Design and Applications of Learning Conversational Agents

Masayuki Okamoto
Abstract

In this thesis, we propose a framework to develop *learning conversational agents* which support information retrieval, guidance, or mediation among humans with a learning functionality. Our goal is constructing learning conversational agents which join human community.

There are two issues to construct a learning conversational agent as an actor in a community. The first issue is providing information through conversation. To provide information, there are many works related to conversational agents which naturally understand a user’s requests and bring information to the user. In general, a usual agent is constructed after making many rules or collecting many corpora. However, it is difficult to debug an agent in an actual communication or collect example conversation in advance.

One solution of this problem is to develop this kind of agents incrementally with human cooperation. Combining collecting corpora and learning dialogue models makes both works easier. In this thesis, we adopt the machine learning technology to collect example dialogues through actual conversation.

The other issue is sharing information for mutual understanding. To share information, a real or virtual communication environment is necessary. However, only few environments support information sharing between people while there are many environments where people can gather from various places. A kind of communication environment where information is shared among agents and humans is a solution.

Moreover, when a conversational agent join communication environ-
ment, another issue appears. In this situation, we should consider occasional conversation, called scene-driven conversation, which appears between task-oriented conversations in communication environment. This kind of conversation itself is not essential for the task accomplishment. However, when we consider a long-term task or application, it comes up between the task.

With respect to the above issues, the thesis studies the following topics.

1. Incremental agent construction process through actual conversation

   We have proposed a new framework to construct learning conversational agents which talks with users from actual dialogues incrementally. In this method, a conversational agent is constructed through the cycle of collecting example dialogues with the Wizard of Oz method and learning dialogue models from example dialogues. At first, the developer should input almost all answers. As the learning proceeds, the system infers appropriate utterances and the cost of human guide’s inputting utterances reduces. Finally, a conversational agent is constructed.

   We have also developed an incremental probabilistic-DFA learning algorithm to construct a finite state machine (FSM) as a dialogue model. This algorithm decreases the recomputation cost with caching the merging information.

   We confirmed the reduction of the human guide’s cost to input or modify utterances as the growth of the dialogue models in the Kyoto tour guide task, and we also confirmed that our incremental learning algorithm reduced the total number of checking compatibilities.

2. An environment where a conversational agent mediates human-human communication

   We have proposed an environment where people visually share their information such as their background or profile, which is provided from agents. This environment is designed to support communication
in the real world or between remote locations with the following three functions: (1) showing topics based on participants’ profiles and cultural background; (2) life-sized, large-screen interface; and, (3) displaying objects which show feelings of identify.

From the evaluation of these systems, we studied that these systems add new functionality to support conversation contents, which may be especially useful in a cross-cultural context where language skills are an issue, and this type of environment may be especially useful in a pre-collaboration context.

3. Agent construction in communication environment

When a developer constructs conversational agents which treat with scene-driven conversation, it is reasonable to put in each occasional interaction in actual use in the developer’s judgement rather than to put interactions in advance. We have proposed a kind of conversational agents which treats with scene-driven conversation which derives from place-based, verbal, or nonverbal cues, and developed a construction system of the agent on the scenario-based agent system with a 3-D virtual space. We also developed rules to grasp typical scene-driven conversation.

Thus, in this research we have made an application-based study of conversational agents from the viewpoint of providing and sharing information. The application design investigated in the thesis makes the research direction in conversational agents which join human-human communication.
Acknowledgments

I would like to express my sincere gratitude to my supervisor, Professor Toru Ishida at Kyoto University, for his continuous guidance, valuable advice, and helpful discussions. I gratefully acknowledge valuable comments of other members of my thesis committee at Kyoto University, Professor Katsumi Tanaka and Professor Hiroshi G. Okuno.

I wish to thank Dr. Katherine Isbister, who played an important role in the system and experiment design. Lecturer Hirofumi Yamaki and Assistant Professor Hideyuki Nakanishi at Kyoto University always gave me kind advice and helpful comments.

I am grateful to other vice advisers, Professor Kenji Mase at Nagoya University, Professor Hiroyuki Tarumi at Kagawa University, Associate Professor Yasuhiro Kitamura at Osaka City University, and Associate Professor Hiroki Takakura at Kyoto University, for their valuable advice and discussions.

I appreciate the constructive discussions and the kindly advices from Professor Hiroshi Ishiguro at Osaka University, Associate Professor Toshikazu Nishimura at Ritsumeikan University.

I also wish to express my special thanks to Dr. Yasuhiro Katagiri at ATR before starting my research of conversational agents. He gave me important ideas about conversational agents. Professor Katashi Nagao at Nagoya University gave me many inspiring ideas about communication support systems at Sony CSL.

Brief mention should be made of the work by my colleagues at Professor Ishida’s laboratory. Yeonsoo Yang and Mitsushiro Sugayama developed an
early version of the Web-based conversational agent interface, and collected many example dialogues. Nobutoshi Yamanaka wrote a program for the topic management functions in the Web-based conversational agent, and developed an interface of the monitoring and editing functions in a 3-D virtual environment.

Finally, I wish to thank all members of Professor Ishida’s laboratory for their help and fruitful discussion.

This research was supported in part by followings:

- The Japan Science and Technology Corporation for the Core Research for Evaluational Science and Technology Universal Design of Digital City Project.


# Contents

1 Introduction ................................. 1
   1.1 Objectives .................................. 1
   1.2 Research Issues ............................. 2
   1.3 Thesis Outline .............................. 4

2 Background .................................... 7
   2.1 Conversational Agents ...................... 7
   2.2 Communication Environment .............. 11

3 Learning Conversational Agents for Information Providing 15
   3.1 Introduction ................................ 15
   3.2 Wizard of Oz Method for Conversational Agents .... 16
   3.3 Dialogue-model Learning .................... 19
      3.3.1 Preliminaries .......................... 19
      3.3.2 General Form of Quadratic Algorithms .. 21
      3.3.3 Incremental PDFA Learning Algorithm .... 22
   3.4 WOZ-based Agent System .................. 29
      3.4.1 System Architecture ..................... 29
      3.4.2 Dialogue Model ........................ 33
      3.4.3 Knowledge and Topic Management ........ 34
      3.4.4 System Flow ............................ 36
   3.5 Experimentation ............................ 38
      3.5.1 Learning Cost .......................... 41
      3.5.2 Wizard’s Load .......................... 43
6.1 Contributions ........................................... 99
6.2 Future Directions ........................................ 102

Bibliography .................................................. 105

Publications .................................................. 115
List of Figures

3.1 Conceptual process of constructing agents .................. 17
3.2 Example of PTA ............................................. 20
3.3 Intuitive expression of merging two states .................. 21
3.4 Procedure merge ............................................. 22
3.5 General form of quadratic algorithms by lexicographic breadth-first order 23
3.6 Range of states which needs recomputation .................. 24
3.7 Procedure incremental_merge with renewal of recomputation set 26
3.8 The general form of incremental quadratic algorithms .... 27
3.9 A step of the state-merging method (probabilities are omitted) 29
3.10 System architecture .......................................... 30
3.11 Screenshot of each client ..................................... 31
3.12 Dialogue model and example dialogues ....................... 34
3.13 Annotation tool .............................................. 35
3.14 Example dialogue (the original dialogue is in Japanese) ... 40
3.15 Comparison of the count for decision of merging ........ 42
3.16 Comparison of perplexities ................................. 44
3.17 Change of the wizard’s load ................................. 45
3.18 Change of dialogue models ................................. 47
3.19 Dialogue model after learning 100 examples ................. 48
4.1 Silhouettell .................................................. 60
4.2 Silhouettell system configuration ............................. 61
4.3 Encounter with Silhouettell ................................. 63
4.4 Questionnaire about using Silhouettell (7 highest, 1 lowest ranking) 69
4.5 Reaction of participants .................................. 70
4.6 Networked Silhouettell ................................. 73
4.7 Actual use of Networked Silhouettell .................. 73
4.8 Contents selection ....................................... 76
4.9 Changing the kind of information on the screen .......... 76
4.10 System configuration .................................... 77
4.11 Questionnaire evaluation of Networked Silhouettell (8 highest, 1 lowest ranking) 79

5.1 Architecture of agent construction environment .......... 89
5.2 FreeWalk screen image ................................... 90
5.3 Description of a parallel observation with guarded commands 91
5.4 Monitoring interface ..................................... 92
5.5 Edit window ............................................. 93
5.6 Example conversation triggered by non-linguistic action .. 95
5.7 Example conversational rule when an avatar stops to turn her body to a shop 96
5.8 Example conversation triggered by relationship between actors 97
5.9 Example conversational rule when an avatar stops to turn her body to a shop 98
List of Tables

3.1 Utterance tags used in the tour-guide task in Kyoto (TS means ‘tour spot’) 38
3.2 Ratio of each tag of user utterance .......................... 39

4.1 Comparison between Silhouettell and Networked Silhouettell 59
4.2 Rates of nationalities ............................................ 65
4.3 Where would Silhouettell best be used? ....................... 65
4.4 What data should be displayed? .............................. 66
4.5 Where would Silhouettell best be used? ....................... 67
4.6 Cultural differences in releasing data ......................... 67

5.1 List of cues and actions ..................................... 91
Chapter 1

Introduction

1.1 Objectives

The number of the Internet users in East Asia is expected to increase to 300 millions. The Internet brings various communications such as cross-cultural communications. Electronic communication support systems such as message services or video conferencing systems are spreading to connect people from various places or countries. Once they are connected, it becomes more important and interesting to consider an aspect of electronic communication support, the support of conversational contents to let people know each other and smooth away misunderstandings among them.

Usual works about electronic communication support systems have focused on The computational aspect of such systems, e.g., system performance, as well as their impact in business offices or laboratories have been discussed extensively. However, the aspects of the process of interpersonal understanding or socialization between co-workers before the actual task, have not been thoroughly examined.

One solution of this problem is sharing conversational contents by providing them with conversational interface. This interface is spreading for information retrieval or guidance task. When you would like to know how we go from a bus stop to another stop, you can call a voice service system and just tell the system starting bus stop and destination stop. Then, you can
hear a proper route from your phone. On the Web, text-based conversational interface is used for virtual salespersons. However, this kind of interface is used only for 1-to-1-style conversation.

Our purpose is to construct a kind of conversational agents which mediates communication among people with the support of mutual understandings by providing shared contents through conversational interface. As such kind of interface, there are many works related to conversational agents which understand a user’s requests and bring information to the user. However, in general, it is difficult to develop an agent in community because a developer have to debug many rules or collect many corpora in advance.

In this thesis, we propose a concept, design, and applications of learning conversational agents, a kind of conversational agent which is incrementally developed from actual conversations with human cooperation, and we propose a communication environment to mediate interpersonal communication with providing shared contents, where a learning conversational agent joins a human community. We also treat with another conversation style, called scene-driven conversation, which occurs between task execution when an agent joins and lives in a communication environment.

1.2 Research Issues

In this thesis, we present the following research issues of learning conversational agents.

1. How to provide information through conversation

2. How to share information in conversation

To solve a former problem, the reduction of the cost of making conversational rules is important. In example-based construction, this cost is divided into the cost of collecting example dialogues and the cost of modeling dialogue models.

To reduce the collecting cost, we use the Wizard of Oz method enhanced by the functionality of learning dialogue models. In this method, a developer
collects annotated dialogues by chatting with users. At first, the developer should input almost all answers. As the learning proceeds, the system infers appropriate utterances and the cost of human guide’s inputting utterances reduces. Finally, a conversational agent is constructed.

To reduce the modeling cost, we develop an incremental learning algorithm to construct a finite state machine (FSM) as a dialogue model. For learning FSMs, learning algorithms which learn cyclic probabilistic finite-state automata with the state merging method are useful. However, these algorithms should learn the whole data every time the number of example dialogues increases. Therefore, the learning cost is larger when we construct dialogue models gradually. We proposed a learning method which decreases the recomputation cost with caching the merging information, and evaluated the number of checking compatibilities for merging states and the perplexities of learned models.

For the latter problem, we believe that the following three functions are important:

- showing topics based on participants’ profiles and cultural background,
- life-sized, large-screen interface, and
- displaying objects which show feelings of identify.

In this paper, we discuss the design, implementation, and the evaluation of two systems that were designed to support communication in the real world or between remote locations. As an applied situation, we focused on the cross-cultural situation in which people gather from various places or countries.

We also address the occasional conversation in communication environment. Usual studies on conversational agents have focused on only task-oriented conversation. However, there is another kind of conversation, or scene-driven conversation, which occurs between task-oriented conversations in communication environment such as a 3-D virtual space.
When a developer constructs conversational agents which treats with scene-driven conversation, it is reasonable to put in each occasional interaction in actual use in the developer’s judgement rather than to put interactions in advance. In this paper, we propose a kind of conversational agents which treat with scene-driven conversations, and propose an architecture of conversational agents which supports scene-driven conversation in communication environment.

1.3 Thesis Outline

This thesis consists of six chapters, including this chapter as the introduction.

Chapter 2 is devoted to introduce the background of this thesis to show why a new approach is necessary to develop an communication environment and conversational agent to mediate interpersonal communication.

Chapter 3 describes a new kind of agents called learning conversational agents, and a framework to develop learning conversational agents by collecting example dialogues with the Wizard of Oz method and learning dialogue models with the machine learning technology. In collecting dialogues, we used the method to infer and add an utterance tag which means the means of each utterance and is decided in advance. This method reduces the cost of human and makes the learning process easier. A finite state machine which is learned by a probabilistic DFA learning algorithm is used for a dialogue model. Since the algorithm is not effective in the incremental model construction, we proposed a technique to reduce computational cost by caching information of the previous learning session. We also present the effect of reduction of learning costs and reduction of the cost of collecting dialogues through the experimentation of tour-guide task.

Chapter 4 introduces a communication support environment with large-screen interface. We developed Silhouettell which promotes people to know each other by providing shared information including the common interests and the difference among them. We also show the result of experimenta-
tion in cross-cultural situations where it is hard to communicate because of differences in the participants’ native language.

Chapter 5 shows the method of constructing conversational agents in communication environment. In the actual conversation, a scene-driven conversation which accidentally occurs without other conversational contexts, such as a conversation which begins with finding out some objects in walking along the street, is important as well as a task-driven conversation such as searching information, guidance, and instruction. We propose an architecture of scenario-based agent in a 3-D virtual space and a construction system of the agent.

Chapter 6 concludes the thesis summarizing the result obtained through this research and the prospect of the future research.
Chapter 2

Background

To construct learning conversational agents which can join human communities, the technology of conversational interface which an agent provides information through conversation with humans and the technology of communication environment where information is shared among agents and people. In this chapter, we take a general view of both research areas.

2.1 Conversational Agents

For providing various information naturally, conversational agents which have spoken or text-based natural language interface is a solution of this problem.

There are two types of conversational agents: one kind of agents uses linguistic aspect such as spoken language or text, and the other kind of agents uses both linguistic and non-linguistic aspect including gestures or facial expressions. The former focuses on telling information briefly and precisely with only speech, such as in hotel reservation or call center domain. For example, VOYAGER [Zue90] is a voice system which communicates with users about tourist and travel information for the Greater Boston area. Whereas, the latter focuses on combine non-linguistic information such as visual cues. Necessarily, this kind of agents has some bodies. Therefore, it is important to pay attention to agents themselves and design them
as a kind of social existence with their characters and emotions.

This kind of conversational agents with their own bodies is called *embodied conversational agent (ECA)* [Cassell00a]. ECAs have their own bodies and communicates another actors with not only speech but also multimodal conversation in which agents acquire many cues and communicate with users in various ways with combinations of some modalities such as indication to objects, speech, gesture, facial expression.

Most agents have their own 3-D-modeled body, and some agents live in the 3-D virtual space. Social Agent [Nagao94a][Nagao94b] has facial expression and join two users’ conversation. Rea [Cassell00b] plays the role of a real estate salesperson in a 3-D virtual house. Mission Rehearsal Exercise Project provides a virtual world which imitates a small town in Bosnia for military soldiers to have experience peacekeeping activity [Traum02].

There are also many synthetic ECAs on the Web which introduce Web sites through task-oriented dialogues. For example, Jennifer in Extempo∗ sells cars virtually, and Luci in Artificial Life† introduces her own Web site. The Nijo Castle bus tour guide in Digital City Kyoto [Isbister00] guides 3D-modeled Nijo Castle to participants of a virtual bus tour. They play the role of guides, and prevent users from clicking all links of these sites. The number of such kind of interface agents is increasing.

An issue of conversational agent research is how to make an agent’s dialogue model. There are two methods for developing a dialogue model: one is a rule-based approach, and the other is a statistical approach. In rule-based approaches, for example, there are scenario design tools such as the CSLU Speech Toolkit [Sutton98]. This approach directly reflects developers’ intentions. ExpertClerk [Shimazu01] models conversation between salesclerks and shoppers and navigates appropriate merchandises by asking or proposing. However, the task/scenario designers should also design the whole rules, keywords, and the program.

In statistical approaches, for example, there are many works about slot-

---

∗http://www.extempo.com/
†http://www.artificial-life.com/
filling dialogue models with the Markov decision process [Singh00]. These methods learn models from dialogues.

Generally, in the actual systems, many conversational agents with finite state machines (FSMs) as their dialogue models have been developed [Alexandersson95][Stent99]. Usually, each system is used for a limited domain, therefore its FSM as a dialogue model is designed only for this domain. We should make up new FSMs whenever we change the scenarios or tasks.

Therefore, learning algorithms in which FSMs can be learned from examples are helpful for solving this problem. In general, it takes much time to collect example dialogues and people who collect dialogues should input many utterances or sentences. We proposed a framework to construct dialogue models incrementally through collecting example dialogues with the Wizard of Oz (WOZ) method [Fraser91] in which a human who pretends of a system (called wizard) talks with a user and learning dialogue models from collected dialogues [Okamoto01].

Deterministic finite-state automata (DFA) learning algorithms are applicable for learning FSMs from examples. There are many studies related to DFA learning, and learning minimum automata with finite positive examples and negative examples belongs to the NP-hard class [Gold78]. Therefore, some algorithms with heuristics for obtaining better results in polynomial time have been proposed (e.g. RPNI algorithm [Oncina92]).

In developing practical systems, there are some problems in example dialogues.

1. Negative examples cannot be collected, though positive examples are collected.

2. Examples may include noises.

3. The number of examples gradually increases.

If only (1) should be solved, an algorithm which needs only positive examples is proposed [Denis01], and if only (1) and (2) should be solved,
probabilistic deterministic finite-state automata (PDFA) with probabilistic parameters which can treat with noises are used. The first algorithm which learns PDFA is ALERGIA [Carrasco94] based on RPNI. ALERGIA learns cyclic automata with the state-merging method. MDI [Thollard00] is a better version of ALERGIA. MDI uses the Kullback-Leibler divergence for a decision of merging states.

However, these algorithms should recompute the whole steps when the number of example data increases. When the condition includes (3), these algorithms take much time. There is no algorithm which satisfies these three conditions, and algorithms which reduce the cost of recomputation are needed.

Another issue of conversational agent research is when and how the agent decides and speaks utterances from many sentences.

Usual conversational agents or conversational systems limit the domain of their users’ utterances by making their tasks or goals clear in order to handle these utterances easily.

Most conversational agents treat with a search/guidance task, or a pedagogical/presentation task. In the search/guidance task, it is important to fill slots to generate output answers with these slots. Therefore, the conversational task is to bring out keywords which fit these slots from the user. InfoWiz [Cheyer99] works as an information kiosk in a laboratory, and CommandTalk [Stent99] is an spoken interface of a battlefield simulator. Bayesian Receptionist [Horvitz99] limits user’s goal with a user model based on the Bayesian network.

In the pedagogical/presentation task, contents presented in a task or the order of them are predefined. An agent executes the task with confirming his/her user’s understanding in each step, changing the manner of speaking according to the degree of each user’s understanding, or answering user’s questions assumed in the task.

PPP Persona [André98] is a presentation agent on the Web. This agent has plans about presentation contents, and answers to a user’s request based on the described plans. Cosmo [Lester99] is a pedagogical agent which interactively advises knowledges about routers or networks, with gestures
and speech. Steve [Rickel98] demonstrates how to inspect a high-pressure air compressor aboard a ship in an immersive virtual space.

From the viewpoint of interaction with human, a brief or short conversation is not always good, and it is necessary to treat with topics which are not related to the task directly. Some agents have introduced some parameters about expressions or personalities to be life-like conversational agents which are easy to talk with.

Agents on Extempo’s Web site have their own social backgrounds, emotional parameters, and emotion-dependent answers for each topic to answer questions about themselves and express their emotions.

In an actual conversation in usual life, especially a conversation in the real world, each situation around a user which does not depend on conversational context or flow effects the conversation occasionally.

For example, deciding a route is a slot-filling task in a guidance. However, various conversations occur when the agent walk with the user along the route, and the time of these conversations is often longer than that of the guidance itself.

However, there is few works about conversational agents which treat with such kind of conversation.

When a developer constructs an agent which supports the occasional conversation, particularly the target space (the real world or a virtual space) is restricted, it is reasonable to put in each occasional interaction in actual use in the developer’s judgement rather than to put interactions in advance.

2.2 Communication Environment

The previous section mentioned some approaches to provide information with conversational interface. To apply this interface to mediate human-human communication, we need consider communication environments where a conversation agent joins.

To support interpersonal communication, various electronic communication support systems have been studied and developed. One approach is
to provide a virtual meeting space where people join from distributed locations.

In the groupware research area, video conferencing tools have been developed for supporting formal meetings between separated people. MediaSpace [Bly93], MAJIC [Okada94], InPerson§, and CU-SeeMe§ provides virtual spaces through computer networks. These systems are used mainly for supporting collaborations in companies or research activities.

There are another kind of communication or community-formation, named communityware or community computing [Ishida98a][Ishida98b]. This approach focuses on informal communication and socialization, which are earlier stages of collaboration; while groupware focuses on the collaboration itself. FreeWalk [Nakanishi96], for example, provides a virtual 3-D space to realize accidental encounters. Socia [Yamaki96] supports asynchronous meeting.

The other approach is to provide contents which socially connect people, while above systems are aimed to provide a virtual meeting space. There are also several projects or systems supporting collaboration or communication among users.

AIDE [Mase98] and a series of CoMeMo-Community activities [Nishida98][Kubota00] promotes discussion or knowledge sharing by structuring or visualizing each user’s topics or knowledge; while our systems provide cues to know each other by showing each user’s profile or interests. HyperDialog [Nagao98] supports face-to-face communications, using mobile agents whereby information is managed in each user’s mobile computer and is displayed personally. Let’s browse [Lieberman99] supports collaborative Web browsing among people by extracting profiles from users’ own web pages.

In recent years, systems which provide people with information using Large-screen device been developed. For example, the Campiello Project [Grasso98] is aimed at supporting interpersonal interactions using a shared

‡http://www.sgi.com/software/inperson/
§http://www.cuseemeworld.com/
large screen. In AgentSalon [Sumi01], each user has a personal agent which comes and goes between the user’s PDA and a shared large screen. In this environment, personal agents gathers on the screen and talk about users’ interests collected through operating their own PDAs.
Chapter 3

Learning Conversational Agents for Information Providing

3.1 Introduction

Our purpose is constructing conversational Web agents which learn task-oriented dialogue models and reduce human guides’ load gradually through collecting text dialogues.

In this chapter, we propose a framework to construct task-oriented conversational agents by the Wizard of Oz (WOZ) method [Fraser91], which is a dialogue-collecting method, and the incremental-learning technology. We also propose an algorithm which reduces recomputation costs against increasing examples by caching what states are merged in the previous learning session. This chapter shows a system for supporting development of these agents. The features of this system are (a) finite-state machine (FSM) based dialogue models learned by an incremental probabilistic DFA learning algorithm, (b) annotations for putting meanings to utterances and accessing information associated to Web pages, and (c) an interface character with a text and voice interface. We also confirm the effect of the system and the learning algorithm from the actual development of a tour-guide agent in Kyoto, Japan.

Section 3.2 describes the framework of developing conversational Web
agents, Section 3.4 shows the architecture of the support system, and Section 3.5 reports the empirical evaluation of the experimental system.

3.2 Wizard of Oz Method for Conversational Agents

The Wizard of Oz [Fraser91] is a method to simulate a dialogue system under development. In this method, a human who pretends of a system (called wizard) talks with a user using a microphone, a synthetic voice, a text interface via a network, and so on. The wizard joins the conversation as a system instead of a natural person because a human-human dialogue should not be applied to human-computer dialogue interface. Since the user thinks the wizard as a real system, the dialogue in the WOZ situation is nearer to the conversation between a user and a system than the normal conversation between humans [Carroll88]. So far, the WOZ method is used in the spoken dialogue system area, and it is also used in the human-computer interaction area for simulating intelligent agents such as instructible agents [Maulsby93].

The WOZ method is proper to simulate the human-computer interaction on a not-fully developed system, i.e., it is proper to confirm the behavior of a system before development and to confirm a partially developed system with complementing functions which are not developed. It also allows the system designer to observe user’s operation with a partially developed system, of which missing services are supplemented by the wizard [Itou99].

The original WOZ method is used for only simulations, and the designers and developers need to implement the actual system based on the result of the simulations. The dialogues used for the prototyping are not fully reflected in the final system.

We believe that we can construct task-oriented conversational agents which meet the tasks and scenarios gradually from example dialogues when we apply the learning technology to the WOZ, and the system absorbs and uses the example dialogues collected by the WOZ well.
Figure 3.1: Conceptual process of constructing agents

Learning Process

Figure 3.1 shows the conceptual process of constructing an agent. In this framework, a conversational agent is gradually constructed through a cycle of collecting example dialogues and learning dialogue models with an incremental learning algorithm. The process is as follows:

1. The wizard collects dialogues with the WOZ method. Only the first step is the same as the usual WOZ method, in which the conversation is actually done between humans. The wizard annotates each utterance to give sufficient information for the learning algorithm.

2. The system learns a small dialogue model from the first example dialogues. The system cannot infer many appropriate utterances from this model.

3. The wizard collects more dialogues with the WOZ and system inference. In this situation, the system shows some candidate answers. If at least one of them is suitable, the wizard selects it; otherwise, he/she inputs his/her own utterance.

4. The system learns a larger dialogue model than the previous model.

5. Through the repeat of this cycle, a conversational agent is constructed incrementally.
This framework includes the following two features:

- Example-based development
  Instead of dialogues which developers guess, dialogues from actual talks between each user the wizard are used for the system. An appropriate situation causes users’ situated actions [Suchman87], though ambiguous situations or purposes may cause digressions. In the book [Reeves96], the range of actions which a person does is limited if the situation is established clearly.

  When a situated scenario is prepared, the example-based approach is suited to the construction of task-oriented conversational agents.

- Development by cooperation between human and machine
  The role of the wizard is to assist the learning process.

  At the beginning, a system with a few example dialogues has little intelligence, and the wizard has a large role of the agent. As the system progresses, the role of the wizard is replaced by the system, and the system can be evolved to the real interface agent. Therefore, we consider the WOZ as a cooperative method by the human and the system for learning interface agents.\(^*\)

The agent developed through this method is a task-oriented agent because it is constructed from actual dialogues on the task with users and it talks with another users on the same task. Moreover, this agent is also wizard-oriented as well as task-oriented. It is because agents by different wizards behave in different ways even if they treat with the same task.

For constructing a learning conversational agent, the following elements are needed:

- An environment and a method for collecting example dialogues of good quality
  First, we need to design the task or scenario. In this work, we should

---

\(^*\)In the context about the WOZ, both the wizard and the system play the role of the agent.
give a role to the agent, and follow the guidelines in each phase of the
dialogue [Isbister01].

Then, we need to use a developing environment. We will show our
agent-developing environment in Section 3.4.1.

• An internal model and an incremental learning mechanism of which
the agent consists

There are many approaches for dialogue models. For example, there
are (a) statistical models such as Markov Models, Hidden Markov
Models, and n-gram models, (b) finite-state machine (FSM) based
models, and so on.

In this chapter, we use an FSM-based dialogue model because an FSM
is suitable for visualizing its dialogue flow. It is difficult for the devel-
opers to understand the learned model in progress. Moreover, some
constraints from the scenario or the task are learnable by using knowl-
dge structures or keyword scoring methods. In this framework, an
incremental algorithm is desirable to reduce the cost of learning. The
detailed description is shown in Section 3.4.2.

3.3 Dialogue-model Learning

3.3.1 Preliminaries

A Probabilistic DFA (PDFA) is defined as a five-tuple \( (Q, \Sigma, \delta, q_0, \gamma) \) where
\( Q \) means a finite set of states, \( \Sigma \) means a finite set of alphabet symbols, \( \delta \)
means a transition function \( Q \times \Sigma \rightarrow Q \), \( q_0 \) means the initial state, and \( \gamma \)
means the next symbol probability function \( Q \times \Sigma \cup \# \rightarrow [0, 1] \) (# means the
end of string symbol).

In each state \( q \), the following equation is satisfied:

\[
\sum_{a \in \Sigma \cup \{\#\}} \delta(q, a) = 1
\]
Figure 3.2: Example of PTA

A tree-formed PDFA in which the initial state is considered as a root is called a prefix tree acceptor (PTA). For example, if there are five inputs \{aa, baa, baa, bba, bbb\}, the PTA is shown as Figure 3.2. Each number \( s \) in a circle means state \( s \). The numbers are named in the lexicographic breadth-first order. In a tuple \([m, n]\) below each circle, \( m \) means the number how many times input strings passed the state (we define \( c(s) \) as this count), and \( n \) means the number how many times inputs finished at the state (we define \( c(s, \#) \) as this count). Each double circle means that the state is an accepted state. Each symbol \( a[l] \) above a transition means that an alphabet \( a \) passed the transition \( l \) times (we define \( c(s, a) \) as this count).

In learning a PDFA with the state merging method, a PTA which can accept example inputs is made first, then a general PDFA is constructed by merging pairs of two states.

The Kullback-Leibler divergence (or relative entropy) \( D(A||A') \) between two PDFA \( A = (Q, \Sigma, \delta, q_0, \gamma) \) and \( A' = (Q', \Sigma', \delta', q_0', \gamma') \) is defined as the following expression [Carrasco97]:

\[
D(A||A') = \sum_{q_i \in Q} \sum_{q'_j \in Q'} \sum_{a \in \Sigma \cup \{\#\}} c_{ij} \gamma(q_i, a) \log \frac{\gamma(q_i, a)}{\gamma'(q'_j, a)}
\]

where \( c_{ij} \) means the probability mass in \( A \) of the example inputs of prefixes common to state \( q_i \) in \( A \) and \( q'_j \) in \( A' \).
Section 3.3.2 and Section 3.3.3 describe polynomial-time learning algorithms which learn PDFA with the state merging method.

### 3.3.2 General Form of Quadratic Algorithms

Merging of state $s_1$ and state $s_2$ means folding two subtrees of a PTA in which $s_1$ and $s_2$ are the roots of these subtrees (Figure 3.3).

Procedure $\text{merge}(A, s_1, s_2)$ describes the process (Figure 3.4).

Figure 3.5 shows the general form of quadratic algorithms by lexicographic breadth-first order.

The quality of each algorithm changes according to the compatibility criterion (b) in Figure 3.5. If needed, preprocess (a) is used.

In the ALERGIA algorithm, there is no preprocess corresponding to (a), and the compatibility measure

\[
\text{compatible}(s_i, s_j, \alpha_H)
\]

is used for (b). It returns $true$ if the Hoeffding bound [Hoeffding63] is satisfied for the common suffixes of two states $s_i$ and $s_j$. $\alpha_H$ is a parameter. The condition is described as follows:

\[
\left| \frac{c(s_i, a)}{c(s_i)} - \frac{c(s_j, a)}{c(s_j)} \right| < \sqrt{\frac{1}{2} \ln \frac{2}{\alpha_H} \left( \frac{1}{\sqrt{c(s_i)}} + \frac{1}{\sqrt{c(s_j)}} \right)},
\]

21
input: PDFA $A$, state $s_1, s_2 (s_1 < s_2)$
output: merged PDFA

begin
  $c(s_1) \leftarrow c(s_1) + c(s_2)$
  $c(s_1, a) \leftarrow c(s_1, a) + c(s_2, a), \forall a \in \Sigma \cup \{\#\}$
  if both $\delta(s_1, a)$ and $\delta(s_2, a)$ for $a \in \Sigma$
    exist then
    merge($A, \delta(s_1, a), \delta(s_2, a)$) // merging of states recursively
  end if
  $A \leftarrow A - \{s_2\}$
  return $A$
end

Figure 3.4: Procedure merge

$\forall a \in \Sigma \cup \{\#\}$

$compatible(\delta(s_1, a), \delta(s_j, a), \alpha_H) = true, \forall a \in \Sigma$

In the MDI algorithm, the preprocess

$A' \leftarrow merge(A, s_j, s_i)$

is used for (a), and the compatibility measure

$$\frac{D(PTA(I+)||A') - D(PTA(I+)||A)}{|A| - |A'|} < \alpha_M$$

is used for (b).

$A'$ is a PDFA which is derived by merging of state $s_i$ and state $s_j$. $|A|$ and $|A'|$ are the numbers of $A$ and $A'$, respectively. $\alpha_M$ is a parameter.

### 3.3.3 Incremental PDFA Learning Algorithm

The algorithm shown in Section 3.3.2 assume that there is the whole learning set. When the number of data increases gradually, these algorithms should confirm all compatibilities again.
**input:** $I_+ // a positive example set

$\alpha // a precision parameter

**output:** PDFA // probabilistic DFA

**begin**

// $n$ means the number of states in $PTA(I_+)$
// $s_0, s_1, \cdots, s_{n-1}$ is sorted in the lexicographic
// breadth-first order of $PTA(I_+)$
$S \leftarrow \{s_0, s_1, \cdots, s_{n-1}\}$, $A \leftarrow PTA(I_+)$

**for** $i = 1$ to $n - 1$

**for** $j = 0$ to $i - 1$

(needed preprocess) \hspace{2cm} \cdots\cdots(a)

if the compatibility is confirmed then \hspace{2cm} \cdots\cdots(b)

$A \leftarrow$ merge($A, s_j, s_i$)

break

end if

end for

end for

**return** $A$

**end**

**Figure 3.5:** General form of quadratic algorithms by lexicographic breadth-first order

In this section, we try to reduce the cost of compatibility checks by storing pairs of merged states and a set of states which need to be recomputed.

**Recomputing all possible changes**

To extend a PDFA-learning algorithm to an incremental one, we need to manage the difference between the current PTA and the previous PTA, and states with the potential that the compatibility changes.

When the phenomenon that a state which was merged in the previous process changes not to be merged or that a state which was not merged in the previous process changes to be merged occurs (we call the state
Figure 3.6: Range of states which needs recomputation

*compatibility-changed state*, the following states have such kind of possibility:

1. States in a subtree of the PTA in which the compatibility-changed state is the root
2. States on a path from the root of the PTA to the compatibility-changed state (all predecessors of the compatibility-changed state in the PTA)
3. States acquired by applying (1) and (2) to the states which are (or were) merged with the changed state

Figure 3.6 shows the range of above processes. The function \( \text{effect}(s) \) returns these states when \( s \) is changed.

In general, the changes to apply a PDFA-learning algorithm to an incremental situation are described below.

There are two additional inputs and the initial values, which are the set of pairs of merged states in the previous learning

\[
\mathcal{M} \leftarrow \{(s_{01}, s_{02}), (s_{11}, s_{12}), \ldots, (s_{m1}, s_{m2})\}
\]

and the set of states which need recomputation

\[
\mathcal{R} \leftarrow \{s_{r1}, s_{r2}, \ldots, s_{rm}\}
\]
where each \((s_{k_1}, s_{k_2})\) corresponds to \(merge(A, s_{k_1}, s_{k_2})\) called directly from the algorithm in the previous learning, and each \(s_{r_k}\) is the state corresponding to a prefix of added examples.

The changes about the process to confirm compatibilities and the post-process are as follows:

- When \(R\) does not include the target two states, that is, there is no difference from the previous learning, they are merged if \(M\) includes the pair. If \(M\) does not include the pair, this step is skipped.

- If at least one state belongs to \(R\), the compatibility is calculated.
  - If the target two states are compatible, two states are merged. Then, \(effect(s)\) for these two states and the merged state are added to \(R\).
  - If the target two states are not compatible and the pair is in \(M\), i.e., the two states become to be not compatible, two states and \(effect(s)\) for them are added to \(R\).
  - If the target two states are not compatible and the pair is not in \(M\), i.e., there is no change, this step is skipped.

This means that the decision of compatibility involving a state which needs recomputation follows

- the previous result when the compatibility is not changed, or

- the result after the renewal of \(R\) and recomputation when the compatibility is changed.

Figure 3.7 is a procedure that above renewing process is added to the state merging procedure described in Figure 3.4, and Figure 3.8 is an algorithm that above renewing process is added to the algorithm described in Figure 3.5.
input: PDFA $A$
states $s_1, s_2 (s_1 < s_2)$
// a set including pairs of merged states
$M \leftarrow \{(s_{01}, s_{02}), (s_{11}, s_{12}), \cdots, (s_{m1}, s_{m2})\}$
// a set including states which need recomputation
$R \leftarrow \{s_{r_1}, s_{r_2}, \cdots, s_{r_m}\}$

output: merged PDFA

begin
  $c(s_1) \leftarrow c(s_1) + c(s_2)$
  $c(s_1, a) \leftarrow c(s_1, a) + c(s_2, a), \forall a \in \Sigma \cup \#$
  if $s_2 \in R$ then
    // because state $s_2$ which will be merged is in $R$,
    // add state $s_1$ which includes state $s_2$ to $R$
    $R \leftarrow R \cup \{s_1\}$
  end if
  $R \leftarrow R - \{s_2\}$
  remove $(s_k, s_2)$ except for $(s_1, s_2)$ from $M$
  if both $\delta(s_1, a)$ and $\delta(s_2, a)$ exist for $a \in \Sigma$ then
    merge($A, \delta(s_1, a), \delta(s_2, a), M, R$) // merging recursively
  end if
  $A \leftarrow A - \{s_2\}$
  return $A$
end

Figure 3.7: Procedure incremental_merge with renewal of recomputation set
input: $I^+ // $ positive examples (including increased examples)  
$\alpha // $ a compatibility parameter  
// a set of pairs of merged states in the previous execution  
$M \leftarrow \{(s_{01}, s_{02}), (s_{11}, s_{12}), \ldots, (s_{m1}, s_{m2})\}$  
// a set of states which need recomputation  
// each state corresponds to a prefix of added examples  
$R \leftarrow \{s_{r_1}, s_{r_2}, \ldots, s_{r_m}\}$

output: PDFA // probabilistic DFA

begin  
// $n :$ the number of states in $PTA(I^+)$  
// $s_0, \ldots, s_{n-1} :$ prefixed of $I^+$ sorted in the lexicographic breadth-first order  
$S \leftarrow \{s_0, s_1, \ldots, s_{n-1}\}, \quad A \leftarrow PTA(I^+)$  
for $i = 1$ to $n-1$  
for $j = 0$ to $i-1$  
if $s_i \in R$ or $s_j \in R$ then  
(needed preprocess)  
if the compatibility is confirmed then  
$A \leftarrow incremental\_merge(A, s_j, s_i, M, R)$  
$R \leftarrow R \cup effect(s_i) \cup effect(s_j)$  
break  
else  
if $(s_j, s_i) \in M$ then  
$M \leftarrow M \setminus \{(s_j, s_i)\}$  
$R \leftarrow R \cup \{s_i, s_j\} \cup effect(s_i) \cup effect(s_j)$  
end if  
end if  
else if $(s_j, s_i) \in M$ then  
$A \leftarrow merge(A, s_j, s_i)$ // normal merging  
break  
end if  
end if  
end for  
end for  
return $A$

end

Figure 3.8: The general form of incremental quadratic algorithms
Recomputing only changed states  In above algorithm, when merging operations which involve states near the initial state (or the root of the PTA) occur, almost all states are included in the recomputation set. Therefore, the cost of checking compatibilities may not reduce.

However, the recomputation occurs in various states of the PDFA when the number of learning data increases gradually. It means that almost same results in quality can be acquired even if we do not consider all possible changes.

The algorithm which merges only changed states is what the two lines (*) and (**) are removed from Figure 3.8.

The algorithm works as follows:

1. A tree-formed PDFA (called prefix tree acceptor or PTA) \(PTA(I_+)\) is constructed from example sequences \(I_+\).

2. For each pair of two states, except for states not in the recomputation set \(\mathcal{R}\), in the lexicographic breadth-first order, a PDFA \(A'\) in which these states are merged from the current PDFA \(A\) is constructed. If the new FSM has non-deterministic states, these states are also merged recursively by the function \(merge\) (Figure 3.9 shows an example of the merging method).

3. The Kullback-Leibler divergence between \(A\) and the PTA \(D(PTA(I_+)||A)\) is computed, and the divergence between the \(A'\) and the PTA \(D(PTA(I_+)||A')\) is also examined. Then, the average increase of Kullback-Leibler divergence per reduced state is calculated by the following expression:

\[
\frac{D(PTA(I_+)||A') - D(PTA(I_+)||A)}{|A| - |A'|}
\]

If this value is less than the precision parameter \(\alpha\), the original PDFA and the merged PDFA are considered as compatible.

4. If the two PDFAs are decided as compatible, the merging function \(incremental\_merge\) is adopted and the merged PDFA is used in the next steps. In this process, these states are added to \(\mathcal{R}\).
5. In the step 2., if the both states are not in \( R \), the decision follows the previous learning session.

6. After the repetition of above steps, a PDFA is constructed. It means the generalized PDFA.

### 3.4 WOZ-based Agent System

We implemented a support system with the mechanism described in Section 3.2. This system is used for developing conversational agent treating with simple tasks easily.

This section describes the system architecture, the user interface, the dialogue model and learning algorithm, and the knowledge and topic management.

#### 3.4.1 System Architecture

The system architecture is shown as Figure 3.10. In our implementation, each client is a Java applet which works on Web browsers. The other components are Java programs which work on many platforms. We also used the Microsoft Agent\(^\dagger\) for interface characters.

\(^\dagger\)http://www.microsoft.com/msagent/
Each component of the architecture has the functions and contents as below:

**User-side Client** The User-side Client (Figure 3.11(a)) is a text-based chat system which is expanded with an interface character. The agent talks both with text and speech. When the agent talks about a topic, Web pages related to the topic will be shown in the user’s Web browser.

**Wizard-side Client** The Wizard-side Client (Figure 3.11(b)) is the extended version of the User-side Client for the WOZ method. It has two additional features. The wizard can input utterances by selecting an utterance from inferred answers by the system or preset utterances from the menu. The wizard can also put two kinds of tags on utterances so that the system could handle dialogues easily. One is an
Figure 3.11: Screenshot of each client
utterance tag on each utterance by a user or the wizard. Each utter-
ance tags are designed based on the task. The other is an inner tag
which is associated to a keyword or a URL related to the content.

**WOZ Interface** The WOZ Interface controls the dialogue-data stream
among each client, the Learner, and the Inference Engine. When the
Wizard-side Client is disconnected (when broken lines are removed
from Figure 3.10), the utterance inferred by the system is directly
sent to the user as the answer. Therefore, the whole system works as
an individual agent system.

**Example Dialogue** The Example Dialogue has the collected dialogues.
Each utterance is annotated by the wizard. From these dialogues,
dialogue models are learned.

**Dialogue Model** The Dialogue Model has the current FSM and all utter-
ances of which the FSM consists. The FSM is constructed by the
Learner, and referred by the Learner and the Inference Engine.

**Learner** The Learner constructs an FSM from annotated dialogues in the
Example Dialogue, and updates each keyword's score of utterances in
the WOZ condition according to the utterances from the WOZ Inter-
face.

**Inference Engine** The Inference Engine infers next utterance from an input
from a user. Then, it sends the utterance to the Wizard-side Client in
the WOZ condition, otherwise it sends the best candidate to the User-
side Client. If an utterance contains annotations related to information
which are to be shown, the concrete sentences from the Knowledge
Base are implanted in the utterance.

**Knowledge Base** The Knowledge Base contains the knowledge which is
presented to users as a topic. It consists of URLs, sentences of con-
tents, some additional information.
**Knowledge Server** The Knowledge Server checks whether words in a user’s utterance exist in the Knowledge Base. If there is one or more information, this component extracts the contents.

### 3.4.2 Dialogue Model

We use FSMs for the dialogue models of WOZ-based conversational agents. There are some FSM-based systems. For example, the VERBMOD-BIL project uses a layered architecture including FSM [Alexandersson95]. CommandTalk [Stent99] uses the combination of small FSMs. We try to add the learning functionality to FSMs for dialogue models.

We consider a dialogue as a series of utterances. Each utterance consists of the following components: (a) the speaker (user or agent), (b) the content input by the speaker, and (c) an utterance tag (decided based on the task design).

An FSM constructed from dialogues as series of utterance tags means the dialogue flow in the detail based on the tag design. Each tag has the keyword scores based on utterances.

We consider an algorithm which learns cyclic automata and treats with noises or errors in a learning set. Thus, we choose a cyclic Probabilistic DFA (PDFA) learning algorithm with a high performance for learning FSMs. In this area, the **ALERGIA** algorithm [Carrasco94], which is originally used for the grammatical inference area, is a famous algorithm. The **MDI** algorithm [Thollard00] has a higher performance than the ALERGIA. However, there are not any algorithms which treat with increasing data or incremental situations.

In this section, we use an incremental algorithm derived from the MDI, which is described in Section 3.3.3.

In this chapter, each utterance tag means an FSM symbol, the learned FSM means a dialogue model, and the original example dialogues mean the plausible sets of a user’s inputs and the agent’s outputs. Figure 3.12 shows the dialogue structure.

We introduce the keyword score of each words to recognize the user’s
Figure 3.12: Dialogue model and example dialogues

utterance. When a user inputs an utterance, the nearest example utterance is calculated by the simple dot product of word score vectors.

3.4.3 Knowledge and Topic Management

We collect knowledges from the Web. There are many activity of the Semantic Web [Berners-Lee01], and we will be able to use a lot of information in future. Currently, there are few pages to which we can apply this technology. Therefore, we put some annotated informations on our knowledge base made with an annotation tool (Figure 3.13), and we use the information for each user’s requests. Each information is searched by keywords or categories.

Each content has the following annotated information.

- name of the content
- aliases of the content
The agent system appropriately answers to each user’s request with this topic-management structure. In this system, the guidance contents and categories are managed. There are three situations for a tour-guiding task. Constraints in each situation is show as follows:

1. Direct request from a user
   When a user directly requests a tour spot as “tell me about the Kinkakuji temple,” the system simply shows the information about the Kinkakuji temple. At this time, “the Kinkakuji temple” is set as the current topic, and “temple” is set as the current category. Then, if
the user requests some additional information as “tell me how to go there,” the system shows the information about the transportation of the current topic “the Kinkakuji temple.”

2. Indirect request from a user
When a user indirectly requests a tour spot as “recommend me a temple,” the system shows a temple which belongs to the “temple” category. In the case of requests such as “tell me another temple” after a guidance of a temple, the system shows a tour spot which belongs to the category “temple.”

3. Suggestion from the agent
When an agent suggests a tour spot as “Can I show a temple?”, the “temple” is set as the current category. When the user accepts the suggestion such as “Yes, please,” the system shows a tour spot of the current category.

### 3.4.4 System Flow

This section describes the system flow.

The following items are needed for constructing a conversational agent of a specific task:

- the task or scenario
- utterance tags which the agent needs for following the scenario (they should be domain-specific tags because the agent is a domain-specific or task-oriented agent)
- Web sites used for guidance and annotated additional information
- preset utterances used by the wizard (they are not necessary, but they reduce the wizard’s cost in the task)

The agent system works as follows:
1. When a user inputs an utterance, the Inference Engine infers the candidate answers, and sends them to the Wizard-side Client via the WOZ Interface. For example, a user inputs “I want to see the Ryoanji temple,” when the current state is 2 in Figure 3.12, then the system searches the pair of a user’s utterance and an wizard’s utterance of which state transitions can be done from the current state with the utterance tags. For each user utterance, the product of the keyword vectors are calculated, and the utterances with the highest scores are shown to the wizard. In Figure 3.12, an utterance “I want to go to the Kinkakuji temple” is selected with the highest score, and then, if the keywords in the utterance are in the Knowledge Base, the actual content is changed to the current topic. In this case, the keyword Kinkakuji temple” is selected with the highest score, and then, if the keywords in the utterance are in the Knowledge Base, the actual content is changed to the current topic. In this case, the keyword “Kinkakuji temple” is changed to “Ryoanji temple,” and the contents about the Ryoanji temple becomes the candidate answer.

If there is not the wizard, the ‘best’ candidate with the highest score is sent to the user.

2. When the wizard selects a candidate answer, the inferred utterance and its utterance tag are sent to the Inference Engine. If there are not any proper candidates, the wizard selects an utterance from the preset utterances or inputs one, and annotates both the user’s utterance and the agent’s utterance. The wizard can add a URL to the utterance if the wizard wants to show an Web page to the user. Finally, the state transition is executed according to the utterance tags.

3. In any cases, the utterance is spoken by the interface agent. If an Web page is associated with the utterance, the page is shown on the User-side client.
Table 3.1: Utterance tags used in the tour-guide task in Kyoto (TS means ‘tour spot’)

<table>
<thead>
<tr>
<th>user utterance tag</th>
<th>agent utterance tag</th>
</tr>
</thead>
<tbody>
<tr>
<td>greeting</td>
<td>open</td>
</tr>
<tr>
<td>response</td>
<td>close</td>
</tr>
<tr>
<td>request for TS</td>
<td>self introduction</td>
</tr>
<tr>
<td>ambiguous request for TS</td>
<td>guide TS</td>
</tr>
<tr>
<td>question about TS</td>
<td>suggest TS</td>
</tr>
<tr>
<td>reject leading to another TS</td>
<td>impossible to guide</td>
</tr>
<tr>
<td></td>
<td>answer for question</td>
</tr>
<tr>
<td></td>
<td>lead to another TS</td>
</tr>
<tr>
<td></td>
<td>lead to closing</td>
</tr>
</tbody>
</table>

3.5 Experimentation

We constructed a tour-guide agent in Kyoto, and examined the learning cost and the changes of the wizard’s load.

The scenario is as follows: First, the agent introduces himself. Second, he confirms the user’s request or suggests a tour spot. When a tour spot is decided, he shows the tour spot and explains the contents. After the explanation, the agent leads the user to another tour spot. If there is no request or the user rejects the leading, he closes the dialogue.

Table 3.1 shows all utterance tags used in this scenario. Each tags are defined according to the scenario phase.

We examined how much the quality of the agent’s inference increased as the number of example dialogues increased.

As the example dialogues, we used Japanese example dialogues about Kyoto tour guide collected by the NEDO for Senior Support System Project. 100 dialogues (2198 agent utterances and 799 user utterances) are collected in the Kyoto tour-guide task by the WOZ method. We used 31 dialogues for the learning data, and 10 dialogues for the test data.
Table 3.2: Ratio of each tag of user utterance

<table>
<thead>
<tr>
<th>tag of user utterance</th>
<th>number</th>
<th>ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>request for TS</td>
<td>384</td>
<td>48.1%</td>
</tr>
<tr>
<td>question about TS</td>
<td>136</td>
<td>17.0%</td>
</tr>
<tr>
<td>meaningless response</td>
<td>101</td>
<td>12.6%</td>
</tr>
<tr>
<td>reject leading to another TS</td>
<td>98</td>
<td>12.3%</td>
</tr>
<tr>
<td>ambiguous request for TS</td>
<td>36</td>
<td>4.5%</td>
</tr>
<tr>
<td>accept suggestion</td>
<td>36</td>
<td>4.5%</td>
</tr>
<tr>
<td>others</td>
<td>8</td>
<td>1.0%</td>
</tr>
</tbody>
</table>

The features of this dialogues are as follows:

- Users are university students in Kyoto area.

- Each dialogue includes not only usual written language but also colloquial language which will be used in informal text chats. There are various style of utterances from users.

- In some dialogues, the user requests another tour spots such as “Could you show me another spots?” Many of them are requests of tour spots in the same categories supported our system such as temples or shrines, and the others includes another categories such as “Tell me another things which is related to Sakyo area.”

- There are two kinds of dialogues, one kind of dialogues includes many omissions or colloquial words, and utterances with which can not be treated by the system such as requests about transportation expenses from somewhere or questions about contents in the Web sites which are shown to users. The other includes many direct requests.

Figure 3.14 shows an example dialogue.  
Table 3.2 shows the ratio of each tag of user utterance in these dialogues.
A. (means agent) Hello, I’m Kyoto tour guide, Peedy.
   Nice to meet you. (open)
A. I was a guide of ... Where do you want to go sightseeing?
   (self introduction)
U. (means user) Please tell me Nijo Castle. (request TS)
A. Nijo Castle is ... (guide TS)
   (A Web page related to Nijo Castle appears in user’s browser)
U. I see. (response)
A. Do you want to go to another place? (lead to another TS)
U. I want to buy my souvenir. (request TS)
A. Can I introduce a famous shrine in Kyoto? (suggest TS)
U. Yes. (request TS)
A. I’ll introduce Kamo-Betsurai shrine in Kita-ku. ...
   (guide TS)
U. I see. (response)
A. Do you want to go to another place? (lead to another TS)
U. Please tell me some shops where I can buy my souvenir. (request TS)
A. How about the paper shop Kakimoto? (guide TS)
   (A Web page related to the shop in user’s browser)
A. There are many Japanese-paper shop in Teramachi street. (guide TS)
      ......
A. How about today’s guide? Is it good? (lead to closing)
A. See you. (close)

Figure 3.14: Example dialogue (the original dialogue is in Japanese)
3.5.1 Learning Cost

In this section, we examined the following two items:

1. the quality of the learned model
2. the number of compatibility checks

We used 100 dialogues of 110 dialogues for the learning set, and the other 10 dialogues for the test set. We examined the number of compatibility checks and the quality of learned dialogue models for a normal PDFA-learning algorithm, an incremental algorithm with \( \text{effect}(s) \), and an incremental algorithm without \( \text{effect}(s) \) which recomputes only compatibility-changed states.

We used the MDI algorithm as the target PDFA-learning algorithm.

The number of comparisons for compatibility

Figure 3.15 shows the number of decisions for the merging method. In this evaluation, we used \( \alpha_M = 0.03 \) for the parameter.

From Figure 3.15, the algorithm with \( \text{effect}(s) \) does not reduce the recomputation cost. It is because when a merge involves a state near the initial state, almost all states are included in \( \text{effect}(s) \), therefore the algorithm works almost same as the usual MDI.

On the other hand, the algorithm without \( \text{effect}(s) \) reduced the number of comparison 33.0% in average and 21.3% in total.

Perplexity

We used the test set perplexity \( PP \) to examine the quality of learned models. It is given by \( PP = 2^{LL} \).

When \( n \) strings \( S = \{s_1, \ldots, s_n\} \) (each \( s_k \) consists of series of \( m_k \) symbols \( x_1^k \cdots x_{m_k}^k \)) are input into the PDFA, the per symbol log-likelihood of strings
Figure 3.15: Comparison of the count for decision of merging
is denoted by:

$$LL = -\frac{1}{||S||} \sum_{i=1}^{n} \sum_{j=1}^{m_i} \log P(x_j^i|q_j^i)$$

where $P(x_j^i|q_j^i)$ denotes the probability (weight) that $i$-th symbol $x_j^i$ in the string $s_i$ will be input in state $q_j^i$. $||S||$ denotes the number of symbols in all strings.

When the transition probability $\delta(q_j^i,x_j^i)$ is used as $P(x_j^i|q_j^i)$, the perplexity cannot be measured if there is an input which is not in the transition. Therefore, we also use the simple occurrence probability (unigram) of each symbol in the test data.

When $P(x)$ denotes the occurrence probability of symbol $x$ in all inputs, $P(x_j^i|q_j^i)$ is denoted as

$$P(x_j^i|q_j^i) = \beta \delta(q_j^i,x_j^i) + (1-\beta)P(x_j^i)$$

where $\beta (0 \leq \beta \leq 1)$ is a parameter. We used $\beta = 0.5$ for the parameter.

Figure 3.16 shows the comparison of the perplexities between the MDI and the incremental algorithm which recomputes only renewed states.

From Figure 3.16, we found the algorithm without $effect(s)$ does not make perplexities worse so much.

As the result, the method which manages only states in which the decision of merging changed reduced the number of comparisons without increase of perplexity, while the algorithm which includes $effect(s)$ in the set of states which need recomputation does not perform more than the usual algorithm.

### 3.5.2 Wizard’s Load

We also examined how much the quality of the agent’s inference improved as the number of example dialogues increased.

We used the same dialogues in Kyoto tour guide task. We used 100 dialogues for the learning data, and 5 dialogues (including 114 agent utterances and 79 user utterances) for the test data.
Figure 3.16: Comparison of perplexities

Figure 3.17 shows the change of the ratio of interventions by the wizard. The horizontal axis means the number of learned dialogues (1, 20, 60, and 100 dialogues). The vertical axis means the ratio of the wizard’s interventions to candidate answers from the system (0 means the system infers all utterances correctly, and 1 means the wizard inputs all utterances). In Figure 3.17, there are two graphs. One is the ratio when the system infers only one candidate. The other is the ratio when the system infers two candidates, and the inference is considered as correct if one of the candidates is right.

From Figure 3.17, we found that the ratio of the wizard’s modification to inferred utterances by the system with the incremental algorithm without effect(s) is almost same as that with the original MDI algorithm. In each utterance, the worst case is the ratio with the incremental algorithm is worse by 6.3% than that with the MDI. This difference means one utterance per example dialogue, and the cost for human is small.
In general, the load of the wizard decreased as the learning process made progress, and the result almost converge when the dialogue model is learned from 100 example dialogues. It means that the task in this evaluation is simple, and the dialogue flow is learned from these examples.

On condition that the system infers two candidates, the quality is continuously increasing. It implies that we can improve the system if we consider the scoring method of utterance recognition and the process of the narrowing candidates.

From each result, we found that the system can decrease wizard’s load as the number of dialogues increase for dialogues which include various phrases such as the colloquial language. In the case that the ratio of the dialogues which request tour spots directly, the system can infer appropriate answers at the beginning of the learning. However, increasing the number of dialogues causes confusions that the system infers answers with another
We also found that for some kind of utterances in which keywords related to tour spots is not included and users use different words such as acceptances such as “Yes, please,” refusals such as “No, thank you,” meaningless responses such as “uh-huh,” the increase of dialogues improves the system inference. However, some meaningless responses include noisy words and caused inference errors. As to utterances requesting another spots in the same categories, some utterances containing specific words caused good inferences, and other utterances caused errors.

Figure 3.18 shows learned dialogue models in the learning process, and Figure 3.19 shows the model after learning 100 dialogues. This 100-dialogue model includes 17 states and 73 transitions. Though there are many transitions, some states and transitions express some parts of the scenario.

3.6 Summary

In this chapter, we proposed a framework to construct a learning conversational agent from example dialogues.

For collecting the example dialogues, an WOZ method enhanced by a machine learning technology is used. For the dialogue model of the agent, we used an FSM constructed by the incremental algorithm derived from a PDFA-learning algorithm with the state-merging method. We proposed two methods for reducing the recomputation cost of PDFA-learning algorithms in the situation that collecting examples takes much cost and the number of examples increases gradually. In these methods, the pairs of merged states and states which need recomputation are cached.

We evaluated the reduction of costs from both aspects of human’s cost and computational cost.

From the simulation with tour guide dialogues, we found that

- the method which manages all states which need recomputation did not performed well, and
Figure 3.18: Change of dialogue models
Figure 3.19: Dialogue model after learning 100 examples

- the method which manages only compatibility-changed states performed reduction of the number of compatibility checks without increase of perplexity.

We also applied the algorithm to an actual Web-based conversational agent, and we confirmed that the cost to input example dialogues and to learn the dialogue models were reduced. We think the tour-guide agent will be applicable in Digital City [Ishida02b].

This kind of methods which reduces the recomputation cost are useful for domains in which it is hard to construct FSMs in advance and they are
constructed gradually from actual examples.
Chapter 4

Communication Environment for Information Sharing

4.1 Introduction

This chapter describes the design and implementation of two communication-support systems which deal with sharing information both in the real world and in remote locations, and describes the evaluation of these systems through experimental use of the systems.

Section 4.2 describes the general features of a communication environment with a large screen. Section 4.4 describes the design, implementation, and evaluation of Silhouettell [Okamoto98][Okamoto99], which supports cross-cultural communication in the real world, and Section 4.5 describes Networked Silhouettell, which supports cross-cultural communications between both local and remote places [Okamoto02c].

4.2 Supporting Communication with a Large Screen

There are three general concepts to design communication tools [Isbister99]. They are also needed to design tools for supporting commu-
Conversational common ground
Conversational common ground is a process by which conversation partners create a shared awareness of what knowledge and assumptions they have in common.

Nonverbal cues in social interaction
Nonverbal cues are signals, which are not in words, that express the emotions of a person, the general qualities such as personalities, the attitude a person has toward the topic other people are speaking, the attitude toward the listener, the relationship with the listener, and so on.

Situation and role expectations in social interactions
Characteristics of a situation help people understand its social context, which then helps them determine which actions and roles are relevant and appropriate.

The systems described in this chapter enable the concepts above by incorporating the following features:

- Showing shared information based on participants’ profiles and cultural background. (common ground)
- A life-sized interface with a large screen. (situation expectations as well as nonverbal cues)
- Displaying participants in the conversation using real images or shadows. (nonverbal cues as well as common ground)

In the next three sections, we will discuss each of these features.

4.2.1 Showing Shared Information
A common feature shared by most of the current communication systems supports conversation by providing a virtual conference room, either on a
desktop monitor or a large screen. Desktop systems, such as CU-SeeMe typically create a conference room by displaying all participants on the screen. FreeWalk [Nakanishi96] also provides a 3-D virtual space for casual meetings. Large screen systems, such as MAJIC [Okada94], connect two or more locations to create conference rooms shared by people who are in the remote locations.

Unlike the systems mentioned above, the two systems described in this chapter focus on supporting conversation contents. There are two approaches to support them. One is a method to enable users to understand one another directly, using tools such as interpretation. The other is, as we propose, a method to tell “who he* is” by presenting the user’s information. These systems provide additional information about cultural characteristics or differences of users to their conversation partners as shared information, projecting topics related to them on-screen. In particular, it is important to consider the following two effects due to share information related to users:

1. The increase of interaction by knowing common information
   At the early stage of communication, people begin conversations by finding their (many kinds of) common interests. It is more difficult to find cues in cross-cultural communication than in communication among people from the same culture. Thus, showing shared information is important in order to promote their communication.

2. The decrease of misunderstanding by recognizing differences among people
   In an international project including companies or research organizations from countries, people who do not know each other collaborate at the same place or from remote places. In such situations, troubles due to misunderstanding about their features or differences easily happen. Thus, it is important to tell their differences, especially cultural differences, in advance.

*“He,” “his,” and “him” should be read as “he or she,” “his or her,” and “him or her” throughout this paper respectively.

53
4.2.2 Large-sized Interface with a Large Screen

Using a large screen is more effective for supporting cultural communications than using a CRT monitor for the following reasons:

- Users can move more freely in a wider space
  If the system has a small display, users must use it in a correspondingly narrow physical space. A large screen allows users to play more freely with the system in a wider sphere, without worrying about where they are in relation to the screen.

- Users can be projected life-sized
  A life-sized, large projection space allows us to display information of users on the screen in front of each user. Providing information related to each user’s position in the real world is more natural than providing information at one location on the screen, as with a traditional desktop interface. In a large-screen environment, each user can see all participants’ information at a glance.

- Shared viewing is easier
  A large screen is especially appropriate for shared viewing, because all the participants can easily see the information on the screen. People who are far away from the screen can also see what is going on, which may draw them into joining the conversation.

- Transmitting nonverbal information is easier
  In meetings between remote locations, especially those that require cross-cultural communication, transmitting nonverbal information such as facial expressions or gestures is important. A large-screen system can make gestures and expressions more legible to everyone who is interacting, and create a more effective nonverbal communication.
4.2.3 Displaying Participants as Real Images and Shadows

In communication systems supporting meetings between remote locations, it is customary to project both conversation partners’ images on the screen. It is clear that showing real images of corresponding participants is useful in this case. We believe that it is also effective to show representations of users in local interactions, in order to attract them to use the system. The implementation form is decided according to the location which a system supports. In the systems discussed in this chapter, the following two display methods are used to attract and engage participants.

- **Local (face-to-face) system (Silhouettell):** displaying a shadow corresponding to each local participant on the screen in front of him.

- **Remote (over-network) system (Networked Silhouettell):** displaying only real images of all remote participants on the screen. Local participants see transparent shadows of themselves projected into the same space as remote participants.

A shadow in Silhouettell is a black shadow or silhouette, and a shadow in Networked Silhouettell is a transparent mirror image instead of a black shadow.

We consider that a user representation which is linked to his information is needed. Moreover, only one clear face (which means his “identity”) is needed per participant. A shared information is shown on the corresponding user’s real image. In the local environment, we can not show the information on the user. Therefore, we use his shadow which is natural representation on the real life, and the information is presented on the screen with the shadow. On the other hand, in the over-network environment, the information of a user is directly presented on the real video image of him. The implementation of the representation depends on the environment in which the system will be used.

We believe that for local interaction, it is easier for users to understand the use of shadows compared to other representation systems. Chatting sys-
tems often use avatars or icons. When using avatars or icons in a real world, however, users may not be able to map which icon relates to themselves and which to their partner—the avatars become a cumbersome additional identity. However, a shadow is a naturally co-present representation of oneself in a physical interaction. Thus, the correspondence between a participant and his shadow is much more natural and intuitive in a local interface. We could of course also simply display mirrored video of all participants. However, users might be uncomfortable about having all other participants “looking” at them all the time. A shadow is more subtle and natural, and does not duplicate the face-to-face interaction already taking place.

In the case of the over-network environment, a black shadow causes the problem that a person cannot see his partner because of overlapping when he stands in front of the partner. Therefore, we decide to use a transparent mirror image instead of a black shadow.

4.3 Cross-cultural Situation

Various electronic communication support systems have been studied and developed for supporting collaborations in companies or research activities. The computational aspect of such systems, e.g., system performance, as well as their impact in business offices or laboratories have been discussed extensively (for example, in the groupware research area, this aspect is considered well [Okada94]). However, the cultural aspect, especially in cases where people from heterogeneous cultures interact with each other, and aspects of the process of interpersonal understanding or socialization between co-workers before the actual task, have not been thoroughly examined. The number of the Internet users in East Asia is expected to increase to 300 millions. The Internet brings about great number of cross-cultural communications. This is, especially true in Japan, where many companies advance abroad and many foreign companies enter domestic markets. Therefore, opportunities for international collaboration and cross-cultural communication among people from heterogeneous cultures (e.g.,
manufacturers which have subsidiaries from different countries doing research and working locally, and research communities at international conferences) increase. Thus, it becomes more important and interesting to consider these aspects of electronic communication support. However, there are few Human-Computer Interaction researches in this area. In these situations, cross-cultural differences must be addressed to enable satisfactory communication and community-formation. We are studying this kind of communication or community-formation, named communityware or community computing [Ishida98a][Ishida98b]. This approach focuses on informal communication and socialization, which are earlier stages of collaboration; while groupware focuses on the collaboration itself. As internationalization in companies or research activities spreads, supporting cross-cultural communication is getting a larger research area in community computing.

There are five factors in community computing [Ishida98b]:

- Knowing each other
- Sharing preference and knowledge
- Generating consensus
- Supporting everyday life
- Assisting social events

Supporting socialization encourages people to know each other in five factors of community computing.

The research described in this chapter adopts the theoretical framework of community computing and extends it into supporting cross-cultural communication. The common concept of this research is “providing an environment which presents shared information or topics to people in the same space and from heterogeneous cultures.” Usual community-computing systems provide virtual spaces or interests among users. However, there are few trials which provide both of them.

By implementing the two systems, we aimed to provide the following two supports for users:
1. Support of mutual understandings by providing shared contents

2. Support of a mode of communication nearer to face-to-face communication

To support cross-cultural communication, we believe it is necessary to provide environments in which participants enjoy conversations that allow them to share one another’s background and profile visually. To satisfy both of these conditions, we propose systems with large screens that can display information shared by many participants.

We implemented two systems based on the above common concepts, named Silhouettell and Networked Silhouettell. The two systems are different systems which aim at the same purpose.

There are several differences among these systems.

- Location
  While Silhouettell supports only local communications, Networked Silhouettell supports both local and remote communications.

- Shared information
  Silhouettell uses web pages including common topics for shared information among users. Networked Silhouettell uses users’ cultural experience. In designing our two systems, we decided each content on the large screen based on two aspects described in Section 4.2.1. That is, we used common topics in Silhouettell because we believe a common topic is the most fundamental factor in community. In Networked Silhouettell, we tried to decrease misunderstandings by showing each user’s backgrounds and differences among users.

- Representation of shadow
  We implemented a shadow as a black silhouette in Silhouettell, and as a transparent image in Networked Silhouettell. The detailed description is shown in Section 4.2.3.

- Selecting on-screen information
  In Networked Silhouettell, the selecting operation for on-screen infor-
Table 4.1: Comparison between Silhouettell and Networked Silhouettell

<table>
<thead>
<tr>
<th>Locations of users</th>
<th>Silhouettell</th>
<th>Networked Silhouettell</th>
</tr>
</thead>
<tbody>
<tr>
<td>Representation</td>
<td>black shadows of all the participants</td>
<td>transparent images for the local users and video images for others</td>
</tr>
<tr>
<td>Shared information</td>
<td>web pages including common topics</td>
<td>each user’s cultural experience</td>
</tr>
<tr>
<td>Selection of information</td>
<td>not supported</td>
<td>supported</td>
</tr>
</tbody>
</table>

...mation is supported, while this operation is not supported in Silhouettell.

The comparison of these two systems is shown in Table 4.1. In the next sections, we try to confirm the effects from these points of view empirically through these systems.

4.4 Supporting Communication in the Real World

This section describes Silhouettell [Okamoto99][Okamoto98], a system that finds topics common to conversation participants, and displays web pages related to those topics on the large screen along with the participants’ shadows. The system provides awareness support, which augments the real world, to enable users to become aware of other users who have common interests and to make communities.

4.4.1 System Design

We believe that the following features are important for supporting and augmenting real-world communication:
Silhouettell uses a large screen to display users, their profiles and common interests. Figure 4.1(a) shows the concept of Silhouettell, and Figure 4.1(b) shows Silhouettell in use.

Silhouettell displays the shadows of the participants as objects on the screen with their profiles above their shadows so that they can identify one another. The system also enriches encounters and encourages conversation by presenting web pages as common topics. The connection lines between each topic and the shadows show groups interested in the same topic.

### 4.4.2 System Implementation

Figure 4.2 shows the system configuration of Silhouettell. It consists of two computers, an SGI ONYX and an SGI INDY, connected via Ethernet. The ONYX is connected to a large graphics screen. The INDY is connected to the output of a video camera.

Images from the video camera are processed by the Video Process in the INDY, and the results are sent to the Main Process in the ONYX. The Main Process generates the screen image from the profile stored in the Profile File, web pages, and the data from the Video Process.

The system works as follows. It generates the shadow of a user by
Figure 4.2: Silhouettell system configuration
comparing the current image to the background image, and distinguishes participants using their clothing color. It averages the value of the pixels corresponding to the user’s chest area and identifies the participants by the stored values (they are input at first use). The system also uses each user’s movement computed from two continuous frames. The use of both color and location reduces misidentification†. When a user is recognized, the system reads his profile (name, birthplace, and interests), which is registered in advance.

We use web pages for various topics. If there are common keywords among several users (they are found by string matching), the system searches web pages for selected keywords and displays the resulting pages between corresponding users on the screen (The system first searches pages registered by the system manager in advance and next searches using a general search engine). In addition, the system changes the size of the topic display windows to reflect the physical distance between users. When they are far from each other, the topic windows are quite small in size. As they approach each other, the topic windows increase in size. This is to encourage users to approach each other with the idea of confirming the topics listed.

**Example Use** Figure 4.3 shows an example of an encounter using Silhouettell.

1. One person shows up:
   User SAGAWA enters the media room, his shadow and profile are displayed on the screen (Figure 4.3(a)). The profile shows, from the top, that his name is SAGAWA Ryusuke, his affiliation is Kyoto University, and his interest is skiing.

†Though we used this method for the experimental prototype system, it is not sufficient for real-world applications (for example, if people change clothes, the system does not work well). We think that some kind of tag-based small device, such as Active Badge system [Want92] or Meme tags [Borovoy98], would be useful for more robust user-recognition in future versions.
Figure 4.3: Encounter with Silhouettell
2. Another person comes to the room:
   Next, another user (name: TANAKA Hironori, affiliation: Kyoto University, interest: Softball) enters the room. A WWW page related to the common topic, Kyoto University, is presented by Silhouettell (Figure 4.3(b)). Lines are drawn to link the page and the two shadows to indicate that the topic is shared by SAGAWA and TANAKA.

3. One approaches to another:
   They are motivated by the topic, and one approaches to the other. The WWW page is then magnified (Figure 4.3(c)). Consequently, conversation about the page occurs.

4. A third person joins:
   The third user (name: MATSUMOTO Hiroaki, affiliation: Kyoto University, interest: Softball) joins, as a result of finding two people talking to each other. The topic of Softball which is also shared by TANAKA is newly displayed, and the page about Kyoto University which is common to all of them moves to the center of the screen (Figure 4.3(d)).

4.4.3 Experimentation

We observed actual use of Silhouettell by people from various countries. This section describes the experiment and our observations and analysis about what happened.

Before the experimentation, we asked 5 Japanese and 25 non-Japanese in Kyoto University to answer a questionnaire to clear the situation to use this environment. The latter were from Europe, America, Asia, and Africa, and they are mainly research fellows or (under)graduate students. Table 4.2 shows the nationalities and their numbers.

The following features were obtained from the results of the questionnaire.

- Where would Silhouettell best be used?
  Table 4.3 shows the results. Over 70% of the subjects selected com-
Table 4.2: Rates of nationalities

<table>
<thead>
<tr>
<th>nationality</th>
<th>Japan</th>
<th>Asia, except Japan</th>
<th>Africa</th>
<th>Europe and America</th>
</tr>
</thead>
<tbody>
<tr>
<td>number</td>
<td>5</td>
<td>11</td>
<td>3</td>
<td>11</td>
</tr>
<tr>
<td>rate (%)</td>
<td>16.7</td>
<td>36.7</td>
<td>10.0</td>
<td>36.7</td>
</tr>
</tbody>
</table>

Table 4.3: Where would Silhouettell best be used?

<table>
<thead>
<tr>
<th>Order (1 means most suited)</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Public Space</td>
<td>2</td>
<td>5</td>
<td>6</td>
<td>15</td>
</tr>
<tr>
<td>Event Hall</td>
<td>10</td>
<td>8</td>
<td>7</td>
<td>4</td>
</tr>
<tr>
<td>Community Space</td>
<td>12</td>
<td>8</td>
<td>8</td>
<td>0</td>
</tr>
<tr>
<td>Private Party</td>
<td>6</td>
<td>7</td>
<td>6</td>
<td>8</td>
</tr>
</tbody>
</table>

Community space (student union buildings and so on) as being very useful (1) or useful (2). 62% also selected event hall (the site for an international conference and so on). In contrast, over 70% thought that Silhouettell would not be useful in public spaces (i.e. a street corner). The responses for Private Party (home party and so on) varied widely.

- Contents of profile which a user shows
  Table 4.4 shows the profile contents that were considered acceptable to the subjects. Most subjects accepted the display of name, affiliation, interest, and nationality. However, few accepted the display of phone number and address.

- Cultural Differences

65
Table 4.4: What data should be displayed?

<table>
<thead>
<tr>
<th>content</th>
<th>number</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>25</td>
</tr>
<tr>
<td>affiliation</td>
<td>21</td>
</tr>
<tr>
<td>nationality</td>
<td>24</td>
</tr>
<tr>
<td>address</td>
<td>9</td>
</tr>
<tr>
<td>phone number</td>
<td>7</td>
</tr>
<tr>
<td>interest</td>
<td>22</td>
</tr>
<tr>
<td>birthday</td>
<td>12</td>
</tr>
</tbody>
</table>

According to the results of this questionnaire, we found the following cultural differences.

- Places suitable for Silhouettell use
  Table 4.5(a) shows the answers of the subjects from Asia, where the data are the those in Table 4.3, while Table 4.5(b) shows the answers of the subjects from Europe and America. While 90% of Asians have affirmative opinions about the use in community space, only half those from Europe and America shared the same opinion. Though 70% of Asians had negative impressions about its use in private parties, European and American people had various opinions.

- Displayed Profile
  Table 4.6 shows the same data as Table 4.4, but they are divided into two columns according to the region the subjects were from. The left column shows the number of positive answers for each content from Asian people, while the right column shows those from European and American people. European and American people have negative opinion for displaying their affiliation, while Asians are more liberal on this issue.

We summarize these results from three points of view.
Table 4.5: Where would Silhouettell best be used?

(a) Asia

<table>
<thead>
<tr>
<th>evaluation</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Public Space (%)</td>
<td>9.1</td>
<td>27.3</td>
<td>36.4</td>
<td>27.3</td>
</tr>
<tr>
<td>Event Hall (%)</td>
<td>18.2</td>
<td>36.4</td>
<td>18.2</td>
<td>27.3</td>
</tr>
<tr>
<td>Community Space (%)</td>
<td>70.0</td>
<td>20.0</td>
<td>10.0</td>
<td>0.0</td>
</tr>
<tr>
<td>Private Party (%)</td>
<td>10.0</td>
<td>20.0</td>
<td>30.0</td>
<td>40.0</td>
</tr>
</tbody>
</table>

(b) Europe and America

<table>
<thead>
<tr>
<th>evaluation</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Public Space (%)</td>
<td>10.0</td>
<td>20.0</td>
<td>10.0</td>
<td>60.0</td>
</tr>
<tr>
<td>Event Hall (%)</td>
<td>27.3</td>
<td>36.4</td>
<td>27.3</td>
<td>9.1</td>
</tr>
<tr>
<td>Community Space (%)</td>
<td>20.0</td>
<td>30.0</td>
<td>50.0</td>
<td>0.0</td>
</tr>
<tr>
<td>Private Party (%)</td>
<td>40.0</td>
<td>20.0</td>
<td>10.0</td>
<td>30.0</td>
</tr>
</tbody>
</table>

Table 4.6: Cultural differences in releasing data

<table>
<thead>
<tr>
<th>content</th>
<th>number (Asia)</th>
<th>number (Europe and America)</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>10 (90.9%)</td>
<td>8 (72.7%)</td>
</tr>
<tr>
<td>affiliation</td>
<td>10 (90.9%)</td>
<td>4 (36.4%)</td>
</tr>
<tr>
<td>nationality</td>
<td>10 (90.9%)</td>
<td>8 (72.7%)</td>
</tr>
<tr>
<td>address</td>
<td>5 (45.5%)</td>
<td>2 (18.2%)</td>
</tr>
<tr>
<td>phone number</td>
<td>4 (36.4%)</td>
<td>2 (18.2%)</td>
</tr>
<tr>
<td>interest</td>
<td>8 (72.7%)</td>
<td>7 (63.6%)</td>
</tr>
<tr>
<td>birthday</td>
<td>3 (27.3%)</td>
<td>3 (27.3%)</td>
</tr>
</tbody>
</table>
1. Utility of Silhouettell
   In community spaces where people who have common topics and do not know each other gather, Silhouettell can be an effective tool in helping people to start conversations.

2. Privacy
   People hesitate to release their complete details to the public.

3. Cultural Difference
   We suggest that there are some differences of sense about presenting topics and privacy between areas. The sense on what topics or attribute one can open to public depends the area he/she come from.

Based on this preliminary result, we designed and studied the following experiment.

Thirteen people participated in our experiment: eight from Japan and five from the United States, Canada, and France. One of them was an industry researcher, and the rest were graduate and undergraduate students. Seven of the Japanese did not have much experience in English conversation. In each session, two persons talked to each other for fifteen minutes with the scenario that they were students of the same department of a university. We paired up the participants in advance (some participants were involved in more than one experiment condition) and prepared common topics (about sports, music, food, and books) for each pair from pre-test questionnaires. We used two to three web pages to represent a single common topic for each pair for the experiment. We also used each participant’s own web page as a topic if he had one. Name, birthplace, and the chosen common topic were used as the user’s on-screen profile.

We obtained the following feedback from the questionnaire administered after the experiment. The items reported here were rated on a seven-point scale by twelve of the thirteen participants (Figure 4.4).

From Figure 4.4, we found:

• Almost all users answered that they could talk smoothly.
- There is a difference between topics that users actually talked about and topics that the system presented. While Japanese users related their own topics to on-screen topics, the others did not do so.

We also observed and analyzed the video record of all conversations to help us understand when and how differently Silouettell might be used in cross-cultural encounters.

Figure 4.5 is a sample section of a conversation between two subjects, Shigeta (from Japan) and Stefan (from the United States). After an interval, Shigeta’s home page (in Japanese language) is shown and they begin to talk about this page.

The two subjects are looking at the screen, to try to find a topic to talk to each other about after the previous topic has ended. Then, Shigeta’s own home page is projected, and Shigeta points the screen (Figure 4.5(a)) and says, ‘This is my web page’ with turning to Stefan (Figure 4.5(b)(c)). Stefan turns to Shigeta after a moment when Shigeta began to talk, and then turns to the screen again (Figure 4.5(d)), just after he recognizes that Shigeta is pointing to the web page. Stefan watches the page, which is his conversation partner’s own page (he examines it for about 3 seconds), and then he indicates that he want to see the corresponding page in English, by saying
Figure 4.5: Reaction of participants

‘I choose English’ while pointing at a link on which the sentence ‘English page’ is written (Figure 4.5(e)). After these actions, the conversation about Shigeta’s page continues for a while. In this particular example, topic information about the subject, that was shown by system successfully triggered the next conversation topic.

The following examples were collected from the video logs:

- At the beginning of the conversation, both users checked their partners’ profile using the screen. Both participants always faced the screen and did not turn to face each other (two users from Japan).

- Conversations related to the screen contents occurred first, and the pair later extended it to include various other topics (two users from Japan).
Participants who could hardly communicate (mainly due to language problems) sometimes talked only about the profile and the web pages shown by the system. In such cases, the pair faced the screen and talked with each other with their face to their partner and frequently turning their face back to the screen. However, when they talked about a topic for a long time, their bodies would also turn toward each other (observed in a pair from Japan and the United States, and a pair from Japan and France).

In every case where a person’s own home page appeared on-screen, there was a strong positive reaction from the individual user, and conversation about the page ensued. Moreover, users moved around in many cases. Through these investigations, we confirmed the following features of cross-cultural communication in the Silhouettell environment.

- Information directly related to participants, such as their own home pages, plays a helpful role in initiating conversation.

- On-screen topics are frequently referred when there is a language barrier between conversation partners. We think this is because using the displayed information as a topic allows them to concentrate on talking itself, instead of having to do the additional work of starting new and appropriate topics.

- We could not find the apparent effect of shadows.

Based on our observations and short questionnaire to participants, areas for future work include:

- Expanding the range of what is displayed on-screen to include general topics such as news to help expand conversation support.

- Allowing users to give feedback to the system about what is displayed during conversations.
4.5 Supporting Communication between Remote Locations

In this section, we describe a system called Networked Silhouettell. The video technology used in Networked Silhouettell is similar to what is used in other meeting systems [Watabe90]. However, we use a large-screen environment, which makes nonverbal information such as eye gaze and expression, facial expression, and gestures easier to read. Also, the life-size representation of the participants helps to mitigate the demonstrated effects of screen size on perception of one’s conversation partner [Reeves96].

There are other large-screen communication environments that have recently been developed. For example, MAJIC [Okada94] is aimed at supporting gaze awareness, which is useful and important, but users need special equipment, and the system cannot really be used by many people at once. HyperMirror [Morikawa98] is another large-screen communication system that projects both local and remote users onto the same screen using video transmitted with chroma key. Networked Silhouettell uses HyperMirror-like display technology, but also augments the display with information about users’ cultural backgrounds.

Below, we will describe the design and implementation of Networked Silhouettell, and we will also describe the results of experimental use of the system by students from Japan and the United States.

4.5.1 System Design

Figure 4.6(a) shows the concept of Networked Silhouettell, and Figure 4.6(b) shows on-screen image of the system in use. Figure 4.7 is an example of actual use.

The key features of the system, shared view and cross-cultural experience information, will be discussed in the next two sections:

- **Shared View**
  In the Networked Silhouettell system, the images of remote users are
Figure 4.6: Networked Silhouettell

Figure 4.7: Actual use of Networked Silhouettell
displayed on a large screen, and the images of local users are overlapped on the same screen as transparent mirror images.

This style of representation in the communication environment has the following benefits. Since participants from both places are mapped onto the same screen, they can have a greater sense of sharing an environment compared to users of traditional, multiple-box video conference displays, and they can indicate with their gestures what they are referring to on the screen. Particularly in cross-cultural communications where verbal communication may be difficult, nonverbal information is an important component for a successful conversation. The life-sized display helps make nonverbal cues more legible, enhancing that aspect of communication. HyperMirror also uses the same kind of displaying technologies. However, it suffers from a problem due to its implementation—sometimes a person cannot see his partner because of overlapping when he stands in front of the partner. Using Networked Silhouettell, on the other hand, participants can still see the behavior of their partners even when he is in front of them, because their own image is transparent.

• Cross-cultural Experience Information

Users of Networked Silhouettell can more easily develop a rapport with their conversation partner, by looking at or touching information about his background that is readily available on-screen. We display the following three kinds of information about each user’s cultural background:

Language knowledge This includes language training and confidence in the partner’s native language as well as any other foreign language (e.g., How many years have you studied the language? How confident are you about your speaking ability?).

Culture literacy and experience This includes any experience in living in any other culture and warmth toward and/or understanding of that culture’s ways (e.g., How long were you immersed the cul-
Culture affinity and ties  This includes information about friends from other cultures, and other ties to the culture (e.g., How many friends from these other countries do you have?).

All information were entered by each user using a web form before they started using the system.

The cultural information is shown on-screen arrayed around the corresponding participant both in the participant’s native language and in his partners’ language, and each person’s information is displayed by a characteristic color. Thus, all participants can easily map information to users based on location as well as color.

Each item is selected by gesturing at the screen with a special device, which is a plate which can be simply recognized by colors†. At first, each item in a given category is shown as a title (Figure 4.8(a)). When a user selects an item, its more detailed version appears (Figure 4.8(b)). By observing changes in the display, a participant can understand what the other participant finds interesting about their background. For example, a participant who sees Figure 4.9 easily understands that the remote participant is trying to change the kind of information displayed on the screen.

4.5.2 System Implementation

This section describes the implementation of the Networked Silhouettell system. Figure 4.10 shows the system configuration.

The system works as follows. In a client machine, the Video Process sends the input images to the other client, and draws the image from the other client and the local client on the screen. The image from the other client is displayed as it is. In the case of the local image, the system detects the difference of the image from the background image recorded in advance,

†This device should be changed to some more robust devices in general applications.
(a) Selecting a title  
(b) The detailed information appears

Figure 4.8: Contents selection

Figure 4.9: Changing the kind of information on the screen
Figure 4.10: System configuration

cuts it out, creates a mirrored image, makes it transparent, and overlays it on the screen. The Voice Process simply sends the input voice and outputs the voice from the other client. As for the server machine, the Server Process gets users’ information from the Cultural Background Data File, and sends it to each client. It also synchronizes on-screen contents of both clients with Video Processes.

The Cultural-Background-Data Processing Component semi-automatically converts each user’s cultural information which is obtained from the web-based form into the data of the system, where some items are manually converted. The contents are displayed in both Japanese and English. Each piece of information is shown near the corresponding participant. The location and contents are sent to the server first, collected, then sent to both
When a user wants to select a kind of cultural information or a specific piece of information, he uses a pointing device. In this implementation, the user holds up a flat plate of a particular color, and the system detects its location using the input image from a CCD camera. If the detected point is at the same place on the screen as an icon, the system changes the kind of cultural information displayed. In the same way, if the point is in the same location as a particular piece of information, the system pops up the detailed information for that item.

As for user identification, we did not use any special technique of user identification but used a simple method in which a user selects his own name in the user menu, because we assumed one-to-one experimental use in this implementation.

The client and the server run on Silicon Graphics O2.

4.5.3 Experimentation

We evaluated the system by observing actual conversations among undergraduate students from Japan and the United States.

Method Seven pairs of students participated. Each pair consisted of a student from Japan and a student from the United States. The students from Japan had all been to other countries or studied conversation in English. The students from the United States were taking classes in Japan and had studied Japanese for a few months. We had participants enter cultural information using a web form, before the experiment began. Since the Networked Silhouettell system is a tool for supporting the users knowing each other, we also instructed them to do a task together after using the Networked Silhouettell system for evaluating the system. The participants were told that their task was to work together to make a mini-guide to their own university for students from their partner’s university, and that they were to use the
Networked Silhouettell system for twenty minutes before the task.

**Result** We obtained the following opinions from the questionnaire administered after the experiment. The items reported here were rated on an eight-point scale by eight or more of the fourteen participants (Figure 4.11).

- Impressions of the conversation partner and the Networked Silhouettell system
  The conversation partner was rated ‘trustworthy,’ and the system was rated ‘pleasant,’ ‘safe,’ ‘secure,’ and ‘entertaining.’

  There were also some positive comments, “bilingual information was easy to use,” “I can easily know my partner’s feeling with gesture,” and “transparent display is easy to use.” As an interesting comment, a user wrote that this interface would be more appropriate for n-to-n communication than 1-to-1 communication.

---

Figure 4.11: Questionnaire evaluation of Networked Silhouettell (8 highest, 1 lowest ranking)

79
• Comparison between Japanese and American
  In almost all items, Japanese students answered more positively than American students. We think it is because Japanese students had less experience of cross-cultural communications than American students.

We also observed the video log of this experiment.

The typical flow of conversation with the Networked Silhouettell system was as follows. First, participants talked about their own affiliations. They enjoyed talking about each others’ birthplaces and travel experiences to each others’ countries. Then, topics about the current environment where the participants lived (living, part-time job, and so on) began.

The following patterns were also observed in use of the cultural information.

• First, a participant read all of the information on the screen. Once the conversation began, however, he concentrated only on speaking.

• A participant talked while staring at the information.

In both cases, there were not any cases in which a participant related the information to the conversation topic directly. We did observe that sometimes, once a participant began to change the contents on the screen, the other participant would also begin to do so. As to the use of gestures, some were observed. For example, an American student showed three fingers to his Japanese partner while speaking, when the partner could not hear his sentence ‘Are you in your third degree?’ Interestingly, there were some bilingual conversations in which participants each spoke the other’s language (the Japanese student spoke English, while the American student spoke Japanese).

For the system performance, since the system has a problem of image latency (0.6 second) due to system resolution and machine speed, there were complaints about system performance.

Through the whole experiment, we confirmed the following features of cross-cultural communication in the Networked Silhouettell environment.
• The non-native users (Japanese) in conversation rated the system more positively.

• Users use gestures in this environment without any special attention to the system.

• The transparent shadow is helpful for the users to understand the pointing function, while we could not find the apparent effect of the feeling of existence.

4.6 Discussion

In this chapter, we have described factors needed to support cross-cultural socialization, both in the real world and between remote locations, and discussed the design, implementation, and evaluation of two systems designed to support this kind of socialization.

From the actual use of two systems, we empirically confirmed the following effects of the systems:

Shared information  On-screen common topics are frequently referred when there is a language barrier. That is, users who do not have much experience of cross-cultural communication or is not non-native frequently referred common topics. The apparent effects of cultural backgrounds are not found.

Large-sized interface  Each user’s movement and gestures were done without any special attention to the systems. Therefore, when we develop a system which aims at a natural mode of communication, the nonverbal functionalities should be designed to be enable without any operation as verbal functionalities, e.g., the voice communication.

Displaying shadows  We believe shadow is useful to handle information on-screen. In the experiment of Networked Silhouettell, we got opinions
that the shadow is helpful in using the pointing devices. We could not find out whether this interface provides the feeling of existence or effects in identifying users.

We also confirmed the importance of response speed in an interactive system. There was no feedback about image latency from users of Silhouettell, which does not incorporate user feedback into the display. However, there were many complaints about image latency from users of Networked Silhouettell, which allows users to make changes on-screen with a pointing device. Thus, latency seems like a larger problem in a system which incorporates user feedback.

We could not find some effects, i.e., the effects of cultural backgrounds, the feeling of existence or effects from shadows. We think it is because the systems are not task-oriented and the conversations in each experiment includes only two persons. We have to consider these points in future works.

4.7 Summary

In this chapter, we discussed the importance of tools to support information sharing, especially focusing on early socialization in cross-cultural collaboration. We proposed two support systems, which present shared information based on each participant’s profile and cultural background by both using a life-sized interface with a large screen and representing participants in intuitive and natural ways on-screen during the interaction. We also discussed experimental evaluation of both systems.

Silhouettell is a system which supports real-world communications by presenting web pages related to common interests of participants located in the same place. The system shows shadows on a large screen, which attract participants and displays common topic web pages based on users’ profiles. Through observations of the actual use of the system, we found that (1) information directly related to participants is referred in initiating conversation, and (2) participants frequently refer to the contents on the screen in a situation whereby it is hard to communicate to each other because of
difference in participants’ native language.

Networked Silhouettell is a remote communication tool, which gives participants a sense of a shared environment by displaying them on the same screen. It encourages understanding among users by showing each user’s cultural experience and background information on-screen. A participant can select and examine detailed information about his partner’s background using gestures with a pointing device. We found out that (1) The non-native users used the system more positively, and (2) the transparent shadow is helpful with the pointing function.

As a result of both studies, we empirically confirmed that: shared information is referred in situations where it is hard to communicate because of differences in the participants’ native language; and users use movement and gestures naturally in the large-sized interface.

We believe that a learning conversational agent can join human community with sharing information between agents and people in this way.
Chapter 5

Participation of Conversational Agents in Communication Environment

5.1 Introduction

In this chapter, we focus on another kind of conversation, called scene-driven conversation, which is a kind of context-independent occasional conversation. This appears when a conversational agent participates the communication environment where there are many agents and people.

We describe this style of conversation, and we propose a framework to develop an agent which treats with scene-driven conversation in a virtual space [Yamanaka03]. We also show the typical example of conversation between a user and an agent.

Section 5.2 describes the scene-driven conversation, Section 5.3 presents the architecture of our implementation, and Section 5.4 shows the design process and an example construction of the conversational agent.
5.2 Scene-driven Conversation

A scene is a set of each actor’s perceptual information, and a scene-driven conversation is a kind of conversation which happens after that an actor speaks or attracts another actor’s attention with a scene as a start. This kind of conversation is independent of previous conversational contexts, while the meaning of each scene depends on the task.

In this section, we describe the differences between task-oriented or task-driven conversations and scene-driven conversations, and scenes in a 3-D virtual space.

5.2.1 Task-driven vs. Scene-driven

An issue of conversational agent is to collect users’ requests. In usual task-driven conversation, the domain of a user’s utterance is restricted by setting the task adequately. For example, ‘where to go,’ ‘how to go,’ and ‘what to guide’ will be tasks in a guidance scenario.

Whereas, in a scene-driven conversation, each scene which an actor perceives restricts his/her utterances. For example, cues such as ‘notice something’ and ‘find out something,’ will be scenes in a guidance scenario.

Moreover, in the process of constructing an agent, method to handle task-driven conversation and method to handle scene-driven conversation are different.

In a task-driven conversation, it is important to model a conversational flow. Therefore, it is necessary to collect a history of conversation and make a model from the history. There are some modeling method such as describing the conversational flow or slots by hand or learning a model statistically. For example, some method to construct dialogue models with the Hidden Markov Model or the Markov Decision Process are proposed [Levin98][Singh00]. Agent construction process with the WOZ and the machine learning technology described in Chapter 3 also models the task flow.

In a scene-driven conversation, it is important to decide each scene pro-
vided by an actor’s behavior. Therefore, it is necessary to define elements of which a scene is composed. We propose a method to decide each scene from actual behavior of user and agent in monitoring them, make conversation rule from the scene, and put the rule into the agent’s scenario.

In the next sections, we deal with conversational agents in a 3-D virtual environment where actors behave in various ways, and developer can capture their behavior more easily than in the real world.

### 5.2.2 Scene-driven Conversation Model

A scene in a 3-D virtual environment consists of the collection of following elements.

- **Place:**
  the position of actor (agent or avatar) or in which area the actor is
  ex.) in front of which shop is the actor

- **Linguistic action:**
  utterance by an actor or association from some keywords in an utterance
  ex.) a topic which appears as an association of guide utterance

- **Non-linguistic action:**
  non-linguistic action by an actor, such as movement or gestures including looking around or pointing

- **Relationship with another actors or objects:**
  relationship with another participants or objects in the environment
  ex.) position, direction, visual field, passing each other

### Example Situation

Some examples of scene-driven conversation are described below.

Suppose a guidance with a guide on the Shijo street in Kyoto. When they walk from Shijo-Teramachi crossroad to Shijo-Kawaramachi crossroad...
eastward, you may find some interesting shops or be interested in another person.

- You find out a building, and then say “What is that shop?” or “Shall we enter this shop?”, or let the partner know your interest by turning to or pointing the building.

- When you listen to the guidance about Fujii-Daimaru department store, you may say a related topic such as “By the way, where is Daimaru?” This topic is association with the word ‘Daimaru.’

- You may pay attention to the clothes of a person who passed each other, and then “His clothes are attractive. Do you know what brand clothes he wears?”

5.3 Agent System in Communication Environment

This section describes the implementation of an agent which uses scene-driven conversation proposed in the previous section and the agent-construction environment.

5.3.1 System Architecture

Figure 5.1 shows the system architecture. The agent system consists of the agent executing environment and the monitoring/editing environment.

5.3.2 Agent Executing Environment

The agent executing environment provides a space in which agents actually behave and an interaction model of agents. This environment consists of the following systems.
agent system
The agent system is a system in which agents behave such as moving or speaking. In our implementation, FreeWalk [Nakanishi99] is used (Figure 5.2). FreeWalk consists of the Community Server which manages agents and avatars in the 3-D virtual space and client systems. On each client system, a 3-D virtual space is drawn with VRML models, and there are two kinds of actors in the space, avatars controlled by the corresponding users and agents controlled by scenario description.

scenario processor
It is a processor to execute scenarios which describe behavior or interaction of agents. In our implementation, a scenario description language $Q$ [Ishida02a] and its processor is used.

In the $Q$ language, each event observed by agents is described as a cue, and each request to agent to change the environment is described as an action. An interaction rule is composed of cues and actions. Interactions among agents and avatars are described with parallel observation of rules and state transitions. Table 5.1 shows the full list of cues and actions implemented in FreeWalk environment.

Figure 5.1: Architecture of agent construction environment
Figure 5.3 shows an example of parallel observation with *guarded command*. In this example, the observation by the agent continues until the agent hears User’s speech “Do you know this shop?”, sees User turn left, or sees Yasaka shrine. If one of these cues is observed, the corresponding action is executed.

5.3.3 Monitoring/Editing Environment

The monitoring/editing environment has an interface for the developer to monitor the behavior of agents and avatars in the agent system, and to append scenes or rules (Figure 5.4). This interface is implemented with DrScheme*, which is an implementation of Scheme language processors.

*http://www.drscheme.org/
Table 5.1: List of cues and actions

<table>
<thead>
<tr>
<th>Function</th>
<th>Cue</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Motion</td>
<td>?position</td>
<td>Movement</td>
</tr>
<tr>
<td></td>
<td>(get location and direction)</td>
<td>!walk</td>
</tr>
<tr>
<td></td>
<td>!observe</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(observe gestures or actions)</td>
<td>!approach</td>
</tr>
<tr>
<td></td>
<td>?see</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(see objects)</td>
<td>!block (block another agent’s way)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Rotation</td>
</tr>
<tr>
<td></td>
<td></td>
<td>!turn (turn body)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>!face (turn face)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Gesture</td>
</tr>
<tr>
<td></td>
<td></td>
<td>!point (point at an object)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>!behave (perform some gesture)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Appearance</td>
</tr>
<tr>
<td></td>
<td></td>
<td>!appear (show up)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>!disappear (hide self)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Conversation</td>
<td>?hear</td>
<td>!speak</td>
</tr>
<tr>
<td></td>
<td>(hear voice)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>!receive</td>
<td>!send (send text messages)</td>
</tr>
<tr>
<td></td>
<td>(receive text messages)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>?finish</td>
<td>!change</td>
</tr>
<tr>
<td></td>
<td>(finish asynchronous actions)</td>
<td>!finish (stop asynchronous actions)</td>
</tr>
<tr>
<td></td>
<td>!input</td>
<td>!output (output logs)</td>
</tr>
</tbody>
</table>

```
(guard
  ((?hear :name User :word "Do you know this shop?")
   (!speak :sentence "This shop is famous Indian restaurant in Kyoto" :to Hanako))
  ((?observe :name User :action turn_left)
   (!speak :sentence "Are you interested in the shop?")
  )
  ((?see Yasaka Shrine)
   (!walk :forward 100.0 :left 50.0)))
```

Figure 5.3: Description of a parallel observation with guarded commands
There are two kinds of monitoring functions, monitoring the scenario processor, and monitoring the agent system. In monitoring of the scenario processor, states in a scenario and rules in a state are checked. In monitoring of the agent system, internal states of each agent and the environment itself are checked.

The developer edits and renews the scenario with the edit dialog (Figure 5.5) based on information checked in the monitoring dialog.

### 5.4 Design Process

This section describes the design process of constructing an agent. There are the following four actors in this process.

- *Task designer* designs and describes scenarios of initial task, e.g., guidance task.
- *Monitor* monitors behaviors of avatars and agents, and decides each
Editor edits the scenarios with putting additional rules treating with scene-driven conversation.

Each user controls an avatar, and the avatar communicates with conversational agents or another avatars.

The same person often plays both roles of monitor and editor. The task design or decision of each scene depends on the task designer’s policy and the editor’s policy, respectively.

The construction process of an agent is as follows.

1. describing scenario:
   the task designer describes the initial scenario following the task

2. decision of scenes:
   avatars controlled by users actually communicates with the agent. monitor/editor defines each scene from observation
3. putting scene-driven conversation:
   editor puts or modifies rules of scene-driven conversation

4. repeat step 2 and step 3

If needed, additional contents or templates of agent’s utterances are prepared.

**Example Agent Construction**

We show an example construction of guide agent in Shijo street.
The initial scenario is as follows.

“The agent introduces some shops while walking along to Shijo street from Shijo-Teramachi crossroad to Shijo-Kawaramachi crossroad.”

In this scenario, the agent only repeats to stop to speak in front of a shop and walk again.

At first execution, the avatar may fail to keep up with the agent or get ahead of the agent. The editor puts the rule “when the distance between the agent and the avatar is more than some threshold, say ‘follow me!’” into the current scenario.

After that, the avatar stops and says “What is that shop?” in front of a variety shop. The agent continues to walk for a minute, and then say “follow me!” as the scenario. The editor puts the rule “when the avatar say ‘What is that shop?’ in front of the variety shop, introduce the shop.” Figure 5.6 shows the example, and Figure 5.7 shows the filled rule. If the avatar asks the agent of related things and the agent knows some contents, another rule which includes “?hear” and “!speak” will be added.

In another execution, an avatar sees another actor (an agent or an avatar) because the actor is attractive or known by the avatar. The agent can tell about the actor if there is any known profile. In this case, a rule which reflects relationship between actors will be added. Figure 5.8 shows an
(1) an avatar follows the guide agent

(2) the avatar stops to turn to a variety shop

(3) the agent turns to the avatar and guide her about the shop

Figure 5.6: Example conversation triggered by non-linguistic action
Figure 5.7: Example conversational rule when an avatar stops to turn her body to a shop

element conversation with such kind of trigger, and Figure 5.9 shows the filled rule. The function “agent-attrib” searches the specified profile of an agent.

Moreover, the avatar says “There is a Mos Burger shop, isn’t it?” when the agent is talking about McDonald’s. Then, the editor puts the rule “when the avatar refers to Mos Burger, introduce some fast-food restaurants.”

As a result, the agent is constructed by combining pre-defined task and scene-driven rules in actual use.

5.5 Summary

In this chapter, we focused on scene-driven conversation which is not treated with in usual conversational agents, and we modeled such kind of conversation. We used position, linguistic actions, non-linguistic actions, and relationships with other actors or objects as the components of each scene, and implemented an agent construction environment.

We believe that this kind of agents can participate in community space such as digital cities [Ishida02b] naturally.
(1) an avatar follows the guide agent

(2) the avatar finds pedestrian agents and turns to them

(3) the agent turns to the avatar and guide her about their information

Figure 5.8: Example conversation triggered by relationship between actors
Figure 5.9: Example conversational rule when an avatar stops to turn her body to a shop
Chapter 6

Conclusion

6.1 Contributions

In this thesis, we proposed a new approach to develop conversational agents which support interpersonal communications. First, we proposed a new framework to develop conversational agents which provide information incrementally. In this framework, both costs of collecting example dialogues and of learning dialogue models are reduced. Next, we designed a communication environment Silouettell where people share conversational topics and communicates with each other easily. We also focused on scene-driven conversation which occurs when an agent joins a communication environment and communicates with people.

The major contributions are summarized into the following respects.

- A framework to construct conversational agents incrementally

We proposed a kind of conversational agent which is incrementally developed through actual conversation, called learning conversational agent. A problem in constructing a usual conversational agent is to debug conversational rules or collect example dialogues or corpus in advance. Our method is an example-based approach, in which the dialogue-collection process and the dialogue-model-learning process are combined. The feature of this method is that the cost of
collecting example dialogues is reduced gradually because the Wizard of Oz method, which itself is famous method to collect corpus, is enhanced by the machine learning technology.

In the construction process, at first the developer should input almost all answers. As the learning proceeds, the system infers appropriate utterances and the cost of human guide’s inputting utterances reduces. Finally, a conversational agent is constructed. We implemented an WOZ-based agent system on the Web, and we confirmed the reduction of the human guide’s cost as the growth of the dialogue models in the Kyoto tour guide task.

• **An expansion of PDFA-learning algorithm to incremental situation**

We expanded a probabilistic-DFA (PDFA) learning algorithm to cope with the incremental situation. In our conversational agents, a finite state machine (FSM) is used as a dialogue model because a developer can interpret the model easily in the agent construction. A PDFA-learning algorithm with the state-merging method is useful to make a cyclic FSM from examples including noises. However, there are no algorithm which considers the gradual increase of example data. We proposed an expansion of this algorithm with a caching technique. This algorithm decreases the number of compatibility checks by caching the state-merging information. With the example dialogues of Kyoto tour guide task, we confirmed that our incremental learning algorithm reduced the total number of compatibility checks.

• **Communication environment for information sharing**

We proposed an communication environment where people visually share their information such as their background or profile, which is provided from agents. Usual communication environment such as electronic conferencing systems provide a virtual meeting space for people. However, in spite of the opportunities of conversation from many places or countries are increasing, the support for conversa-
tional contents are not thoroughly examined. Therefore, our environment focuses on sharing information among users.

This environment is designed to support communication with the following three functions: (1) showing topics based on participants’ profiles and cultural background; (2) life-sized, large-screen interface; and, (3) displaying objects which show feelings of identity. From the evaluation of these systems in the cross-cultural situation, we studied that these systems add new functionality to support conversation contents, which may be especially useful in a cross-cultural context where language skills are an issue, and this type of environment may be especially useful in a pre-collaboration context.

- **An architecture of conversational agent which supports scene-driven conversation**

  We focused on scene-driven conversation which occurs when an agent joins a communication environment and communicates with people. The feature of this kind of conversation is that it is independent of the conversational context or flow, while the meaning of each scene depends on the task. When a developer constructs conversational agents which treat with scene-driven conversation, it is reasonable to put in each occasional interaction in actual use in the developer’s judgement rather than to put interactions in advance.

  We proposed an agent-construction framework to add scene-driven-conversation rules, and implemented a monitoring/editing interface for an agent system with a 3-D virtual space. We also developed conversational rules which cope with typical scene-driven conversation.

- **Common grounding among human and agent**

  To design a conversational agent which is used in a daily life, a common ground among humans and agents should be clear. It is important to limit conversational topics to make it. From the evaluation of the Web-based Kyoto tour guide, we found that the task “Kyoto tour
“guide” is too wide for an agent handles. For example, a university student who lives in Kyoto want to know very local information such as bars, while a user from outside of Kyoto want to know typical famous places. Whereas, we also found the number of digressions is totally small. This experience gives that a conversational agent can limit topics which a user chooses. The experiment of Silhouettell also showed that people use provided topics when there is a barrier which comes from the difference of mother languages. In this case, the topic is clearly limited.

Above result gives us a direction to make conversational agents which behave in human community with providing and sharing information.

6.2 Future Directions

We conclude this thesis with the list of possible future research issues of conversational agents.

- **Storing and updating personal information for daily use**
  
  Each user’s profile including interest changes according to the flow of conversation, while our communication environment treated with each user’s profile as static. These information should be collected in daily work such as Web browsing or reading/writing e-mails. This idea raises an idea of personal agents. A personal agent will be work as a secretary in the daily life, and work as a mediator between people in a meeting or a conversational situation.

- **Combining various displaying devices according to privacy levels**
  
  In Silhouettell, only shared information is used for conversational topics. Shared information is useful for the first topic to know each other. However, once people make a community, more private information is needed.
One solution of this problem is using small devices such as PDA (Personal Digital Assistants) for displaying private information. In this case, there are two kinds of forms of the conversational agent. One style is that an agent moves between a PDA and a large screen as a personal agent we mentioned above. In this style, each personal agent communicates with other personal agent and decide the topics in the conversation. The other style is that there is a personal agent in a PDA and there is a conversational agent in a large screen. In this style, the conversational agent gather each personal agent’s request, and provides topics as a chair of the conversation.

- **Conversational agents which join group conversation**

  In our framework, each conversation occurs between an agent and a human. However, in a group conversation, there is a various kinds of conversation such as intervention or participation in other actors’ conversation. To realize these kind of conversation, each agent should have an ability of understanding complex conversational contexts. In a real world or a 3-D virtual environment, gesture recognition and multi-user speech recognition techniques are required. This functionality will be one of the most important features of conversational agents in human community, while it is a challenging goal.

- **Debugging Environment for Humanoid-Human Communication**

  In future, humanoid robots with physical bodies [Ishiguro01] as well as software agents in virtual worlds will join human-human communication, while we focused on conversational agents in virtual environments as communication mediator in this paper.

  An issue in constructing a communicative humanoid is to construct a debugging environment because human behavior is difficult to reproduce. A large-screen environment with video capturing devices will be a powerful tool to reproduce the same humanoid-human interaction.
Additionally, captured video tracks in the debugging process will be contents as topics.
Bibliography


[Borovoy98] R. Borovoy, F. Martin, S. Vemuri, M. Resnick, B. Silverman and C. Hancock, “Meme Tags and Community Mirrors: Mov-


Publications

Major Publications

Journals


International Conference

Other Publications

Workshops


**Conventions**

