ROSAnnotator: A Web Application for ROSBag Data Analysis in Human-Robot Interaction

Yan Zhang
University of Melbourne
Melbourne, Australia
yan.zhang.1@unimelb.edu.au

Haoqi Li
University of Melbourne
Melbourne, Australia
haoqi2@student.unimelb.edu.au

Ramtin Tabatabaei
University of Melbourne
Melbourne, Australia
stabatabaeim@
student.unimelb.edu.au

Wafa Johal
University of Melbourne
Melbourne, Australia
wafa.johal@unimelb.edu.au

Abstract—**Human-robot interaction (HRI) is an interdisciplinary field that utilises both quantitative and qualitative methods. While ROSBags, a file format within the Robot Operating System (ROS), offer an efficient means of collecting temporally synched multimodal data in empirical studies with real robots, there is a lack of tools specifically designed to integrate qualitative coding and analysis functions with ROSBags. To address this gap, we developed ROSAnnotator, a web-based application that incorporates a multimodal Large Language Model (LLM) to support both manual and automated annotation of ROSBag data. ROSAnnotator currently facilitates video, audio, and transcription annotations and provides an open interface for custom ROS messages and tools. By using ROSAnnotator, researchers can streamline the qualitative analysis process, create a more cohesive analysis pipeline, and quickly access statistical summaries of annotations, thereby enhancing the overall efficiency of HRI data analysis. https://github.com/CHRI-Lab/ROSAnnotator**

Index Terms—**Human-Robot Interaction, Data Analysis, Web Application, LLM**

I. Introduction

Human-robot interaction (HRI) is a field that acknowledges the importance of empirical study, which involves the collection of multimodal data to gain insights into human behaviour and interactions with robots. Given that many measurements cannot be simply quantified, qualitative methods are critical and typical research approaches in HRI [1], [2]. Conducting qualitative analysis requires recording various types of data during experiments, including but not limited to video, audio, and robot actions. This process often involves using multiple devices for different purposes and necessitates careful time synchronization, which can be challenging. In HRI studies with a real robot, ROSBag offers an effective solution to streamline the recording and synchronization of such data.

ROSBag is a powerful tool within the Robot Operating System (ROS) [3] that allows for the recording of message data, offering researchers the flexibility to customise the types of data captured during a study. This data can be automatically synchronised and easily replayed for further analysis. Previous studies have advanced the development of ROS tools to support HRI studies. For instance, HRItk [4] is a toolkit designed to provide speech, gesture, and gaze recognition topics. Another study [5] introduced the ROS4HRI framework, which is open-sourced and includes new message types specifically designed for HRI-related topics, along with a set of conventions and standard interfaces. These contributions enriched the types of data that can be collected using ROSBag. However, despite these advancements in data collection, the analysis of qualitative data remains a challenge.

Annotation, or coding, is an essential process for methods such as observation and interview, which are the two most frequently used approaches in HRI qualitative research [1]. The annotation of observational data is selective and highly dependent on the study's purpose [6]. Traditionally, annotation has been conducted manually, involving tasks like coding participants' utterances and emotions [7], often utilizing established tools such as ELAN [8] and ATLAS.ti [9]. Moreover, to ensure objectivity and reliability, this process typically requires the involvement of multiple researchers and the assessment of inter-rater reliability [10]. Therefore, data analysis is often time-consuming and labour-intensive.

To fasten the qualitative pre-processing, recent work in human-computer interaction has been exploring ways to automate the coding of data. For instance, prior studies have successfully captured kinematic features from videos to interpret actions and gestures, thereby facilitating qualitative analysis [11]. In addition, facial emotion recognition has been extensively investigated [12]. With the advent of large language models (LLMs) and their advanced capabilities in processing text-based data, some researchers are now exploring their usage for deductive coding, achieving higher inter-rater reliability with experts by using a pre-defined codebook [13]. However, these methods are not integrated and lack support for HRI researchers who are dealing with multimodal and robot logs packaged in ROSBags.

Therefore, we present ROSAnnotator, a web application (WebApp) designed for HRI qualitative data analysis that is particularly suited for multimodal data. It supports both manual annotation and the multimodal Large Language Model (LLM)-facilitated annotation, which makes the analysis process more efficient. Additionally, ROSAnnotator supports cross-modal annotation, allowing users to annotate across different modalities, such as using audio data to inform video content annotation. The primary features (see Figure 1) of ROSAnnotator include:

1) Extracting messages from ROSBags and exporting annotation results.

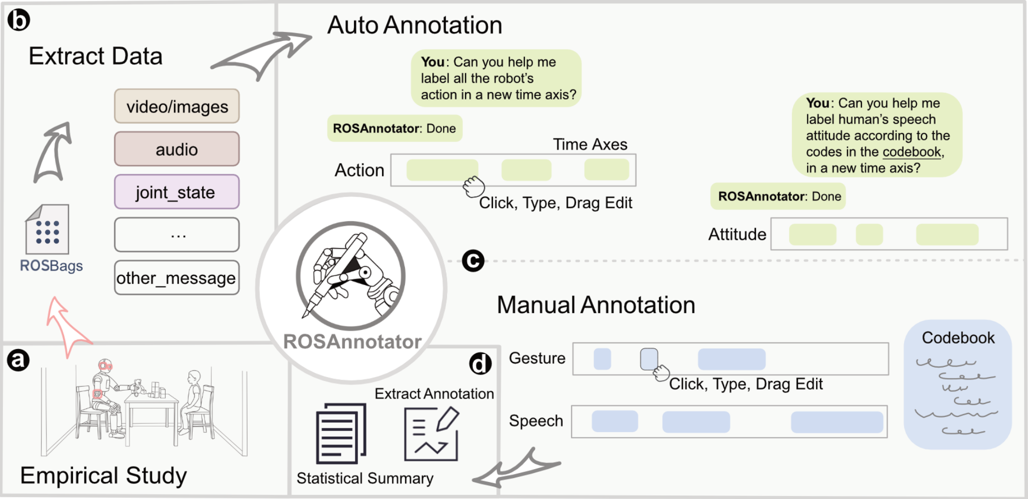


Fig. 1. This figure illustrates the workflow of ROSAnnotation. (a) The process begins with users collecting ROSBags during an empirical study. (b) Once the ROSBags are imported into ROSAnnotator, multimodal data is extracted. (c) ROSAnnotator supports both automatic and manual annotation. For automatic annotation, users can provide instructions via a chatbox, and ROSAnnotator will perform the annotations accordingly. Users can then make manual adjustments if needed. For manual annotation, users can refer to the codebook, create multiple time axes, generate annotations, and modify the time intervals for annotations. (d) Finally, users can view a statistical summary of their qualitative analysis and extract the resulting annotations.

2) Displaying video, time axes for multiple annotation tiers, a multimodal LLM chatbox, and other toolbars within a web interface.
3) Automatically transcribing audio and presenting it as annotations along a time axis.
4) Allowing for a customised codebook and manual annotation of multimodal data on separate annotation tiers.
5) Allowing for automated annotation, with codes generated on time axes by the multimodal LLM.
6) Providing interfaces for integrating other customised messages and tools.
7) Offering statistical summary for all annotations.

II. ROSAnnotator Functions

The user journey begins with the process of data import (see Figure 2a).

Data Import: Users can select the desired ROSBag from a list to begin their work. It is also recommended that a predefined codebook in JSON format be uploaded to streamline the annotation process. Additionally, if audio transcription is required, users have the option to transcribe the audio simultaneously.

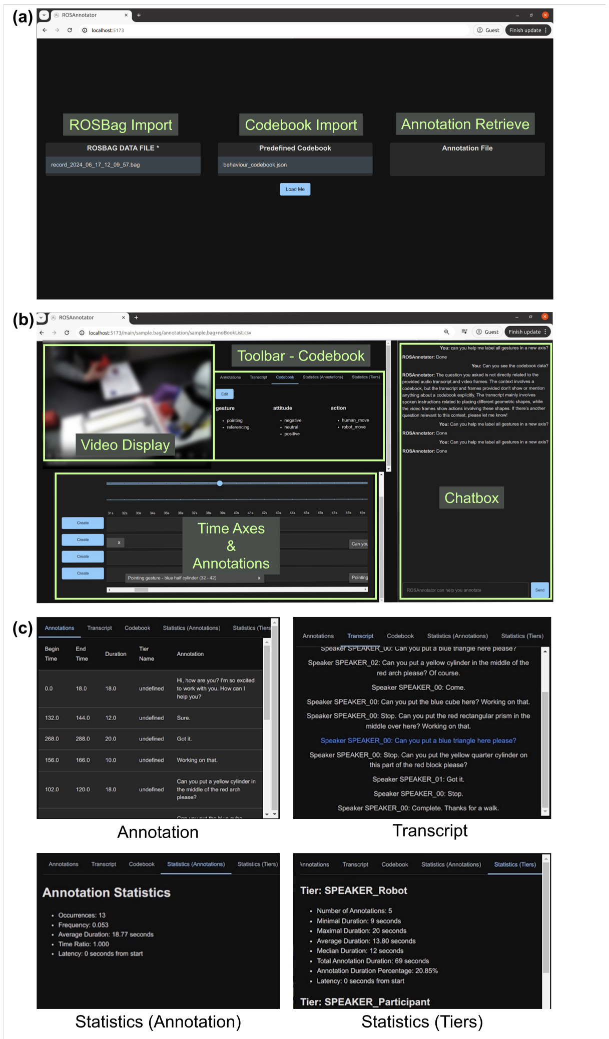
The main annotation interface is shown as Figure 2b.

Data Visualisation: The interface features a video player located in the top left corner, which synchronises with audio playback if audio extraction has been performed. In the top middle section of the interface, a toolbar *see Figure 2b and c) allows users to conveniently manage and edit their codebook, review annotations, access transcription data (which plays in tandem with the audio), and view a statistical summary of their annotations. The bottom section of the page displays the time axes, where users can create multiple tiers for various purposes, such as one tier for utterances, another for gestures, and another for emotions. When transcription data is available, utterances from each speaker are automatically organised into separate tiers, with each utterance displayed in its own tier and marked at the time interval during which it was spoken. On the right-hand side of the page, a chatbox is available for multimodal LLM-assisted annotations.

Manual Annotation: Users can create two types of annotation tiers: one that allows selection from a pre-defined codebook via a drop-down menu and another that enables free text input. Both types support direct editing and dragging along the time axes. Users can also define the start time, end time, duration, tier name, and the content of the annotation through the toolbar.

Automated Annotation: The embedded multimodal LLM serves to expedite the annotation process. Users can interact with the LLM via a chatbox, providing instructions, such as requesting assistance with annotations based on the transcription or video content. The LLM will then create a new tier and apply the corresponding annotations at the relevant time intervals. Additionally, users have the flexibility to edit the LLM-generated annotations either within the time axes or through the annotation toolbar. To address privacy concerns, a local algorithm has been implemented to detect and remove frames containing human faces before the video data is uploaded to the LLM.

Statistical Summary: The ROSAnnotator provides a detailed statistical summary for all the annotations, as well as for each individual tier. Key metrics for all tiers, including occurrences, frequency, average duration, time ratio, and annotation latency, are computed. For each tier, annotations are separately summarised, providing detailed statistics such as the total number of annotations, the minimum and maximum duration of a single annotation, the average and median duration, the total annotation duration, the percentage of time occupied



(a)

ROSBAG DATA FILE *: record_2024_06_17_12_09_57.bag

Predefined Codebook: behaviour_codebook.json

Annotation File

(b)

gesture
- pointing
- referencing

attitude
- negative
- neutral
- positive

action
- human_move
- robot_move

Pointing gesture - blue half cylinder (32 - 42)

You: can you help me label all gestures in a new axis?

ROSAnnotator: Done

You: Can you see the codebook data?

ROSAnnotator: The question you asked is not directly related to the provided audio transcript and video frames. The context involves a codebook, but the transcript and frames provided don't show or mention anything about a codebook explicitly. The transcript mainly involves spoken instructions related to placing different geometric shapes, while the video frames show actions involving these shapes. If there's another question relevant to this context, please let me know!

You: Can you help me label all gestures in a new axis?

ROSAnnotator: Done

You: Can you help me label all gestures in a new axis?

ROSAnnotator: Done

ROSAnnotator can help you annotate

(c)

Begin Time	End Time	Duration	Tier Name	Annotation
0.0	18.0	18.0	undefined	Hi, how are you? I'm so excited to work with you. How can I help you?
132.0	144.0	12.0	undefined	Sure.
268.0	288.0	20.0	undefined	Got it.
156.0	166.0	10.0	undefined	Working on that.
102.0	120.0	18.0	undefined	Can you put a yellow cylinder in the middle of the red arch please?

Annotation

Speaker SPEAKER_00: Can you put a blue triangle here please?

Speaker SPEAKER_02: Can you put a yellow cylinder in the middle of the red arch please? Of course.

Speaker SPEAKER_00: Come.

Speaker SPEAKER_00: Can you put the blue cube here? Working on that.

Speaker SPEAKER_00: Stop. Can you put the red rectangular prism in the middle over here? Working on that.

Speaker SPEAKER_00: Can you put a blue triangle here please?

Speaker SPEAKER_00: Stop. Can you put the yellow quarter cylinder on this part of the red block please?

Speaker SPEAKER_01: Got it.

Speaker SPEAKER_00: Stop.

Speaker SPEAKER_00: Complete. Thanks for a walk.

Transcript

Annotation Statistics

- Occurrences: 13
- Frequency: 0.053
- Average Duration: 18.77 seconds
- Time Ratio: 1.000
- Latency: 0 seconds from start

Statistics (Annotation)

Tier: SPEAKER_Robot

- Number of Annotations: 5
- Minimal Duration: 9 seconds
- Maximal Duration: 20 seconds
- Average Duration: 13.80 seconds
- Median Duration: 12 seconds
- Total Annotation Duration: 69 seconds
- Annotation Duration Percentage: 20.85%
- Latency: 0 seconds from start

Tier: SPEAKER_Participant

Statistics (Tiers)

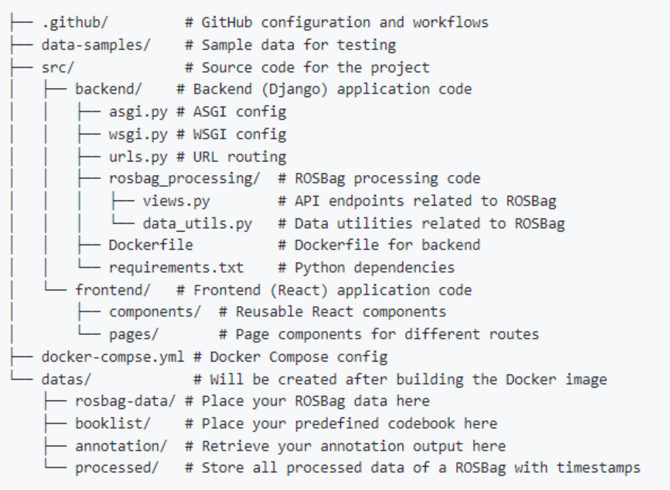
Fig. 2. The interface of ROSAnnotator WebApp. (a) shows the import page, (b) shows the annotation page with the codebook function in the toolbar, (c) shows other functions in the toolbar

by annotations, and the latency. The meanings of statistical metrics are summarised in Table I.

Data Export: ROSAnnotator allows for the export of various types of data. Processed video, audio, and transcription files are saved within the designated data directory. If the predefined codebook has been modified within ROSAnnotator, it will be updated accordingly. All annotations, including their content, tier name, start time, and end time, can be exported to a CSV file for further analysis. Additionally, a statistical summary of the annotations can be exported if required.

III. Installation

As the ROSAnnotator is developed as a web application, it is platform-independent and thus not constrained by the user's operating system. Before initiating the installation process, it is essential to ensure that Docker [14] is installed on your machine. If Docker is not already installed, it can be downloaded and installed from the official Docker[1] website. Prior to launching the server, navigate to the root directory of the project and build the image, which has an approximate size of 20 GB. ROSAnnotator requires an OpenAI Key and a Hugging Face Access Token to enable certain functionalities. To configure these, a `.env` file must be created in the root directory of the backend. In this file, the two required environment variables, OPENAI_API_KEY[2] and HUGGINGFACE_AUTH_TOKEN[3], should be manually added. Proper configuration of these credentials is essential for the application to operate as intended.



```
├── .github/         # GitHub configuration and workflows
├── data-samples/    # Sample data for testing
├── src/             # Source code for the project
│   ├── backend/    # Backend (Django) application code
│   │   ├── asgi.py # ASGI config
│   │   ├── wsgi.py # WSGI config
│   │   ├── urls.py # URL routing
│   │   ├── rosbag_processing/  # ROSBag processing code
│   │   │   ├── views.py        # API endpoints related to ROSBag
│   │   │   └── data_utils.py   # Data utilities related to ROSBag
│   │   ├── Dockerfile          # Dockerfile for backend
│   │   └── requirements.txt    # Python dependencies
│   └── frontend/   # Frontend (React) application code
│       ├── components/  # Reusable React components
│       └── pages/       # Page components for different routes
├── docker-compose.yml # Docker Compose config
└── datas/            # Will be created after building the Docker image
    ├── rosbag-data/ # Place your ROSBag data here
    ├── booklist/    # Place your predefined codebook here
    ├── annotation/  # Retrieve your annotation output here
    └── processed/   # Store all processed data of a ROSBag with timestamps
```

Fig. 3. Code structure of the ROSAnnotator WebApp

Both the frontend and backend are hosted locally within Docker containers. Once the server has been successfully initiated, the web application can be accessed through a web browser at http://localhost:5173/, which will provide users with the locally hosted interface of the application.

IV. Code Architecture

Figure 3 shows the code architecture of the ROSAnnotator WebApp. The frontend is built using the React library [15], while the backend, responsible for managing ROSBags and delivering data to the frontend, is implemented with the Django framework [16]. Within the `rosbag_processing` directory, the `data_util.py` file contains functions for data extraction and allows users to define and extract messages. By default, the functions support video and audio extraction as well as audio transcription. However, users are required to modify the topic names in `data_util.py` to align with the ones defined in their respective ROSBags. The `views.py` file in the same directory hosts the API endpoints, which currently support a manual annotation function and a privacy-secured LLM-based automatic annotation function. Users have the option to extend this file by adding other customised tools, such as gesture recognition. Furthermore, the file includes

[1] https://www.docker.com/get-started/
[2] https://platform.openai.com/api-keys
[3] https://huggingface.co/docs/hub/datasets-polars-auth

TABLE I
THIS TABLE LISTS THE MEANING OF THE STATISTICAL METRICS.

	Metrics	Meaning
Overall	Occurrences	The total number of annotations.
	Frequency	The rate at which annotations appear over a given period of time.
	Average Duration	The average length of time an annotation lasts.
	Time Ratio	The proportion of total annotation time relative to the observation period.
	Latency	The time delay before the first annotation occurs.
Individual Tier	Number of Annotations	The total number of annotations for this tier.
	Minimal Duration	The shortest duration of annotations for this tier.
	Maximal Duration	The longest duration of annotations for this tier.
	Average Duration	The average duration of annotations for this tier.
	Median Duration	The median duration of annotations for this tier.
	Total Annotation Duration	The sum of the duration of all annotations for this tier.
	Annotation Duration Percentage	The proportion of the total observation time in this tier that is occupied by annotations.
	Latency	Same as the overall one, but only for this tier.

default prompts and actions for the LLM model, though users can modify these prompts or switch to a different model if necessary.

After building the Docker environment using the `docker-compose.yml` file, a `datas` folder will be automatically generated. Users can place their ROSBags in the `rosbag-data` directory and their predefined codebooks in the `booklist` directory. Annotations are saved in the `annotation` folder, and users can access their incomplete annotations from this location to resume their work. To enhance processing efficiency, each ROSBag is processed only once, with the processed data stored in the `processed` directory for direct access in subsequent operations.

V. Usage Scenarios

Data analysis: ROSAnnotator is a valuable tool for facilitating qualitative data analysis in HRI. For instance, in [17], researchers examined participants' switch-hand behaviour, number of tool drops, and head movements while collaborating with a UR5 robot, using video footage captured by a camera. Similarly, in [18], video coders assessed recordings from a Pepper Robot, rating the level of warmth exhibited by participants during interactions. ROSAnnotator can streamline these types of measurements, enhancing the efficiency and accuracy of the analysis process.

Construct dataset: ROSAnnotator also supports the labelling and construction of datasets. For instance, in [19], researchers published a multimodal dataset, recorded using ROSBags, that included camera video, joint states, and other sensor data from robots. Additionally, researchers developed a conversational dataset [20] comprising both video and audio data. By using ROSAnnotator to label these recordings, datasets can be better organized and more adaptable for various research purposes.

VI. Limitations and Future Works

As ROSAnnotator is a newly developed web application with novel features, it currently faces several limitations. First, the automated annotation relies on a commercial API, incurring costs and limiting accessibility, while transcription poses privacy risks due to external audio processing. Future plans include hosting models locally to ensure data security and anonymising transcripts with generic labels (e.g., "Speaker 1") to further protect identities. Additionally, due to the constraints of the multimodal LLM's capabilities, the performance of the automated annotation is not robust. While users can define codebooks with detailed code lists for auto-annotation, and the system supports multi-tier and multimodal annotations to facilitate more subtle analyses, addressing nuanced HRI contexts remains a challenge for auto-annotations. Furthermore, the current version of ROSAnnotator primarily supports video (\image_raw topic) and audio (\audio topic) as default data types and also enables the use of user-defined messages. While users can already parse customised messages from ROSBags, the integration of AI-based analysis capabilities is planned for future development. To address these limitations, we will open-source ROSAnnotator and encourage HRI researchers to contribute by integrating custom message types and tools.

VII. Conclusion

Qualitative research is gaining a presence in the HRI research landscape, as illustrated by a new dedicated study track for qualitative methods. In this paper, we present ROSAnnotator, a web-based tool designed to support multimodal qualitative data analysis. Its primary advantages include: 1) the ability to directly extract messages from ROSBags and visualise data and time axes within a unified interface; 2) support for both manual annotations with codebooks and automated annotation through multimodal LLM; and (3) an open-source framework, which facilitates customised messages and tools for user-specific needs. ROSAnnotator aims to improve the efficiency of the qualitative data analysis process in empirical HRI studies and assist in labelling ROSBag data for the creation of datasets that can serve various research purposes.

Acknowledgment

We gratefully acknowledge the support provided by the Melbourne Research Scholarship (University of Melbourne) and the Australian Research Council Discovery Early Career Research Award (Grant No. DE210100858). We thank Bowen Fan, Tianqi Wang, Guanqin Wang, Yujie Zheng, Yuchen Song, Yucheng Peng, and Abhishek Tummalapalli for their assistance during the early stages of this work.

References

[1] L. Veling and C. McGinn, "Qualitative research in hri: A review and taxonomy," *International Journal of Social Robotics*, vol. 13, pp. 1689–1709, 2021.

[2] C. L. Bethel and R. R. Murphy, "Review of human studies methods in hri and recommendations," *International Journal of Social Robotics*, vol. 2, no. 4, pp. 347–359, 2010.

[3] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, A. Y. Ng, *et al.*, "Ros: an open-source robot operating system," in *ICRA workshop on open source software*, vol. 3, p. 5, Kobe, Japan, 2009.

[4] I. Lane, V. Prasad, G. Sinha, A. Umuhoza, S. Luo, A. Chandrashekaran, and A. Raux, "Hritk: the human-robot interaction toolkit rapid development of speech-centric interactive systems in ros," in *NAACL-HLT Workshop on Future directions and needs in the Spoken Dialog Community: Tools and Data (SDCTD 2012)*, pp. 41–44, 2012.

[5] Y. Mohamed and S. Lemaignan, "Ros for human-robot interaction," in *2021 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pp. 3020–3027, IEEE, 2021.

[6] U. Papen, "Participant observation and field notes," in *The Routledge handbook of linguistic ethnography*, pp. 141–153, Routledge, 2019.

[7] N. R. Prabhu, M. Tsfasman, C. Oertel, T. Gerkmann, and N. Lehmann-Willenbrock, "Dynamics of collective group affect: Group-level annotations and the multimodal modeling of convergence and divergence," *arXiv preprint arXiv:2409.08578*, 2024.

[8] P. Wittenburg, H. Brugman, A. Russel, A. Klassmann, and H. Sloetjes, "Elan: A professional framework for multimodality research," in *5th international conference on language resources and evaluation (LREC 2006)*, pp. 1556–1559, 2006.

[9] T. Muhr, "Atlas/ti—a prototype for the support of text interpretation," *Qualitative sociology*, vol. 14, no. 4, pp. 349–371, 1991.

[10] M. L. McHugh, "Interrater reliability: the kappa statistic," *Biochemia medica*, vol. 22, no. 3, pp. 276–282, 2012.

[11] J. P. Trujillo, J. Vaitonyte, I. Simanova, and A. Özyürek, "Toward the markerless and automatic analysis of kinematic features: A toolkit for gesture and movement research," *Behavior Research Methods*, vol. 51, pp. 769–777, 2019.

[12] F. Z. Canal, T. R. Müller, J. C. Matias, G. G. Scotton, A. R. de Sa Junior, E. Pozzebon, and A. C. Sobieranski, "A survey on facial emotion recognition techniques: A state-of-the-art literature review," *Information Sciences*, vol. 582, pp. 593–617, 2022.

[13] Z. Xiao, X. Yuan, Q. V. Liao, R. Abdelghani, and P.-Y. Oudeyer, "Supporting qualitative analysis with large language models: Combining codebook with gpt-3 for deductive coding," in *Companion proceedings of the 28th international conference on intelligent user interfaces*, pp. 75–78, 2023.

[14] D. Merkel *et al.*, "Docker: lightweight linux containers for consistent development and deployment," *Linux j*, vol. 239, no. 2, p. 2, 2014.

[15] C. Gackenheimer, *Introduction to React*. Apress, 2015.

[16] J. Forcier, P. Bissex, and W. J. Chun, *Python web development with Django*. Addison-Wesley Professional, 2008.

[17] Y. Wang, G. Ajaykumar, and C.-M. Huang, "See what i see: Enabling user-centric robotic assistance using first-person demonstrations," in *Proceedings of the 2020 ACM/IEEE International Conference on Human-Robot Interaction*, pp. 639–648, 2020.

[18] I. Kuyucu, A. Dogan, S. Akay, S. C. Bagci, and J. Kanero, "From human-human to human-robot: how social psychology research methods can inform hri evaluation," in *Companion of the 2024 ACM/IEEE International Conference on Human-Robot Interaction*, pp. 637–640, 2024.

[19] Y. Chen, Y. Luo, C. Yang, M. O. Yerebakan, S. Hao, N. Grimaldi, S. Li, R. Hayes, and B. Hu, "Human mobile robot interaction in the retail environment," *Scientific Data*, vol. 9, no. 1, p. 673, 2022.

[20] D. B. Jayagopi, S. Sheiki, D. Klotz, J. Wienke, J.-M. Odobez, S. Wrede, V. Khalidov, L. Nyugen, B. Wrede, and D. Gatica-Perez, "The vernissage corpus: A conversational human-robot-interaction dataset," in *2013 8th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pp. 149–150, IEEE, 2013.