Dynamic Infrastructure for Personal Robot : DynI

S. G. Roh*, K. H. Park*, K. W. Yang**, J, H, Park*, H. S. Kim**, H. G. Lee**, and H. R. Choi*

* School of Mechanical Engineering, Sungkyunkwan University, Suwon 440-746, Korea
** Korea Institute of Industrial Technology, Chonan, 330-825, Korea

Abstract: *The advanced infrastructure for accelerating the development of personal robots is presented. Based on this structure, effective ways for integrating the various commercial components and interfacing among them are studied. The infrastructure includes the technology such as modularization based on independent processing and standardization open to other developers. The infrastructure supports not only that each hardware component of a personal robot can be easily attached to and detached from the whole system mechanically but also that each software of the components can be functionally distributed. As a result, we developed the fully modularized personal robots mechanically, and a virtual machine for the control of these robots. In this paper the proposed infrastructure and its implementations are described.*

Keywords: Personal robot, Home Robot, Infrastructure, Modularization, Standardization, IEEE1394, Virtual Machine

1. Introduction

As the standard of human being's living is elevated, the research about personal robots, which appropriately coincides with the improvement of IT (Information Technology) and the realization of home automation, appears to be the worldwide trend. This tendency has something to do with integrated intelligent environments such as intelligent room [1], intelligent space [2], and smart room [3], which originate in ubiquitous computing of the third wave in computing [4]. These environments need the physical agent for interacting with people, and personal robots can be considered as the proper agent. The personal robot for interfacing between people and the environments seems to require few sensors and not to be smart because the robot is supported by the environment with integrated intelligence. Consequently, the personal robot technology will be influenced by and may be dependent on the intelligent environments. On the other hand, the research about the more advanced personal robot, which has its own capability such as sensibility, mobility and smart intelligence without the intelligent environments, is still attractive area. This is because the personal robot will be a new independent creature beyond mere devices which a human being has designed to extend his life zone.

In concert with these changes, The Project of Basic Technology Development for Personal Robots was set about in 2002, Korea and it will be supported by Ministry of Commerce, Industry and Energy for ten years. The project, in which scores of research groups in universities and companies participate, consists of five sub-projects: I) Development of Entertainment and Game Robots, II) Development of Control and Sensing Technology for Personal Robots, III) Development of Information and Intelligence Technology for Personal Robots, IV) Development of Mechanism and Core Parts, and V) Development of System Engineering Technology for Personal Robots.

The project is caused by the investigation that the paradigm in the personal robot industry proceeds to the modular-based development as shown in Fig. 1. The sub-project V in which

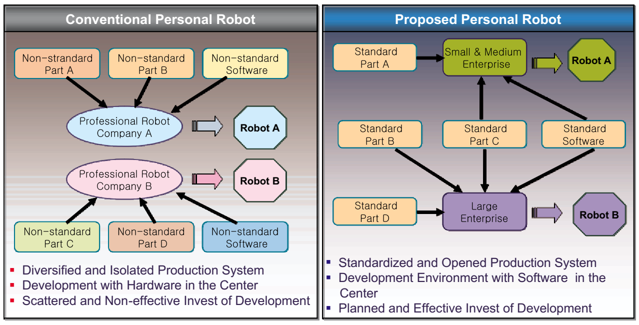

Fig. 1. Change of Paradigm for Development of Personal Robots

authors are involved is the study required for the integration of the other sub-projects because the project composed of the sub-projects covers the whole field of personal robot technology. With the study on the integration of the other sub-projects, our goal is concerned with the fact that personal robots should be able to registered as consumer products. As a premise to achieve this goal, however, the fundamental infrastructure in the related technologies such as standardization and modularization is need.

In the previous work[5], we mainly introduced how to prepare a consistent and systematic framework for the development of personal robots, and described several issues related to the proposed framework such as modular designs and connections of hardware. With DHR I (Digital Home Robot I) of our original personal robot platform composed of some modules, the framework was evaluated through the experiment such as hot-plugging, the navigation using method pointing vision images[16].

In the present study, we introduce an infrastructure named 'DynI' (Dynamic Infrastructure for personal robot). It means not only software framework but also hardware framework composed of some 'Module-D's (Module of DynI). It is improved and materialized in caparison with the abstract framework of the previous study. DynI which is breathed into by our original 'VM-D' (Virtual Machine of DynI) supports that each hardware module of the personal robot can be easily attached to and detached from the whole system

Corresponding Author, Email : hrchoi@me.skku.ac.kr. This research is supported by The Project of Basic Technology Development for Personal Robots of Ministry of Commerce, Industry and Energy in Korea.

mechanically. It also presents a methodology so that each software of the module can be functionally distributed. To go into more details, it provides methods for the connection between a higher and lower task. Based on this environment, a software development is classified into some gradual levels, so that the developers for lower-level-task can provide the developers for the higher-level-task with the subordinate device driver. This is because VM-D, which controls and arbitrates Module-Ds, supports independency and portability. Therefore, DynI can be concerned with the subject of integration, or rather, it is a advanced infrastructure for accelerating the development of personal robots. In the section II, we present the idea and concept of DynI. The section III describes VM-D and Module-D which are intimately associated with DynI in detail. In the section IV, we discuss the present and future work, and concludes the paper.

2. DynI : Dynamic Infrastructure for personal robot

This section illustrates two point of view about the proposed infrastructure. One is the basic concept coping with the situation of the exiting framework or architecture of computational systems such as personal computers. The other is the extended concept as the advanced infrastructure for accelerating the development of personal robots.

2.1. Problem Statements and Basic Idea

Personal robots are frequently presented in the recent years. Most of them have wheels[18], [19], [21], [22], [23]. To all outward appearances, because they look like moving vehicles such as cars though a few have often manipulators and legs, the development of them can be seem to be simple. However, it is not easy to develop them because personal robots require high technologies such as navigation and intelligence. On the other hand, the difficulty of developing them is due to the fact that they should satisfy not specific demands but various requirements from a lot of users as the terminology personal implies.

In comparison with a personal computer, the characteristic of the personal robot is similar to that of the personal computer not only because both of them require the computational technology but also because the personal computer should satisfy demands of multiform users. There, however, are some difference between the two. Firstly, the personal computer has already become widespread. The popularity of it originated from its standardization based on compatibility and extendability. The standardization enables companies to develop it speedily though it requires the professional and high-level technology. This means that the progress and popularization of the personal robot on the whole also depends on the standardization as described in Fig. 1. Secondly, a personal computer has the centralized control architecture which consists of CPU on a main board and functional devices such as VGA card, sound card and monitor, and allows time-delayed communication for the data transmission among devices, whereas the personal robot has the distributed and multi-processing framework such as OPEN-R, OpenHRP, BERRRA, and PETER owning to requiring heavy process[10], [11], [12], [13], [14]. These models represent considerable progress, and will be referred by other developers. On the other hands, they have their own merits which may be still far from standardization. For example, AIBO of Sony copes with a flexible and adaptable platform [10]. It chose OPEN-R as system architecture and modular components as mechanical hardware. The system layer of AIBO handles the in-output of sound and vision, control joints of legs, etc., while application layer is for users. In addition, the legs and head of the version ERS-210 are compatible with those of ERS-220. AIBO is continuously updated to new model type as platform based on OPEN-R, but it is designed for a pet robot and still seems to be a model for Sony products, not for other companies.

The proposed infrastructure is designed for effective integration of commercial and common products so that the effort of robotic companies are sufficient as only possessing its own development capability for a certain module or devices: camera, infrared sensor, mobile, joint, casing, etc. In other words, it corresponds to the standardization for personal robots covering compatibility and modularization. Fig. 2 shows that the various Module-Ds (to be described in the next section) are connected mechanically and electrically, and integrated into some assemblies.

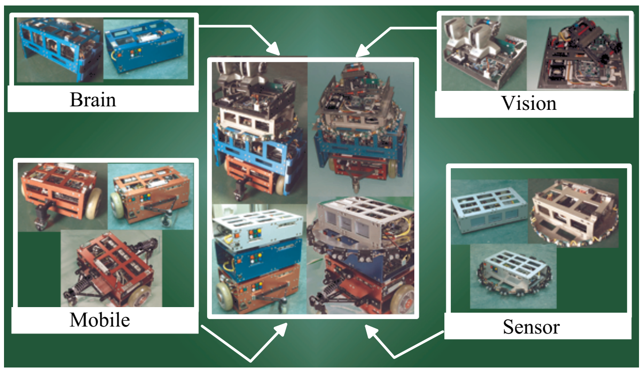

Fig. 2. Integration of Module-D based on standardization

2.2. Advanced Concept

The proposed infrastructure supports the standardization basically as well as that of personal computers. The standardization of the personal robot, however, has its own constitution different from that of the personal computer. The personal computer is operated passively by user's instructions in static place, whereas personal robots should respond to the surroundings such as obstacles, and should be able to move actively from place to place without the command of users. This technology of the personal robot is based on the intelligence which requires sensing, localization, etc. In other words, the personal robot should be not static but locomotive, and also adapt to the constantly changeable intelligence technologies which requires various strategies and algorithms difficult for generalization. These dynamical features of personal robots enforce themselves to react real-timely for speedy locomotion and to adapt itself to the new and various artificial intelligence. DynI (Dynamic Infrastructure for personal Robots) is planed on the basis of these concepts;

the terminology 'Dynamic' of DynI also has its origin in the concept. In other words, DynI implies two meanings. One is to support realtime control such as behavior-based control. The other is to have a adaptable construction such as combination of its capability. Consequently, DynI is the infrastructure for the alternative standardization of the personal robot which is different from the typical normalization of the personal computer.

3. Construction of DynI

As a whole, two main elements support DynI. They enable DynI to be the integration framework for the modular-based standardization, and furthermore, make DynI the advanced infrastructure for accelerating the development of personal robots. They are VM-D (Virtual Machine of DynI) and Module-D (Module of DynI). VM-D is mainly concerned with software and becomes the fundamental element for DynI. Module-D means the module of proposed robot which is controlled and arbitrated by VM-D, and it accomplishes DynI. In this section, VM-D and Module-D are presented in detail.

3.1. VM-D : Virtual Machine of DynI

Virtual machine is a common technique to give portability to the embedded systems incorporated with compilers or interpreters. In 1995, Sun Microsystems released Java technology, Java language and Java virtual machine. They make the software on Java environment run on different kinds of OSs and hardwares installed Java virtual machine. The technology has greatly influenced the development paradigm of products. Though Java virtual machine provides a high portability to a system when installed, it brings drawbacks in the development time and execution performance when used for developing robot modules distributed over network. Since neither can Java have low-level commands for the hardware control nor execute byte codes fast on its interpreter, it is not suitable for the time-critical control systems like robots which should be controlled in real-time. A new virtual machine called VM-D (Virtual Machine of DynI) with RPL (Robot Programming Language) is presented. It has the same advantages like system-independency and portability as Java virtual machine, its execution speed is improved fast, and it provides the frequently used functions and algorithms as the intrinsic functions in itself. The structure of the standardized module interface and how to utilize VM-D are also studied and implemented.

3.2. Configuration of VM-D

VM-D can be ported into different kinds of hardware platforms and operating systems such as Microsoft's Windows, Linux and RTLinux. It is used for running the byte codes of a control program on Module-D. Module-Ds can correspond with each other through virtual ports, and the ports are the communication channels of point-to-point connections. Actual port mapped for physical communication is the FireWire IEEE1394. It can be applied to USB, RS-232C and Ethernet with TCP/IP.

VM-D is stack-oriented. it takes one or more operands from the operand stack of its current frame, and pushes the results back onto the operand stack. A new frame is created each time a method is invoked, and the frame is created a new operand stack and set of local variables for usage by the method. VM-D is composed of register, memory and executor, and defines various runtime memory areas that are used during the execution of a application program: byte code, global variable, function pointer , stack and heap areas. The

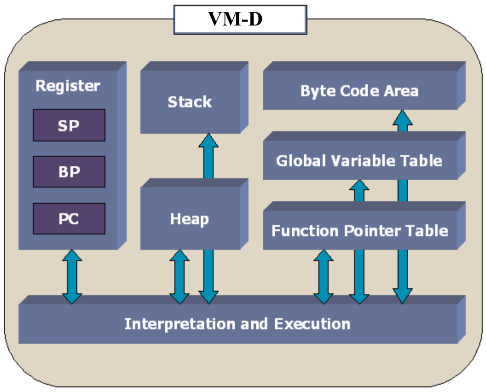

Fig. 3. Configuration of VM-D

structure of VM-D is shown in Fig. 3 and its constituents are as follows:

•Register

VM-D has three 32-bit register (SP, BP, PC). They are used for accessing a stack or contain the address of VM-D instruction which is currently executed. SP (Stack Pointer) holds the address of the top element of a stack in memory. BP (Base Pointer) holds the address of the base point of a stack in memory. PC (Program Counter) contains the address of the next byte code to be executed.

• Stack

Each task of VM-D has a private stack which created at the same time as the task. The stack is analogous to that of a conventional language such as C: it holds local variables and partial results, and passes parameters to methods and receives method results. All instructions of VM-D take operands from the operand stack, operate on them, and return the results to the stack.

• Heap

A heap is a runtime data area from which the memory for all strings and arrays is allocated. The heap is created on the start-up of VM-D .

• Byte Code Area

A byte code area is similar to the store for the compiled code in conventional languages or text segment. It stores the byte code and symbol tables.

• Global Variable Table

A global variable table is shared among all tasks of VM-D.

• Function Pointer Table

A function pointer table contains the address of a function body.

• Instruction Set

An instruction called a byte code consists of a one-byte opcode and operands. The one specifies the operation to be performed. The other supplies parameters or data which will be used by the operation. Instructions have no operands, or have one or two opcode. The byte code instruction stream is only byte-aligned. These decisions keep the code of VM-D for the compiled program compacting

3.3. RPL : Robot Programming Language

To write the application program of personal robots, RPL (Robot Programming Language) is presented. In other words, VM-D is developed in order to control and arbitrate each heterogeneous OS or Mini-OS, while RPL supports the developers of the application program for personal robots on the basis of the unification of various OSs with VM-D. It is a programming language such as C++, FORTRAN and PASCAL. The program written in RPL is compiled by RPL-compiler to translate into the byte codes as shown in Fig. 4. The byte codes are downloaded into Module-D through the

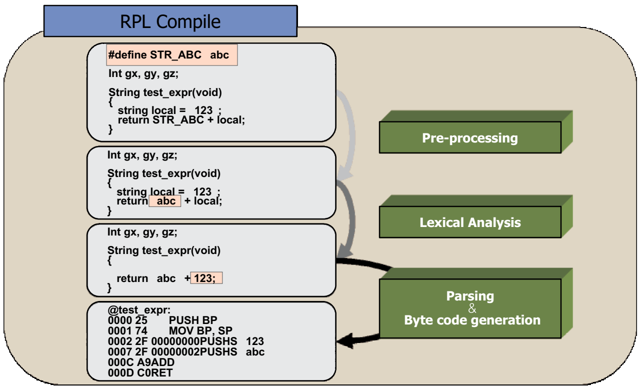

Fig. 4. RPL compile

ports, interpreted and executed on VM-D of Module-D.

DynI is the hardware and software environment and Module-D based on DynI is the hardware and software platform. To control Module-D, VM-D calls the function of Module-D such as 'mobile.goto(x, y)' or the function of components which constitutes Module-D such as 'mobile.readEncorder(motor1); the function is named the API (Application Programming Interface). In other words, API means the collection of low-level function or low-level function itself. As shown in Fig. 5, API and VM-D layer sep-

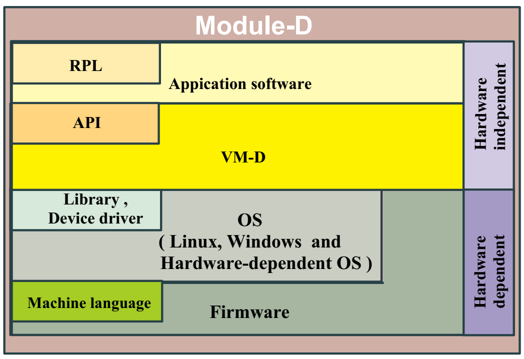

Fig. 5. Software structure of Module-D

arate and insulate the application software from the hardware dependencies. VM-D is the base for Module-D and is ported onto various hardware. API is a large collection of ready-made software functions that provide many useful capabilities. Module-D programs such as navigation and image processing are run on its own VM-D. Furthermore, one Module-D can execute API of the other Module-Ds. The variable of RPL has a data type. A data type of the variable determines the values which the variable can contain and the operations which can be performed on it. All of the primitive data types which are integer, floating point, and string are supported by RPL.

While compiling, a character in RPL source code is expressed in continuous tokens. RPL compiler recognizes five types of token: identifiers, keywords, literals, operators and miscellaneous separators. White spaces (tab, carriage return and line feed) and comments can serve to separate tokens. As shown in Fig. 4, pre-processor and compiler pass four steps while the source code compile into the byte code : pre-processing, lexical analysis, parsing and link.

3.4. Module-D for Developer for Personal Robots

This subsection presents that the approach based on Module-D of DynI offers new methodology to developers of various fields such as design, construction, implementation, and evolution of the personal robot application. The applications can be assembled from Module-Ds with a variety of sources, that is, Module-Ds themselves may be written in several different programming languages and run on several different platforms. Nevertheless, Module-Ds can be reconfigured, replaced or reprogrammed, as long as they continue to provide the same interface to the same level of quality. This gives the considerable flexibility to the whole systems. Therefore, Module-D supports the integrated environment and platform so that companies should develop their low-level components. It is realized and accomplished because Module-Ds are controlled by VM-D. VM-D is applied to Module-D with OS such as Windows and Linux. It is also based on the components with its own controller which belongs to Module-D; in this paper, the methodology by Module-D is presented for simple description. Fig. 6 shows VM-D and public interfaces. Each Module-D has its own property, method. The property

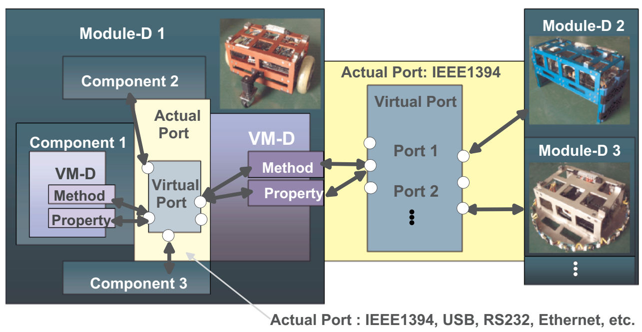

Fig. 6. Module-D interface and port configuration

is the specific value defined in Module-D. It determines the characteristic of Module-D and is used in order to allocate a certain value to Module-D or to be allocated a certain value from Module-D. The method means the procedure for regulating some actions of each Module-D. For example, when it is said that a robot moves, the verb move is one of methods which belongs to Mobile Module-D. With the property and method, the internal states and objects of one Module-D can be accessed by the other Module-Ds.

3.5. Dual characteristic of Module-D for control

In general, a module is defined as a unit of independent software and hardware performing a specified function or task. The proposed Module-D can be installed, replaced and used independently according to the electrical and mechanical standards. These features of Module-D are similar to those of a common module. Module-D, however, has special and peculiar features. Each Module-D is composed of the components (often called devices or units) which have similar characteristics or are grouped with specific requirements. In other words, one Module-D which is independent of the other Module-Ds is the part of a whole system, and also the assembly which integrates various components. This concept of Module-D was designed to satisfy both of deliberative and reactive control as shown in Fig. 7. To execute

As a part of DHR	As an assembly of components
Integrated into DHR	Integrating its components
Combination of Module-Ds	Decision by Module-D itself
Dependent on DHR	Independent of DHR
Deliberative control	Reactive control
Slower response	Real-time response
High-level Intelligence	Low-level Intelligence
Variable latency	Simple computation
Representation-dependent	Representation-free
Predictive capabilities	Reflexive capabilities

Fig. 7. A dual characteristic of Module-D

a certain mission or scenario, Module-Ds are combined as parts of the proposed personal robot under the control of the integrated high-level intelligence; this means the feature of adaptable construction as mentioned in the previous section. Whereas, each of Modules can behavior immediately by itself in case of emergency because each of them is a assembly which combines inner components of itself under control of its own low-level intelligence. For example, when the robot listens the commands of "Go to the living room and turn on TV", it analyzes the assigned mission, and execute the mission with the combination of Module-Ds. On the other hand, when a present mission is to clean a room and the robot detects a baby on the moving path, Mobile Module-D controls motors to avoid a baby without regard to the conditions of any other Module-Ds or the mission. To communicate the large-sized information for the deliberative control and transmit fast the low-level data for the behavior-based reactive control, IEEE1394 is used as the communication network among Module-Ds (bandwidth of Maximum 400Mbps). To be suited to these controls, Module-D has its own controller for interfacing internal components. Each Module-D is characterized as follows:

• Brain Module-D learns sequential behaviors such as symbolic planning and reinforcement learning. It synthesizes the information or low data from the other Module-Ds to analyze a circumstance and the state of whole system. In addition, it takes charge of communication of DHR as human being's brain controls a nervous system.

• Sensor Module-D has various components to detect and sense surroundings like human being's sense organ. It integrates various sensor interfaces. A kind of sensor belongs to the other Module-Ds as occasion demands. For example, Mobile Module-D has a close-range infrared sensor for a emergency stop. Sensor Module-D of DHR has SBC(Single board computer) as a controller. It will be replaced to the integrated board for sensor-interface developed by the researchers of the sub-project II; the board interfaces ultrasonic, laser, infrared.

• Mobile Module-D provides mobility of DHR. It has been tested with various mechanisms such as differential drive and synchro-drive; synchro-drive mechanism was developed by the researchers of the sub-project IV.

• Vision Module-D was designed for the complicated processing of vision signals. Advanced Vision Module-D such as 3D-stereo vision and CMOS image sensor has been developed by the researchers of the sub-project IV.

• Other Modules which are Voice recognition & synthesis, Sound localization, Modular-arm, Artificial skin, and Navigation & localization have been developed and will be developed by the researchers of the sub-project II and IV.

3.6. Materialization of DynI

To evaluate the infrastructure DynI for the integration of Module-Ds, authors developed DHR (Digital Home Robot) I and II which are our original personal robot platforms. Module-Ds of DHR have been renewed continuously. Fig. 8

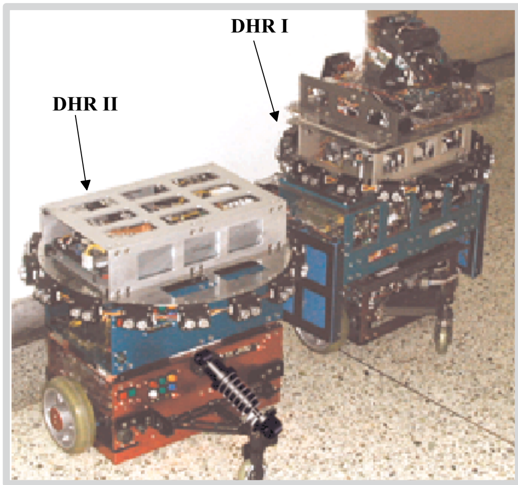

Fig. 8. DHR I and II based on DynI

shows the platform. Each Module-D of DHR can be easily attached to and detached from the whole system mechanically. It is the shape of a rectangular parallelepiped and connected each other with the standardized hardware connector. The mechanism was designed in order to put together the new models of Module-D and component. The more hardware information of Module-D and DHR were described in the previous paper [5]. Fig. 9 shows some navigation experiments by combination of Module-D; it represents the navigation of DHR when Vision and Sensor Module-D are attached to and detached from two Module-D which are Brain and Mobile Module-D. Each of Fig. 9(a), (b), (c) and (d) mean map-based, sensor-based, vision-based and all-equipped navigation. DynI and DHR of its robot platform have been applied

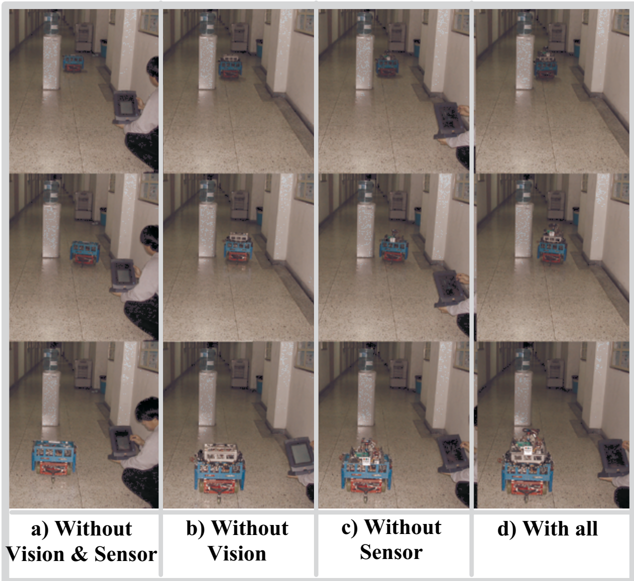

Fig. 9. Navigation by combination of Module-Ds

to RDC (Robot Design Center) which is developed by our co-worker. It is graphical design tool for the developers and end-users of personal robots [6]. Furthermore, Authors have developed new applications for Dyni. They are studies on the control of electric home appliances using vision image and the localization using RF-ID in the intelligence home based on ubiquitous computing. They will be presented in the next paper.

4. Conclusions

In this paper, we introduced the infrastructure DynI for personal robots, and also presented VM-D and Module-D as its essential elements of software and hardware. They support the integration technology and modular-based development because they are independent of hardware and OS. Consequently, DynI can realize the standardization for personal robots. Furthermore, it can accelerate the development of person robots. We applied the proposed DynI to our platform DHR I and II, and proved its capability as the infrastructure for personal robots. Our goal is to develop an the more effective and adaptable infrastructure to be shared by the other developers. In the future work, we will apply DynI to the application for personal robots which are missions and scenarios in intelligent home environments.

References

[1] Rodney. A. Brooks, "Intelligent Room Project", *Proc. of Second International Conference on Cognitive Technology*, pp. 271-278, Aug 1997.

[2] Joo-ho Lee, Hideki Hashimoto, "Intelligent Space - concept and contents", *Advance Robotics*, vol. 16, No. 3, pp. 265-280, 2002.

[3] Alex Pentland, "Smart rooms, smart clothes", *Proc. of Fourteenth International Conference on Pattern Recognition*, vol. 2, pp. 949-953, Aug 1998.

[4] M. Weiser, "The Computer for the Twenty-First Century", *Scientific American*, pp. 94-104, September 1991.

[5] S. G. Roh, S. M. Baek, D. H. Lee, K. H. Park, T. K. Moon, S. W. Ryew, J. Y. Kim, T. Y. Kuc, H. S. Kim, H. G. Lee, H. R. Choi , "Development of Personal Robot Platform : Approach for Modular Desing", *International Conference on Control, Automation and System (ICCAS)*, pp. 2313-2318, October 2002.

[6] H. R. Kim, D. W. Kim, H. S. Kim, H. G. Lee, "Toward the Personal Robot Software Framework", *International Conference on Control, Automation and System (ICCAS)*, pp. 2307-2312, October 2002.

[7] P. Fiorini, E. Prassler, "Cleaning and Household Robots: A Technology Survey," *Autonomous Robots*, vol. 9, No. 3, pp. 227-235, December 2000.

[8] E. Prassler, A. Ritter, C. Schaeffer, P. Fiorini, "A Short History of Cleaning Robots," *Autonomous Robots*, vol. 9, No. 3, pp. 211-226, December 2000.

[9] B.J Lee, H.G. Lee, J.H Lee, G.T. Park, "New architecture for mobile robots in home network environment using Jini," *Proc. of IEEE International Conference on Robotics and Automation*, vol. 1, pp. 471-476, May 2001.

[10] M. Fujita, "Digital Creatures for Future Entertainment Robotics" *Proc. of IEEE International Conference on Robotics and Automation*, vol. 1, pp. 801-806, April 2000.

[11] F. Kanehiro, K. Fujiwara, S. Kajita, K. Yokoi, K. Kaneko, H. Hirukawa, Y. Nakamura, K. Yamane, "Open Architecture Humananoid Robotics Platform" *Proc. of IEEE International Conference on Robotics and Automation*, vol. 1, pp. 24-30, May 2002.

[12] A.H. Jones, G.N. DeSouza, A.C. Kak, "A Multi-processing Software Infrastructure for Robotic System," *Proc. of IEEE International Conference on Robotics and Automation*, vol. 1, pp. 193-198, May 2001.

[13] M. Lindstrom, A. Oreback, H.I. Christensen, "BERRA: A Research Architecture for Service Robots," *Proc. of IEEE International Conference on Robotics and Automation*, vol. 4, pp. 3278-3283, April 2000.

[14] E. Guglielmelli, C. Laschi, P. Dario, "Robots for personal use: humanoids vs. distributed systems," *International Symposium on Humanoid Robots,*, October 1999.

[15] C. Sayers, *Remote Control Robotics*. New York: Springer Verlag, 1998.

[16] T. Sekimoto, T. Tsubouchi, and S. Yuta, "An Implementation of a Human Interface using a Touch Panel and Visual Images for Controlling a Vehicle," *IEEE International Workshop on Robot an Human Communication*, pp. 451-455, September 1996.

[17] Linux 1394 Library. http://www.linux1394.org

[18] T. Fong, C. Thorpe "Vehicle Teleoperation Interfaces," *Autonomous Robots*, vol. 11, No. 1 pp. 9-18, July 2001.

[19] R. Alami, M. Herrb, B. Morisset, R. Chatila, F. Ingrand, P. Moutarlier, S. Fleury, M. Khatib,T. Simeon, "Around the lab in 40 days," *Proc. of IEEE International Conference on Robotics and Automation*, vol. 1, pp. 88-94, April 2000.

[20] G. Dudek, M. Jenkin *Computational Principles of Mobile Robotics,*. Cambridge, United Kingdom : Cambridge University Press, 1998.

[21] A. Goeurguiev, P.K. Allen, E. Gold, P. Blaar, "Design, Architecture and Control of a Mobile Site-Modeling Robot," *Proc. of IEEE International Conference on Robotics and Automation*, vol. 4, pp. 3266-3271, April 2000.

[22] F. Michaud, J. Audet, D. Letourneau, L. Lussier, C. Theberge-Turmel, S. Caron, "Experiences with an Autonomous Robot Attending AAAI," *IEEE Intelligent Systems*, vol. 16, No. 5, pp. 23-29, Sept 2001.

[23] M. Beetz, T. Arbuckle, T. Belker, A.B. Cremers, D. Schulz, M. Bennewitz, W. Burgard, D. Hahnel, D. Fox, H. Grosskreutz, "Integrated, plan-based control of autonomous robot in human environments," *IEEE Intelligent Systems*, vol. 16, No. 5, pp. 56-65, Sept 2001.