Spinning Sensors: Middleware for Robotic Sensor Network

Soko Aoki*, Yukihiko Kirihara†, Jin Nakazawa* and Hideyuki Tokuda*
*Graduate School of Media and Governance, Keio University, Kanagawa, Japan
Email: {soko, jin, hxt}@ht.sfc.keio.ac.jp
†Triple Double Corporation, Tokyo, Japan
Email: kirihara@triple-double.co.jp

***Abstract*—This paper proposes Spinning Sensors, a middleware system for creating robotic sensor network applications. This middleware enables application programmers to easily write application software which utilizes both sensors and actuators in their network. In the Spinning Sensors model, a sensor node is attached to a robotic actuator to change its position and/or direction. The continuous position change enables the single robotic sensor node to cover greater sensing area and sensing time. This paper describes robotic sensor node model that the sensor and the actuator are attached together as one node, then present our design and implementation of the software which achieved both versatility and functionality simultaneously. We constructed three applications using the middleware: an environment monitoring, a radio control robot, and context-aware services. We show the result of experiment in which we utilized multiple robotic sensor nodes to monitor the environment.**

KEYWORDS

Sensors, Actuators, Robotic Sensors, Sensor Network, Coordination, Middleware, Robotic Interactions with Sensors

I. Introduction

A variety of sensor-based ubiquitous computing applications, such as environment monitoring, industrial monitoring, and context acquisition are proposed. The major issue in these applications is achieving maximum effectiveness (e.g. coverage and granularity of sensing targets) with minimum consumption (e.g. the number of sensors and maintenance cost). One approach to construct a sensor network system with large coverage and high granularity is to deploy as many sensor nodes as possible in the environment. This is feasible in case the sensor node is cheap. The other approach is to make a sensor node mobile so as to change its coverage area dynamically. This approach is feasible when the sensor node is expensive or when the administrator wants to reduce the number of the sensor nodes.

We propose Spinning Sensors system that increases coverage of a sensor node, and decreases the number of sensors required in an application by realizing a robotic sensor node. In the system, a sensor node is attached to a robotic actuator in order to move or rotate the sensor node to follow moving objects or to increase its coverage, respectively. The three major features of Spinning Sensors are hardware abstraction of sensors and actuators, communication and coordination mechanism for sensors and actuators, and application programming interface for developers.

As the newest information system is composed of heterogeneous general purpose information devices, the sensor network system will be composed of general purpose sensors and actuators in the near future. Since there are many kinds of sensors, actuators, and robots in our network, the Spinning Sensors needs to hold versatility and functionality simultaneously. In terms of versatility, we have divided the hardware control software into two classes: abstract class and implementation class. With this design, we could separate hardware specific programming and general purpose middleware. The communication and coordination mechanism is provided by this middleware layer so that the application programmers can easily construct multiple robotic sensor nodes environment.

The contributions of this paper are the following. Spinning Sensors showed the concept of putting together general purpose sensors and actuators to increase the sensing coverage and sensing granularity by showing the design and implementation of Spinning Sensors. The middleware of Spinning Sensors realized both versatility and functionality by adopting modular design of the software so that the system can be utilized for many kinds of hardware devices and applications. Finally we could show the result of implementation of Spinning Sensors by showing the experiment data telling that there is an improvement in the quality of sensed data by using the Spinning Sensors model.

The rest of the paper is organized as follows. Sections 2 categorizes the conventional sensors and actuators and explain the necessity of the robotic sensor node which combines general purpose sensors and actuators as one node. Section 3 presents the design and implementation of Spinning Sensors. In section 4, we show the result of experiment using multiple robotic sensor nodes. Section 5 surveys related work and section 6 concludes this paper.

II. Robotic Sensor Network

In this section, we classify the sensors and actuators by their characteristics. Sensors can be classified by two criteria: sensing range and sensing direction. Some of sensors cover a few meters (long range) and some others cover only a few centimeters (short range). The sensing angles also vary. For example, thermometers and hygrometers cover 360 degrees (non-directional) in a sense that they measure the condition of surrounding air. Contrarily, illuminometers, cameras, and

ROBOCOMM 2007, 14th − 16th Oct 2007, Athens, Greece.
Copyright © 2011 − 2012 ICST
ISBN 978-963-9799-08-0
DOI 10.4108/ICST.ROBOCOMM2007.2198

Node	Spatial Model	Measurement
Sensor	● Coverage: Circle	Area
Sensor	▲ Coverage: Sector	Area
Actuator	↔ Movement: Linear	Distance
Actuator	↻ Movement: Circular	Angle
Fusion	▲ × ↻ : Area = Sector Area x Angle	
Fusion	▲ × ↔ : Area = Sector Area x Distance	
Fusion	● × ↔ : Area = Circle Area x Distance	

Fig. 1. Robotic Sensor Node Model

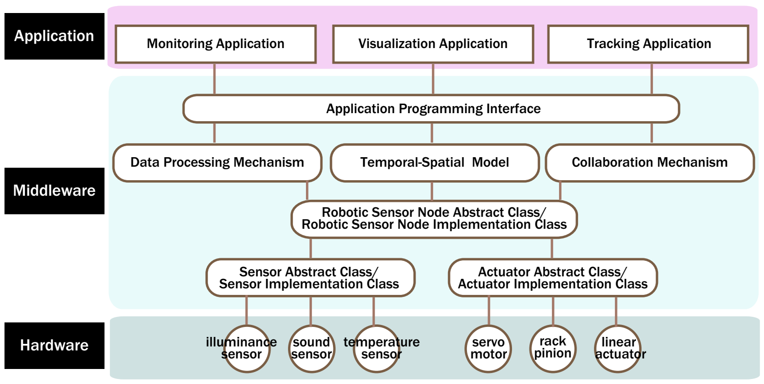

Fig. 2. Spinning Sensors System Architecture

microphone cover limited range of angle (directional). If, for example, an illuminometer is placed on a table, on which a table lamp is placed, the light value would vary according to the direction the illuminometer faces. For these kinds of directional sensors, the position and direction they are placed is important, therefore the actuators which can change the position and direction of directional sensors have significant value.

We assume two kinds of actuators using motors; one is translatory actuators which move linearly, and the other is rotational actuators which move circularly. These two kinds of actuators are well-suited to be attached to the pillar or ceiling of the room in terms of size and shape. Translatory actuators include pistons, cylinders, belt conveyors, lifters, and traveling rail with a car moving on it. By using these translatory actuators, the sensors attached to these actuators can change their coverage area linearly. Rotational actuators includes servo motors and stepping motors. With these rotational motors, the sensors attached to them can change their facing direction by changing the motors' rotational state. Both kinds of actuators increase the area covered by a sensor as shown in Fig 1.

III. Design and Implementation

In this section, we describe the design and implementation of the Spinning Sensors system, which dynamically senses environment and objects by actuating both sensors and actuators. We have implemented a prototype of Spinning Sensors by using Java programming language. Spinning Sensors consists of three layers: hardware layer, middleware layer, and application software layer as shown in Fig 2. The hardware layer mainly has two modules: interface to sensors and interface to actuators. The middleware layer manages the sensors and actuators and provides communication and coordination mechanism for them. This layer also provides Spinning Sensors application programming interface (API) so that the application software can utilize the functionalities which the Spinning Sensors provides. We constructed three kinds of prototype applications: environment monitoring, radio control robot, and context-aware service.

The design principle of the Spinning Sensors can be divided into two challenges. The one is the versatility of the middleware and the other is the functionality of the middleware.

There are many kinds of sensors and actuators to be a part of our robotic sensor network environment. There are also many kinds of applications which can be realized by sensors and actuators. To cope with this heterogeneity, we adopted the layer architecture as shown in Fig 2. By adopting this modular architecture, we realized the versatility of middleware system.

By using the robotic sensor network, the application programmer may want to write various kinds of application software. To maximize the potential of robotic sensor network environment, we mainly designed three kinds of functions: data processing, multiple nodes coordination, and temporal spatial modeling. These functions are provided to the application programmers through the Spinning Sensors API.

A. Hardware Layer

The hardware layer includes the basic software to control sensors and actuators. In particular, this layer provides the interface to a temperature sensor, an illuminometer, a movement sensor, a servo motor, and a rack pinion. To provide the versatility, this layer is divided into two software modules: abstract class and implementation class. We have developed the implementation classes as shown in Table I. All these implementation class extends either an abstract sensor manager class or an abstract actuator manager class.

B. Middleware Layer

The coordination mechanism let the application programmer easily writes the application which uses multiple robotic sensor nodes. It is highly possible that the application programmer would like to construct one robotic sensor node by using one sensor and one actuator. In this case, the application programmer uses the fusion class which is provided by Spinning Sensors middleware. This fusion class is the abstract class which integrates more than two sensors and actuators as one node. In addition to the fusion class, the programmer can attach the observer function to each robotic sensor node so that each of the nodes can change its state according to the other node connected to itself by this observer.

By implementing the fusion class, the programmer can easily combine more than two devices and create a new self-defined robotic sensor node. Since each of these nodes are

TABLE I
LIST OF SUPPORTED SENSORS AND ACTUATORS

Group	Hardware Name	Function	Implemented Class Name	Line	Size
Sensor	TECO uPart	light, temperature, movement	UpartSensorImpl	117 lines	2.83KB
Sensor	LEGO Mindstorms	light, sound, ultrasonic, touch	MindstormsSensorImpl	82 lines	2.18KB
Sensor	Phidgets	temp, light, rotation, slider, etc.	PhidgetsSensorImpl	98 lines	2.22KB
Sensor	Phidgets RFID	RFID reader and tags	PhidgetsRFIDImpl	135 lines	2.70KB
Robotic Actuator	LEGO Mindstorms	motor	MindstormsActuatorImpl	107 lines	2.81KB
Robotic Actuator	Phidgets	motor	PhidgetsActuatorImpl	126 lines	2.94KB
Service Actuator	Aviosys IPPower	power control	IpPowerImpl	62 lines	1.68KB
Service Actuator	JFreeChart	graph viewer	DataViewerImpl	194 lines	5.95KB
Service Actuator	Apple Quicktime	movie player	VideoControllerImpl	57 lines	1.46KB

TABLE II
LIST OF IMPLEMENTED PROTOTYPE APPLICATIONS

Name of Application	Function	Implemented Class Name	Hardwares	Line	Size
Environment Monitoring	Environment Monitoring	EnvMonitoring	Sensor, Motor,and Chart	150 lines	4.89KB
Radio Cotrol Robot	Robot Controlled by Sensors	SensorControlRobot	Sensors and Robot	183 lines	5.23KB
Context-aware Service	Light ON/OFF by RFID	RFIDControlLight	RFID and Power Control	109 lines	2.93KB
Context-aware Service	Movie ON/OFF by RFID	RFIDControlVideo	RFID and Movie Player	108 lines	2.94KB

attached with the observer class, when the sensed value of one sensor changes, the message will be automatically sent to another node to work collaboratively. All the communication between several nodes is done by using this event driven architecture. Since the Spinning Sensors provides its application programming interface to the application programmer, they can write the sensors and actuators implementation class easily and can register these implementation classes to the middleware.

C. Application Layer

The programmers can use Spinning Sensors API to create application software. We developed three kinds of application prototype as shown in Table II. All these applications are written by extending the abstract fusion class. First one is the environment monitoring application which uses UpartSensorImpl class and DataViewerImpl class to show the room's illuminance in the graph type of GUI. The second one is the radio control robot application which uses PhidgetsSensorImpl class as controllers and MindstormsActuatorImpl class. The third one is the context-aware service which uses PhidgetsRFIDImpl class, IpPowerImpl class, and VideoControllerImpl class as services activated by nearing RFID tag.

IV. EXPERIMENT

Three kinds of experiments using Spinning Sensors middleware were conducted to evaluate usability and performance of robotic sensor nodes.

The first experiment uses a pair of an light sensor and a servo motor. This robotic sensor node is placed on a desk and one spotlight which outputs directional light is placed at multiple locations. Since the light sensor is attached onto the rotating motor, the sensor outputs different values according to the direction the robotic sensor node faces. We placed the spotlight at four positions. The position 1 is 8cm away from the node. The position 2 is 16cm away from the node. The position 3 is 24cm away from the node. And the position 4 is 16cm away from the node but the light is non-directional one. Fig 3 shows the result of this experiment. In every position, the light is settled straight in front of the robotic sensor node. In these three positions, the result differs depending on the distance from the sensor node to the light. In position 1, the sensor node is placed too close to the light so that the sensor can not verify the direction of the light. In position 2, since the distance between the sensor node and the light is appropriate, the sensor could figure out the directions of the light properly. In position 3, the distance between the sensor node and the light is a little too far so that it is hard to figure out the directions of the light. In position 4, since we used the non-directional light, the result shows that we can not figure out the direction of the light from the sensed value.

The second experiment uses a sound sensor, an ultrasonic sensor, a light sensor, and a robot which can move around. One speaker making pink noise, a paper box, and a spotlight are placed around this robotic sensor node. Fig 4 shows the result of this experiment. In terms of light sensor and ultrasonic sensor, we can figure out the placed angles of a spotlight and position of a paper box. However, since the speaker's sound proliferated, the output value of sound sensor does not differ in all positions. From these experiments, we could do the operation check of our middleware and application. Furthermore, we found that the robotic sensor node can cover greater directions than the fixed sensor node and could even figure out the object's facing direction or position. By using the middleware and the robotic sensor nodes, we could know about our environment more accurately.

The third experiment uses a rotation sensor, a slider sensor, and a motor. In this experiment, either the rotation sensor or the slider sensor is utilized as a controller of the motors. We measured the total time required for sensor sensing, event dispatch, and actuator execution. Fig 5 shows the result. The result shows that when we increase the total number of motors to control, the total time increased. This result also shows

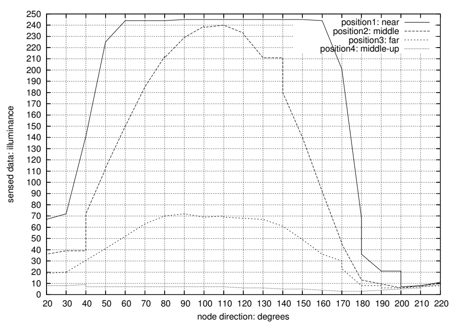

Fig. 3. Experiment: Light Sensor and Motor

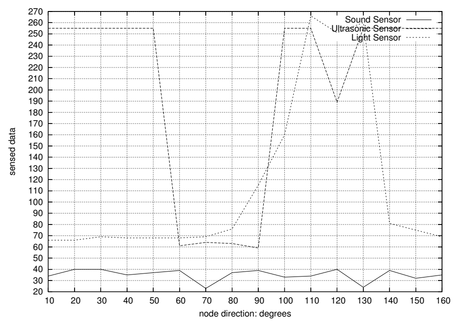

Fig. 4. Experiment: Sound Sensor, Ultrasonic Sensor, Light Sensor and Robot

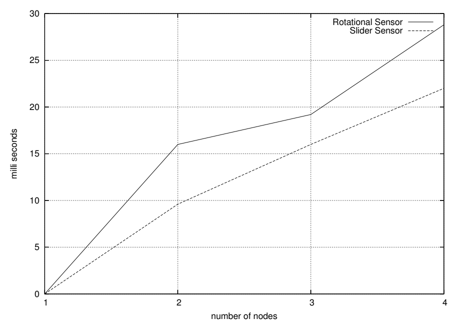

Fig. 5. Experiment: Rotational Sensor, Slider Sensor and Motor

that the required time is less than 30 milli-seconds and it is reasonable result especially in the environment such as home or office where there is not strong need of real-time applications.

V. Related Work

Reference[1] describes a sensor network application construction kit. In this research, they have developed a configuration programming language called SNACK to reduce memory and power consumption of sensor nodes especially for the Crossbow's MOTE hardware. They focus attention on the NesC's inefficiency and do not pay much attention on general versatility, spatial modeling, or application for robotic sensor nodes.

There is also an issue of sensor coverage optimization. Reference[2] and Reference[3] proposed an approach to establish a sensor network with robotics so that the sensor can move freely to change its coverage area. Although this approach can increase the mobility of sensors and their coverage area, the whole architecture tends to be complicated and the power consumption would be relatively high. Reference[4] is an experimental laboratory for pervasive computing research and is equipped with various kinds of stable sensors. Spinning Sensors gets the best of both model, mobility of robotic actuators and simple design and low maintenance cost of stable sensors.

RT Middleware[5] is a middleware for robotics. It realizes modular design of robotic software based on distributed component technology called CORBA [6]. Since this technology is based on CORBA, the developers are forced to install many kinds of software before they use RT Middleware. Although they succeeded in realizing modularity of robotic actuators, there is no discussion regarding the robotic sensor node.

Reference[7] discusses the sensor's exposure in wireless adhoc sensor networks. Although this research provides valuable formulation and experimental results by using assuming stable sensors, Spinning Sensors differs from this research in mobility of sensors. Spinning Sensors optimizes the sensor coverage area by not increasing the number of sensor nodes nor their power usage, but spinning the sensors themselves.

VI. Conclusion

In this paper, we presented Spinning Sensors, a novel sensors and actuators collaborative usage model realizing dynamic adaptation of coverage and increasing the coverage area. Spinning Sensors model illustrates the relationship among sensors, actuators, and target objects. Spinning Sensors system consists of three layers: sensors and actuators driver layer, middleware layer, and application layer. The middleware is designed and implemented to achieve versatility and functionality so that it can be used for many kinds of hardware devices and application programs. The experiment using the Spinning Sensors showed that a light sensor and ultrasonic sensor output wide range of data depending on the direction it towards. Therefore spinning the sensors is meaningful especially in case the sensor has unidirectional characteristic.

With the emerging technology of robotic sensor network, the internet not only can acquire real world environmental and physical data but also the physical feedback to the real world can be realized using the robotic actuators and robots. And as in the case of the current information system, the sensor network system might be built not by using the application specific hardware but by utilizing the general purpose sensors and actuators. When integrating these general purpose hardware devices the middleware would play an important role to connect each device. In this way, Spinning Sensors can accelerate the integration of real world and virtual world by providing the new middleware for robotic sensor network.

References

[1] B. Greenstein, E. Kohler, and D. Estrin, "A sensor network application construction kit (snack)", in *The Second ACM Conference on Embedded Networked Sensor Systems (Sensys)*, 2004.

[2] A. LaMarca, W. Brunette, D. Koizumi, M. Lease, S. Sigurdsson, K. Sikorski, D. Fox, and G. Borriello, "Making Sensor Network Practical with Robotics", in *Proceedings of 1st International Conference of Pervasive Computing*, Aug. 2002.

[3] M. Laibowitz and J. Paradiso, "Parasitic Mobility for Pervasive Sensor Networks", in *Proceedings of 3rd International Conference of Pervasive Computing*, May 2005.

[4] S. Intille, K. Larson, E. Tapia, J. Beaudin, P. Kaushik, J. Nawyn, and R. Rockinson, "Using a Live-In Laboratory for Ubiquitous Computing Research", in *Proceedings of 4th International Conference of Pervasive Computing*, May 2006.

[5] N. Ando, T. Suehiro, K. Kitagaki, T. Kotoku, and W. Yoon, "Rt-component object model in rt-middleware - distributed component middleware for rt (robot technology) -", in *IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA)*, 2005.

[6] Object Management Group (OMG), "OMG: CORBA Component Model", 1999, http://www. corba.org/.

[7] S. Meguerdichian, F. Koushanfar, G. Qu, and M. Potkonjak, "Exposure in wireless ad-hoc sensor networks", in *International Conference on Mobile Computing and Networking (MOBICOM)*, 2001.