A P2P Sensor Framework for Collaborative Robots Manipulation

Md. Abdur Rahman, Md. Suruz Miah, Wail Gueaieb and Abdulmotaleb El Saddik
School of Information Technology and Engineering
University of Ottawa
Ottawa, Canada
Email: rahman@mcrlab.uottawa.ca, smiah069@site.uottawa.ca, wgueaieb@site.uottawa.ca and abed@mcrlab.uottawa.ca

***Abstract*— We propose an open hardware and software architecture for the cooperative coordination of multiple mobile robots operating in a common environment. It is designed to meet the stringent requirements of loosely coupled multi-robot architectures, such as flexibility, reliability, and fault-tolerance. As such, the proposed architecture enables the robots to deal with various types of uncertainties in their operating environments. The proposed architecture takes advantage of scalable sensory-based P2P framework. It also offers an inter-robot communication protocol, which is specifically designed to satisfy the requirements of physical sensory data publishing and fusion. The architecture is implemented and evaluated on a team of indoor mobile robots. The test results manifest the architecture's distinguished features and capabilities.**

I. Introduction

Mobile robots are being steadily introduced into modern everyday life and are expected to play a key role in the near future. Typically, mobile robots are deployed in situations where it is too dangerous, expensive, tedious, and/or complex for humans to operate. Such applications can be, for instance, the treatment of hazardous materials, space exploration, rescue missions, remote surveillance, and vacuum cleaning, to name a few. Although many of the real-life applications may only need a single robot, a large number of them require the cooperation of a team of robots to accomplish a certain mission. In the case of fire fighting, for instance, the simultaneous deployment of a number of coordinated mobile robots leads to extinguishing the fire more quickly and limiting the damage that would have occurred if only one robot was launched. The use of multiple robots of overlapping capabilities offers a cost-effective and more robust solution. This redundancy in the robots' capabilities makes the overall system more flexible and fault-tolerant.

In spite of the large body of research and the significant advances in the control of single mobile robots, not much has been done for cooperative (weakly coupled) mobile robots. The design of an efficient collaborative multi-robot framework that ensures the autonomy and the individual requirements of the involved robots is a very challenging task. This requires designing an efficient sensor network, reliable inter-robot communication protocol, a smart robot navigation strategy with obstacle avoidance, and sensory data dissemination techniques to remote locations. Another major design challenge is the distribution of sensory data. Peer-to-peer (P2P) networking is the next generation in distributed Internet computing. Instead of clients and servers, each entity participating in the system can act as both, a client and a server, simultaneously. This technique leaves a provision of a peer with very high resources and processing capabilities or a peer having the least resources to consume any P2P service. P2P is network independent, scalable and can operate over TCP/IP, Bluetooth, and other wired and wireless technologies. As a result, P2P technology shows many attractive features to use it as a service oriented framework for sensory data publishing and collaborative multi-robot environments.

Developing a multi-robot control architecture has been the subject of several studies. Asama et al. proposed ACTor-based Robot and Equipments Synthetic System (ACTRESS) [1]. The ACTRESS architecture tackles the issues of communication protocols with different abstraction levels, path planning, and task assignment through multi-stage communication protocols. Fukuda et al. proposed a biologically inspired CEllular roBOTics system (CEBOT) [2]. Robots within the CEBOT architecture are tightly coupled and can dynamically reconfigure their physical structure in response to different environment changes. GOFER is another distributed multi-robot problem solving architecture [3]. It is based on a centralized task planning and scheduling module, which keeps track of the task allocation and the availability of all the robots through direct communication with them. Despite its satisfactory performance in applications, like following a wall and pushing objects, most of the GOFER implementations involved no more than three robots. In order to alleviate the naturally inherited problems in most of the aforementioned framework, the ALLIANCE architecture was introduced [4], [5]. This architecture emphasizes some key features in the design of multi-robot coordination architectures with a particular focus on fault-tolerance, reliability, and adaptability. The architecture is designed for small to medium-sized teams of loosely coupled heterogeneous mobile robots operating in dynamic environments to perform tasks with possible ordering dependencies. The L-ALLIANCE architecture allows for the automatic tuning of the ALLIANCE control parameters through reinforcement learning techniques [4], [6]. ALLIANCE and L-ALLIANCE architectures have been among the most commonly used architectures for loosely coupled multi-robot cooperation systems. Recent approaches are also

Second International Conference on Systems (ICONS'07)
0-7695-2807-4/07 $20.00 © 2007 IEEE

considering energy consumption as an additional constraint in designing modern cooperative robotic architectures [7]. Reliable inter-robot communication protocols represent a crucial component in multi-robot coordination architectures. Their goal is to secure a favorable communication scheme between the individual robots, which satisfy their respective requirements in exchanging relevant sensory information and control actions. All-to-all broadcasting communication protocols are very inefficient as they flood the network with unnecessary information and processing resources. An alternative solution is suggested by Das et al. [8] who evaluated three ad hoc network protocols to support the communication for a team of mobile robots equipped with different sensors. Hua et al. [9] proposed an Internet-scale framework for publishing, browsing, and analyzing sensory data.

The proposed approach is based on entering search criteria into a local client browser to request information about the sensed environment using a P2P framework. In this paper, we present a P2P service oriented framework for distributed collaborative multi-robot environments. It supports 'n' number of teams of mobile robots, where each team consists of several loosely coupled mobile robots that can autonomously explore a common environment, avoid obstacles, inter communicate among themselves, and supply sensory information to the P2P network. As a proof of concept of the test scenario, the mobile robots are given a task to look for human body in the assigned environment. Once the human body is found, it sends sensory information, such as the image of the person, the location of the sensed human body, and the temperature of the environment to the P2P-based collaborative group. The architecture is developed to satisfy the stringent design requirements of cooperative mobile robots, such as flexibility and fault tolerance allowing any number of mobile robots to dynamically join or retreat from the framework without affecting the service. The main differences between the proposed and the aforementioned architectures are: proposed framework (i) offers an open hardware architecture with physical parameters such as the sensors' spatial and chronological layout, (ii) uses state-of-the-art P2P technology, and hence inheriting all the advantages this technology has to offer including redundancy, fault tolerance, communication security, and full autonomy, to name a few. It is worth mentioning that the framework is specifically designed to satisfy the requirements of physical sensory data publishing and fusion. The rest of the paper is organized as follows: Section II describes the architecture and design of the framework. The prototype of proposed system is illustrated in Section III. Section IV presents the test results we have found. The paper is concluded with concluding remarks in Section V.

II. System Design

The high level architecture of the proposed system is shown in Figure 1. All the peers are connected through the P2P network. We define two types of peers in our design, the Base Peer (BP) and the Service Consumer Peer (SCP). The BP provides the necessary services and the SCPs simply

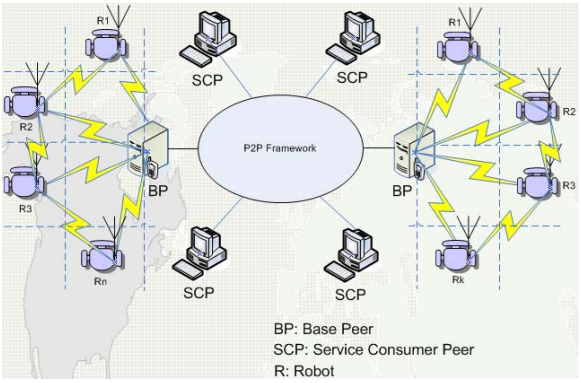

Fig. 1. High level architecture of the framework.

view/receive the provided services. By service we mean taking images and/or other sensory information, analyzing the collected sensory information in the BP using some intelligent algorithms, and finally sending this sensory information to the SCPs to take further action(s) necessary. There might be a number *n* of BPs according to this architecture. Each BP is in wireless contact with several mobile robots. Each robot is equipped with several sensors and can also communicate with other robots. Consequently, the goal of this P2P architecture is to provide scalability by the introduction of new peers without the need to restart or reboot the environment and to support fault tolerance, since a defect of a peer no longer affects the functionality of other peers. In fact, SCPs can switch among one or more BPs in order to get the sensory information of the respective sites. Now we elaborate each subsystem in detail.

A. Base Peer Architecture

The BP receives different sensory data, such as ambient temperature, distance of detected obstacles and objects, and images in real-time from each robot if accompanied with cameras. In order to process and disseminate those sensory data efficiently, we propose several components inside the BP as shown in Figure 2. The BP maintains a *FCFS (First Come First Serve) queue*, which receives frames of sensory data coming from different robots. The sensory data *Filter Module* intakes one incoming frame from the queue, logically divides them into chunk of OCTETs, and passes each of them to the appropriate sensory data processing modules. Sensory data falls into numerous categories. Some sensory data needs only to be received from the lower layers, and be converted to high level data while some sensory data, such as image, requires to be processed using intelligent algorithms, which might be further necessary for any specific application. Accordingly, we deploy two components, one is the application independent *Sensory Data Processing Engine* and the other is the *Application Specific Sensory Data Processing Module*. In order to make the system deployable for any particular application, only the application specific module needs to be customized. The *Sensory Data Processing Engine* receives all the sensory data except the application specific one; if any. It decodes the binary patterns of the frame and parses

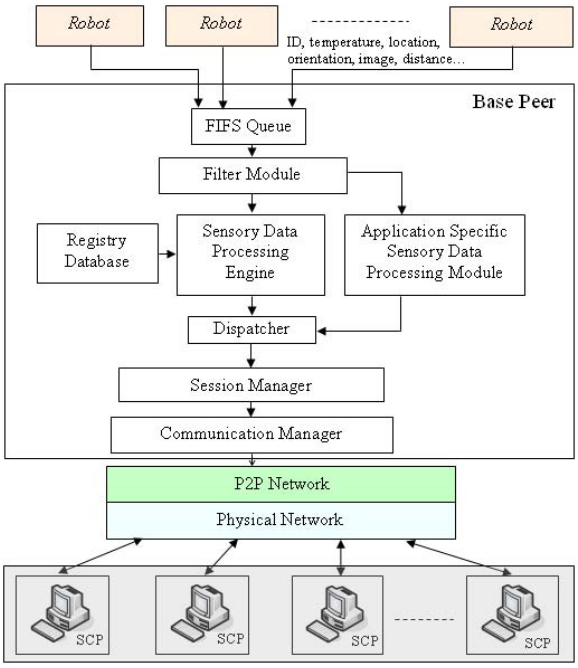

Fig. 2. Base Peer (BP) components.

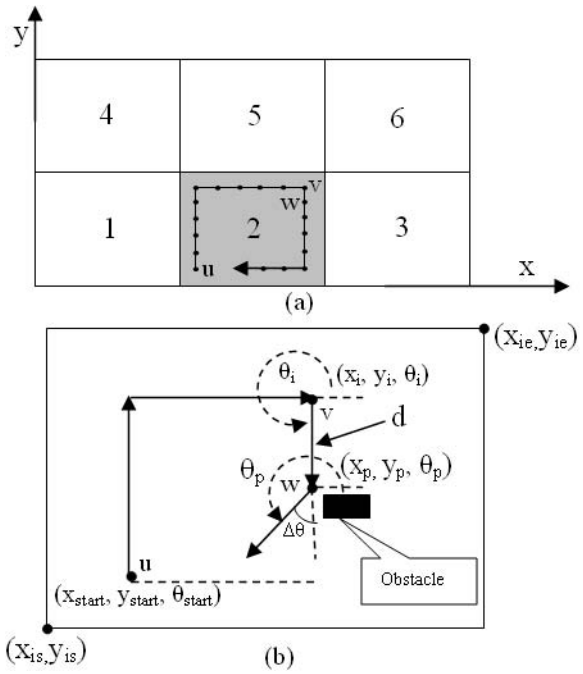

Fig. 3. (a) Dividing the test region into blocks (b) Position and orientation of a robot inside a block.

them into high level sensory data. This component is also responsible for figuring out the obstacles' and robots' position relative to the environment. For this, the BP needs some more parameters that are kept in an another component named the *Registry Database*. This database keeps a record of each robot deployed in the environment such as the robot ID, the absolute geometry and co-ordinate of each block, the initial location and orientation of each robot etc. By joining the sensory data sent by a robot with its record set inside the registry, the *Sensory Data Processing Engine* can calculate the obstacle location relative to the test environment. After processing, this unit dispatches the following information to the *Dispatcher* unit: the ambient temperature and the physical location of the detected obstacle. The *Application Specific Sensory Data Processing Engine* is application specific and might vary from set-up to set-up. In our current architecture, we intend to use some multimedia applications such as image compression technique within the framework. Because the end-to-end communication is P2P, compressing the image data will require less bandwidth for each SCP so that any SCP having dial-up connection can even use the framework with a satisfactory data rate. Because the above two sensory data processing modules might take uneven time, the *Dispatcher* unit receives the high level sensory data until they all arrive, combines them, and sends them to the P2P communication layer. We have created a service layer on top of the P2P overlay network in which BPs and SCPs form a collaboration group. Each peer consists of four logical layers: 1) Collaborative Application (CA); 2) Workspace Manager (WM); 3) Session Manager (SM); and 4) Communication Manager (CM). All of these four layers operate on top of the P2P platform, which hides the physical network lying underneath it. Due to the fact that we designed the BP to run without any human operator, it does not need the CA and the WM layers. A BP needs only the SM and the CM layers. However, each SCP needs all the four logical layers in order to receive the necessary services.

B. Robot Positioning System

In order to define the robot's instantaneous position, we first describe how a robot calculates the distance it traveled and its angular rotation. The distance traveled can be easily calculated using the physical parameters of the robot's wheels and the encoders associated to their respective driving motors, such as their radius, gear ratio, sensitivity, etc. The next level of positioning to be considered is to identify the robot's position with respect to the environment under surveillance. Before defining such a positioning system, we need to take into consideration the geometry of the test environment. In this research work, we have chosen the Multimedia Communication Research Laboratory (MCRLab) and the Machine Intelligence, Robotics and Mechatronics (MIRaM) Laboratory at the University of Ottawa as the virtual target environments. Although we logically divide the lab spaces into six blocks (see Figure 3(a)), the architecture is not limited to the geometry of the blocks. Every block i is defined by two corner points (x_{is}, y_{is}) and (x_{ie}, y_{ie}). We also define the initial position and orientation of a robot inside a block by $(x_{start}, y_{start}, \theta_i)$ as shown in Figure 3(b).

C. Localization and Inter-robot Communication

Once the physical movement space for each robot is assigned, the next challenge lies in localizing the robots [10]

in their respective blocks. Let us consider a single robot operating in one of the blocks as shown in Figure 3(b). The movement of the robot falls into two categories. One is when it moves without any hindrance and the other is when it faces an obstacle. In the first case, the robot moves autonomously in its pre-assigned block avoiding drifting outside the block. As it moves, it periodically calculates its new position and orientation based on the previous position, orientation and the distance traveled. When the robot encounters no obstacle, it travels inside its block in a routinely pre-defined path as illustrated in Figure 3(a). In the second case, if it faces an obstacle as depicted in Figure 3(b), the robot re-plans its path iteratively as defined by (1).

$$\left.\begin{aligned} x_p &= x_i + d\cos\theta_i \\ y_p &= y_i + d\sin\theta_i \\ \theta_p &= \theta_i + \Delta\theta \end{aligned}\right\} \quad (1)$$

where, (x_p, y_p) and (x_i, y_i) are the new and old coordinates of the robot, while θ_p and θ_i are its new and old orientation, respectively. If the robot finds an obstacle around it after traveling distance d, then it will try to re-orient itself by $\Delta\theta$. In this work, $\Delta\theta$ is computed with the means of a Fuzzy Inference System (FIS) [11]. The robot then maintains its new orientation until it finds another obstacle or reaches the border of its working block. In the latter case, it resets its location to $(x_{start}, y_{start}, \theta_i)$ and goes back to its starting point and orientation within that block.

Moreover, for whatever reasons and when deemed necessary the proposed architecture enables a robot to cover up for another robot by taking over its territory (in addition to the original robot's territory). This situation may arise in cases where a robot needs to recharge or retreat for maintenance, for example. This strategy gives a certain degree of fault tolerance to the system. An efficient inter-robot communication mechanism is of crucial importance for such a scheme. Each robot generates a unique wireless beacon signal at predefined time intervals, which carries its ID. Each robot also maintains a dynamic neighborhood table similar to Table I. The neighborhood table consists of every robot's ID, absolute start location, current start location, absolute end point, current end point and neighbor information. A robot always looks for its current start and/or end location from the table while initializing its journey. The values of the neighbor columns are the directions of the neighbor(s) (left, right, top and down), where 'none' represents not a neighbor and empty refers to the record of the robot itself. When any robot fails to send the beacon signal to its neighbors within a threshold time, the framework offers a mechanism to enable its neighbors to decide who would take over the failed robot's block. Consider a simple scenario as in Figure 3(a) where the robot at block 2 failed to send its beacon signal to its neighbors 1, 5 and 3 within the predefined time. Robots 1, 5 and 3 will detect that failure. Each of them will execute a pre-defined algorithm to decide who will take over depending on the back off time. The neighbor whose back off time expires first (say robot 1, for instance), will take over the inactive robot's block. As soon as robot 1 penetrates block 2, it sends a beacon signal to its new neighbors (3 and 5) to inform them that it is now replacing robot 2. Upon receiving the beacon signal, robot 3 and 5 simply give up the intention of taking over.

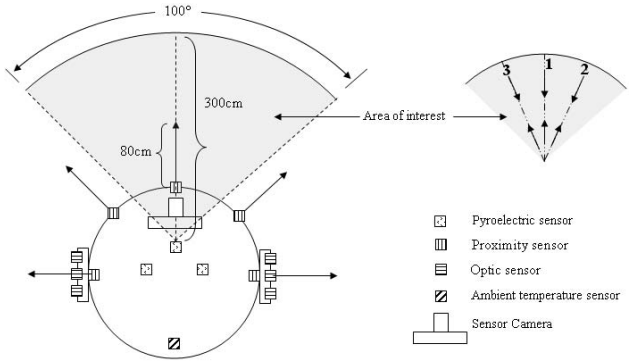

Fig. 4. Spatial layout of sensors.

D. Dynamic Behavior of a Robot

Dynamic behavior refers to how the robot behaves when it faces an obstacle while moving around its territory. The dynamic behavior of the robot falls into two categories: 1) when the robot simply detects a non-target object, such as a wall or a rock, which will then be treated as an obstacle; and 2) when it detects its target, which is a human body in this case. In our current framework, each robot employ the FIS to avoid obstacles, where the input from five infrared proximity sensors are used (see Figure 4). These inputs are left obstacle distance, left corner obstacle distance, front obstacle distance, right corner obstacle distance and right obstacle distance. The output in this case is the change of the robot's orientation $\Delta\theta$. When the robot comes across its target, it senses the obstacle, collects the sensory data depending on its designated mission, and then avoids it using the FIS. For example, when any human body comes within the range, the robot stops for a predefined time, collects images of the human body and other sensory data such as the distance of that person, the ambient temperature of the environment and its current position and orientation to send to the BP. The robot will move along the paths 1, 2 and 3 respectively (see Figure 4) inside the area of interest and follow the above activities. While following the three paths, even if it faces obstacle in one of the paths, it will still follow the rest of the paths to capture more sensory data for greater coverage of the area. The above explanation is also applicable for the left and right side obstacles. We also consider two extreme cases of obstacle detection in our design. The first one is when the obstacle is in the border of two blocks and the second one is when the obstacle is at the corner point of four blocks (we have two such corners as shown in Figure 3(a)). In the first case, both robots of the respective blocks will detect the obstacle and send the sensory information to the BP as stated above. For the second case, all the four robots will detect the obstacle and follow the same procedure.

TABLE I

NEIGHBORHOOD TABLE OF ROBOT WITH ID 2

ID	Absolute Start Location	Current Start Location	Absolute End Location	Current End Location	Neighbor
1	$(x_{1start},y_{1start},\theta_1)$	$(x_{1start},y_{1start},\theta_1)$	(x_{1e}, y_{1e})	(x_{1e}, y_{1e})	left
2	$(x_{2start},y_{2start},\theta_2)$	$(x_{2start},y_{2start},\theta_2)$	(x_{2e}, y_{2e})	(x_{2e}, y_{2e})	
3	$(x_{3start},y_{3start},\theta_3)$	$(x_{3start},y_{3start},\theta_3)$	(x_{3e}, y_{3e})	(x_{3e}, y_{3e})	right
4	$(x_{4start},y_{4start},\theta_4)$	$(x_{4start},y_{4start},\theta_4)$	(x_{4e}, y_{4e})	(x_{4e}, y_{4e})	none
5	$(x_{5start},y_{5start},\theta_5)$	$(x_{5start},y_{5start},\theta_5)$	(x_{5e}, y_{5e})	(x_{5e}, y_{5e})	up
6	$(x_{6start},y_{6start},\theta_6)$	$(x_{6start},y_{6start},\theta_6)$	(x_{6e}, y_{6e})	(x_{6e}, y_{6e})	none

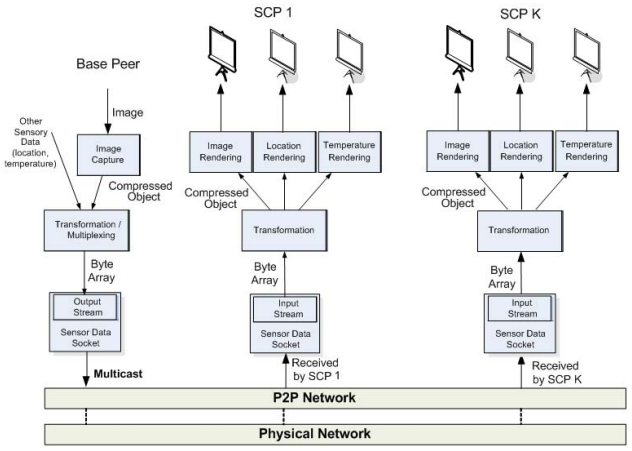

Fig. 5. Socket communication architecture.

E. P2P Service Oriented Framework

In this section we will assume an image capturing application to facilitate the explanation of the framework. As Figure 5 portrays, the image that the BP wants to send to SCPs is first compressed and then transformed into byte arrays. In turn the byte arrays are passed through multicast sockets, which convert the byte arrays into output streams that eventually use the P2P network for transportation. To send an image stream to N peers simultaneously, the CM layer uses N sockets to multiplex the image stream. At the receiver end, the dedicated socket receives the stream and generates a byte array, which in turn is passed to the transformation engine to reconstruct the compressed image object. The image rendering module decompresses the image frame to render it in the Graphical User Interface (GUI). The location information of obstacles is similarly sent by the SM layer of the BP as an uncompressed message through the socket, which is finally received by the CA layer of an SCP. The location renderer then simply draws a virtual 2D space based on the geometry of the remote site and points the obstacle in that space. Other sensory data, such as temperature for instance, can be similarly sent by the BP and received by the CA layer of the SCPs to render them to the appropriate location in the GUI.

III. IMPLEMENTED PROTOTYPE

In this research work, we have chosen different types of sensors such as infrared proximity sensors, optic sensors, temperature sensors, pyroelectric sensors, and sensor cameras to be mounted on the mobile robots. The sensors exhibit two different models in our design: a spatial and a temporal model. Because the sensors are mounted on robots, the robots' space model is also taken into account in the sensors' spatial layout design. Geometric placement of the sensors helps to perceive sensory data with more precision [12]. The spatial model is shown in Figure 4. Each robot carries three pyroelectric sensors that cover a front angle of approximately 100 degrees (shaded region of Figure 4). These sensors are capable of detecting heat radiating bodies, such as humans and animals.

We also use five infrared proximity sensors mounted at the front, left and right sides of the robot for detecting objects at the corresponding sides. An infrared proximity sensor emits an electromagnetic field or beam toward its line of sight and looks for changes in the field. Three optic sensors are equipped with each robot wheel. The data from these optic sensors are treated as odometry information that is used for the calculation of relative position and orientation of the robot. A temperature sensor is placed at the back of the robot to measure the ambient temperature of the environment. A sensor camera is also mounted on each robot to enable it to take snapshots and send them to the BP whenever needed.

The temporal modeling of some of these sensors refers to the fact that they are triggered with predefined time events. The optic sensors, for example, are triggered as soon as the robot starts its mission. The pyroelectric sensors are triggered when they detect infrared radiating objects that come within their sensing range. The infrared proximity sensors, on the other hand, do not fire unless an obstacle is detected within a rectilinear distance from its sensing element. For demonstration purposes, we assume that the robots are deployed for a rescue mission where they are supposed to patrol a particular area and send images and coordinates of living subjects (humans and animals) as well as the environment's ambient temperature to the BP. As such, as soon as the infrared proximity sensor fires, two types of sensors follow: the camera takes a photo image of the object and the temperature sensor measures the ambient temperature of the environment.

IV. TEST RESULTS

In order to test the efficiency of our proposed framework, we deployed the robots in the MCR and MIRaM Labs at their initial positions. We then tested the robots with static and moving humans and non-human objects. The robots could successfully locate both types of subjects and send only images and locations of the target objects and the environment temperature to the BP. We also tested the prototype in the

TABLE II
AVERAGE ROUND TRIP DELAY BETWEEN BP AND SCPS

BP location	Connected SCPs	Round trip delay (in ms) for	
		image	other sensory data
MCRLab	5	91.3	67.5
	4	89.5	
	3	89.2	
MIRaM	5	90.5	67.5
	4	89.1	
	3	90.6	

case where one of the robots is temporarily taken out for maintenance. We analyzed the sensory data communication delay from the BP to the SCPs. Each BP multicasts the sensory data only to the SCPs of the same session. So, the type of communication is one to many in this case. Because the system is loosely coupled, we did not employ any Network Time Protocol. Instead, we took into consideration the round trip delay among peers to measure the latency. We define end to end delay as follows:

Sensory data processing delay (at BP) + network delay (at P2P network) + network resynchronization delay + processing resynchronization delay (at SCP) + processing delay (at SCP).

We took 11000 instances of delay for each of the sensory data and depicted only the average delay. For the test results, we have considered the image, the location of the detected obstacle relative to the test environment and the temperature of the environment. Among the sensory data, the image module in the BP requires some processing for compression during which other sensory data are stored in the *Dispatcher* unit of the BP. The image frame delay represents the average transmission delay of sending an image frame from a BP and receiving the response from all the SCPs of the group, taking into consideration that the frame travels from the SM layer of the BP and reaches the application layer of SCPs in the same session and finally the acknowledgment message again reaches the SM layer of the sender. The average round trip delay of 5 SCPs connected to the MCRLab and MIRaM Lab sessions was approximately 91.3 and 90.5 milliseconds, respectively. In the case of 4 SCPs connected to both Lab sessions, the delay was 89.5 and 89.1 milliseconds, respectively. For 3 peers, the delay we found was 89.2 milliseconds for the MCRLab and 90.6 milliseconds for the MIRaM Lab sessions, respectively. The test results are summarized in Table II. Because the obstacle coordinates and the ambient temperature are simply text data, the round trip delay was significantly less than that of image data. Also, the coordinates and temperature data processing delays in the BP are in the order of microseconds. Hence, they can be neglected when compared to the round trip delay. Since the test results reflect the delay in the form of the round trip delay, the end-to-end delay is indeed less than the delay in each case once we subtract the returning time of the trip. The returning time is the time required for the acknowledgment message from the SCPs indicating that the sensory data is successfully received.

V. CONCLUSION

In this paper we described the design and implementation of a P2P service oriented environment, which is suitable for delivering sensory information from a remote site with the aid of different sensors intelligently mounted on each of the mobile robots. The sensors are designed in space and time and later on, are mounted on the robot in such a way that they can capture sensory data optimally. Then, through a designated P2P platform, the BP can disseminate these sensory data to the SCPs. The proposed framework is shown to be quite scalable as it can accommodate a large number of mobile robots without affecting the quality of service offered. In addition, it is also equipped with a fault-tolerance mechanism to overcome possible failures in one or more of the robots involved.

ACKNOWLEDGMENT

The authors would like to acknowledge the financial assistance of the Natural Sciences and Engineering Research Council of Canada (NSERC) and the Ontario Research Network on E-commerce (ORNEC).

REFERENCES

[1] H. Asama, K. Ozaki, A. Matsumoto, Y. Ishida, and I. Endo, "Development of task assignment system using communication for multiple autonomous robots," *J. Robot. Mechatron.*, vol. 4, no. 2, pp. 122–127, 1992.

[2] T. Fukuda and G. Iritani, "Construction mechanism of group behavior with cooperation," in *Proc. IEEE/RSJ IROS'95*, 1995, pp. 535–542.

[3] C. LePage, "A combination of centralized and distributed methods formulti-agent planning and scheduling," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 1990, pp. 488–493.

[4] L. E. Parker, "Alliance: An architecture for fault tolerant multirobot cooperation," *IEEE Transactions on Robotics and Automation*, vol. 14, no. 2, pp. 220–240, April 1998.

[5] L. E. Parker, F. Schneider, and A. Schultz, Eds., *Multi-Robot Systems: From Swarms to Intelligent Automata*, 2005.

[6] F. Fernandez, D. Borrajo, , and L. E. Parker, "A reinforcement learning algorithm in cooperative multi-robot domains," *Journal of Intelligent and Robotic Systems*, vol. 43, no. 2-4, pp. 161–174, 2005.

[7] Y. Mei, Y.-H. Lu, Y. C. Hu, and C. S. G. Lee, "Deployment of mobile robots with energy and timing constraints," *IEEE Transactions on Robotics*, vol. 22, no. 3, pp. 507–522, June 2006.

[8] S. Das, Y. Hu, C. Lee, and L. Yung-Hsiang, "Supporting many-to-one communication in mobile multi-robot ad hoc sensing networks," in *IEEE International Conference on Robotics and Automation*, vol. 1, 2004, pp. 659–664.

[9] K. A. Hua, R. Peng, and G. L. Hamza-Lup, "Dissemination of sensor data over the internet," in *Proceedings of the Embedded and Ubiquitous Computing, International Conference EUC 2004*. Aizu-Wakamatsu City, Japan: Springer, August 25-27 2004, pp. 745–754.

[10] J. Borenstein, H. R. Everett, and L. Feng, *Navigating Mobile Robots: Sensors and Techniques*, J. Borenstein, Ed. A. K. Peters, Ltd., Wellesley, MA, 1996.

[11] D. Parhi, "Navigation of mobile robot using a fuzzy logic controller," *Journal of Intelligent and Robotic Syst.*, vol. 42, no. 35, pp. 253–273, March 2005.

[12] L. Navarro, J. Dolan, and P. Khosla, "Optimal sensor placement for cooperative distributed vision," in *Proceedings of the International Conference on Robotics and Automation (ICRA)*, April 2004, pp. 939–944.