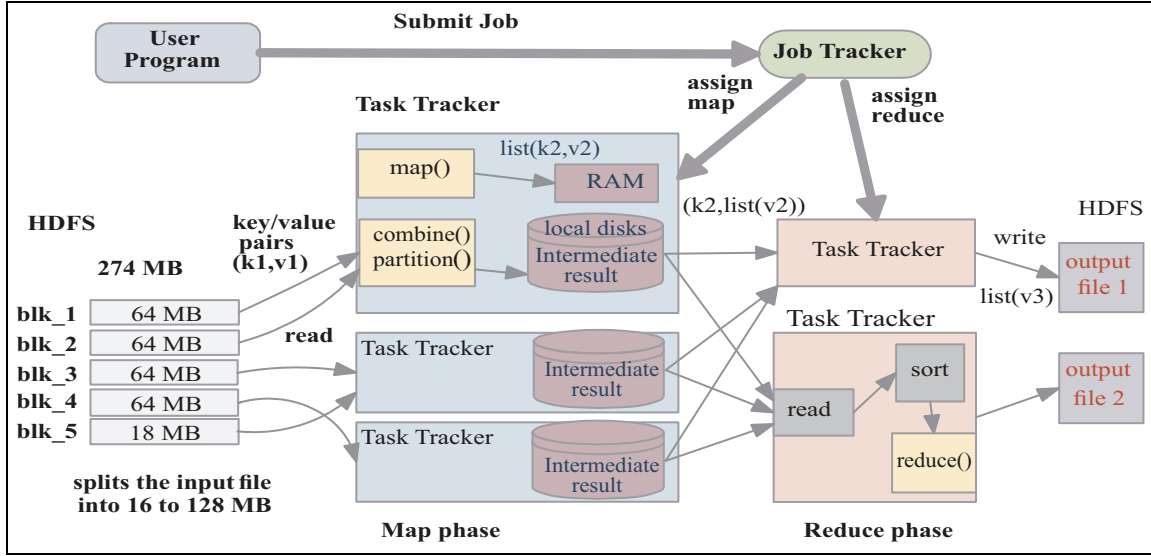


Figure 1. Overview of the Map/Reduce operations.¹⁹

tasks. Hadoop uses a logical representation of the data—named as input splits. Input split is a part of the input processed by an individual map task. The number of splits determines the number of mappers. The split size can be calculated by three parameters of Hadoop properties: `mapred.min.split.size`, `mapred.max.split.size`, and `dfs.block.size` as shown in the formula: Input Split Size = $\max(\text{minimum Size}, \min(\text{maximum Size}, \text{block Size}))$.

Figure 2 presents an example to show how the task deals with the data as key/value pairs.²⁰ Each map task receives data according to the split size. There are three map tasks that classify the data as odd and even numbers, then each map task sends the results to a certain reduce task based on its key.

Proposed method—cooperative SLAM as service in the cloud

Nossair et al.,³ performed three experiments to evaluate the two alignment strategies (data association and inter-robot observations) under different conditions such as a number of overlaps between the partial maps and an effect of the noise. The results of these experiments proved that the data association method does not efficiently work in case of disjoint maps.

Using the relative distance measure between robots avoids this problem; thus, the estimation of coordinate transformation depends on the pose of robot and the performed observation only and not on correspondences between landmarks. Therefore, the inter-robot

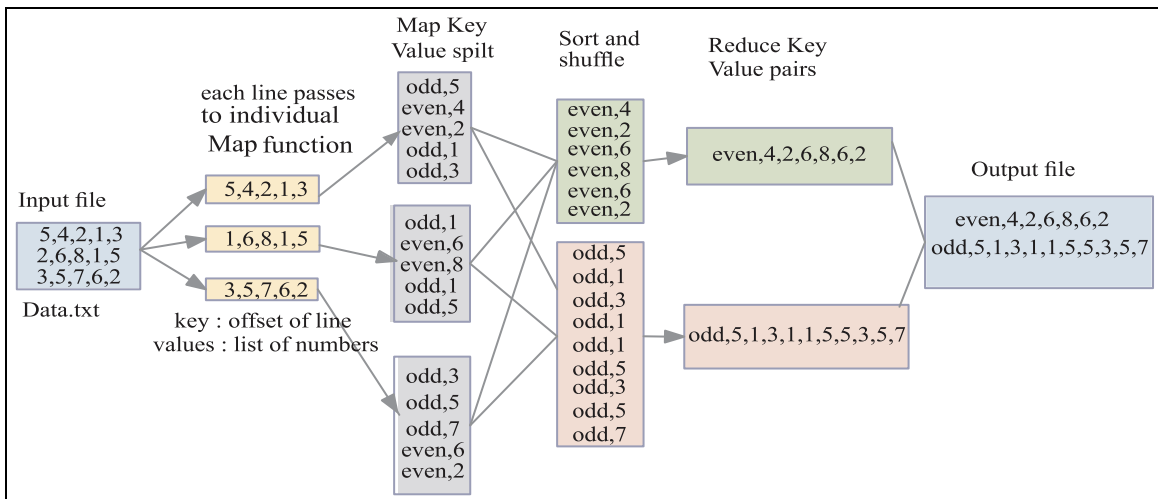


Figure 2. Map/Reduce example for categorizing a set of numbers as even or odd.¹⁹

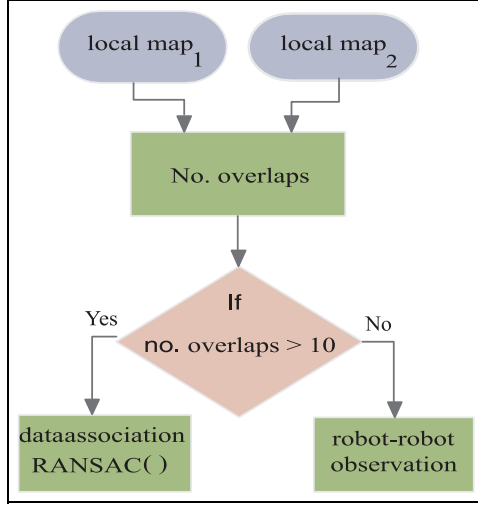


Figure 3. Selecting one of the alignment methods.

observations method is used in case of small degree of the overlap, and the data association method is used in case of large noise. As a result, robots can travel in any environment and merge their partial maps with the highest quality. The leader robot has the two alignment methods (data association and inter-robot observations), and after receiving the information from the follower robots, it begins to check the degree of overlap between their partial maps using landmark's signature and then decides which of the two methods is currently more appropriate for calculating the coordinate transformation between the robot reference frames. The selection between the two methods is illustrated in Figure 3. Results of these experiments prove that if the number of overlap landmarks between the two partial maps is greater than a certain threshold ($th_r = 10$), the data association method gives better results than inter robot-robot observations.

This article proposes an improved map alignment method for obtaining on an accurate merging of the partial maps. In addition, the proposed architecture exploits the parallel processing via the cloud to reduce the merging time with high accuracy and quality in any conditions of the very large environments.

The proposed method is split into two Hadoop jobs. First, the local maps are created by each robot. Second, the global maps are generated by choosing the suitable alignment method depending on the degree of overlap between the partial maps to merge them accurately. There are different approaches (several choices) for launching multiple Map/Reduce jobs,²¹ such as Apache Oozie, JobControl object, and job chaining, which rely on using the output of one as the input to the next, meaning the last reduce output will be used as input for the next map job.

In Figure 4, the first job consists of 400 map tasks. Each robot executes 100 map tasks using the

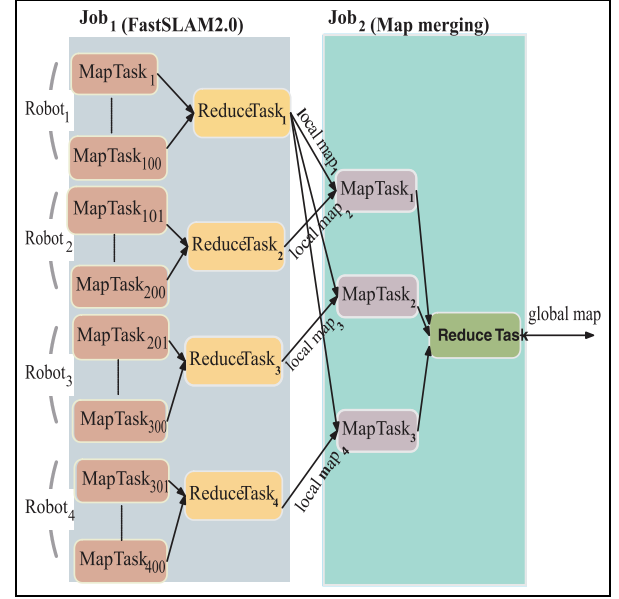


Figure 4. Implementation of cooperative SLAM in Map/Reduce framework (cooperative SLAM) as service in the cloud.

FastSLAM 2.0 algorithm to build its local map using 100 particles. Then, four reduce tasks are used to choose the local map of each robot, which has the highest importance weight. The result of this job produces four local maps. The second job checks the degree of overlapping between each local map and the local map of the master robot to choose one of alignment methods, as a result, there are three map tasks and one reduce task to generate the single global map.

Experimental results

In this section, the experiment is presented to evaluate the proposed approach on two different grid map dimensions (100×100 and 300×300) as shown in Figure 5. The algorithm is implemented by using a single-node, three-nodes, and finally eight-nodes Hadoop cluster by four robots and a single robot. The proposed algorithm is written as multiple pass Map/Reduce tasks using C++ language and is run on nodes that are dual-core server at 7.5 GB RAM.

As shown in Figure 6, the results show that the computation time of the proposed approach by the four robots is reduced 60% compared to its execution on a single-robot platform using a single PC. When the robot—single or four—requests the SLAM service from the cloud, the execution time further reduces. It can be noticed that in small environments (100×100) and with increasing number of VM, the computational cost for the single robot and four robots is nearly equal; as a result, the cloud saves number of the devices—used robots—for performing any tasks. In case of very large

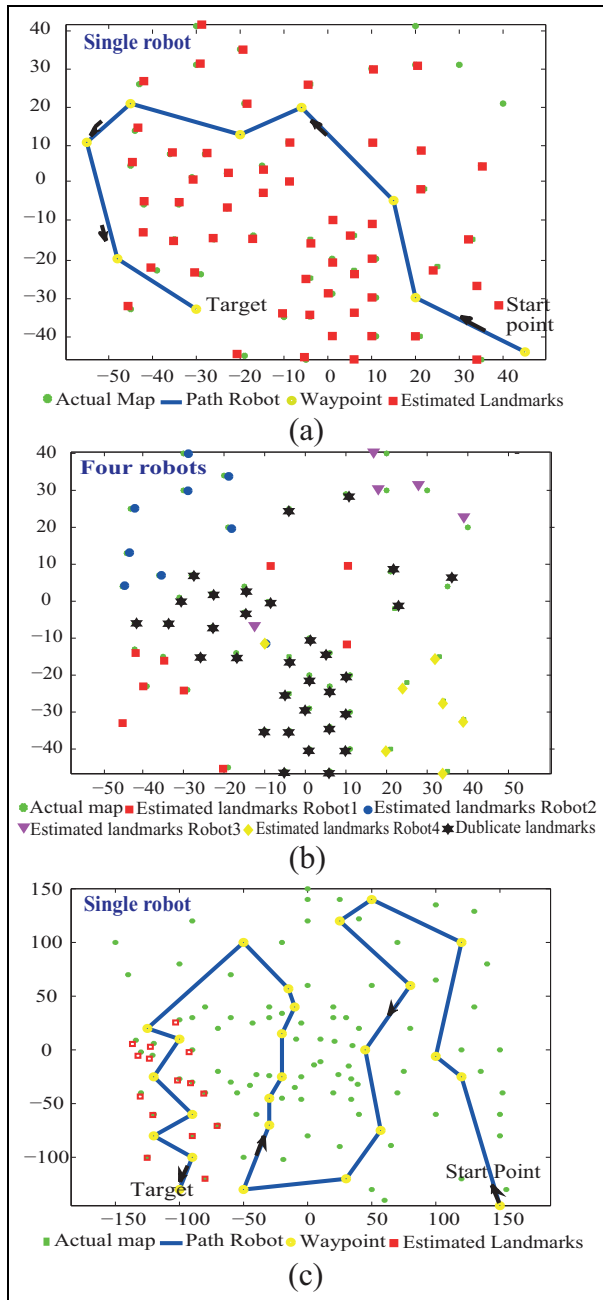


Figure 5. The global maps with dimensions (a) 100×100 , (b) 100×100 , and (c) 300×300 .

environments (300×300), the four robots execute cooperative SLAM as service cloud with 94.8% reduced computation time using an eight node Hadoop cluster only. The presented method maps a large area with lower computational time and higher accuracy than the onboard system of a robot.

Conclusion

This article presented a novel approach for distributing cooperative SLAM, where the map building and the

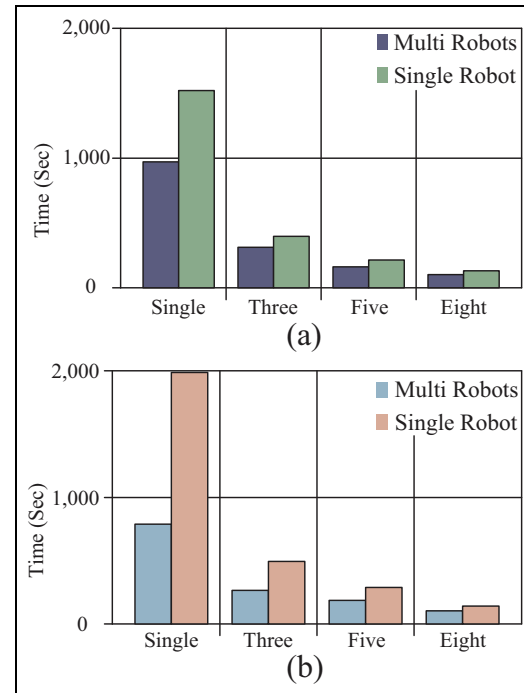


Figure 6. Execution time of FastSLAM 2.0 in Hadoop versus number of nodes by four robots and single robot: (a) grid dimensions: 100×100 and (b) grid dimensions: 300×300 .

merging tasks are run outside the robot—the cloud. Each robot only collects its sensor information, then sends it to a remote server. A direct consequence is that the robots are freed from all computational load. In addition, the presented work proposed an improved method for map alignment. The system can decide which of the two alignment methods (data association method and inter robot-robot observations) is currently more appropriate for calculating coordinate transformation between the robot reference frames. The results show that the proposed technique enhances creation of the global map for very large environments from a group of the robots in a reasonable time with higher quality and efficiency.


Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) received no financial support for the research, authorship, and/or publication of this article.

ORCID iD

Shimaa S Ali  <https://orcid.org/0000-0002-2324-2788>

References

1. Song KT, Tsai CY and Chiu Huang C-H. Multi-robot cooperative sensing and localization. In: *2008 IEEE international conference on automation and logistics (ICAL)*, Qingdao, China, 1–3 September 2008. New York: IEEE.
2. Kwak N. *Improved particle filtering and exploration algorithms for a mobile robot*. PhD Thesis, Seoul National University, Seoul, South Korea, 2008.
3. Nossair Z, Alsammak A, Eldien A, et al. Networked robots path planning using simultaneous localization and mapping algorithm. *Int J Eng Res Dev* 2013; 8: 82–92.
4. Goldberg K and Kehoe B. *Cloud robotics and automation: a survey of related work*. Technical report no. UCB/EECS-2013-5, 27 January 2013. Berkeley, CA: EECS Department, University of California.
5. Waniek N, Biedermann J and Conradt J. Cooperative SLAM on small mobile robots. In: *IEEE conference on robotics and biomimetics*, Zhuhai, China, 6–9 December 2015. New York: IEEE.
6. Howard A. Multi-robot simultaneous localization and mapping using particle filters. *Int J Robot Res* 2006; 25: 1243–1256.
7. Yan Z, Jouandeau N and Cherif AA. A survey and analysis of multi-robot coordination. *Int J Adv Robot Syst*. Epub ahead of print 1 January 2013. DOI: 10.5772/57313.
8. Benavidez P, Muppidi M, Rad P, et al. Cloud-based real-time robotic visual SLAM. In: *2015 9th annual IEEE international systems conference (SysCon) proceedings*, Vancouver, BC, Canada, 13–16 April 2015, pp.773–777. New York: IEEE.
9. Kamburugamuve S, He H, Fox G, et al. Cloud-based parallel implementation of SLAM for mobile robots. In: *Proceedings of the international conference on Internet of things and cloud computing (ICC'16)* (eds DE Boubiche, F Hidoussi, L Guezouli, et al.), Cambridge, 22–23 March 2016, Article 48. New York: ACM.
10. Kamburugamuve S, Christiansen L and Fox G. A framework for real time processing of sensor data in the cloud. *J Sensors* 2015; 2015: 468047.
11. Arumugam R, Enti VR, Bingbing L, et al. DAvinCi: a cloud computing framework for service robots. In: *2010 IEEE international conference on robotics and automation (ICRA)*, Anchorage, AK, 3–7 May 2010, pp.3084–3089. New York: IEEE.
12. Riazuelo L, Civera J and Montiel JMM. C2TAM: a cloud framework for cooperative tracking and mapping. *Robot Auton Syst* 2014; 62: 401–413.
13. Mohanarajah G, Usenko V, Singh M, et al. Cloud-based collaborative 3D mapping in real-time with low-cost robots. *IEEE T Autom Sci Eng* 2015; 12: 423–431.
14. Romero VA and Costa OLV. Map merging strategies for multi-robot FastSLAM: a comparative survey. In: *Latin American robotics symposium and intelligent robotics meeting*, Sao Bernardo do Campo, Brazil, 23–28 October 2010. New York: IEEE.
15. Zhou X and Roumeliotis SI. Multi-robot slam with unknown initial correspondence: the robot rendezvous case. In: *Proceedings of the IEEE/RSJ international conference on intelligent robots and systems (IROS2006)*, Beijing, China, 9–15 October 2006. New York: IEEE.
16. Ozkucur NE and Akin HL. Cooperative multi-robot map merging using fast-slam. In: *Proceedings of the RoboCup international symposium 2009*, Graz, 29 June–5 July 2009, pp.449–460. Berlin; Heidelberg: Springer-Verlag.
17. Dinnissen P. *Using reinforcement learning in multi-robot SLAM*. Master's Thesis, Carleton University, Ottawa, ON, Canada, September 2011.
18. Ballesta M, Reinoso O, Gil A, et al. Analysis of map alignment techniques in visual SLAM systems. In: *Proceedings of the IEEE international conference on emerging technologies and factory automation (ETFA'2008)*, Hamburg, 15–18 September 2008. New York: IEEE.
19. Dean J and Ghemawat S. MapReduce: simplified data processing on large clusters. *Commun ACM* 2008; 51: 107–113.
20. Ali SS, Hammad A and Tag Eldien AS. A novel implementation for FastSLAM 2.0 algorithm based on cloud robotics. In: *IEEE 2017 13th international computer engineering conference (ICENCO)*, Cairo, Egypt, 27–28 December 2017. New York: IEEE.
21. Multiple Map/Reduce approaches, <http://stackoverflow.com/questions/2499585/chainingmultiple-mapreduce-jobs-in-hadoop>