D4AC: A Tool for Developing Multimodal Dialogue Systems without Coding

Mikio Nakano[*]
Institute of Innovation for Future Society
Nagoya University
Nagoya, Aichi, Japan
mikio.nakano@c4a.jp

Ryuichiro Higashinaka
Graduate School of Informatics
Nagoya University
Nagoya, Aichi, Japan
higashinaka@i.nagoya-u.ac.jp

Abstract

To enable the broader application of dialogue system technology across various fields, it is beneficial to empower individuals with limited programming experience to build dialogue systems. Domain experts, where dialogue system technology is highly relevant, may not necessarily possess expertise in information technology. This paper presents D4AC, which works as a client for text-based dialogue servers. By combining D4AC with a no-code tool for developing text-based dialogue servers, it is possible to build multimodal dialogue systems without coding. These systems can adapt to the user's age, gender, emotions, and engagement levels obtained from their facial images. D4AC can be installed, launched, and configured without technical knowledge. D4AC was used in student projects at a university, which suggested the effectiveness of D4AC.

1 Introduction

Recent advancements in large language models have made it possible to develop various kinds of dialogue systems easily. In applying dialogue systems to various fields, it is effective for dialogue system engineers to cooperate with experts in the fields. However, there is a problem that the resources of dialogue system engineers are limited. Therefore, it would be desirable that experts in those fields could construct a dialogue system without coding.

Although there are tools for building text-based dialogue systems without coding, there have been no no-code tools for building multimodal dialogue systems that can exploit the user's attributes and social signals (Vinciarelli et al., 2009) obtained from the user's images. Such systems are expected to engage in communication better than text-based systems, as they can adapt their communication strategies to the user's attributes and states.

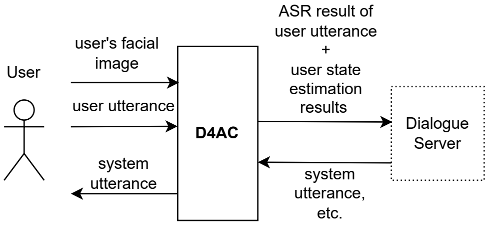


Figure 1: The role of D4AC.

This paper describes the design and implementation of D4AC,[1] which works as a client of a text-based dialogue server (Figure 1). The dialogue server is supposed to receive user utterances in text format and return system responses in text format. D4AC recognizes user utterances and estimates the user's age, gender, emotions, and engagement (referred to as "user states"[2]) from camera images using external APIs. It sends this information to the dialogue server at the time specified in the configuration. Then, it uses speech synthesis to generate the responses from the dialogue server.

The advantages of D4AC are as follows:

- It enables the dialogue server to use estimated user states for dialogue management.
- It enables the dialogue server to utilize estimated user states at times different from the ends of user utterances without writing complicated dialogue management knowledge. (This will be explained in detail in Section 3.2.)
- It can be easily installed, configured, and launched without technical knowledge.

---

[*] Also affiliated with C4A Research Institute, Inc., Setagaya, Tokyo, Japan.

[1] D4AC stands for 'Dialogue for All – Client', which represents a client program used in a project aimed at enabling anyone to build dialogue systems.

[2] Strictly speaking, age and gender should be referred to as attributes rather than states, but for simplicity, this paper will refer to them as states.

Proceedings of the 15th International Workshop on Spoken Dialogue Systems Technology, pages 183–189
May 27–30, 2025. ©2025 Association for Computational Linguistics

- It is released as an open-source software under the Apache License ver. 2.0.[3]

2 Related Work

There are studies on dialogue systems that handle information obtained from the user's image. For example, there are studies on a dialogue system that recognizes affirmative/negative attitudes from prosody and head movements (Fujie et al., 2006) and a reception dialogue system that smoothly facilitates turn-taking by estimating user engagement from standing positions and recognizing turn-yielding or taking intentions from the gaze and gestures (Bohus and Horvitz, 2009, 2011). An interview dialogue system that changes dialogue strategies by estimating the willingness to speak from gestures and speech (Ishihara et al., 2018) is also studied. SimSensei Kiosk (DeVault et al., 2014) is a system that conducts interviews for PTSD diagnostic support, estimating the user's emotions from facial expressions, head movements, gaze, and prosody.

While there are various studies in the field, there are few tools available as off-the-shelf solutions. '$\backslash psi$' (Bohus et al., 2021) and Virtual Human Toolkit (Hartholt et al., 2022, 2013) are exceptions, but technical knowledge is required to use them. In the Dialogue System Live Competition 6 (Higashinaka et al., 2024), a tool that enables the development of dialogue systems using user state estimation results without coding was distributed to the participants, but, unlike D4AC, it can send the user states to the dialogue server only at the end of user utterances, and it is not publicly available.

3 Design of D4AC

3.1 Overview

D4AC works as a client of a dialogue server (Figure 1). D4AC recognizes user utterances and sends the results to the dialogue server, along with user states (engagement, emotions, age, and gender) estimated from the user's facial image. The dialogue server determines the system utterance based on those inputs and dialogue context and returns it to D4AC in text format. D4AC then generates the system utterance using speech synthesis.

For user state estimation, D4AC uses Face++.[4] If only estimating engagement, it is also possible to use FaceMesh of MediaPipe.[5] For speech recognition and synthesis, it utilizes the Web Speech API of Google Chrome. Alternatively, Amazon Polly[6] can be used for speech synthesis.

D4AC can work as a client of dialogue servers that can be built without coding. It assumes NTT Docomo's xAIML SUNABA (SUNABA hereafter)[7] and DialBB[8] (Nakano and Komatani, 2024) as tools for building and deploying dialogue servers. They provide GUI-based editors for state transition networks to make it possible to write dialogue scenarios without coding. They also enable running dialogue servers as HTTP servers.

3.2 Communications between D4AC and a Dialogue Server

D4AC can send the results of user state estimation to the dialogue server at the following times.

(1) When sending the speech recognition result for a user utterance.

(2) When finished generating a system utterance.

(3) When the user has not spoken for t seconds after a system speech has ended.

Whether or not to send at each of these times and t in (3) can be configured with the configuration GUI. The reason for allowing the system to send not only at (1) but also at (2) and (3) is to allow the system to make utterances based on the user's reactions to what the system has said. When sending to SUNABA, in the case of (1), the following form is used.[9]

```
tuna sandwich+{engagement:high}{emotion:
happiness}
```

Here "tuna sandwich" is the recognized user utterance. Which user states to send can be specified in the configuration. The value of engagement is high, middle, or low; the value of emotion is anger, disgust, fear, happiness, neutral, sadness, or surprise; the value of age is child, teenager, young, middle, senior, or unknown, and gender is either male or female. By transmitting user states in this manner, the dialogue server can perform state transitions according to the user states. For example,

---

[3] `https://github.com/nu-dialogue/D4AC`

[4] `https://www.faceplusplus.com`

[5] `https://chuoling.github.io/mediapipe/`

[6] `https://aws.amazon.com/polly/`

[7] `https://docs.sunaba.docomo.ne.jp/`. Currently, SUNABA can deal only with Japanese.

[8] `https://github.com/c4a-ri/dialbb`

[9] In the case of DialBB, a JSON format is used to represent user states.

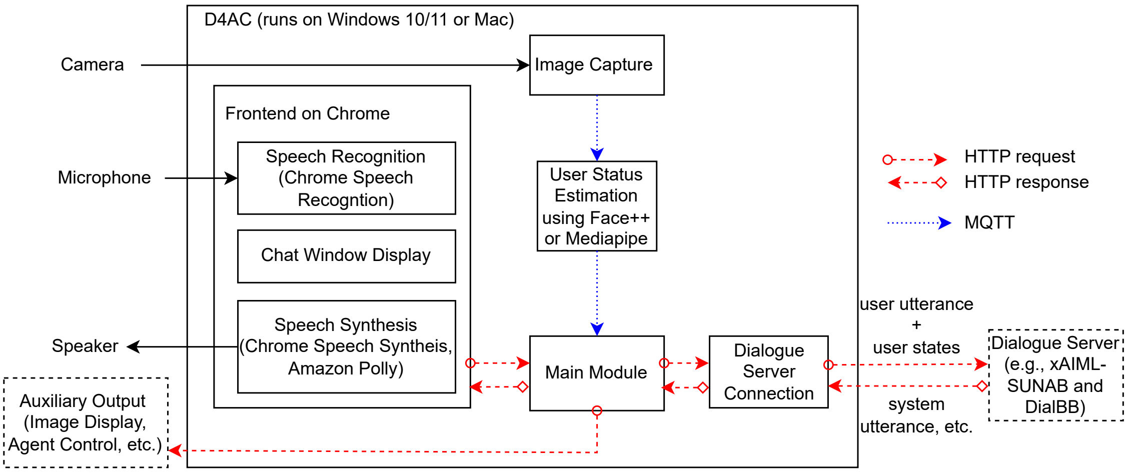

D4AC (runs on Windows 10/11 or Mac)
Camera
Image Capture
Frontend on Chrome
Microphone
Speech Recognition (Chrome Speech Recogntion)
User Status Estimation using Face++ or Mediapipe
HTTP request
HTTP response
MQTT
Chat Window Display
user utterance + user states
Speaker
Speech Synthesis (Chrome Speech Synthesis, Amazon Polly)
Main Module
Dialogue Server Connection
Dialogue Server (e.g., xAIML-SUNAB and DialBB)
system utterance, etc.
Auxiliary Output (Image Display, Agent Control, etc.)

Figure 2: Architecture of D4AC.

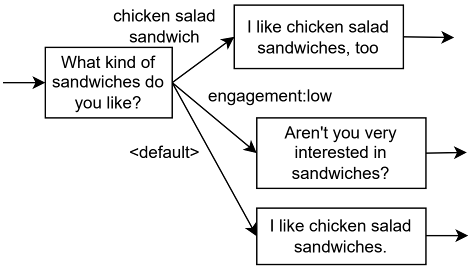

chicken salad sandwich
I like chicken salad sandwiches, too
What kind of sandwiches do you like?
engagement:low
Aren't you very interested in sandwiches?
<default>
I like chicken salad sandwiches.

Figure 3: A dialogue state transition network using user states. Rectangles represent dialogue states, and the texts inside them are system utterances in those states. Arrows indicate state transitions. The text near an arrow means that if the user utterance includes this text, the transition will follow that arrow. `<default>` indicates that the transition can occur with any user utterance. The actual description style depends on the dialogue server.

using the network shown in Figure 3, it can transition to a different dialogue state when user engagement is low.

In the cases of (2) and (3), the following forms are sent, respectively.

```
su_end+{engagement:high}{emotion:happiness}
silence+{engagement:high}{emotion:happiness}
```

In the cases of (2) and (3), the system does not necessarily have to respond. In such cases, the system utterance should be specified as "empty", which D4AC does not send to the speech synthesizer.

One issue arises here. The dialogue server is designed to transition states whenever there is an input of user utterance. Therefore, it is necessary

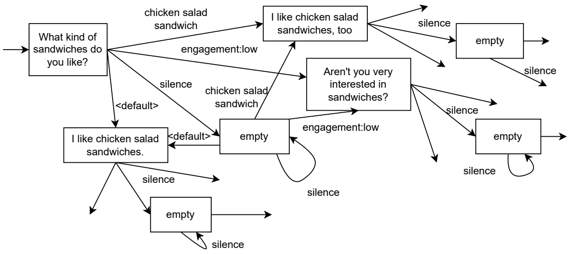


Figure 4: A state transition network for a client that cannot handle requests to send user states.

to describe state transitions for cases (2) and (3) as well, meaning that for every dialogue state, one must write the transitions that would occur when only the user states are input. For instance, if we decide to send the user states at the time of (3), a state transition network like Figure 4 is necessary, and a huge effort for writing networks is required.

Therefore, we made it possible for the dialogue server side to send user-state requests to D4AC. This request asks D4AC to send the user state at the times of (2) and (3) only when the user state becomes a specific value. Specifically, the request is appended to the end of the system utterance text in the form of [⟨list of types⟩:⟨list of values⟩]. The lists of types and values are concatenated with '|'. For example, let us consider the case the system utterance is like the following.

```
How are you feeling? [emotion:
happiness|sadness]
```

Then, in the cases of (2) and (3), the user state is sent to the dialogue server only when the emotion becomes either happiness or sadness. This

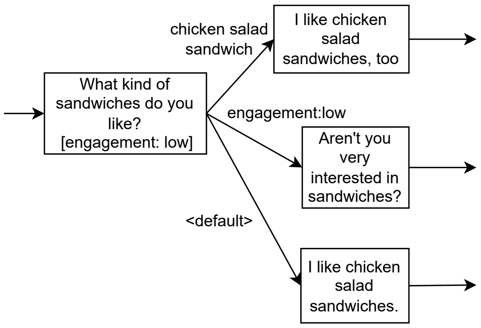

chicken salad sandwich
I like chicken salad sandwiches, too
What kind of sandwiches do you like? [engagement: low]
engagement:low
Aren't you very interested in sandwiches?
<default>
I like chicken salad sandwiches.

Figure 5: A state transition network that requests the client to send user states.

simplifies the state transition network as shown in Figure 5.

4 Implementation

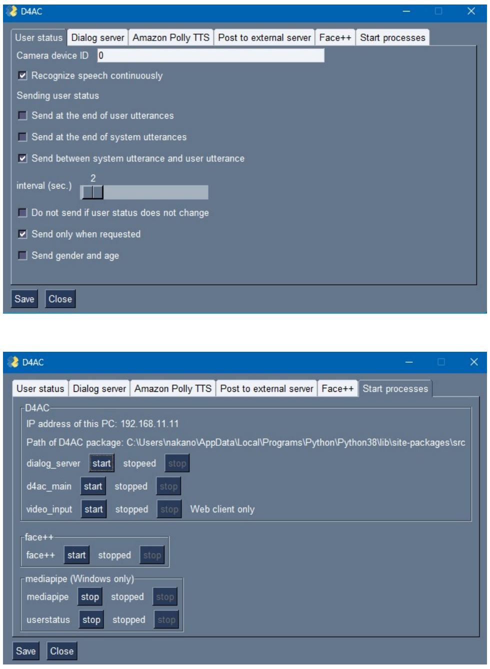
Here we explain the implementation details of D4AC according to Figure 2.

The modules for user state estimation communicate with each other using MQTT,[10] which enables lightweight, asynchronous communication. The image capture module captures camera images and sends them to the MQTT broker. User state is estimated from these images using services like Face++ and others. In the current version of D4AC, engagement is estimated using only the face directions obtained from Face++ or MediaPipe. We plan to replace this with a more accurate algorithm using facial expressions and gaze (De Carolis et al., 2019; Ishii et al., 2013) in the future. Adding and replacing modules are easy thanks to the MQTT-based architecture.

The dialogue server connection module sends the results of speech recognition and user state estimation to the dialogue server and returns the system utterance received in response to the main module. If the system utterance includes a user-state transmission request, it is remembered and used to decide whether to send the user state at the times of (2) and (3) in Section 3.2.

Additionally, if specified, D4AC can send the system utterance text strings received from the dialogue server to an external auxiliary output server. By embedding commands within the system utterance text, it becomes possible to display images or to operate CG agents or robots.

The frontend part is written in JavaScript, and other modules are written in Python.



Camera device ID 0
☑ Recognize speech continuously
Sending user status
☐ Send at the end of user utterances
☐ Send at the end of system utterances
☑ Send between system utterance and user utterance
interval (sec.) 2
☐ Do not send if user status does not change
☑ Send only when requested
☐ Send gender and age

D4AC
IP address of this PC: 192.168.11.11
Path of D4AC package: C:\Users\nakano\AppData\Local\Programs\Python\Python38\lib\site-packages\src
dialog_server start stopeed stop
d4ac_main start stopped stop
video_input start stopped stop Web client only
face++
face++ start stopped stop
mediapipe (Windows only)
mediapipe stop stopped stop
userstatus stop stopped stop

Figure 6: Configuration GUI. The tab for configuring sending the user states (upper panel) and the tab for starting processes (lower panel).

4.1 Installation and Configuration

D4AC can be installed easily by just invoking "`pip install`" command using a `.whl` file. Then running the "`d4ac`" command in a command prompt displays the configuration GUI. With the GUI, developers can specify the dialogue server, access information to Face++, Amazon Polly, etc., the times of sending user state estimation, and others (Figure 6 (upper panel)).

In addition, it is possible to start and stop the processes in D4AC from the GUI (Figure 6 (lower panel)).

5 Sample Application

This section describes a simple example application. An example dialogue between this application and a user is shown in Figure 7.

The dialogue server is built using DialBB. The scenario involves light conversations about food. In this scenario, ChatGPT (gpt-4o[11]) is used to generate system utterances in some states. The third, fourth, and sixth system utterances in Figure 7 were generated by ChatGPT. The configuration of the DialBB application includes the situations and persona of the system.

---

[10] https://mqtt.org/

[11] https://openai.com/index/hello-gpt-4o/

Speaker	Utterance (*speech recognition result*)	Remarks
System	Hello I'm a chatbot. If you don't mind, could you tell me your name?	
User	Hello, I'm Mike. (*oh, I'm Mike*)	
System	Thank you Mike! Let me ask you about sandwich. Do you have sandwiches very often?	
User	Yes, I like sandwiches very much. (*yes I like sandwiches very much*)	
System	What's your favorite kind of sandwich, Mike? I love a good BLT, but I'm always looking for new ideas to try!	The dialogue server requests D4AC to tell if the user engagement level is low or middle at the end of and two seconds after the system utterance.
User	I like chicken salad sandwiches. (*I got chicken salad sandwiches*)	
System	Chicken salad sandwiches sound delicious! Do you make them yourself or do you have a favorite place to get them?	The dialogue server requests D4AC to tell if the user engagement level is low or middle at the end of and two seconds after the system utterance. At the end of the system utterance, D4AC informs the dialogue server the user engagement level is low.
System	Aren't you interested?	The system asks the user if she/he is interested in the conversation.
User	Oh, yes. I'm interested. Sorry. (*oh yes I'm interested sorry*)	
System	No worries, Mike! Do you add anything special to your chicken salad sandwiches, like grapes or nuts, or do you prefer to keep it simple?	Since the user response is affirmative, the system goes back to the conversation. (Otherwise, the system wraps up the conversation.)

Figure 7: A sample dialogue between a user and the example application.

At some states, the dialogue server requests D4AC to send the user's engagement level if it is "middle" or "low". Then when the dialogue server receives the engagement level, it makes an utterance to ask if the user is bored with the conversation, and if so, the conversation is terminated.

D4AC is configured to send the user's state to the dialogue server only when requested by the dialogue server. This occurs either at the end of the system's utterance or if the user does not speak for two seconds after the system's utterance has ended.

The source files for this sample application are publicly available as a part of the D4AC repository.

6 Using D4AC in Student Projects

During the development of D4AC, we utilized it in student projects at a university. The student projects were conducted twice, in the academic years 2021 and 2022, where students were tasked with building simple multimodal dialogue systems. Four students participated in each project.

In Academic Year 2021, the four students built separate dialogue systems following their interests. Specifically, these were (1) a system for scheduling appointments, (2) a system for presenting quizzes, (3) a system explaining the use of shared cars, and (4) a system providing useful information for LGBTQ+ individuals. The student who developed the fourth system was affiliated with a humanities research lab, while the others belonged to information technology or engineering labs. Of these, systems (3) and (4) utilized D4AC along with SUNABA, but they did not use user state estimation results. At that time, D4AC could only send user state information at the end of user utterances, which limited its utility. Following this experience, we modified D4AC to transmit user state estimation results at times other than the end of user utterances.

In Academic Year 2022, the project involved four students working collaboratively to build a single system. Among these students, two were from humanities and two were from information technology or engineering labs. They developed a dialogue system designed to calm the irritations of drivers while driving. The system converses with the driver, calming them if they are irritated, and the progression is determined based on the user state estimation results indicating whether the irritation is resolved. Insights from humanities research (e.g., (Nakai, 2021)) were utilized in creating the scenarios. This system used DialBB to build the dialogue server.

This project showed that a system utilizing user state estimation results can be easily built with D4AC. We plan to provide opportunities for people from various fields to use the system and validate its effectiveness in future projects.

7 Concluding Remarks

This paper described D4AC, which is a tool that allows for the construction of multimodal dialogue systems without coding. It estimates user attributes and states from facial images and alters the course of dialogue based on these results. D4AC enables people from various fields, not just information technology, to build multimodal dialogue systems, contributing to the integration of dialogue system technology with other disciplines.

Although D4AC does not include novel technologies that improve empirical evaluation results, it concretizes a reasonable architecture for creating a multimodal dialogue system without coding. We believe that it will contribute to multimodal dialogue system technologies being used in various fields.

We are considering the following enhancements and features. First, we aim to enable the use of tools and APIs other than Face++ and MediaPipe for user state estimation. Another goal is to integrate prosodic information from user utterances and text-based data for emotion estimation. Additionally, we are considering the use of other user states such as sentiment (Katada et al., 2023) and affirmative/negative attitudes (Fujie et al., 2006). We also think that multimodal foundational models such as GPT-4o might be useful for user state estimation.

As D4AC is released as an open source, we hope we will receive feedback for improvements while it is being used in an actual system development.

Limitations

Although D4AC was used in student projects, it has not been evaluated in a way that involves gathering a large number of users. Therefore, its usability remains unclear. However, the value of such tools cannot be measured solely through evaluations in controlled situations. It is important to release it publicly, allowing it to be used in the development of various systems, and continuously improve it based on feedback. D4AC is believed to have sufficient performance as a starting point for this process.

Ethical Considerations

One potential ethical issue with this tool is the risk of personal information leakage, as users' facial images and voice data are sent to image and voice recognition systems managed by private companies' cloud services. Whether this becomes a problem depends on the application. Therefore, we encourage developers using D4AC to raise awareness and recommend obtaining proper consent from users.

Estimating emotions from facial images and linking them to the content of conversations is important for facilitating more appropriate interactions. However, depending on how it is used, there is a risk of obtaining personal inner feelings that the user has not intentionally disclosed; it might be illegal in some countries. We also raise awareness among developers using D4AC about this issue.

Acknowledgments

The development of D4AC was supported by Nagoya University's the Graduate Program for Lifestyle Revolution based on Transdisciplinary Mobile Innovation (TMI) program. We thank Prof. Nobuo Kawaguchi, the coordinator of this program. We also thank Mizuho Aoki, Xueqin Huang, Ryo Hori, Sanae Yamashita, Ai Liya, Ibrahima Kane, Qiuhao Cui, Ailing Xie of the TMI program for participating in the dialogue system project and for providing valuable feedback on D4AC.

References

Dan Bohus, Sean Andrist, Ashley Feniello, Nick Saw, Mihai Jalobeanu, Patrick Sweeney, Anne Loomis Thompson, and Eric Horvitz. 2021. Platform for situated intelligence. *CoRR*, abs/2103.15975.

Dan Bohus and Eric Horvitz. 2009. Learning to predict engagement with a spoken dialog system in open-world settings. In *Proceedings of the SIGDIAL 2009 Conference*, pages 244–252, London, UK. Association for Computational Linguistics.

Dan Bohus and Eric Horvitz. 2011. Multiparty turn taking in situated dialog: Study, lessons, and directions. In *Proceedings of the SIGDIAL 2011 Conference*, pages 98–109, Portland, Oregon. Association for Computational Linguistics.

Berardina De Carolis, Francesca D'Errico, Nicola Macchiarulo, and Giuseppe Palestra. 2019. "engaged faces": Measuring and monitoring student engagement from face and gaze behavior. In *IEEE/WIC/ACM International Conference on Web Intelligence - Companion Volume*, WI '19 Companion, page 80–85, New York, NY, USA. Association for Computing Machinery.

David DeVault, Ron Artstein, Grace Benn, Teresa Dey, Ed Fast, Alesia Gainer, Kallirroi Georgila, Jon

Gratch, Arno Hartholt, Margaux Lhommet, Gale Lucas, Stacy Marsella, Fabrizio Morbini, Angela Nazarian, Stefan Scherer, Giota Stratou, Apar Suri, David Traum, Rachel Wood, Yuyu Xu, Albert Rizzo, and Louis-Philippe Morency. 2014. Simsensei kiosk: a virtual human interviewer for healthcare decision support. In *Proceedings of the 2014 International Conference on Autonomous Agents and Multi-Agent Systems*, AAMAS '14, page 1061–1068, Richland, SC. International Foundation for Autonomous Agents and Multiagent Systems.

Shinya Fujie, Yasushi Ejiri, Hideaki Kikuchi, and Tetsunori Kobayashi. 2006. Recognition of positive/negative attitude and its application to a spoken dialogue system. *Systems and Computers in Japan*, 37(12):45–55.

Arno Hartholt, Ed Fast, Zongjian Li, Kevin Kim, Andrew Leeds, and Sharon Mozgai. 2022. Rearchitecting the virtual human toolkit: towards an interoperable platform for embodied conversational agent research and development. In *Proceedings of the 22nd ACM International Conference on Intelligent Virtual Agents*, IVA '22, New York, NY, USA. Association for Computing Machinery.

Arno Hartholt, David Traum, Stacy C. Marsella, Ari Shapiro, Giota Stratou, Anton Leuski, Louis-Philippe Morency, and Jonathan Gratch. 2013. All together now. In *Intelligent Virtual Agents*, pages 368–381, Berlin, Heidelberg. Springer Berlin Heidelberg.

Ryuichiro Higashinaka, Tetsuro Takahashi, Michimasa Inaba, Zhiyang Qi, Yuta Sasaki, Kotrao Funakoshi, Shoji Moriya, Shiki Sato, Takashi Minato, Kurima Sakai, Tomo Funayama, Masato Komuro, Hiroyuki Nishikawa, Ryosaku Makino, Hirofumi Kikuchi, and Mayumi Usami. 2024. Dialogue system live competition goes multimodal: Analyzing the effects of multimodal information in situated dialogue systems. In *Proceedings of IWSDS*.

Takuya Ishihara, Katsumi Nitta, Fuminori Nagasawa, and Shogo Okada. 2018. Estimating interviewee's willingness in multimodal human robot interview interaction. In *Proceedings of the 20th International Conference on Multimodal Interaction: Adjunct*, ICMI '18, New York, NY, USA. Association for Computing Machinery.

Ryo Ishii, Yukiko I. Nakano, and Toyoaki Nishida. 2013. Gaze awareness in conversational agents: Estimating a user's conversational engagement from eye gaze. *ACM Trans. Interact. Intell. Syst.*, 3(2).

Shun Katada, Shogo Okada, and Kazunori Komatani. 2023. Effects of physiological signals in different types of multimodal sentiment estimation. *IEEE Transactions on Affective Computing*, 14(3):2443–2457.

Hiroshi Nakai. 2021. The overview of research on aggressive driving and proposal of preventive measures. *Traffic sciences*, 52(1):3–12. (in Japanese).

Mikio Nakano and Kazunori Komatani. 2024. DialBB: A dialogue system development framework as an educational material. In *Proceedings of the 25th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 664–668, Kyoto, Japan. Association for Computational Linguistics.

Alessandro Vinciarelli, Maja Pantic, and Hervé Bourlard. 2009. Social signal processing: Survey of an emerging domain. *Image and Vision Computing*, 27(12):1743–1759.