Interdisciplinary
Research perspective

Article

A Cloud-Based Cartographer Framework for Robotic Navigation

Kun Ma*[1,2], Chuangyue Hu[1] and Yuzhi Zhang[2]

[1] Jiaxing Yangtze Delta Region Blockchain Technology, Jiaxing, China

[2] College of software, Nankai University, Tianjin, China;

* Correspondence: makuning@foxmail.com

https://doi.org/10.63138/irp010102

Abstract: Cartographer technology, a sophisticated method in the Simultaneous Localization and Mapping (SLAM) field, employs advanced laser and visual data fusion for accurate robot mapping and localization. Traditionally, Cartographer implementations have depended heavily on the robot's own computing resources, often leading to computational delays and inefficient resource use. This paper introduced a novel cloud-based Cartographer framework, inspired by cloud robotics architecture, which aims to improve the efficiency of Cartographer across various robotic applications, significantly reducing computational demands and enhancing robot energy efficiency. Furthermore, the framework includes a hash elimination technique to boost data efficiency and increase overall energy savings. Experimental validations show that this framework significantly enhances the processing speed of Cartographer on cloud computing platforms, reduces robots' energy consumption, and optimizes data transmission.

Keywords: Cloud Robotics; Cartographer; Data Transmission Deduplication; Energy Efficiency Optimization

1. Introduction

In the modern field of Cyber-Physical Systems (CPS), robotics technology has increasingly become a pivotal element in boosting the automation and efficacy of practical operations, notably in sectors like Industry 4.0, agriculture, healthcare, and disaster management. These sectors prioritize latency sensitivity, extensive data handling, and computation-heavy tasks. Yet, robots encounter numerous challenges stemming from limited internal computing and storage capacities. In response, the multi-agent cloud robotics paradigm has surfaced, facilitating a cooperative mode among robots. This paradigm strives to foster a synergistic environment for executing large-scale tasks, effectively leveraging both edge and cloud computing resources [1]. The situation grows more complex with varying energy usage patterns, the cost of operations specific to robots and computing units, and delays in data transmission between computing nodes [2]. These aspects jointly affect the system's real-time interaction and service quality, posing new technical hurdles for researchers.

Google's Cartographer [3] is an advanced, open-source SLAM (Simultaneous Localization and Mapping) tool that efficiently creates 2D and 3D maps from varied sensor data, such as LIDAR, IMU, and odometry. It excels in both large-scale and real-time mapping applications, utilizing advanced algorithms like loop closure detection and pose graph optimization to improve the maps' accuracy and consistency. Consequently, it is well-suited for robotics and augmented reality applications, providing strong support across various platforms and sensors.

The work [4] explores a novel SLAM approach that combines edge and cloud computing capabilities. This method enhances SLAM computational efficiency and accuracy by allocating processing tasks between edge devices and cloud infrastructure. The synergy between edge and cloud computing facilitates rapid local processing at the edge, while more complex and computation-intensive tasks are managed by the cloud, thus optimizing response times and resource utilization. However, modern robotic systems are increasingly reliant on computationally intensive models, such as Deep Neural Networks (DNNs), for tasks involving localization, perception, planning, and object detection. This reliance makes it challenging for resource-constrained robots, such as low-power drones, to effectively operate these advanced models. The introduction of cloud robotics technology allows these robots to outsource computational tasks to centralized servers, enabling them to handle more precise and demanding models [5].

To fully enable on-demand access and usage of robot Cartographer cloud services, several challenges must be addressed. Two key issues are: (1) the architectural challenge of a service-oriented Cartographer, involving the design of architecture, interfaces, and infrastructure to present the cloud's Cartographer capabilities as "Software as a Service" to front-end robots, thus allowing multiple robots to access, customize, and utilize these services as needed; (2) the challenge of ensuring efficiency in the cloud robotics Cartographer process. The cloud robotics-based Cartographer process introduces various new factors that could impact real-time performance, with the reduction of latency in continuous Cartographer data communication between robots and the cloud being a critical concern.

This study explores these significant challenges by constructing and implementing a prototype model of a Cartographer service centered on "Cloud + Robotics." Building on this model, we conducted experimental tests and introduced an optimized communication strategy to enhance the efficiency of Cartographer data communication, confirmed through experimental validation. The structure of the article is organized as follows: Section 1 provides an overview of the research background and related literature; Section 2 elaborates on the design of the cloud robotics-based Cartographer service architecture and its data communication optimization measures; Section 3 discusses the experimental test results, validating the functionality of the proposed architecture; Section 4 offers conclusions.

2. Related Work

As a kind of SLAM technique, Cartograph leverages visual sensors to map environments and track the position of robots. Ali [6] explores the effectiveness of the Cartographer algorithm in a simulated ROS (Robot Operating System) environment using Gazebo and a TurtleBot for 2D mapping in indoor settings. The research highlights the ability of the Cartographer to integrate both odometry and pose estimation effectively, making it a suitable choice for SLAM (Simultaneous Localization and Mapping) processes in static indoor environments. The paper specifically notes the algorithm's performance in constructing accurate 2D maps using LIDAR data collected from a mobile robot, demonstrating its applicability and efficiency in such scenarios. Łukasz [7], focus on a methodology that uses a simulation framework to determine the most effective combination of hardware components for SLAM tasks. Their key findings include that the optimal hardware configuration for their robot, used in decontamination tasks within indoor environments, incorporates three 2D LiDARs, an Inertial Measurement Unit (IMU), and wheel odometry sensors. This combination was shown to provide the best performance in terms of accuracy and operational efficiency. The study emphasizes the advantages of simulation-based approaches, offering a cost-effective, scalable method for preliminary testing before real-world application. This allows for fewer physical tests and modifications, speeding up the development process and reducing costs. Nguyen [8] focuses on enhancing the accuracy and efficiency of mapping and navigation for mobile robots in indoor environments using various 2D-SLAM algorithms and path planning techniques. It features an adaptive four-wheel robot system, optimized through the integration of sensor data and real-time path planning algorithms, to achieve high precision in map building and effective autonomous navigation.

However, the challenge with Cartographer lies in its need to process large amounts of data from these sensors to achieve precise mapping and navigation. Accurate mapping, crucial for generating detailed and reliable environmental models, depends heavily on the robustness of the underlying algorithms. Moreover, the computational complexity of Cartographer is significant, especially when processing extensive sensor data. In practical applications, a robot's computing capabilities are often constrained by hardware, battery life, and cost factors. Thus, optimizing Cartographer for efficient operation with limited resources becomes a critical issue.

To address this issue, researchers have begun to divide the computing process of local robots into two parts: sensor data collection and cloud data processing, and move computationally intensive tasks to resource rich clouds for execution. Relevant work combining robots and the cloud has been actively pursued: Xu [9] proposed a Digital Twin-based Industrial Cloud Robot (DTICR) framework, integrating cloud computing with industrial robot technology. In the cloud, robot control functions are encapsulated as services and combined with digital models for manufacturing simulation. Through digital twin technology, it ensures that the digital model in the cloud is synchronized with the actual manufacturing system, thus achieving precise robot control. This combination enhances the flexibility and accuracy of industrial robots. Chinchali [10] and others have collected robotic perception tasks like localization, perception, and object detection in the cloud under resource-limited conditions to use Deep Neural Networks (DNNs) for higher accuracy while minimizing the cost and latency of cloud communication. The results demonstrate that their strategy effectively achieves task offloading between robots and the cloud, not only significantly improving the robots' perception accuracy but also optimizing cloud communication costs and resources.

3. Cloud-based Framework for Cartographer

As discussed in the previous section, existing work on cloud robotics Cartographer remains at the stage of validating feasibility and effectiveness. To successfully apply the cloud robotics architecture to Cartographer practice, several challenges need to be addressed. The two most prominent issues are: (1) the service-oriented problem, which involves designing an architecture, interfaces, and infrastructure to provide cloud-based Cartographer capabilities as "Software as a Service" to front-end robots; and (2) the communication efficiency problem, which concerns how to ensure the efficiency of cloud robotics collaborative Cartographer in a networked and dynamically changing resource environment, such as how to reduce the latency of Cartographer data communication between robots and the cloud. This section proposes solutions to these problems.

3.1. Framework Structure

In the design system of cloud robotics, the cloud provides Cartographer functionality on demand through service-oriented means. This concept of "service-oriented" and "on-demand" implies that robotic terminals do not need to deeply understand the specific implementation of cloud algorithms. Instead, they only need to incorporate the Cartographer cloud service calling toolkit and continuously transmit key mapping and positioning data to obtain real-time feedback on map construction and positioning. The cloud service offers a unified interface capable of providing customized and efficient Cartographer services to many robots. To achieve this goal, this study proposes a Cartographer service structure based on the cloud robotics design concept, as shown in Figure 1.

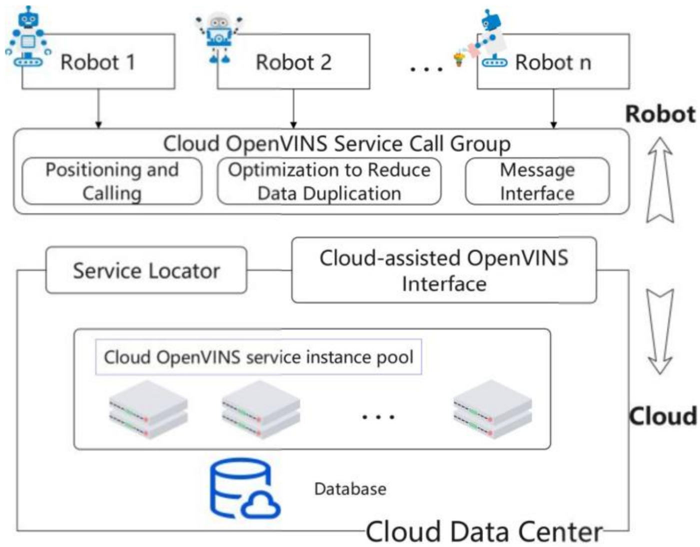

Figure 1: Cartographer service-oriented framework based on cloud-robot architecture

The architecture depicted in Figure 1 primarily consists of two parts: the robot side and the cloud side. The design philosophy and implementation approach for both parts are explained below.

On the robot side, the main goal is to implement a Cartographer service call package, enabling the utilization of "Software as a Service" provided by the cloud for robots. This service call package should include the following functionalities:

1. Methods to start and stop the Cartographer process, meaning at least two interfaces should be provided, allowing robots to initiate or cease the Cartographer service.

2. A method to receive mapping and positioning data, enabling the transmission of data collected by the robot's sensors to the cloud for processing.

3. A method to query the current map and location, allowing the results processed by the cloud to be returned to the service call package, providing interfaces for other modules within the robot to access the results of Cartographer processing.

On the cloud side, the primary role is to provide "Software as a Service" for the robot side. This part includes at least the following modules:

1. Service Entry Module: This module is responsible for receiving requests and forwarding them to the respective Cartographer execution instance.

2. Instance Pool Management Module: This module handles the allocation of resources for each Cartographer execution instance, such as finding an idle Cartographer execution instance from the existing pool or dynamically adding a new Cartographer virtual machine to the instance pool.

3. Service Instance Pool Module: This module configures and runs instances.

4. Resource Repository Module: It stores various implementations of the Cartographer algorithms, such as virtual machine images containing different algorithm implementations.

3.2. Optimization of Data Transmission Between Cloud and Robots

The data typically transmitted between robots and the cloud is large, such as images, and the robot's network is often a resource-limited wireless network. Therefore, it is necessary to reasonably control the scale of data transmitted between robots and the cloud. To address this challenge, this paper employs data deduplication technology, especially using hash values to identify and eliminate duplicate data blocks, thereby enhancing the execution efficiency and usability of the framework, the deduplication process of this method is shown in Figure 2.

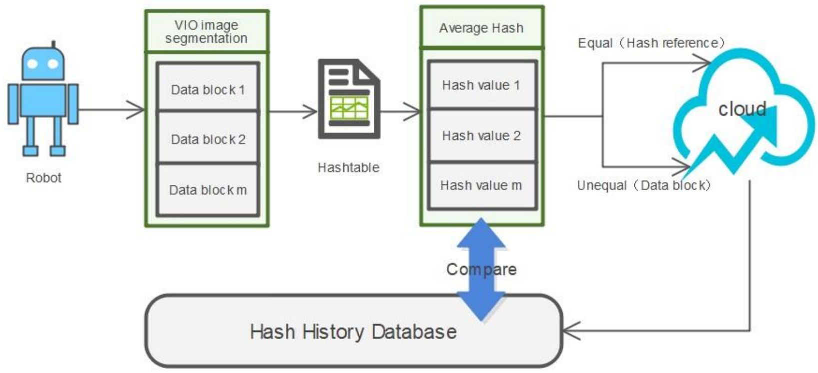

Figure 2 The process of data transmission deduplication

3.3. Prototype Implementation

Based on the proposed framework, this paper has implemented a corresponding prototype system. Considering that the framework aims to provide a unified service platform for various robots, it has been designed with broad applicability for different robotic systems. Therefore, the Robot Operating System (ROS), which is widely used across various robots, was chosen as the foundation. To facilitate implementation and verification, the cloud model was simplified by using pre-deployed virtual machines configured with

Cartographer modules to replace cloud instances. This implementation primarily draws on the foundational work by [11].

The implementation is divided into the following parts:

(1) Robots, operating on ROS, run the image acquisition module (such as the Primesense module) to capture visual data using devices like cameras.

(2) Cloud-based virtual machines, also based on ROS, execute computational modules, such as the Gmapping module, to generate map data and positional information.

(3) Communication between robots and the cloud is synchronized through the ROS Bridge[11] module.

(4) A data deduplication module, positioned between the ROS Bridge and the cloud-based virtual machines, utilizes the previously mentioned hash-based data deduplication method to optimize data transmission.

The entire execution process unfolds as follows: Robots initiate the image acquisition module and send a request to the cloud; the ROS Bridge identifies an available cloud-based virtual machine and synchronizes it with the robot; the robot transmits data, from which redundant data has been eliminated, to the cloud; the cloud-based virtual machine processes the data using Cartographer and returns the resultant maps and positional information back to the robot.

This approach not only ensures a consistent environment for both local and cloud computations but also allows for the scalable integration of different robotic systems into the proposed framework, enhancing the efficiency and reducing the computational load on individual robots.

4. Experiments and Results

This section tests the Cartographer service under the cloud robotics architecture, focusing on Cartographer processing speed, energy consumption, and network bandwidth usage, to verify the effectiveness of the work presented in this paper. The experiment used two private servers as the cloud, connected through a gigabit bandwidth switch. The specific configuration of the servers was as follows: Intel Xeon E5-1650 processor, 32GB memory, Ubuntu 14.04 operating system. The robot side used an advanced autonomous navigation vehicle equipped with an Raspberry Pi and a Primense depth camera. The TUM VI Benchmark [12] was used as the data source.

First, we tested the performance comparison between the Cartographer service based on cloud robotics architecture and standalone Cartographer for the same Cartographer algorithm (Figure 5 a). The main test compared the performance of the Cartographer algorithm under this service architecture and in a standalone robot situation. From the results shown in the figure, compared to standalone Cartographer, our service framework increased the frame rate from 4 Fps to 17 Fps and reduced energy consumption from 15.5 J/10 frames to 6.5 J/10 frames. This demonstrates the advantage of the cloud, enhancing Cartographer speed and effectively reducing energy consumption.

Next, we tested the network bandwidth usage during Cartographer operation under a standard point-to-point network and our service framework, figure 4 shows that in the entire Cartographer process, the bandwidth usage of a regular point-to-point network remained between 4.3 MB/s and 5.1 MB/s. In contrast, in our service architecture, the network bandwidth usage dropped to between 2.3 MB/s and 2.8 MB/s. This demonstrates that through data communication optimization, our service framework effectively reduced network bandwidth usage, thus reducing the network load.

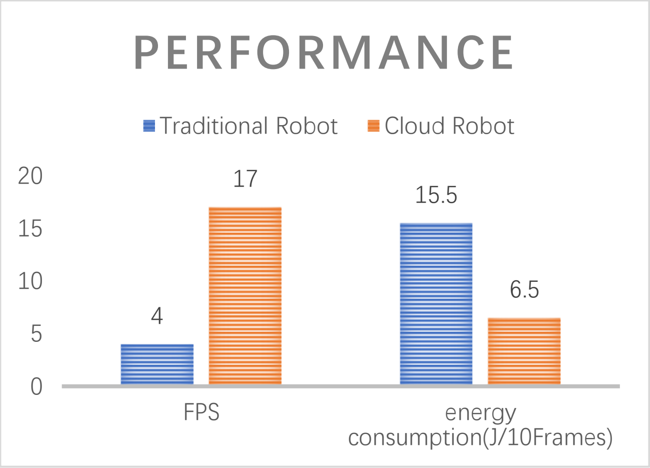

Figure 3 Performance Comparison

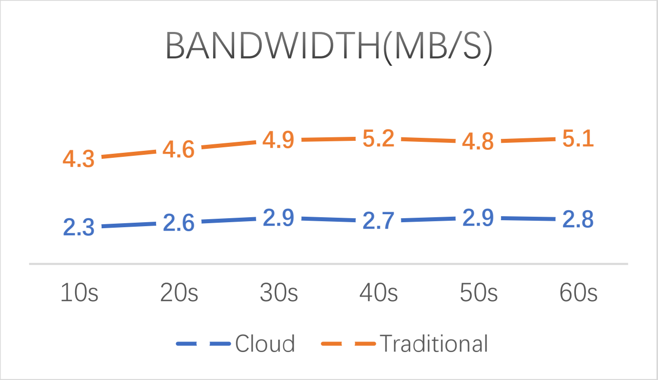

Figure 4 Bandwidth Comparison

5. Conclusions

This paper has conducted a thorough analysis of the service-oriented architecture of Cartographer based on cloud robotics, exploring its design, implementation, and testing in depth. The experimental data confirms that this architecture significantly accelerates the processing efficiency of Cartographer while achieving notable optimization in energy consumption and network bandwidth usage. This method offers a cutting-edge and efficient approach, which provides research and reference value for the future development of robotics technology. While the presented framework shows promising results in integrating Cartographer with cloud robotics, future work could significantly benefit from addressing scalability and security concerns. Testing the framework under more dynamic and unpredictable conditions with a broader array of robots and tasks is essential to ensure its robustness and practical applicability. Additionally, as the architecture relies heavily on cloud computing, enhancing security protocols and implementing privacy-preserving techniques are crucial to safeguard data integrity and confidentiality, especially in applications where sensitivity is paramount.

References

[1] Afrin M, Jin J, Rahman A, et al. Resource allocation and service provisioning in multi-agent cloud robotics: A comprehensive survey[J]. IEEE Communications Surveys & Tutorials, 2021, 23(2): 842-870.

[2] Wu, M., Yeong, C.F., Su, E.L.M., Holderbaum, W. and Yang, C. (2023), "A review on energy efficiency in autonomous mobile robots", Robotic Intelligence and Automation, Vol. 43 No. 6, pp. 648-668. https://doi.org/10.1108/RIA-05-2023-0060

[3] Dillemuth J. Camilla, Cartographer[M]. American Psychological Association, 2021.

[4] Lv T, Zhang J, Chen Y. A SLAM algorithm based on edge-cloud collaborative computing[J]. Journal of Sensors, 2022, 2022.

[5] hinchali S, Sharma A, Harrison J, et al. Network offloading policies for cloud robotics: a learning-based approach[J]. Autonomous Robots, 2021, 45(7): 997-1012.

[6] Ali W, Sheng L, Ahmed W. Robot Operating System-Based SLAM for a Gazebo-Simulated Turtlebot2 in 2d Indoor Environment with Cartographer Algorithm[J]. International Journal of Mechanical and Mechatronics Engineering, 2021, 15(3): 149-157.

[7] Sobczak Ł, Filus K, Domańska J, et al. Finding the best hardware configuration for 2D SLAM in indoor environments via simulation based on Google Cartographer[J]. Scientific Reports, 2022, 12(1): 18815.

[8] Nguyen Q H, Johnson P, Latham D. Performance evaluation of ROS-based SLAM algorithms for handheld indoor mapping and tracking systems[J]. IEEE Sensors Journal, 2022, 23(1): 706-714.

[9] Xu W, Cui J, Li L, et al. Digital twin-based industrial cloud robotics: Framework, control approach and implementation[J]. Journal of Manufacturing Systems, 2021, 58: 196-209.

[10] Chinchali S, Sharma A, Harrison J, et al. Network offloading policies for cloud robotics: a learning-based approach[J]. Autonomous Robots, 2021, 45(7): 997-1012

[11] Koubaa A, Alajlan M, Qureshi B. ROSLink: bridging ROS with the internet-of-things for cloud robotics[J]. Robot Operating System (ROS) The Complete Reference (Volume 2), 2017: 265-283.

[12] Schubert D, Goll T, Demmel N, et al. The TUM VI benchmark for evaluating visual-inertial odometry[C]//2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2018: 1680-1687.