Implementing a Multiparty Support in a Tour Guide system with an Embodied Conversational Agent

Aleksandra Čereković (1), Hsuan-Hung Huang (2), Takuya Furukawa (2), Yuji Yamaoka (3), Igor S. Pandžić (1), Toyoaki Nishida (2), Yukiko Nakano (4)

(1) Faculty of Electrical Engineering and Computing, University of Zagreb, Croatia, (2) Graduate School of Informatics, Kyoto University, Japan, (3) Department of Computer, Information and Communication Sciences, Tokyo University of Agriculture & Technology, Japan (4) Dept. of Computer and Information Science, Faculty of Science and Technology - Seikei University, Tokyo, Japan

Abstract

In this project we aim to implement multiparty support in a Tour guide system with an Embodied Conversational Agent (ECA). We studied multiparty dialogue issues and adapted some of them to our system. As the outcome of the project, the first prototype of a tour guide system has been developed. The system can automatically detect presence of humans and invite them to a tour session. During the session, according to the number of users and their positions, ECA uses a gaze model implemented from conversational theory to address a specific user. ECA also reacts to decreased level of attention and tries to keep the interest in the session. Although conversational model in our system is rather simple, we find this work a first step for further system development in the ECA-human multiparty domain. Experiments with such systems can provide useful research studies in the field of interaction between ECA and humans.

Index Terms— multiparty communication, multiparty issues, ECA gaze model

I. PRESENTATION OF THE PROJECT

A. Introduction

Embodied Conversation Agents (ECAs) offer great promise to achieve natural interaction: they are anthropomorphic and have the potential to act like humans. Effects of ECAs are being studied through different roles they have in educational purposes, information delivery services, health care, sales and marketing. These studies indicate that ECAs can improve user engagement and significantly increase users’ positive perceptions of learning experiences [10]. Despite all the positive effects, ECAs still mostly exist in research laboratories. As a recent state-of-the-art paper [14] claims, there is substantial research to be done on the way towards believable human-like ECAs.

So far, research community has mostly been oriented towards making ECAs capable of handling dyadic communication with a human or with another ECA. Proposed dialogue models have focused on monitoring just one conversational partner, which enables the ECA to participate in a “real” conversation. In multiparty ECA systems, most issues belong to the dialogue management domain. In [9] Traum identifies those issues and groups them into three parts:

- **Participants’ roles**: identification of local roles of participants which shift during interaction (addressee, listener, speaker), responsibilities in the dialogue, social roles
- **Interaction management**: Managing a communication flow in a multiparty system is far more difficult than in a dyadic system. Some of the difficult issues are how to give and recognize a turn and how to handle different channels (and backchannels). Besides, conversations can be easily split and merged together and attention can be paid to several persons.
- **Grounding and obligations** are notions commonly used to model local state of dialogue. In multiparty communication usage of these models can become very complex; e.g. if there are more addresses, it can become unclear what a proper grounding should be.

The issues identified by Traum have been used in the MRE project [12] for creating a negotiation model for virtual agents. Although this model is being used for several agents and one human, the ideas and factors presented can also be used as general guidelines for implementing multiparty support in ECA systems. Since this research was published after the workshop, it was outside the scope when developing the eENTERFACE ’08 project.

Apart from Traum [1, 9, 12] there are relatively few researchers who studied conversation of multiple users and ECA. In [11] Rehm and Wissner point out that “the literature on the behavior of multiple users is sparse”. Exception is the work from Vertegaal et al. [7] who studied gaze behavior shifts depending on conversational flow. They perform a real study on humans gazing in three-party conversation and propose a computational model of gazing which follows the findings from their study. They note that on average, subjects looked about 7 times more at the individual they listened to than at others and about 3 times more at an individual they spoke to than at others. They conclude that gaze, or looking at faces, is an excellent predictor of conversational attention in multiparty conversations. Rehm and Wissner [8] developed a
gambling system in which ECA plays a dice game with two humans. Game rules define the system scenario and make a simple dialogue model in which turn-taking mechanism and participants’ roles are round-based. Their system lacks the active gaze model which follows human users. By using this system Rehm and Andre [11] investigated human gazing patterns in interaction with a real human and agent. Conclusions from their study are the same as Vertegaal’s, however they also find that “People spent more time looking at an agent that is addressing them than at a human speaker.” This phenomenon can be explained by the fact that prolonged eye contact in a social interaction can be found as impolite and rude; hence, the agent in this game might be considered an artifact and not a human.

Summarizing the state of the art, we conclude that the area of multiparty ECA systems is so far very little studied. A basis in communication theory has been set up by Traum, gaze behaviors have been studied in terms of conversational attention, but what lacks are systems which support it and which can serve as a test bed for identifying other issues or drawing interesting conclusions. In this project we aim to create a multiparty ECA system which will interact with two human users. Since we had a tour guide system at our disposal [6], we decided to use some of existing components (character animation player, platform) and investigate how the multiparty aspect can be incorporated into this system. Although it looked simple at first sight, multiparty communication support forced us to develop completely new components.

B. Domains and challenges

In the system design phase, we planned it in such a way that it can recognize and respond to typical situations in a multiparty conversation. We added certain additional features we found mandatory or interesting also for dyadic communication.

1) The multiparty aspect

Since our research group lacks knowledge in natural language processing and understanding we decided to predefine a system scenario and predict users’ behaviours. We chose a dialogue model in which the tour guide ECA has a role of narrator and keeps the initiative during the tour session. Narration is related to Dubrovnik’s cultural heritage, history, and monuments in each of the five specific scenes we use in the system. During interaction users can interrupt the ECA and ask questions or just make a comment about the current topic. We predict topic-related questions and define it by using specific keywords in the speech recognition engine (such as “where”, “when”, “how”). To keep attention during the session ECA can also ask users simple “yes/no” answer-based questions.

Regarding the multiparty/multimodal aspect, we reference Traum’s paper [9], focus on a limited number of issues and propose their solution in our system. Since our system doesn’t use advanced dialogue models, there is no grounding, goals, or natural turn-management:

- **Appearance of users.** The system can dynamically recognize how many users are present and where they are standing. Besides, it can invite users to join the session. Depending on the number and positions of users, we use the agent’s gaze direction to maintain the conversational flow. If there is nobody present, the system will be restarted.
- **Channel management.** We combine users’ verbal channels with nonverbal behaviours (face orientation, gazing) to resolve users’ utterance. However, the system can understand only a limited number of users’ behaviours: decreased level of attention, making a request and leaving the system.
  - **Speech collision.** During interaction it might occur that users ask questions at the same time. We handle this issue by using the agent to address one of the speakers and give him a turn, e.g. tour guide agent Dubravka says “You can speak one by one, I’ll respond to you both. You can be the first one (addressing one user by pointing)”
- **Identification of conversational roles.** The system identifies local roles of participants during the session. Communication workflow is defined in the scenario; the agent keeps initiative and during the session he is the speaker most of the time, while the users are addressees. In the case when a user speaks, we identify him by using a localized microphone. It is a situation when the agent becomes an addressee, unless this user turns to another user and they start to communicate with each other.

Additionally in the multiparty domain we add two system features:

- **Handling users’ mutual conversation.** This is a situation which we handle in “unnatural way”. When users start to communicate with each other, the agent will try to get attention and continue his narration. Example is behaviour: “(waving) OK, let’s continue with the tour”. Since ECA can not understand the topic of their conversation we were forced to choose “a step back from empathy”, but still retain ECA’s politeness.

- **Inviting users to join the session.** The system can recognize human observers which are standing in the background and watching the interaction. They are treated as a group and, if present, the system will address them only if there are free places for users available. Example is the situation when user leaves the system and there is one person standing in the background. Then, Dubravka will look at the back and say “We have one free place, please come closer and join the tour”.

2) Additional Features

We are also concerned with system features which can make interaction with the system fluent: the system can recover from failures, such as failure of speech recognition. In
case the user’s speech is not recognized, we propose two-stage recovery. First, agent Dubravka will ask the user to repeat their question, and in case second failure occurs she will respond: “I’m sorry. I don’t know a response to your question”.

We also use the feature of Okao’s vision [17] to detect gaze direction and recognize a situation in which the user is paying less attention to the session. If this occurs, agent Dubravka will speak until the end of her utterance, turn to the user and say “Seems you are not interested in this topic. Would you like to hear something else?”

3) Discussion

Issues addressed above are certainly not the only ones which must be solved to establish and maintain normal three-party interaction with ECA in a narration-based session. For this project these issues were general guidelines to implement a test bed for further research. After being integrated, it was planned to evaluate the system with random users (not during the workshop). By the end of the workshop we didn’t complete all plans; but we managed to integrate the first prototype of the tour guide system. The system is capable of running the first scene in which we aimed to handle all planned issues.

Input part is bimodal: it captures users’ verbal and nonverbal behaviours (face orientation and gazing direction). For verbal input we produced a component which detects specific keywords and defines the user’s preferences. As for nonverbal input we produced two components which are combined to resolve the following situations: users’ arrival and departure, numbers of users present, situations in which users are paying less attention. Unfortunately, one month of the workshop wasn’t sufficient to complete a tracking algorithm which was mandatory to detect a situation when users are starting their own conversation (they rotate their faces to each other and activate verbal channels). Also, we didn’t have time to deal with image processing and divide the space to distinguish between the users and audience. This issue turned out to be rather complex because the area where the system was installed wasn’t large enough (we couldn’t distinguish between users and audience members by using only size of the face since there was no space behind the users). That’s the reason why we allow a maximum of two users in the system’s scope.

To manage input multimodalities and choose an appropriate agent’s response we initially intended to use MIDIKI’s toolkit [4]. Idea was to integrate multiparty support in MIDIKI’s state update mechanisms: e.g. nonverbal events and specification of gazing. During the workshop we realized that toolkit’s documentation was inadequate for fast implementation and in the end we adapted an existing scenario component [6]. However, not all issues in the central part are implemented: it lacks speech collision handling and there is no smart scheduling and behaviour control (e.g. continuous control of gazing).

In the animation player we implemented patterned gaze behaviour towards users. Patterns follow findings from computational theory [7, 11]. We also modelled behaviours (gestures, facial expression) to increase ECA’s believability. Animation player also lacks some features. We are not handling users’ comments at the moment because there is no smart control for this level. We handle interruptions by deleting all queued behaviours and waiting for new behaviours from the central part of the system.

In the remainder of this paper, we describe the overall system. First, we depict and explain hardware configuration, while in the third chapter we explain system design and components’ functionalities in detail. For the end, we discuss the system, evaluation and conclude the work completed on eNTERFACE ’08 workshop.

II. HARDWARE CONFIGURATION

System was installed at LIMSI, which was venue of eNTERFACE ’08 workshop. It is depicted on Fig. 1.

Two equal microphones are installed on two machines under the same settings. During installation we had to make sure that distance between microphones was great enough to avoid speech overlapping between microphones. Results of several experiments showed that Loquendo ASR [21] was stable enough not to recognize speech from approx. 0.5 m between users. We positioned a table between users to make sure the system can distinguish between users. The system was installed on two machines: one sends input to Motion Detector which detects motions in the system’s scope, while the other is connected to
Okao’s vision component which can recognize face orientation and gaze direction if face doesn’t exceed 60 degrees relative to the position of the camera (more details about how these components work will be presented in the third section.) To handle data from cameras we use another computer.

Scenario component runs on the third machine. It does not inquire any additional hardware, and neither does GECA server.

For system output we use another machine, projector and a large panel to display the agent and its behaviours. We calibrate the window position to make the agent large enough to make the conversation circumstances more natural (Fig.1)

III. SYSTEM DESIGN AND COMPONENTS

System design is depicted in Fig. 2. System features explained in the first section are granulated and mapped into components which are divided into three categories: input, central part, and system output. All designed components are connected to GECA platform which can transport and mediate messages in real time.

A. System Input

Input data processed within the system are human speech, appearance, faces and direction of gazing. Combination of this data can resolve one the following situations:

- User arrival
- Number of users
- Interruptions
- Unrecognized speech
- Decreased level of attention
- Leaving the system

We also planned to detect and handle speech collision, conversation between users and presence of the audience (to invite them if there is a free place in the system). However, there was no time to complete the tracking algorithm and divide the space. Speech collision detection issue is simple from the system input point of view, but takes more effort on the central part.

1) Speech Recognition

Speech recognition component is developed with Loquendo ASR [21]. Experiments performed at the very beginning of the workshop showed that Loquendo is very satisfying for our needs; it is stable in noisy environments, speaker-independent, there is no significant speech overlapping at reasonable distance between users (0.5 meters approx.), and no keyword spotting issue, which was a problem with Microsoft SAPI. Besides, it has garbage-rules definition to reject words that are not defined in vocabulary, confidence score recognition and timing specification of recognized words, which are additional features used in our project.

Once the speech recognition component is started, it waits for users’ speech input. When it receives a speech start signal, it starts recognizing the speech. The system also recognizes who is speaking. The results of recognizing the speaker and speech content are sent to OpenAirServer. The process flow is as follows.

1. Once the Speech Recognition component is started, this module starts listening to the audio stream.
2. When there is an utterance input from the user, this module gets the start time of utterance from Loquendo, as well as the utterance duration.
3. Finally, it transmits the results obtained in flow 2 as character string. It repeats flow 2 and flow 3.

SR sometimes reacts to voices which are not from the user. We think such an error can be prevented by using a Speaker Verification function in Loquendo.

2) Nonverbal Behavior Recognition

To detect nonverbal behaviours we developed two components: Okao Vision and Motion Detector. In cooperation these components have two functions: they can
detect the number of users and calculate the user’s attention level towards the agent.

**Figure 3. Detection of user’s face and estimation of gaze directions with Omron’s Okao Vision**

**Figure 4. Input image processing with Motion Detector**

Okao Vision is a commercial product made by Omron [17]. It is a library that provides accurate face detection and extra functions like face orientation, gaze direction, the positions and openness of eyes and mouth, gender detection, age identification and face identification from one single image. Fig. 3 is an example output of a program written with Okao Vision. Requirements and limitations vary with functions. Face detection requires a 10x10 pixel square image of the face and the head rotation must be within 60 degrees in left-right direction (yaw) and 30 degrees in up-down direction (pitch). Face orientation and eyes-mouth openness detection require a 40x40 pixel image, and the rotation of the head must still be within 60°(yaw) and 30°(pitch). Gaze direction requires an 80x80 pixel image and is limited to 30°(yaw) and 15°(pitch). In our test program with a 960x720@15fps web cam, accuracy of face detection is sufficiently high and undoubtedly usable, but most of the other functions are not reliable and can only be treated as an extra bonus. Since Okao Vision does not require stereo cameras, it is an acceptable result. We thus decided to use the face detection, face orientation and eye-mouth openness. Face orientation is also used to approximately determine the users’ gaze direction. This is not very accurate but should be sufficient if we only need to distinguish rough directions like the screen or another user.

Motion Detector is a component that can detect a moving object in distinct areas, and is developed by using OpenCV library [20]. The component divides the viewed region into two distinct areas and detects motions inside each area. Fig. 4 shows video image data divided into two areas surrounded with a blue and a red square. Moving objects are recognized when the sum of differences in pixel values between the current and previous frame is above a certain threshold.

To determine the number of users we use both Okao Vision and Motion Detector. Okao Vision has a function to determine the number of user faces, but it cannot detect a user’s departure because it fails in face detection when rotation of user head exceeds 60 degrees. Hence, we use Motion Detector to track movements in the area.

The number of users should be equal or less than two and they stand at positions decided beforehand. Therefore we can know the user existence in the area and the number of users by detecting the movement in region by using Motion Detector. The result of this function is the greater of the values for the number of people provided by Motion Detector and Okao Vision, respectively. This function has been implemented and tested.

Although the topics that the users are talking about serve as a very important cue for knowing whether the users are interested in the system, it is not practical to use speech recognition in our noisy real-world application. Therefore, we hypothesized that users’ interest can be approximated as the users’ attention toward the system during a relatively long interval of time, such as dozens of seconds. Attentions of the users can be approximated as the patterns at which users are looking at a high ratio of time. The meaningful patterns we consider are either the agent or other users. The attention of the users toward the system or the agent can also be discovered by tracking whether there is a change in the users’ activities immediately after a remarkable action done by the agent.

B. Central Part

1) Nonverbal Input Understanding

This component combines input data coming from Motion Detector and Okao’s Vision and uses simple heuristic methods to resolve the number of users and level of interest in the system. As Okao’s Vision fails in detection when users rotate their head beyond 60 degrees, it is important to save data in short time periods and combine it with information from Motion Detector. For example, two users talk/listen to the agent and the left user turns his head to see who entered the room behind him. In this situation Okao sends value of **UserNumber** variable set to one, as if there is just the right
user in the system. At the same time, Motion Detector detects motions in the left area, and Nonverbal Input Understanding component sets UserNumber value to two and sends it to the Decision Making Component.

2) Decision Making Component

At the beginning of the workshop we planned to implement a central part following the Information State Theory [2] [3], which is a general theory of human-agent dialogues. A dialogue based upon this theory is defined as a set of variables (or information) that describe the current state of the dialogue. According to the modeled dialogue, the information state is updated by dialogue moves that compose the agent’s and the user’s utterances. The agent’s dialogue strategies which trigger particular dialogue moves in response to current conditions that are caused by both of the user’s and the agent’s dialogue moves are defined as plans. The agent then behaves as a regular rational autonomous agent that chooses and executes plans to achieve its goals. This is the general idea of the dialogue theory; dialogue moves, dialogue plans and lexicons are user-defined and require intensive work. The prototype was being implemented based on one of the implementations of information state dialogue move engine [4] to be capable to deal with multi-modal, multi-party conversations, dynamically changing behaviors accompanying the emotion dynamics simulating component, etc.

However, due to lack of time to complete the dialogue manager by the end of the workshop, instead of a fully functional dialogue manager, a power-up version of existing GSML (Generic Embodied Conversational Agent Scenario Language) [5] script interpreter was developed.

Compared to the previous system which merely matches recognized speech inputs and non-verbal inputs with predefined patterns, a variable system is introduced. Following information state theory, interaction between the agent and one of two users is described with a set of variables like a snapshot. For example, SpeechInput represents the most recent result from one of the speech recognition components, Speaker represents the id of the speech recognition component, UserNumber represents the number of users who are standing in the user area, UserStatus represents the ability of the users, UserAttention represents how many of the users are paying attention to the system, Addressee specifies who should be the addressee of the agent’s next utterance, etc.

The values of these variables are updated with the agent system’s internal status and perception events sent from the speech recognition components and non-verbal input interpretation component. How the value of the variables should be updated can also be specified by the script designer in the script as the effects of particular input patterns. Effect element is introduced into Template element for this purpose. An input event can cause the values of particular variables to be bound to, added with, or subtracted from certain values.

The syntax of the patterns defined in GSML scripts is also extended. Predicate element is introduced to represent a test on the values of a variable. The value of the variables can be tested to be equal to, lesser or greater than certain values.

The chatbot-like ECA system is then extended to a more powerful rule-based autonomous system. The agent or the script execution engine updates its internal status variables via the perceptions from the outside world or the users, and picks the first valid template which made all of the conditions (predicates) true to perform. Therefore, the rules like that the tour guide agent should walk to the front to greet when there are users present in the user area, say goodbye to the user and go back to the initial position when the user has left the user area and so on can be specified in the script.

States limit possible patterns that will be used in matching the current conversation situation and thus isolates the interference from other states which may happen to have the same triggering patterns. Because of the absence of context management mechanism in the agent’s behavior control, there is no way to determine whether a user answer is related to the last question asked by the agent. However, for example, when the agent is going to ask a yes/no question like “Do you need a tour guide? “, a transition to a specific state representing the question can isolate the question from other yes/no questions.

GlobalState is introduced for error and interruption handling. When a failed or unknown recognition occurs, appropriate response will be searched among the categories defined in the global state. When interruptions from the user like “excuse me” or “pardon” occur, they are also matched with the patterns defined in this state.

Unlike a full dialogue-managing central component, the disadvantage of this approach is: the agent does not conduct a plan that contains multiple steps to achieve a certain goal. The agent’s behaviors are driven by the events that occurred in the outside world. The management mechanism for information like grounding or topics is not included in the script execution kernel. These features are still implementable but are left as script programmer’s responsibility.

During the period of eNTERFACE’08 workshop, only a few issues specific to multi-party conversation could be addressed. Because it is possible that there can be two users standing in front of the tour guide agent, the gaze direction of the agent becomes essential in three-party dialogue. Because we do not do natural language processing and have no way to understand the meaning of the conversation, our simplified model is: the agent should distribute different ratio of its gaze depending on who the addressee is. It will look at both users at the same frequency or look at the user standing at the left hand side more frequently if he is the addressee. Detailed gaze control is not done by the central component but is by the player. An Addressee attribute is introduced in the Utterance element to specify the addressee of this utterance to be left, right or both. Addressee specification is done by the rules in the scenario script by the script programmer; for example, this utterance of the agent should be directed towards the last speaker and so on.
C. Animation Player

The component which displays the city of Dubrovnik and tour guide agent Dubravka was developed by using Visage|SDK tool from Visage Technologies company [15]. As this product does not have support for virtual environments, we used a normal image as background. Position of the agent was calculated by using ARToolkit software from eNTERFACE '06 workshop [6]. Size of the agent was adjusted to large in order to efficiently address the user by gazing (Fig. 5).

![Example of agent's behaviours in Animation player](image)

**Figure 5. Example of agent's behaviours in Animation player**

Agent’s behaviors defined in system scenario are produced and tested by using GSML syntax [5] which synchronizes nonverbal behaviors (gestures, locomotion, facial expressions) with speech. Animations for the character were either modeled manually in 3ds Max or procedurally in Visage.

Gazing model runs on the utterance level and is controlled by the central part of the system according to the *Addressee* attribute. E.g. if the addressee is the left user, the agent will gaze first at the left side, then it will glance to the right for a while (if there are two users) and gaze back to the left user again. Since we cannot predict the duration of the utterance, we repeat patterns and stop when the utterance is finished.

Animation Player receives messages from Decision Making component and runs them in an appropriate way. We distinguish between four types of messages:

- **Utterances.** Contain GSML behavior description to be performed by an agent. After they have been received, they are put into queue.
- **Interruptions.** Messages which stop the agent from performing “planned” behaviors. When received, the behavior queue is cleaned.
- **Stop messages.** Force player to shut down and save current content.
- **Reset messages.** Reset the player to the initial state. This message is received when there are no users within the system scope.

It was also planned to add message priority within Utterance category to avoid unnecessary re-planning in case of interruptions like comments. Example is a situation when the agent is speaking and the user says “That’s very nice”. From the system input it is recognized as interruption, however in system output we treat it in this way: the agent stops with speech, responds to the user (“Yes, it is”), and then continues with the behavior which was running before the comment occurred. A level of “smart scheduling” according to the utterance priority should be achieved for this feature.

IV. Discussion and Conclusion

A. Summary of software produced.

A first prototype of ECA tour guide system which can communicate with two users has been developed. The system supports some of the multiparty aspects that were obtained from the literature and planned from the beginning of the workshop. The system can:

1. Recognize presence of people and invite them to the session
2. Resolve the number of users and their level of attention
3. Handle interruptions, such as the users’ requests, and respond to these requests by using the agent’s gaze direction
4. Recover from failure (e.g. if a user’s speech isn’t recognized)
5. Recognize users’ departure and automatically finish the session.

It was also planned to have a system with five different scenes, but as writing a scenario and manual modeling of the agent’s behavior was rather time-consuming, we focused only on first scene. Extension of the system on five scenes doesn’t require modification of existing components- it is enough to write a new script and, if necessary, add new animations to the player.

B. Towards system evaluation

Since some of the planned multiparty aspects were not yet implemented and tested, up to now no system evaluation has been performed.

After we have completed our work, the first thing to do is to see how our system acts in normal interaction with one and two humans (what lacks at the moment is that system doesn’t handle users’ mutual conversation). As it cannot “understand” natural language, we are in particular interested in learning what type (and level) of interaction can be achieved with this scenario. Of course, we want to improve it, if necessary and possible. Compared to the previous dyadic tour guide system which was only served for demo purposes, this system is several steps ahead.

On the system-level, evaluation involves three things: evaluation of user-ECA performance, e.g., how fluent and efficient it is, evaluation of the user experience from a...
subjective standpoint and evaluation of the effectiveness of the ECA-enabled application in achieving its goals [19]. At this point we are interested in the first and second type of evaluation. Usually, results of experiments are collected by using questionnaires or monitoring users’ bio-signals in interaction, such as gazing or speech. Overviews of relevant factors which influence human-character interaction are given in [18], and will serve us as general guidelines to set up an experimental system and choose an appropriate system study.

Compared to dyadic ECA systems our system has one strength from evaluation point of view: it provides a possibility to measure and compare human-human interaction and ECA-human interaction at the same time. Although the full system does not “allow” communication between humans, measurements can be simplified by observing attempts of humans to establish communication and neglect ECA. It would also be interesting to compare evaluation results to the recent work by Rehm [16], who performs evaluation study on an emotionally tense lying game. The agent in this system is even less “smart” than ours (it can understand some digits and “yes/no”-variants as well) and is treated in three different ways by human users: as a normal conversational partner, as an interlocutor and as an artifact. This is, however, expected, but the question is: what are actual variations among these categories and what are their values? What is the level of empathy and effectiveness which an ECA has to have to keep attention, and what is the duration of this attention? How to model an ECA which can be “interesting” for prolonged communication? Our system can serve as a useful test bed for further studies.

C. Conclusion

Implementing multiparty support in ECA systems partly relies on communication theory which has set up several bases (e.g. [9]). However, systems which can establish and maintain multiparty communication with two humans are scarce. During this workshop we attempted to implement a first prototype of our system which handles this type of communication. Although some of the tasks were not completed, first experiments in interaction with the system are encouraging. For example, the system can detect the number of users, distinguish local roles of users in a conversation and use ECA’s gazing direction to specify an addressee.

The weakest point in the system is the conversational model which is rather simple, but overall outcome of this workshop is just a step towards further experiments and studies in the field of multiparty communication in ECA systems.

ACKNOWLEDGEMENTS

This research is partially supported by the Ministry of Education, Science, Sports and Culture of Japan, Granfin-Aid for Scientific Research (S), 19100001, 2007.”Studies on Construction and Utilization of a Common Platform for Embodied Conversational Agent Research” and the Ministry of Science and Education of the Republic of Croatia, grant nr. 036-0362027-2028” Embodied Conversational Agents for Services in Networked and Mobile Environments.” We would also like to thank Dr. Lao Shihong from OMRON Corporation Research & Development Headquarters, regarding licensing of OKAO Vision.

REFERENCES


Aleksandra Ćerekovčič is a Ph.D. student at the Department of Telecommunications, Faculty of Electrical Engineering and Computing in Zagreb, Croatia. She obtained her diploma degree on audio-visual speech processing (lip-sync) in year 2006 from the same University. Currently she is working on topics of body animation and control for Embodied Conversational Agents (ECAs). Her research interests are: body communication, gesture theory, animation modeling, and computational learning.

Hung-Hsuan Huang graduated from the Computer Science Department of National Chen-Chi University, Taiwan in 1998 and obtained his master degree of computer science and information engineering from National Taiwan University, Taiwan in 2000. After a two-year military service where he was the political warfare director and the co-commander of an army company, he went abroad to Japan. After the learning in a Japanese language school for one year, he entered the Ph.D. course of the Graduate School of Informatics of Kyoto University, Japan in 2003. His research interests include intelligent software agent, information visualization, photo management, gesture interface.

Takuya Furukawa (Japan, October 1984) graduated from the Human and AI Systems Department of Fukui University, Japan in 2007 and entered the master course of the Graduate School of Informatics of Kyoto University, Japan in 2007. His research interests include multimodal conversational interfaces, and human behaviors in communication with embodied conversational agents. He has an experience of fellowship job in Works Applications is Japanese ERP software company. Mr. Furukawa is a member of JSAI (Japanese Society for Artificial Intelligence).

Yuji Yamaoka is a student of a Master course program of Ubiquitous and Universal Information Environment, at the Graduate School of Informatics, Tokyo University of Agriculture and Technology, Japan. His research interests include culture in nonverbal behaviors of ECAs.

Toyoaki Nishida is a professor of Department of Intelligence Science and Technology, Graduate School of Informatics, Kyoto University. He received the Doctor of Engineering degree from Kyoto University in 1984. His research centers on artificial intelligence and human computer interaction. In 2001, he founded a series of international workshops on social intelligence design (see http://www.iisti.ist.i.kyoto-u.ac.jp/sid for more details). Then, he broadened the scope of research to include understanding and augmenting conversational communication, and opened up a new field of research called Conversational Informatics. Currently, he leads several projects on social intelligence design and conversational informatics. He is a member of the board of directors of IPS (Information Processing Society) of Japan and JSAI (Japanese Society for Artificial Intelligence). He serves as an editorial board member of several academic journals, including Web Intelligence and Agent Systems, AI & Society, and Journal of JSAI (editor-in-chief).

Associate Prof. Igor S. Pandžič received his BSc degree in Electrical Engineering from the University of Zagreb in 1993, and MSc degrees from the Swiss Federal Institute of Technology (EPFL) and the University of Geneva in 1994 and 1995, respectively. He obtained his PhD from MIRALab, University of Geneva, Switzerland in 1998. In the same year he worked as a visiting scientist at AT&T Labs, USA. In 2001-2002 Igor was a visiting scientist in the Image Coding Group at the University of Linköping, Sweden, and in 2005 visiting researcher at Kyoto University, Graduate School of Informatics, Nishida & Sumi Lab. He is now an Associate Professor at the Department of Telecommunications, Faculty of Electrical Engineering and Computing, University of Zagreb, Croatia. His main research interests are in the field of computer graphics and virtual environments, with particular focus on facial animation, embodied conversational agents, and their applications in networked and mobile environments. Igor also worked on networked collaborative virtual environments, computer generated film production and parallel computing. Igor was one of the key contributors to the Facial Animation specification in the MPEG-4 International Standard for which he received an ISO Certificate of Appreciation in 2000.

Asst. Prof. Yukiko Nakano received her bachelor degree in psychology from Tokyo Women’s Christian University, Japan in 1988 and one master degree in educational psychology from the University of Tokyo and one in media arts and sciences from Massachusetts Institute of Technology, USA in 1990 and 2002, respectively. She obtained her Ph.D. in information science and technology from the University of Tokyo, Japan in 2005. Yukiko was a researcher of NTT Research Laboratories from 1990 to 2000 and was a sub-leader researcher of Research Institute of Science and Technology for Society from 2002 to 2005. She was working at the Department of Computer, Information and Communication Sciences of Tokyo University of Agriculture and Technology as an associate professor from 2005 to 2008. From summer 2008 she has been working at Dept. of Computer and Information Science Faculty of Science and Technology, Seikei University. With her special interest in Embodied Conversational Agents (ECA), she has been studying human face-to-face communication in psychology and communication science, and creating multimodal conversational interfaces based on a model of human communication behaviors.