International Journal of Smart Home
Vol. 9, No. 7 (2015), pp. 195-202
http://dx.doi.org/10.14257/ijsh.2015.9.7.20

An Enhanced Method of Face Recognition for Cloud Robot

Shuqing Tian[1], Sung Gyu Im[1] and Suk Gyu Lee[1]

[1]*Department of Electrical Engineering*
Yeungnam University, Gyeongsan, Korea 712-749
tianshuqing@gmail.com, tamop87@nate.com, sglee@ynu.ac.kr

Abstract

This paper describes a cloud-based robot system which connects cloud computing infrastructure for accessing distributed computing resources and big data and executing multitask like face detection, face recognition and etc. The ROS (Robot Operating system) has been employed as the operating system of the RC-Cloud server. We have deployed one application, "real-time face recognition application" in the RC-Cloud robot system. The experimental results show that the RC-Cloud robot system is capable of executing computation-intensive tasks for robot client efficiently and sharing information among all the clients in the system.

Keywords: *Cloud robotics, RC-Cloud robot system, cloud robot, face recognition robot*

1. Introduction

As the scale and scope of robotics continues to grow, writing software for robots is more difficult. Different types of robots can have wildly varying hardware. The source code for a robot usually contains a deep stack starting from driver-level software and including logical processing, sensor data processing, data structure and algorithms. High-speed data processing for robots are necessary. [1] Beyond that, the capabilities and architectures must also support large-scale software integration efforts. The basic requires for robot development are commonly the same, for example, image processing for all robot vision system are required, image capturing and image intensification are basic functions for this kind of systems. The implementation of the same functions for multiple robots makes unnecessary cost. In addition, for a multi-robot system, the environmental information for the robot groups is largely identical but with minor differences. Since the deviations of the sensors in each robot are different, unifying the sensor data value is also essential.

Many robotics researchers have created a wide variety of frameworks which fits the cloud-based requirement to manage complexity and facilitate rapid prototyping of software for experiments, resulting in the many robotic software systems currently used in academia and industry [2-3]. ROS (Robot Operating System) [4] is one of them which integrate robotics and cloud computing technology to enhance the performance of the robot. The philosophical goals of ROS are peer-to-peer, tools-based, multi-lingual, thin, free and open-source [5]. Our proposed RC-Cloud robot system is an implemented application in ROS. Benefiting by the distributed calculation ability of ROS, three PCs form a cloud host which allows computation to be relocated at run-time to match the available resources.

The ability to do face recognition is a key function for improving robot interaction with people. There are many approaches to implement it. A Real-time face recognition application commonly consists of image preprocessing, feature extraction, applying algorithm and feature matching. It's highly consumptive calculation for onboard robot.

ISSN: 1975-4094 IJSH

2. RC-Cloud Robot System

Using SOA (service Oriented Architecture) for robotics has certain inherent advantages over the traditional robot. The main benefit is to have a layer of common services with standard

Interfaces. Each service represents a component such as sensor, actuator, or effector. The RC-Cloud clients are a group of devices connected to the RC-Cloud server. They can be a PC, laptop, robot or even a simulated robot in the system. The RC-Cloud server is a distributed system of three connected PCs by ROS. The philosophical goals of RC-Cloud robot system can be concluded as: High-performance cloud server, large database for robot, Sharing knowledge and skills among robots,

All the robots connected to the RC-Cloud server can share information and skills. The RC-Cloud server can host a database or library of skills or behaviors that map to different task requirements and environmental complexities. In addition, there are some public web services available for robots. For example, the "Global Weather" web service [7] can be accessed by the RC-Cloud server and the information is available for all robots that connect to the server.

As a result of these advantages, the proposed RC-cloud robot system has a wide range of potential applications in data-intensive or computation-intensive tasks, such as SLAM, grasping and navigation.

In our system, the following steps have been executed to connect three machines.

a) One of the three computers has been chosen as the master.
b) All the implemented nodes in the three computers must be configured to use the same master, via ROS_MASTER_URI.
c) Set up the network for all pairs of machines.
d) Advertising each machine must itself by a name that all other machines can resolve.

After the above steps have been done, run the master machine, start a listener on master and start a talker on other machines. Then connection will be built in the three machine group.

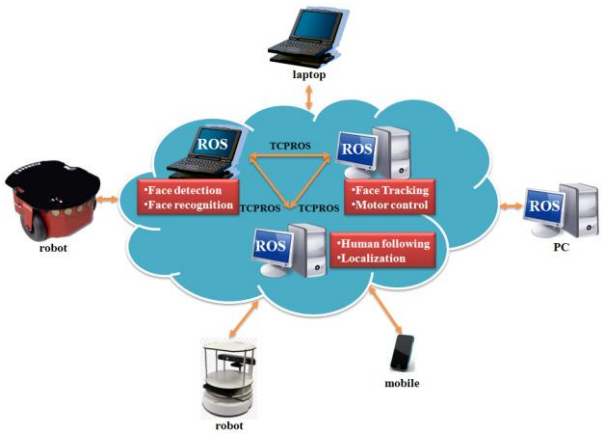


Figure 1. Delivery Model of RC-Cloud Server

Figure 1 shows the delivery model of our proposed RC-Cloud Server. In the system, there are three computers. ROS has been deployed in each computer as the hardware abstract layer. Based on ROS, various applications have been developed, such as face detection, face recognition, and so on. In order to employ the resources in RC-Cloud server adequately, different applications have been deployed in different machines to balance the computational ability and data accessing ability.

Any device supports Wi-Fi connection is capable of connecting RC-Cloud system to get and receive resources

Since employing the SOA architecture, our system has a number of outstanding characteristics :

Reusability: Services can be re-used for any robot which is authorized to connect to the system.

Substitutability: Since the deployed functions are gathered in the cloud server, the service are easily modified and updated without changing the access interface.

Extensibility and scalability: Since our system employs ROS, all resources in ROS all can be used. ROS is licensed under an open source, BSD license. In addition, more than 2000 libraries are available in ROS [12], it's easy to employ the libraries and combine them to make new libraries.

Customizability: The more than 2000 available libraries are all open source in ROS. Once the library has been downloaded, all the source code are also available. Any library can be modified and published.

Composability: More complex composite services can be constructed from existing services. For example, in the face recognition service, before recognizing the face, face detection should be employed first. And based on the face recognition service, the face tracking is possible.

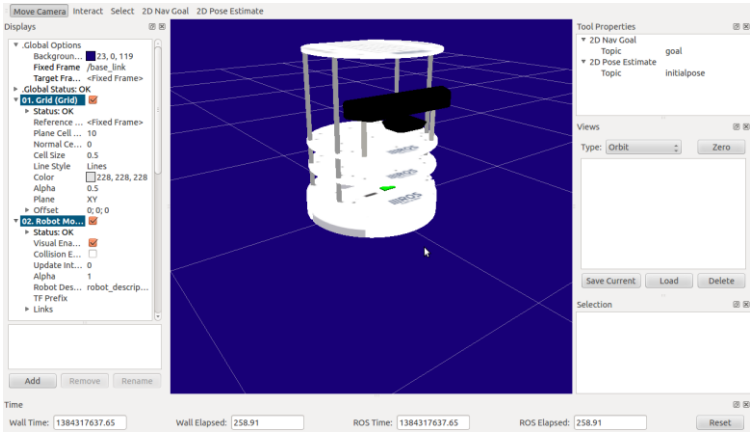


Figure 2. The Simulated Turtlebot Robot in Rviz Simulator

Figure 2 shows the simulated TurtleBot robot in Rviz. In ROS, the package "robot_model" contains a few packages for modeling various aspects of robot information, specified in the Xml Robot Description Format (URDF). The core package of this stack is "urdf", which parses URDF files, and constructs an object model of the robot. A number of different packages and components make up "urdf" package. Figure 5 shows the relationship between these components.

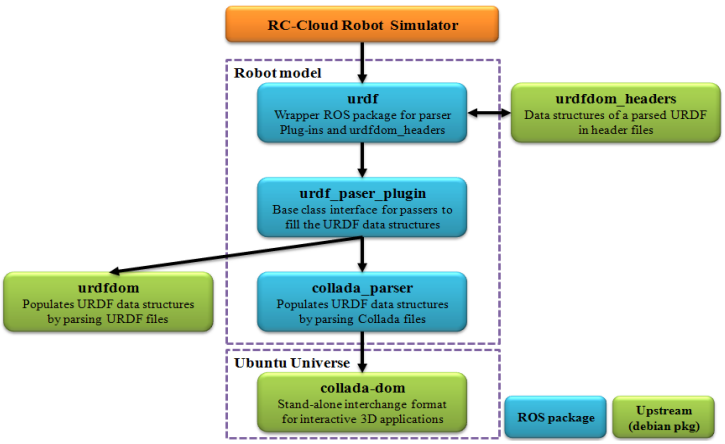


Figure 3. The Relationship between the Packages and Components in "URDF" Package

We've implemented a real robot and a simulated robot client for RC-Cloud Server. The robot is based on the Pioneer P3-DX robot which is the world's most popular research mobile robot. The Pioneer's versatility, reliability and durability have made it the reference platform for robotics research. Unlike hobby and kit robots, Pioneer is fully programmable, and will last through years of tough classroom and laboratory use. The simulated robot is a Pioneer P3-DX model which supports the Pioneer P3-DX robot interface.

3. System Implementation

The pioneer robot requires a serial communication link with a client for operation. In our implementation, we've implemented the Laptop-Pioneer connection model. The laptop with ROS installed in it plays a role of "brain" for the robot.

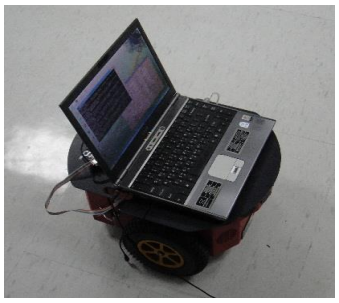


Figure 4. Implementation of Laptop-Pioneer Connection

The simulated pioneer robot is also a client for RC-Cloud system. The simulated pioneer publishes its topics for other robot in the system and subscribes necessary topics to get useful information.

For the connection between robots and the server, the Wi-Fi protocol is employed. Wi-Fi is usually access point-centered, with an asymmetrical client-server connection with all traffic routed through the access point, while Bluetooth is usually symmetrical, between two Bluetooth devices. However, Bluetooth access points do exist and ad-hoc connections are possible with Wi-Fi though not as simply as with Bluetooth. Wi-Fi Direct was recently developed to add more Bluetooth-like ad-hoc functionality to Wi-Fi.

By using RC-Cloud server and the applications in it, many sophisticated applications are capable to be implemented. A face recognition application has been deployed.

OpenCV 2.4 comes with the very new FaceRecognizer class for face recognition, the currently available algorithms are: Eigenfaces, Fisherfaces, Local Binary Patterns Histograms. The "vision_opencv" stack in ROS provides packaging of the popular OpenCV library for ROS. For OpenCV vision_opencv provides several packages:
cv_bridge: Bridge between ROS messages and OpenCV, as shown in Figure 4, it converts between ROS Image messages and OpenCV images.

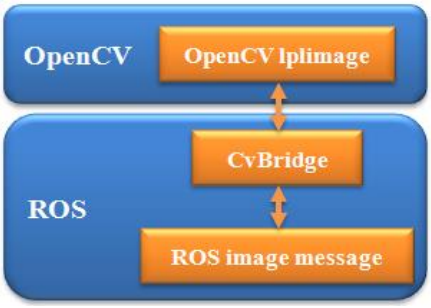


Figure 5. Communication Model between Open CV and ROS

image_geometry: It interfaces the calibration parameters in sensor_msgs/CameraInfo messages with OpenCV functions such as image rectification, much as cv_bridge interfaces ROS sensor_msgs/Image with OpenCV data types.

ROS passes around images in its own "sensor_msgs/Image" message format, but to do the face recognition, the images in conjunction with OpenCV are required. CvBridge is a ROS library that provides an interface between ROS and OpenCV. CvBridge is in the "cv_bridge" package in the "vision_opencv" stack.

4. Experiments

The FaceRecognizer has been employed to do the face recognition. The face recognition application in the RC-Cloud robot system is called "face recognition" node. In order to communicate with the "face recognition" node, we've implemented a "face recognition client" node. The client node communicates with the "face recognition" node by sending command. As corresponding with the commands, the "face recognition" node executes "acquire training images" task, "do face recognition" task or "exit the system" task. The flowchart of this application is shown in the figure below.

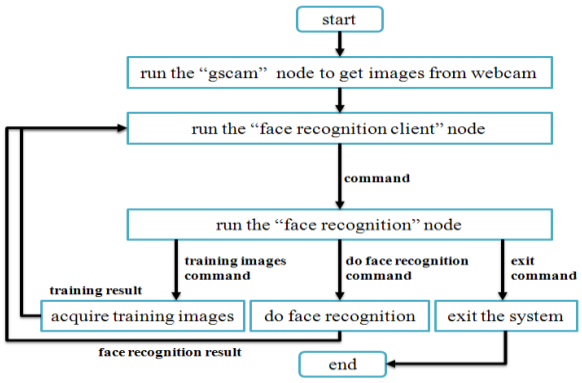


Figure 6. Face Recognition Application Flowchart

As shown in Figure 8, in this experiment, there are three enrolled people in the database who are "StevenZhou", "EricSong" and "ShuqingTian". The images data in the database are shown in Figure 20.

Figure 7. Training Face Images for Enrolled Person

In order to enhance the recognition ratio, there are 26 face images stored for each person enrolled in the system. Once the application starts, the training data (face images) will be loaded. As shown in the figure, 75 training images of 3 people are loaded in the beginning.

There are "face recognition result", "face recognition feedback" and "face recognition status". For the "face recognition client" application, it subscribes all the topics published by "face recognition" application. In the meanwhile, it also publishes its own topics to communicate with the "face recognition" application. The two topics are "face recognition cancel" which tells the "face recognition" application to stop and "face recognition goal" which tells the "face recognition" application to do face recognition or enroll a new person. There is a "fr_order" topic in the system which is a interface for users to send command to the "face recognition clienet" and then indirectly control the "face recognition" node.

The False Rejection Rate (FRR) and False Acceptance Rate (FAR) are used as performance metrics for biometric systems. The FRR is the measure of the likelihood that the biometric security system will incorrectly reject an access attempt by an authorized user. The FAR refers to the measure of the likelihood that the biometric security system will incorrectly accept an access attempt by an unauthorized user.

The FRR and FAR depends on the threshold in the face recognition system. The threshold value is used to indicate the accuracy of the recognition result. The range for threshold is form 0 to 1. We've set it to be 0.80 in the program. As shown in the figure bellow, with the increase of threshold value, the FRR increase and the FAR decreases. In this case, it's "difficult" to recognize a face but once the face is recognized, the result is more reliable. Once the threshold value decreases, it becomes "easy" to recognize a face, however, the recognized face may be not the right person that means it's less reliable.

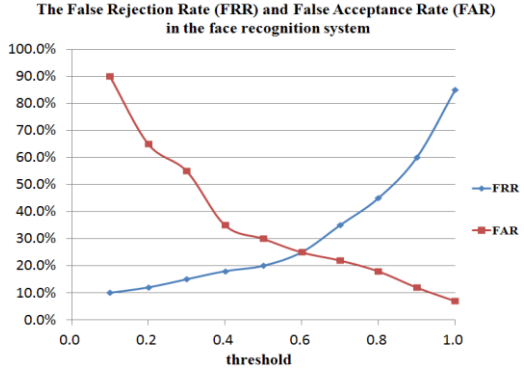
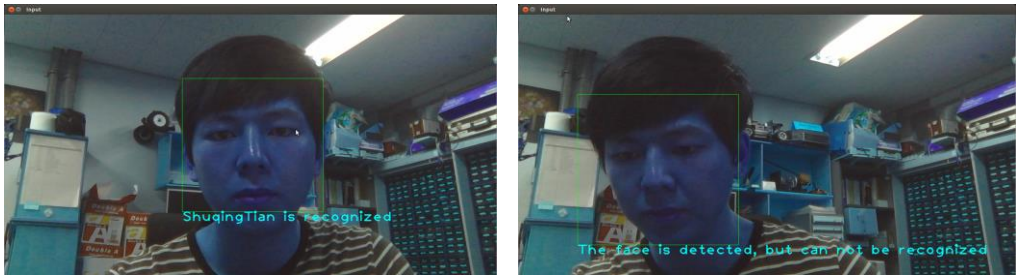


Figure 8. The False Rejection Rate and False Acceptance Rate in the Face Recognition System



a) b)

Figure 9. a) Person Recognized Case b) Person Not Recognized Case

Since we've set the threshold 0.8 in the application, the lowering head or rotating head will cause no recognition. In the application, with the threshold is set to be lower, the face recognition ratio increases. If the threshold is higher, as shown in our case, the recognition ratio lowers; however, the recognition accuracy has improved.

5. Conclusion

This paper proposes a noble design and implementation of a cloud based robot system, the RC-Cloud robot system. Based on the cloud robotics technology, robots are capable of executing computation-intensive tasks, such as face recognition, SLAM and grasping.

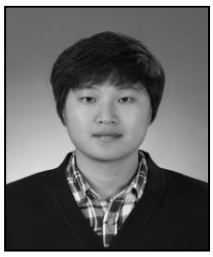
Based on the ROS, we've implemented our RC-Cloud robot system which consists of RC-Cloud server and RC-Cloud client. The RC-Cloud server is a tiny cloud server of 2 computers. The ROS system is deployed on the 2 computers which allow the 2 computers to execute distributed task as a union. Some packages in ROS have been employed to provide necessary interfaces for the RC-Cloud server, such as the OpenCV library, the "Rviz" and "Stage" simulators.

References

[1] "Robotics History Project", IEEE Robotics & Automation Society, http://www.ieee-ras.org/educational-resources-outreach/robotics-history-project.
[2] T. Balch, R. C. Arkin, "Behavior-based formation control for multi-robot teams", Robotics and Automation, IEEE Transactions, vol. 14 , no. 6, **(2002)**.
[3] J. Kramer and M. Scheutz, "Development environments for autonomous mobile robots, A survey", Autonomous Robots, (2007).
[4] ROS wiki : http://wiki.ros.org/
[5] Q. Morgan, G. Brian, C. Ken, F. Josh, F. Tully, L. Jeremy, B. Eric, W. Rob and N. Andrew, "ROS, an open-source Robot Operating System", ICRA Workshop on Open Source Software, **(2009)**.
[6] Y. I. Kato, T. Tsuchiya, Y. Narita, M, Ueki, M. Murakawa and Y. Okabayashi, "RSi-cloud for integrating Robot Services with internet services", IECON 2011 - 37th Annual Conference on IEEE Industrial Electronics Society, **(2011)**.
[7] H. Guoqiang, P. T. Wee and W. Yonggang, "Cloud robotics, architecture, challenges and applications, Network, IEEE, **(2012)**.
[8] "Robots Using ROS", http://wiki.ros.org/Robots
[9] "Possibilities for Robot Applications", : http://www.willowgarage.com/pages/pr2/applications
[10] TurtleBot: http://turtlebot.com/
[11] C. Song, Z. Tao, Z. Chengpu and C. Yu, "A real-time face detection and recognition system for a mobile robot in a complex background", Artificial Life and Robotics, **(2010)**.
[12] C. Cruz, L. E. Sucar and E. F. Morales, "Real-time face recognition for human-robot interaction", Automatic Face & Gesture Recognition, 2008. FG '08. 8th IEEE International Conference, **(2008)**.
[13] "GlobalWeather webservice", http://www.webservicex.com/globalweather.asmx?WSDL
[14] Simultaneous localization and mapping: http://en.wikipedia.org/wiki/Simultaneous_localization_and_mapping
[15] S. Ashutosh, D. Justin and Y. N. Andrew, "Robotic Grasping of Novel Objects using Vision", International Journal of Robotics Research (IJRR), vol. 27, no. 2, **(2008)**, pp. 157-173.
[16] "Mobile robot navigation", http://en.wikipedia.org/wiki/Mobile_robot_navigation

Authors

Shuqing Tian, received the B.S. degree in information Security Engineering from Yunnam University, China, in 2009, and he received the M.S. degree in Robotics Engineering from Yeungnam University in 2004. His research interests include robotics, intelligent device, and biometric recognition.



Sung Gyu Im, is currently a graduate student in M.S program in Department of Electrical Engineering, Yeungnam, University. His research interests include mobile robotics, and SLAM.

Suk Gyu Lee, received the B.S. and M.S. degree in Electrical Engineering from Seoul National University in 1979, and 1981 respectively, and he received the Ph.D. degree in Electrical Engineering from UCLA in 1990. His research interests include robotics, SLAM, nonlinear control and adaptive control.