

Paper:

Design and Implementation of the Reconfiguration Mechanism for a Modular Humanoid Robot

Tetsuya Taira, and Nobuyuki Yamasaki

School of Science for Open and Environmental Systems, Keio University

3-14-1 Hiyoshi, Kohoku-ku, Yokohama, Kanagawa 223-8522, Japan

E-mail: {taira, yamasaki}@ny.ics.keio.ac.jp

[Received October 31, 2005; accepted April 17, 2006]

This paper describes the design and implementation of the reconfiguration mechanism for a modular humanoid robot. To aid researchers in their works and enable users to request various tasks, humanoid robots are expected to require such reconfiguration mechanism. A robot with the proposed reconfiguration mechanism potentially consists of several functional modules such as arms, mobile components, and heads, and can be used as some kinds of humanoid robots or as several autonomous functional robots. We evaluated the efficiency of our proposed reconfiguration mechanism through the experiences using reconfigurable modular humanoid robot prototype R1. Experimental results show that the proposed mechanism achieves expandable and flexible reconfiguration for researchers and users by changing the robot configuration to different types of robots for many purposes. We believe that our humanoid robot with the proposed reconfiguration mechanism will enable user-specific humanoid robots more easily than ever before.

Keywords: robot system architecture, humanoid robots, modular robot systems, reconfigurable mechanism, parallel/distributed systems, embedded systems

1. Introduction

Humanoid robots can potentially assist human activities in human daily environments such as offices, homes, and hospitals. We propose an innovative architecture enabling humanoid robots to be reconfigured for different tasks into different types of robots.

Many kinds of elemental functions including bipedal walking and 3D-vision, and integrated functions such as high-level behavior integrating 3D-vision-based perception and motion-planning-based behavior are investigated. Both elemental and integrated functions are important issues to develop a practical humanoid robot which works in a real world together with mankind. The expandability and flexibility for improving the functionality of robots are fundamentally important.

Users must be able to reconfigure their own robots to suit their needs and preferences, the same as they can customize their own personal computers, upgrade devices,

add new functions to them, and troubleshoot.

Our solution to these needs of researchers and users is a reconfigurable modular humanoid robot that is adaptable to new environments and diversified tasks. Such robots would enable researchers to construct customized robots for their research platforms and different companies interested in robotics to build their own products and prototypes based on modular humanoid robot specifications.

The reconfigurable robot architecture we designed and developed [1] differs from conventional humanoid robots architecture by pursuing modularity of mechanical, electronic, and software parts based on elemental functions.

According to a variety of uses, reconfigurable modular humanoid robot R1 [2] is used as some kinds of humanoid robots or wheeled robots, arm robots, and so on. Our three key technologies are 1) modular humanoid robots, 2) functional robots, and 3) reconfiguration mechanism. We focus on the design and implementation of our proposed reconfiguration mechanism in this paper.

Our paper is organized as follows. Section 2 describes our reconfigurable modular humanoid robot R1 including requirements of humanoid robots and related works. Section 3 introduces the design concept of our reconfiguration mechanism. Section 4 details our proposed reconfiguration mechanism. Section 5 discusses basic experiments and evaluation. Finally, section 6 presents conclusions.

2. Reconfigurable Modular Humanoid Robot

2.1. Reconfigurable Modular Humanoid Robot Architecture

In this section, we describe a novel architecture, reconfigurable modular humanoid robot architecture [1]. The development of our new architecture was motivated by the mismatch between ideal and conventional humanoid robots.

The major advantage of this architecture is that users can reconfigure the modular humanoid robot to different types of robots by combining several functional robots as shown in **Fig.1**. The functional robots operate by themselves without being the parts of humanoid robots. For example, the functional robot operates as an arm part of the humanoid robot when integrated into the humanoid



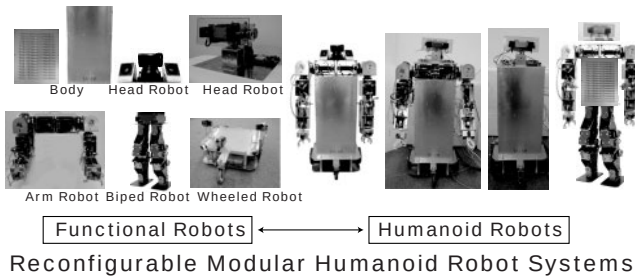


Fig. 1. Proposed reconfigurable modular humanoid robot.

robot, while it operates as an arm robot when separated from the humanoid robot.

Our goal is to design a robot such that researchers and users can use it as some kinds of humanoid robots or autonomous robots. We focus on the concept of a modular system [3, 4] consisting of several simple modules with mechanical and electronic parts. Modular robots operate as one robot and are reconfigured into different types of robots. Open-R [5, 6], an entertainment robot system architecture, provides robots with expandability and flexibility by dividing robots into several modules with hardware and software parts.

Basically, humanoid robot functions operate in parallel as the robot walks, processing the image and sound. Individual modules are thus optimally designed based on elemental functions of a humanoid robot to apply the modular concept to a humanoid robot. Thus, individual module autonomously executes one function of a humanoid robot using its own resources such as processing unit, sensors, and actuators. We design a humanoid robot composed of functional modules with exclusive mechanical, electronic, and software parts required for an elemental function. Individual module is defined as “functional robot” shown in **Fig.2** and its component parts are defined as follows:

- Mechanical components:
 - Mechanical components with expandability and flexibility,
 - Reconfigurable mechanical connectors between functional robots,
- Electronic components:
 - I/O devices such as various sensors and actuators required for an elemental function,
 - Embedded processor with a processing unit, memory, and I/O peripherals,
 - Real-time communication interfaces,
- Software components:
 - Real-time OS running on an embedded processor,
 - Device drivers,
 - Real-time network protocols,
 - Application programs.

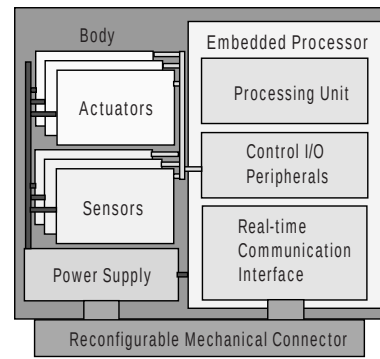


Fig. 2. Functional robot.

Our approach is effective in realizing ideal humanoid robots.

2.2. Related Works

Most research-oriented humanoid robots H7 [7], HRP [8] are classified as centralized systems having one or two high-performance PC/AT computers running a real-time OS such as RT-Linux [9] or ART-Linux [10] inside its torso and all devices are connected to the computer. The centralized approach has the advantage of simple hardware and software design, but is not suitable for humanoid robots having numerous physically distributed sensors and actuators. Signal and electrical cables must be wired from the central device to each I/O devices. All cables must be rewired when adding new devices and functions. The number of devices connected to a robot is limited to the number of processor I/O ports. PC/AT computers require considerable power.

HERMES [11] and QRIO [12] are classified as distributed systems. All actuators of HERMES are designed as modules with compatible mechanical and electrical interfaces. A CAN bus [13] connects them to a main computer. QRIO is designed based on OPEN-R architecture and an OPEN-R bus connects some FPGA, ISA (Intelligent Servo Actuator), and sub-CPU's to a main computer. Whole-body motion is controlled by the main computer mounted inside the torso, and low-level actuator control is processed by each module. Even though the mechanical parts of the distributed system are expandable thanks to the modular design, it is difficult to research elemental functions separately and customize the system because motion planner, image processing, and sound processing controllers are built in in much the same way as centralized systems.

Table 1 compares proposed with conventional humanoid robots. The centralized approach has the advantage of simple hardware and software design. The distributed approach focuses on expandability and robustness more than the centralized approach. However, the centralized and distributed approaches have the problems caused by the physical controller arrangement. Because these approaches have main controllers inside the torso, researching elemental functions separately and adding

Table 1. Humanoid robots comparison.

Approach	Centralized	Distributed	Proposed
Controller	PC/AT computer	Embedded processor and/or PC/AT computer	Embedded processor for distributed real-time control
Power consumption of controllers	Higher (PentiumIII: 25W, PentiumIV: 100W)	Higher (with PC) or Lower (without PC)	Lower (Responsive Processor: 2W)
Physical arrangement of controllers	Central device	Individual and central devices	Individual modules
Expandability	Poor	Fair	Excellent
Weight	Lighter than modular system	Lighter than modular system	Heavier than other systems
Redundant parts	Tiny	Tiny	More than other systems
Robustness	Poor	Good	Good
Control software	Simple	Complex	Complex
Reconfiguration mechanism	Fair	Fair	Excellent

new functions to the existing robot are difficult. That is, the conventional approaches have limited expandability and flexibility. And they have also the problem of controller power consumption.

We solve the problem of expandability and flexibility by introducing reconfigurability. For power consumption, we use an embedded processor as a module controller. Although Pentium III processor consumes about 25.0W to 50.0W, the Responsive Processor requires only about 2.0W. However, the modular humanoid robot has some problems such as how to achieve reconfiguration mechanism, how to obtain whole-body motions, control software complexity, and increased weight caused by redundant parts of body. Once these problems are solved, our proposed approach will be effective in realizing a humanoid robot.

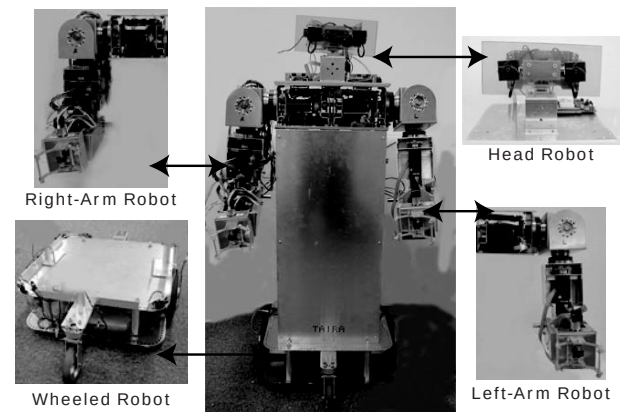
2.3. Modular Humanoid Robot R1

We developed a first-generation reconfigurable modular humanoid robot R1 as shown in **Fig.3** as a testbed for our proposed architecture. With R1, users reconfigure the robot manually while it is not operating. The R1 is positioned as a breakthrough for a practical humanoid robot that is reconfigurable manually or automatically while operating. We thus discuss the offline manual reconfiguration mechanism in this paper.

R1 is about 1000mm high and weighs about 27.0kg. It has two 1-DOF wheels, two 6-DOF arms, and a 6-DOF head. R1 consists of a main module, a head robot, a wheeled robot, a left-arm robot, and a right-arm robot. All functional robots of R1 are modules with exclusive mechanical, electronic, and software parts. Specifications are detailed below. The design policy for mechanical, electronic, and software parts is detailed in Sections 3 and 4.

1) *Main Module*: The main module plans and schedules the whole-body motion based on the applications.

2) *Head Robot*: The head robot consists of the communication and vision systems to interact with users. The communication system has two microphones and two

**Fig. 3.** Reconfigurable modular humanoid robot: R1.

speakers, and processes voice recognition, speech synthesis, and sound localization. The vision system has two CCD cameras and 6-DOF, and processes face/object detection, face/object/3D-gesture recognition, and 3D measurement.

3) *Wheeled Robot*: The wheeled robot has 2-DOF, a fiber gyro, infrared sensors, and ultrasonic sensors. This robot achieves accurate movement, map generation, and self-localization using sensor data fusion based on a Kalman filter, and executes an emergency stops using its own sensors.

4) *Left-Arm Robot*: The left-arm robot has 6-DOF, joint power (torque, speed), joint movement range for whole-body motion and tactile sensors. This robot calculates trajectories to the target and applies PD control to destination angles autonomously. This robot grasps and manipulates objects by integrating planning with its own tactile sensing data and visual data provided by the head robot. This robot also generates collision-free paths by communicating with the right-arm robot.

5) *Right-Arm Robot*: The right-arm robot performs the same tasks as the left-arm robot.

Table 2. Reconfigurable robots.

Robot	Type	Module composition	Online or Offline	Manual or Automatic	Control
M-TRAN [14]	Mobile robot	Homogeneous	Online	Automatic	Centralized
Cebot [3]	Mobile robot	Heterogeneous	Online	Automatic	Distributed
AIBO [5]	Four legs robot	Homogeneous	Offline	Manual	Centralized
HERMES [11]	Upper-body humanoid robot	Homogeneous	Offline	Manual	Centralized
R1	Upper-body humanoid robot	Heterogeneous	Offline	Manual	Distributed
Next-generation	Whole-body humanoid robot	Heterogeneous	Online	Automatic and Manual	Distributed

3. Design Policy of Reconfiguration Mechanism

This section details the offline manual reconfiguration design policy.

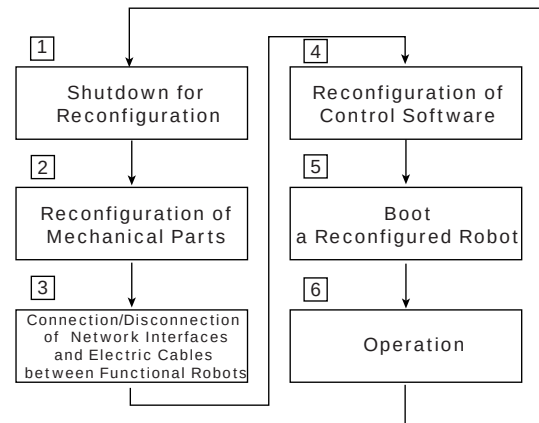
Table 2 shows that comparisons of reconfiguration mechanisms. It should be noted that we apply modularity to a humanoid robot, and each module operates as a functional robot or as a part of the humanoid robot. Connection methods between modules are manual or automatic. Obviously an automatic inter-module connection is required for the self-reconfiguration. M-TRAN [14] enables the online automatic connection and disconnection by reconfiguration mechanism using permanent magnets and springs. AIBO [5] and HERMES [11] support offline manual reconfiguration mechanism combining several homogeneous modules.

Problems these reconfigurable robots have include 1) mechanical restrictions on module development and 2) the inability of individual modules to operate autonomously because reconfigurable robots are controlled by a central computer.

Although R1 is reconfigurable manually, it does not restrict module development because it conducts reconfiguration by combining heterogeneous modules. It is possible for each module to operate autonomously by introduction of the functionally distributed control.

Figure 4 shows the block diagram of our proposed reconfiguration mechanism. First of all, humanoid robot must be shut down whenever users reconfigure them. Second, users customize the robot's mechanical configuration manually by combining functional robots, which requires user-friendly mechanical interfaces to make reconfigurable humanoid robots available to all users. Third, electrical cables and network interfaces are connected or disconnected to operate. It is necessary to consider avoiding disconnection that may cause a system to crash. The cable connection must be as easy as possible. Next, users change software parameters based on information on the reconfigured robot, which requires a flexible software model to adapt to a variety of reconfigurations. A real-time network is also needed to guarantee communication latency for any reconfiguration. The reconfigured robot is then booted and operates as a user-specific robot.

Reconfigurable humanoid robots can use user-specific mechanical, electronic, and software components with-

**Fig. 4.** Offline manual reconfiguration mechanism.

out using ready-made components, which requires an approach involving all robot elements, including mechanical, electronic, and software design. The following sections detail the design and implementation of our reconfiguration mechanism.

4. Design and Implementation of the Reconfiguration Mechanism

4.1. Mechanical Parts with Reconfigurable Mechanical Connectors

Each functional robot requires a mechanical connector that supports connection freely and easily for reconfiguration mechanism.

There are some requirements for the connectors. Connectors must be simple mechanical structure to make the humanoid robot available to all users. The connector should have network interfaces to communicate among functional robots. On the one hand, researchers may need to change network interfaces and topology for development. On the other hand, users may want to use a simple interface to connect functional robots. We designed a *communication interface port* as shown in **Fig.5** having four Responsive Link ports (or Gbit Ethernet ports), two IEEE1394 ports, two USB 2.0 ports, and one patch cable port. We chose these interfaces as explained below. The



Fig. 5. Communication interface port.

patch cable ports are connected to those of other communication interface ports by a simple patch cable made of a 68-pin twisted pair cable shown in Fig.5 when the functional robot connects with other robots to reconfigure the humanoid robot. The power supply is also connected by the patch cable.

A reconfigurable modular humanoid robot may have a serious problem of redundant parts increasing the robot weight over that of conventional humanoid robots. The redundant parts are mainly mechanical connectors and functional robot controllers. Too many redundant parts may make it difficult to walk by its foot. Since it is difficult to reduce the number of redundant parts for reconfiguration mechanism, functional robots should be designed specifically to minimize robot weight.

We developed the prototype reconfigurable mechanical connectors in Fig.6 using CAD and machine tools.

The connector A links the top of each functional robot and the connector B links the bottom of each functional robot. The prototype connector made of aluminum weighs about 0.5kg. R1 has six connectors, for a total connector weight of 3.0kg. The prototype modular humanoid robot including these connectors weighs about 30.0kg, meaning that connector weight accounts for less than 10% of the weight.

Figure 7 shows how to reconfigure the mechanical parts. First of all, the reconfigurable mechanical connectors A and B are mounted on functional robots. Next, the connector A of the functional robot is connected to the connector B of the other, and then these connectors are joined manually by two screws. Finally, the communication interface and power supply line are connected using the patch cable.

Measurement showed that the total weight of redundant parts was about 3.0kg. The proposed robot thus obtains the capability of changing its configuration to different types of robots for a tradeoff weight up to about 3.0kg, which might be a problem for existing humanoid robots but it would be reduced the weight by the development of lighter connectors made of FRP (Fiber-Reinforced Plastic) that is lighter and stronger than aluminum.

We think that humanoid robots must implement reconfiguration mechanism to realize expandability and flexibility.

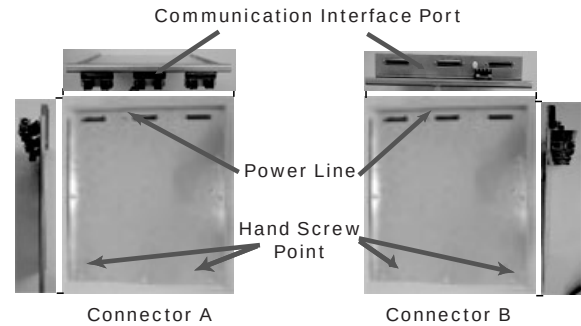


Fig. 6. Reconfigurable mechanical connector.

4.2. Electronic Parts

Various types of processors including embedded processors, general-purpose processors, and workstations are candidates for functional robot controllers. We use an embedded processor with a processing unit, memory, and I/O control peripherals such as SH [15], PowerPC, and Responsive Processor [16] because modular systems must be lightweight, compact, and energy-saving.

The network interface between functional robots requires two indispensable factors, 1) hard real-time communication and 2) nonrestrictive topology. It's strongly recommended that network interfaces support hard real-time communication because functional robots must cooperate to obtain whole-body motions. Network interfaces must also have few or no restrictions on network topology to avoid restrictions on reconfigurability.

Many robots use such high-speed communication interfaces as Gbit Ethernet, USB 2.0 [17], and IEEE1394 [18] as the control network. These interfaces do not support hard real-time communication, however, because their designs are aimed at computer or inter-I/O communications. The lack of hard real-time communication makes it difficult to realize whole-body motions. Additionally, these interfaces have limited network topology, in other words, the system which uses these interface has limited expandability. The networks using USB support only up to 127 peripherals and a tree structure with six levels, and require centralized control. The IEEE1394 has no such restrictions but does not allow loops, restricting flexibility. System resets may also cause problems in reconfigurations, especially in IEEE1394 reconfiguration, where each device ID is cleared and reassigned. Robot system resets are risky because the system must always communicate between modules to conduct tasks such as control.

From a hard real-time communication standpoint, real-time communication interfaces such as the CAN bus, FlexRay [19], and Responsive Link [20] are potential candidates. Considering network topology, the bus connection lacks the expandability of the processor compared to link connection because bus control becomes more complex as the number of processors increases.

We use *Responsive Processor* as an embedded processor and *Responsive Link* as the communication interface to realize our proposed reconfiguration mechanism for

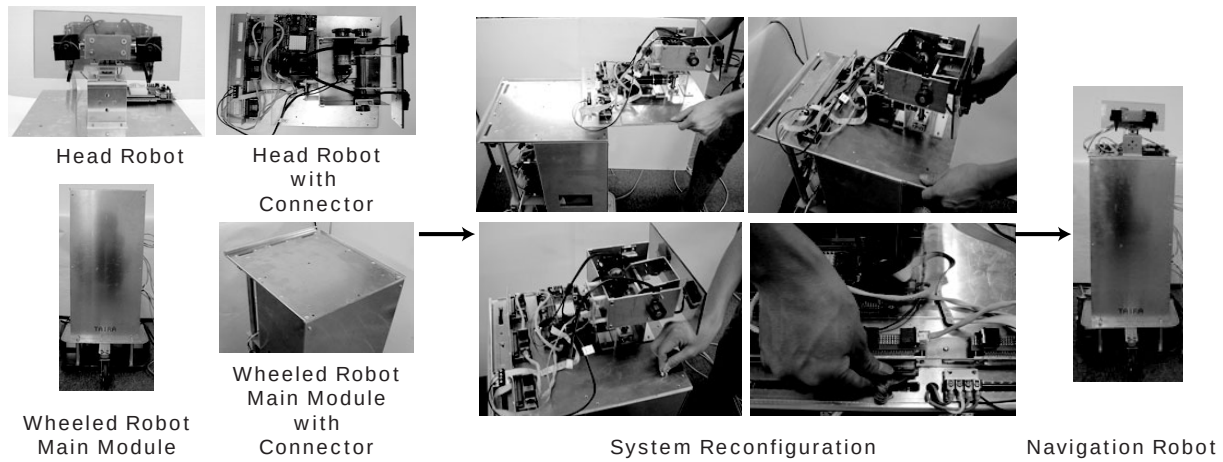


Fig. 7. Mechanical reconfiguration.

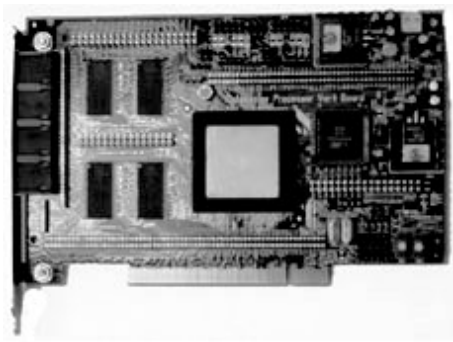


Fig. 8. Responsive Processor and Responsive Link.

three reasons: First, the processor provides Responsive Link. Second, power consumption is low because the processor uses only 2.0W. Third, the processor is lightweight at about 0.15kg. The prototype R1 has five controllers, compared to the conventional humanoid robot that has one or two PC/AT computers with its I/O boards which weigh 0.5kg to 2.0kg. Controllers in prototype modular humanoid robots may thus weigh less than controllers in conventional humanoid robots. Responsive Link realizes real-time communication and a topology-free network.

Responsive Processor and Responsive Link [16] are summarized below. The PCI card of Responsive Processor is shown in Fig.8.

1) *Responsive Processor*: This embedded processor integrates functions into an ASIC chip, such as a RISC MPU core (SPARC), memory, and many control I/O peripherals including 50MHz PWM generators (9ch), 24bit pulse counters (9ch), 10bit A/D converters (8ch), 8bit D/A converters (2ch), SIO (2ch), PIO (16bit digital port), and Responsive Link (5ch).

2) *Responsive Link*: Responsive Processor has five Responsive Links. One Responsive Link consists of one Event Link and one Data Link, whose packet size is 16 bytes and 64 bytes respectively. Each link consists of

a point-to-point, full-duplex, and serial communication link. The link connection is changed flexibly. One Responsive Link is connected directly to the RISC MPU core, and the other four sets are arranged on the chip. A packet has two priority bits in its header field. This is used for two purposes in the network switch built-in Responsive Processor. One is the arbitration and the other is packet routing. The priority may be changed when the packet crosses the switch, depending on the routing table. We use these advantages to rank messages.

The communication interface ports have other interfaces, including Gbit Ethernet, USB 2.0, and IEEE1394, to apply conventional robots or parts to a modular humanoid robot.

4.3. Software Parts

Software parts of the reconfigurable modular humanoid robot are designed based on the following concepts:

1. Module information,
2. Reconfiguration support script,
3. Motion planning,
4. Transparent software architecture,
5. Communication mechanism for a reconfigurable modular humanoid robot.

These are explained below.

1) *Module Information*: Control information of functional robots is required to reconfigure the modular humanoid robot and obtain real whole-body motion by combining functional robots. Our solution is that each functional robot has the following *module information* that it shares with other functional robots. The module information is as follows:

1. Information on identifications of functional robots: type ID (module, e.g., mobile robot, arm robot), function ID (specifications, e.g., 6-DOF arm, 2-DOF arm).

2. Information on physical quantities of functional robots:
shape, weight, center of gravity, mobility range, power consumption, physical arrangement of I/Os.
3. Information on functions and capabilities of functional robots:
function (e.g., navigation, grasping), shared data (kind of data, update frequency, data sharing port ID).

Information for 1) identifies each functional robot. That for 2) shows physical quantities required for control. That for 3) shows functions and control data, including update frequency, and data sharing port ID. Functional robots acquire control data on other robots such as sensor information at update frequency by accessing their port ID.

2) *Reconfiguration Support Script*: After mechanical reconfiguration as shown in **Fig.7**, module information must be shared with all functional robots of a reconfigured humanoid robot. The modification of functions and control parameters caused by system reconfiguration also must be reflected in modular humanoid robot software. We designed *reconfiguration support script* that reads all module information and then replaces old parameters with new ones, adds new functions, or deletes old functions automatically. Reconfiguration support script judges whether the reconfigured robot has been used up to now or not, and handles the system initialization program, including control parameters and network routing tables. The script saves information on all reconfigured robots. If the robot configuration has been used, users can operate the reconfigured robot without having to write new programs. Control software programs are loaded automatically based on the reconfigured robot by reconfiguration support script.

3) *Motion Planning*: Modular humanoid robot control software is more complex than that of the centralized approach because the modular humanoid robot must obtain whole-body motions by combining functional robots. The modular humanoid robot requires a functional robot to plan the whole-body motions. Arbitration is also needed if a conflict occurs among modules, so the main module is prepared as one of functional robots. Each functional robot plans trajectory generation, the navigation path, and so on, based on the task requested by the main module. The main module does not use resources such as sensors and actuators of functional robots directly but functions provided by functional robots during whole-body motions. That is, the main module is the brain robot that plans and schedules whole-body motions, arbitrates module conflicts, and maintains system consistency. The main module is developed as one robot like other functional robots. It is necessary for users to write the programs that control the whole-body motions on the main module if the robot configuration is new and never used. Both researchers and users can use humanoid robots with different brains by changing the main module of the humanoid robot. So, our proposed modular humanoid robot is classified as a functionally hierarchical distributed system.

Our modular system operates without the main module through cooperative control of two or more functional robots, for example, manipulation, visual feedback control, and emergency response. The main module is a functional robot that solves the problems of control software complexity. If the system has no main module, adding new functional robots to the existing system, development of whole-body motions, and system reconfiguration may require that software configurations be modified, requiring developers to rewrite programs for each functional robot. If the system has a main module, they may require that only main module software be modified. The main module separates research on integrated functions from research on elemental functions. The main module enables expandable robot system with simple software architecture.

4) *Transparent Layered Software Model*: Functional robots should have standardized software interfaces for reconfiguration mechanism and cooperative control between functional robots. Our proposed architecture defines a five-layer model shown in **Fig.9** as a control software framework for reconfigurable modular humanoid robots to realize the following factors:

- Real-time processing of both whole-body motions and elemental functions,
- Motion planning for whole-body motions,
- Cooperative control between functional robots,
- Network framework for communication between functional robots.

This layered model consists of *function control*, *functional module control*, *network*, *integration*, and *application layers*. Each layer is clearly and simply explained.

Application Layer: The application layer enables researchers and users to define system-wide tasks such as interaction with users and target grasping.

Integration Layer: The integration layer, for researchers on humanoid robots, consists of *whole motion planner*, *task manager*, and *status checker*. The whole motion planner transforms system-wide tasks into sub-tasks that are transmitted to functional robots by the task manager. The status checker compensates for whole-body status by checking messages from functional robots at 1msec cycle and creates the *status table* based on data for the whole-motion planner and task manager.

Network Layer: The network layer supports real-time data sharing between functional robots. A *global network* is defined as a network among the main module and functional robots for system-wide cooperation. The *global network agent* transfers tasks requested by the task manager to suitable functional robots. The *local network* is defined as a network among functional robots for cooperative control such as reactive control between head and wheeled robot and visual feedback control between head and arm robots. When cooperative control is required, the

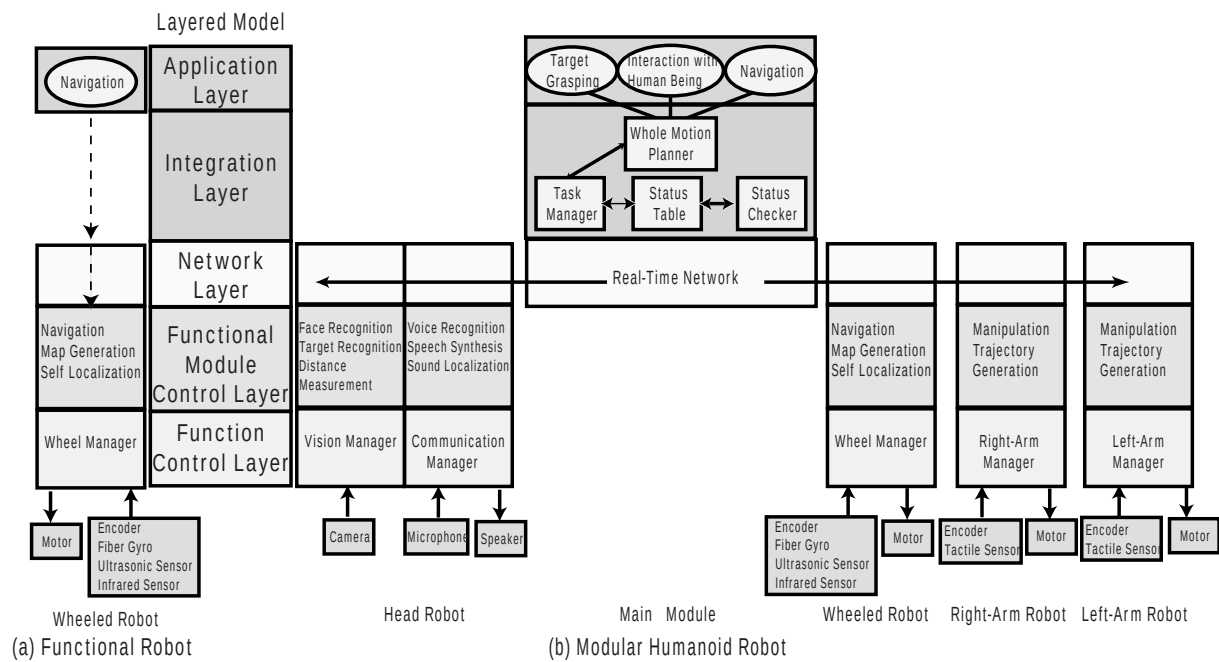


Fig. 9. Transparent layered software model.

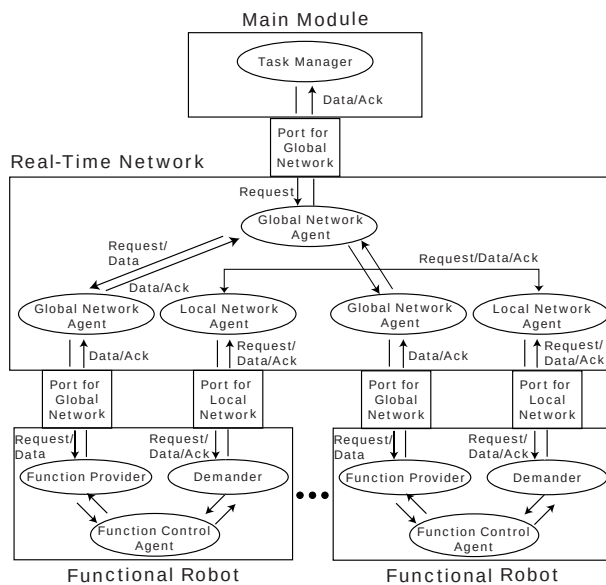


Fig. 10. Communication mechanism for reconfigurable modular humanoid robot.

local network agent accesses other functional robots to meet requests from the *demand*.

Figure 10 outlines our communication network.

Functional Module Control Layer: The functional module control layer is for researchers on functional robots. This layer consists of *function control agent*,

function provider, and *demand*. Each function control agent executes tasks based on task requests from upper layers and achieves high-level functions. The functional provider supplies high-level identifiable data or functions such as face/object recognition, speech recognition, and target grasping to the upper layer or other functional robots. The demander requests the local network agent to collect control data on other functional robots via local network if needed.

Function Module Layer: In the function module layer, I/O peripherals such as actuators, sensors, and cameras are directly controlled in real-time. This layer provides the upper layer with filtered intermediate data such as velocity and optical flow.

5) **Communication Mechanism for Reconfigurable Modular Humanoid Robot:** Communication mechanism for a reconfigurable modular humanoid robot realizes following factors:

- Real-time communication that guarantees communication latency,
- Communication protocols,
- Networking abstraction.

We applied the Responsive Link based real-time network to real-time communication among processors. The event and data packet transmission are separated, so the Event and Data Link are routed independently. Each routing table is created based on the type ID and data sharing

Table 3. Priority and communication line of communication data.

Communication data	Line	Priority
Emergency task	Event Link	Highest
Global synchronization	Event Link	Second highest
Flow control	Event Link	Second lowest
Heart beat	Event Link	Lowest
Task requests from main module to functional robots	Data Link	Highest
Data sharing between main module and functional robots	Data Link	Second highest
Task requests between functional robots	Data Link	Second lowest
Data sharing between functional robots	Data Link	Lowest

port ID of module information. The message communication protocol is simple. The functional robot gets new data from functional robots registered in the routing table by reading the data buffer associated with the port ID, and it transmits data to other modules registered in the routing table by writing to the data buffer associated with port ID. We use two communication paradigms called *event message* and *state message* to communicate between data buffers. The event message is designed on each link for synchronization between the sender and the receiver. The state message is designed on each link for asynchronous processes between the sender and the receiver.

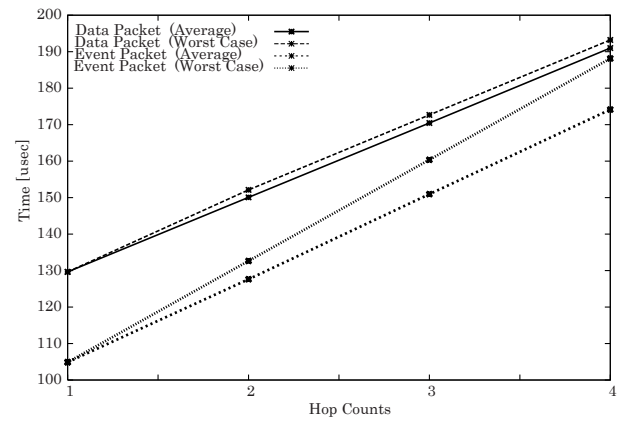
There are four types of communication data in the network.

- 1) Task requests,
- 2) Data sharing,
- 3) Emergency tasks,
- 4) Synchronization including flow control, heart beat, and global synchronization.

The first communication data is for task requests from the main module to other functional robots or between functional robots. The volume of data is large, but the latency requirement is high. To manage whole-body motions, time limitations of task requests from the main module are severe compared to task requests between functional robots.

The second communication data is data sharing for the whole-body planning and the cooperative control. The volume of data is large, but the latency requirement is high. Throughput is more important than latency because it is necessary to simultaneously transmit much more data such as control parameters and sensor data.

The third communication data is for emergency tasks caused by obstacle avoidance or network down time. The volume of data is small, and the latency requirement is the highest among the four types of communication data, because these accidents significantly and negatively influence the system. Latency is more important than throughput.

**Fig. 11.** Transmission latency over Responsive Link.

The last communication data is for synchronization including heart beat, flow control, and global synchronization. The volume of this data is small, but the latency requirement is high to stabilize whole system. Time limitations are not severe with the order of heart beat, flow control, and global synchronization since the impact given on the system is small.

Individual data is assigned to the priority and communication line as shown in **Table 3** based on their characteristics.

Figure 11 shows worst-case transmission latency including software overhead of our communication mechanism. The horizontal axis represents hop counts and the vertical axis represents the latency. Latency is calculated by the following equations, where H is hop count, E is Event Link, and D is Data Link:

$$E = 64.1 + 27.8 * H (\mu\text{sec}) \quad (1)$$

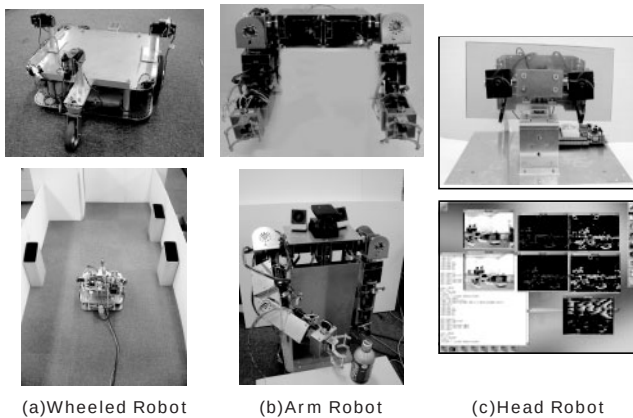
$$D = 107.2 + 22.5 * H (\mu\text{sec}) \quad (2)$$

Worst-case transmission delays over event and data packets are bounded by Eqs.(1) and (2). Because R1 creates the distributed control loop between functional robots using these equations, R1 realizes stable data sharing even if the robot is reconfigured by any combination of functional robots.

6) *RT-Frontier*: We used *RT-Frontier* [21] as the real-time OS controlling Responsive Processor and Responsive Link. *RT-Frontier* features 1) mostly written in C programming language, 2) deadline resolution in 1msec, 3) execution resolution in 100 μ sec, and 4) support of two communication paradigms, i.e., state message and event message. Software including control software for individual robots, network software, and application software is implemented on *RT-Frontier*. **Table 4** evaluates control software for R1. Real-time control of integrated and elemental functions and real-time data sharing are realized.

Table 4. Control and communication cycles.

Task	Cycle (msec)
Status checker (main module)	1.0
Task manager (main module)	10.0
Whole-body planner (main module)	100.0
Actuator control (head, wheeled, arm robot)	1.0
Task manager (head, wheeled, arm robot)	10.0
Motion planner (head, wheeled, arm robot)	100.0
Self-localization (wheeled, arm robot)	10.0
Sound localization (head robot)	100.0
Vision processing (head robot)	33.0
Data sharing between left and right-arm robot	10.0
Data sharing between head robot and each arm robot	10.0
Data sharing between head robot and wheeled robot	10.0
Data sharing between main module and functional robots	10.0

**Fig. 12.** Research on functional robots: (a) map generation, (b) manipulation, and (c) 3D vision.

5. Evaluation

5.1. Experiments

1) *Functional robot*: **Fig.12** shows typical research and experiments using stand-alone functional robots.

Researchers design functional robots and users use them without considering they are a part of a humanoid robot. **Fig.12(a)** shows the wheeled robot generates an environmental map by using several infrared sensors for navigation. **Fig.12(b)** shows the experiments of a dual-arm control algorithm for collision-free manipulation. **Fig.12(c)** shows the head robot generates a 3D vision depth map for environment recognition. **Fig.12** shows individual modules that configure our modular humanoid robot operate as functional robots or as a part of the humanoid robot.

2) *Modular Humanoid Robot*: We show three typical modular humanoid robot operations. The first example is handshaking (**Fig.13(a)**). Recognizing a person's location

by vision processing, the robot turns toward the person, and says hello during a handshake.

The second example is an operation of handing a bottle from a person to a robot as shown in **Fig.13(b)**. Since the robot is not yet skilled enough to recognize the current location of the bottle, it works with the person, who directs the robot to position its arm so that the person can easily hand it the bottle. The robot waits for a touch event that can be recognized by its proprioceptive sensors. The touch event is detected as an involuntary unexpected movement of joints caused by external forces or signals of tactile sensors. After the robot takes the bottle successfully, it put down its hand.

Figure 13(c) shows automatic robot control. The modular humanoid robot measures the distance to a known object 3-dimensionally. The robot plans whole-body motions, then moves to where it can take the object, and grasps it.

The evidence that the modular humanoid robot can obtain the real whole-body motion by combining functional robots is shown in **Fig.14** that describes the processing flow of the operation shown in **Fig.13(b)**. The figure shows how the robot is controlled by combining several functional robots. The main module plans whole-body motion and requests functional robots to process tasks. When the head robot recognizes a person, the event is transmitted to the main module. After the event, the main module issues tasks to a wheeled robot to stop. After the stop, the main module requests the head and arm robots to interact with the person to hand over the bottle.

Functional robots communicate with other robots periodically in parallel with the application task to realize synchronization between functional robots, visual feedback control, and collision-free motions. Though elemental functions such as trajectory calculation, face/object recognition, speech recognition, and navigation are processed asynchronously by each functional robot, integrated functions such as system-wide tasks are achieved by whole-body motion planning, scheduling, and cooperative control.

5.2. Results and Discussions

The experimental results show that our modular humanoid robot solves problems of expandability and flexibility better than conventional humanoid robots by our proposed reconfiguration mechanism.

Figures 12 and 13 show the advantage of the reconfigurable modular humanoid robot. R1 which consists of several functional robots are used for researches on the whole-body planning as shown in **Fig.13**. To develop and research elemental functions, the humanoid robot is separated into several functional robots, e.g., arm robots, wheeled robots, and head robots as shown in **Fig.12**. The prototype modular humanoid robot accomplishes various tasks by reconfiguration mechanism. The modular humanoid robots enable researchers to construct customized robots for their research platforms.

Applying reconfigurability to robots, reconfigurable robots can achieve any task in any situation by selecting

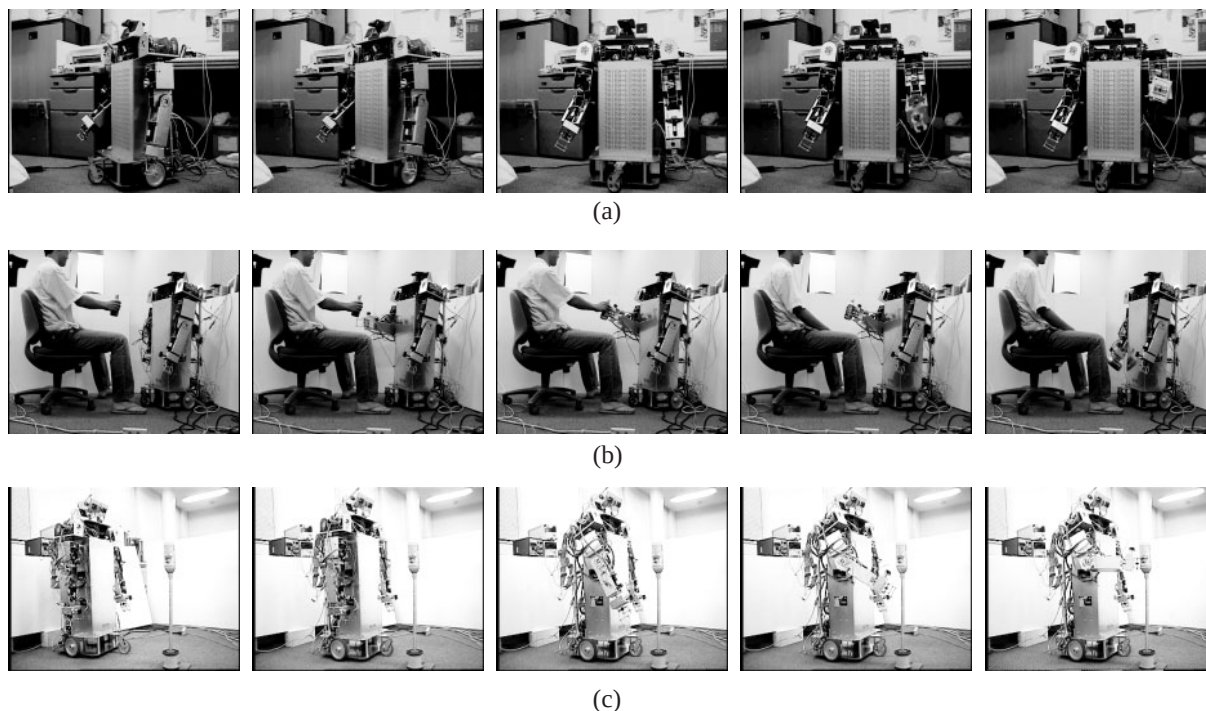


Fig. 13. Research on modular humanoid robots: (a) handshaking, (b) interaction with a person, and (c) automatic control.

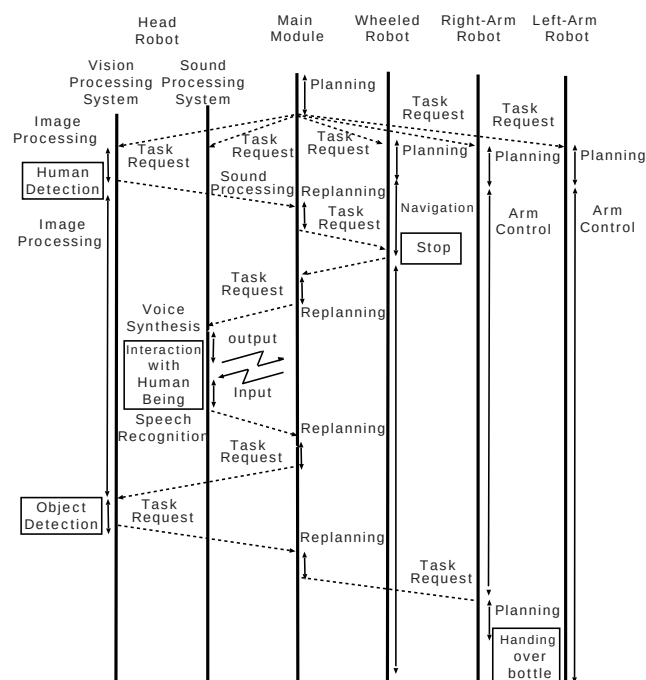


Fig. 14. Processing flow of operation shown in Fig.13(b).

the most suitable functional robots, e.g., two head robots, one with one eye and the other with two eyes. **Table 5** lists head robot specifications.

The type A head robot with one eye shown in **Figs.13(a), (b)**, and **15(a)** has the advantage of lower calculation cost and lower power consumption than the type

Table 5. Comparison of head robots.

Type	A: Fig.13(a), (b)	B: Fig.12(c), Fig.13(c)
DOFs	2	6
Sensors	CCD camera×1	CCD camera×2
Dimension	2D	3D
Applications	Simple tasks	Complex tasks

B head robot with two eyes shown in **Figs.12(c), 13(c)**, and **15(b)**. The type A head robot recognizes environments at high speed and quality and operates for a long time, but because the robot does not qualify for 3D measurement, we expect it to do simple tasks such as face and color recognition. **Fig.15(a)** shows that it is difficult for the humanoid robot with the type A head robot to reach for a bottle on the hand of the man accurately because it cannot recognize the exact location of the bottle.

The type B head robot was developed to improve environmental recognition. In practical 3D measurement, we expect the robot to be suitable for navigation, automatic control, etc. **Fig.15(b)** shows that the humanoid robot with the type B head robot reaches for a bottle on the hand of the man accurately by generating trajectories. This, however, requires more computing resources, calculation time, and power consumption.

Replacing the type A head with the type B head does not require that mechanical, electronic, and software parts of the system other than the head be modified. Users of our proposed robots use the appropriate functions for the appropriate situation. This enables researchers to consider what modules can be used in different situations.

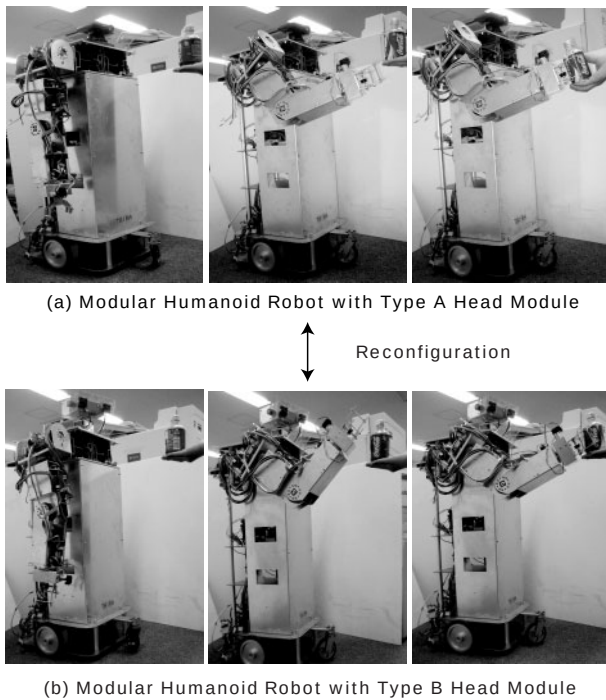


Fig. 15. Advantage of proposed reconfiguration mechanism.

Considering also about mobile modules like head robots, though robots cannot traverse stairs using the wheels, the wheels are expected to enable robots to move rapidly on gravel path and on flat terrain. In contrast, robots with two legs traverse bumps but have limited speed. Alternatives must thus be considered.

Reconfigurability enabling robot with optimum functions for any tasks is something that conventional robots have lacked.

Our proposed robot does not require useless calculation cost, functions, or power consumption, and it is able to execute high-quality tasks at right situation as needed.

Therefore, it seems reasonable to conclude that our proposal is an effective approach for constructing humanoid robots.

The R1 experiment showed, however, that a robot with manual offline reconfiguration mechanism presents two problems. First, it takes time for users to customize the robot because the robot needs the system shutdown whenever the robot is reconfigured. This would be solved by introducing automatic reconfiguration. Second, despite having more expandability and flexibility than conventional humanoid robots, researchers and users may need more flexible reconfigurable robots such as those with upper and lower arm modules. This is solved by providing modules with appropriate sub-modules.

We are now developing a next-generation reconfigurable modular humanoid robot R2 to solve these problems and confirm the effectiveness of reconfiguration mechanism.

6. Conclusions

We have discussed the design and implementation of reconfiguration mechanism for a modular humanoid robot. R1 with our proposed reconfiguration mechanism is reconfigurable manually while not operating. It may serve as a breakthrough for practical next-generation humanoid robots reconfigured manually or automatically while operating. Reconfiguration solves the problems of expandability and flexibility through functional robot design, reconfigurable mechanical connectors, and a reconfigurable software model.

Experimental results and evaluations show that our proposed modular humanoid robot can be used as some kinds of humanoid robots and autonomous robots such as wheeled robots, arm robots, and head robots by our reconfiguration mechanism. The proposed robot achieves whole-body motions by combining functional robots and users of our proposed robots use the appropriate functions for the appropriate situation can be also seen.

In summary, our humanoid robot with the proposed reconfiguration mechanism is one of the effective approaches for constructing a humanoid robot.

This innovative concept of reconfigurable humanoid robots will become a hot topic in robotics field in the future.

Acknowledgements

This study was supported by Special Coordination Funds of the Ministry of Education, Culture, Sports, Science and Technology of Japan funds from CREST, JST, and Grants for the Advancement of Research at the Graduate Schools of Keio University.

We thank Masahiro Noda of Sanwa Metal Industrial Co., Ltd., for his constructive criticism and helpful advice on robot development.

References:

- [1] T. Taira, N. Kamata, and N. Yamasaki, "Design and Implementation of Reconfigurable Modular Humanoid Robot Architecture," In Proc. of the 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 1071-1076, 2005.
- [2] T. Taira, and N. Yamasaki, "Development of Modular Humanoid Robot based on Functionally Distributed Modular Robot Architecture," Journal of Robotics and Mechatronics, Vol.17, No.3, pp. 236-247, 2005.
- [3] T. Fukuda, and Y. Kawauchi, "Cellular Robotic System (CEBOT) as One of the Realization of Self-Organizing Intelligent Universal Manipulator," In Proc. of IEEE International Conference on Robotics and Automation, pp. 662-667, 1990.
- [4] S. Murata, E. Yoshida, K. Tomita, H. Kurokawa, A. Kamimura, and S. Kokaji, "Hardware Design of Modular Robotic System," In Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2210-2217, 2000.
- [5] M. Fujita, and K. Kageyama, "An Open Architecture for Robot Entertainment," In Proc. of International Conference on Autonomous Agents, pp. 435-442, 1997.
- [6] M. Fujita, H. Kitano, and K. Kageyama, "A Reconfigurable Robot Platform," Robotics and Autonomous, Vol.29, pp. 119-132, 1999.
- [7] S. Kagami, K. Nishiwaki, J. Kuffner, Y. Kuniyoshi, M. Inaba, and H. Inoue, "Online 3D Vision, Motion Planning and Bipedal Locomotion Control Coupling System of Humanoid Robot: H7," In Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2557-2562, 2002.
- [8] K. Kaneko, F. Kanehiro, S. Kajita, K. Yokoyama, K. Akachi, T. Kawasaki, S. Ota, and T. Isozumi, "Design of Prototype Humanoid Robotics Platform for HRP," In Proc. of IEEE/RSJ International

Conference on Intelligent Robots and Systems, pp. 2431-2436, 2002.

- [9] V. Yodaiken, and M. Barabanov, "RT-Linux," <http://www.rtlinux.org>.
- [10] Y. Ishiwata, and T. Matsui, "Development of Linux which has Advanced Real-Time Processing Function," In Proc. 16th Annual Conference of Robotics Society of Japan, pp. 355-356, 1998.
- [11] R. Bischoff, and V. Graefe, "Integrating, Vision, Touch and Natural Language in the Control of a Situation-Oriented Behavior-Based Humanoid Robot," In Proc. of IEEE International Conference on Systems, Man, and Cybernetics, pp. 999-1004, 1999.
- [12] T. Ishida, "Development of a Small Biped Entertainment Robot QRIO," In Proc. of IEEE International Symposium on Micromechatronics and Human Science, 2004.
- [13] Bosch, "CAN specification version 2.0," Published by Robert Bosch GmbH, September, 1991.
- [14] A. Kamimura, S. Murata, E. Yoshida, H. Kurokawa, K. Tomita, and S. Kokaji, "Self-Reconfigurable Modular Robot," In Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 606-612, 2001.
- [15] SH, <http://www.renesas.com/>.
- [16] N. Yamasaki, "Responsive Processor for Parallel/Distributed Real-Time Control," In Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 1238-1244, October, 2001.
- [17] "Universal Serial Bus Specification Revision 2.0," <http://www.usb.org>, 2000.
- [18] Microsoft Corp, "Plug and Play Design Specification for IEEE1394 V.1.0c," <http://www.microsoft.com>, 1999.
- [19] Flexray, <http://www.flexray-group.com/>.
- [20] Responsive Link, <http://www.itscj.ipsj.or.jp/ipsj-ts/02-06/toc.htm>.
- [21] H. Kobayashi, and N. Yamasaki, "RT-Frontier: A Real-Time Operating System for Practical Imprecise Computation," In Proc. of the 10th IEEE Real-Time and Embedded Technology and Applications Symposium, pp. 255-264, May, 2004.



Name:

Tetsuya Taira

Affiliation:

Ph.D. candidate at Keio University's Graduate School program

Address:

3-14-1 Hiyoshi, Kohoku-ku, Yokohama 223-8522, Japan

Brief Biographical History:

2003 Received the B.E. degree in information and computer science from Keio University, Yokohama, Japan

2005 Received the M.E. degree in science for open and environmental systems from Keio University, Yokohama, Japan

2005 Joined Toyota Motor Corporation

2005- Ph.D. candidate at Keio University's Graduate School program

Main Works:

- T. Taira, N. Kamata, and N. Yamasaki, "Design and Implementation of Reconfigurable Modular Humanoid Robot Architecture," IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 1071-1076, August, 2005.

Membership in Learned Societies:

- The Robotics Society of Japan (RSJ)
 - The Japan Society of Mechanical Engineers (JSME)
-



Name:

Nobuyuki Yamasaki

Affiliation:

Associate Professor, Department of Information and Computer Science, Keio University

Address:

3-14-1 Hiyoshi, Kohoku-ku, Yokohama 223-8522, Japan

Brief Biographical History:

1996- Researcher with Electrotechnical Laboratory, Agency of Industrial Science and Technology, MITI

1997-2000 Researcher with PRESTO (Sakigake21), JST

1998- Research Associate with Faculty of Science and Technology, Keio University

1998-2001 COE Researcher with Electrotechnical Laboratory

2000- Assistant Professor with Faculty of Science and Technology, Keio University

2002- Guest Researcher with Digital Human Laboratory, AIST

2004- Associate Professor with Faculty of Science and Technology, Keio University

Main Works:

- N. Yamasaki, "Responsive Multithreaded Processor for Distributed Real-Time Systems," Journal of Robotics and Mechatronics, Vol.17, No.2, pp. 130-141, April, 2005.

Membership in Learned Societies:

- The Institute of Electrical and Electronics Engineers, Inc. (IEEE)
 - The Robotics Society of Japan (RSJ)
 - Information Processing Society of Japan (IPSJ)
-