FastSLAM 2.0 tracking and mapping as a Cloud Robotics service

Article *in* Computers & Electrical Engineering · November 2017

DOI: 10.1016/j.compeleceng.2017.11.012

3 authors:

Shimaa S Ali
Benha University
6 PUBLICATIONS 6 CITATIONS
SEE PROFILE

Abdallah Hammad
Benha University
10 PUBLICATIONS 23 CITATIONS
SEE PROFILE

Adly S. Tag Eldien
Benha University
35 PUBLICATIONS 75 CITATIONS
SEE PROFILE

Computers and Electrical Engineering 69 (2018) 412–421

Contents lists available at ScienceDirect

Computers and Electrical Engineering

journal homepage: www.elsevier.com/locate/compeleceng

FastSLAM 2.0 tracking and mapping as a Cloud Robotics service[☆]

Shimaa S. Ali[*], Abdallah Hammad, Adly S. Tag Eldien

Faculty of Engineering at Shoubra, Benha University, Egypt

A R T I C L E I N F O

Article history:
Received 28 May 2017
Revised 7 November 2017
Accepted 8 November 2017
Available online 27 November 2017

Keywords:
Cloud Robotics
Hadoop
FastSLAM
SLAM
Map/Reduce

A B S T R A C T

The Simultaneous Localization and Mapping (SLAM) by an autonomous robot is an intensive computational problem and is considered to be a time consuming process. A major limitation of the pose tracking is the real-time constraint. The pose estimation should be done at an acceptable latency to get accurate position information. In this paper, FastSLAM 2.0 approach is proposed, where the computational process is divided into two parallel tasks, the pose tracking and the map optimization. The presented work depends on a distributed architecture where the tracking and mapping tasks concurrently operate as a service in the Cloud. Therefore, the robot onboard system is freed from all the heavy computations. The experiments are performed on public dataset comparable to state-of-the-art techniques. The results show that the computational cost of the tracking process in the Cloud is reduced by 83.6% as compared to its execution on a single robot platform.

1. Introduction

For more than three decades, the robotics community has focused extensively on a concept of simultaneous localization and mapping (SLAM) [1]. SLAM is the process that an autonomous mobile robot can construct a map of its surroundings; and at the same time, compute its location. There are many techniques which have been used for solving the SLAM problem such as Extended Kalman Filter (EKF) [2], scan-matching SLAM [3], and the particle filter [4]. These techniques have many limitations such as complexity, data association ambiguity, and the quadratic cost of computing a joint map covariance matrix, in recent years, the Factored Solution to Simultaneous Localization and Mapping (FastSLAM) algorithm [5] has attracted enormous attention from researchers as an efficient new solution to the SLAM problem in mobile robotics. The FastSLAM is based on the integration of the Kalman filter and the particle filter [6] however it has some limitations such as the sample impoverishment and the particle depletion problems [7]. The modified version of FastSLAM – called FastSLAM 2.0 – [8], which is developed by Montemerlo, attempts to solve these problems.

The FastSLAM 2.0 approach has been introduced to solve the limitations of the previous techniques for the SLAM problem. Also, it solves an error that accumulates in the robot's odometry that decreases the accuracy of map estimation, but with a massive computation process. If the pose estimation takes too much time, the robot might have moved and already be in another place. As a consequence, the tracking process should be executed at a high frequency and with strong real-time constraints.

[☆] Reviews processed and recommended for publication to the Editor-in-Chief by Associate Editor Dr. L. Bittencourt.
[*] Corresponding author.
E-mail addresses: shimaa.salama@feng.bu.edu.eg (S.S. Ali), abdallah.hammad@feng.bu.edu.eg (A. Hammad), adlytag@feng.bu.edu.eg (A.S. Tag Eldien).

https://doi.org/10.1016/j.compeleceng.2017.11.012
0045-7906/© 2017 Elsevier Ltd. All rights reserved.

In recent years, the possibility of real-time sharing of huge data resources and massive parallel computation – known as Cloud Computing – has attracted attention of the researchers in a field of communication technology [9]. Cloud Computing can be simply defined as storing and accessing data over the Internet instead of a single standalone system. The Cloud widely appeared in Info-Communication Technologies (ICT) areas, such as Google Docs, which extends access to Cloud-based storage and software as Software as a Service (SaaS).

Recently, many contributions have been reported in the literature for performance enhancement of robotics by applying concepts of Cloud Computing, which have opened a new line of research referred to as Cloud Robotics [10]. Robots could benefit from the use of this technology by overcoming the major challenges and limitations, which are faced by their hardware constraints (for example, processors, memory and power consumption). The contribution of this paper is reconfiguring the FastSLAM 2.0 algorithm by distributing the tracking and the mapping tasks to several servers in the Cloud. Thus, the proposed implementation allows an inexpensive robot to perform the SLAM problem faster than relying on its onboard computer with low computational power and memory requirements.

This paper is organized as follows: Section 2 refers to the related work. Section 3 provides the formulation of FastSLAM 2.0 algorithm and a brief introduction to the Cloud Robotics, which shows its benefits and its effects on a performance of the robots. The details of the proposed method are discussed in Section 4. Section 5 shows the results that illustrate the advantages of the proposed approach. The last section concludes this paper and provides future works in Cloud networked robotics.

2. Related work

Cloud Robotics is one of the most important technologies within the last decade for mobile robots. In [11], Jordn and others, provided a systematic overview of many definitions, concepts, and technologies linked to Cloud Robotics. In [12], Chen and others, defined the concept of Robot as a Service (RaaS) based on Service-Oriented Architecture (SOA) and presented an idea of combining robot services with the Cloud, using Microsoft Robotics Developer Studio (MRDS) and Visual Programming Language (VPL). Hu and others [13], extend the information sharing capabilities of networked robotics by proposing Cloud Robotics architecture. In 2010, Kuffner introduced the concept of Cloud Robotics and mentioned the applicability of distributed computing of robotics and automation system [14]. Kwak [1] implemented a prototype for a Cloud-based robot grasping system and object recognition.

The DAvinCi project reported in [15] provides a platform for solving a laser-based FastSLAM algorithm as a Map/Reduce task in the Cloud by distributing the particles among several computing nodes. This paper evaluates the real-time performance analysis for single robot navigation in two different scenarios: the traditional method of Matlab as a sequential algorithm and the presented scheme as a parallel algorithm.

Riazuelo and others [16] introduced a Cloud framework for cooperative tracking and mapping (C2TAM) that runs the expensive mapping to a service operating in the Cloud, while a camera tracking runs on a local CPU. C2TAM utilizes the monocular SLAM algorithm described in [17] – known as Parallel Tracking and Mapping (PTAM) – based on a distributed framework. The main contribution of our paper is an implementation of the FastSLAM 2.0 algorithm with the ability to avoid any computations on the onboard system of the robot by moving all these computational tasks to the Cloud. As a consequence, the proposed implementation can overcome a strong real-time constraint and latency for the tracking task.

In [18], the authors present collaborative 3D mapping in the cloud with low-cost ground robots. The pose of the robot is estimated using the color and the depth frames (images) from the RGB-D sensor. The robots send visual odometry to the cloud for parallel map optimization and merging process, and then the cloud system updates pose information of the local key frame to get the optimal pose graph. This approach excludes the incorporation of the depth error into visual odometry. The proposed approach converts the depth and the color 2D images to 3D point clouds [19] to validate the proposed FastSLAM 2.0 algorithm using a state-of-the-art RGB-D dataset.

3. Background

This section presents a discussion about an improved version of the FastSLAM algorithm and an overview of Cloud Robotics.

3.1. FastSLAM 2.0 overview

The secret to the success of the FastSLAM is that it exploits conditional independencies that are a consequence of the sparse structure of the SLAM problem to factor the posterior into a product of low dimensional estimation problems: a robot localization problem, and a collection of landmark estimation problems. Thus, this algorithm becomes efficient to the large maps, robust to a significant ambiguity in data association, and faster than existing EKF-based SLAM algorithm [20]. Kwak [1] mentioned two serious drawbacks of the original FastSLAM algorithm. First, its performance will eventually degrade when the robot's motion is very noisy and the robot's sensor is too accurate, which is known as the sample impoverishment problem in the sampling process. The second problem is the particle depletion problem in the resampling process, where improbable particles generated in the sampling process are rejected, and the particles with high weights dominate the particle set. The FastSLAM 2.0 has solved this problem by incorporating the most recent measurement in the pose prediction process and the particle set can effectively estimate the SLAM posterior. The basic steps of the FastSLAM 2.0 algorithm are as follows [1]:

Step1: Sampling a new robot pose x_t based on both the most recent motion command u_t and the sensor measurement z_t at time t as follows.

$$x_t^k \sim \rho(x_t | x_{t-1}^{[k]}, u_t, z_t, n_t), \tag{1}$$

where $x_{t-1}^{[k]}$ is the robot pose at time t−1 in accordance with each kth particle and n_t is data association of observation at time t. FastSLAM samples the path $x_{1:t}^{[k]}$ using the particle filter, where each particle consists of N independent EKFs. The kth particle X_t^k contains path $x_{1:t}^{[k]}$ with N landmark estimates, described by mean $\mu_{n,t}^k$ and error covariance $\Sigma_{n,t}^k$.

$$X_t^k = x_{1:t}^{[k]}, \underbrace{\mu_{1,t}^k, \Sigma_{1,t}^k}_{landmark\theta_1}, \ldots, \underbrace{\mu_{N,t}^k, \Sigma_{N,t}^k}_{landmark\theta_N}. \tag{2}$$

Step2: Measurement update. FastSLAM updates the observed landmark estimate, according to the following probabilistic – termed as the posterior – which takes the most recent measurement z_t acquired at time t into consideration:

$$\rho(\theta_{n_t} | x_{1:t}, z_{1:t}, u_{1:t}, n_{1:t}) \overset{Markov}{=} \eta \rho(z_t | \theta_{n_t}, x_t, n_t) \rho(\theta_{n_t} | x_{1:t-1}, z_{1:t-1}, u_{1:t-1}, n_{1:t-1}), \tag{3}$$

where $z_{1:\,t}$, $u_{1:\,t}$, and $n_{1:\,t}$ are the measurements, controls, and correspondences up to time t, respectively, and η is a constant. Each landmark θ_n is described by its location in space with the data association n_t. If the observation does not correspond to any of the landmarks in the current map, a new one is incorporated into the map. Once correspondences for measurements occur, the improved mean and covariance of these landmarks are obtained using the standard EKF update equations [21] given as:

$$K_t^k = \Sigma_{n_t,t-1}^k G_\theta^T Q_t^{[k]-1}, \tag{4}$$

$$\mu_{n_t,t}^{[k]} = \mu_{n_t,t-1}^{[k]} + K_t^k \left(z_t - \hat{z}_t^{[k]} \right), \tag{5}$$

$$\Sigma_{n_t,t}^{[k]} = (I - K_t^k G_\theta) \Sigma_{n_t,t-1}^{[k]}, \tag{6}$$

where K_t^k is the Kalman gain, G_θ is the Jacobian (first derivative) of the predicted measurement with respect to landmark pose θ, and $Q_t^{[k]}$ is the innovation covariance matrix.

Step3: Importance weight. The FastSLAM 2.0 algorithm uses a different proposal distribution than the original algorithm. The importance weight must also be updated to reflect this change. Thus, the importance weight $w^{[k]}$ for kth particle is given by the following expression:

$$w_t^{[k]} = |2\pi L_t^{[k]}|^{-\frac{1}{2}} exp\{-\frac{1}{2}(z - \hat{z}_t^{[k]})^T L_t^{[k],-1} (z - \hat{z}_t^{[k]})\}, \tag{7}$$

where $L_t = G_{x_t} P_t G_{x_t}^T + G_\theta \Sigma_{n_t,t-1}^{[k]} G_\theta^T + R_t$,R_t is denoted linearized vehicle measurement noise, P_t Linearized vehicle motion noise, and G_{x_t} is the Jacobian of the predicted measurement with respect to robot pose.

Step4: Resampling. Each particle is sampled with a probability proportional to its weight $w^{[k]}$.

3.2. Cloud Robotics

Cloud robotics [22] became the remote-brain of the robots by services provided via the Internet [17]. The Cloud enables on-demand network access to a shared pool of computing resources (for example, servers, storage, applications, and resources), which can be speedily provisioned with minimal management effort. Fig. 1 shows the high level architecture components of the Cloud in detail. There are several types of robots like TurtleBot, which has the ROS – referred to an open source operating system- capability, whereas AscTech Quadrotor robot does not have this capability. Thus, the Cloud framework provides this ROS service, storage unit, and Hadoop Map/Reduce computing cluster, which is described in [23]. When the robot demands a particular service from the Cloud, it sends a message to the master node. Then, the Cloud controller takes a control from the ROS master node and uses a service administration point that checks the authorization of the robot to access the Cloud services or not. This point also checks the availability of service at a service registration/removal point. If the demand service is not available, then the demand is placed in a queue to serve it later.

While the robots cooperate with other machines, they override their physical limitations (for example, less computing power, less storage memory and less battery capacity) and become more intelligent. Thus, they are free from computational expensive tasks. Autonomous robots are equipped with multiple cameras and sensors; the collected data from these sensors are used for making decisions such as localization and the path determination for the robot. Sharing these data among several robots increases their volume, and the robots have limited capacities of an onboard system. The Cloud provides offloading the complex tasks to external servers and access to enormous data sets and remote libraries. A direct consequence is that the robots only possess substantial actuators, sensors, and basic processing elements on their embedded chip. There are several features of the Cloud Robotics for performance enhancement of robotics [24], which leads to cheaper, lighter, and smarter robotic systems with longer battery life.

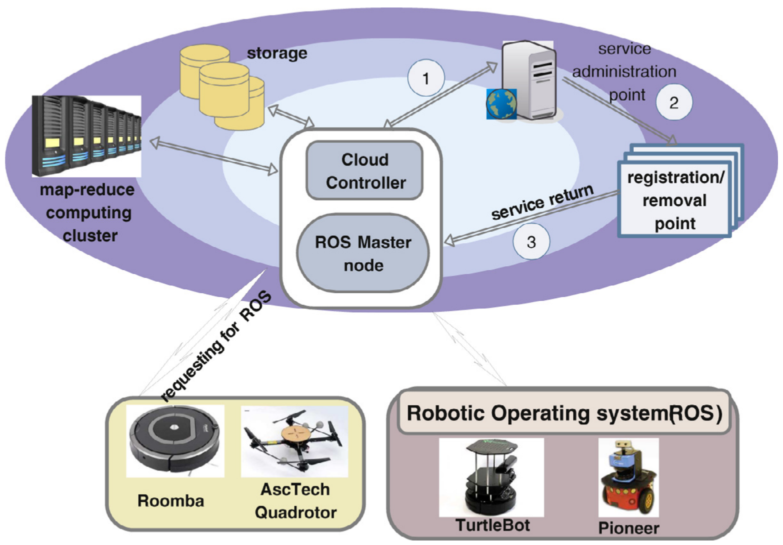
storage
1
service administration point
2
map-reduce computing cluster
Cloud Controller
ROS Master node
service return
3
registration/ removal point
requesting for ROS
Roomba
AscTech Quadrotor
Robotic Operating system(ROS)
TurtleBot
Pioneer

Fig. 1. High level overview of the Cloud Robotic architecture.

4. Proposed method

4.1. FastSLAM 2.0 in Hadoop

A lot of researchers developed a Cloud Computing to exploit the advantages of Hadoop Map/Reduce for a service of robotics. Hadoop [25] is a Java software framework that implements a distributed processing model – known as Map/Reduce framework – which allows distributing and processing massive computational tasks in parallel on several computing nodes easily. This paper shows one of the robotic algorithms, FastSLAM 2.0, that is implemented by Bailey for single robot navigation. The proposed method will focus on implementing this algorithm as Map/Reduce tasks on the Cloud for solving SLAM problem instead of executing with traditional methods on the onboard resources of a mobile robot.

FastSLAM uses the particle filter to compute the posterior over robot paths and a separate estimator (EKFs) for estimating each feature locations in the map. The feature estimators are conditioned on the robot path, that is, each particle consists of a robot localization problem and a collection of landmark estimation problems. Due to this conditional independence, this algorithm can be executed as Map/Reduce tasks in the Cloud; thus, each particle is processed by an individual map task. Then, only one reduce task will be needed to choose the best particle, which has the highest importance weight [1].

4.2. Tracking as a Cloud service

One of the key profits of the Cloud is the ability of offloading heavy tasks to external computers. However, a decision of offloading a specific task should take into account several factors: amount of data exchanged and the deadline time to finish this task. Depending on the delay sensitivity of this task, the computing will be processed on a local unit or Cloud platform. The objective of the proposed work is to minimize the quantity of energy consumed through the robot, under the constraint that the task should be finished within a certain deadline.

Reliable localization with respect to a recognized map is one of the main problems in mobile robotics. The FastSLAM2.0 is an incremental mapping method; the tracking and the mapping are nearly linked, because the current pose of the robot and a position of every landmark are updated jointly every time step. The robot executes controls and collects observations of the features of its surrounding environment; both the controls and the observations are corrupted by a noise.

The FastSLAM 2.0 can recover a map of the environment and a path of the robot from a set of the noise by the increasing number of particles. However, the large number of particles, improves the quality of the estimated uncertainty, but will increase the execution time. The tracking component should be done with robust real-time bonds. If real-time is missing, the estimated pose will also be wrong.

The presented approach exploits the fact that the tracking task can run on a previous sub-optimal map until the next optimization is finished on the Cloud. This paper contributes with decomposing the computing tasks of FastSLAM 2.0 into tracking and mapping tasks that are executed concurrently as Cloud services; Fig. 2 depicts the flow chart of the presented scheme for one particle.

Once new observed features, the robot sends only the sensor information to the server for requesting the tracking and the mapping services from the Cloud as shown in Fig. 3. As a result, this presented scheme saves the energy consumed in transmitting this sub-optimal map from the mapping server to the robot for running the tracking locally as in [16].

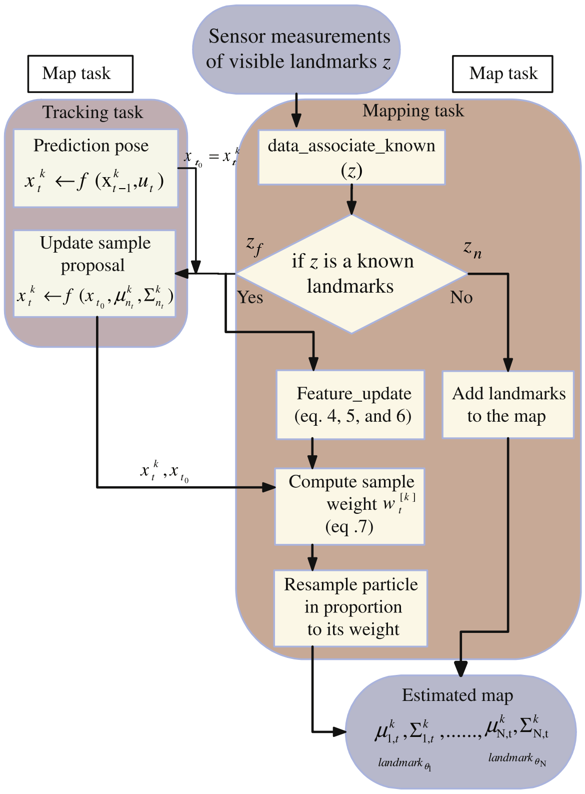
Sensor measurements of visible landmarks z
Map task
Tracking task
Prediction pose
$x_t^k \leftarrow f(x_{t-1}^k, u_t)$
Update sample proposal
$x_t^k \leftarrow f(x_{t_0}, \mu_{n_t}^k, \Sigma_{n_t}^k)$
$x_{t_0} = x_t^k$
Map task
Mapping task
data_associate_known (z)
z_f
if z is a known landmarks
z_n
Yes
No
Feature_update (eq. 4, 5, and 6)
Add landmarks to the map
x_t^k, x_{t_0}
Compute sample weight $w_t^{[k]}$ (eq .7)
Resample particle in proportion to its weight
Estimated map
$\mu_{1,t}^k, \Sigma_{1,t}^k, \ldots\ldots, \mu_{N,t}^k, \Sigma_{N,t}^k$
$landmark_{\theta_1}$
$landmark_{\theta_N}$

Fig. 2. Flow chart of the proposed method.

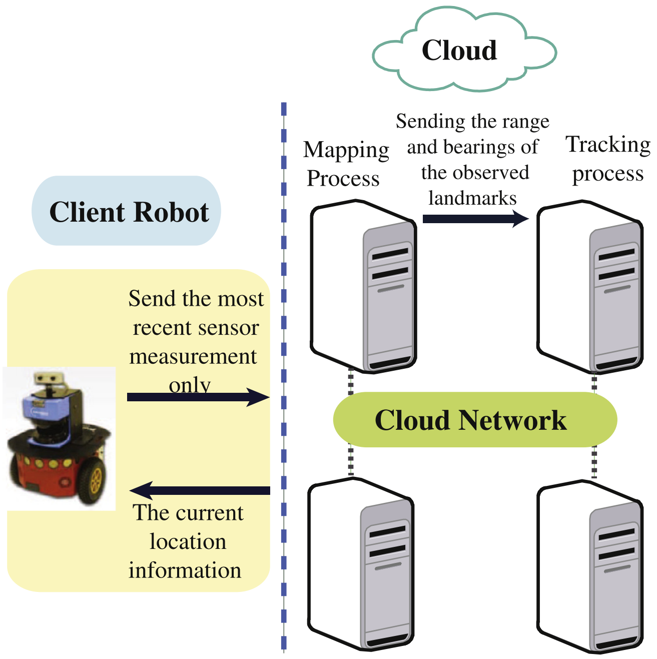
Cloud
Client Robot
Mapping Process
Sending the range and bearings of the observed landmarks
Tracking process
Send the most recent sensor measurement only
Cloud Network
The current location information

Fig. 3. The proposed system architecture.

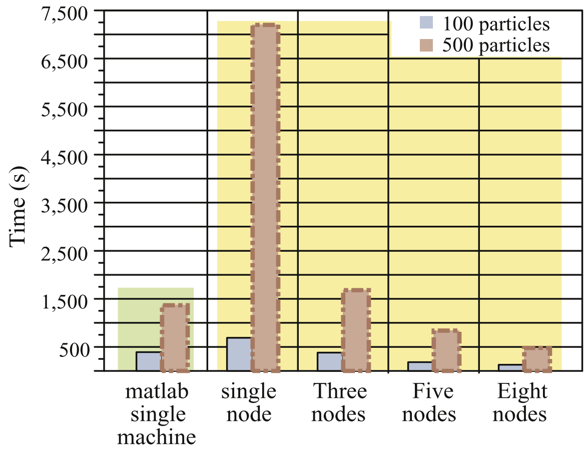

Fig. 4. The performance of FastSLAM 2.0 as Map/Reduce job vs. number of nodes and Matlab sequential algorithm for 100 and 500 particles.

5. Experimental results

In this section, two experiments are carried out that show the performance of the proposed approach for single robot navigation, which is equipped with laser range finder. The first compares the real time performance of the proposed method and the traditional method. The second evaluates the computational cost required for the tracking process as a Cloud service. Amazon Elastic MapReduce (Amazon EMR) service is used for running the Hadoop job based on map and reduce functions. It can be implemented in any of the following supported languages: Java, PHP, R, Ruby, Perl, Python, Bash, and C++. The proposed FastSLAM 2.0 program is written as map and reduce functions with a language C++ and run it on nodes that are dual core server at 7.5GB RAM.

5.1. Computational cost analysis

This experiment executes FastSLAM 2.0 algorithm for single robot navigation by using a single, three, five, and eight-node Hadoop cluster and evaluates the tradeoffs in terms of efficiency FastSLAM 2.0 performance and the execution speed. Also, our aim is to illustrate the computational advantages of the proposed approach as a parallel algorithm and compares it with the traditional method by Matlab as a sequential algorithm.

As a consequence of the above facts, it is possible to run the distributed method with several particles rather than the traditional method for gaining mapping and localization quality. Fig. 4 shows that the classical method with 100 particles takes approximately the same amount of time to process the parallel FastSLAM 2.0 algorithm with 500 particles by 8 nodes. The results prove that the performance of FastSLAM 2.0 is improved, when it is parallelized and executed on a cluster of machines. When the number of nodes is increased from a single node to eight nodes, it leads to reduce the execution time as compared to MATLAB on a single machine. There is a big difference between the execution time of FastSLAM 2.0 algorithm by MATLAB on a single machine and a single node. This is because; a lot of Matlab routines depend upon highly optimized libraries that are tuned to the CPU architecture to take a feature of fast memory.

Fig. 5 shows the map of simulated environment (5.6 km^2 contains a small number of landmarks), which proves that the proposed scheme obtains the same performance as the traditional method with 100 particles at a relatively high speed.

Hence, very large maps – huge information gathered from the robot's sensors and a heavy computation – can be optimized in a short time. While the robot connects to the Hadoop system and requests the SLAM service, this will be performed effectively in an acceptable time for a service robot in the real world.

5.2. Tracking latency

The second experiment is performed to evaluate the computational cost required for the tracking operation as a service in the Cloud and as a client in the robot. Fig. 6 shows the time of process for executing the tracking job on a local computer and on external servers (four nodes) within different noisy controls. It can be noticed that the computational cost of the tracking, which is done by the Cloud compared to the onboard robot is reduced to 83.6% in case of 500 particles. The results confirm that the proposed approach achieved high accuracy in real-time without losing the track. Thus, the low computational complexity allows us to raise the number particles to get higher quality SLAM service by using the Cloud Computing to the robotic environment.

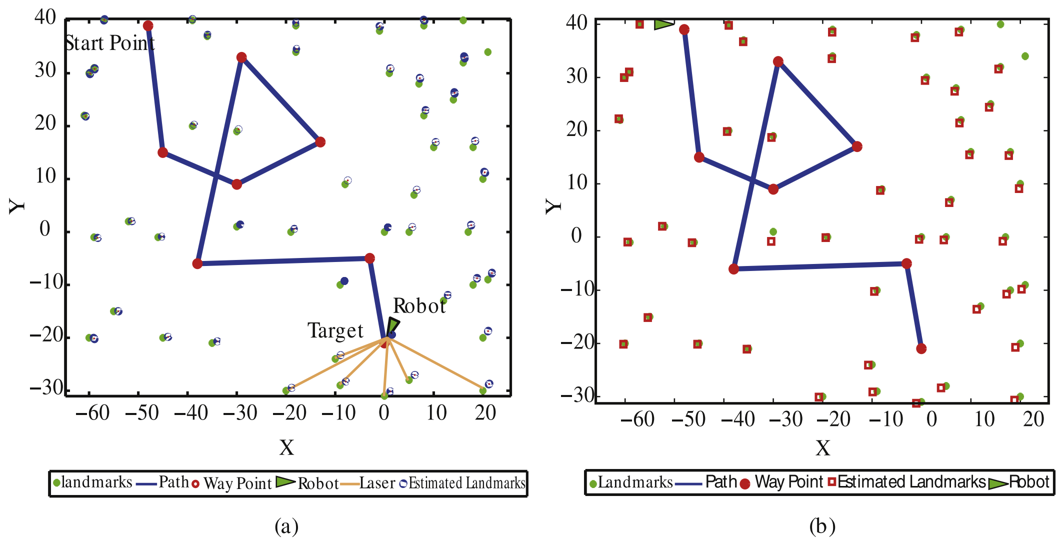

Fig. 5. Simulation map with 100 particles for a single robot, (a) by Matlab within a single machine, (b) by Hadoop Map/Reduce.

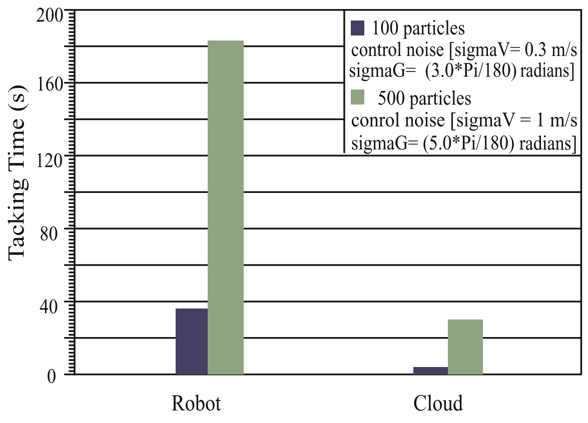

Fig. 6. Estimation of the latency of the tracking process as a client in the robot and as a Cloud service.

5.3. Efficiency of the proposed method

The third experiment is performed on the publicly available dataset [19]. Fig. 7 shows a sample frame that is used in this experiment, where the generated map contains to thousands of landmarks (204,859 point features). Fig. 8 shows global map optimization times against the number of Hadoop nodes. The results show that the proposed method achieves fast map optimization for large scale 3D point cloud datasets by a single robot with high-quality and lower time than direct RGB-D SLAM system [18] that uses of two robots. The proposed system overcomes communication latencies by running the mapping and the tracking processes as service in the cloud. While in [18], the updated key-frame poses are sent back to the robot to enhance its localization accuracy that causes latency.

There are several factors in the SLAM that allow to evaluate performance and quality of this algorithm such as the processing time that reflects the computational effectiveness of this algorithm as shown in Fig. 8, where the time is substantial to reduce the localization error. Also, localization precision is major property of the SLAM algorithm. To estimate the accuracy of the proposed algorithm, comparisons are also performed on the TUM-RGBD benchmark (fr2 desk dataset) with recent visual SLAM algorithms such as ORB-SLAM [26], RGBD-SLAM, and LSD-SLAM [27]. For comparison, key frame localization error in RMSE is computed for the proposed method and compared to the mentioned algorithms as shown in Fig. 9. The result shows that the presented algorithm is more robust and better than RGBD-SLAM and LSD-SLAM, which have big RMSE as a result; these algorithms could not gain stable tracking. The proposed method is slightly accurate than ORB-SLAM, which needs post processing and manual set of the trajectory scale to reduce the error that is rejected for robots that do tasks in real-life environments.

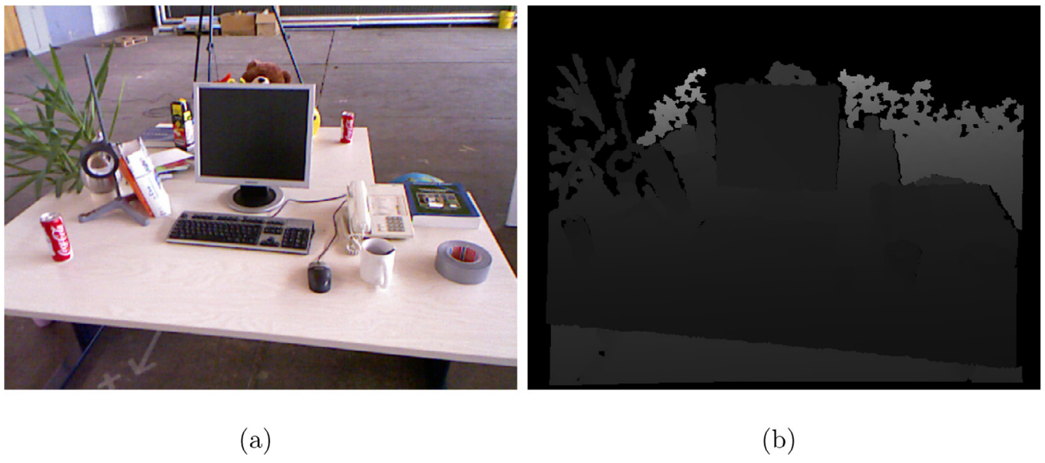

Fig. 7. The 'fr2 desk' dataset – (a) RGB map (b) Depth map.

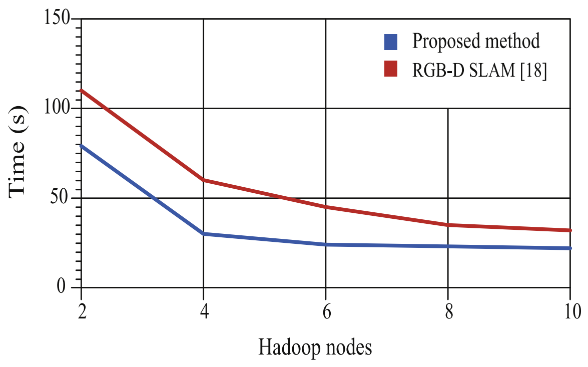

Fig. 8. Map optimization times against the number of Hadoop nodes on the 'fr2 desk' dataset.

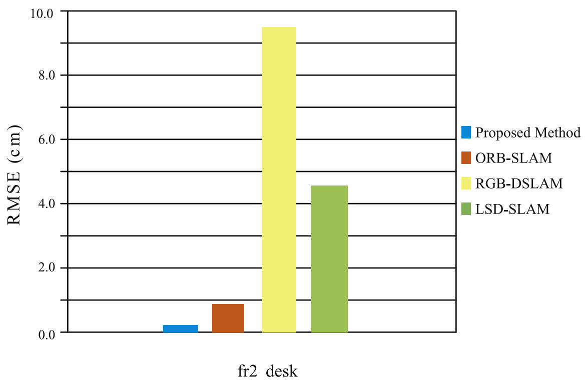

Fig. 9. Comparing key frame position error (RMSE) on the 'fr2 desk' dataset.

6. Conclusion

This paper proposes a novel architecture for distributing the mapping and the tracking tasks-based FastSLAM 2.0 outside the robot – the Cloud. The contribution of this paper is the pose tracking and the large mapping simultaneously executed as a service in the Cloud. The results demonstrate that the pose tracking is not affected by the delays while the Cloud Computing is used. Furthermore, the experiment is performed on public TUM datasets to test the real-time performance of the presented approach. Results show that the system reduces the computation time of mapping process in large environments comparable to state-of the art visual odometry algorithms. The Cloud allows the robots to robustly perform many real world applications by using the machine resources than the robot onboard computers without loss of performance, lower delay, and in an economical way.

The paper also presents a parallel implementation of the FastSLAM 2.0 algorithm as a Map/Reduce task on different number of Hadoop cluster for two cases 100 and 500 particles. The results show that the processing time has been accelerated by a factor of 1.35 compared to its execution using the sequential implementation on the CPU. Therefore, the distributed platform enables the excess of the workload up to values that would be not achievable if the computation is performed on the classical single robot, i.e. raise the number of particles that leads to significant gains in mapping precision and localization accuracy.

References

[1] Kwak N. Improved particle filtering and exploration algorithms for a mobile robot. Ph.d. thesis, Seoul National University; 2008.
[2] Bailey T. Mobile robot localisation and mapping in extensive outdoor environments. PhD dissertation, University Sydney, Australian Ctr Field Robotics; 2002.
[3] Nieto J, Bailey T, Nebot E. Recursive scan-matching slam. Rob Auton Syst 2007;55:39–49.
[4] Howard A. Multi-robot simultaneous localization and mapping using particle filters. Int J Rob Res 2006;25(12):1243–56.
[5] Stentz A, Fox D, Montemerlo M, Montemerlo M. Fastslam: a factored solution to the simultaneous localization and mapping problem with unknown data association. In: Proceedings of the AAAI national conference on artificial intelligence, AAAI; 2003. p. 593–8.
[6] Dinnissen P, Givigi SN, Schwartz HM. Map merging of multirobot SLAM using reinforcement learning. In: Proceedings of IEEE international conference on SMC, Seoul, Korea; 2012. p. 5360.
[7] FPerdomo E. Test and evaluation of the fastslam algorithm in a mobile robot. University Institute of Sistemas Inteligentes Aplicaciones Numericas en Ingenieria; 2009. Master's thesis.
[8] Bailey T, Nieto J, Nebot E. Consistency of the fastSLAM algorithm. In: Proceedings IEEE international conference on robotics and automation, ICRA, Orlando, FL; 2006. p. 424–9.
[9] Savu L. Cloud computing: deployment models, delivery models, risks and research challenges. In: Computer and management(CAMAN), 2011 international conference on, Wuhan; 2011. p. 1–4.
[10] Goldberg K, Kehoe B. Cloud robotics and automation: a survey of related work. Technical report, UCB/EECS-2013-5. EECS Department, University of California, Berkeley; 2013.
[11] Jordn S, Haidegger T, Kovcs L, Felde I, Rudas I. The rising prospects of cloud robotic applications. In: 2013 IEEE 9th international conference on computational cybernetics (ICCC), Tihany; 2013. p. 327–32.
[12] Chen Y, Du Z, Garca-Acosta M. Robot as a service in cloud computing. In: 2010 fifth IEEE international symposium on service oriented system engineering (SOSE), Nanjing; 2010. p. 151–8.
[13] Hu G, Tay WP, Wen Y. Cloud robotics: architecture, challenges and applications. IEEE Netw 2012;26(3):21–8.
[14] Kuffner J. Cloud enabled robots. IEEE-RAS international conference on humanoid robots; 2010.
[15] Arumugam R, Enti V, Bingbing L, Xiaojun W, Baskaran K, Kong F, et al. DAVinci: a cloud computing framework for service robots. In: IEEE international conference on robotics and automation (ICRA), Anchorage, AK; 2010. p. 3084–9.
[16] Riazuelo L, Civera J, Montiel J. C2tam: a cloud framework for cooperative tracking and mapping. Rob Auton Syst 2014;62(4):401–13.
[17] Klein G, Murray D. Parallel tracking and mapping for small AR workspaces. Sixth IEEE and ACM international symposium on mixed and augmented reality; 2007.
[18] Mohanarajah G, Usenko V, Singh M, D'Andrea R, Waibel M. Cloud-based collaborative 3d mapping in real-time with low-cost robots. IEEE Trans Autom Sci Eng 2015;12(2):423–31.
[19] The RGB-D TUM datasets. [Online].. Available: https://vision.in.tum.de/data/datasets/rgbd-dataset.
[20] Castellanos J.A., Neira J., Tardos J.D.. Map building and slam algorithms,335. 2005.
[21] Maybeck PS. Stochastics models, estimation, and control, 1. New York: Academic Press; 1979.
[22] Goldfeder C, Ciocarlie M, Dang H, Allen PK. The columbia grasp database. In: IEEE international conference on robotics and automation, ICRA, Kobe; 2009. p. 1710–16.
[23] Dean J, Ghemawat S. Mapreduce: simplified data processing on large clusters, commun. ACM 2008;51:107113.
[24] Bhardwaj A, Kumar A. Performance enhancement of robotics and automation using cloud computing. Int J Adv Found Res Sci Eng(IJAFRSE) 2015;1.
[25] Sarjolta SV. A study of hadoop: structure and performance issues. Int J Adv Res Comput Sci Software Eng (IJARCSSE) 2015;5:550–4.
[26] Mur-Artal R, Montiel JMM, Tardós JD. Orb-slam:a versatile and accurate monocular slam system. IEEE TransRob 2015;31:1147–63.
[27] Engel J, Schops T, Cremers D. LSD-SLAM large-scale direct monocular SLAM. In: European conference on computer vision (ECCV), Zurich, Switzerland; 2014. p. 834–49.

Shimaa S. Ali received B.Sc. degree in communications engineering and M.sc. in networked based robotics from Benha university, Egypt, in 2010 and 2013, respectively. She is currently an assistant lecturer in the electrical engineering department – Benha university. Her current researches include Cloud Robotics and intelligent computing systems.

Abdallah Hammad received B.Sc., M.Sc. and Ph.D. degrees from Benha University, Egypt, in 1998, 2004, and 2012 respectively. He is currently an assistant professor in the department of electrical engineering – Benha university. He spent two years (2009–2011) as a visiting scholar at the Advanced Robotics & Intelligent Systems (ARIS) Laboratory at the University of Guelph, in Canada.

Adly S. Tag Eldien received B.Sc., M.Sc. and Ph.D. degrees from Benha University, Egypt, in 1984, 1989, and 1993 respectively. He is currently an associate professor in the department of electrical engineering – Benha university. He was the x-head of Benha university network and information center his research interests include, robotics, networks and mobile communication.