# Evaluation for ROS based Distribute Robot Searching System

LIU YIFAN[†1] MIDORI SUGAYA[†1]

***Abstract***: Currently, multiple robots are expecting to provide useful services corporately at the event of disaster and medical treatment, nursing care, home security etc.. Autonomic robots such as searching and following people should have the functionalities of collecting data from its physical environment and analyzing to share the information to each other. We build up these mechanisms with the ROS middleware that had been widely used as a software framework for robotic systems. ROS provides a structured communication layer above host operating system and can be used to build a distribute system. However, while we bring our ROS based distribute searching system from local network to the internet, it did not work as well as expected because of latency in this system which should be improved. In this research, in order to make this system to achieve appropriate performance, we did a preliminary experiment to make an evaluation of ROS communication performance.

***Keywords***: multiple robots, ROS, latency, communication performance

## 1. Introduction

Multiple robots' services are required for rescue, nursing, security and etc. These robots working together is more efficient than the robots that work separately. These cooperated works with information exchange could avoid repeated works while searching people. Moreover, it saves time to search survivals and is able to save more people. In other field such as nursing robot, security robot, there are also such demands existed for multiple robots to collect data so that we can know abnormal situation such as when nursing robots detect a falling down of person and invasion detected by security robot in time.

To achieve the goal to provide the service with multiple robots that can share information with each other to do their work efficiently, we consider that a software framework for building the distributed robotics service is required to satisfy the requirements. The one of the most widely used software framework for robotic systems is ROS (Robot Operating System) [1, 2], and it provides a structured communications layer above the host operating systems. It also provides libraries and tools to help software developers create robot applications. The ROS node makes no assumptions about where in the network it runs, allowing computation to be relocated at run-time to match the available resources, which makes it suitable for distribute robotic system.

In robotic mapping and navigation, simultaneous localization and mapping (SLAM [3]) is the computational problem of constructing or updating a map of an unknown environment while simultaneously keeping track of an agent's location within it. The implementation of RGB-D SLAM [4] algorithms uses RGB image and depth information from Kinect [5] and it is required for a pretty large amount of calculation which needs to process above 10 images with depth information a second to build up a map. SLAM is also used for auto-drive. Machine learning gives computers the ability to learn without being explicitly programmed. Recently, machine learning [8] had also been used in robotic system for image process, object recognition and so on.

However, both SLAM and machine learning methods requires huge calculation load to process a large amount of images which can not be done only on robot itself because the resource of robot is limited. Therefore, distributing the calculation load to a powerful server would be an appropriate. With these reasons, 1. Multiple robots works together improves efficiency, 2. Resource of robot is limited, it is clearly that there is a high demand for multiple distribute robots.

We developed a distribute robot searching system with a general architecture composed with the general applications such as SLAM and machine learning using ROS. It is expected that the system will be satisfied for the requirements of the variety of services. It is important to consider the issues that have aroused in the development. Based on this development experiment, we consider the issues for the distributed search robotics systems as follows. (1) How to consider the load of the calculation and delay of communication. (2) How to organize the distributed robots and distributed their work. (3) How to guarantee the resource for some special application. We proposed flexible and reconfigurable distributed robotics systems that consider the overcome the problems and issues that are described above.

As a first step, we evaluate the basic performance of the distributed searching robotics systems to understand the issues of one that we described above. In this paper, we will show firstly the detail about the basic architecture of searching robotic system. Then, we will show the performance of collecting logs from the distributed robot system.

In this paper, in section 2, we described that the construction of the ROS based distribute robot searching system that we developed and issues of multiple robotic system. In section 3, first we introduced the multiple robotic system that we proposed and then discussed about the possibility of using ROS for load distribution. Finally, we made a comparison between ROS topic communication (ROSTCP based) and original socket TCP [7]. The last section makes a conclusion of this paper and describes about future work.

## 2. Issues

As we described in the section one, we had developed the distribute robot searching system with intelligent which used the advanced machine learning algorithms and it has the ability for searching efficiently for the target object based on target's features. The system is shown in the Figure 1, The first step, a cascade classifier [6] had been built up from a large amount of positive images and negative images. Then, while multiple robots

†1Shibaura Institute of Technology

are moving around, the images taken by the robot are sending to the server, and we use the cascade classifier to recognize whether there is a target thing in these images. At the same time, RGB images and depth information from RGBD camera was used to build up a map by SLAM algorithm on server. All of the calculation work was finished on server which leaded to the reduction of load on robots.

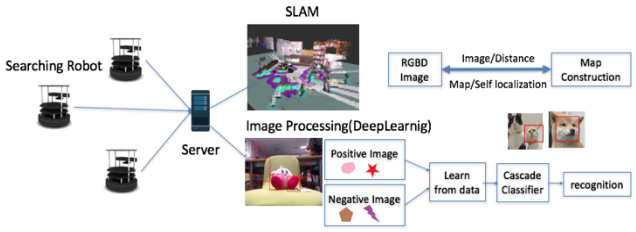


Figure 1 Distribute Robot System

It is realized with the ROS, RGBDSLAM, and deep learning libraries. However, there is not any approach about using multiple robots to improve the efficiency for searching.

### 2.1 Calculation load

As illustrated in Figure 2, the calculation load of building up a cascade classifier is high, it costs about 35 percent of the CPU and 20 percent of RAM resource in almost an hour for building up a cascade classifier which is not reasonable for robotic applications.

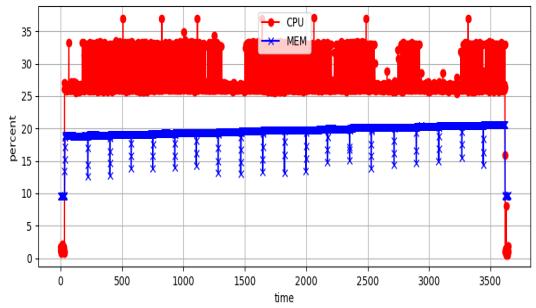


Figure 2 Calculation Load

- Using Raspberry pi to build up a cascade classifier
- CPU: Coretex-A53, 1.2Hz, 4 core, RAM: 1GB
- Time of building up cascade classifier: 1 hour

Therefore, in order to make this part of time short, it is necessary for the implementation to distribute this part of load to a powerful server.

### 2.2 Evaluate the latency of collecting data

While we build up the distribute searching systems with a ROS middleware, it can be used under a wide area network. Generally, multiple service robotics systems use a small local network to communicate and share the information with each other. However, in the future, it is required to consider the wide area network distributed systems. For the purpose of making it easy to establish network connections, and easy to obtain the benefit of sharing information with each other, TCP/IP is one of the option. TCP/IP protocol provides a reliable communication for the system, however, there are not much evidences to evaluate ROS at distributed systems over TCP/IP. Especially, when we use ROS with the internet base, there are not much actual evaluation having been provided.

To build up the distributed systems with ROS, it is required for the actual performance comparison to the existed approach. Then we need to consider about the performance problem of the system. If the data collected by robots and the commands to control the robots may not arrive in time which can lead to a reduction in efficient and even some damage to the robots. For example, if the latency of the network is considerable high, maybe all of the robots can not know others' searching process which can even lead to a crash of the system. And for commands which need a lower latency, if the users notice that the robot will fall or be crashed by the dangerous environment such as some bricks will fall in front of the robot's moving route, and then send a command for the robot to stop immediately, the command may not arrive in time because of the network latency which can not prevent the robot from being damaged.

## 3. Idea

In this research, we propose a system that using multiple robots for searching efficiently with considering the performance problem of distributed robotics system with connection to a network.

### 3.1 Proposed System

In Fig 3, we show the whole system that is based on ROS middleware and cloud server. The multiple robot systems can share information through topics which means that when information exchange is required, the publish side (no matter server or robot) will publish a topic and the subscribe side will subscribe the topic that hold the same name as the publisher is publishing. For example, if robot(client) want the information that the server holds, the server will publish a topic and the robot can subscribe this topic and then information exchange can be finished. and all the robots are controlled by ROS master which is running on server. If robot wants to get data from others, the master node will tell the robot the IP address where the data is. While heavy load of calculation is needed, the data will be sent to server for calculation and server will send result back to the node that needs the result. For example, while using RGBD-SLAM for multiple robots to build up one big map, all the image data collected by each robot should be sent to server and after doing SLAM calculation, the map can be sent back to the robot which need this map for navigation.

Currently, we consider to implement the cloud server by using Amazon Web Service (AWS) to build up the server consists of several virtual machines for complex calculation. And in order to make the calculation quickly, we consider that building up a Hadoop cluster will be helpful. For image process, we plan to use Hadoop Image Process Interface(HIPI) for the procession of the large amount of image.

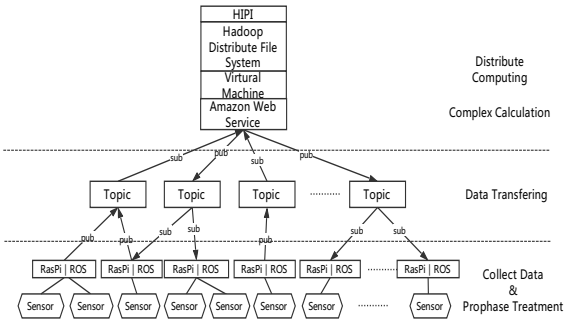


Figure 3 Proposed System

### 3.2 Load Distribute

Before distributing load, all the calculation had to be finished on Robot itself and as showed in Table 1, percentage of Robot's CPU usage was 41.4 percent. After distributing load which means the robot was just used for collecting data from camera and all the calculation was finished on server, the CPU usage was down to 1.8 percent. As the result, the CPU usage was from 41.4 percent down to 1.8 percent which means that ROS is proper for distributing load.

Table 1 Distribution Result (percent)

| %CPU | Robot (Client) | Server |
|---|---|---|
| Before | 41.4 | 0.7 |
| After | 1.8 | 24.3 |

### 3.3 Basic Evaluation of Latency

An experiment had been performed to make a comparison of ROS topic communication (based on ROSTCP) and original socket TCP communication.

We did this experiment in one machine for two reasons described as following. First, we have to exclude other influencing factors such as network latency, router's performance's effect and etc. Second, we try to use NTP (Network Time Protocol) to synchronize the time of two machines, however, it did not work as expected because NTP is also relying on internet environment. Since it is too difficult for two machines to synchronize time exactly, we have to do this experiment in one machine in order to get a precise result.

The experiment is about comparing ROS's two node communication's delay time with socket's TCP connection's delay time. For ROS, one node published 4 bytes' data with a time stamp every second and another node subscribed the topic and received data and made a comparison with the time now 500 times. The measuring time is from the publisher published message to the subscriber got a message. For socket TCP, the client sent 500 4 bytes' message with a time stamp(microsecond) to server and when server received the message, it made a comparison with the time now and then we can get the delay time with the accuracy at microsecond. The measuring time of socket TCP is from socket's send() function to recv() function.

Table 2 Latency

| | Max Latency | Average latency | Standard deviation |
|---|---|---|---|
| ROS | 881 us | 408 us | 47.2 us |
| socket | 544 us | 346 us | 83.9 us |

- Ubuntu 14.04, ROS indigo
- CPU: 2.2 GHz Intel Core i7
- RAM: 16 GB

As showed in Table 2, compared with original TCP socket, the max latency and average latency of ROS topic communication is a little higher. However, about 60 micro seconds at average latency is acceptable in the distribute robot searching system and it is not the main reason why the system was not working as expected while bringing this system to the internet. On the other hand, from standard deviation, the ROS topic communication is more stable than original TCP socket which proves that using ROS as middleware for multiple robot system is satisfied for the requirements. Therefore, latency of network is the reason why our system did not work as expected while bringing it to the internet.

## 4. Conclusion

We used ROS to build up our distribute robot searching system and the evaluation part indicates that ROS is suitable for distribute computing.

For future work, we want to develop a framework to guarantee the resource for some special application based on ROS among multiple robots.

## Acknowledgment

This research was funded by JSPS Grant-in-Aid for Scientific Research 17K 00084.

## Reference

[1] "About ROS". http://www.ros.org/about-ros/, (accessed 2017-11-16).

[2] Morgan Quigley, Brian Gerkey, Ken Conley, Josh Faust, Tully Foote, Jeremy Leibs, Eric Berger, Rob Wheeler, and Andrew Ng, "ROS: an open-source Robot Operating System", ICRA Workshop on Open Source Software, 2009.

[3] H. Durrant-Whyte, and T. Bailey, "Simultaneous localization and mapping: part I", IEEE Robotics & Automation Magazine, Volume: 13, Issue: 2, June 2006.

[4] Felix Endres, Ju¨rgen Hess, Ju¨rgen Sturm, Daniel Cremers, and Wolfram Burgard, "3-D Mapping With an RGB-D Camera", IEEE Transactions on Robotics, VOL. 30, NO. 1, 2014.

[5] Ayrton Oliver, Steven Kang, Burkhard C. Wünsche, Bruce MacDonald, "Using the Kinect as a navigation sensor for mobile robotics", Proceedings of the 27th Conference on Image and Vision Computing New Zealand, Pages 509-514, 2012.

[6] Paul Viola, and Michael Jones, "Rapid Object Detection using a Boosted Cascade of Simple Features", Conference ON Computer Vision and Pattern Recognition, 2001.

[7] Limi Kalita, "Socket Programming", International Journal of Computer Science and Information Technologies, Vol. 5 (3), 2014.

[8] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton, "Deep Learning", Nature 521, 436–444, 2015