ISSN 1598-0170 (Print)
ISSN 2287-1136 (Online)
http://www.jksii.or.kr

클라우드 로봇을 위한 적응형 그룹통신 기반 정보공유 모델

Information Sharing Model based on Adaptive Group Communication for Cloud-Enabled Robots

마테오 로미오[1] 이 재 완[2*]
Romeo Mark Mateo Jaewan Lee

요 약

클라우드 로봇에서, 효율적인 정보공유 모델은 연구과제이다. 본 논문에서는 클라우드 기반 로봇들이 서로 연동하고 지식을 공유하기 위한 지식공유 모델을 제시한다. 효율적인 메시지 전송을 위해 멀티 에이전트 기반 적응형 그룹 통신을 제시하며, 연결(link) 노드는 주요 링크를 결정하기 위한 비중함수를 사용한다. 성능평가 결과, 제시한 알고리즘이 최소 메세지 오버헤드를 나타내었으며, 전통적인 그룹통신 방법에 비해 주요 링크를 사용함으로 인해 질의에 대한 응답이 빠름을 보였다.

☞ 주제어 : 클라우드 컴퓨팅, 클라우드 로봇, 적응형 그룹통신

ABSTRACT

In cloud robotics, the model to share information efficiently is still a research challenge. This paper presents an information sharing model for cloud-enabled robots to collaborate and share intelligence. To provide the efficient message dissemination, an adaptive group communication based on multi-agent is proposed. The proposed algorithm uses a weight function for the link nodes to determine the significant links. The performance evaluation showed that the proposed algorithm produced minimal message overhead and was faster to answer queries because of the significant links compared to traditional group communication methods.

☞ keyword : Cloud computing, cloud robotics, adaptive group communication

1. Introduction

The cloud-enabled robot, or cloud robotics" which is a term for some developers and researchers [1], is a recent topic in the Cloud. Cloud-enabled robots are mainly designed to access services in the Web to provide smart functions to robots. E.g., in disaster support, the perception of a person who is in danger involves a large volume of data images and/or a complex algorithm, can be processed using the resources in the Cloud or use a cloud service to process this function. Moreover, a high complex computation to process a task quickly drains the battery life of a robot. However, if processed in a computer with faster processor and stable power source, the computational time and energy consumption will be efficient. Google introduced cloud robotics in [2] to enable cloud services to be used by robots. They emphasized that the computer processor consumes more power than the motors of a PR2 robot. It is common for service robots to be used for a specific domain or application classified in [3]. An example in the study of Mukai, et al [4], a robot provides nursing care for patients and its function is specialized that it cannot perform tasks in other domain. In spite of the differences on its domain, with common components, e.g. sensors, arms, wheels, etc., robots can perform each other function by sharing intelligence among the service robots and thus there is no need to configure or train a robot to understand the unknown perception. This is the goal of our proposed information sharing model where the appropriate components for the sharing information are tackled. Also, the knowledge of

1 Research & Development Team, NEXUSCOMMUNITY, Seoul, 152-724, Korea
2 Dept. of Information and Telecommunication Engineering, Kunsan National University, Gunsan, 573-701, Korea
* Corresponding author (jwlee@kunsan.ac..kr)
[Received 21 May 2013, Reviewed 30 May 2013, Accepted 1 August 2013]
☆ This research is partially supported by the Institute of Information and Telecommunication Technology of KNU

robots can vary over time where robots with more information has higher probability to answer queries, on contrary, a reply with no answer is considered as unnecessary message overhead explained in of Khan, et al [5]. Therefore, there is a necessity to adjust the communication links from robots.

In this paper, a model of information sharing for cloud-enabled robots is presented and then the algorithm that provides efficient collaboration and information sharing in the Cloud is proposed. To efficiently disseminate the information in the virtual group of robots, an adaptive group communication based on multi-agent approach is proposed. A node link weight value which is based on the calculation of a Brownian agent approach determines the significant links to a robot agent in a group. An implementation of the proposed information sharing model is also shown in this paper.

2. Cloud-Enabled Robots and Group Communication

The network of robots provided an efficient way of sharing data and collaboration in [6, 7]. Most of the studies used the network capabilities in mobile robots and information sharing for coordination like in [8, 9, 10, 11]. Researchers and developers studied the use of robots in different applications but the functionalities are still limited because of the need in storing large amount of information. The research challenges in complex analysis and optimization methods specified in [12] are also possible by using Cloud. Extending the functions of network robots to cloud will enhance the capability to process information and extend the functionalities by sharing information among robots. Using this concept, the interoperations of robots and sharing of information are still a challenge by researches. Having standard of sharing information using a message based commands like an approach in ROS [13] can solve this issue. In [14], a robot framework utilized a cloud computing environment to parallelize the computation of mapping. The efficient message passing for collaboration from the mentioned previous researches was not considered. An efficient method for communication by minimizing the message overhead in queries and accurate in finding the robots with correct information is considered in this paper.

Group communication is useful in many fields such as telecommunications and robotics which have various considerations. In [15], a group communication is used by mobile robots where a mesh construction and pruning are integrated to improve the multicast group communication. The main challenge presented in this research is how the information is disseminated in all robots to coordinate each robot movement in real time. In peer-to-peer or multipeer, the communication takes place when several senders are able to send user data to the group members. In [16], a communication protocol for multi-robot system is based on multi-agent approach where the protocol ensures the understanding of intentions. The design enables sharing data and resources on a very large scale by eliminating any requirements for separately managed servers and their associated infrastructure. However, the knowledge of robots in the group can vary over time where the frequently updated robots can answer more queries while robots with low frequent updates have low probability to answer queries. In this paper, we used the multipeer based on a published-subscribe paradigm and the group communication is based on multi-agent approach to adjust the node links.

3. Intelligence Sharing and Collaboration Based on Adaptive Group Communication

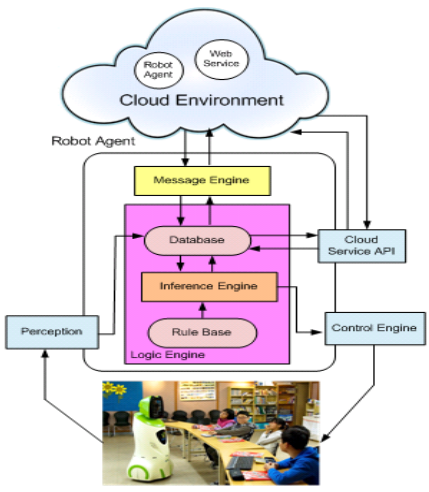
In the proposed information sharing model, the components for integrating cloud technologies to robots are considered. A proxy robot is a program in a computer node representing the real robot which is the main component in the proposed model. Primitive controls and commands of a robot are processed in the control engine and these are abstracted in the logic engine. The logic engine consisted of database, inference engine and rule base, is the “brain” of a proxy robot. Also, the logic engine includes the configuration and training modules to acquire the action rules of a robot. In [5], three types of robots were identified which are also considered in the proposed model. In our design, the inconspicuous and visible robots use a message

based controls for easy integration of primitive commands while virtual robots only use the message engine in performing collaboration. The proposed design can support wireless transmission of commands by employing simple messages composed of small bytes. This method is efficient in energy consumption by assigning the highly complex processes to the computer system while basic commands are transmitted wirelessly. The proposed model uses a message-based mechanism to manage the communication between cloud services and hardware controls of a robot, and also, a group method is integrated to form virtual groups and promotes collaboration. The group communication of robots is based on a multi-agent technology to provide efficient collaboration and information sharing.

3.1 Cloud-Enabled Robot based on Agent System

The proposed information sharing model has three main components which are the message engine, proxy robot and control engine. The message engine pre-processes the received message to be understood by the logic system of a proxy robot and formats the message to be sent to the cloud service or other cloud-enabled robots. The logic engine performs analysis to its environment based on perceptions, and by using the rule base, it processes the information and executes the specified response or action. A similar approach in [6], where robots are provided with a translator, however, this approach does not consider the limited hardware. In our approach, a message-based command scheme is used for limitation of complex processes. The proxy robot processes the data received from sensors in the logic engine which decides the commands based on the action rules to perform robot controls. Also, the logic engine processes the messages received from other robots. The control engine receives and executes the commands imposed by the proxy robot.

Figure 1 shows the architecture of a robot agent which is used to communicate with other cloud enabled robots. In Figure 1, the interactions of information sharing model components to gather data in the real environment and to process perceptions in the logic engine are shown. The architecture is based on a rule base system where a list of conditions and its associated actions are stored in a database. At runtime, these conditions are retrieved which are used to map perceptions from sensors and then maps a certain action rule. Within the logic engine, the interactions of its sub-components to process a perception or a message from other robots are shown. A database is contained with list of conditions and interpretation of its sensor value. For example, a given temperature sensor value of 28 degrees Celsius is mapped in the database with its interpreted categorical value like HIGH or LOW temperature. These values are configured manually by the owner/developer of robots as well as the action rules. The perception module always checks the database for matched conditions. If there is a match then it checks the action rules in the rule base which is processed in the inference engine. After it founds a matching rule then it will call the necessary commands to the control engine. However, if there are no rules found then it will query other robot agents to ask about the current perception. In the collaboration, the perception query received from the cloud environment is processed in the message engine and this is processed in the logic rule by mapping the conditions. Moreover, the cloud service API enables the use of cloud services. If a perception requires the use of a cloud service then it executes the cloud service through its API. The result from a cloud service will be processed in the logic engine by matching the conditions based on the outputs to check if there is a necessary action to perform.



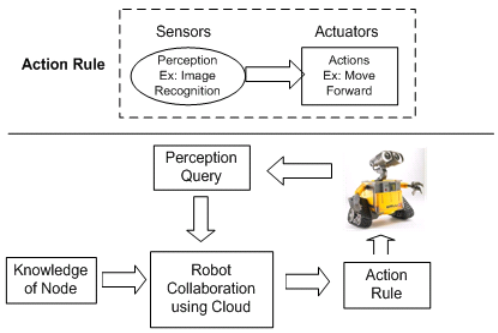
(Figure 1) Architecture of a robot agent based on a rule base system

3.2 Model for Intelligence Sharing and Collaboration

The message-based mechanism uses two formats for the two different functions; Collaboration Message and Control Message. The Collaboration Message is used by a robot agent to collaborate with other peer robot agents after it joined a virtual group of robot agents. The message format for collaboration uses XML format. The structure of the message is consisted of source and destination addresses, and the message data which is represented by *M*={*source address*, *destination address*, *message*}. The message follows the ACL format from multi-agent standard communication. The query message for perception contains *M*={*source address*, *destination address*, *message*(*query*, *perception*)} where *message*(*query*, *perception*) encapsulate the perception to be queried. The reply message for a perception query contains *M*={*source address*, *destination address*, *message*(*reply*, *action rule*)} where *message*(*reply*, *action rule*) encapsulate the action rule for the reply. On the other hand, the Control Message is used to manage the sensors and mechanisms of a robot. Each control is processed by a message where these messages are interpreted by the control engine to be understood when processed in the logic engine. The translation of native message for controlling the robot mechanisms is processed in the control engine and vice versa.

Robot agent ontology defines the robot functions and is used to interoperate with the semantic web. The robot agent ontology is used in the information extraction when the robot is performing autonomous information search and other web services. The outputs from the message engine and control engine are processed in the logic engine. Incoming and outgoing messages in the message engine are based on the ACL Message of multi-agent technology, whereas, the message command represents primitive message commands from the specific robot to perform a control. These primitive commands are specific on the installed actuator and sensor modules and these are entered as variables for the rule mapping are defined in the database.

The collaboration among robots is processed in the message engine. In this paper, we define an intelligence sharing as the exchange of intelligence in the form of action rules. An action rule contains a sensor perception with its value and an associated action for the perception. On top of Figure 2, the composition of an action rule is shown. If a robot cannot understand the perception then it queries to its virtual group to ask about the action for that perception. A perception query is a message used by a robot agent to find an action rule in a virtual group about a perceived input. A robot agent sends a perception query message and this is received by other robot agents members of the virtual group. At the bottom of Figure 2, the query process for collaboration is shown. A perception query is sent to the Cloud environment and each robot agent in the same virtual group will receive the perception query. After a robot agent received the perception query message, it will process to its message engine and logic engine, and if the robot agent has similar input/perception parameters then it maps its action rule and replies back, else, it replies with empty rule.



(Figure 2) An action rule for robots (top) and process of query about a perception (bottom)

A robot agent, representing a proxy robot, uses the multi-agent approach in performing the communication. The virtual groups of robot agents are formed based on a published-subscribed method. When a robot agent establishes its connection for the first time, it sends a multicast message to the coordinators of virtual groups. The grouping service will assist the robot agent to choose the appropriate coordinator. After the coordinators received the message, all will reply with information of its group. If the robot agent decides to join a group then it requests the addresses of group members. To efficiently share the information to another robot, the adaptive group communication is proposed

which is explained on the next section.

3.3 Adaptive Group Communication based on Brownian Agent

We refer a robot agent as a peer agent (*pa*) after it joined a group. The group is based on a publish-subscribe paradigm where a publisher posts messages to initialize the formation of a virtual group. We assume that the robots in each group are used for specific application domain, e. g., healthcare, rescue operations, etc. The peer grouping is managed by coordinators (*C*) which are the publishers. Each coordinator establishes its communication to other coordinators in able to forward query messages if a request perception query is not found within the group. After forming the group, the adaptive group communication scheme is operational. A robot agent is provided with the list of members and these links are used for query. Each node link is evaluated using the adaptive scheme. The knowledge of robots in the group can vary over time. In our design, the database of perceptions and action rules of a robot can be updated frequently by offline training or autonomous information extraction in the web, and as a result, it can answer more queries. Therefore, it is a necessity to adjust the communication links from robot agent to direct more queries to the robots which are frequently updating its database. Using this adjustment scheme will prevent unnecessary message overheads.

The communications among peer agents use the weighted node links (*w*) which is based on the Brownian agent approach [17] to identify the nodes with significant knowledge and have high probability to answer queries. Brownian agent approach combines the features of reactive and reflexive agent concepts. A reflexive agent has an internal model about its environment that allows it to understand the given situation and then perform certain actions. On the other hand, the reactive agent simply "reacts" to signals from the environment. A set of state variables for the Brownian agent is described by $u^{(k)}{}_i$ where index i = 1, ..., N refers to the agent i, while k indicates the different variables used to measure the significant node links. In this paper, the number of action rules of a robot represents the external variable of Brownian agent. The external variable, represented by $u^{(1)}{}_i$, is the ratio value of rule count of the peer agent i (a_i) to the rule count of a peer with the highest rule count shown in Equation 1.

$$\theta_1 = \frac{a_i}{\max A} \tag{1}$$

In Equation 1, a_i is the number of rules of peer agent i divided by max$A$ which is the peer agent with the highest count of rules. The internal variable represented by $u^{(2)}{}_i$. The internal variable is calculated by the number of replies with correct action rule for a perception query of the robot agent which is shown in Equation 2.

$$\theta_2 = \frac{r_i}{R_n} \tag{2}$$

R is the number of requests and r_i is the score of a node link every time a correct rule was replied. An important continuous state variable in the context of Brownian agents is the internal energy depot $u^{(3)}{}_i = e_i$, which determines whether agent i may perform a certain action or not. This approach is used by the group communication to adapt its link by determining the significant nodes which have high probability to answer the queries. N is the number of peer agents in the group and n is the index of peer agent. w_i is the link value of a peer agent n to other peer agent i in its list of link in W={ w_1, w_2 ⋯, w_i }. In our method, w represents the energy depot of a Brownian agent (w_i=e_i) which decides if the query will be sent to the peer agent i. Each member of a group sends the number of action rules to calculate the external variable using Equation 1 and the peer agent n updates its node link values after it receives a reply to its query. Peer agent n adjusts its node links depends on how much a member contributes to queries calculated Equation 2. The links are processed in a weight function in Equation 3 which uses two variables stated in the Brownian agent approach; external (Θ_1) and internal (Θ_2) variables. A peer agent n calculates its link to peer agent i by,

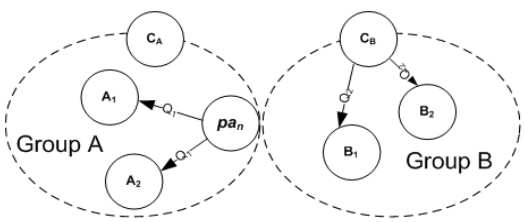
$$w_i = e_i = \frac{\theta_1 + \theta_2}{2} \tag{3}$$

In Equation 3, the external and internal variables are averaged represent the significant values from the number of rules and contribution to answer queries. Using a threshold value, a peer agent decides which node it will send the query. In Equation 4, the node links are collected after it was compared to a threshold value which is shown as the function of ϕ in Equation 5.

$$W' = \sum_{i=1}^{W} \varphi(w_i) \tag{4}$$

$$\varphi = \begin{cases} pa_{i,} \text{ if } w_i > \Phi \\ 0, \quad \text{if } w_i < \Phi \end{cases} \tag{5}$$

Equation 5 is used to decide whether the message query will be sent to the specified peer agent pa_i. All peer agents that were collected in Equation 5 are sent with the perception query in multicast. The result of choosing only the significant links minimizes the message overheads generated in the group communication.



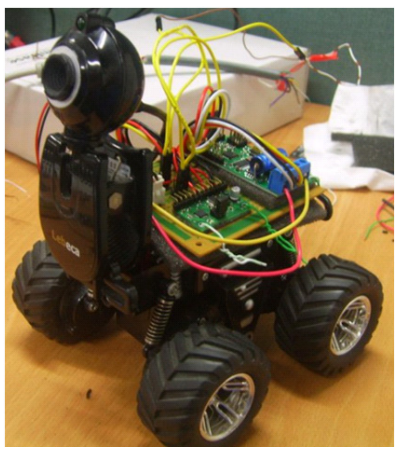
(Figure 3) A peer agent, which is a member of group A, queries Q_1 to its members (left) and queries Q_2 the coordinator of group B if a rule is not found in its group (right)

Figure 3 illustrates the query procedure of a peer agent with the group members. Whenever a peer agent was not successful in querying its group, then it queries the peer agents on other groups. In this case, the peer agent sends the query to the coordinator of other groups and then the coordinator will forward the message to its members. The query message identified in this case is Q_2. In the left side of Figure 3, a peer agent pan performs a query (Q_1) in its group where the member of Group *A* receives the message, and if the rule is not found in the group then it queries the coordinator of Group *B* to query its member for the second query (Q_2) shown in right of Figure 3. After receiving the Q_2, the peer agent in other groups will append the result to *m* and *m* is returned directly to the source peer agent which is pa_n.

4. Implementation and Performance Evaluation

4.1 Implementation of Cloud-Enabled Robot

The proposed information sharing model was developed in Java included with the library of Jade Framework to implement the collaboration based on agent approach. A Pololu motor controller was used to control the motor speed and direction of the robot. In the message-based control approach, specific commands, in byte format, are used to control the motor which is transferred from the computer to the motor controller through by serial communication. E. g., the forward command is written as {0xAA, device number, 0x55, speed 1, speed 2} where 0x55 in the third element of the message is forward command. The primitive commands were mapped in C# and then a socket network program was used to communicate with the proxy robot and the motor controllers. The curious robot, which is the name of our prototype robot, is a visible type of robots. ZigbeX wireless sensors from Hanback Company were also used which are inconspicuous types in our implementation.



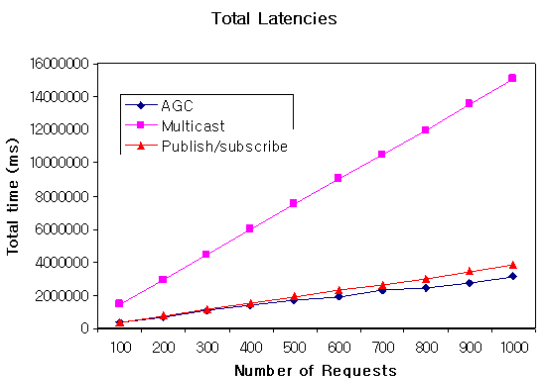
(Figure 4) Prototype of the curious robot (visible type) using the proposed framework

In Figure 4, the curious robot performs perception using a web camera. It was designed to perceive objects and images to be recognized and perform actions based on its perceptions. A simple function was tested for our simulation. The curious robot was trained for character recognition where libraries for image processing were integrated in the proposed middleware. If the curious robot reads "FORWARD" written in a paper then it tries to understands this and perform a forward motion which is mapped through its list of action rules. The library of Jade Framework was integrated in the information sharing model to implement the agent communication. The communication and establishment of proxy robots are initialized by booting the Jade.

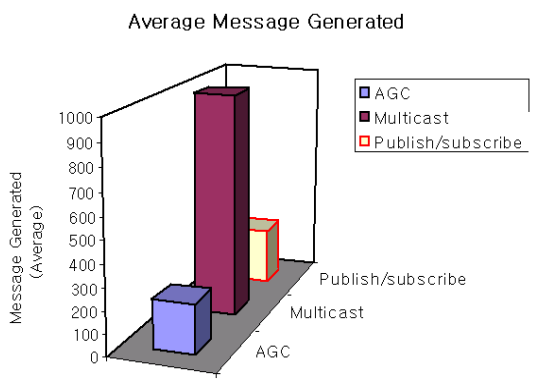
4.2 Performance Evaluation

The adaptive group communication was used by the cloud-enabled robots and this was evaluated in simVO [18] which is a discrete event simulator for virtual networks. The modules in simVO were extended with the necessary components to simulate group communication of robots in the network environment. The simulation environment used 1000 nodes, represents robot agents, and 1000 action rules which were used to process the perception query. In the physical network, nodes were divided equally into 100 domains where each domain serves 10 nodes. In a domain, nodes were connected to a router with a latency of 10 milliseconds (ms) to send a message. Routers were connected in a ring topology having two router neighbors in each router with a latency of 100 ms in each connection. The action rules were replicated and distributed randomly throughout the nodes and, in default, a node has 10 action rules which were assigned. We assigned four groups with coordinators in the simulation. A perception query was sent to the members of the group and return to the sender after searching for action rules.

The multicast group communication [19] and publish-subscribe method [20, 21] were compared to adaptive group communication (AGC). In the multicast, all robot agents were connected in a single group. A single query from a robot agent was distributed using multicast. In publish-subscribe, the members were divided into four groups where a coordinator published a topic and subscribers were assigned to the groups initiated by coordinators. For AGC, the threshold value was set with 0.5 for the node link weight value. A robot agent was only assigned in a single group in both publish-subscribe and AGC. The number of perception queries was increased in each case of simulation from 100 to 1000 queries. Each query contained a time to perform the query, source node of the query and the perception which were randomly selected. A learning model was also configured which simulated the variation of learning methods from robots. In the simulation, nodes were selected to update its action rule database. The following were determined; 1) total latencies of messages, 2) average message generated by a query and 3) number of rules found in a query.



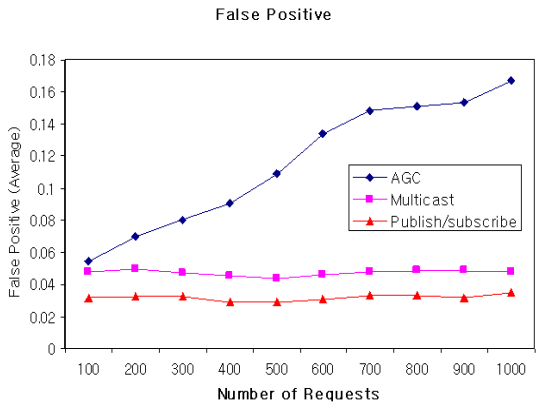
(Figure 5) Message overheads by total latencies generated from queries



(Figure 6) Average message generated in each query

The result from Figure 5 adds all latencies generated by the queries. In Figure 5, the AGC has the lowest latency because it has fewer links to send the query message. AGC adjusted its node links to identify the significant links. Both AGC and publish-subscribe have almost similar trends of message latency because of the separations between groups. It was also observed that the message overhead of AGC was improved in longer time from 300 to 1000 queries because the node links were adjusted to identify more significant links. The latency performance in multicast was proportional to the number of queries where a constant increase of latencies was observed. Multicast has a very high latency because of messages were multiplied in sending a query to the members of the group.

In Figure 6, the average message generated in each perception query was observed. AGC has the lowest volume of message generated and better of almost 5 times than the multicast. Note that the multicast was constantly generating 999 messages in each query within the network while the published-subscribe generated approximately 252 messages or 0.25 of total nodes. The AGC with 221 messages was lower than publish-subscribe because it only sent the query to the significant links after it adjusted from the previous queries. Multicast generated the most messages which indicate that it tried to find the rule for perception query throughout the network of peer agents. Figure 6 is also related to the latency performance in Figure 5 where multicast shown a high message overhead.



(Figure 7) Average false positive based on number of requests

The accuracy of query was measured by a false positive approach where the result is shown in Figure 7. This is calculated by the number of replies with the action rule over the number of perception queries. A higher false positive value means the search has better retrieval of rules. It was obvious that multicast provided high message overhead shown in Figures 5 and 6 but it was for the cost of finding correct rules for the robot to response with the perception query shown in Figure 7. However, this was significantly improved by the AGC because of the adaptive scheme. In the first 100 requests, the differences of values from algorithms were low, especially with multicast and AGC. However, in the longer period of time, AGC was increasing its accuracy because of the adjustment from node links using the adaptive scheme. It is also observed in Figure 7 that the accuracy of the ACG is increased from 500 until 700 queries compared to the previous cases but the accuracy is slightly increased from 800 to 900.

In summary, the graph results from Figures 5 and 6show that the AGC is more efficient in handling message overheads, and in Figure 7, AGC considers the significant links to find a rule compared to multicast and published-subscribe. Separating the publish-subscribe in groups simply overcomes the message overhead shown in Figures 5 and 7 but it was not efficient in removing the unnecessary links. The multicast tries to find the rules throughout the group members with the cost of message overhead by comparing Figures 5 and 7. However, in the longer period time, AGC increased its accuracy because of the node links were adjusted using the adaptive scheme.

5. Conclusion

The current problems in robotics are the limitation of resources to process complex computations and limited data storage. Integrating a robot in the Cloud with a proper design can be a solution to these problems. In this paper, the efficient interaction among robots and integration of cloud services to robots were considered to improve the information sharing in the Cloud environment. The proposed information sharing model for the cloud-enabled robots was designed to manage the hardware controls of a robot, to support the use of cloud services, and to collaborate with

other robots by sharing its intelligence. The design of logic system to perceive and perform actions was done using Java while the primitive functions for the hardware controls of a robot were abstracted by RMI and socket network programs. To provide the efficient message dissemination, an adaptive group communication based on Brownian agent approach was proposed. Using the proposed adaptive scheme, a node link weight value determined the significant links of a robot agent to query.

A network simulation environment was configured to test the performance of the proposed adaptive scheme compared to traditional group communication methods. In publish-subscribe, separating the robots in groups simply overcomes the message overhead but was not efficient in removing the unnecessary links while the multicast tries to find the rules throughout the group members with the cost of message overhead. However, AGC outperformed the two methods by having a minimal message overhead and efficient in finding rules for perception queries because of the significant links which reduced the message generated of a query and provided the high probability of finding rules.

The information sharing model is the main backbone of cloud robots to share data and do autonomous learning. This will enable the creation of various services for the cloud robot and these are what we want to implement in our future work.

References

[1] E. Guizzo, “Cloud robotics: connected to the cloud, robots get smarter,” at http://spectrum.ieee.org/ automaton/ robotics/robotics-software/cloud-robotics.

[2] Google’s android apps for cloud robotics, at http://rj3sp.blogspot.com/2011/05/googles-android-apps-for-cloud-robotics.html.

[3] Technical Committee on Service Robots, at http://www.service-robots.org/technologies.htm.

[4] T. Mukai, S. Hirano, H. Nakashima, Y. Sakaida, and S. Guo, “Realization and safety measures of patient transfer by nursing-care assistant robot RIBA with tactile sensors,” Journal of Robotics and Mechatronics, Vol.23,No.11, 2011, pp. 360-369.

[5] S. Khanna, J. S. Naor and D. Raz, “Control message aggregation in group communication protocols,” in Proc. Automata, Languages and Programming, LNCS Vol.2380, 2002, pp.135-146.

[6] H. Tezuka, N. Katafuchi, Y. Nakamura, T. Machino, Y. Nanjo, S. Iwaki, and K. I. Shimokura, “Robot platform architecture for information sharing,” Journal of Robotics and Mechatronics, Vol.18 No.3, 2006, pp. 325-332.

[7] A. Sanfeliu, N. Hagita and A. Saffiotti, “Network robot system,” Robotics and Autonomous System, Vol.56, 2008, pp.793-797.

[8] J. Baxter, E. Burke, J. Garibaldi, M. Norman, “Multi-robot search and rescue: a potential field based approach,” studies in computational intelligence, Vol.76, 2007, pp. 9-16.

[9] K. Nagatani, Y. Okada, N. Tokunaga, K. Yoshida, S. Kiribayashi, K. Ohno, E. Takeuchi, S. Tadokoro, H. Akiyama, I. Noda, T. Yoshida, E. Koyanagi, ″Multi-robot exploration for search and rescue missions: A report of map building in RoboCupRescue 2009,″ in Proc. IEEE International Workshopon Safety Security Rescue Robotics, 2009, pp. 1-6.

[10] H. Sugiyama, T. Tsujioka and M. Murata, ″Collaborative movement of rescue robots for reliable and effective networking in disaster area,″ In Proc. International Conference on Collaborative Computing: Networking, Applications and Worksharing, 2005.

[11] U. Witkowski, M. El-Habbal, S. Herbrechtsmeier, A. Tanoto, J. Penders, L. Alboul and V. Gazi, ″Ad-hoc network communication infrastructure for multirobot systems in disaster scenarios,″ In Proc. EURON/ IARP International Workshop on Robotics for Risky Interventions and Surveillance of the Environment, 2008.

[12] K. Schiling, “Networked robots: research challenges” in ETSI: Networked Mobile Wireless Robotics Workshop.

[13] Robot Operating System (ROS), at http://www.ros.org/wiki/ROS.

[14] R. Arumugam, V. R. Enti, L. Bingbing, W. Xiaojun, K. Baskaran, F. F. Kong, A. S. Kumar, K. D. Meng, and G. W. Kit, “DAvinCi: a cloud computing

framework for service robots," in Proc. IEEE International Conference on Robotics and Automation, 2010, pp. 3084-3089.

[15] S. M. Das, Y. C. Hu, C. S. George Lee and Y. H. Lu, "An efficient group communication protocol for mobile robots," in Proc ICRA, 2005, pp. 87-92.

[16] S. Piao, Q. Zhong, Y. Liu and Q. Li, "Research of group communication method on multi-robot system," Communications in Computer and Information Science, Vol. 159, No. 9, 2011, pp. 457-461.

[17] F. Schweitzer, "Active brownian particles: artificial agents in physics," Stochastic Dynamics, Berlin: Springer, Lecture Notes in Physics, Vol. 484, 1997, pp. 358 - 371.

[18] R. M. A. Mateo, H. H. Yang and J. Lee, "Managing virtual organizational tasks using simvo in grid environment," in Proc. ICONI & APIC-IST, 2010, pp. 669-673.

[19] A. Tiderko, T, Bachran, F. Hoeller and D. Schulz, "RoSe - A framework for multicast communication via unreliable networks in multi-robot systems," Robotics and Autonomous Systems, Vol. 5, No. 12, 2008, pp. 1017-1026

[20] M. Matteucci, "Publish/Subscribe Middleware for Robotics: Requirements and State of the Art," Technical Report 2003.

[21] M. A. Mastouri and S. Hasnaoui, "Performance of a publish/subscribe middleware for the real-time distributed control systems," IJCSNS, Vol. 7, No. 1, 2007.

◐ 저 자 소 개 ◑

Romeo Mark Mateo

2004년 West Visayas State University, Philippines BS in Information Technology
2007년 Kunsan National University, South Korea, Master of Engineering major in Information and Telecommunications
2012년 Kunsan National University, South Korea, Ph.D. of Engineering major in Information and Telecommunications
2012년～현재 NEXUSCOMMUNITY, South Korea, Software Engineer
관심분야 : Cloud computing, distributed systems, data mining, fuzzy systems, multi-agents, ubiquitous sensor networks, cloud computing
E-mail : mark@nexus.co.kr

이 재 완 (Jaewan Lee)

1984년 중앙대학교 이학사-전자계산학
1987년 중앙대학교 이학석사-전자계산학
1992년 중앙대학교 공학박사-전자계산학
1996년 3월～1998년 1월 한국학술진흥재단 전문위원
1992년～현재 군산대학교 교수
관심분야 : 분산 시스템, 운영체제, 유비쿼터스 시스템, 클라우드 컴퓨팅 등
E-mail: jwlee@kunsan.ac.kr