Personalization of Dynamically Loaded Service Programs for a Human-Robot Communication Channel

Akihiro Kobayashi† Yasuyuki Kono† Atsushi Ueno†† Izuru Kume† Masatsugu Kidode†

†Graduate School of Information Science,
Nara Institute of Science and Technology (NAIST)
8916-5 Takayama, Ikoma 630-0192, JAPAN
{akihi-ko, kono, kume, kidode}@is.naist.jp

††Faculty of Engineering, Osaka City University
3-3-138 Sugimoto, Sumiyoshi-ku,
Osaka 558-8585, JAPAN
ueno@info.eng.osaka-cu.ac.jp

Abstract

This paper presents a middleware architecture for personal robots when applied to various environments. This architecture allows a user's personal robot to consistently integrate environment-oriented applications to the robot's personality in its own application. These applications share sensors and actuators, and generate consistent actions in sequence. However, integration of two different kinds of independently developed applications causes a lack of its continuity and a slow response in human-robot communication. To solve these problems, we have designed a new middleware for personal robots, called PRIMA (Personal Robots' Intermediating Mediator Adaption to environment) which can schedule the outputs of these applications to keep open the human-robot communication channel. We call the integration of these applications a personalization of dynamically loaded service programs. In addition, our experiment demonstrates the effects of PRIMA on human-robot communication.

1 Introduction

Recent research studies provide several points of view for the use of autonomous mobile robots, for example, a mobile and intelligent interface for information systems [3, 9, 8], and a familiar and amusing robot like a pet. Especially as a pet robot, human-like expression in its appearance and motion is an important issue [5, 6]. A *personal robot* is developed for home use and has to be controlled based on the above two view points [1]. In the near future, a personal robot should be an interface which adapts to its visiting environment, e.g., an office building, a museum, a library, and an amusement park. We attempt to make a personal robot load an application dynamically in its visiting environment in order to adapt to the environment [7].

A personal robot should accomplish a set of tasks given in a visiting environment, keeping its personality. A owner feels its personality from its familiar motions and conversations, in its applications. The personality might be an important factor for the commercial value of the personal robot. For example, a robot behaves as a pet interacting with the owner, when he or she is accompanied by the robot. However, originally programmed applications and newly loaded applications for an environment tend to interfere with each other in actions of the robot. This kind of interference causes a lack of continuity and response speed in human-robot communication. Here, the problem to be solved is how to reduce its interference and integrate two kinds of applications. We suggest a mediating middleware, which schedules requests to devices of a robot from applications developed in the same way as typical robot applications. The middleware keeps the human-robot communication channel, in order to balance continuity against response speed.

In Section 2 and 3, we introduce the concept of "mediation" to keep the communication channel established. Implement techniques of a middleware "PRIMA (Personal Robots' Intermediating Mediator Adaptation)" are described in Section 4. Finally, we demonstrate some experiments with discussion in Section 5.

2 Mediation

2.1 Integration of Two Different Kinds of Applications

A new mediation framework is required for a robot to execute multi-applications in parallel, and these applications should be developed independently. *Subsumption Architecture* [2] and *Situated Multi-Agent Architecture* [4] are robot architectures which allow a robot to execute multi-behaviors in parallel. The *Subsumption Architecture* model represents a process as parallel data flows from sensors' inputs to outputs by actuators. This model introduces layers of system modules according to the various levels of goals. Interventions from upper layers to lower layers are allowed. Designing under *Subsumption Architecture*, a programmer must understand details of the lower layers before designing their upper layers.

Situated Multi-Agent Architecture integrates multi-behaviors to learn relations among modules running in parallel, but these relations are fixed under *Subsumption Architecture*. *Situated Multi-Agent Architecture* requires learning time and predefined links between modules which are related to each other, for robots to behave appropriately. PRIMA, when compared with these architectures does not need cooperation between *FA* (**Familiarity-oriented application**) and *EA* (**Environment-oriented application**), to keep it flexible enough to develop these applications, because personal robots should have various original motions, and act in various environments.

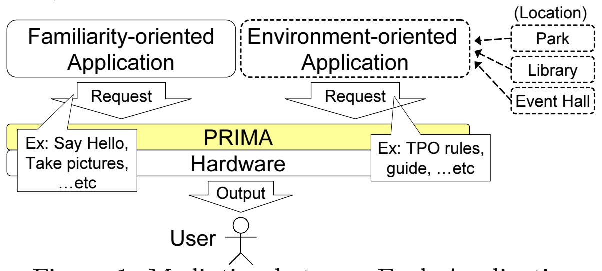
Familiarity-oriented Application
Environment-oriented Application
(Location)
Park
Library
Event Hall
Request
Request
Ex: Say Hello, Take pictures, …etc
PRIMA
Hardware
Ex: TPO rules, guide, …etc
Output
User

Figure 1: Mediation between Each Application

As Fig.1 shows, a personal robot has its original application to represent familiar actions (*FA*). On the other hand, the robot dynamically loads a new application to perform an information service program in the current environment (*EA*). PRIMA is an online interpreter that can block competitive device accesses, and can dynamically schedule applications' requests without applications' internal states.

2.2 Communication Interference

Interference among requests from *EA* and *FA* is one of the most important issues in mediating between them. When *EA* and *FA* simultaneously access the same device, or when an application accesses a device which another application needs to be still, these interferences make a robot unnatural. For example, when *FA* requests the robot to speak something, the request sometimes interferes with *EA*'s request, e.g., speech recognition. Moving to a goal away from a user requested by *EA* interferes with approaching the user in response to the user's touch. The robot cannot quickly reply to the user because of the interference in such a case. PRIMA gives importance to continuity and response speed, which such interference reduces, because continuity and response speed are the essentials for natural communications between human and robot.

Continuity : Human-robot communication needs consistency in its sequence of interactions. However, both *EA* and *FA* interrupt outputs of each other when one application interferes with the other. An interruption in the robot's behavior causes recognition errors by the robot, and misleads the user.

Response Speed : A personal robot should respond to its user's inputs as quickly as possible because most of its tasks need interactions with its user, rather than, for example, batch processing, which doesn't require input from the user. However, a personal robot may delay its reactions to its user, when *EA* and *FA* interfere with each other.

If an extreme continuity of one application is kept in PRIMA, then the response speed of the other application will be reduced. To estimate the trade-off between the continuity and the response speed, we propose a communication model for personal robot as described in Section 3. PRIMA schedules the requests from *EA* and *FA* following the communication model to solve these interferences.

3 Communication Channel

We define the communication channel between a personal robot and its user as established when the robot faces its user. We assume that the user feels that the robot acts naturally when information units, e.g., gesture, voice, or touch, are transmitted over the channel.

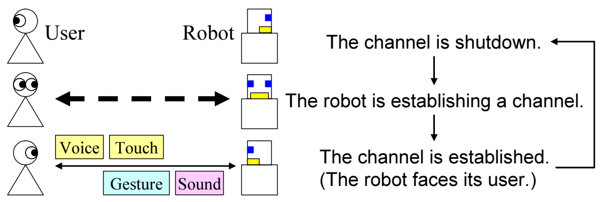
User
Robot
The channel is shutdown.
The robot is establishing a channel.
Voice
Touch
Gesture
Sound
The channel is established.
(The robot faces its user.)

Figure 2: Human-Robot Communication Channel

In this paper, we assume that a personal robot interacts with only its owner. In such a situation, a robot owner feels that his/her robot's actions are natural if they are scheduled by following rules.

(1) PRIMA should not interrupt an information unit.
(2) One application should not shutout a communication channel when the other will transmit an information unit.
(3) Information units should be transmitted as quickly as possible.

PRIMA adopts these rules in this order. For Rule (1), we defined one of the smallest continuous sequences which a robot or an owner needs to understand, as an information unit. For example, a turn of utterances, a sequence of gestures, and a couple of question and answer pairs are information units. Rule

(2) calls for a robot to face its owner when speaking to the owner. Rule (3) guarantees response speed. We implement these rules using an internal expression called an **Atomic Behavior** (*AB*), which is an external running unit. PRIMA never simultaneously executes an *AB* which interferes with other *AB*s, and guarantees that an *AB* is never interrupted. PRIMA defines appropriate block size of *AB* based on Rule (1), and schedules using *AB*'s priority based on Rule (2) and (3).

4 Mediation Model

4.1 Atomic Behavior

Controlling exclusive execution by keeping sequences included in robot behaviors is easy if the expressing relations among *AB*s are made into a tree, called as *AB* tree. Fig.3 shows *AB*s divided from the behavior of a robot as it guides its owner through a museum and explains certain museum displays. Each node represents an *AB*, and each link represents a running sequence. Each node runs from a root to the leaves of a tree, and all nodes of a branch run in parallel.

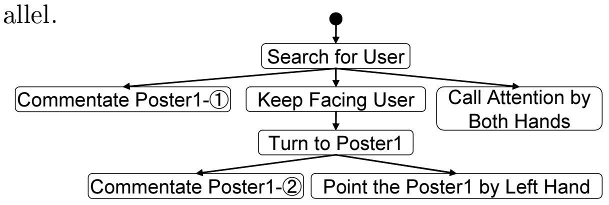

Figure 3: Atomic Behavior Tree

Two types of *AB*s are defined by rule (1). A turn of speech and recognition are generally defined as two of the smallest communication units. However complex gestures and emotional dances are application specific sequences which need continuity for an owner to understand them. Consequently, PRIMA allows applications to group *AB*s into an *AB*. We call this operation by applications an *Expansion of AB Blocks*. The smallest motions of the personal robot used to control stability or the smallest motions limited by devices are also defined as *AB*s, for example, setting the speed during a minimum sampling rate of a device.

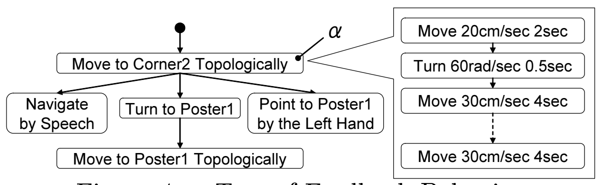

Figure 4: a Tree of Feedback Behaviors

PRIMA defines each motion in feedback loops as *AB*, which a robot often needs to behave reactively. For example, Fig.4 shows the behavior of a robot navigating its owner in a museum, indicating the direction of the goal in each corner of the museum room. The node α includes many instances of behaviors adjusting their trajectories. The owner feels it is natural that his/her robot interrupts some behaviors which have a feedback loop, and resumes easily from interruption because the robot has a flexibility in the trajectories moving toward its goals. PRIMA reconfigures an *AB* tree, when *AB*s are created depending on the state of robots at each moment.

4.2 Execution Model

Both *EA* and *FA* request *AB* trees to PRIMA asynchronously. As in Fig.5, each *AB* has information about concrete actions, links, and occupying devices. PRIMA checks devices occupied with each *AB*, and exclusively executes *AB*s which occupy the same devices because simultaneous access to the same device causes interference between *FA* and *EA*. The priority of each *AB* decides the execution order of competitive *AB*s.

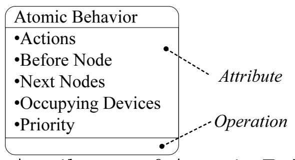

Figure 5: Attributes of Atomic Behavior Class

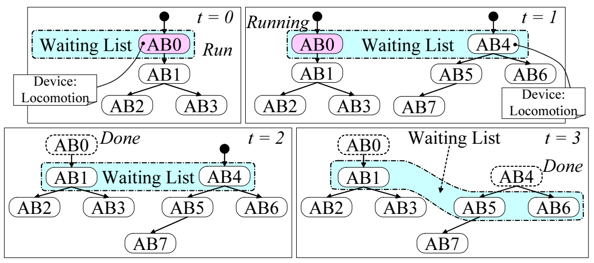

Figure 6: Working Memory of PRIMA

Fig.6 shows a snapshot of the working memory (*WM*) of PRIMA. In Fig.6, one application requests the *AB* tree (0-3) at the time of (t=0), and the other application requests the *AB* tree (4-7) at (t=1). PRIMA updates *WM*, when PRIMA accepts *AB* trees, and when each *AB* completes its actions. When PRIMA accepts an *AB* tree, PRIMA adds the root node of the tree into the waiting list in *WM*, and starts the *AB* of the root node as long as the *AB* doesn't interfere with other running *AB*s. For example, PRIMA doesn't start AB4 at (t=1), because AB0 and AB4 have a same occupying device. When each *AB* completes its action, PRIMA adds the child node of an accomplished *AB* into the waiting list, and selects starting *AB*s. If there is any interference among nodes in the waiting list, PRIMA tries to start the nodes which have a higher priority and does not interfere with any running AB. For example, at (t=2), PRIMA adds AB1 into the waiting list, and checks interference between AB1 and AB4. In the case of (t=2), AB4 is selected. After the end of AB4, PRIMA adds the children of AB4 into the waiting list and selects starting nodes.

This exclusive execution model guarantees continuity in an AB. Exceptionally, PRIMA interrupts an AB, when the AB tree conflicts with other trees requested by the same application. In order to make a robot reactive, a robot application often requests a new behavior before other past behaviors been completed. PRIMA cancels and deletes executing AB trees in cases which have the same occupying devices with later AB trees requested from the same application. Therefore, PRIMA can assume that two ABs in the waiting list are requested from a different application if they interfere with each other.

4.3 Priority Control

PRIMA allows a robot to keep its communication channel because a priority of AB is defined to follow rule (2) and (3) described in Section 3. We assume that the general functions of a personal robot are locomotion and communication. Furthermore these functions are categorized as personal robot behaviors according to their effect on the channel. Fig.7 shows the categories. We give higher priority to the behavior included in a higher category.

A) Establish Channel
- Active-Sensing-User, Turn-to-User

B) Transmit over Channel (Give no Effect)
- Speak, Q&A, LED, Arm-Gesture, Turn-and-Speak-to-User

C) Keep Channel
- Tracking-User

D) B) or E) (Depending on Parameters)
- Move-Robot-Coordinate, Turn, Move-Head

E) Shutdown Channel
- Move-Topological-Position, Move-Global-Coordinate

Figure 7: Design Policy of Priority

Category A) is given a higher priority than B) because a robot starts communication facing to the wrong direction if a behavior of B) can interrupt a behavior of A). Category B) is given higher priority than C), because it follows Rule (3) described in Section 3. Category C) is given higher priority than D) because a behavior of C) probably shutdowns a communication channel, even if other applications need a communication channel. Category D) is given higher priority than E) because a behavior of E) doesn't usually show information.

4.4 Implementation

On PRIMA, FA and EA call interfaces of libraries named **Command**, which implement concrete hardware actions, and create ABs which have appropriate block size and attributes as described in Section 4.1 and 4.3. Fig.8 shows the implementation of PRIMA. FA and EA request Command trees, which express execution sequences of Commands. A Command creates ABs, and **Request Listener** reconstructs an AB tree based on the Command tree. **Manager** executes them using a time-sharing execution. An AB carries concrete parameters accessing devices created by a Command, and tells these parameters to devices when the Manager starts the AB.

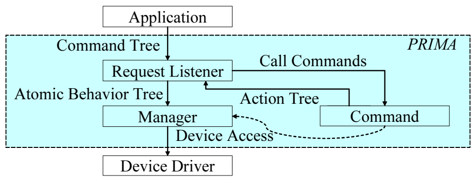

Figure 8: Implementation of PRIMA

5 Experiment

5.1 Simulation

We defined ideal outputs of the mediation between example applications, and estimated how close the outputs of PRIMA were from the ideal outputs, and we demonstrated the effects of the rules described in Section 4.1 and 4.3. We designed the example applications shown in Table.1. FA1-5 are the FAs of a personal robot. For example, Fig.9 shows the outputs of FA2. EA1-3 are EAs in each environment. EA4-5 are functions of an EA in an environment. They are considered to cooperate with each other as shown in Fig.10. In Fig.9 and Fig.10, a line mostly shows an AB, and a colored line shows the event which causes a change to the robots' behaviors.

Table 1: Example Applications

FA	1	Express Emotions
	2	Walk Following User
	3	Dance to a Tune
	4	Take a Picture
	5	Set Video Timer
EA	1	Join in a Parade of an Amusement Park
	2	Guide a Museum and Explain Displays
	3	Search Books in a Library
	4	Navigate to a Room
	5	Tell Messages from Hosts of Rooms

First, we designed mediation examples as shown in Fig.11, which were made from EA4-5 and FA2 by ideal time-sharing execution. We made 14 mediation examples as in Fig.11, changing applications and situations. We interviewed 10 test subjects, and tested whether they felt the 14 mediation examples were natural or not. The subjects checked dialogs as in Figs.9-11, which showed the outputs of 14 mediation examples and the 10 applications shown in Table 1. We asked the subjects to observe the following rules.

```
 0 I  The user touched the robot sleeping.
 1 R  "What's the matter with you?"
 2 U  "Come on, Follow me."
 3 R  The robot started the special behavior Turn-and-Search, saying
      "Where are you?"
      # if (The robot could not find the human after turning 360 degrees)
      then goto 4;
      # if (The robot found human) then goto 6;
 4 I  The robot missed the Turn-and-Search.
 5 R  The robot entered sleeping mode saying, "Sorry, I could not find you."
 6 I  The robot found the user.
 7 R  The robot started following the user, saying "I found you, I'll start
      following you."
 8 R  The robot modified its position and pose to keep an appropriate
      distance from the user.
      # The robot repeated 8 behaviors;
 9 I  The user touched the robot.
10 R  "May I stop following you?"
11 U  "Yes."
12 R  The robot entered sleeping mode.
13 I  The robot lost the user.
      # goto 3;
      *** special behavior Turn-and-Search ***
*1 R  The robot stays still, until the robot will recognize the human by its
      camera.
      # if (The robot doesn't find the human for ten seconds) then goto *2;
      # if (The robot found the human) then succeed the behavior;
*2 R  The robot turns the angle of its camera's view, and will stay still and
      find the human.
      # if The robot doesn' t find the human after turning 360 degrees)
            then fail the behavior;
      # if (The robot doesn't find the human for ten seconds) then goto *2;
      # if (The robot finds the human) then succeed the behavior;
```

Figure 9: Output Example of FA2

```
 0 R  The robot entered the building, and started the EA4-5.
 1 R  "Where are you going?"
 2 U  "Prof. Kidode's room."
 3 R  "Come on, let's go"
 4 R  The robot started topological moving to the goal.
 5 I  The robot reached a corner.
 6 R  The robot pointed a hand and turned to the direction of the goal,
      saying "Turn to ***"
      # The robot repeated 5 and 6 n times
 7 I  The robot got a socket message from Prof. Kidode, while navigating the
      user to the goal.
 8 R  The robot stopped there, and said "I've just got a message."
 9 R  "I'm Kidode, I'll be late for your appointment about 5 minutes, because
      of meeting now"
10 R  "Do you reply anything?"
11 U  "Yes." "I see. I'll wait in your room."
12 R  The robot send the message to the message server, saying "O.K. I'll
      send your message."
13 R  The robot restarted the topological moving.
14 I  The robot reached the goal.
15 R  The robot stopped, and said "I've just reached the goal."
```

Figure 10: Output Example of EA4-EA5

- A subject can interchange between lines when he/she prefers to do so.
- A line marked by "*" includes multiple *AB*s which allow other lines to interrupt them.
- A line closed by "[" is composed of an *AB* that doesn't allow interruption.
- A subject cannot change the timing of events, because PRIMA can not control the timing.
- A subject cannot add or delete lines, because PRIMA can not add or delete *AB*s.

As the result of the interview, 91% of 140 trials felt natural to the subjects. 8 subjects agreed that 11 of 14 mediation examples felt natural. The other 3 mediation examples tended to depend on the preference of the subjects. For example, at line 13 in Fig.11, 3 of

```
     0 R  E  The robot entered the building, and started the EA4-5.
     1 R  E  "Where are you going?"
     2 U  E  "Prof. Kidode's room."
     3 R  E  "Come on, let's go"
     4 R  E  The robot started topological moving to the goal.
     5 I  E  The robot reached a corner.
*    6 R  E  The robot pointed a hand and turned to the direction of the goal,
             saying "Turn to ***"
     7 R  E  The robot restarted topological moving to the goal.
     8 I  F  The user touched the moving robot.
     9 R  F  The robot stopped and said "What's the matter with you?"
    10 U  F  "Come on, Follow me."
*   11 R  F  The robot repeated the special behavior Turn-and-Search, saying
             "Where are you?"
    12 I  E  The robot got a socket message from Prof. Kidode, while searching
             the user.
*   13 R  F  The robot repeated the special behavior Turn-and-Search, saying
             "Where are you?"
    14 I  F  The robot found the user.
    15 R  F  The robot started following the user, saying "I found you, I'll start
             following you."
  [ 16 R  E  The robot stopped there, and said "I've just got a message."
  [ 17 R  E  "I'm Kidode, I'll be about five minutes late for my appointment
  [          because I'm still in a meeting."
  [ 18 R  E  "Do you have a reply?"
  [ 19 U  E  "Yes." "I see. I'll wait in your room."
  [ 20 R  E  The robot sent the message to the message server, saying "O.K.
  [          I'll send your message."
    21 R  F  The robot resumed following the user.
*   22 R  F  The robot repeated a modification of its positions and pose to keep
             an appropriate distance from the user.
    23 I  F  The user touched the robot.
    24 R  F  "May I stop following you?"
    25 U  F  "Yes."
*   26 R  E  The robot restarted topological moving to the goal.
    27 I  E  The robot reached the goal.
    28 R  E  The robot stopped, and said "I've just reached the goal."
```

Figure 11: Ideal Output

10 subjects preferred replying to the message before searching for the owner. Nonetheless, over half of the subjects responded that all mediation examples felt natural. Therefore, these mediation examples can be assumed as ideal outputs of time-sharing execution, most of which feel natural.

5.2 Estimation of AB and Priority

In order to execute the 14 ideal outputs, application programmers should describe not only command trees, but also additional information. The additional information expresses *Expansion of AB Blocks* and application specific priority which overrides default priority following the rule in Section 4.3. We estimated the amount of additional information and the programmers' efforts needed by the 14 ideal outputs.

The 14 ideal outputs needed the 2 descriptions of *Expansion of AB Blocks* and no description of application specific priority for the 10 applications in Table.1. The *Expansion of AB Blocks* were needed by behaviors whose sequence had some means, such as a parade of EA1 and stillness during taking a picture of FA4. As the result shows, the cases needing additional information are limited to application specific behaviors. Application programmers can easily recognize these cases. Therefore, the practical efforts of application programmers can keep robots natural by mediation of PRIMA.

5.3 Discussions

In this section, we discuss the probability of PRIMA based on the interviews in Section 5.1. According to the interviews, the subjects felt that the ideal output was satisfactory to them, at 71% of 140 trials. The default priority and block size of ABs are correctly defined. However, the subjects said 29% of 140 trials needed scheduling improvement. The following (1)-(5) are the main points the subjects would like to improve if possible.

(1) rejection of *AB* to keep robot natural
(2) interruption of speech recognition
(3) interruption of a turn of speech communication
(4) system message which informs interruption
(5) recovery state of a robot before interruption

(1) is difficult, because rejection of an *AB* needs PRIMA to decide the necessity of the AB. (2) and (3) are requirements of response speed. In order to reduce errors of speech recognition, PRIMA defines a turn of interaction as an AB, and does not allow the robot to speak during speech recognition. As a result, unnatural behavior remains. For example, a robot doesn't show any reaction to the owner's touch, while the robot is waiting for owner's utterance. We attempt to solve this problem by the way of (4) and (5). For example, (4) shows that an owner can allow his/her robot to decrease response speed, if the robot can show its status as "interruption arises", "cannot interrupt", and "recovering", using its LED, and speakers, for example. The response speed of the robots becomes fast if PRIMA can insert a recovering *AB* as (5), for example, by retrying the same interrupted question. These recovery behaviors need PRIMA to understand the states which the robots are in, and the states which the applications require. In addition, PRIMA needs a function to customize an owner, because each owner favors different scheduling, as described in Section 5.2.

6 Conclusions

In this paper, we proposed a framework to give a personal robot the ability to supply local services in several environments, keeping its personality. We attempt to integrate a dynamically loaded application and its own application which brings its personality. The integration of two different kinds of applications which have been independently developed causes a lack of continuity and response speed in human-robot communication. We solve the trade-off between continuity and response speed, by appropriately keeping a human-robot communication channel. We proposed a middleware named PRIMA, which schedules an Atomic Behavior tree expressing effects to the communication channel by a time-sharing execution. We demonstrated the effect of PRIMA with a consideration of a programmers' efforts to make robots ideal. We are planning to test the effects of PRIMA using concrete hardware, and to improve libraries and description methods for application programmers.

References

[1] `http://www.incx.nec.co.jp/robot/`.

[2] R. A. Brooks. A Robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation, RA-2*, 2(1):14–23, March 1986.

[3] J. Buhmann, W. Burgard, A. Cremers, T. H. D. Fox, F. Schneider, J. Strikos, and S. Thrun. The Mobile Robot Rhino. *AI Magazine*, 16(1):31–38, 1995.

[4] I. Hara and Y. Motomura. Situated Multi-Agent Architecture for an Autonomous Robot. In *5th Robotics Symposia*, pages 86–91, 2000.

[5] M. Imai, T. Kanda, T. Ono, H. Ishiguro, and K. Mase. Robot Mediated Round Table: Analysis of the Effect of Robot's Gaze. In *Proc 11th International Workshop on Robot and Human Communication (RO-MAN)*, pages 411–416. IEEE, September 2002.

[6] T. Kanda. *A Constructive Approach for Communication Robots.* PhD thesis, Kyoto University, 2003.

[7] A. Kobayashi, A. Ueno, I. Kume, Y. Kono, and M. Kidode. Robot middleware architecture mediating familiarity-oriented and environment-oriented behaviors. In *Proc. International Symposium on Computational Intelligence in Robotics and Automation.* IEEE, 2003.

[8] T. Matsui, H. Asoh, J. Fry, Y. Motomura, F. Asano, T. Kurita, I. Hara, and N. Otsu. Integrated Natural Spoken Dialogue System of Jijo-2 Mobile Robot for Office Services. In *Proc. 16th National Conference on Artificial Intelligence (AAAI-99)*, Florida, July 1999.

[9] S. Thrun, M. Bennewitz, W. Burgard, A. B. Cremers, F. Dellaert, D. Fox, D. Hahnel, C. Rosenberg, N. Roy, J. Schulte, and D. Schulz. MINERVA: A Second-Generation Museum Tour-Guide Robot. In *Proc. International Conference on Robotics and Automation (ICRA)*, pages 1999–2005. IEEE, 1999.