Functionally Distributed Control Architecture for Robot Systems

Tetsuya Taira
School of Science for
Open and Environmental Systems
Keio University
Yokohama, Kanagawa
223-8522, JAPAN
taira@ny.ics.keio.ac.jp

Nobuyuki Yamasaki
School of Science for
Open and Environmental Systems
Keio University
Yokohama, Kanagawa
223-8522, JAPAN
yamasaki@ny.ics.keio.ac.jp

Abstract

This paper describes the design and implementation of functionally distributed control architecture for real-time control of robot systems. Robot systems must control several functions in parallel for decision, action, and recognition. In order to fulfill this requirement, we propose the architecture based on parallel/distributed control. The robot system based on this architecture is designed as a parallel/distributed computer, in which each module has different I/O peripherals and executes functionally distributed and coarse-grained tasks in real-time, cooperating with each other. We also evaluate the efficiency of the proposed architecture through the experience in implementing a prototype robot.

1 Introduction

The need for robots that assist humans has been recognized. Several researchers are endeavoring to realize such robots. Such robots should have both decision, action, and recognition. Thereby a significant number of interrelated functions is required. In order to operate a robot system in human society, it is necessary to achieve both local control and global control in real-time[5]. The words *local control* and *global control* used herein are defined as follows:

- *Local control* : the policy that controls low level I/O peripherals of each function distributed in a robot system
- *Global control* : the policy that controls whole system

These two words are used for behavior description architecture level in this paper. In order to achieve these control, a robot system requires both appropriate computer architecture and behavior description architecture.

Typical robot systems [1][2][4] have adopted load balancing control by multiprocessors. However, because many sensors and actuators are spread widely in a robot system and the robot system needs huge computational power, it is difficult to realize real-time performance.

The operations of robots are basically processed in parallel such as moving while sensing, etc. I/O peripherals are unevenly distributed. Therefore it would be more efficient to process in parallel than in serial. Here, we propose a functionally distributed control architecture based on the concept of parallel/distributed processing, taking both hardware and software into consideration. The robot system based on this architecture consists of several functional modules with exclusive processors and can perform global action by cooperation of such modules. We also design and implement a prototype robot for evaluating the efficiency of the architecture.

This paper is organized as follows; In section.2, the design of the functionally distributed control architecture is presented. Section.3 describes the implementation of a prototype robot based on our proposed architecture. The experimental results are shown in section.4. Finally, the paper is concluded in section.5.

2 Design

2.1 Architecture Overview

We design the architecture for real-time control of robot systems called the functionally distributed control architecture as shown in Fig.1.

The most different point compared with conven-

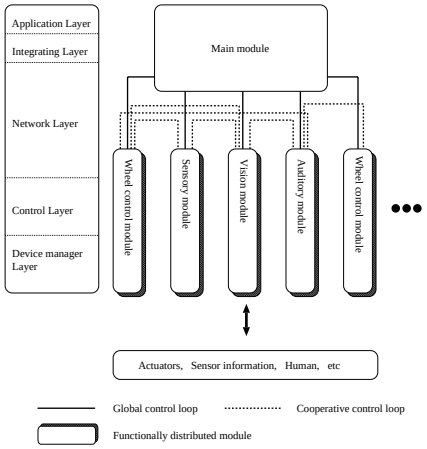

Figure 1: Architecture overview

tional architecture is that this proposed architecture is based on parallel/distributed control. A parallel/distributed control system processes system-wide tasks by cooperation several functionally distributed modules which have exclusive PU(Processing Unit) and process local tasks in real-time. Thanks to this approach, robot system can realize huge computational power and real-time performance.

Our proposed robot system is designed as a parallel/distributed computer, in which each module has different I/O peripherals and executes functionally distributed coarse-grained tasks in real-time, cooperating with each other.

2.2 Functional Classification

In order to achieve real-time control of each function, for example image processing, speech recognition, and wheel control, a robot system is divided into functionally distributed modules with their own exclusive PU. Each module controls a function of the robot system locally in real-time and notifies information to the other modules. Division policy is based on parts of human body, such as auditory, vision, arm, and etc., in consideration of real-time performance and grain size of elemental technologies.

For achieving global control, a module to plan the behavior of the whole system is necessary. Therefore, a main module is prepared as one of the functionally distributed control module. The main module plans and schedules the whole system.

Meanwhile, a method to balance the load of this main module and to operate in parallel with global control is indispensable. Thus cooperative control as shown in Fig.1 is used. The cooperative control refers to a control that modules beside main module cooperate with each other by using an intermediate network. In consequence, the system can reduce load of the main module and improve parallelism and robustness.

2.3 Hierarchical Layer

This proposed architecture is not only functionally-classified but also hierarchically-classified parallel/distributed architecture for effective development of a robot system. The functionally distributed control architecture consists of five layers. Detail of each layer is described below.

Device manager Layer

I/O peripherals such as motors, sensors, cameras etc. are directly controlled in real-time. Raw sensory data such as oriented data obtained by gyros and encoders, image pixel data obtained by cameras and sound signals obtained by microphones are processed.

Control Layer

Each module achieves high-level elemental technologies using the filtered intermediate level data computed from row data. As the result of processing, high-level identifiable data like human face, natural language, and navigation information are acquired. In order to receive the tasks from other modules, each module has an event handler.

Network Layer

The high-level identifiable data generated by any module must be offered to main module and/or other modules as events, and task must be sent to the suitable modules. A Network Layer hides communication interface among modules. Then, interrupts are used to communicate every event that occurs among modules, regardless of whether it is an exceptional processing event or a normal processing event. In short, the robot system is built as an event driven system. Thanks to this layer, each module can carry out data transfer to suitable modules by only writing data, source address, and destination address in its own message buffer.

Integrating Layer

Integrating Layer consists of three parts: a global planner, a map manager, and a database system. A global planner achieves global control as follows.

Table 1: Specification of processors

	Responsive Processor	General-purpose Processor
CPU	SPARC 100MHz	Celeron 1.3GHz
Memory	16MByte	256M Byte
Interface	Responsive Link	IEEE802.11b
OS	RT-Frontier[3]	Linux

1. Events occur at any functionally distributed module and these module interrupts the main module.
2. Global planner receives events and updates correspond data of the database system that manages the states of all modules.
3. By using the database system and a map created by a map manager for navigation, this planner carry out batch processing of these events and generate global actions.
4. The planner requests tasks to the relevant modules and these modules perform according to the task.

Application Layer

Users can define a system-wide task by using abstract command and situation provided by Integrating Layer without being conscious of device control and the functions.

3 Prototype Robot

In order to evaluate the efficiency of the functionally distributed control architecture, wheel robot P1 is implemented as shown in Fig.2. Modules are functionally divided into a main, a wheel, a vision, an auditory, a voice, a sensory, and a remote control module. The main, the wheel, and the sensory module are implemented by using the Responsive Processor[6], and other modules are implemented by using the general-purpose processor. Table.1 shows each specification of processors. Responsive Link[6] and PCI bus are used to connect modules as shown in Fig.3. Fig.4 shows various functions and software of P1.

4 Experiment and Result

4.1 Typical Operation of P1

Fig.5 shows typical operation of P1. The main module requests the wheel module to navigate P1 according to the route information, which the main module plans. Simultaneously, the main module requests other modules and they performs their own task. The sensory module interrupts the main module, when a human is sensed in front of the robot. The vision module also detects the human and interrupts the main module. The main module which received interruptions begins to re-plan, and requests tasks to the wheel module, the auditory module, and the voice module. The robot stops in front of the human and the main module begins to interact with the human by cooperation of the auditory module and the voice module. Although each module processes different tasks locally, one system-wide task is achieved as a robot system by global control of the main module.

Figure 2: Prototype robots

Figure 3: P1 computational structure

4.2 Evaluation of P1

Table.2 shows periodic task and processing time of local control at each distributed modules in this implementation. P1 achieved several functionally-classified tasks in real-time and in parallel by the functional distribution.

Then, processing time until the wheel module carries out the emergency stop was measured 10 times as shown in Table.3. As a delay of processing time can be bounded, time needed for stop processing can be

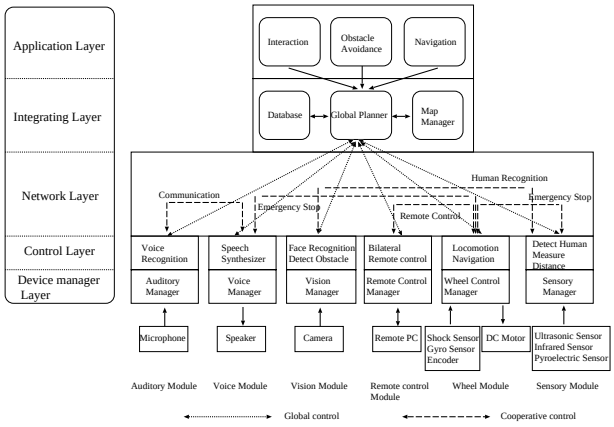

Figure 4: Function and software of P1

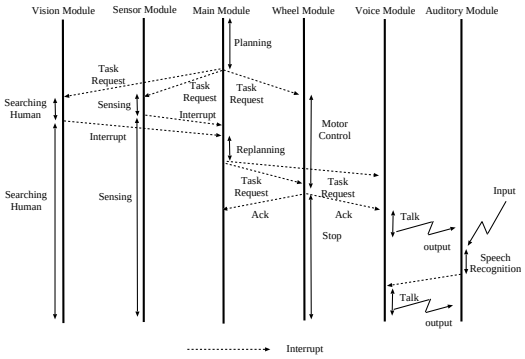

Figure 5: Typical operation of P1

predicted. So, stable and reliable control methods are achieved by the hierarchical layer.

5 Concluding Remarks

In this paper, we designed and implemented functionally distributed control architecture based on parallel/distributed control, which can realize real-time control of robot systems. The robot system based on the architecture can process every functionally distributed module at its exclusive processor in parallel. By evaluating, the robot system achieved local control of each function and stable global control in real-time, so the validity of this architecture was able to be shown.

Acknowledgments

This study was performed through Special Coordination Funds of the Ministry of Education, Culture, Sports, Science and Technology of the Japanese Government.

Table 2: Periodic task and processing time

	Periodic Task (ms)	Processing Time (ms)
Motor control	1.0	0.8
Sensor processing	2.0	1.8
Image processing	200	180
Speech recognition	—	10.0
Language analysis	—	1000.0
Speech synthesis	—	5.0

Table 3: Processing time of several control methods

	Global control	Cooperative control	Local control
Average time(μs)	3700	230	0.96
Standard variation	1.3	0.20	0.003

References

[1] Rodney A. Brooks, Cynthia Breazeal, Matthew Marjanovi, Brian Scassellati, and Matthew M. Williamson. The Cog Project: Building a Humanoid Robot. *Computation for Metaphors, Analogy and Agents*, Vol. 1562, , 1998.

[2] Tatsuzo Ishida, Yoshihiro Kuroki, Jinichi Yamaguchi, Msahiro fujita, and Toshi Doi. Motion Entertaiment by a Small Humanoid Robot Based on OPEN-R. In *IEEE/RSJ Intl. Conference on intelligent Robots and Systems*, 2001.

[3] Hidenori Kobayashi and Nobuyuki Yamasaki. An Integrated Approach for Implementing Imprecise Computations. *IEICE transactions on Information and Systems*, Vol. E86-D, No. 10, 2003.

[4] Yoshiaki Sakagami, Ryujin Watanabe, Chiaki Aoyama, Shinichi Matsunaga, Nobuo Higaki, and Kikuo Fujimura. The intelligent ASIMO: System overview and integration. In *IEEE/RSJ Intl. Conference on intelligent Robots and Systems*, 2002.

[5] John A. Stankovic. Misconceptions about real-time computing. *IEEE Computer*, pp. 2–10, 1988.

[6] Nobuyuki Yamasaki. Responsive processor for parallel/distributed real-time control. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1238–1244, 2001.