Proceedings of the 2012 IEEE
International Conference on Robotics and Biomimetics
December 11-14, 2012, Guangzhou, China

Towards Cloud Robotic System: A Case Study of Online Co-localization for Fair Resource Competence

Lujia Wang*, Ming Liu†, Max Q.-H, Meng*
* Department of Electronic Engineering, The Chinese University of Hong Kong
† Autonomous Systems Lab, ETH Zurich, Switzerland
{ljwang,max@ee.edu.cuhk.hk}, ming.liu@mavt.ethz.ch

***Abstract*—The cloud transforms the potential of robotics, which enable poor-equipped robots to fulfill complex tasks. Robots are relieved from hardware limitation, while large amount of available resources and parallel computing capability are available in the "cloud". We implemented a data management system using Twisted-based server-client platform and Robotic Operating System (ROS), aiming at co-localization of cloud robots. However, resource competition is pervasive for practical applications of networked robotics. As a major bridge, the limited bandwidth becomes a bottleneck needs to be considered for the architecture design. We propose an infrastructure which considers multi-robot autonomous negotiation (MRAN) module. The framework is validated by enabling several poor-equipped robots to retrieve location data from a dynamically updated map which is built by a well-equipped robot. Experiment results demonstrate that the proposed framework is feasible for current robotic applications. Furthermore, it achieves better performance under resource competition, and optimizes Quality of Service (QoS) using a shared network with limited bandwidth.**

I. Introduction

Service robots have become an integral part of our life. In addition, requirements of services are more complicated than ever before. As for classic robotic system, robots had to carry enough physical processing power and various sensors. However, it is impossible to develop a universal robot that covers all possible services due to the cost and limitations of power consumption, payload, sensory and kinematic constraints etc. As the basis of all major services, such as delivery, mapping, attendance etc, localization is the core function. A typical cloud robotic system is shown as Fig 1. Shown by this sketch, all primary information can be retrieved from the "cloud" (usually, a data center is adopted). With such a framework, the requirement on equipments is relieved, even under the condition of inevitable competition of data retrieval.

In this paper, we introduce a generic framework to fit the requirements of current robotic system with consideration of resource competence. As a case study of the proposed framework, a co-localization experiment scenario is demonstrated. A well-equipped leading robot generates and shares the available information over the network, whereas a host server can aid the localization of several poor-equipped follower robots to realize precise metric localization. Different from existing frameworks, the negotiation for resource allocation among client robots is managed. Moreover, it considers the computational and facility constraints of all the client robots.

As co-first authors, Ming Liu and Lujia Wang have equal contributions to this paper.

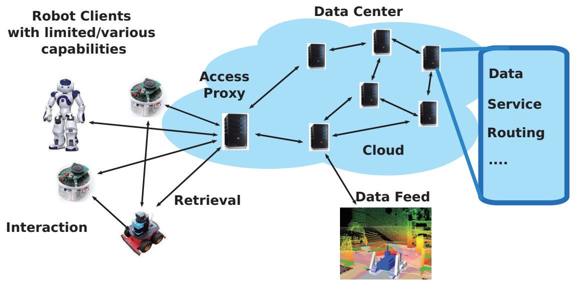


Fig. 1. A typical cloud robotics architecture

A. Service-oriented Architecture in Cloud Computing

Service-oriented architecture (SOA) [1] is a widely used framework for cloud computing, where the host and clients are synthesized under an elastic architecture. It represents computing in three parallel processes: service development, service publication and application composition using services that have been published. This characteristic leads to efficient task realizing of online multi-robot system. Most existing works with this framework are web-service based or database dependent. All major information technology companies and providers, including Google [2], IBM [3], Intel [4], Oracle [5], SAP [6] have adopted and supported this computing paradigm. In addition, many of them use web-based platform to configure the infrastructure that processing power, memory capacity, and communication bandwidth. For example, Microsoft Robotics Developer Studio (MRDS) [7] was a vital product in applying SOA to embedded systems [8], [9].

B. Current Cloud Robotic System

Researchers at ASORO laboratory in Singapore built a cloud computing infrastructure "DAvinCi" [10] to generate 3D models of environments which allow robots to perform simultaneous localization and mapping (SLAM). A SOA-based Robot is implemented as a Service (RaaS) [11] platform, which enforces the design and implementation of a robot or a device to be all-in-one SOA unit, that is, the unit includes services for performing functionality, service broker for discovery and publishing, and applications for client's direct access. In [12], a robot cloud center is designed following the general cloud computing paradigm to address the current limitations in capacity and versatility of robotic applications. Waibel et al built a web community called RoboEarth [13] for robots to autonomously share descriptions of environments and object models, as well as the tasks that have been assigned.

The RoboEarth cloud Engine [14] is a "platform as a service (PaaS)" [15] framework, which allows users to run their robotic software on the cloud with minimal configuration. Although the above researches are close to the overall optimized goal of cloud robotics, there are still drawbacks and challenges to be further addressed. Respect to many benefits mentioned previously, providing seamless and costless of service robot is one of the most meaningful field to implement. In order to simplify the problem, most of the robotic systems assumed that the resource in cloud is unlimited. Actually, we must admit that most resources in cloud robotics system are limited. For instance, network bandwidth for transmitting image data, CPU occupancy for parallel computation, as well as available number of hosts (proxy) in multi-robot system are limited. Therefore, how to design a module that maximize the utility of available resources on demand is a challenging problem. Especially, when multiple robot users request the same kind of resource or service in an asynchronized manner.

C. Resource Management Framework

As discussed in [16], because of heterogeneous sensor data retrieval, the cloud is usually addressed by a common middle-ware to get interoperability throughout the entire infrastructure. Many researches on resource management are constrained in the field of e-commerce and enterprise computing systems, such as resource management architectures of Eucalypus of Amazon EC2 [17], OpenNebula [18] and Nimbus [19]. Besides, it is also emerging in cloud computing. Therefore, autonomous negotiation among multiple robots becomes a crucial problem in cloud robotics system when they require the resources in parallel.

In a highly dynamic environment, such as grid computing and cloud robotic system, it is essential to take dynamic interactions into consideration. Many researchers are working on how to derive market-based negotiation strategy among agents and resource allocation in a dynamic system [20]. As one of the market-based method, game theory [21] has a long history being applied in the decision making and resource allocation. For robotic system, [22] introduced a single task centric iterated auctions in robotic system, Vickery auction is applied for multi-robot system [23]. In [24], combinatorial auction is utilized to allocate multi-task in a multi-robot system, they got good results with limited number of task bundles. The maximization model of scheduling cost is proposed in [12] according to the solution of Knapsack problem. The above works solved the problem by emphasizing feasible implementations of task. However, few real-time robotic scenario are conducted in the cooperation system.

D. Contributions

In this paper, we address the following characteristics, which are deployed in physical devices and embedded systems for cloud robotics.

- A generic framework of cloud robotic system is introduced in order to extend the benefits of cloud into robotic applications.
- A resource negotiation module is proposed, which can relieves the competition among robots as well as functions as a bridge between cloud computing and robotics.
- A real-time experiment is implemented in a typical indoor environment. It validates the above technical contributions by several physical robot clients that performed services based on the location information retrieval.

Without loss of generality, negotiation agents are supposed to have incomplete information about other agents. For example, all robot agents know the price menu of the resource and the number of trading competitors; however, they don't know the willingness payment of each other. The experiment results indicate the feasibility of online resource allocation for cooperate localization of multi-robot system. The application of such system can be in a wide range. For example, teams of robotic systems acting in hostile, unsafe environments [25], [26]; synchronized data retrieval for multi-sensor fusion [27]; computational agents that facilitate distributed design and tasks; intelligent agents that collaborate to provide updated information from service providers.

E. Arrangement

The rest of the paper is organized as follows. Section II introduces the whole framework of multi-agent cloud robotics system in details. The implementation of experiments scenarios, the evaluation results of the framework as well as negotiation protocol is demonstrated in section III. Section IV discuss the results of resource allocation and access control and the generalization of the proposed frameworks. Finally, section V concludes the paper followed by co-localization experiment video of qualitative results.

II. System Design

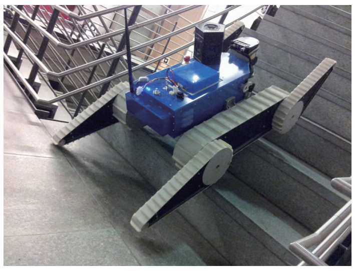
Regarding cloud robotic system, as a specific instance, a typical multi-robot system is illustrated in the Fig. 1. As major features, it distributes the workload of sensing, localization, computation and communication among a group of robot agents. As for the designation of the robotic system conducted in this research, we consider the following constraints.

A. Hardware Setup

The hardware system is comprised of two major robot categories. We consider leading robots acts as information providers, while follower robots as consumers requiring data.

1) Well-equipped Leading Robot: the leading robot is shown as 2, it equips with several sensors like rotating laser scanner (for 3D point-cloud), omni-camera (Ladybug^{TM}) and inertial measure unit (IMU) with a GPS module, providing an online database with mapping and localization information.

2) Poor-equipped Follower Robot: the follower robot "epuck" is shown as Fig. 2, it equips a Firefly TMcamera and a Wifi module. It uses camera to capture 2D bar-codes on the wall in the target environment, then send it to the host to request the location or regional map around it.


(a) Leading robot: nifti

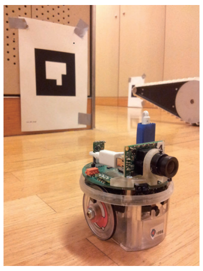

(b) Follower robot: epuck

Fig. 2. Robots used in the experiment

B. Co-localization Strategy

1) 2D Bar-codes: Augmented Reality (AR) marker, as shown in Fig. 2, is utilized as 2D bar-codes for localization. It provides a package named ARToolKit, which uses computer vision algorithms to calculate the real camera position and orientation relative to physical markers in real time.

2) Pose Estimation: at the beginning, the leading robot builds a 3D map with both images and registered local maps. Then the position of each AR marker is registered on map by subscribe the related ROS topic, such as transformations. The relative poses are calculated by ARToolKit module[1]. At last, the pose of marker i in the map can be obtained by equation (1), where T_b^a is the transformation matrix from frame b to a, namely target frame b in base frame a. Similarly, the pose retrieval for each follower robot j regarding marker i is formulated as equation (2).

We could see that the data retrieval efficiency will determine the efficiency of the whole co-localization system since it provides a required link. This problem is non-trivial when the system scale is large. Due to the limited bandwidth constraints and constraints of computational ability, the response of such information needs to be negotiated among clients, and to be managed by the central host.

C. Software and Communication

1) ROS System and Topic Bridging: in ROS system, services provide task request and responses among nodes. Moreover, the `actionlib` package provides tools to create servers that execute long-running goals that can be preempted. However, it does not support the queue management, especially asynchronized access for multiple tasks on the waiting list. For real-time robotic tasks, it is not sufficient in general for multi-robot systems. As a complement, we utilize twisted [2] socket communication introduced in the following subsections.

[1]www.rog.org/wiki/artoolkit
[2]www.twistedmatrix.com

2) Twisted Asynchronized Communication: Twisted [28] is a framework for deploying asynchronous, event-driven and multi-thread supported network system coded in Python. It is compromised of the following three primary elements.

a) Reactor: the reactor is the core of the event loop within Twisted – the loop which drives applications using Twisted. The event loop is a programming construct that waits for and dispatches events or messages in a program. It works by calling some internal or external "event providers", which generally blocks until an event has arrived. It then calls the relevant event handler respectively. The reactor provides basic interfaces to a number of services, including network communications, threading, and event dispatching.

In Fig. 3, we use looped circles to represent the running reactors. Please note that the reactor pattern separates application code from the reactor implementation, which has been mostly predefined by Twisted. It means that application functions can be simply divided into modular, compact parts. We could also easily write specific data query and data retrieval modules for the clients and hosts respectively.

b) Protocol: the protocol defines the specifications for transmitting and receiving behaviors. Functions regarding received data and sent data can be constructed following the predefined virtual function names. A protocol begins and ends its life with two predefined virtual functions: *connectionMade* and *connectionLost*, which are called whenever a connection is established or dropped.

c) Factory: the factory is responsible for two tasks: creating new protocols, and keeping global configuration and state. In addition to abstractions of low-level system calls, it also includes a large number of utility functions and classes, which facilitates the establishment of new types of servers. Twisted includes support for popular network protocols, e.g. SOCKETS, HTTP and SMTP etc.

D. Resource Allocation

1) Access Control: when a service request is first submitted, server utilizes the proposed admission control mechanism to interpret the submitted request before determining whether to accept or reject it regarding QoS requirements. Thus, it ensures that there is no overloading of information, where overwhelmed robot client requests can not fulfilled successfully due to limited resources.

2) Market-based Negotiation: market-based scheduling is a typical quick and concise strategy for multi-agent system to make decisions on distribution of resources. The host keeps track of response for all clients requiring service, and make a rank list of response sequence When new request arrives, the host filters the rank list which records do not meet the minimum requirements. Then it update the response priority sequence and send matched data back. Benefits from the scheduling management, the proposed system can deal with the uncertainty in the environment in real-time. In addition, the system can handle large-scale problems since the computation is distributed among the robot clients.

$$\text{Marker Registration: } T^{/map}_{/marker_i} = \underbrace{T^{/map}_{/leading\_robot}}_{SLAM} \cdot \underbrace{T^{/leading\_robot}_{/camera\_leading\_robot}}_{SensorCalibration} \cdot \underbrace{T^{/camera\_leading\_robot}_{/marker_i}}_{ARToolKit} \tag{1}$$

$$\text{Robot Pose Retrieval: } T^{/map}_{/robot_j} = \underbrace{T^{/map}_{/marker\_i}}_{DataRetrieval} \cdot \underbrace{(T^{/camera\_robot_j}_{/marker\_i})^{-1}}_{ARToolKit} \cdot \underbrace{T^{/camera\_robot_j}_{/robot_j}}_{SensorCalibration} \tag{2}$$

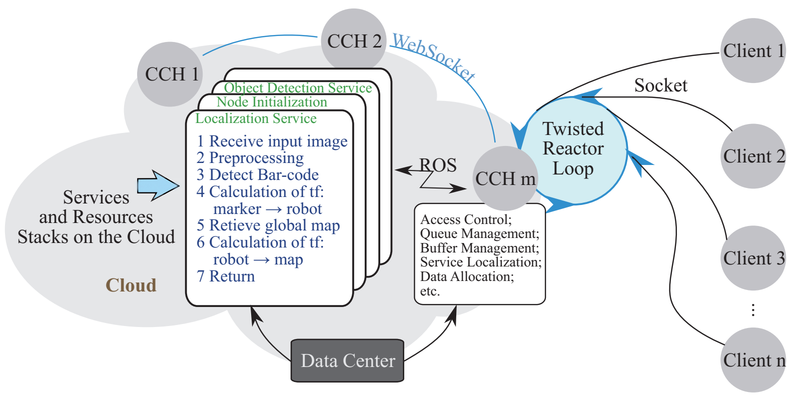


Fig. 3. Data flow and resource management of the proposed system

3) Protocol Management: in the proposed system architecture, we set a threshold for admitted number of robot client according to CPU, bandwidth usage and predicted response time in the twisted factory. Moreover, we define several global visible variables in the Factory, which is however omitted in the figure 3 to save space. One of them is the Local Data Buffer which stores the recent requests and queries. The host factory will manage the connections to all the client reactor loops. At the same time, it is also in charge of updating the existing data center, registering to new multi-sensor readings.

III. Experiment and Evaluation

The goal of this case study is realizing a co-localization between well equipped leading robot and follower robot in the proposed cloud robotic infrastructure.

A. Experiment Process

For the specific task, a structure for data flow and resource management is described in Fig. 3.

1) Environment: a typical indoor environment is shown as Fig. 4. We put up several AR marker in the environment, which pose can be estimated using ARToolKit. The well-equipped robot build a full 3D map with marker location registered on it as shown in Fig. 4(D). All information on the map is stored in a data center and can be subscribed by follower robots according to the response rank. In the aspect of poor-equipped robots, they can use their camera to take picture of AR markers as depicted in Fig. 4(A) and 4(B).

2) Marker Pose Registration and Retrieval: all this location registration is implemented by ROS since ARToolKit is a package belonged to it. Not all AR markers can be accurately recognized and stable registered, we set a threshold to classify them. Only when the confidence is higher than the preset threshold, the location can be recorded in the database.

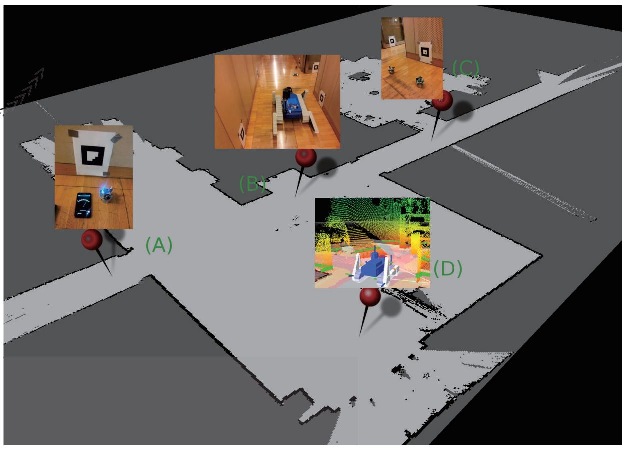


Fig. 4. 3D point cloud and map of a typical indoor environment for multi-robot co-localization

B. Qualitative Results on Localization Behavior

In the online co-localization scenario, all robot clients can be localized in real-time by request from the server. In this work, localization accuracy is mainly affected by matched marker ratio and marker location registered in the data center. Therefore, as shown in Table I we compare the matched marker number with confidence threshold which is pre-set in the `ar_multi.cpp`. For better deriving the results, the Wifi strength is recorded when follower robots are requiring location. Moreover, the localization accuracy of ARToolKit is tested in [29], the localization accuracy is evaluated and

TABLE I
MATCHED MARKER NUMBER VS CONFIDENCE THRESHOLD UNDER DIFFERENT WIFI STRENGTH

Number of markers	Threshold of confidence	Averaged ratio of matched markers	
		$-40dB \sim -50dB$	$-50dB \sim -60dB$
30	0.5	84.45%	71.24%
	0.9	73.28%	60.15%
100	0.5	80.97%	69.22%
	0.9	62.64%	55.34%

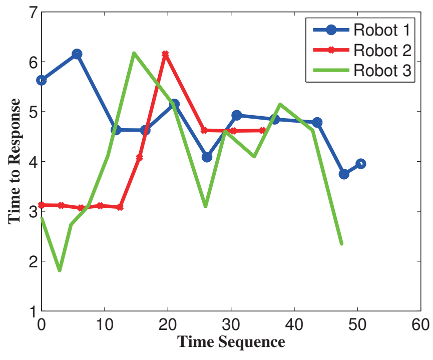
shown as in Table II. The robot positions are well localized. In order to show the real-time property, we also supply a video to show the performance.

TABLE II
MEASURED ACCURACY RESULTS OF X-Y-Z OFFSET ERROR

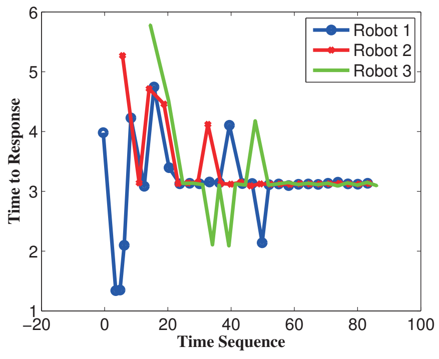
Error Marker (in mm)	Z-offset	X-offset	Y-offset
Standard derivation	32.25884	10.97756	5.197521
Average	22.87656	-2.36570	0.480358

C. Quantitative Results on Resource Allocation

The typical characteristic of cloud robotic is the large number of robot requests in parallel. Reliability-of-response (RoR) [30] is utilized to evaluate since we set 10 epuck to request location in real-time. The RoR is depend on the localization data registered in the database which can be easily calculated according to the predefined formula. The priority setup for robot 1, 2 and 3 are 1, 3 and 2 respectively. At first, we compare the ToR of continuous requests from 3 clients. Fig. 5 demonstrates that the ToR of request from all robots in a period. With higher priority, the corresponding requests get a faster response. Comparing with the case “without control” depicted in (a), the control strategy managed to reduce the ToR according to the priority setup.



(a) Without control, in unit of millisecond



(b) With control, in unit of millisecond

Fig. 5. ToR comparison

IV. DISCUSSION AND FUTURE WORK

In the real-time experiment, the proposed system distributes the workload of sensing, localization, computation and communication among a group of robot agents. The regarding characteristics are discussed in this section.

A. Optimized Constraints of Resource Allocation

It greatly reduced the response time for the localization task by deployment of access control, scheduling and protocol management in the proposed cloud robotic system. The optimization is derived from solving the following constraints.

- Data retrieval constraint: in cloud robotic system, a robot can retrieve information from a dynamic updated data center which is built by various types of robots and sensors, therefore it is a heterogeneous structure that needs standard design and regulation to fit it in applications.
- Communication constraint: synchronization of data is hard for distributed system, especially when asynchronized tasks are performed[31]. If multiple sensor information are distributed in the network on each client, information sharing among robots is low-efficient without resource allocation.
- Autonomous negotiation constraint: different from research on target tracking [32], automated identification of targets [33], and automated behavior reasoning [34]. Practical autonomous negotiation for resource allocation in real-time systems such as cloud robotics is multi-dimensional problem, therefore both revenue of resource provider and utility of user are considered in this paper.

B. Generalization of the Proposed Framework

By using such framework, the major generalization can be derived as follows:

- Extension to cloud robotic system with many poor-equipped robots for information retrieval and communication. It considers of letting robots access a large amount of computational power on demand. The framework sidesteps drawbacks include high computational cost, high configuration, maintenance and update overheads.
- Extension to more complex hierarchical topology of the system including various types of robots. Especially the negotiation strategy can be extended according to complex task and environment that it applies.

V. CONCLUSION

In this paper, we introduced a generic infrastructure of cloud robotic system. A fair resource competence module is implemented, which can relieve the competition among robot clients. Moreover, we propose an online dynamic database which is sharing knowledge between well-equipped leading robot and follower robots. A real-time experiment scenario is implemented by demonstrating a co-localization framework in a typical indoor environment. At last, the matched maker ratio and ToR of QoS criteria is utilized to evaluate the proposed framework and fair resource allocation strategy. The experiment results demonstrate that this system can be extend into practical application. At the same time, the resource allocation strategy optimize the usage of information, which can be extended in cloud robotics.

VIDEO SUPPLEMENT

In the supplied video, we demonstrate a typical qualitative result of one robot moving in the neighborhood of an observable marker. It shows that one of the follower robots can be localized in real time by referring the marker's pose, which was previously registered on the map by the leading robot. It illustrated that the experiment is able to be carried out in real-time with good precision. The lower animation is created by embedding `/tf` information to blender 3D, using an in-house designed interface subscribing to projected robot poses, where green markers show the position of the detected markers, red cone indicates the robot pose.

REFERENCES

[1] M. Bichier and K.-J. Lin, "Service-oriented computing," *Computer*, vol. 39, no. 3, pp. 99 – 101, march 2006.

[2] E. C.-L. M.Chang, J. He, "Service-orientation in the computing infrastructure," *2nd IEEE International Symposium on Service-Orientd System Engineering*, pp. 27–33, Oct. 2006.

[3] J. Clark, "Inside "indigo" – infrasture for web services and connected applications," *Microsoft Press*, 2005.

[4] "Service-oriented enterprise, the technology path to business transformation." *http://www.intel.com/business/bss/technologies/soe/*, 2005.

[5] C. F. Heinemann, "Web programming with the sap web applications server," *SAP Press*, 2003.

[6] M. M. McMurtry C. and N. Watling, "Microsoft windows communication foundation: Hands-on," *Sams Press*, 2006.

[7] M. Corporation, "Microsoft robotics developer studio." [Online]. Available: http://www.microsoft.com/Robotics

[8] W. Tsai, Q. Huang, and X. Sun, "A Collaborative Service-Oriented Simulation Framework with Microsoft Robotic Studio," in *Simulation Symposium, 2008. ANSS 2008. 41st Annual*, april 2008, pp. 263 –270.

[9] W. Tsai, X. Sun, Q. Huang, and H. Karatza, "An ontology-based collaborative service-oriented simulation framework with microsoft robotics studio," *Simulation Modelling Practice and Theory*, vol. 16, no. 9, pp. 1392 – 1414, 2008. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1569190X08001421

[10] R. Arumugam, V. Enti, L. Bingbing, W. Xiaojun, K. Baskaran, F. F. Kong, A. Kumar, K. D. Meng, and G. W. Kit, "Davinci: A cloud computing framework for service robots," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, may 2010, pp. 3084 –3089.

[11] Y. Chen, Z. Du, and M. García-Acosta, "Robot as a Service in Cloud Computing," *2010 Fifth IEEE International Symposium on Service Oriented System Engineering*, pp. 151–158, Jun. 2010. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5570010

[12] Z. Du, W. Yang, Y. Chen, X. Sun, X. Wang, and C. Xu, "Design of a robot cloud center," in *Autonomous Decentralized Systems (ISADS), 2011 10th International Symposium on*, march 2011, pp. 269 –275.

[13] M. Waibel, M. Beetz, J. Civera, R. D'Andrea, J. Elfring, D. Galvez-Lopez, K. Haussermann, R. Janssen, J. Montiel, A. Perzylo, B. Schiessle, M. Tenorth, O. Zweigle, and R. van de Molengraft, "Roboearth," *Robotics Automation Magazine, IEEE*, vol. 18, no. 2, pp. 69–82, 2011.

[14] G. M. Hunziker Dominique, "The roboearth cloud engine." [Online]. Available: http://doc.roboearth.org/rce

[15] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "A view of cloud computing," *Commun. ACM*, vol. 53, no. 4, pp. 50–58, Apr. 2010. [Online]. Available: http://doi.acm.org/10.1145/1721654.1721672

[16] A. Sheth and A. Ranabahu, "Semantic Modeling for Cloud Computing, Part 1," *Internet Computing, IEEE*, vol. 14, no. 3, pp. 81 –83, may-june 2010.

[17] D. Nurmi, R. Wolski, C. Grzegorczyk, G. Obertelli, S. Soman, L. Youseff, and D. Zagorodnov, "Eucalyptus : A technical report on an elastic utility computing architecture linking your programs to useful systems," 2008.

[18] O. penNebula Project, "Opennebula.org - the open source toolkit for cloud computing." [Online]. Available: http://www.opennubula.org/

[19] P. Sempolinski and D. Thain, "A comparison and critique of eucalyptus, opennebula and nimbus," in *Cloud Computing Technology and Science (CloudCom), 2010 IEEE Second International Conference on*, 30 2010-dec. 3 2010, pp. 417 –426.

[20] M. Dias, R. Zlot, N. Kalra, and a. Stentz, "Market-Based Multirobot Coordination: A Survey and Analysis," *Proceedings of the IEEE*, vol. 94, no. 7, pp. 1257–1270, Jul. 2006. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1677943

[21] S. Kraus, "Negotiation and cooperation in multi-agent environments," *Artificial Intelligence*, vol. 94, no. 12, pp. 79 – 97, 1997.

[22] M. G. Lagoudakis, E. Markakis, D. Kempe, and P. Keskinocak, "Auction-Based Multi-Robot Routing."

[23] A. Pongpunwattana, R. Rysdyk, J. Vagners, D. Rathbun, and T. I. Group, "FOR MULTIPLE AUTONOMOUS VEHICLES," pp. 1–10.

[24] M. Berhault, H. Huang, P. Keskinocak, S. Koenig, W. Elmaghraby, P. Griffin, and A. Kleywegt, "Robot exploration with combinatorial auctions," in *Intelligent Robots and Systems, 2003. (IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, vol. 2, oct. 2003, pp. 1957 – 1962 vol.2.

[25] M. Gianni, P. Papadakis, F. Pirri, M. Liu, F. Pomerleau, F. Colas, K. Zimmermann, T. Svoboda, T. Petricek, G. Kruijff *et al.*, "A unified framework for planning and execution-monitoring of mobile robots," in *Workshops at the Twenty-Fifth AAAI Conference on Artificial Intelligence*, 2011.

[26] G.-J. Kruijff, M. Janicek, S. Keshavdas, B. Larochelle, H. Zender, N. Smets, T. Mioch, M. Neerincx, J. van Diggelen, F. Colas, M. Liu, F. Pomerleau, R. Siegwart, V. Hlavac, T. Svoboda, T. Petrcek, M. Reinstein, K. Zimmermann, F. Pirri, M. Gianni, P. Papadakis, A. Sinha, P. Balmer, N. Tomatis, R. Worst, T. Linder, H. Surmann, V. Tretyakov, S. Corrao, S. Pratzler-Wanczura, and M. Sulk, "Experience in system design for human-robot teaming in urban search & rescue," in *Proceedings of 8th International Conference on Field and Service Robotics*, ser. STAR. Spring Verlag, 2012.

[27] M. Liu, L. Wang, and R. Siegwart, "DP-Fusion: A generic framework for online multi sensor recognition," in *IEEE Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*. IEEE, 2012.

[28] G. L. Moshe Zadka, "The twisted network framework," 2010. [Online]. Available: http://twistedmatrix.com/user/glyph/ipc10/paper.html

[29] B. Berard, C. Petrie, and N. Smith, "Quadrotor uav interface and localization design," in *A Major Qualify Project of Missile Defense Agency under Air Force Contract F19628-00-C-0002 (1Apr00-31Mar05)*, 2010.

[30] Q.-H. M. L. Wang, M. Liu and R. Siegwart, "Towards real-time multi sensor information retrieval in cloud robotic system," vol. 1, 2012.

[31] N. Kaempchen and K. Dietmayer, "Data synchronization strategies for multi-sensor fusion," in *Proceedings of the IEEE Conference on Intelligent Transportation Systems*. Citeseer, 2003.

[32] A. Brooks and S. Williams, "Tracking people with networks of heterogeneous sensors," in *Proceedings of the Australasian Conference on Robotics and Automation*. Citeseer, 2003, pp. 1–7.

[33] D. Klimentjew, N. Hendrich, and J. Zhang, "Multi sensor fusion of camera and 3d laser range finder for object recognition," in *Multisensor Fusion and Integration for Intelligent Systems (MFI), 2010 IEEE Conference on*. IEEE, 2010, pp. 236–241.

[34] M. Kam, X. Zhu, and P. Kalata, "Sensor fusion for mobile robot navigation," *Proceedings of the IEEE*, vol. 85, no. 1, pp. 108–119, 1997.