International Journal of Multimedia and Ubiquitous Engineering
Vol. 9, No. 10 (2014), pp. 251-260
http://dx.doi.org/10.14257/ijmue.2014.9.10.24

Real-time Face Recognition for Cloud Robot

ShuqingTian, DilshatSaitov and Suk Gyu Lee

Yeungnam University, Korea
tianshuqing@gmail.com, dilshat_saitov@yahoo.com, sglee@ynu.ac.kr

Abstract

This paper discusses our work with the design and implementation of real-time face recognition for cloud robot. Face recognition has been widely used for surveillance and airport security. Robot systems which support face recognition are available too. As for robots, real-time feedbacks from sensors are always required. In order to recognize a face in real time, the images captured by camera have to be stored and then processed by face recognition algorithm. This processing procedure runs several times per second which is computationally intensive tasks for usual robot systems. Benefit from cloud computing technology, robot can offload computational tasks to cloud server which composes of several computational units (PCs) and is capable of executing distributed tasks. In our approach, real-time images will be captured and sent back to the cloud server. PCs in cloud side process the images in parallel. Face detection and recognition algorithm will be applied to each image and then the recognition results will be returned to the robot as the response. Our experimental results demonstrate that as a implementation of cloud robotics system, the proposed cloud robot is capable of doing real-time face recognition.

Keywords: *cloud robot, cloud computing, real-time face recognition, cloud robotics*

1. Introduction

Robots have been widely used in the industry, healthcare and daily life which have brought significant economic and benefits human lives. The reliability and stability of the robots make it possible replace mortal to do tedious, repetitive, or dangerous tasks, such as assembly, packaging and welding. With the wide use of robots, some constraints come out. 1) For the traditional embedded robots, skill updating means rewriting program into the onboard chips. 2) Each robot is a single unit in robot system and there is no effective mechanism to share information among robots. 3) Computational capability of onboard system is limited. With development of cloud computing technology, these constraints can be overcome by the Cloud Robotics technology.

Cloud robotics is the combination of robots and cloud computing technology. Cloud computing technology allows robots to benefit from the powerful computational, storage, and communications resources of modern data centers. In addition, it removes overheads for maintenance and updates, and reduces dependence on custom middleware [18].It provides the following advantages.

- Ability to offload computation-intensive tasks to the cloud. The computation ability of the traditional robots relies on the embedded chipsets. Ordinarily, high performance robots have to carry bulky batteries to keep the electric output which make the robots be in a large size. Benefit by the cloud computing technology, the robots only has to keep the necessary sensors, actuators. The battery life is extended, and the robotic platform becomes lighter and less expensive with easier to maintain hardware. The maintenance of software onboard the robots also become simpler.

- Access to vast amounts of data. Via the cloud storage technology (a core technology of cloud computing), the robots can acquire information and knowledge to execute tasks through databases in the cloud server. Even more, robots can share information and learn new skills from each other.

Robots with face recognition application have been developed and used in practice. In the paper [15], a novel approach for face recognition using multiple face patterns obtained in various views for robot vision was introduced. The paper [16] presents a design and experimental study of human-robot interaction via face recognition and image tracking. However, these implementations are limited since the computational ability of embedded robot. Benefit by cloud robotics technology, cloud robot with real-time face recognition application comes true. In our approach, the challenge of long latency between the robot and the cloud server has been solved which makes images be transmitted in real time. On the cloud server side, 5 distributed computers were connected by using the robot operating system (ROS). Captured images can be processed individually in different computers at the same time.

The remainder of the paper is organized as follows. The architecture of overall system will be introduced in Section 2. Face recognition algorithm and implementation are given in Section 3. In the Section 4, the hardware implementation is presented. The experimental results are shown in Section 5. And we will conclude the paper in Section 6.

2. Related Works

A great number of cloud robotics applications have been implemented. The RoboEarth [3] is a giant network and database repository where robots can share information and learn from each other about their behavior and their environment. Google engineers developed android-powered robot software that allows a smart phone to control robots based on platforms like Lego Mindstorms, iRobot Create, and Vex Pro [4]. The DAvinCi is a software framework that provides the scalability and parallelism advantages of cloud computing for service robots in large environments [5]. The RSi Research Cloud (RSi-Cloud) enables integration of robot services with Internet services and mash-up of robot services [6]. Researchers from China designed a framework of "Robot Cloud Center" (RCC) following the usual cloud computing paradigm to address the current limitations in capacity and versatility of robotic applications [7].

Cloud-based face recognition system has been implemented and presented in [20], a mobile-cloudlet-cloud architecture called MOCHA as a platform for our target face recognition application.

In this paper, we proposed our method on implementing a cloud-based and real-time face recognition system for robot. The work we have done is a new study of cloud robotics.

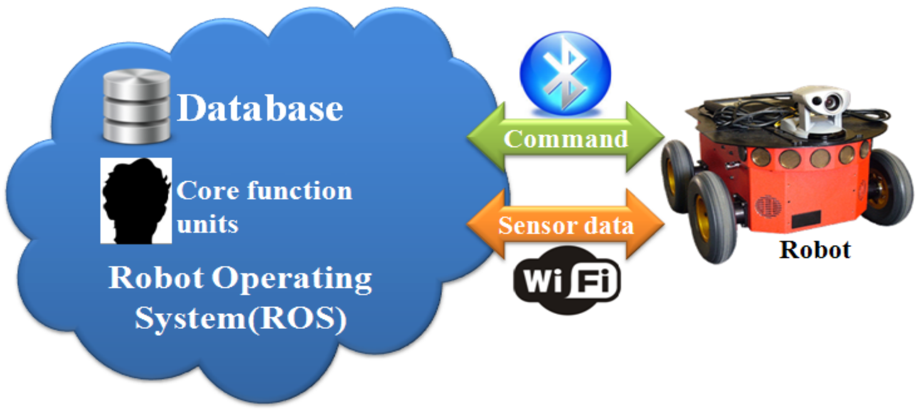
3. The Architecture of Implemented Cloud Robot

In the implemented architecture, cloud server plays a role of “brain” in the system. However, one of the well-known challenges for using the cloud as a server is the long latency between the mobile device and the cloud server in comparison to localized computing and small-scale distributed computing called cloudlet [16].Given this challenge, the proposed architecture must support high-speed data transmission and distribute computing load to achieve the minimal response time. Regard to data transmission, short range wireless communications have to be considered.

As a solution for cloud robot, the flexibility and extensibility are considered. Meanwhile, for real-time face recognition, the bandwidth is more important than transmission distance. In addition, scope of the protocol being used is also essential. Therefore, the Wifi protocol is employed to transmit image data in our approach. Wifi has a transmission range of 35 meters indoors and 100 meters outdoors and the 11Mbps bandwidth is also capable for image transmission. Above all, the use of Wifi is widest among the listed protocols. Bluetooth as a common-used protocol is used for send command between robot and cloud server in our system.

The architecture of our cloud robot is shown in figure 1.The robot operating system has been used as the operating system for cloud server. The implemented face recognition algorithm is an executable program which is called “node” in ROS. All developed libraries are stored in the “core function units”, which is separated from ROS build-in functions. Necessary information is stored in database, such as the system configuration, user information, command list and etc...In the client side, the robot is supposed to support Wifi and Bluetooth communication. For using Wifi protocol, some sensor data like images, ultrasonic data can send back to cloud server in real-time. The Bluetooth protocol in our system is used to transmit command between robots and server.

Each component in the architecture will be discussed detailedly as follows:



A. Robot Operating System (ROS)
ROS (Robot Operating System) provides libraries and tools to help software developers create robot applications. It provides hardware abstraction, device drivers, libraries, visualizers, message-passing, package management, and more. ROS is licensed under an open source, BSD license [17]. In our case, 5 PCs have been connected as a distributed cloud server. The cloud server running ROS is capable of processing face recognition algorithm rapidly and interacting with robot client in real time.

B. Core function units
Based on ROS, we implemented the face recognition algorithm. The executable face recognition program is called “node”[17] in ROS. The robot client sends image data and requests to this model where the actual program runs on 5 computers in parallel, and the results are sent back to the clients. In our future work, more algorithms and functions will be implemented, and they will be managed as individual library in the core function units.

C. Database
The SQLite [] has been employed in the ROS as the database instance. The system configuration, user information and command list are stored in the database. In “core

function units", face recognition has been implemented. Template images and Eigen face data are stored in the database.

D. Sensor data

In our implementation, the Pioneer robot [19] integrated with a webcam forms the robot client. Real-time images with the size of 640*480 will be captured by webcam per second and 10 of them will be transferred to the cloud server by using Wifi protocol. There are 8ultrasonic sensors plugged in the Pioneer robot, the ultrasonic data will be collected to indicate the robot avoiding obstacles.

E. Command

We have defined some commands to control and get feedback from robot. The commands are listed in Figure 2.

Command code	Keyboard input	Function
0x01		Robot active request
0x02		Disconnecting robot
0x03		Successful connection
0x04		Failed connection
0x30	w/W	Robot goes forward
0x31	s/S	Robot backs up
0x32	a/A	Robot turns left
0x33	d/S	Robot turns right
0x34	Space	Robot stops
0x40	1	Turn on webcam
0x41	2	Turn off webcam
0xb0		Face detection succeeds
0xb1		Face recognition succeeds

Figure 2. System Command Definition

In the command list, we define the 0x01~0x29 as the system reserved command code. System status information will be described by these codes. 0x30~0x39 codes are reserved to control the robot and 0x40~0x50 are reserved for webcam. The function description code starts from 0xb0, and now in the system, "face detection" and "face recognition" have been defined.

F. Robot client

One or more robots with webcam can form the client side. Each robot in the client side is an individual. The robot communicates with the cloud server and exchanges information. Once one robot captures some information from sensor, the information will be send back to the cloud server and then send to other robots. In this way, the information is shared among robots. In our system, we have a Pioneer robot in the client side and a simulated Pioneer in the computer. The images captured by Pioneer robot are sent back to the cloud server, and then face recognition algorithm is applied. The face recognition result will send back to the Pioneer robot and can be used by simulated Pioneer.

4. Face Recognition Algorithm and Implementation

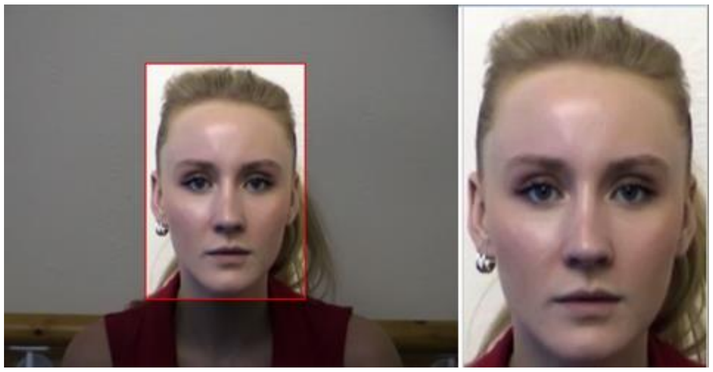


Figure 3. Face Detection Result

A. Face Detection

As shown in Figure 3, the face detection determines the potential locations of the human faces within an image. We have utilized the Haar Features and Haar Classifiers described in [14] to perform face detection. This iterative approach begins with fairly primitive classifiers that group potential face candidates based on a small number of features. These simple classifiers in this initial stage have low computational complexity but must operate on a large amount of data, and they produce a large number of face candidates. The algorithm then progressively eliminates some of these candidates by using increasingly more sophisticated classifiers based on additional features at successive stages of the detection pipeline, such that the final stage outputs the detected faces with high confidence. Although the number of remaining candidates is significantly less at each successive stage, the complexity of the calculations increases at almost the same rate, and thus the overall computational complexity of each pipeline stage of this detection algorithm stay somewhat constant. In our implementation, we use a 32-stage pipeline.

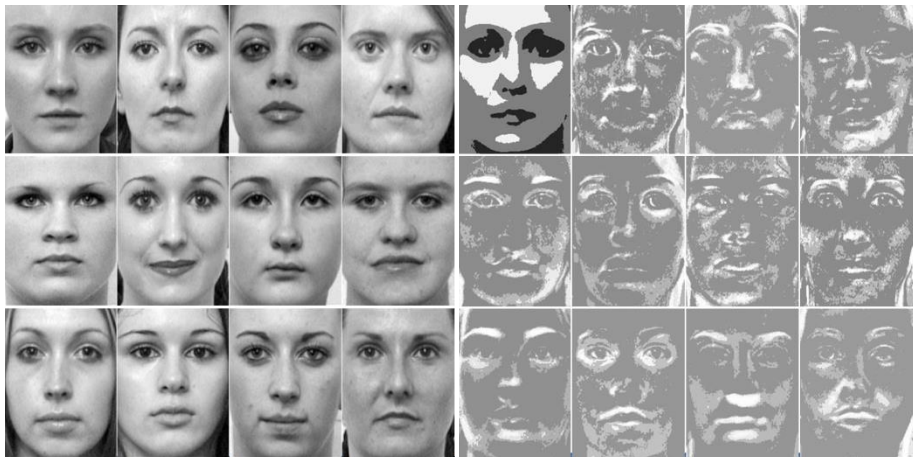


Figure 4. Template Image and Eigenface

B. Face Recognition

The face recognition determines the match-likelihood of each face to a template element from a database. The potential locations of the faces determined in the previous face detection phase are fed into this phase for recognition. The recognition algorithm yields one of a few potential results for each face candidate determined by the detector: (1) not a face, (2) a face, but not in the database, and (3) a face and in the database. We have employed the widely-accepted Eigenfaces approach [13], which calculates an orthogonal set of M Eigenfaces for a given training set of N faces, where M N. Figure 5 shows the 12Eigenfaces calculated for a set of 20images (i.e., N = 20 and M = 12). Thus each face from the original N faces can be represented as a point within the M dimensional space spanned by the M Eigenfaces. This permits a significant reduction in the amount of computation that has to be performed to recognize a face within a given database, as well as a significant reduction in the amount of storage required for the template images (database). To recognize a face, the algorithm simply transforms the face into a point within the M-dimensional space spanned by the Eigenfaces and calculates the Euclidean distances between the point of the face to be detected and the points of all template faces from the database. If the Euclidean distance is above a large threshold, meaning that the detection algorithm yielded a false positive. Otherwise, if the Euclidean distance is above a small threshold to all the template faces, the algorithm determines that the face is not in the database (result (2)). In this case, if desired, this face can be added to the database by re-calculating the Eigenfaces and the Eigenfaces representation of the newly introduced face and updating the databases of all the cloud servers. If the Euclidean distance to one of the template faces is below a small threshold, the algorithm detects a match for the face (result (3)).

5. Experimental Results

Our experiment hardware platform is a distributed heterogeneous cluster of 5 servers running Ubuntu 12.10. The ROS has deployed in each server to execute distributed calculation task in parallel. Our development platform is Eclipse C++, Sqlite database and Open CV libraries for development of our face detection and face recognition programs.

A. Face Recognition Experiment

In the client side, the Pioneer robot has been integrated with a camera. As shown in Figure 5, the robot captures images of the user and the onboard-CPU sends the image data to the cloud server.

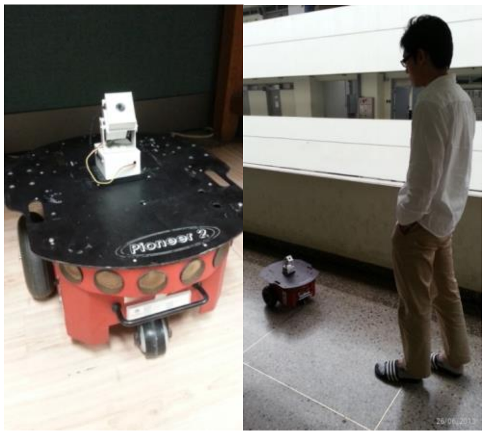
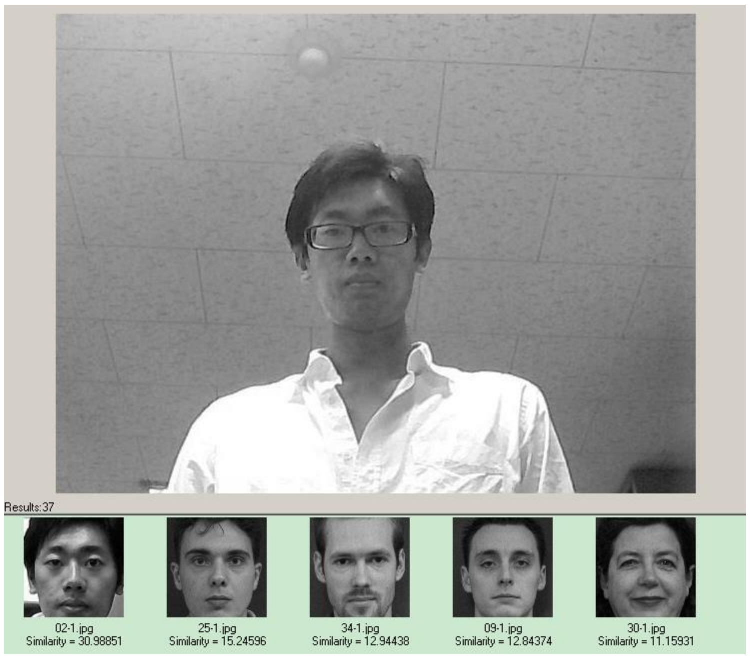


Figure 5. Robot Client



Results:37

02-1.jpg
Similarity = 30.98851

25-1.jpg
Similarity = 15.24596

34-1.jpg
Similarity = 12.94438

09-1.jpg
Similarity = 12.84374

30-1.jpg
Similarity = 11.15931

Figure 6: Face Recognition Result

The robot captures images in real-time. The more images have been saved the longer latency will be. In our approach, due to the performance, we saved 10 images per second. The Face recognition result is shown in figure 6.The similarity will be calculated between captured image and template images. In this way, the function of face recognition has been validated.

B. Sharing Information among Multiclients

Another advantage of cloud robot is sharing information. In our system, the face recognition result can be shared among robot clients. There is one Pioneer robot in the client side and another simulated Pioneer robot run in the ROS system (shown in Figure 7).

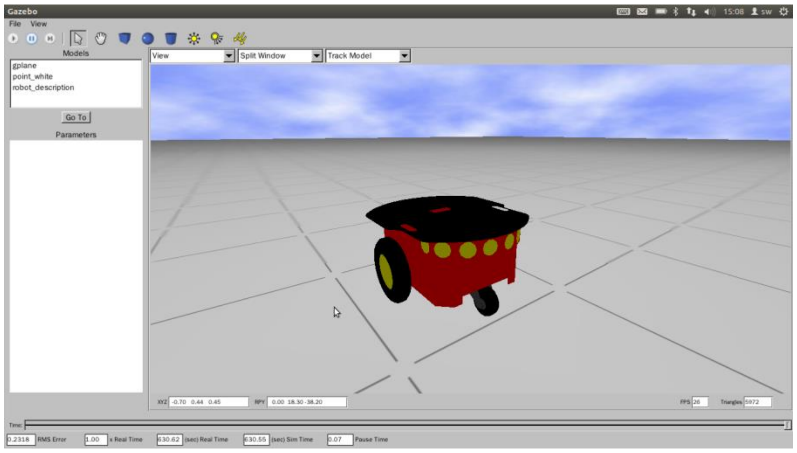


Figure 7. Simulated Pioneer Robot in ROS

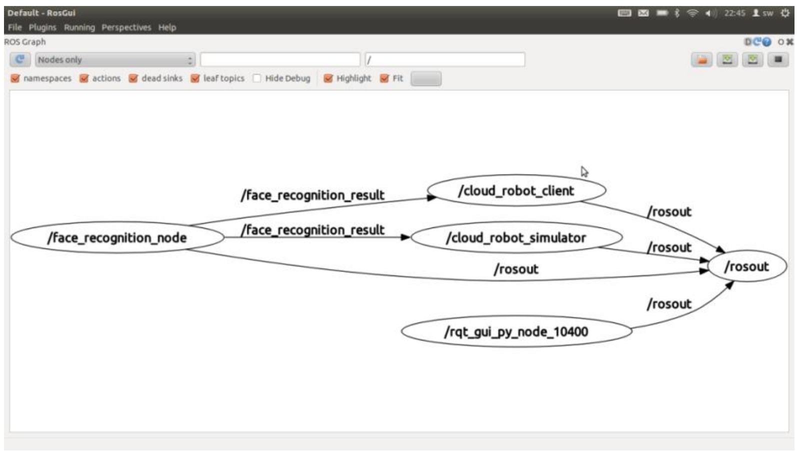


Figure 8. Nodes in ROS Graph

The face recognition result is published as a "publisher node" in ROS system (shown in Figure 8). Node is executable program in ROS system. The "face recognition node" sends message to a "topic" which is "face recognition result". The Pioneer robot client and simulated robot are nodes in the ROS system. Both of them subscribe the "face recognition result "topic. Once the face recognition task has been processed, the "face recognition node" will publish the result to the topic and whichever subscribed the topic can receive the result. In this way, any client subscribed the "publisher node" can share the information.

In our experiments, information sharing among different robot client has been implemented. The real-time face recognition has been executed too.

6. Conclusion

We have presented a cloud robot which excuses face recognition task. The architecture of presented cloud robot is an implemented prototype of cloud robotics. In the architecture, the ROS (Robot Operating System) has been used to execute distributed tasks, like face recognition in our project. We have taken the advantage of cloud robotics which offloads computational intensive tasks form robot to cloud server. Robot supporting Wifi protocol can be the client side in our proposed system and communicate with the cloud server. In addition, if the robot carries a camera and is capable of sending image to the cloud server, face recognition function can be executed. In our approach, the Pioneer robot has been used and a on board camera has plugged in it. The Wifi protocol has been employed to transmit real-time images to cloud server and the face recognition results and commands sent through Bluetooth protocol. The experimental results show that 1)face recognition for cloud robot has been implemented 2)delay of transferring data between cloud server and robot client is little, which appears to be real-time.3)the proposed cloud server is capable of multi-clients. We plan to extend the proposed project to integrate more algorithms and robots. Our future work also includes extension of cloud server, linking more computers to excuse distributed tasks.

References

[1] G. Hu, W. P. Tay and Y. Wen, "Cloud robotics: architecture, challenges and applications", Network, IEEE, vol. 26, (**2012**) May-June, pp. 21-28.
[2] J. M. Quintas, P. J. Menezes and J. M. Dias, "Cloud Robotics Towards Context Aware Robotic Networks", Proceedings of the IASTED International Conference, (**2011**) November 7 - 9, Pittsburgh, USA.
[3] RoboEarth, http://www.roboearth.org/

[4] Google cloud robotics: http://www.google.com/events/io/2011/sessions/cloud-robotics.html
[5] A. Rajesh, R. E. Vikas, B. B. Liu, X. J. Wu, B. Krishnamoorthy, F. K. Foong, A. K. Senthil, D. M. Kang and W. K. Goh, “DAvinCi: A Cloud Computing Framework for Service Robots.In: IEEE International Conference on Robotics and Automation,Anchorage, Alaska, USA **(2010).**
[6] K. Yuka, I. Toru, T. Yosuke, N. Masahiko, U. Miwa, M. Yoshihiko and O. Keijyu, “RSi-Cloud for Integrating Robot Services with Internet Services”, In: 37th Annual Conference on IEEE Industrial Electronics Society, Tokyo, Japan **(2011).**
[7] D. Zhihui, Y. Weiqiang, C. Yinong, S. Xin, W. Xiaoying and X. Chen, “Design of a Robot Cloud Center”, In: Tenth International Symposium on Autonomous Decentralized Systems, Beijing China **(2011).**
[8] Model–view–controller: http://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller
[9] Universal asynchronous receiver/transmitter,\url{http://en.wikipedia.org/wiki/UART}
[10] OpenCV, \url{http://opencv.org/}
[11] E. W. Neerja, “Face Recognition Using Improved Fast PCA Algorithm”, In 2008 Congress on Image and Signal Processing, vol. 1, **(2008**), pp.554-558, Washington, DC, USA.
[12] M. Pakdaman, “A line follower robot from design to implementation: Technical issues and problems”, In Computer and Automation Engineering (ICCAE), 2010 The 2nd International Conference, Babol, Iran **(2010).**
[13] M. Turk and A. Pentland, “Eigenfaces for recognition,” Journal of Cognitive Neuroscience, vol. 3, no. 1, (**1991**), pp. 71–86.
[14] P. Viola and M. J. Jones, “Robust real time face detection”, In Second international Workshop on Statistical and Computational Theories of Vision - Modeling, Learning, Computing, and Sampling”, (**2001**) July, pp. 1–25.
[15] Face Recognition Using Multi-viewpoint Patterns for Robot Vision, 11th International Symposium of Robotics Research **(2003)**, pp.192-201, 2003, Kazuhiro Fukui and Osamu Yamaguchi
[16] Corporate Research and Development Center, TOSHIBA Corporation
[17] The Case for VM-Based Cloudlets in Mobile Computing, Mahadev Satyanarayanan,
[18] Pervasive Computing, IEEE vol. 8, no. 4.
[19] ros node reference
[20] http://www.roboearth.org/cloud_robotics
[21] http://www.mobilerobots.com/ResearchRobots/PioneerP3DX.aspx
[22] Cloud-Vision: Real-time Face Recognition Using aMobile-Cloudlet-Cloud Acceleration Architecture.