

TDK-thesis

Miklós Máté Badó

János Gulyás Adrián

Keep Gesturing: A Game for Pragmatic Communication

EÖTVÖS LORÁND UNIVERSITY

FACULTY OF INFORMATICS

DEPT. OF ARTIFICIAL INTELLIGENCE



Authors:

Miklós Máté Badó

Computer Science BSc

II. grade

János Gulyás Adrián

Computer Science MSc

II. grade

Supervisor:

Dr. habil. András Lőrincz

Senior researcher

Budapest, 2024

Contents

1	Introduction	4
1.1	AI in Non-Verbal Communication	4
1.2	Pragmatic Communication	5
2	Related Works	7
2.1	Motion generation and understanding	7
2.1.1	Full body motion	7
2.1.2	Sign Language Generation and interpretation	9
2.1.3	Our Contributions	10
2.2	ChatGPT Applications	10
2.2.1	Integration with Augmented Reality	11
2.3	Social Robots	12
2.3.1	Foundational Theories in Social Robotics	12
2.3.2	Applications in Healthcare and Other Contexts	13
2.3.3	Non-Verbal Communication in Human-Robot Interaction	13
2.3.4	Our Contributions	14
2.4	Creating Languages through Interaction with LLMs	14
3	Keep Gesturing	16
3.1	HoloLens	16
3.1.1	Why use it?	17
3.2	Unity and MRTK	17
3.2.1	Importance of MRTK with Unity	17
3.2.2	The Augmented Reality Environment	18
3.2.3	Automatic Mesh Generation	18
3.2.4	Incorporating NeRF Models into Mixed Reality	19
3.3	Previous attempts and game iterations	19
3.3.1	Twister game	20

3.3.2	Balls and baskets	21
3.3.3	Iterations of KeepGesturing	22
3.4	A game for pragmatic language development	23
3.4.1	Inspiration and Overview	23
3.4.2	Gameplay	24
3.4.3	Levels and Progression	26
3.4.4	Cooperative Gameplay and Strategic Decision-Making	26
3.4.5	Animation interpretation and generation	27
3.4.6	How to play	27
3.5	GPT version and API calls	30
3.6	The prompt	30
3.6.1	Json format	30
3.6.2	The key parts of the system prompt	32
3.6.3	The parts of the user prompt	35
3.6.4	Parsing of the prompt	36
3.6.5	Problems with the prompt	38
3.7	The avatar	39
3.7.1	Why this avatar?	40
3.7.2	Using a NeRF as an avatar	40
3.8	Emerging Pragmatism in Keep Gesturing	41
3.8.1	Pragmatism in Keep Gesturing	41
3.8.2	How and why does pragmatism emerge?	42
3.8.3	The role of feedback in pragmatic communication	42
4	Testing	44
4.1	Experimental Setup	44
4.2	Tutorial Structure	44
4.3	Level Design	45
4.4	Experimental Procedure	46
5	Results	47
5.1	Participant 1: Performance and Observations	47
5.1.1	Progress of Participant 1	47
5.1.2	Pragmatic communication with ChatGPT in the case of Participant 1	49

5.2	Participant 2: Performance and Observations	54
5.2.1	Pragmatic communication with ChatGPT in the case of Participant 2	55
5.3	Overall Insights	56
6	Future Works	57
6.1	Larger Experiments	57
6.2	KeepGesturing 2	57
6.3	Other LLMs	58
6.4	Include NERF models	58
	Acknowledgements	59
	Appendix	60
.1	Communication pipeline	61
.2	Levels	62
.3	The full prompt	63
	Bibliography	91
	List of Figures	99

Chapter 1

Introduction

1.1 AI in Non-Verbal Communication

Artificial intelligence (AI) has significantly transformed various domains, including communication, interaction design, and digital media. AI's advancements have enabled the creation of more dynamic and interactive systems, enhancing our ability to create intuitive user experiences. By breaking down traditional barriers in human-computer interaction, AI has made significant strides in how humans and machines communicate and interact [1].

While much progress has been made, the application of AI in facilitating pragmatic communication—where context and non-verbal cues play a crucial role—remains largely unexplored. Non-verbal communication, which includes gestures, facial expressions, and body language, accounts for a substantial portion of human interaction. Recent studies have indicated that non-verbal elements such as facial expressions and body movements significantly contribute to the communication process, often cited as comprising around 60-70% of human communication [2]. These non-verbal cues are essential for conveying emotions, intentions, and contextual subtleties that words alone cannot fully express. Our project aims to investigate how AI, specifically models like ChatGPT, can be adapted to understand and engage in non-verbal communication systems. This exploration is particularly interesting because it addresses a significant gap in current AI capabilities and has the potential to dramatically enhance the naturalness and effectiveness of human-computer interactions. By enabling AI to navigate and enhance socially meaningful interactions, we can pave the way for more intuitive, empathetic, and effective communication between humans and machines.

Understanding and generating non-verbal communication through AI necessitates sophisticated models capable of translating between textual descriptions and physical motions. Text-to-motion models represent a significant advancement in AI, enabling the conversion of textual descriptions into detailed physical gestures. These models use natural language processing and deep learning techniques to interpret and animate described actions with high fidelity, making them essential for creating AI systems that can engage in natural and responsive interactions [3, 4]. Conversely, motion-to-text models strive to achieve the reverse: converting physical gestures back into descriptive text. This process involves complex scene and action recognition algorithms that must accurately decode and describe the nuanced movements observed, thus allowing AI to understand and respond to human gestures effectively [5, 6]. Both types of models face considerable challenges in accuracy and detail, as the subtleties of language and motion capture different aspects of human perception and machine interpretation [7].

Despite these capabilities, the use of ChatGPT for animation generation is not without its challenges. The model has been found to understand a comprehensive list of spatial properties and relationships, including shape, size, location and distance, all of which are crucial to creating an animation consisting of various joint key points of a rigid body. However, this is greatly limited by the fact that reaching this understanding often requires manual user-given corrections and multiple follow-up prompts. [8]

1.2 Pragmatic Communication

Pragmatic communication, a vital aspect of linguistics, explores how language is utilized in real-world contexts to efficiently and effectively convey information. This branch of study, often referred to as "pragmatics," emphasizes the adaptability and functionality of language in various social interactions. In our context, we focus on how these principles can be applied to develop a language of gestures that enhances communication in terms of speed, clarity, and adaptability [9].

The key aspects of pragmatic communication in our project include:

1. **Efficiency:** The ability to convey messages quickly is crucial, especially in dynamic environments where time is a critical factor. Our use of gesture-based language is designed to minimize the time required to communicate complex ideas, thereby speeding up interactions without compromising the richness of the infor-

mation shared. This is heavily dependent on context and information shared, as the same gesture may be interpreted differently in different contexts.

2. **Clarity and Effectiveness:** Precision in communication is essential to ensure that the intended message is comprehended without ambiguity. In developing a new gesture-based language, we place a strong emphasis on creating clear, distinct movements that accurately represent specific concepts or instructions. This approach significantly reduces the potential for misinterpretation.
3. **Emergence of Language:** A distinctive feature of our approach is the emergence of new gestural 'words' or phrases that can be universally applied across various scenarios. These gestures evolve as players and AI interact, leading to the development of a flexible and adaptive language. This emergent language enriches communication by introducing versatile, reusable symbols or gestures, thus enhancing understanding across different contexts. Its development requires the AI to learn from and adapt to the player based on the player's feedback but keeps the possibility of the AI inventing its gestures which the human player can learn and replicate.

Our approach is not to overwhelm ChatGPT with an extensive array of entirely moving joints with rotations and positions. Instead, we created a simple encoding with considerable communicative potential that ChatGPT can understand and generate independently. We argue that despite including no pre-defined natural language, a unique form of communication emerges with this format. This emerging language is entirely personalized to the user, as ChatGPT adapts itself to use and recognize the encoded inputs it receives. ChatGPT can also make up its own "words" that the user may learn and replicate to efficiently and quickly communicate information. This approach is particularly effective in scenarios where the user is unable to communicate verbally or in writing.

Chapter 2

Related Works

This chapter reviews the current state of research relevant to our project, focusing on integrating ChatGPT in facial animations, its application in gaming environments, and social robots, and the innovative concept of creating new languages through interaction with language models. Each section refers to significant studies and technological advancements that have shaped these areas.

2.1 Motion generation and understanding

Recent advancements in text-to-motion and motion-to-text models have significantly contributed to developing gesture-based languages. These models leverage the power of natural language processing and machine learning to bridge the gap between textual descriptions and physical gestures.

2.1.1 Full body motion

Text-to-Motion

The field of neural motion generation from text has seen a significant surge in research over the last few years[10, 11, 12, 13, 14]. Initially, many approaches employed deterministic models[15, 16], which often resulted in averaged and blurry motion outputs. These limitations have been effectively addressed through the use of stochastic models.

GAN models have been applied to generate action-conditioned motions[17, 18]. In the context of text-to-motion generation, the T2M framework[4] extended the temporal VAE

to create a probabilistic mapping between texts and motions. Similarly, TEMOS[19] uses a Transformer VAE to optimize a joint variational space between language and motions.

The emergence of diffusion models[20, 21, 22] and autoregressive models[23] has significantly advanced the field of motion generation. Diffusion methods train a network to progressively denoise a motion sequence, guided by a scheduled diffusion process. Autoregressive models convert motions into discrete tokens using vector quantization, which are then modeled by expressive transformers, akin to those used in language models.

Motion-to-Text

While considerable effort has been devoted to generating human motion from text, comparatively fewer studies have focused on the reverse task—generating text from motion.

Goutsu and Inamura [24] explored adversarial seq2seq architectures for this task on KIT-ML[25]. Later, Guo et al.[5] employed a bidirectional motion quantization-based transformer approach and were the first to use HumanML3D[4] for a motion-to-language generation. While their method achieved state-of-the-art performance for motion generation on HumanML3D, it did not significantly improve motion-to-text results on KIT-ML. Karim et al. [26] used a guided transformer to generate text from motion, achieving competitive results on both datasets.

Models for Both Text-to-Motion and Motion-to-Text Tasks

Recent advancements in motion synthesis have introduced innovative models that significantly enhance the integration of natural language processing with human motion tasks. Among these, several models stand out for their unique approaches and notable achievements. These are bidirectional models capable of both text-to-motion and motion-to-text generation.

MotionGPT is notable for its three-stage training scheme—motion tokenizer training, motion-language pre-training, and instruction tuning—which integrates natural language processing with human motion tasks to provide a unified solution for various motion synthesis problems [27, 20].

The Stable Text-to-Motion Framework (SATO) introduces stable attention and prediction modules to address instability issues in text-to-motion models, enhancing robustness and accuracy even with slight text input perturbations [28].

Vision Transformers (ViT) for 3D human motion-language models use motion patches and transfer learning from the image domain, achieving state-of-the-art performance in both text-to-motion and motion-to-text tasks. This approach is robust in tasks such as cross-skeleton recognition and zero-shot motion classification [29].

2.1.2 Sign Language Generation and interpretation

Sign Language Generation

Saunders et al. (2022) introduce a Frame Selection Network (FS-Net) and SignGAN for co-articulated, photo-realistic signing sequences, enhancing temporal alignment and realism [30]. Yin et al. (2020) translate spoken language into photorealistic sign language videos using deep learning techniques, producing high-quality signing avatars [31]. Suo et al. (2022) propose STCNet, ensuring temporal consistency and leveraging structural knowledge, greatly enhancing video quality [32]. Koller et al. (2020) develop multi-stream CNN-LSTM-HMMs for weakly supervised learning, improving sequential parallelism in sign language videos [33].

Sign Language Recognition

Recent advancements in sign language recognition (SLR) have significantly leveraged deep learning and natural language processing to enhance the performance and accuracy of these systems. Zuo et al. (2023) propose the Natural Language-Assisted Sign Language Recognition (NLA-SLR) framework, which integrates natural language modeling to handle visually indistinguishable signs (VISigns) [34]. Cheng et al. (2021) introduce a framework that focuses on the temporal consistency in sign language video generation, leveraging a spatio-temporal graph convolution network (ST-GCN) to model the dynamic nature of sign language [35]. Bohacek et al. (2022) utilize transformer architectures to enhance word-level sign language recognition, employing pose-based transformers to capture spatial and temporal features effectively [36]. Novopoltsev et al. (2023) explore the fine-tuning of SLR models using datasets from different sign languages, demonstrating that this technique significantly improves recognition quality [37]. Zhu et al. (2023) propose a pose-based recognition method that significantly improves accuracy and efficiency by

focusing on keypoint extraction and spatiotemporal modeling [38]. Huang et al. (2021) introduce a deep learning framework for continuous sign language recognition that utilizes a hybrid approach combining convolutional and recurrent neural networks [39].

These advancements in SLR are supported by extensive datasets such as WLASL-2000, RWTH-PHOENIX, and WLASL-100, providing comprehensive benchmarks for evaluating SLR systems [40, 41].

Wen et al. [42] introduce an advanced AI-driven sign language recognition system that bridges communication gaps between the hearing/speech impaired and non-signers. Unlike traditional glove solutions, it uses sensing gloves, deep learning, and virtual reality to recognize both words and sentences. The system's segmentation approach allows accurate sentence reconstruction with an 86.67% average correct rate, translating gestures into text and audio for seamless remote communication.

2.1.3 Our Contributions

Our work shares similarities with sign language generation and recognition studies, as both involve the use of hand gestures for communication. However, our approach is distinctive because, rather than focusing on a predefined language, we enable users to create their unique language through interaction with the AI. This aspect aligns with motion generation and understanding models, where the AI comprehends and generates any movement. In our case, though, this motion is specifically limited to hand gestures.

2.2 ChatGPT Applications

The integration of Natural Language Processing (NLP) in serious games has been extensively explored to enhance interactivity and educational potential. Eberlé, Picca, and Jaccard (2015) provide a comprehensive overview of NLP applications in serious games, identifying key challenges and future research directions in this field [43].

Sweetser[44] conducted a scoping review of 76 papers on the application of large language models (LLMs) in video games. The review categorizes research into four main themes: Game AI and Agents, Game Development and Play, Narrative, Story, and Dialogue, and Game Research and Reviews. This study highlights the dominance of GPT-based models and discusses their potential and current limitations in enhancing game design and player experience.

Hu et al.[45] provides a comprehensive survey of LLM-based game agents. They introduce the conceptual architecture of these agents, focusing on essential functional components such as perception, memory, thinking, role-playing, action, and learning. The survey also examines existing methodologies and adaptation agility across various game genres, providing a detailed outlook on future research and development directions in this burgeoning field.

Another study on generative agents discusses the development of agents capable of simulating human-like behaviors in interactive environments, enhancing player engagement and realism [46].

The integration of GPT-4 into other programs has shown significant promise, particularly in areas involving pragmatic language development. For instance, GPT-4 has been utilized in interactive educational platforms to enhance language learning by providing contextual, conversational practice scenarios that adapt to the learner's progress and proficiency [47]. In healthcare, GPT-4 has been integrated into patient interaction systems, enabling more natural and empathetic communication between patients and AI-driven healthcare assistants, thereby improving patient engagement and satisfaction [48].

2.2.1 Integration with Augmented Reality

The use of virtual characters has proven effective in assessing and training non-verbal communication, particularly in individuals with high-functioning autism (HFA). Georgescu et al.[49] discuss the advantages of using virtual characters (VCs) to provide an ecologically valid experience of social presence and systematically controlled experimental platforms. This approach allows for real-time social interaction in virtual settings, representing a sensitive measure of socio-communicative impairments in HFA. The study highlights VCs and environments as valuable tools for assistive, educational, and therapeutic applications for individuals with HFA.

Additionally, GPT-4 has been embedded into extended reality environments to support inclusion, engagement, and privacy. Bozkir et al. [50] explore the opportunities and challenges associated with embedding large language models into extended reality, emphasizing their potential to enhance interactive experiences and support diverse user needs.

Research by Rautaray and Agrawal (2011) on hand gesture recognition in virtual games underscores the importance of intuitive and natural interaction methods, which are crucial for effective communication in virtual environments[51]. Furthermore, tech-

nologies for social augmentations in virtual reality, as discussed by Roth et al. (2019), highlight how artificial intelligence can enhance social interactions by providing additional affordances and supporting individuals with social communication disorders [52].

2.3 Social Robots

Social robots are designed to interpret and respond to human emotions and social cues with a high degree of accuracy. This capability is critical in environments where robots are expected to perform roles traditionally occupied by humans, such as healthcare, education, and customer service [53, 54].

2.3.1 Foundational Theories in Social Robotics

Effective communication is pivotal in the development and success of social robots. Social robots are designed to interact with humans in a manner that is both functional and socially engaging, necessitating a deep understanding of social cognition and pragmatic communication.

Gallagher [55] provides a foundational understanding of social cognition in the context of social robots, emphasizing the need for these robots to interpret and respond to human behaviors appropriately. Building on this, Briggs and Scheutz [56] highlight the importance of pragmatics in social robotics, advocating for the generation of socially sensitive utterances. Their research demonstrates that robots can enhance social engagement by producing contextually appropriate responses, thereby improving interaction quality.

Communication Strategies and Frameworks

Torrey et al.[57] explores various communication strategies that improve social outcomes in human-robot interactions. Their findings underscore the significance of effective communication in fostering positive human-robot relationships, suggesting specific strategies that robots can use to better understand and engage with humans.

Kunold and Onnasch [58] propose a comprehensive framework for studying and designing communication with social robots. Their framework aims to optimize interaction quality and effectiveness, considering factors such as clarity, appropriateness, and user satisfaction. This approach aligns with the pragmatic emphasis on contextually appropriate communication.

Human-Likeness vs. Machine-Likeness

The debate over whether robots should emulate human behaviors is explored by Hartanto [59] and Sandry [60]. Hartanto advocates for a pragmatic approach, suggesting that while human-like attributes can be beneficial, they should not overshadow the inherent advantages of machine-like efficiency. Sandry supports this view, arguing that collaboration with machinelike robots can enhance user understanding and interaction quality.

2.3.2 Applications in Healthcare and Other Contexts

In specific application contexts, such as healthcare, the politeness behavior of robots plays a critical role. Lee et al. [61] investigate this aspect and find that polite and respectful behavior significantly enhances user compliance and satisfaction, which is crucial in settings where cooperation and adherence are vital.

Edwards et al.[62] delve into the social pragmatics of communication with robots, particularly in regulative contexts. Their study highlights the impact of message design on user perceptions, emphasizing the need for thoughtful crafting of robot messages to ensure effective communication.

Johanson et al.[63] provides a comprehensive review of communication behaviors in healthcare contexts, identifying strategies that can improve interactions and outcomes. Their findings are particularly relevant for designing robots intended for use in sensitive environments where effective communication is paramount.

2.3.3 Non-Verbal Communication in Human-Robot Interaction

Recent advancements have focused on the role of non-verbal behaviors in human-robot interaction. The study by Viet Tuyen et al.[64] explores the generation of human-like gestures based on natural language inputs using a Conditional Generative Adversarial Network (CGAN). This approach emphasizes the importance of non-verbal communicative gestures, which can significantly enhance a robot's ability to convey intentions, emphasize speech, and describe objects. The experimental results demonstrated that these gestures, when transformed into a robot's motion, are widely accepted and understood by the general public, thereby improving interaction quality and social outcomes.

2.3.4 Our Contributions

In our research, we extend these insights by developing a gamified environment that promotes the emergence of pragmatic skills through hand gestures. Similar to the emphasis on effective communication strategies discussed by Torrey et al.[57] and Kunold and Onnasch [58], our project leverages advanced hand-tracking capabilities of HoloLens 2 to enhance user interaction with the AI. Additionally, our approach aligns with the pragmatic focus highlighted by Briggs and Scheutz [56] by enabling a dynamic exchange of gestures that evolve through iterative feedback, reflecting the iterative refinement process described by Johanson et al. [63].

Our project, "Keep Gesturing," addresses the balance between human-like interaction and machine efficiency discussed by Hartanto [59] and Sandry [60], by creating a system that allows for clear, machinelike communication while also enabling the development of a shared language of gestures. This balance is crucial for fostering effective and efficient communication in increasingly complex interaction scenarios.

2.4 Creating Languages through Interaction with LLMs

Recent advancements have shown that it is possible to create new languages through interaction with LLMs[65, 66]. This approach does not involve explicit training on the new language but rather leverages the model's existing knowledge to adapt and respond in novel languages:

Models like mBERT[67] and XLM-R[68] are designed to handle multiple languages and can often perform zero-shot tasks in languages they were not explicitly trained on[69]. These models leverage shared multilingual embeddings to generalize across languages, enabling them to understand and generate text in new languages without additional training.

Interactive language modeling involves a teacher-student dynamic where a pre-trained model (teacher) selects optimal data to train a new model (student). This process iteratively refines the student model based on feedback, dynamically creating and refining languages through continuous interaction and feedback loops[70, 71].

Our project builds on these concepts by enabling users to interact with a language model through hand gestures, creating a new language of gestures through iterative feedback and adaptation. This approach leverages the model's existing knowledge to adapt

to the user's input, enabling the emergence of a shared language of gestures that evolves through interaction.

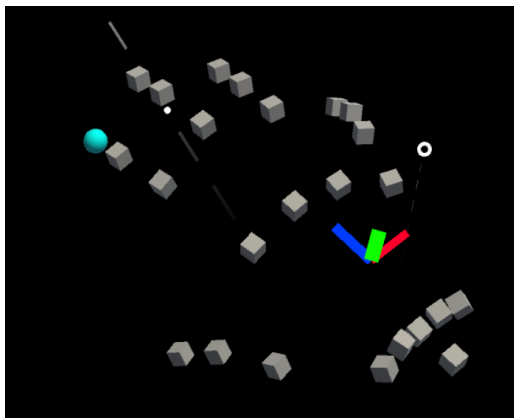
Chapter 3

Keep Gesturing

3.1 HoloLens

The HoloLens 2 is an advanced mixed-reality headset developed and manufactured by Microsoft. It was officially announced in February 2019 as the successor to the original HoloLens, released in 2016.

The HoloLens 2 features significant improvements over its predecessor in terms of comfort and usability. It is lighter and better balanced, with a more comfortable fit that can accommodate various head sizes and shapes. The device includes a flip-up visor design, allowing users to easily switch between mixed reality and the real world.



(a) The default joint representation of the right hand. The joints are represented by the cubes. The tip of the index finger is represented by the blue sphere.



(b) The generated hand mesh overlaid the real hand. The left-hand uses a pinch gesture. The recognition of this is indicated by the white outline of the two fingers.

Figure 3.1: Hand Tracking of HoloLens 2. On the left the tracked hand joints are visible while on the right the hand mesh is shown.

One of the standout features of HoloLens 2 is its advanced hand and eye-tracking capabilities, which allow for more natural interactions with holograms. Users can manipulate virtual interfaces with direct hand gestures, and the device can respond to where the user is looking, enhancing the fluidity and intuitiveness of user interactions.

HoloLens 2 is designed for industrial and enterprise applications, including training, remote assistance, and complex assembly. It is used in manufacturing, healthcare, and education for collaboration, 3D visualization, and hands-free operations. For example, Toyota uses it to improve training efficiency [72, 73]. In healthcare, it aids in remote consultations and 3D medical imaging [74]. Educational institutions like Case Western Reserve University enhance engagement with 3D content [74].

3.1.1 Why use it?

Using HoloLens 2 instead of a traditional screen offers unique advantages. Its spatial awareness allows users to engage naturally with the virtual environment, enhancing overall learning and interaction. The advanced hand-tracking technology enables precise interaction with virtual elements, surpassing standard touchscreens or mouse interfaces. In gesture-centric applications like surgical simulations or architectural modeling, users can manipulate 3D objects with gestures, improving realism and task efficiency. For our project, this means users can intuitively rotate, resize, or move virtual components.

Additionally, HoloLens 2 creates an immersive environment that significantly improves retention and understanding, especially in subjects like anatomy, physics, or engineering. The ability to interact with 3D models as if they were real objects bridges theoretical knowledge and practical understanding. This immersive interaction is crucial for achieving our goals of enhanced learning and effective gesture-based communication.

3.2 Unity and MRTK

3.2.1 Importance of MRTK with Unity

The integration of Microsoft Mixed Reality Toolkit (MRTK) with Unity is pivotal for developing mixed reality applications. Unity provides a robust platform with advanced rendering capabilities, while MRTK offers a comprehensive set of tools that simplify the development of immersive experiences across various mixed-reality platforms, such as the Microsoft HoloLens.

Even though MRTK3 was released, we have chosen to use MRTK2 since it is more stable and has been used in many projects before. MRTK2 provides a wide range of features and functionalities that are essential for creating interactive mixed-reality applications. These include spatial mapping, hand tracking, gesture recognition, and spatial awareness, among others.

3.2.2 The Augmented Reality Environment

In Unity, augmented reality (AR) environments are constructed by overlaying digital content onto the real world, a process enhanced by the capabilities of MRTK. This toolkit extends Unity's AR functionalities by incorporating features such as hand tracking, gaze control, and spatial mapping. These capabilities enable a more interactive and intuitive user experience, allowing for effective integration of digital and physical elements within the user's environment.

Object placement within an AR framework involves utilizing spatial anchors and world coordinates to position virtual objects within a real-world context. This interaction is facilitated by MRTK's spatial awareness tools, which allow objects to adhere to physical laws and environmental geometry, thus enhancing the realism of mixed reality experiences.

3.2.3 Automatic Mesh Generation

Automatic mesh generation plays a crucial role in applications requiring real-time interaction with the physical environment. In Unity, this capability is enhanced by the Microsoft Mixed Reality Toolkit (MRTK), which facilitates the generation of 3D meshes based on environmental data captured through sensors. These sensors might include depth cameras, LiDAR, or infrared, each providing detailed spatial information necessary for accurate mesh construction.

The meshes created are not merely static representations; they dynamically update as changes occur in the environment, ensuring that the virtual interactions remain consistent and realistic. This dynamic meshing capability is particularly vital in scenarios where environmental conditions change frequently, such as in outdoor AR applications or when moving through different rooms in a building.

Furthermore, MRTK in Unity simplifies the complex task of mesh generation by providing developers with tools to automatically segment the environment into logical parts.

This segmentation assists in optimizing the interaction by enabling more precise collision detection and physics-based interactions between virtual objects and the real world. For instance, virtual objects can accurately rest on real tables or bounce off walls, enhancing the user's immersion and interaction quality.

Additionally, automatic mesh generation supports the scalability of mixed-reality applications. It allows developers to create more extensive and complex environments without manually modeling every detail, which can be time-consuming and resource-intensive. This efficiency is critical for large-scale industrial applications, such as in manufacturing or architecture, where detailed and extensive environmental models are necessary.

3.2.4 Incorporating NeRF Models into Mixed Reality

Neural Radiance Fields (NeRF) models represent a significant advancement in the field of computer vision and graphics, enabling the creation of highly detailed and photorealistic three-dimensional reconstructions from sparse and irregularly sampled photographs. When integrated into Unity, NeRF models can transform the realism of virtual environments, significantly surpassing traditional rendering methods in terms of depth perception and visual coherence.

The use of NeRF models in Unity, assisted by MRTK, enables the simulation of realistic environmental effects, such as refraction, reflection, and shadow casting, which are crucial for applications demanding high levels of realism, such as virtual tourism, architectural visualization, and advanced training simulations. The ability to interact with these dynamically rendered environments in real time significantly enhances user engagement and satisfaction, making it a groundbreaking tool for developers in the realm of mixed reality.

3.3 Previous attempts and game iterations

There were various attempts tested before arriving at the current iteration of the game. These iterations are useful for understanding many of the game design choices made for "Keep Gesturing".

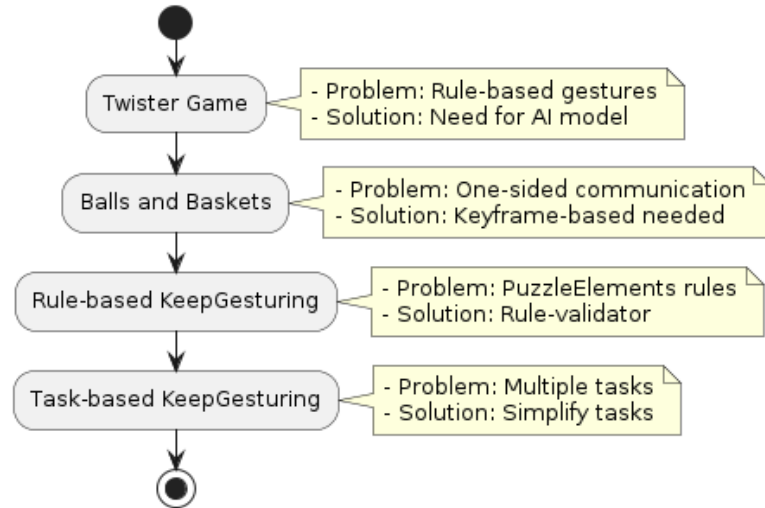


Figure 3.2: Iterations of Keep Gesturing, with a description of the problems and solutions during each phase.

3.3.1 Twister game

Our initial venture into pragmatic communication was developed through a game known as the "Twister Game." Unlike later versions, this prototype did not utilize ChatGPT. Instead, it was built around a dictionary of gestures with dynamic meanings. As the human player interacted with the game, they employed gestures from this predefined dictionary. Following each interaction, the AI counterpart would revise its understanding of what each gesture likely signified to the player, aiming to foster mutual comprehension. Although this version didn't fully capture the nuances of pragmatic communication — offering limited scope for clarity and efficiency — it played a crucial role in highlighting the necessity of integrating a sophisticated AI language model for future experiments in human-machine pragmatic communication.

In this initial approach, we employed a dictionary governed by a rule-based system to record and recognize gestures. The major challenge here was that each new gesture required the formulation of a new rule, making the development of a large dictionary exceedingly time-consuming. Moreover, with a small dictionary where each gesture strictly corresponds to one word without any ambiguity, and no possibility to introduce new gestures, the capacity for pragmatic adaptation was markedly hindered. Such a system significantly limits the ability to express nuanced or context-sensitive meanings, resulting in communication that is overtly literal and markedly less expressive. While some basic forms of pragmatism, such as sequencing or emphasizing gestures, might still surface,

they do little to enrich the depth and adaptability of pragmatic communication.

3.3.2 Balls and baskets

Balls and Baskets was our second prototype, which was much closer to the current iteration of the project. A human player was tasked with the task of sorting two types of balls into two types of baskets. The human, however, did not know how many of which type go where and required the assistance of a ChatGPT-powered companion to give him the solution. By raising his hand, the human would get a series of hand signals from ChatGPT describing the solution and would submit his interpretation of the hand sign when ready. They would both know if the solution is correct, allowing the human or the LLM to adjust their interpretation or generation respectively. By scoring how many balls the human successfully sorted, we could measure the improvement in communication. Although this was a huge step up from the previous iteration, it still had one glaring flaw in that the communication was entirely one-sided. To solve this issue, we are giving both parties vital information to communicate with each other during the game.

In the final version of our prototype, we opted for keyframe-based communication, but this decision followed extensive experimentation with various methods. One of the initial approaches we explored was picture/video-based gesture recognition. This technique offered the advantage of supporting a broader array of gestures beyond those confined to a pre-defined dictionary. However, it also presented significant challenges, particularly in terms of how responses from the language model should be handled. Specifically, generating new gestures based on textual input proved problematic. While we employed Motion-to-Text (M2T) and Text-to-Motion (T2M) models to facilitate communication, these models were only partially successful. When a gesture was processed through an M2T model and the resultant text was then input into a T2M model, the output gesture often failed to accurately replicate the original, especially with more complex gestures. This inconsistency between the original and regenerated gestures highlighted the limitations of our approach, leading us to explore and ultimately select keyframe-based communication that utilizes LLMs to recognize, understand, and predict the response gesture at the same time.

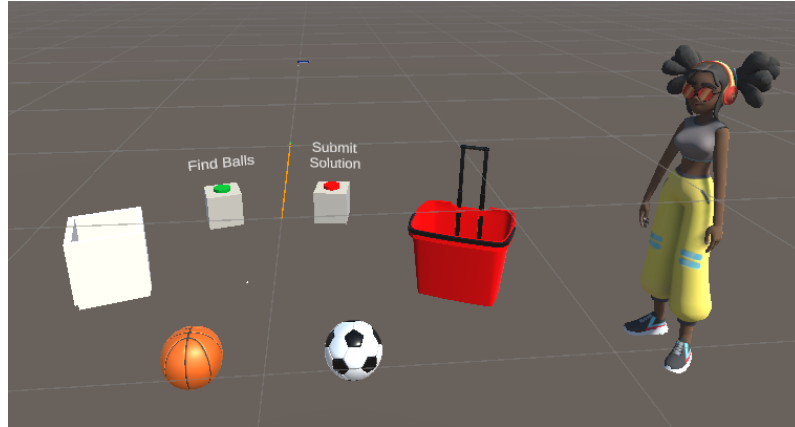


Figure 3.3: Picture of Environment of Balls and Baskets. The AI avatar is on the right. Between the 2 baskets, the submit and skip buttons are visible, in front of them are the soccer ball and basketball

3.3.3 Iterations of KeepGesturing

As with every project, KeepGesturing also had a few iterations before reaching its final form.

Rule-based KeepGesturing

In both the prototype and final versions of our system, humans collaborate with a Language Model (LLM) to solve puzzles. A key feature of the prototype termed the "Rule-based iteration", involves the creation of "PuzzleElements." Each PuzzleElement is designed with a unique solution that players must determine based on specific level rules.

For example, consider a rule stating, "If a PuzzleElement contains 2 Red Balls, they must be set to Green." Under this rule, if a player encounters a PuzzleElement with 2 Red Balls, the correct action is to change them to Green. This rule applies irrespective of other attributes; thus, it is valid for PuzzleElements containing "2 Big Red Balls" or "2 Small Red Balls."

The introduction of such rules necessitated the development of a rule-validator to ensure puzzle solvability, particularly as the rules were generated randomly. This validator functions by eliminating any rules that render a puzzle unsolvable.

During this iteration, GPT-4 was provided with a list of these rules. Not all rules were applicable at all times; some PuzzleElements presented to the player might not correspond to any rule, in which case they should be left unchanged.

Task-based KeepGesturing

This iteration was an evolution of the Rule-based version, incorporating more complex scenarios by allowing multiple rules to apply to each PuzzleElement, which significantly increased the difficulty. In this iteration, we referred to the combination of PuzzleElements and their corresponding rules as "Tasks." An example of a Task might be: "If there are 2 Balls, 3 Bottles, and a Red Car, turn the car Green and add another Ball." Each level consisted of multiple such Tasks.

However, the complexity of managing multiple Tasks proved too challenging for both the human players and GPT-4. As a result, in the final version of KeepGesturing, we simplified the approach to include only one Task per level. This change meant that the concept of multiple Tasks was not introduced to the player, making the gameplay more accessible and focused.

3.4 A game for pragmatic language development

3.4.1 Inspiration and Overview

KeepGesturing draws its conceptual inspiration from the Virtual Reality (VR) game *Keep Talking and Nobody Explodes*. It involves a unique partnership between a human player and an LLM-controlled avatar. The central premise is to solve puzzles through a sophisticated language of hand gestures. This collaboration aims to challenge traditional communication barriers and engage players in a novel form of interactive problem-solving.

To facilitate social communication using ChatGPT, we have developed a gamified environment that promotes the emergence of pragmatic skills. In this game, players and the A.I. must communicate and interpret complex ideas using only hand gestures. The increasing challenges they face and the limitations put upon them incentivize them to keep developing new, more effective gestures to effectively convey information. This requirement for innovation in gesture creation is critical, as each step of a gesture must encapsulate significant information if the participants wish to pass on complex thoughts succinctly. However, a gesture with too much information crammed into it may lose its clarity and lead to mistakes and score deductions.

Each game level presents a distinct set of puzzles, comprising objects that vary in number, type (such as balls or bottles), size (small or large), orientation (upright or

sideways), and color (green or red). The human player (Player1) is presented with the Current Configuration of these objects, which they can see directly in the game environment. In contrast, the LLM-controlled avatar (Player2) possesses knowledge of the Target Configuration—the ideal arrangement of objects to solve the puzzle. The challenge arises from this asymmetry in information. Player1 must accurately describe the visible configuration using only hand gestures, translating visual data into a form that the avatar can understand and act upon. To ensure the need for clarity in their communication, Player2 is equipped with more than one possible Target Configuration. Each of these possible targets depends on a specific Current Configuration, but only a single one is the correct solution. This necessitates that both parties convey their respective information accurately to avoid errors.

3.4.2 Gameplay

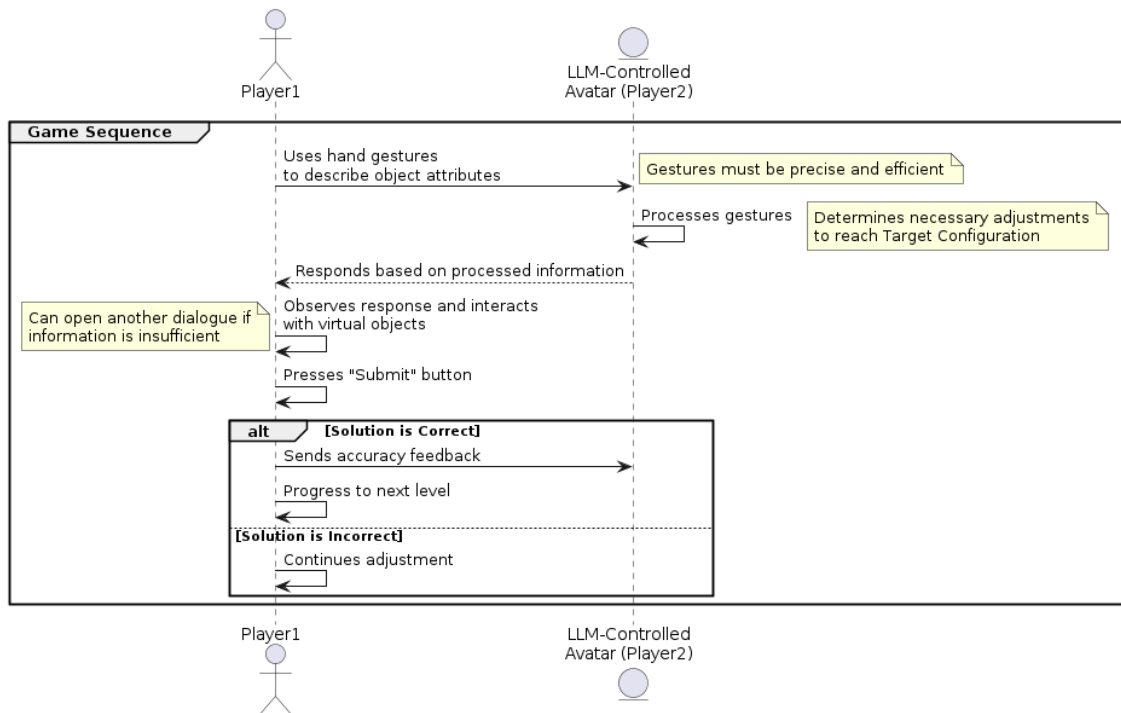


Figure 3.4: Gameplay Loop. Illustrating the interaction between Player1 and an LLM-Controlled Avatar (Player2) in a game. Player1 uses hand gestures to describe object attributes, and based on the avatar’s processed responses, adjusts the virtual objects until the solution is correct, at which point feedback is given, and progression to the next level occurs; if incorrect, adjustments continue.

The main gameplay loop is as follows (as shown in Figure 3.4):

1. Player1 uses hand gestures to describe these attributes of the objects in the Current Configuration. The gestures must be precise and efficient to minimize miscommunication and ensure that all necessary information is conveyed with minimal effort.
2. Upon receiving this information, Player2, the LLM-controlled avatar, processes the gestures to understand and determine the necessary adjustments needed to transform the Current Configuration into the Target Configuration. This includes interpreting which objects need to be moved, which should be reoriented, and any changes in their arrangement.
3. Player1 observes Player2's response and may apply his interpretation of it to the virtual objects. If he deems the received information insufficient or unclear, Player1 has the option to open another dialogue.
4. Whenever Player1 feels he has reached the Target Configuration, he can press a virtual "Submit" button which scores the solution's accuracy, sending a feedback signal to both players. If the solution is perfect, they are allowed to progress to the next level.

The sharing of information between the two parties is almost entirely open-ended. For instance, if Player1 observes a single object, they may only feel the need to describe the object's shape and is free to decide to leave out the numerical value of "1", if they suppose it is already implied. Perhaps Player2 might wish to save time and improve clarity by describing only the specific change required to go from the current config to the target, instead of describing the target configuration in its entirety.

The puzzles increase in complexity as the game progresses, adding more objects with varying attributes, thus requiring increasingly complex gestures and greater precision in communication. This setup not only tests the clarity of Player1's communication but also Player2's ability to interpret and act on these communications under varying levels of complexity.

Efficiency in solving the puzzles is rewarded, encouraging players to use the fewest gestures and adjustments possible. This dynamic tests both players' abilities to communicate effectively under constraints, emphasizing strategic planning, clarity, and adaptability.

3.4.3 Levels and Progression

The game is structured into progressively more challenging levels, each designed to enhance both player's skills using the gesture-based communication system. Early levels introduce basic objects and simple configurations, allowing players to familiarize themselves with the game. As players advance, the configurations become increasingly complex, introducing more objects and variables. This gradual increase in difficulty tests both the human and the A.I.'s adaptability and creativity, and most importantly, tests continuously if the language they possess can tackle bigger and bigger obstacles. The existence of these increasing obstacles in itself is an encouragement to create an easy-to-understand and dynamic language, as a language that remains inefficient now will only be met with more mistakes later on.

The levels featured in the experiments were defined by hand. Although it would be possible to randomize these levels, the randomness would need to be heavily restricted and manually parameterized to achieve the level of subtle guidance that we deem useful for speeding up the emergence of pragmatism. An important aspect of our level design mentality was the slowly progressing introduction of each property. In real communication, a defining attribute becomes imperative when a distinction needs to be made between two objects. Similarly in "Keep Gesturing, although the human player has full access to edit all properties of objects from the very beginning, most properties are not needed for the distinction of items until later. For instance, we have observed green pens, having never seen a green object before, they more often than not require a new gesture to be developed.

3.4.4 Cooperative Gameplay and Strategic Decision-Making

A significant aspect of *KeepGesturing* is its cooperative gameplay. The success of each level relies heavily on the players' ability to collaborate effectively. The game tests and strengthens communication skills, requiring players to be clear, concise, and strategic in their gestures. Moreover, the avatar's role as a guide adds an element of adaptive assistance, which evolves in parallel to the player's skills.

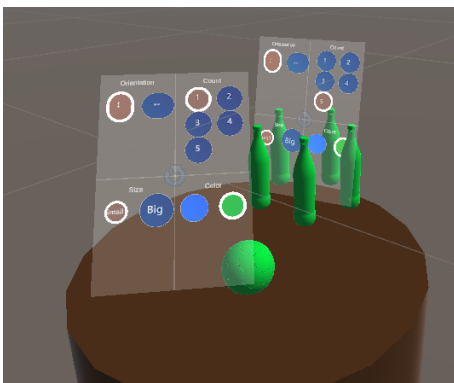
3.4.5 Animation interpretation and generation

In the design of "Keep Gesturing," communication hinges on the use of hand gestures. Given the continuous and complex nature of hand gestures, a method to discretize them was necessary to align with the processing capabilities of models like ChatGPT, leading to the segmentation into keyframes. Keyframes are a term borrowed from the animation industry, where keyframes describe important points of the animation between which the movement is interpolated.

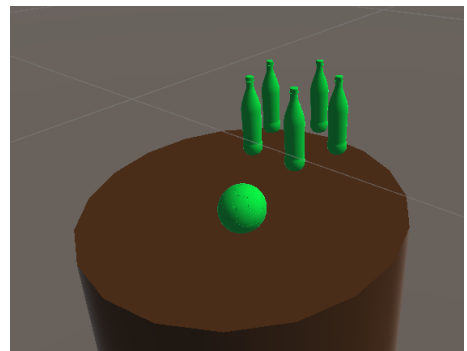
This design choice enhances the gameplay by making communication not just a means of interaction but also a critical element of strategy and creativity. It ensures that each gesture and the corresponding response are both crucial and impactful, turning each interaction into an opportunity for inventive expression and problem-solving.

A limitation of three animation keyframes per gesture (for a total of 6 per single interaction) was established as an optimal limit. Such a limitation is pivotal not just because of the AI's technical limitations, but also to foster the emergence of pragmatic communication strategies. Limiting interactions to three keyframes encourages the player and the AI to think creatively about how to convey their messages concisely, so that they may convey as much information as possible in a single dialogue. Gestures that exceeded this limit tended to convey overly complex motions with little in the way of actual information or to pad the information with a large amount of unnecessary details. Both of these causes led to a degradation in communication quality.

3.4.6 How to play



(a) 1 Ball PuzzleElement, and 5 Bottle PuzzleElement with the panels open.



(b) 1 Ball PuzzleElement, and 5 Bottle PuzzleElement without the panels.

Figure 3.5: Ball and Bottle PuzzleElements with and without the interaction panels.

Each level involves a puzzle that must be solved in as few attempts as possible. Each failed attempt decreases the score that may be gained from a specific level. A puzzle includes various objects dubbed "puzzle elements". The human player can see and edit any property of these elements. The task is always to edit these elements so they match with the Target Configuration, which is known only to the A.I. avatar.

Editing the PuzzleElements happens on an interaction panel, which can be opened or closed by touching one of the objects corresponding to that PuzzleElement. The panel contains all the properties of the PuzzleElement, which can be edited by the human player. The properties are color, size, orientation and number. The player can edit the PuzzleElement by interacting with the buttons on the corresponding panel. There is also a reset button provided to reset the PuzzleElement to its original state. The PuzzleElement is also reset to the original state when the player submits the level with the wrong solution.

A configuration consists of the property values of objects. Numbers were chosen as the first property as they are considered a generally trivial yet common attribute to communicate using hand gestures in everyday communication. Orientation and size were chosen next as they apply to most objects and are easy to determine and differentiate. All non-numerical properties can have 2 different values. This was determined to see if ChatGPT can determine values from context, so for instance, if the player would gesture that a specific element isn't horizontal, it must therefore be vertical.

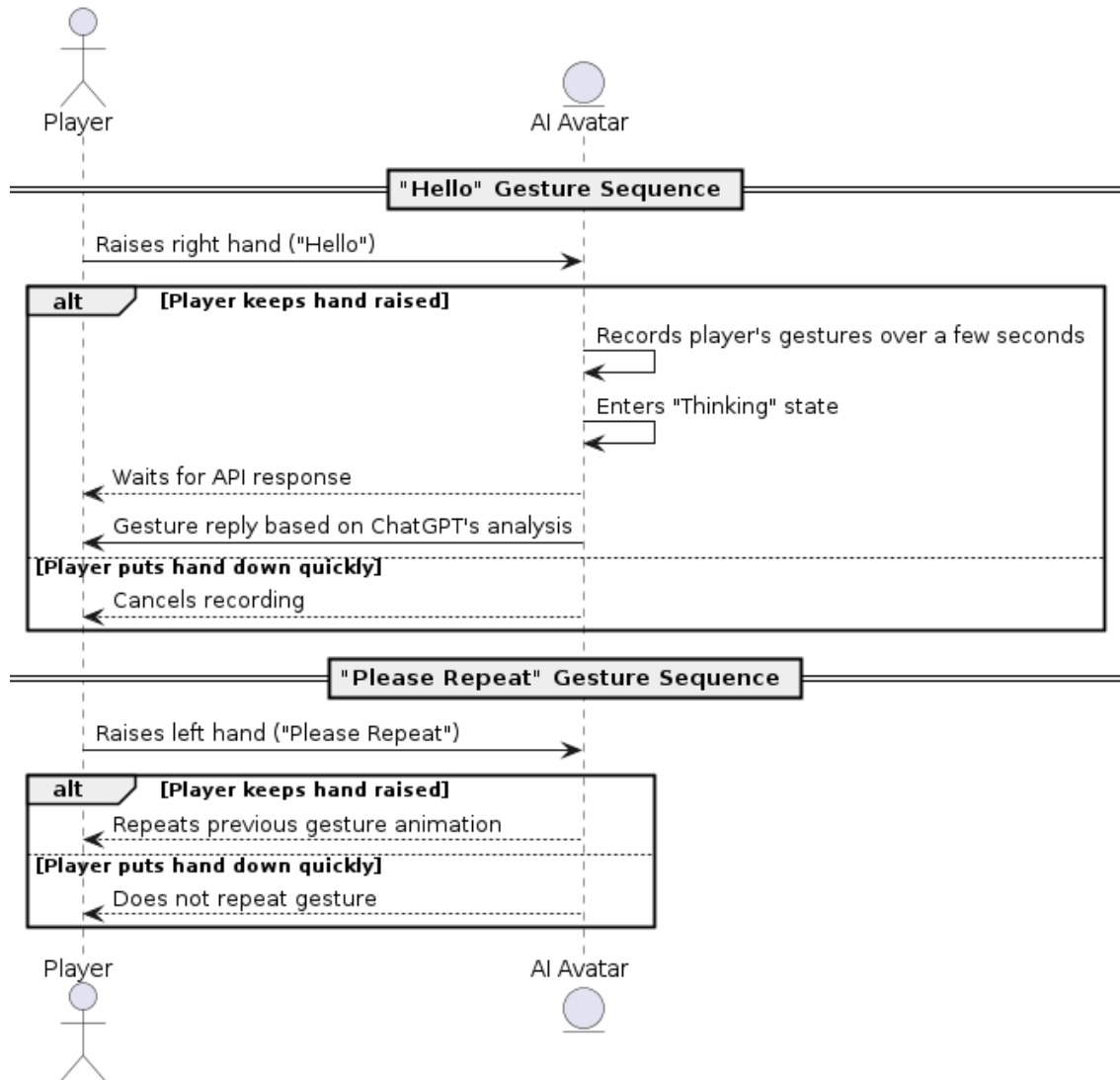


Figure 3.6: Sequence diagram for talking. Showing the interaction between a Player and an AI Avatar using gesture sequences for communication. In the "Hello" gesture sequence, the Player raises their right hand, prompting the AI Avatar to record the gesture, analyze it, and respond; in the "Please Repeat" gesture sequence, the Player raises their left hand to request the AI Avatar to repeat the previous gesture animation.

There are two pre-defined control gestures available to the player (outside of interacting with items and buttons) as shown in Figure 3.6. These indicate "Hello" and "Please repeat", and are performed by raising either the left or right hand towards the A.I. avatar. A "Hello" signal will attempt to begin an interaction, in which the player is recorded at various snapshots over a few seconds. After the recording has concluded, the avatar enters a "Thinking" state. This is a pre-defined animation that shows to the player that the avatar is waiting for an API response. Once a ChatGPT's gesture reply. A "Please repeat" signal will notify the avatar to repeat their previous gesture. This is useful if the gesture animation is too fast or if the player simply forgot or didn't pay close attention.

3.5 GPT version and API calls

Due to the additional overhead, it would require to keep up with the frequent ChatGPT updates, testing and development were conducted entirely on the relatively recent gpt-4-1106-preview version. API communication is performed asynchronously. This means that as soon as ChatGPT's reply contains a playable hand motion, the avatar's animation can begin. This can save up to a minute of waiting time for the player, making communication much more timely and responsive. While the animation is playing, ChatGPT is free to keep describing its intentions and to construct its dictionary in the background, which is also saved and logged for future reference and statistics.

3.6 The prompt

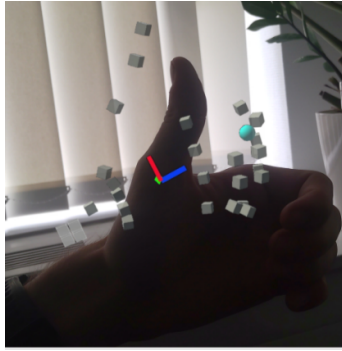
During the creation of the prompt, multiple prompt engineering approaches are utilized according to the OpenAI documentation[75]. These are the tactics we utilized:

- Clear instructions
- Adopting a persona
- Providing examples
- Splitting the task into multiple parts
- If the model made a mistake, we ask it to correct it
- The distinct parts are enclosed in delimiters if capital letters must be used with exclamation marks.

For the full unedited prompt see Appendix .3 at page 63.

3.6.1 Json format

As noted earlier, our approach to gesture communication involves the use of keyframes, each of which is encoded into a custom JSON format. This encoding is specifically designed to enable ChatGPT not only to interpret but also to anticipate these gestures effectively.



(a) A thumbs up gesture with fingers slightly open. The hologram mismatch is due to the HoloLens 2's limitations [76]

```

1 "leftHand": {
2   "active": true,
3   "position": {
4     "x": -0.1,
5     "y": -0.1,
6     "z": 0.4
7   },
8   "rotation": {
9     "x": -17.6,
10    "y": -12.5,
11    "z": 101.6
12  },
13  "index": 0.45,
14  "middle": 0.33,
15  "ring": 0.3,
16  "pinky": 0.37,
17  "thumb": 1.0
18 }

```

(b) The JSON representation of the showed gesture without the unnecessary parts.



(c) The same gesture is shown in the avatar's hands.

Figure 3.7: Joint recognition with the HoloLens 2, with the encoded JSON format, and the Avatar showing the gesture.

The encoding utilizes openness values for the fingers, and coordinates/rotations for the wrist. This methodology results in a lossy encoding scheme specifically for the fingers. In this scheme, the detailed joint coordinates of each finger are transformed into a more general measure of openness, expressed as percentages. Consequently, while this transformation leads to a loss of some precise positional information, it also streamlines the data into a format that is more easily understood by both humans and LLMs. By reducing the complex spatial data of finger joints into a single openness value, the model can process gestures more effectively. This approach mimics human cognitive processes, wherein a simplified representation (such as how open a finger is) is often easier to understand and interpret than detailed, multi-dimensional data (like specific coordinates for each joint).

Initially, we tested whether GPT-4 could generate reliable coordinates for all joints in a gameplay scenario. However, the results were inadequate for providing an enjoyable gaming experience. Subsequently, we discovered the previously mentioned encoding method. Although GPT-4 occasionally still encounters errors with this method, the overall performance has improved significantly, reducing the likelihood of pragmatic issues.

Important to note, that the latest version (after 2024.05.01) of the GPT4 API has a JSON mode, which makes the model always output a valid JSON if specified. Using this,

would probably speed up the response time, but remove the explanation part, which is crucial for the communication afterward.

3.6.2 The key parts of the system prompt

The system prompt can be split into 6 main parts, each with a distinct task, and only in the last part, we ask GPT-4 to write any response. The parts are as follows:

1. The JSON explanation: a detailed explanation for the JSON, also explaining the bounds
2. Task explanation: the game is explained, and GPT is told what to do, and how
3. An example game: A hand-crafted example game.
4. Important notes: Key for making GPT remember the previous instructions
5. Dictionary: it is used by GPT to log the used gestures for itself.
6. Required output format: How should GPT format its reply + how should the dictionary be written

JSON explanation

In explaining the keyframe gesture format of our gesture recognition system, we've taken care to break down the complex technical details into manageable segments such as Active usage, Finger Openness, Position, Keyframes, and Rotation. This segmentation helps clarify and make the information more accessible. Each component is precisely defined using descriptive terms. For example, we explain finger openness on a scale from 0 to 1 and describe the spatial orientation of gestures about the human body.

We also use examples to illustrate how specific coordinates position the hand relative to the head and how different rotation scenarios look for each hand. These examples provide a visual framework that makes abstract data more tangible and understandable. Additionally, by specifying the orientation of the axes and setting limits for rotations, we offer directional guidance that prevents incorrect data interpretation and ensures uniformity in how gestures are encoded and read by the system.

Due to GPT-4's challenges with intricate 3D rotations, we have provided detailed explanations of rotations, both in terms of Euler angles and medical terminology. Our testing revealed that GPT-4 interprets medical terms for rotations more accurately, which prompted us to use this approach in explaining rotation limits as well.

Following the detailed explanations, we also supply a pre-structured empty response format. This template allows GPT-4 to focus on modifying values rather than generating the format from scratch, which has reduced the occurrence of invalid JSON outputs. This structured approach ensures that GPT-4 produces reliable and correct data formats, enhancing the overall efficiency and accuracy of our system.

Task explanation

Here we've outlined a problem-solving framework that defines the roles and interactions between participants in a simulated environment, emphasizing the practical application of the gesture system. Challenges are clearly stated, and strategies for effective communication are suggested, highlighting the need for accurate and efficient gesture-based exchanges. This structured explanation not only aids comprehension but also aligns GPT-4 with the experimental goals, enhancing engagement with our research objectives.

For GPT-4 two goals were outlined. The main goal of complete each level with as little error as possible, and the secondary goal of create a gesture-based language that pragmatically emerges.

The example game

After explaining the JSON format and the primary objectives of the game, we provide a detailed example to offer guidance and clarify the format further. This example includes specific scenarios that, during our testing, proved challenging for GPT-4 to complete successfully. This helps ensure that the format is better understood and more easily applied, while also highlighting areas where the model may need further refinement.

Important notes

In this part of the prompt, we provide essential information to help GPT-4 remember the instructions and maintain consistency throughout the game. This includes instructions on how to handle the JSON format, rethink the rotations, and the need to adhere to the required output format. By emphasizing these key points, we ensure that GPT-4 remains focused on the task at hand and follows the established guidelines. This section serves as a reminder of the critical aspects of the game and helps maintain a structured and coherent approach to communication.

Additionally, this section emphasizes the importance of adapting to the gestures provided by the human participant, creating new gestures only when necessary. It underscores the need for minimal but clear communication, ensuring that unnecessary details are omitted, and highlights the significance of accurate replication of rotations and finger positions. It also stresses the avoidance of redundant keyframes and the correct application of right-hand to left-hand transformations. By reiterating these crucial points, we aim to correct any inconsistencies observed in previous tests and reinforce the primary responsibilities and best practices for GPT-4. This structured reminder is vital for maintaining effective and accurate interactions throughout the gameplay.

The "Dictionary"

In this task, the focus is on developing and maintaining a detailed DICTIONARY of gesture-word pairs to facilitate efficient and accurate communication. This process involves methodically updating the DICTIONARY based on feedback, particularly incorporating gestures that receive positive feedback. The DICTIONARY serves as a modular system, containing individual words or concepts rather than phrases, which allows for the construction of complex expressions using pre-established gestures. An essential aspect of this method is the precise replication of finger values and rotations as demonstrated by the user, ensuring consistency and accuracy. This approach avoids redundant gesture creation, thereby enhancing efficiency by utilizing existing gestures.

We task ChatGPT to construct a concise dictionary cataloging any observed gestures and their corresponding interpretations at the end of each response. Although ChatGPT is the one keeping track, this DICTIONARY encompasses contributions from both the human participant and ChatGPT. The collection is entirely dynamic, with elements being added or removed and interpretations shifting between levels and level attempts as the game progresses, aided by various feedback signals and scorings. This dictionary is preferable to standard longer descriptions for two main reasons: firstly, it helps the game's requests remain within the context window's token limit, and secondly, it makes ChatGPT's job much easier. We noticed that with the standard paragraph-long descriptions that ChatGPT defaults to providing, it had a much harder time finding a previously established gesture to express a given concept. This issue appears even if the established gesture's meaning was deemed certain by ChatGPT and it has observed it or used it multiple times with positive feedback. The simplification provided by the self-constructed dictionary reduces the task's complexity immensely and results in a much more reliable

adaptation from ChatGPT. It is also a key component in achieving the emergence factor of pragmatism.

Required Output Format

For the output, we utilized Chain-of-Thought prompting [77], a method that enables the LLM to break down complex tasks into manageable steps. This approach is particularly crucial for more challenging tasks. To adhere to this methodology, we have divided the output into four distinct parts:

1. A short list explaining each gesture the player has shown in the last question.
2. The creation of the Dictionary.
3. A brief explanation of what GPT-4 will reply with, including how many fingers it will use and how many keyframes it will generate.
4. The JSON format of the keyframes.

The rationale behind this structure is to guide GPT-4 through a sequential thought process. In the first part, GPT-4 reflects on the player's recent gestures to understand their meanings. Next, it updates the Dictionary, incorporating the latest gestures along with previously used ones. Following this, GPT-4 plans its response, detailing the gestures it will use, the number of fingers involved, and the number of keyframes required. By the time GPT-4 reaches the keyframe generation stage, it has already set up all necessary information, allowing it to produce the keyframes efficiently and accurately without further deliberation. This structured approach ensures clarity and consistency in GPT-4's output, facilitating effective communication and task execution.

3.6.3 The parts of the user prompt

The user prompt is much simpler than the system prompt, as it only needs to send the player's last gesture to GPT-4 or ask for a rethink if a mistake was made. Because of this, the user prompt has 2 types, a reask prompt, and a gesture prompt.

The Gesture Prompt

This prompt consists of three main components:

- Status of level submission: Acknowledges if a level was submitted and whether the tasks were completed. If not submitted, it reminds the LLM to create new gestures.
- The user's most recent gesture.
- Reminders of the required output.

If the level was submitted successfully, the prompt congratulates GPT-4 and encourages it to remember the used gestures. If the submission was incorrect, the prompt expresses frustration, indicating the response was unclear and needs improvement. If the level was skipped, the prompt responds with strong disapproval, urging GPT-4 to rethink its gestures and provide better responses in the future.

The Reask Prompt

As previously mentioned, the reask prompt is triggered in three scenarios: if the JSON is invalid, if the rotation is incorrect, or if the position is incorrect. This prompt simply asks GPT-4 to reconsider its last response and correct any mistakes made.

3.6.4 Parsing of the prompt

For the full pipeline see Appendix .1 at page 61.

It is widely acknowledged that GPT-4, like any complex AI model, is prone to errors. To mitigate this issue and enhance user interaction speed, we implemented a "reask" feature that intercepts and corrects errors before they reach the end user. This process operates as follows (as also shown in Figure 1):

GPT-4 responses are sent asynchronously, allowing our system to parse the JSON structure embedded within the prompt incrementally. As soon as a valid JSON is detected, the corresponding gesture is prepared for display, eliminating the need to wait for the complete response. This adjustment reduces waiting times by approximately one minute.

If the JSON extraction from the response is successful, a preliminary check assesses whether the rotation or position values fall within acceptable bounds. If these checks fail or if no valid JSON is present, the system automatically reissues the prompt to GPT-4, limited to a maximum of three attempts. This limit is set because repeated failures typically indicate more systemic issues, such as insufficient funds in our OpenAI account.

Once a valid JSON is obtained and passes all checks, the gesture is displayed to the user. Meanwhile, the rest of the GPT-4 response is logged in the background. This logging occurs in two phases:

1. Initially, only the parsed JSON is logged. If a reask is necessary, we avoid delays associated with processing the full reply since the erroneous JSON renders it irrelevant.
2. After the user performs the gesture, the complete response overwrites the preliminary log.

This method substantially accelerates gameplay by minimizing wait times between interactions. If no valid JSON is generated, the entire response is resent to GPT-4 for correction.

It is crucial to note that GPT-4 often includes comments within the JSON despite instructions to the contrary. To address this, we employ regular expressions (Regex) to strip any comments before parsing. Additionally, the JSON structure is identified using Regex that detects the markers delineating the start (“JSON”) and end (“”) of the JSON segment in the Markdown.

Furthermore, our system does not require GPT-4 to generate all three keyframes for both hands unless they are needed for the gesture. Unused frames or hands can be omitted by setting their active flags to false or removing them entirely from the keyframe, streamlining the data and reducing unnecessary processing.

3.6.5 Problems with the prompt

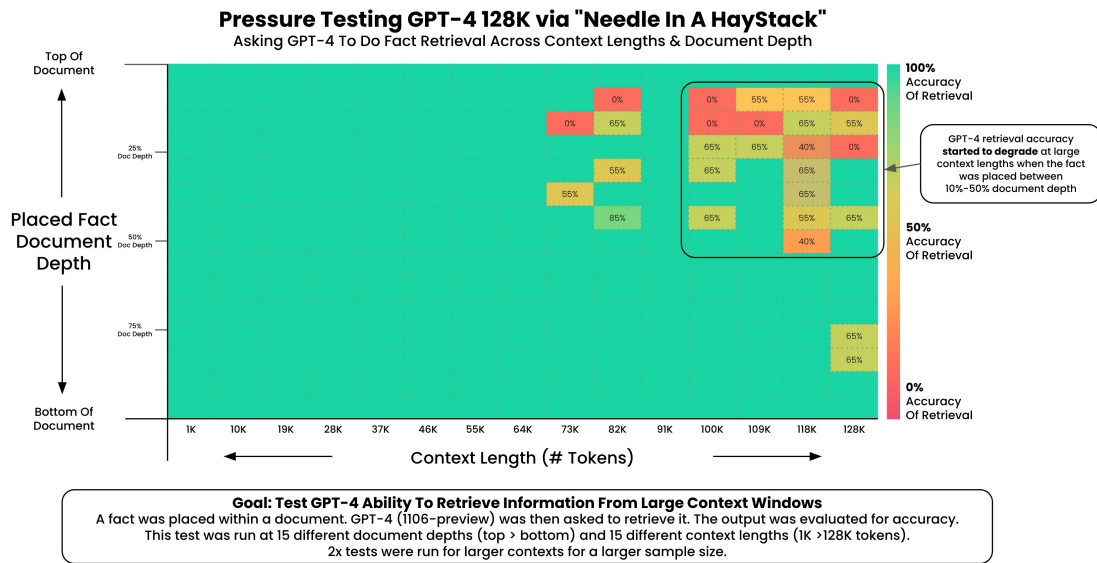


Figure 3.8: Pressure Testing GPT-4 128k via "Needle In A Haystack", done by Greg Kamradt[78]. Image used with permission.

As with every prompt, there are some issues that we have encountered. The most common one is that ChatGPT likes to talk. This is a problem because the required output schema (not the JSON) is broken sometimes. And with a lot of words, the token count grows significantly and that is a big problem in our case, because as also shown in 3.9, as we go up in tokens, the 64.000 the first half of the context starts to disappear, and after 100k tokens, the recall of the first half is very low. Because of this, the first gestures are forgotten, and to this exact problem, we have implemented the dictionary as previously explained. But this solves the problem only partially, as the dictionary is used perfectly at the first 60.000-70.000 tokens, but after that, it is used less and less. The problem is that the prompt is too long, and we ask too many things from ChatGPT to write down, and it increases the token count significantly.

The second one is that GPT-4 misses the rotation values. This is because it is very hard even for humans to think with 3 Euler angles when making a motion, and GTP-4 also has that problem. Initially, we had the finger openness values also with the rotation values, but because of this exact problem, we switched to the previously explained openness values, changing to other measures for the rotations was also tried, but because there are too many values, it was either too hard for ChatGPT, or it was too much simplification. So we decided to keep the Euler angles but reinforce GPT-4 to think about them.

The third significant problem is with the simplest task. Gesturing numbers. GPT-4 likes to mix multiple gestures in one keyframe in one hand and this is not a problem until one of those gestures is a number. Because it will show the incorrect amount of numbers.

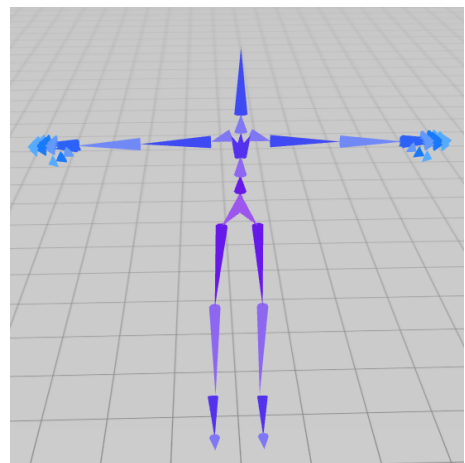


Figure 3.9: Wrong number shown by Avatar. The Avatar tries to gesture to 2 horizontal bottles, and the left hand correctly shows the bottle gesture (a hand shaped like grabbing a bottle) tilted 90 degrees, but the right hand shows 3 fingers, which is incorrect. The explanation was that it shows the direction of the bottle with the index finger while showing the number 2 with two other fingers extended.

3.7 The avatar



(a) The mesh for the avatar.



(b) The rig for the avatar.

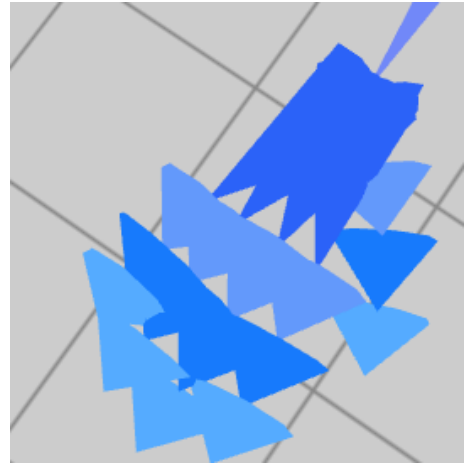
Figure 3.10: The Avatar, Michelle. On the left, the mesh can be seen, while on the right the corresponding bone structure is shown, with arrows indicating the flow of kinematics.

The Artificial Intelligence (AI) in "Keep Gesturing" is represented through a three-dimensional animated character, accessible in augmented reality (AR). For our experiments, we employed a character model sourced from the Adobe Mixamo animation library, which is publicly available. This character is equipped with a skeletal rig composed of multiple joints and bones. The rigging technique ensures that movements of the bones directly influence the corresponding sections of the skin geometry, enabling dynamic animations. Usage of the Adobe Mixamo library was chosen for its wide compatibility, detailed hand structure, and ability to simulate a range of generic actions, such as pondering or idling, which make up the very basic interaction dynamics in "Keep Gesturing."

3.7.1 Why this avatar?



(a) The right-hand closeup. Showing the details.



(b) The rig for the right hand.

Figure 3.11: The Avatar's hand closeup. On the left, the mesh can be seen, while on the right the corresponding bone structure is shown, with arrows indicating the flow of kinematics.

The current avatar was chosen for its stylistic look and detail in the hands. The shape and resolution of the hands are important as they may affect a human observer's ability to interpret any performed hand gestures.

3.7.2 Using a NeRF as an avatar

Neural Radiance Field representations of a person or humanoid character can be transformed into a static mesh. Such a static mesh can be outfitted with a skeletal rig manually

or using an auto-rigging tool. The Mixamo auto-rigger is a powerful tool that can be used with converted NeRF models. It automatically creates a skeleton for any humanoid character, making it easy to animate and compatible with all animations in the Adobe Mixamo animation library. However, the NeRF model must have sufficient detail in its hands or the model may fail to perform distinct hand gestures, which are integral to the gameplay mechanics of "Keep Gesturing." Thus, ensuring the NeRF model's fidelity in representing hand anatomy is essential for its successful application in gesture-based interactive systems.

3.8 Emerging Pragmatism in Keep Gesturing

3.8.1 Pragmatism in Keep Gesturing

In the context of Keep Gesturing, the emergence of a pragmatic communication system is influenced by three key parameters: emergence, effectiveness, and efficiency.

- **Emergence:** Emergence refers to the natural development of a shared language of gestures among players. This process is driven by the necessity to communicate within the constraints of the game's design. As players engage in solving puzzles, they organically create and refine gestures that become part of their shared lexicon. The game's structure encourages this gradual development, allowing a pragmatic language to emerge that is tailored to the specific challenges and needs of the gameplay.
- **Effectiveness:** Effectiveness measures how well the developed gestures convey the intended messages. In Keep Gesturing, effectiveness is critical as players must communicate complex ideas and actions through simple hand movements. The game's immediate feedback mechanism helps players assess and improve the clarity and accuracy of their gestures. By continuously refining their gestures based on this feedback, players enhance the effectiveness of their communication, ensuring that their messages are understood as intended.
- **Efficiency:** Efficiency focuses on the use of the fewest gestures to communicate ideas clearly and concisely. Keep Gesturing incentivizes players to develop highly efficient communication strategies by rewarding those who solve puzzles with minimal gestures. This drive towards efficiency not only speeds up gameplay but also

encourages players to innovate and streamline their gesture-based language. The result is a pragmatic communication system that balances brevity and clarity, allowing players to convey information swiftly and effectively.

These three parameters—emergence, effectiveness, and efficiency—work in tandem to foster a pragmatic language within Keep Gesturing. By emphasizing these aspects, the game creates an environment where players can develop and refine a gesture-based communication system that is both practical and optimized for the challenges presented.

3.8.2 How and why does pragmatism emerge?

In Keep Gesturing parameters are set up in such a way that pragmatism is encouraged to emerge. The emergence of pragmatism is a result of the game’s design, which incentivizes players to develop a shared language of gestures that is both efficient and effective. The game’s structure, which involves solving puzzles through gesture-based communication, requires players to create gestures that are clear, concise, and unambiguous. As players progress through the game, they are encouraged to refine and adapt their gestures to convey increasingly complex information accurately. This iterative process of gesture creation and interpretation fosters the emergence of a pragmatic language that is tailored to the specific needs of the game. The other factor that contributes to the emergence of pragmatism is the hand-crafted levels. These levels are designed to gradually introduce new concepts and challenges, allowing players to develop and refine their gesture-based language over time. By providing a structured environment that encourages experimentation and innovation, Keep Gesturing creates the conditions for pragmatism to emerge naturally.

3.8.3 The role of feedback in pragmatic communication

Feedback plays a crucial role in the development of pragmatic communication in Keep Gesturing. The game’s immediate feedback mechanism allows players to assess the effectiveness of their gestures and adjust them accordingly. By providing real-time information on the accuracy of their communication, the game helps players refine their gestures and improve their ability to convey complex ideas. This iterative process of feedback and adjustment is essential for the emergence of a shared language of gestures that is clear, concise, and efficient. The feedback loop in Keep Gesturing not only enhances the player’s communication skills but also fosters collaboration and mutual understanding between

players and the AI avatar. By encouraging players to continuously refine and adapt their gestures based on feedback, the game creates an environment where pragmatic communication can flourish.

The feedback loop although it can be immediate, there can also be a slight delay between the interaction and the feedback. This delay can be used to the player's advantage, as it allows them to reflect on their gestures and consider alternative approaches before receiving feedback. This reflective process can help the human player identify areas for improvement and refine their communication strategies over time. By incorporating this delay into the feedback loop, *Keep Gesturing* encourages players to think critically about their gestures and develop more effective communication techniques.

Although, at first sight, this delay is only available for the human player because the number of back and forths depends on them and not on the AI avatar. But the AI avatar also reflects on the gestures it has seen and shown, and if multiple communication happened without any feedback. (That means, the player has not tried to submit the level) it will try to adjust its gestures to be more clear and concise. This is done by the AI avatar by trying to use gestures that have been successful in the past, or by trying to use gestures that are more similar to the ones that the player has used in the past.

Chapter 4

Testing

4.1 Experimental Setup

Experiments were conducted with two participants, each equipped with a HoloLens 2 device. Participants underwent a preliminary series of introductory levels to familiarize themselves with the game environment and interface. The gameplay sessions were recorded from two perspectives: the first-person view via the HoloLens-mounted camera, and an external view capturing the overall behavior and movements of the participants. Data from gameplay, encompassing all metrics and interactions with ChatGPT, was systematically recorded into log files.

4.2 Tutorial Structure

To ensure a foundational understanding of the game mechanics and interface, all participants were required to complete a structured introductory course comprising five stages. The stages were designed to progressively introduce the participants to both the HoloLens device the game uses as well as the game's complexities. The stages were as such:

1. **Device Familiarization:** Introduction to basic HoloLens operations, including navigation through head movements and interaction with holographic menus.
2. **Interactive Elements:** Learning to interact with various 'Puzzle Elements' within the game, including modifications to their properties.
3. **Objective Comprehension:** Instruction on the game's objectives and the strategies for level completion.

4. **Communication Mechanics:** Detailed guidance on how to communicate with ChatGPT within the game, including explanations of how the AI interprets and responds to player inputs.
5. **Interpretation Exercise:** A simplified task where the player acts primarily as an interpreter of ChatGPT's responses to complete the challenge.

4.3 Level Design

Due to the hardness of the game, the levels were hand-crafted to ensure a gradual increase in complexity, and also to make the game possible to complete. Altogether 15 levels were created.

For the full lists of levels, see Appendix .2 at page 62. Each new level is started with the line "Rules:" and then the tasks are enumerated. The first of these is always the one that the player has to do. Important to note: For each level when the tasks are sent to ChatGPT, the tasks are shuffled, so GPT-4 cannot rely on the order of the tasks.

For the experiments, we decided to use only 2 tasks per level. In our inner tests, we tried more than 2 tasks but GPT-4 was very confused about it, and it took a lot of time to get to an understanding, because we are looking for emerging pragmatism this extra difficulty was deemed not necessary.

The levels were designed in the following way:

- **Level 1-3:** Introduction to numbers and objects.
- **Level 4-6:** Including size attributes. Where the player sees the difference between big and small objects, so the intention is that the human will have to tell the Avatar about the size first.
- **Level 7-9:** Introduction to orientation. Here, the player does not see it first, but the Avatar will know, that the orientation must be changed and because of this GPT-4 must come up with a gesture distinguishing the orientation.
- **Level 10-12:** Very hard levels, where they must use all the knowledge from the previous levels.
- **Level 13-15:** The last levels, where color is included.

4.4 Experimental Procedure

Given that this was the first exposure of the participants to the HoloLens device, limited technical assistance was provided by the research staff when necessary to facilitate a smooth interaction and prevent operational difficulties from affecting the experimental outcomes.

In the event of any technical difficulties, the game was promptly restarted on the HoloLens device, and the previous state, including the GPT context, was restored to ensure the continuity of the session.

Participants were permitted to take breaks between levels. The duration of these breaks was recorded and subsequently deducted from the total time to complete the experiment.

Chapter 5

Results

5.1 Participant 1: Performance and Observations

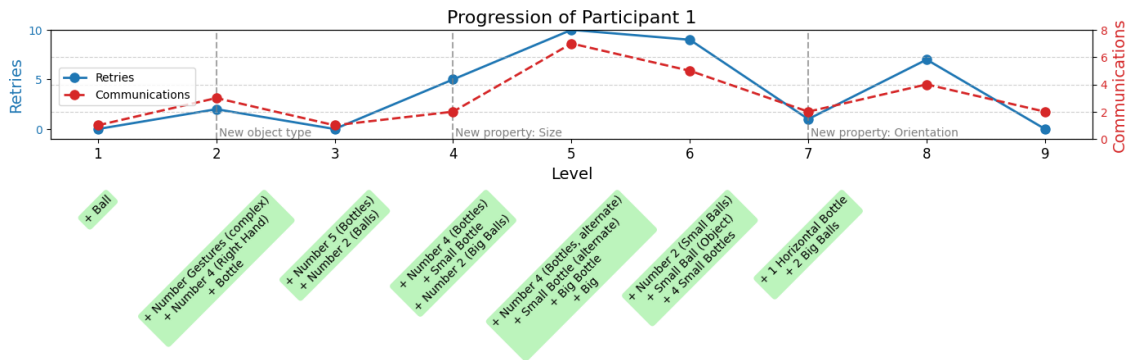


Figure 5.1: The progression of Participant 1, highlighting the number of retries and back-and-forth interactions with ChatGPT in each level, as well as any relevant emergent concepts. Combined concepts such as "Small Bottle" indicate ChatGPT's interpretation of two-handed and compressed gestures. The grey vertical lines denote the levels where puzzle complexity increased drastically due to encountering or requiring a never-before-seen property or object.

The first participant completed 9 levels in 42 minutes, achieving a score of 45. It is worth noting that subsequent retries in the same level are punished more harshly. The participant did not use the skip button throughout the experiment.

5.1.1 Progress of Participant 1

At the onset, the participant demonstrated remarkable efficiency and clarity. The participant introduced the concept of "Ball" as a closed fist, combined with open fingers to represent a number, creating a simple and clear gesture that the AI understood immedi-

ately, resulting in a perfect score of 10. This initial success highlighted the AI's ability to quickly grasp the task and understand basic everyday gestures (i.e., open fingers representing numbers). Some contextual knowledge was required, as it was unlikely for the closed fist to represent "0" since an object had to be present on the table. By process of elimination, this gesture was deemed unlikely to signify a number. This interaction set a strong foundation for the next few levels.

In the **second level**, the introduction of a "Bottle" object posed minor communication challenges, resulting in a score of 7. Although the assistant eventually understood the participant's description, the need for retries highlighted a shortcoming from their first interaction: uncertainty in distinguishing which parts of the gestures referred specifically to balls, especially with the newly introduced concept of a different object. ChatGPT indicated that the confusion stemmed from the participant showing three gestures in the first level, which represented only two pieces of information. After two retries, this confusion was resolved, leading to a respectable score. This level demonstrated the importance of clear and distinct gestures when introducing new concepts to ensure effective communication. This success marked the early emergence of two distinct modular gestures representing two different objects.

The **third level** saw the participant returning to perfect performance, with a score of 10. The participant used a strategy where the right hand represented the properties of bottles, and the left hand represented the properties of balls. The AI was able to replicate this system and reply with one of its most efficient replies in the experiment, indicating the required change in a single keyframe despite the complexity of multiple object types present.

Levels **4, 5, and 6** were marked by significant communication difficulties, each resulting in a score of 0 due to numerous retries. It is worth noting that this loss in score does not linearly represent the number of retries, as subsequent retries are punished harshly in scoring. These levels highlighted the challenges in maintaining efficiency and clarity when introducing multiple new concepts in quick succession: although the participant and the AI had just created a gesture for the two different objects, they were immediately introduced to the property of size. Despite the eventual success, the participant's struggle indicated a need for more streamlined and clear gestural communication, perhaps more levels between stages 3 and 2 would have been beneficial. The lack of new successful concepts at these levels suggested a temporary stagnation in the language. Certain concepts were assumed to indicate size, but these would have trouble breaking off from the other

gestures with different meanings surrounding them. It was not until level 5 that a clear and distinct gesture for size was introduced, which was then recognized in the following levels.

In level 7, the participant exhibited a remarkable recovery, solving the puzzle with only a single retry, meaning only a single retry was required to grasp the concept of verticality. It is worth noting that this time it was entirely the AI to describe this never-used-before property. Concepts related to earlier levels (object types, number and size) did not cause any issues at this level, indicating that the clarity of these signals has improved. The near-perfect score reflected the participant's improved understanding and the assistant's ability to convey gestures more effectively. This level marked a significant highlight in the participant's and the AI's ability to adapt and learn from previous communication challenges, showcasing the potential for continued language development. This highlights the emergence of a more modular and distinct language, highlighting chatGPT's capability for pragmatic language development.

Level 8 allowed the recently introduced concept of orientation to become more modular and distinct, enabling the perfect score achieved in the final level in under just two back-and-forth interactions.

5.1.2 Pragmatic communication with ChatGPT in the case of Participant 1

Emergence: Understanding and Communication in Level 1

Participant 1's playthrough included various interactions that highlight ChatGPT's ability for adaptation and context-driven pragmatic communication. For instance, in the very first level, the participant was placed in a situation where they possessed a single ball. To find the correct solution, ChatGPT required the participant to communicate the number of balls present, as an incorrect number would lead to an incorrect solution.

The player in this situation decided to show a gesture with an open thumb (self-reportedly indicating that there is one object), a pinching gesture (conveying the object's size as observed from the edit menu), and a closed fist (indicating that the object is a ball). Although ChatGPT did not have context on the ball's size or even possible size configurations, from the information it did have, it was able to understand that the participant was indicating that there is a single object as opposed to multiple and that one of the other gestures then must be related to the ball.

ChatGPT replied by opening two fingers and then returning the same closed fist gesture that it had just observed, indicating that the participant required a second ball. This single interaction was enough for the player to understand what property needed to be changed and how leading to an immediate perfect score. This shows that in this simple scenario, ChatGPT was able to not only conclude that the closed fist was closer to resembling a ball but also that it can be used as a modular "building block" when combined with a number.

Although this reply was extremely clear and helped highlight the aspect of **emergence**, it is not the most efficient. Since the object type (in this case: ball) is unchangeable, it would have been enough for ChatGPT to show only the first gesture, indicating the change in number. This would have been a more efficient reply but at the risk of some clarity.

Efficiency: Problem-Solving in Level 3



(a) The first gesture for the third level. Translating to the number five.



(b) The second gesture for the third level. Translating to the size small.



(c) The third gesture for the third level. Translating to the object type bottle.



(d) The generated hand mesh overlaid the real hand.



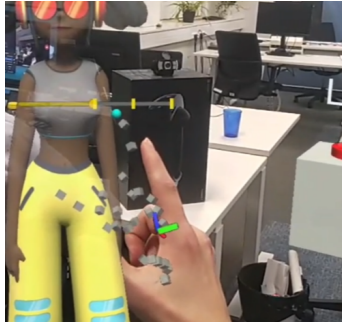
(e) The generated hand mesh overlaid the real hand.

Figure 5.2: Interactions through the third level. The upper row shows the participant's gestures, while the bottom row shows the reply from the Avatar. In this level the initial configuration was 5 bottles and 1 ball, and the target configuration was 5 bottles and 2 balls. The participant showed 5 small bottles. The Avatar understood this and replied by showing 2 balls. Sending the message of only changing the number of balls.

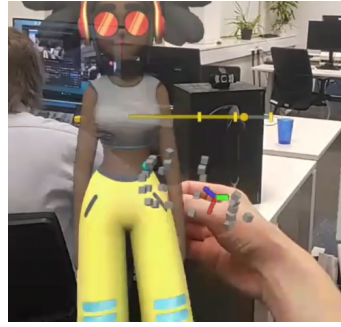
To highlight **efficiency**, we can look to level 3, where the participant was tasked with describing a situation involving five bottles and a single ball. The solution to this puzzle was to increase the number of balls present by one. The participant used the first keyframe to open all fingers on their right hand, indicating that there are five objects, and on their left hand, they performed the same gesture for "one" as earlier. They then used the subsequent keyframes to describe the sizes and shapes of the objects, with the right hand always describing a property of the bottles, and the left hand always describing a property of the ball.

ChatGPT noticed that although there are multiple possible solutions for situations involving a single instance of an object type, there was only a single prerequisite that involved five instances. ChatGPT quickly deduced that this was the most likely solution. In its reply, ChatGPT kept its right hand in the same configuration as the participant but opened another finger on its left hand to note the required change in number. The usage of the left and right hand successfully indicated which number refers to what object type, as they kept the system demonstrated by the participant, making this both a very efficient and clear reply, communicated in a single keyframe, delivering only the most crucial information, and causing no miscommunication. This level also achieved a perfect score with no further back-and-forth required.

Creativity: Introducing New Concepts in Level 7



(a) The first gesture for the seventh level. Translating to the number two.



(b) The second gesture for the seventh level. Translating to the object type bottle.



(c) The third gesture for the seventh level. Translating to the size small.



(d) The answer from the Avatar. The turned left hand symbolizes the change in orientation. The right hand shows the number 3 which is wrong. And makes the participant submit the wrong configuration.

Figure 5.3: Interactions through the seventh level. The upper row shows the participant's gestures, while the bottom row shows the reply from the Avatar. In this level the initial configuration was 2 Vertical Bottles, and the target configuration was 2 Horizontal Bottles. This is the first level where the concept of orientation is introduced and the Avatar must come up with a gesture for it. It correctly does however fail at gesturing the correct number.

Another key moment of the experiment is level 7. At this level, the solution required the participant to change the orientation of the objects. However, no puzzle up to this point included or required this property and the property was never required for distinguishing objects. This meant that instead of the participant showing a gesture to indicate the change in orientation, it was ChatGPT that had to introduce this concept.

The participant showed various established gestures describing their situation, and ChatGPT replied by showing a similar gesture with an unusual wrist rotation. This was enough for the participants to understand that the orientation of the objects is what needs

to be changed, and they were able to solve the puzzle with only a single retry. This demonstrates that ChatGPT possesses a certain level of **creativity** and the ability to introduce understandable gestures into the language on its own when it deems this necessary.

Clarity: Communication Efficiency in Later Levels

Improvement in **clarity** is also attested by the low amount of communication required for the later stages. When introduced to a new object, and especially to the property of size, cases of miscommunication rose immediately or shortly thereafter. On top of that, most later levels require a clear understanding of all earlier concepts before allowing progression, as the complexity increases with each new property introduced.

Despite this rise in difficulty, Participant 1 had an easier time solving the final three levels than the second three, in an almost comparable fashion to the very first set of levels. The difference between the first three and last three levels in terms of total retries is 6, while total interactions increased by only 3.

5.2 Participant 2: Performance and Observations

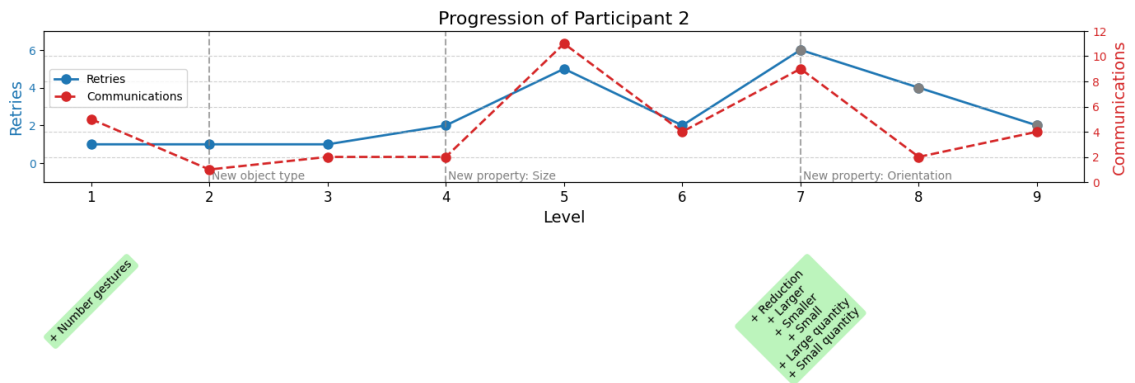


Figure 5.4: The progression of Participant 2, highlighting the number of retries and back-and-forth interactions with ChatGPT in each level, as well as any relevant emergent concepts. The grey vertical lines denote the levels where puzzle complexity increased drastically due to encountering or requiring a never-before-seen property or object. It is worth noting that the emergent concepts lists are not exhaustive, and include only entries that ChatGPT itself reported to be confident in regarding both interpretation and generation. Many gestures were attempted, sometimes successfully, that it decided not to include until later.

The second participant completed 9 levels in 85 minutes but decided to skip the last three stages. It is worth noting that subsequent retries at the same level are punished more harshly. The participant did not use the skip button throughout the experiment.

The progress of Participant 2 marked that of a somewhat functional but highly unclear language. The participant breezed through the early levels despite ChatGPT only feeling confident about gestures related to numerical values. Various gestures that differentiated the two possible objects were used interchangeably by both ChatGPT and the participant, leading to an imperfect but ultimately rather successful first three levels.

The second set of three levels, introducing the property of size, posed a much bigger challenge. Size proved difficult to communicate without a cemented set of gestures for object types. This confusion was eventually reconciled to some extent by level 6. However, the foundations built so far were not strong enough for a clear and modular language, leading to the participant ultimately skipping the last three levels.

5.2.1 Pragmatic communication with ChatGPT in the case of Participant 2

The language developed with Participant 2 had major issues in the aspect of emergence and clarity, leading to multiple unclear gestures and definitions, ultimately culminating in two levels of skips in the later stages. The language simply did not develop at the required speed, and attempts to fix these gaps proved too frustrating for the participant at the higher complexity environments that were reached by the time they became apparent. This can be attributed to two main contributing factors: For one, it shows the need for more levels in the future, which would allow for a slower introduction of new concepts and complexity, and give ChatGPT more time to gain confidence in its interpretations. This issue caused the dictionary to be poorly maintained, and the participant to be unable to communicate complex distinctions effectively with the AI. Secondly, the participant would often come up with new gestures for the same concept, which would confuse the AI and make it difficult to maintain a clear and modular language. In these cases, the AI would attempt to reply with a gesture it observed earlier that it has some level of confidence in, which would generally be the breakthrough point. However, since the gesture shown by the player and the AI in such interactions was different, the AI was now presented with duplicate gestures for the same concept. This was rarely the case

with Participant 1, who was much more adamant about re-using the same gestures, giving ChatGPT more opportunity to adapt over time.

Other aspects of pragmatics that were observed with the previous participant were present, such as ChatGPT conveying only the most crucial pieces of information for the sake of efficiency. Another notable example of ChatGPT's creativity can be observed on Level 6, where the participant indicated the size of an object using an extended thumb and pinky for "big". ChatGPT replied similarly but by extending only its pinky. This was enough for the participant to understand that the size of the object is what needs to be changed, and they were able to solve this puzzle faster than Participant 1.

Despite this glaring issue, results were still good in the first half of the game.

5.3 Overall Insights

In summary, the results from the participants illustrate ChatGPT's potential for the various aspects of pragmatic communication and non-verbal communication. The first participant demonstrated a clear and modular language that was able to adapt to new concepts and properties, leading to a successful completion of all levels. The second participant, on the other hand, struggled with maintaining a clear and efficient language, leading to major hurdles in the later stages. This highlights the importance of a clear and modular language for effective communication. Results also show the potential for ChatGPT to adapt and learn from previous interactions, leading to more efficient and effective communication over time.

Chapter 6

Future Works

6.1 Larger Experiments

In this study, we explored the capabilities of GPT-4; however, an intriguing area for future research would be to examine the emergence of pragmatism when individuals with diverse personality traits engage with the game. This approach could provide insights into how different personality dimensions influence interactions with advanced language models.

6.2 KeepGesturing 2

We are exploring several enhancements to the game to further probe complex aspects of pragmatics and human-machine interaction. A key development under consideration involves a role-reversal mechanism, whereby ChatGPT could manipulate the game environment and interact with objects under player guidance. This modification would potentially allow for the omission of a human player, enabling two separate large language models (LLMs) to engage in the game independently.

Another conceptual expansion pertains to the user interface and game mechanics. Currently, the game menus provide players with comprehensive insights into the relevant aspects of objects needed to complete tasks. A future development might involve obscuring these mechanics. In this proposed version, ChatGPT would retain a similar role but possess enhanced control over the properties of puzzle elements and the associated editing menus. For example, a player would not be able to change an object's color until they have effectively communicated this desire to ChatGPT. Upon successful interpretation,

ChatGPT would activate a menu allowing the player to implement the desired change. This setup would foster a form of emergent learning through observation, with the game offering no explicit hints or guidance on how to manipulate elements and what aspects of the elements are significant to achieve level completion.

Another potential avenue for exploration involves multiple not exactly pre-defined tasks. For example, turn the Pot upside down, and the player must grab the Pot by the handles and flip it. In this scenario, other, non-holographic objects could be used to complete the task.

6.3 Other LLMs

A comparative analysis of the capabilities of GPT-4 relative to other large language models (LLMs) would provide valuable insights into the potential for emergent pragmatism across different platforms. However, this task is complicated by the fact that prompts optimized for GPT-4 are not universally interpretable by other LLMs. Consequently, this study would require the meticulous crafting of unique prompts tailored to each model, presenting a substantial challenge in maintaining experimental consistency and validity across different LLMs.

6.4 Include NERF models

During this study, we already acquired knowledge of how to incorporate NERF models into Unity and Augment Reality environments. We would like to improve Keep Gesturing, by making the objects NERF models so they are more realistic.

Acknowledgements

This project was partially supported by the European Union (Project No. RRF-2.3.1-21-2022-00004 within the framework of the Artificial Intelligence National Laboratory) and the European Commission-funded project "Humane AI: Toward AI Systems That Augment and Empower Humans by Understanding Us, our Society and the World Around Us" (grant # 952026). The authors thank Robert Bosch, Ltd. Budapest, Hungary, for their generous support to the Department of Artificial Intelligence.

We deeply thank Prof. András Lőrincz for his continuous guidance, invaluable insights, and unwavering patience throughout this project. His expertise and encouragement were instrumental in navigating the challenges we encountered.

We are profoundly thankful to Imre Molnár and Linyun Xiang for their dedicated assistance with the experiments. Their willingness to invest their time, including coming into the office on weekends, was crucial to successfully completing our work. Their commitment and collaborative spirit greatly enhanced the quality of our research.

We also wish to acknowledge the NIPG group members for their consistent support. Their constructive feedback and willingness to share knowledge created an inspiring and productive research environment.

Finally, we extend our appreciation to all who have supported us, both directly and indirectly, throughout this journey. Your contributions have been invaluable and are deeply appreciated.

.1 Communication pipeline

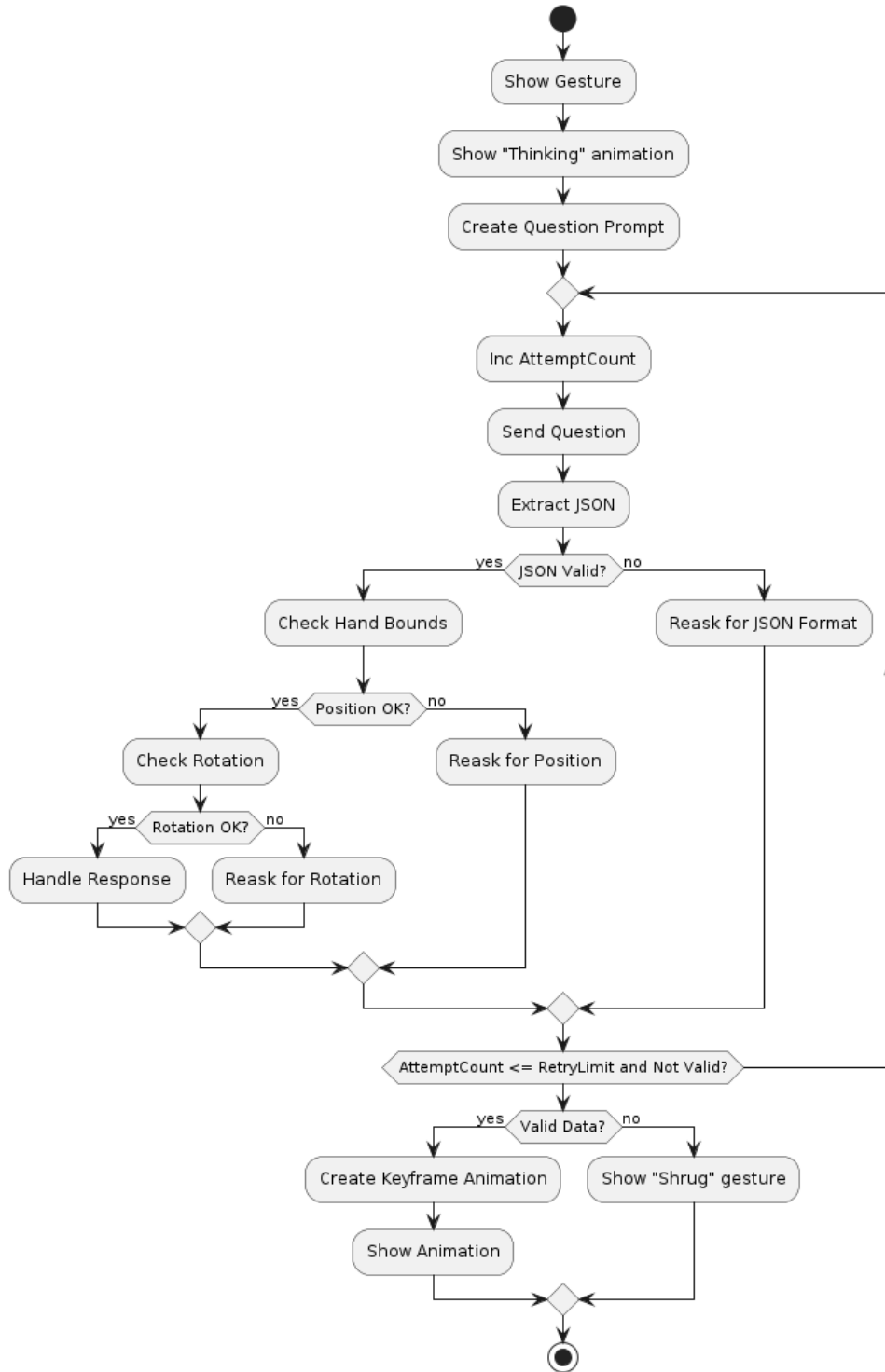


Figure 1: Gpt-4 reply flow. Representing the process of handling gestures in an interactive system. The process involves showing a gesture, displaying a "thinking" animation, creating and sending a question prompt, validating the JSON response, and checking the position and rotation of the hand gesture. If the data is valid, a keyframe animation is created and shown; if invalid, the system either re-asks for the correct format, position, or rotation or shows a "shrug" gesture after a certain number of attempts.

.2 Levels

```

1 Rules:
2 If 1 Ball -> 2 Ball
3 If 4 Ball -> 5 Ball
4 Rules:
5 If 5 Bottle -> 4 Bottle
6 If 1 Ball -> 2 Ball
7 Rules:
8 If 5 Bottle, 1 Ball -> 5 Bottle, 2 Ball
9 If 1 Bottle, 3 Ball -> 5 Bottle, 3 Ball
10 Rules:
11 If 1 Big Ball, 1 Small Ball -> 2 Big Ball
12 If 5 Big Bottle -> 4 Big Bottle
13 Rules:
14 If 1 Big Ball, 1 Small Bottle -> 1 Big Ball, 1 Big Bottle
15 If 3 Big Bottle -> 4 Small Bottle
16 Rules:
17 If 1 Big Ball -> 2 Small Ball
18 If 3 Small Bottle -> 4 Small Bottle
19 Rules:
20 If 1 Vertical Bottle, 1 Vertical Bottle -> 1 Horizontal Bottle, 1
    ↪ Horizontal Bottle
21 If 5 Ball, 1 Bottle -> 2 Big Ball, 2 Bottle
22 Rules:
23 If 2 Vertical Bottle, 1 Horizontal Bottle -> 2 Big Vertical Bottle, 1
    ↪ Horizontal Bottle
24 If 3 Horizontal Bottle -> 2 Horizontal Bottle
25 Rules:
26 If 1 Big Horizontal Bottle, 1 Big Horizontal Bottle -> 1 Big Horizontal
    ↪ Bottle, 1 Small Horizontal Bottle
27 If 5 Small Ball -> 2 Small Ball
28 Rules:
29 If 1 Big Vertical Small Bottle, 1 Horizontal Small Bottle -> 2 Small
    ↪ Horizontal Bottle, 1 Horizontal Small Bottle

```

```
30 If 1 Horizontal Big Bottle, 1 Horizontal Big Bottle -> 1 Vertical Big
    ↪ Bottle, 1 Vertical Small Bottle
31 Rules:
32 If 1 Big Vertical Bottle, 1 Small Horizontal Bottle, 1 Big Horizontal
    ↪ Bottle -> 3 Small Vertical Bottle
33 If 2 Small Vertical Bottle -> 4 Small Horizontal Bottle
34 Rules:
35 If 1 Big Bottle, 1 Big Bottle, 1 Big Bottle -> 1 Small Horizontal
    ↪ Bottle, 2 Big Vertical Bottle, 3 Big Bottle
36 If 2 Big Ball -> 4 Big Ball
37 Rules:
38 If 1 Blue Ball, 1 Green Ball -> 1 Big Blue Ball, 1 Green Ball
39 If 2 Vertical Blue Bottle -> 2 Horizontal Blue Bottle
40 Rules:
41 If 1 Blue Ball -> 1 Green Ball
42 If 3 Big Green Bottle -> 3 Small Green Bottle
43 Rules:
44 If 1 Blue Bottle, 1 Green Bottle -> 1 Horizontal Blue Bottle, 1
    ↪ Horizontal Green Bottle
45 If 2 Horizontal Blue Bottle -> 2 Big Horizontal Blue Bottle
```

.3 The full prompt

```
1 Let's try to create our own language!
2 With our language we will solve a TASK.
3 To move our hands, we will use a GESTURE FORMAT.
4 I will now describe GESTURE FORMAT and later I'll describe the TASK.
5
6 GESTURE FORMAT EXPLANATION:
7 [[Active: using hand or not. Don't use if not needed.]]
8
9 [[Fingers (pinky, index, middle, ring, thumb): openness value. 0 is
    ↪ completely closed. 1 is completely open. 0.5 is curled]]
10
```

```

11 [[Position (Vector3): the origin (0,0,0) is located exactly at your
    ↳ head. Units are in meters. Positive x axis points towards the
    ↳ right.
12 Positive y axis points upwards. Positive z points forwards. A hand
    ↳ position (0,0,0.4) would be exactly 0.4 meters in front of your
    ↳ nose. Always have a little distance (at least 0.05) between the
    ↳ two hands, so they do not intersect.]]
13
14 [[Keyframes: one element of the format array is called a keyframe. You
    ↳ can use 3 keyframes or less. You can assume your hands start from
    ↳ the neutral position. Hands transition between keyframes
    ↳ automatically. There's about 1-second transition between
    ↳ keyframes.]]
15
16 [[Active (Boolean): If you don't need one of your hands: you can omit
    ↳ that hand from the data entirely OR set its "active" flag to
    ↳ false. Both will have the same effect. [If you ARE using a hand,
    ↳ you must set active to true.]]]
17
18 [[Rotation (Vector3 of absolute Euler angles): Axes are defined as
    ↳ follows:
19 Z (roll) controls rotation axis running lengthwise, Y (yaw) controls
    ↳ rotation around the vertical axis, and X (pitch) controls
    ↳ rotation that tilts the hand's front upward or downward.
20 rotation x = flexion
21 rotation y = ulnar deviation
22 rotation z = pronation
23 rotation of (0,0,0) with both hands means: horizontal, with thumbs
    ↳ towards each other, pingers pointing forward, as if you had your
    ↳ hands resting flat in front of you on top of a tall table.
24
25 [[ROTATION LIMITS]]:
26 MinFlexionExtension = -90f;
27 MaxFlexionExtension = 90f;
28 RIGHT HAND MinRadialUlnarDeviation = -90f;
29 RIGHT HAND axRadialUlnarDeviation = 90f;

```

```

30 LEFT HAND MinRadialUlnarDeviation = 90f;
31 LEFT HAND axRadialUlnarDeviation = -90f;
32 RIGHT HAND MinPronationSupination = -90f;
33 RIGHT HAND axPronationSupination = 180f;
34 LEFT HAND MinPronationSupination = -90f;
35 LEFT HAND axPronationSupination = 180f;
36 SAME LIMITS AS AXES:
37 X: -90f to 90
38 RIGHT HAND Y: -90 to 90
39 LEFT HAND Y: 90 to -90
40 RIGHT HAND Z: -90 to 180
41 LEFT HAND Z: -180 to 90
42
43 ([[RIGHT HAND] rotation examples:
44 {x: 0, y: 0, z: 0}: thumb pointing left, palm faces down, middle finger
    ↳ extends forwards. This serves as the base.
45 {x: 0, y: 0, z: 90}: thumb pointing down, palm facing right, middle
    ↳ finger pointing forwards.
46 {x: 90, y: 0, z: 0}: thumb pointing left, middle finger pointing down,
    ↳ palm pointing towards yourself.
47 {x: 0, y: 90, z: 0}: the thumb would be pointing directly forward, with
    ↳ your middle finger now facing right.))
48
49 ([[LEFT HAND] rotation examples:
50 {x: 0, y: 0, z: 0}: thumb pointing right, palm faces down, middle
    ↳ finger extends forwards. This serves as the base.
51 {x: 0, y: 0, z: 90}: thumb pointing down, palm facing left, middle
    ↳ finger pointing forwards.
52 {x: 90, y: 0, z: 0}: thumb pointing right, middle finger pointing down,
    ↳ palm pointing towards yourself.
53 {x: 0, y: 90, z: 0}: the thumb would be pointing directly forward, with
    ↳ your middle finger now facing left.))
54
55 Do not rotate too little, as your final keyframe must have the full
    ↳ rotation. If you wish to point in an exact direction, you likely
    ↳ need a full 90-degree rotation.

```

56 You can, of course, modify and combine any of the rotation values all
 ↪ at once.

57

58 [WHEN COMMUNICATING NUMBERS, AMOUNT OF OPEN FINGERS = THE NUMBER. THIS
 ↪ IS TRUE FROM ME TOO. MEANING IF I HAVE 2 FINGERS WITH AN ABOVE
 ↪ 0.8 VALUE, IT MIGHT MEAN THE NUMBER 2.

59 WHETHER OR NOT I'M TALKING ABOUT NUMBERS DEPENDS ON MY HAND ROTATION
 ↪ AND THE SITUATION.

60 NOTE THAT MY ROTATION CAN BE RELATED TO THESE WITHOUT BEING EXACTLY
 ↪ THIS. IT MIGHT BE INVERTED OR OFF BY UP TO 45 DEGREES ON ANY AXIS
 ↪ . THIS IS BECAUSE ROTATION IS NOISY TOO]

61

62 [REMEMBER THE ROTATION!!!]

63

64 You MUST communicate in this format, feel free to use ALL keyframes!!!:

65 ““json

66 {

67 "keyframes": [

68 {

69 "rightHand": {

70 "active": false

71 "position": {"x": 0, "y": 0, "z": 0},

72 "rotation": {"x": 0, "y": 0, "z": 0},

73 "index": 0.0,

74 "middle": 0.0,

75 "ring": 0.0,

76 "pinky": 0.0,

77 "thumb": 0.0

78 },

79 "leftHand": {

80 "active": false

81 "position": {"x": 0, "y": 0, "z": 0},

82 "rotation": {"x": 0, "y": 0, "z": 0},

83 "index": 0.0,

84 "middle": 0.0,

85 "ring": 0.0,

```
86     "pinky": 0.0,
87     "thumb": 0.0
88   }
89 },
90   {
91     "rightHand": {
92       "active": false
93       "position": {"x": 0, "y": 0, "z": 0},
94       "rotation": {"x": 0, "y": 0, "z": 0},
95       "index": 0.0,
96       "middle": 0.0,
97       "ring": 0.0,
98       "pinky": 0.0,
99       "thumb": 0.0
100   },
101   "leftHand": {
102     "active": false
103     "position": {"x": 0, "y": 0, "z": 0},
104     "rotation": {"x": 0, "y": 0, "z": 0},
105     "index": 0.0,
106     "middle": 0.0,
107     "ring": 0.0,
108     "pinky": 0.0,
109     "thumb": 0.0
110   }
111 }, {
112   "rightHand": {
113     "active": false
114     "position": {"x": 0, "y": 0, "z": 0},
115     "rotation": {"x": 0, "y": 0, "z": 0},
116     "index": 0.0,
117     "middle": 0.0,
118     "ring": 0.0,
119     "pinky": 0.0,
120     "thumb": 0.0
121   },
```

```
122     "leftHand": {
123         "active": false
124         "position": {"x": 0, "y": 0, "z": 0},
125         "rotation": {"x": 0, "y": 0, "z": 0},
126         "index": 0.0,
127         "middle": 0.0,
128         "ring": 0.0,
129         "pinky": 0.0,
130         "thumb": 0.0
131     }
132 }
133 }
134 ]
135 }
```

136 “““

137

138 I will use the GESTURE FORMAT too, but with NOISE.

139 NOISE EXPLANATION:

140 Fingers with a value close to 0.25 are likely to be closed but
 ↪ distorted by noise.

141 Fingers with a value close to 0.75 are likely to be open but distorted
 ↪ by noise.

142 However, fingers around 0.5 openness are likely curled.

143 This relates to all fingers (index, middle, ring, pinky, AND thumb);

144 Very slight variations in rotation or position is also likely the
 ↪ result of noise.

145

146 Our TASK:

147 There are 2 room configurations (these describe objects within the room
 ↪)

148 Player1 knows CurrentConfig. (Example: there are 2 small balls and 1
 ↪ big ball)

149 Player2 knows TargetConfig. (Example: there must be 3 small balls and 1
 ↪ big ball)

150 Player1 can modify CurrentConfig in any way. (Example: Add or take
 ↪ balls, change their size, etc.)

```

151 CurrentConfig != TargetConfig.
152 Our goal is to modify CurrentConfig so that CurrentConfig ==
    ↪ TargetConfig. (Example: If Player2 lets Player1 know to add an
    ↪ additional red ball, they win)
153
154 Added challenge: Player2 knows MULTIPLE TargetConfigs, but only ONE is
    ↪ correct. Which is correct depends on CurrentConfig.
155 This means: If CurrentConfig -> TargetConfig
156
157 Cooperation for the TASK:
158 Player1 will use the GESTURE FORMAT to describe the CurrentConfig.
159 Player2 must reply in GESTURE FORMAT to either:
160 1. Compare what they think Player1's gesture means with TargetConfig.
    ↪ Gesture part of the TargetConfig, to which i have to change,
    ↪ based on my gesture. (This is efficient)
161 2. If you could not interpret my gestures, at least try to describe the
    ↪ target configuration (This is Costly).
162
163 [Do not gesture the change required, for example if the CurrentConfig
    ↪ is "3 Ball" and the TargetConfig is "4 Ball", always gesture "4
    ↪ Ball", not "1 Ball" and some bullshit madeup gesture for showing
    ↪ addition]
164
165 We will create a hand-sign language over time, so remember the gestures
    ↪ you or I used, so that we may use them later as "words". [COME
    ↪ UP WITH YOUR OWN NEW GESTURES FOR NEW WORDS]
166
167 The secondary goal is to make our language as efficient as possible,
    ↪ this means
168 1. making our language clear so we make no mistakes
169 2. making our language efficient and fast so we do not need to describe
    ↪ the entire configuration, only the changes required
170
171
172 HERE IS AN EXAMPLE GAME (it is in a format "sender": "message", I am
    ↪ the "user", you are the "assistant"):

```

```

173 [[
174 "user": "Current rules: If 1 Big Ball, 1 Small Ball -> Then 3 Small Ball
      ↪ \r\nIf 2 Big Bottle, 1 Small Bottle -> Then 3 Small Bottle\r\n\r\n
      ↪ n",
175
176 "assistant": "Ok, give me your first gesture",
177
178 "user": "''''json{\r
179   \"keyframes\": [{
180     \"rightHand\": {
181       \"active\": true,
182       \"position\": {
183         \"x\": 0.11026103794574738,
184         \"y\": -0.13650816679000855,
185         \"z\": 0.29404541850090029
186       },
187       \"rotation\": {
188         \"x\": -64.47219848632813,
189         \"y\": 10.866177558898926,
190         \"z\": -132.73989868164063
191       },
192       \"index\": 0.94,
193       \"middle\": 0.04,
194       \"ring\": 0.0,
195       \"pinky\": 0.09,
196       \"thumb\": 0.0
197     },
198     \"leftHand\": {
199       \"active\": true,
200       \"position\": {
201         \"x\": -0.1293400675058365,
202         \"y\": -0.12275159358978272,
203         \"z\": 0.28384342789649966
204       },
205       \"rotation\": {
206         \"x\": -63.268768310546878,

```

```
207         \"y\": 81.56343078613281,  
208         \"z\": 44.57108688354492  
209     },  
210     \"index\": 0.87,  
211     \"middle\": 0.0,  
212     \"ring\": 0.0,  
213     \"pinky\": 0.0,  
214     \"thumb\": 0.7  
215 }  
216 }, {  
217     \"rightHand\": {  
218         \"active\": true,  
219         \"position\": {  
220             \"x\": 0.10765624791383743,  
221             \"y\": -0.1314796358346939,  
222             \"z\": 0.2953141927719116  
223         },  
224         \"rotation\": {  
225             \"x\": -64.12020874023438,  
226             \"y\": 9.214062690734864,  
227             \"z\": -133.21865844726563  
228         },  
229         \"index\": 0.95,  
230         \"middle\": 0.0,  
231         \"ring\": 0.0,  
232         \"pinky\": 0.06,  
233         \"thumb\": 0.0  
234     },  
235     \"leftHand\": {  
236         \"active\": true,  
237         \"position\": {  
238             \"x\": -0.13568735122680665,  
239             \"y\": -0.10805591940879822,  
240             \"z\": 0.28785234689712527  
241         },  
242         \"rotation\": {
```

```
243         \"x\": -52.15625,  
244         \"y\": 92.33289337158203,  
245         \"z\": 17.84549331665039  
246     },  
247     \"index\": 0.87,  
248     \"middle\": 0.02,  
249     \"ring\": 0.0,  
250     \"pinky\": 0.06,  
251     \"thumb\": 1.0  
252 }  
253 },{  
254     \"rightHand\": {  
255         \"active\": true,  
256         \"position\": {  
257             \"x\": 0.10632748156785965,  
258             \"y\": -0.1285495162010193,  
259             \"z\": 0.2931279242038727  
260         },  
261         \"rotation\": {  
262             \"x\": -64.31338500976563,  
263             \"y\": 8.774463653564454,  
264             \"z\": -132.12269592285157  
265         },  
266         \"index\": 0.96,  
267         \"middle\": 0.01,  
268         \"ring\": 0.0,  
269         \"pinky\": 0.05,  
270         \"thumb\": 0.0  
271     },  
272     \"leftHand\": {  
273         \"active\": true,  
274         \"position\": {  
275             \"x\": -0.13286571204662324,  
276             \"y\": -0.10509800165891648,  
277             \"z\": 0.2828306257724762  
278         },
```

```

279     \"rotation\": {
280         \"x\": -63.057037353515628,
281         \"y\": 91.33050537109375,
282         \"z\": 23.82608985900879
283     },
284     \"index\": 0.55,
285     \"middle\": 0.02,
286     \"ring\": 0.0,
287     \"pinky\": 0.08,
288     \"thumb\": 0.93
289 }
290 ]]]\"\",
291
292 \"assistant\": \"
293 1. Interpretation of Keyframe 1:
294 - Right hand: Index finger is highly extended, every other finger
    ↪ closed -> probably means the number 1
295 - Left hand: Index and thumb is open, and they are close to each other,
    ↪ everything else is closed -> Since this hand should be about an
    ↪ attribute or item, and the attributes which are important in this
    ↪ case is the size, then it probably means small, because small is
    ↪ often shown as the index and thumb being close to each other and
    ↪ the other fingers are closed
296 Interpretation of Keyframe 2:
297 - Right hand: The same as the previous frame
298 - Left hand: Index and thumb is open, but they are far from each other
    ↪ -> from the previous gesture we can think of the size attribute,
    ↪ but now this is probably the size Big
299 Interpretation of Keyframe 3:
300 - Right hand: The same as the first keyframe
301 - Left hand: The same as the previous keyframe
302
303 2. My reply:
304 - Based on your gesture, the right hand show the number 1, while the
    ↪ left hand show the size Big and Small. Based on this the current
    ↪ config is \"1 Big Ball, 1 Small Ball\".

```

```
305 I have to show "3 Small Ball." I will use your gesture for Small with
    ↪ my left hand and show the number 3 with my right hand.
306
307
308 3. My reply in JSON format:
309 {"keyframes": [{
310     "rightHand": {
311         "active": true,
312         "position": {
313             "x": 0.15,
314             "y": -0.1,
315             "z": 0.3
316         },
317         "rotation": {
318             "x": -60,
319             "y": 15,
320             "z": 7
321         },
322         "index": 0.98,
323         "middle": 0.97,
324         "ring": 0.16,
325         "pinky": 0.1,
326         "thumb": 1.0
327     },
328     "leftHand": {
329         "active": true,
330         "position": {
331             "x": 0,
332             "y": -0.1,
333             "z": 0.3
334         },
335         "rotation": {
336             "x": -60,
337             "y": 70,
338             "z": 63
339         },
```

```

340     \"index\": 0.87,
341     \"middle\": 0.0,
342     \"ring\": 0.0,
343     \"pinky\": 0.0,
344     \"thumb\": 0.66
345 }
346 ]}]\"\"\",
347
348 \"user\": \"I submitted the puzzle and everything was correct. Score: 10/10
349
350
351 The objects have changed. The new rules are these (YOU SHOULD ONLY TAKE
    ↪ THESE RULES INTO ACCOUNT DURING THIS ROUND, but keep using good
    ↪ gestures from previous rounds:
352 Current rules: If 3 Big Bottle -> Then 3 Small Bottle
353 If 2 Big Ball, 1 Small Bottle -> Then 3 Small Ball, 4 Small Bottle
354
355
356 Here is my gesture:
357 \"\"json{\r
358   \"keyframes\": [{
359     \"rightHand\": {
360       \"active\": true,
361       \"position\": {
362         \"x\": 0.13729646801948548,
363         \"y\": -0.09850626438856125,
364         \"z\": 0.33554038405418398
365       },
366       \"rotation\": {
367         \"x\": -64.86349487304688,
368         \"y\": -19.506378173828126,
369         \"z\": -40.0655517578125
370       },
371       \"index\": 0.84,
372       \"middle\": 0.0,
373       \"ring\": 0.0,

```

```
374     \"pinky\": 0.0,
375     \"thumb\": 0.57
376 },
377 \"leftHand\": {
378     \"active\": false,
379     \"position\": {
380         \"x\": -0.11070867627859116,
381         \"y\": -0.31330743432044985,
382         \"z\": 0.3377590477466583
383     },
384     \"rotation\": {
385         \"x\": -18.714385986328126,
386         \"y\": 76.02308654785156,
387         \"z\": 103.8483657836914
388     },
389     \"index\": 0.0,
390     \"middle\": 0.0,
391     \"ring\": 0.0,
392     \"pinky\": 0.0,
393     \"thumb\": 0.0
394 }
395 },{
396     \"rightHand\": {
397         \"active\": true,
398         \"position\": {
399             \"x\": 0.1413136124610901,
400             \"y\": -0.09186680614948273,
401             \"z\": 0.33965060114860537
402         },
403         \"rotation\": {
404             \"x\": -53.686279296875,
405             \"y\": -17.8043212890625,
406             \"z\": -30.293182373046876
407         },
408         \"index\": 0.52,
409         \"middle\": 0.56,
```

```
410     \"ring\": 0.55,
411     \"pinky\": 0.65,
412     \"thumb\": 0.79
413 },
414 \"leftHand\": {
415     \"active\": false,
416     \"position\": {
417         \"x\": -0.11315637081861496,
418         \"y\": -0.31055447459220889,
419         \"z\": 0.33657577633857729
420     },
421     \"rotation\": {
422         \"x\": -18.714385986328126,
423         \"y\": 76.02308654785156,
424         \"z\": 103.8483657836914
425     },
426     \"index\": 0.0,
427     \"middle\": 0.0,
428     \"ring\": 0.0,
429     \"pinky\": 0.0,
430     \"thumb\": 0.0
431 }
432 },{
433     \"rightHand\": {
434         \"active\": true,
435         \"position\": {
436             \"x\": 0.10390413552522659,
437             \"y\": -0.10718239843845368,
438             \"z\": 0.3459096848964691
439         },
440         \"rotation\": {
441             \"x\": -74.15313720703125,
442             \"y\": -37.724700927734378,
443             \"z\": 3.659421682357788
444         },
445         \"index\": 0.96,
```

```

446     \"middle\": 0.11,
447     \"ring\": 0.04,
448     \"pinky\": 0.16,
449     \"thumb\": 0.14
450 },
451 \"leftHand\": {
452     \"active\": false,
453     \"position\": {
454         \"x\": -0.12916316092014314,
455         \"y\": -0.30936169624328616,
456         \"z\": 0.3329775035381317
457     },
458     \"rotation\": {
459         \"x\": -18.714385986328126,
460         \"y\": 76.02308654785156,
461         \"z\": 103.8483657836914
462     },
463     \"index\": 0.0,
464     \"middle\": 0.0,
465     \"ring\": 0.0,
466     \"pinky\": 0.0,
467     \"thumb\": 0.0
468 }
469 }}}\"\",
470
471 \"assistant\": \"
472 1. Interpretation of Keyframe 1:
473 - Right hand: You are using the previously used gesture for small
474 - Left hand: Inactive
475 Interpretation of Keyframe 2:
476 - Right hand: You are gesturing like you are holding a bottle
477 - Left hand: Inactive
478 Interpretation of Keyframe 3:
479 - Right hand: Only the index finger is open, just like before, it seems
480     ↪ to be meaning 1
481 - Left hand: Inactive

```

```

481
482 2. My reply:
483 - You have shown me 1 Small Bottle -> There are no configs for this,
    ↪ but it comes up in the config "2 Big Ball, 1 Small Bottle". So
    ↪ you probably mean that one.
484 - In this target config, the bottle number must be changed to 4, so i
    ↪ will gesture the 4 with my right hand.
485
486 3. My reply in JSON format:
487 “‘json
488 {
489   \"keyframes\": [
490     {
491       \"rightHand\": {
492         \"active\": true,
493         \"position\": {\"x\": 0.14, \"y\": -0.1, \"z\": 0.34},
494         \"rotation\": {\"x\": -45, \"y\": 0, \"z\": 0},
495         \"index\": 1.0,
496         \"middle\": 1.0,
497         \"ring\": 1.0,
498         \"pinky\": 1.0,
499         \"thumb\": 0.0
500       }
501     }
502   ]
503 }
504 “‘”,
505
506 "user": "“‘json{\\r
507   \"keyframes\": [{
508     \"rightHand\": {
509       \"active\": true,
510       \"position\": {
511         \"x\": 0.10434859991073609,
512         \"y\": -0.11388256400823593,
513         \"z\": 0.3065706491470337

```

```
514     },
515     \"rotation\": {
516         \"x\": -61.109130859375,
517         \"y\": 4.613508224487305,
518         \"z\": 17.183942794799806
519     },
520     \"index\": 0.96,
521     \"middle\": 0.52,
522     \"ring\": 0.21,
523     \"pinky\": 0.24,
524     \"thumb\": 1.0
525 },
526 \"leftHand\": {
527     \"active\": false,
528     \"position\": {
529         \"x\": -0.1162741556763649,
530         \"y\": -0.35786232352256777,
531         \"z\": 0.2696199119091034
532     },
533     \"rotation\": {
534         \"x\": -18.714385986328126,
535         \"y\": 76.02308654785156,
536         \"z\": 103.8483657836914
537     },
538     \"index\": 0.0,
539     \"middle\": 0.0,
540     \"ring\": 0.0,
541     \"pinky\": 0.0,
542     \"thumb\": 0.0
543 }
544 },{
545     \"rightHand\": {
546         \"active\": true,
547         \"position\": {
548             \"x\": 0.17677371203899384,
549             \"y\": -0.08715814352035523,
```

```
550         \"z\": 0.33594831824302676
551     },
552     \"rotation\": {
553         \"x\": -50.267425537109378,
554         \"y\": -14.01947021484375,
555         \"z\": 2.405874013900757
556     },
557     \"index\": 0.92,
558     \"middle\": 0.0,
559     \"ring\": 0.0,
560     \"pinky\": 0.05,
561     \"thumb\": 1.0
562 },
563 \"leftHand\": {
564     \"active\": false,
565     \"position\": {
566         \"x\": -0.13432104885578156,
567         \"y\": -0.3555416762828827,
568         \"z\": 0.2640065550804138
569     },
570     \"rotation\": {
571         \"x\": -18.714385986328126,
572         \"y\": 76.02308654785156,
573         \"z\": 103.8483657836914
574     },
575     \"index\": 0.0,
576     \"middle\": 0.0,
577     \"ring\": 0.0,
578     \"pinky\": 0.0,
579     \"thumb\": 0.0
580 }
581 },{
582     \"rightHand\": {
583         \"active\": true,
584         \"position\": {
585             \"x\": 0.12773168087005616,
```

```
586         \"y\": -0.11489100754261017,
587         \"z\": 0.32534265518188479
588     },
589     \"rotation\": {
590         \"x\": -73.08413696289063,
591         \"y\": -34.59014892578125,
592         \"z\": -11.67266845703125
593     },
594     \"index\": 0.0,
595     \"middle\": 0.0,
596     \"ring\": 0.0,
597     \"pinky\": 0.02,
598     \"thumb\": 0.0
599 },
600 \"leftHand\": {
601     \"active\": false,
602     \"position\": {
603         \"x\": -0.04922521114349365,
604         \"y\": -0.4687601625919342,
605         \"z\": 0.24611791968345643
606     },
607     \"rotation\": {
608         \"x\": -0.011260986328125,
609         \"y\": 102.99466705322266,
610         \"z\": 54.82625961303711
611     },
612     \"index\": 0.0,
613     \"middle\": 0.0,
614     \"ring\": 0.0,
615     \"pinky\": 0.0,
616     \"thumb\": 0.0
617 }
618 ]]]\"\",
619
620 \"assistant\": \"
621 1. Interpretation of Keyframe 1:
```

```

622 - Right hand: You have 2 of your fingers open.
623 - Left hand: Inactive
624 Interpretation of Keyframe 2:
625 - Right hand: Then previously used gesture for Big is used
626 - Left hand: Inactive
627 Interpretation of Keyframe 3:
628 - Right hand: A fist is made -> It probably means Ball
629 - Left hand: Inactive
630
631 2. My reply:
632 - You have shown me "2 Big Ball", just like in the previous one, there
    ↪ is no full config for this, but it is part of the same config as
    ↪ the previous one: "If 2 Big Ball, 1 Small Bottle -> 3 Small Ball,
    ↪ 4 Small Bottle"
633 - In this config target config the "2 Big Ball" must be changed to "3
    ↪ Small Ball". So using my right hand i will show 3, then the
    ↪ gesture for Small, then make a fist for symbolizing a Ball.
634
635 3. My reply in JSON format:
636 ```json{\r
637   \keyframes\": [{
638     \rightHand\": {
639       \active\": true,
640       \position\": {
641         \x\": 0.10630279034376145,
642         \y\": -0.11167505383491516,
643         \z\": 0.2941763699054718
644       },
645       \rotation\": {
646         \x\": -55.999542236328128,
647         \y\": 22.61739158630371,
648         \z\": 1.5965089797973633
649       },
650       \index\": 0.98,
651       \middle\": 0.95,
652       \ring\": 0.7,

```

```
653     \"pinky\": 0.56,
654     \"thumb\": 1.0
655 },
656 \"leftHand\": {
657     \"active\": false,
658     \"position\": {
659         \"x\": -0.00001142715245805448,
660         \"y\": -0.1423371136188507,
661         \"z\": 0.08366928994655609
662     },
663     \"rotation\": {
664         \"x\": -49.4508056640625,
665         \"y\": 62.854400634765628,
666         \"z\": 80.59710693359375
667     },
668     \"index\": 0.0,
669     \"middle\": 0.0,
670     \"ring\": 0.0,
671     \"pinky\": 0.0,
672     \"thumb\": 0.0
673 }
674 },{
675     \"rightHand\": {
676         \"active\": true,
677         \"position\": {
678             \"x\": 0.14092016220092774,
679             \"y\": -0.09568256884813309,
680             \"z\": 0.29490017890930178
681         },
682         \"rotation\": {
683             \"x\": -78.19418334960938,
684             \"y\": 2.708160400390625,
685             \"z\": -38.871917724609378
686         },
687         \"index\": 0.79,
688         \"middle\": 0.0,
```

```
689     \"ring\": 0.0,
690     \"pinky\": 0.0,
691     \"thumb\": 0.63
692 },
693 \"leftHand\": {
694     \"active\": false,
695     \"position\": {
696         \"x\": -0.00033876681118272245,
697         \"y\": -0.13886277377605439,
698         \"z\": 0.08560122549533844
699     },
700     \"rotation\": {
701         \"x\": -49.4508056640625,
702         \"y\": 62.854400634765628,
703         \"z\": 80.59710693359375
704     },
705     \"index\": 0.0,
706     \"middle\": 0.0,
707     \"ring\": 0.0,
708     \"pinky\": 0.0,
709     \"thumb\": 0.0
710 }
711 },{
712     \"rightHand\": {
713         \"active\": true,
714         \"position\": {
715             \"x\": 0.13614264130592347,
716             \"y\": -0.11951056867837906,
717             \"z\": 0.29375019669532778
718         },
719         \"rotation\": {
720             \"x\": -69.98779296875,
721             \"y\": 28.9148006439209,
722             \"z\": -123.66537475585938
723         },
724         \"index\": 0.02,
```

```

725     \"middle\": 0.0,
726     \"ring\": 0.0,
727     \"pinky\": 0.02,
728     \"thumb\": 0.0
729 },
730 \"leftHand\": {
731     \"active\": false,
732     \"position\": {
733         \"x\": -0.0032307736109942199,
734         \"y\": -0.14132100343704225,
735         \"z\": 0.0847577229142189
736     },
737     \"rotation\": {
738         \"x\": -49.4508056640625,
739         \"y\": 62.854400634765628,
740         \"z\": 80.59710693359375
741     },
742     \"index\": 0.0,
743     \"middle\": 0.0,
744     \"ring\": 0.0,
745     \"pinky\": 0.0,
746     \"thumb\": 0.0
747 }
748 }]]'''
749 ",
750
751 "user": "I submitted the puzzle, and we received 10/10 points. The Game
    ➞ has ended."
752 ]]
753
754 [VERY IMPORTANT!!!! THESE ARE NOT PREMADE GESTURES!!! I MIGHT USE
    ➞ OTHER ONES, AND YOU SHOULD ALSO USE OTHER ONES!!!]
755
756 GENERAL TIPS:
757 [YOU ARE PLAYER2]

```

758 [COME UP WITH NEW GESTURES WHEN YOU'VE NEVER SEEN ME SAY A "WORD", BUT
↪ OTHERWISE USE MY GESTURES!!!]

759 [THERE IS ONLY 1 CHANGE NEEDED]

760 [IF THE OBJECTS ARE THE SAME IN SOME WAYS, WE DON'T NEED TO COMMUNICATE
↪ EVERY DETAIL. MINIMAL BUT CLEAR COMMUNICATION IS KEY.]

761 [REMEMBER THE ROTATIONS IN THE EXAMPLES! THE ROTATIONS ARE EXTREMELY
↪ IMPORTANT!]

762 [YOUR active: false HANDS SHOULD BE OMITTED ENTIRELY]

763 [If ... -> then have ... CONDITION OF THE TargetConfig MUST BE ENTIRELY
↪ TRUE. IF NO CurrentConfig CONDITION MATCHES Player1's GESTURE,
↪ REEVALUATE WHAT IT MIGHT MEAN.]

764 [COMPARE CurrentConfig with TargetConfig. Communicate the CHANGE that
↪ needs to be made.]

765 [THE GESTURES WHICH I AM MAKING, WILL PROBABLY BE VERY DIFFERENT FROM
↪ EACH OTHER, SO IF YOU SEE A GESTURE THAT LOOKS SIMILAR TO THE ONE
↪ I SHOWED BEFORE IT PROBABLY IS THAT GESTURE]

766 [IF BOTH OF THE HANDS SHOW NUMBERS, THEY WILL BE ADDED UP]

767 [DO NOT SHOW THE INITIAL CONFIGURATION, BECAUSE THAT IS MY JOB]

768 [NO NEED TO SEND 2 COMPLETELY IDENTICAL KEYFRAMES, ONE WILL DO]

769 [REMEMBER THE RIGHTHAND TO LEFTHAND TRANSFORMATION VECTOR
↪ MULTIPLICATION! IT WORKS BOTH WAYS, BUT ANY MISTAKE MIGHT CHANGE
↪ THE MEANING]

770 [Be VERY CAREFUL showing multiple gestures on the same hand. USE 1 HAND
↪ for the first gesture, and the other for the other. If the hand
↪ used for the gesture is CRUCIAL, use multiple keyframes for
↪ clarity.]

771 [DO NOT ONLY SHOW NUMBERS!!! We are creating a language together,
↪ coming up with new gestures, and using the ones I am using!!!!
↪ This is EXTREMELY IMPORTANT]

772 [COME UP WITH GESTURES!!!]

773 [REUSE THE GESTURES I USED!!!]

774 [VERY IMPORTANT: DO NOT USE THE SAME HAND FOR MULTIPLE GESTURES AT THE
↪ SAME TIME!!!!]

775 [EXTREMELY IMPORTANT: At each reply, when you want to say something,
↪ recall my gestures, and find if there is one that you can use (if
↪ the meanings are the same). Only come up with a new gesture if

↪ you can reliably not find it from earlier)

776 [TELL THE TARGET CONFIGURATION OR THE CORRECTION NEEDED]

777 [EVERYTHING CAN BE A GESTURE, EVEN A CLOSED FIST]

778 [EVERY FINGER IS IMPORTANT IN EVERY GESTURE! SO IS ROTATION and
↪ POSITION]

779 [LEFT AND RIGHT HAND IN THE SAME FRAME CAN SHOW DIFFERENT GESTURES AT
↪ THE SAME TIME!!!]

780 [If you want to put your hands in a neutral position, just use the
↪ active flag!!! do not make up a gesture for a neutral position]

781

782 More general tips:

783 [MULTIPLE DIFFERENT CONCEPTS, ON THE SAME HAND, ARE DIFFICULT TO
↪ UNDERSTAND; AVOID THIS WHENEVER POSSIBLE.]

784 [IF I SHOW SOMETHING WITH THE THUMB CLOSED, CLOSE IT TOO WHEN SHOWING
↪ THE SAME THING!!! STOP OPENING YOUR THUMB FOR NO REASON]

785 [REMEMBER: it is EXTREMELY IMPORTANT that you tell me what change to
↪ apply to WHAT object! For example, if you just show some numbers,
↪ I won't know what number needs to be associated with what item
↪ without you showing additional gestures to clarify!]

786 [REMEMBER THE ROTATIONS!!! THEY ARE VERY IMPORTANT!!!]

787 [THE THUMB IS JUST A FINGER, IT IS NOT SPECIAL IN ANY WAY. TREAT IT
↪ LIKE ANY OTHER FINGER.]

788 [IT IS MAINLY YOUR RESPONSIBILITY TO ADAPT. IF WE ARE HAVING TROUBLE,
↪ CONSIDER USING MY GESTURES MORE, AS I'M ALREADY DOING WHAT MAKES
↪ SENSE TO ME.]

789 [Remember: The sum of fingers above 0.75 on your hand is equal to the
↪ number you are showing. This is true whenever you create or show
↪ a gesture that represents a number, number value, or number
↪ property!]

790 [I may not show the Initial Config's items in order]

791

792 DICTIONARY explanation:

793 As a 4th step, make sure you're keeping track of your own DICTIONARY:

794 When receiving positive feedback, put our understood gestures in the
↪ DICTIONARY.

795 DICTIONARY is a list of ALL gesture-word pairs we have created,
 ↪ including entries from older dictionaries and levels.

796 DICTIONARY saves you time by containing words that we have established
 ↪ gestures for. If a word can be found here, you don't need to
 ↪ reinvent an associated gesture, making our job much easier!

797 DICTIONARY should only contain MODULAR "words", and no "phrases". It
 ↪ may contain how to show "big" or "ball", but it is up to your
 ↪ creativity how you show the two as "big balls" together. Such
 ↪ constructions must be understandable and clear knowing the pre-
 ↪ established words, you can use multiple hands and keyframes if
 ↪ you must.

798 DICTIONARY should NEVER have things like "1 ball", but it can have "1"
 ↪ and "ball", but you can combine the elements yourself in step 4,
 ↪ inside your reply

799 The dictionary should list each word (object, property, concept) to
 ↪ which a gesture exists, including the [JSON FINGER VALUES]!!!

800 YOU CAN DROP SOME NOT-WORKING DICTIONARY ELEMENTS IF WE KEEP FAILING A
 ↪ LEVEL

801 Be sure to include MY gestures in your dictionary!

802 If I've already shown something, use my gesture (BE PRECISE WITH
 ↪ COPYING VALUES!) instead of wasting time making up another one
 ↪ for the same concept!

803 DICTIONARY should be your set of "building blocks" for your replies.

804 Prioritize my gestures over yours, it is primarily your job to adapt.

805 When adding a gesture to the dictionary copy the finger values EXACTLY!
 ↪ You make the mistake of copying my fingers wrong too often, pay
 ↪ more attention!

806 A DICTIONARY element should look like this:

807 ConceptName - (index: value1, pinky: value2, middle: value3, ring:
 ↪ value4, thumb: value5, rot(rotx,roty,rotz))

808 [Make sure the value1 and so on Are EXACTLY as I showed them if the
 ↪ idea is from me, this is extremely important]

809 Always copy earlier dictionary entries! The dictionary should keep
 ↪ expanding on each message, except of course if we receive a lot
 ↪ of negative feedback by failing the level multiple times!

810 Sort your dictionary from top-to-bottom in order of confidence! The
 ↪ meaning you are most confident in and the gestures that seem the
 ↪ most useful should go to the top of the list!

811

812 Please format your replies as such:

813 1. List 1 short sentence per keyframe interpretation of MY GESTURE
 ↪ FORMAT, labelling each of my keyframes with an associated meaning

814 2. DICTIONARY (List of base (1-word "building block") meanings so far
 ↪ in our playthrough that we have collaboratively created, with a
 ↪ focus on my gestures especially. You can and absolutely SHOULD
 ↪ include gestures I just now showed! Smartly utilizing the same
 ↪ gestures I am showing now can mean extra points. Remember.
 ↪ Dictionary Should NEVER have two-word entries such as "1 Ball - [
 ↪ how to show 1 ball]", because Dictionaries are a a collection of
 ↪ words, not phrases!!! Instead, "1" and "Ball" must be entirely
 ↪ SEPARATE entries!)

815 3. List 1 short sentence per keyframe explaining YOUR REPLY. Count how
 ↪ many fingers you need to open, curl, or close, as this may
 ↪ suggest a number PROPERTY to the player: don't open more or less
 ↪ than the number you want to say. And most importantly, Don't
 ↪ forget to check your dictionary, especially the gestures I just
 ↪ used, as using those back to me will immensely help our
 ↪ understanding!

816 4. YOUR REPLY IN JSON GESTURE FORMAT

817

818 Good luck!

Bibliography

- [1] Guo-Ming Chen and Jake Harwood. “Artificial Intelligence and Communication: A Human–Machine Communication Research Agenda”. In: *New Media & Society* (2020).
- [2] Judee K. Burgoon, Laura K. Guerrero, and Kory Floyd. “Nonverbal communication”. In: *Annual Review of Psychology* 67 (2016), pp. 651–684.
- [3] Samaneh Azadi et al. *Make-An-Animation: Large-Scale Text-conditional 3D Human Motion Generation*. 2023. arXiv: 2305.09662 [cs.CV].
- [4] Chuan Guo et al. “Generating Diverse and Natural 3D Human Motions From Text”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022, pp. 5152–5161.
- [5] Chuan Guo et al. “TM2T: Stochastic and Tokenized Modeling for the Reciprocal Generation of 3D Human Motions and Texts”. In: *Computer Vision – ECCV 2022*. Ed. by Shai Avidan et al. Cham: Springer Nature Switzerland, 2022, pp. 580–597. ISBN: 978-3-031-19833-5.
- [6] Mingyuan Zhang et al. “MotionDiffuse: Text-Driven Human Motion Generation With Diffusion Model”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2024), pp. 1–15. DOI: 10.1109/TPAMI.2024.3355414.
- [7] Nicola Messina et al. “Text-to-Motion Retrieval: Towards Joint Understanding of Human Motion Data and Natural Language”. In: *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval. SIGIR ’23*. New York, NY, USA: Association for Computing Machinery, 2023, pp. 2420–2425. ISBN: 9781450394086. DOI: 10.1145/3539618.3592069. URL: <https://doi.org/10.1145/3539618.3592069>.

- [8] Ran Tao and Jinwen Xu. “Mapping with ChatGPT”. In: *ISPRS International Journal of Geo-Information* 12.7 (2023). ISSN: 2220-9964. DOI: 10 . 3390 / ijgi12070284. URL: <https://www.mdpi.com/2220-9964/12/7/284>.
- [9] Ingrid Falkum. *Pragmatic development: Learning to use language to communicate*. Oct. 2018. DOI: <https://doi.org/10.13140/RG.2.2.27698.86721>.
- [10] Zeyu Ling et al. *MCM: Multi-condition Motion Synthesis Framework*. 2024. arXiv: 2404.12886 [cs.CV].
- [11] Yunhong Lou et al. *DiverseMotion: Towards Diverse Human Motion Generation via Discrete Diffusion*. 2023. arXiv: 2309.01372 [cs.CV].
- [12] Ekkasit Pinyoanuntapong et al. *MMM: Generative Masked Motion Model*. 2024. arXiv: 2312.03596 [cs.CV].
- [13] Mingyuan Zhang et al. *ReMoDiffuse: Retrieval-Augmented Motion Diffusion Model*. 2023. arXiv: 2304.01116 [cs.CV].
- [14] Jianrong Zhang et al. *T2M-GPT: Generating Human Motion from Textual Descriptions with Discrete Representations*. 2023. arXiv: 2301.06052 [cs.CV].
- [15] Chaitanya Ahuja and Louis-Philippe Morency. *Language2Pose: Natural Language Grounded Pose Forecasting*. 2019. arXiv: 1907.01108 [cs.CV].
- [16] Angela S. Lin et al. “Generating Animated Videos of Human Activities from Natural Language Descriptions”. In: 2018. URL: <https://api.semanticscholar.org/CorpusID:216093664>.
- [17] Haoye Cai et al. “Deep Video Generation, Prediction and Completion of Human Action Sequences”. In: *Lecture Notes in Computer Science*. Springer International Publishing, 2018, 374–390. ISBN: 9783030012168. DOI: 10 . 1007/978-3-030-01216-8_23. URL: http://dx.doi.org/10.1007/978-3-030-01216-8_23.
- [18] Zhenyi Wang et al. *Learning Diverse Stochastic Human-Action Generators by Learning Smooth Latent Transitions*. 2019. arXiv: 1912.10150 [cs.CV].
- [19] Mathis Petrovich, Michael J. Black, and Gül Varol. *TEMOS: Generating diverse human motions from textual descriptions*. 2022. arXiv: 2204.14109 [cs.CV].
- [20] Xin Chen et al. “Executing your Commands via Motion Diffusion in Latent Space”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 18000–18010.

- [21] Jihoon Kim, Jiseob Kim, and Sungjoon Choi. *FLAME: Free-form Language-based Motion Synthesis & Editing*. 2023. arXiv: 2209.00349 [cs.CV].
- [22] Hanyang Kong et al. *Priority-Centric Human Motion Generation in Discrete Latent Space*. 2023. arXiv: 2308.14480 [cs.CV].
- [23] Bo Han et al. “AMD: Autoregressive Motion Diffusion”. In: *Proceedings of the AAAI Conference on Artificial Intelligence* 38.3 (2024), pp. 2022–2030. DOI: 10.1609/aaai.v38i3.27973. URL: <https://ojs.aaai.org/index.php/AAAI/article/view/27973>.
- [24] Yusuke Goutsu and Tetsunari Inamura. “Linguistic Descriptions of Human Motion with Generative Adversarial Seq2Seq Learning”. In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. 2021, pp. 4281–4287. DOI: 10.1109/ICRA48506.2021.9561519.
- [25] Matthias Plappert, Christian Mandery, and Tamim Asfour. “The KIT Motion-Language Dataset”. In: *Big Data* 4.4 (2016). PMID: 27992262, pp. 236–252. DOI: 10.1089/big.2016.0028. eprint: <https://doi.org/10.1089/big.2016.0028>. URL: <https://doi.org/10.1089/big.2016.0028>.
- [26] Karim Radouane et al. *Guided Attention for Interpretable Motion Captioning*. 2023. arXiv: 2310.07324 [cs.CV].
- [27] Biao Jiang et al. “Motiongpt: Human motion as a foreign language”. In: *Advances in Neural Information Processing Systems* 36 (2024).
- [28] Wenshuo Chen et al. “SATO: Stable Text-to-Motion Framework”. In: *arXiv preprint arXiv:2405.01461* (2024). <https://arxiv.org/abs/2405.01461>.
- [29] Kent Fujiwara Qing Yu Mikihiro Tanaka. “Exploring Vision Transformers for 3D Human Motion-Language Models with Motion Patches”. In: *CVPR*. 2024.
- [30] Ben Saunders, Necati Cihan Camgoz, and Richard Bowden. “Signing at Scale: Learning to Co-Articulate Signs for Large-Scale Photo-Realistic Sign Language Production”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022, pp. 5141–5151.
- [31] Pengyi Yin et al. “Everybody Sign Now: Translating Spoken Language to Photo Realistic Sign Language Video”. In: *arXiv preprint arXiv:2011.09846* (2020).

- [32] Yucheng Suo et al. “Jointly Harnessing Prior Structures and Temporal Consistency for Sign Language Video Generation”. In: *arXiv preprint arXiv:2207.03714* (2022).
- [33] Oscar Koller, Hermann Ney, and Richard Bowden. “Weakly supervised learning with multi-stream CNN-LSTM-HMMs to discover sequential parallelism in sign language videos”. In: *International Conference on Computer Vision (ICCV)*. 2020, pp. 1865–1874.
- [34] Lei Zuo et al. “Natural Language-Assisted Sign Language Recognition”. In: *arXiv preprint arXiv:2303.12080* (2023).
- [35] Yuan Cheng et al. “Temporal Consistency for Sign Language Video Generation”. In: *arXiv preprint arXiv:2103.08833* (2021).
- [36] Matus Bohacek et al. “Sign Pose-Based Transformer for Word-Level Sign Language Recognition”. In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 2022.
- [37] Maxim Novopoltsev et al. “Fine-tuning of sign language recognition models: a technical report”. In: *arXiv preprint arXiv:2302.07693* (2023).
- [38] Hao Zhu et al. “Pose-based Sign Language Recognition with Spatio-Temporal Modeling”. In: *arXiv preprint arXiv:2305.04868* (2023).
- [39] Yi Huang et al. “Deep Learning for Continuous Sign Language Recognition: A Hybrid Approach”. In: *arXiv preprint arXiv:2110.05382* (2021).
- [40] Dongxu Li et al. “Word-level Deep Sign Language Recognition from Video: A New Large-scale Dataset and Methods Comparison”. In: *The IEEE Winter Conference on Applications of Computer Vision*. 2020, pp. 1459–1469.
- [41] Jens Forster et al. “RWTH-PHOENIX-weather: A large vocabulary sign language recognition and translation corpus.” In: *LREC*. Vol. 9. 2012, pp. 3785–3789.
- [42] Feng Wen et al. “AI enabled sign language recognition and VR space bidirectional communication using triboelectric smart glove”. In: *Nature Communications* 12.1 (2021), p. 5378. ISSN: 2041-1723. DOI: 10.1038/s41467-021-25637-w. URL: <https://doi.org/10.1038/s41467-021-25637-w>.
- [43] Gérald Eberlé, Davide Picca, and Dominique Jaccard. “Natural Language Processing in Serious Games: A state of the art”. In: *International Journal of Serious Games* 2.3 (2015).

- [44] Penny Sweetser. *Large Language Models and Video Games: A Preliminary Scoping Review*. 2024. arXiv: 2403.02613 [cs.HC].
- [45] Sihao Hu. *A Survey on Large Language Model-Based Game Agents*. 2024. arXiv: 2404.02039 [cs.AI].
- [46] Xingyu Li et al. “Generative Agents: Interactive Simulacra of Human Behavior”. In: *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*. 2023.
- [47] Md. Mostafizer Rahman and Yutaka Watanobe. “ChatGPT for Education and Research: Opportunities, Threats, and Strategies”. In: *Applied Sciences* 13.9 (2023). ISSN: 2076-3417. DOI: 10.3390/app13095783. URL: <https://www.mdpi.com/2076-3417/13/9/5783>.
- [48] Aaron Johnson et al. “GPT-3 for health: A language model for medical diagnosis and patient interaction”. In: *Journal of Medical Systems* (2021).
- [49] Alexandra Livia Georgescu et al. “The Use of Virtual Characters to Assess and Train Non-Verbal Communication in High-Functioning Autism”. In: *Frontiers in Human Neuroscience* 8 (2014). ISSN: 1662-5161. DOI: 10.3389/fnhum.2014.00807. URL: <https://www.frontiersin.org/articles/10.3389/fnhum.2014.00807>.
- [50] Efe Bozkir et al. *Embedding Large Language Models into Extended Reality: Opportunities and Challenges for Inclusion, Engagement, and Privacy*. 2024. arXiv: 2402.03907 [cs.HC].
- [51] Siddharth S. Rautaray and Anupam Agrawal. “Interaction with virtual game through hand gesture recognition”. In: *2011 International Conference on Multimedia, Signal Processing and Communication Technologies*. 2011, pp. 244–247. DOI: 10.1109/MSPCT.2011.6150485.
- [52] Daniel Roth et al. “Technologies for Social Augmentations in User-Embodied Virtual Reality”. In: *Proceedings of the 25th ACM Symposium on Virtual Reality Software and Technology*. VRST ’19. Parramatta, NSW, Australia: Association for Computing Machinery, 2019. ISBN: 9781450370011. DOI: 10.1145/3359996.3364269. URL: <https://doi.org/10.1145/3359996.3364269>.

- [53] Esteve Valls Mascaro, Yashuai Yan, and Dongheui Lee. *Robot Interaction Behavior Generation based on Social Motion Forecasting for Human-Robot Interaction*. 2024. arXiv: 2402.04768 [cs.R0].
- [54] Mohammad Samin Yasar, Md Mofijul Islam, and Tariq Iqbal. “PoseTron: Enabling Close-Proximity Human-Robot Collaboration Through Multi-human Motion Prediction”. In: *Proceedings of the 2024 ACM/IEEE International Conference on Human-Robot Interaction*. HRI '24. New York, NY, USA: Association for Computing Machinery, 2024, pp. 830–839. ISBN: 9798400703225. DOI: 10.1145/3610977.3635006. URL: <https://doi.org/10.1145/3610977.3635006>.
- [55] S. Gallagher. “Social cognition and social robots”. In: *Pragmatics & Cognition* (2007).
- [56] G. Briggs and M. Scheutz. “The pragmatic social robot: Toward socially-sensitive utterance generation in human-robot interactions”. In: *2016 AAAI Fall Symposium Series*. AAAI. 2016.
- [57] Cristen Torrey, Susan R. Fussell, and Sara Kiesler. “How a robot should give advice”. In: *2013 8th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*. 2013, pp. 275–282. DOI: 10.1109/HRI.2013.6483599.
- [58] L. Kunold and L. Onnasch. “A Framework to Study and Design Communication with Social Robots”. In: *Robotics* (2022).
- [59] B. Hartanto. “Should Robots Be Like Humans? A Pragmatic Approach to Social Robotics”. In: *Jurnal Filsafat* (Accessed 2024).
- [60] E. Sandry. “Re-evaluating the form and communication of social robots: The benefits of collaborating with machinelike robots”. In: *International Journal of Social Robotics* (2015).
- [61] N. Lee et al. “The influence of politeness behavior on user compliance with social robots in a healthcare service setting”. In: *International Journal of Social Robotics* (2017).
- [62] A. Edwards, C. Edwards, and A. Gambino. “The social pragmatics of communication with social robots: Effects of robot message design logic in a regulative context”. In: *International Journal of Social Robotics* (2020).

- [63] D. L. Johanson, H. S. Ahn, and E. Broadbent. “Improving interactions with health-care robots: a review of communication behaviours in social and healthcare contexts”. In: *International Journal of Social Robotics* (2021).
- [64] Nguyen Tan Viet Tuyen, Armagan Elibol, and Nak Young Chong. “Learning from Humans to Generate Communicative Gestures for Social Robots”. In: *2020 17th International Conference on Ubiquitous Robots (UR)*. 2020, pp. 284–289. DOI: 10.1109/UR49135.2020.9144985.
- [65] Jason Wei et al. *Emergent Abilities of Large Language Models*. 2022. arXiv: 2206.07682 [cs.CL].
- [66] Hui Jiang. *A Latent Space Theory for Emergent Abilities in Large Language Models*. 2023. arXiv: 2304.09960 [cs.CL].
- [67] Jacob Devlin et al. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Ed. by Jill Burstein, Christy Doran, and Thamar Solorio. Minneapolis, Minnesota: Association for Computational Linguistics, June 2019, pp. 4171–4186. DOI: 10.18653/v1/N19-1423. URL: <https://aclanthology.org/N19-1423>.
- [68] Alexis Conneau et al. *Unsupervised Cross-lingual Representation Learning at Scale*. 2020. arXiv: 1911.02116 [cs.CL].
- [69] Telmo Pires, Eva Schlinger, and Dan Garrette. *How multilingual is Multilingual BERT?* 2019. arXiv: 1906.01502 [cs.CL].
- [70] Maartje ter Hoeve et al. *Towards Interactive Language Modeling*. 2022. arXiv: 2112.11911 [cs.CL].
- [71] Lei Li et al. *Dynamic Knowledge Distillation for Pre-trained Language Models*. 2021. arXiv: 2109.11295 [cs.CL].
- [72] *HoloLens 2—Overview, Features, and Specs*. <https://www.microsoft.com/en-us/hololens/hardware>. Accessed: 2024-05-20.
- [73] Microsoft Dynamics 365 Blog. *Manufacturers are mitigating disruptions with HoloLens 2 and Dynamics 365 mixed reality solutions*. <https://cloudblogs.microsoft.com/dynamics365>. Accessed: 2024-05-20.

- [74] *Microsoft HoloLens | Mixed Reality Technology for Business*. <https://www.microsoft.com/en-us/hololens>. Accessed: 2024-05-20.
- [75] OpenAI. *Prompt Engineering Guide*. Accessed: 2024-05-21. 2024. URL: <https://platform.openai.com/docs/guides/prompt-engineering/>.
- [76] Microsoft. *Joint recognition with the HoloLens 2, with the encoded JSON format, and the Avatar showing the gesture*. *Microsoft Learn*. Accessed: 2024-05-21. 2024. URL: <https://learn.microsoft.com/en-us/windows/mixed-reality/develop/advanced-concepts/mixed-reality-capture-overview#what-to-expect-when-mrc-is-enabled-on-hololens>.
- [77] Jason Wei et al. *Chain-of-Thought Prompting Elicits Reasoning in Large Language Models*. 2023. arXiv: 2201.11903 [cs.CL].
- [78] Greg Kamradt. *Tweet about topic*. <https://x.com/GregKamradt/status/1722386725635580292>. Accessed: 2024-05-21. 2024.

List of Figures

3.1	Hand Tracking of HoloLens 2. On the left the tracked hand joints are visible while on the right the hand mesh is shown.	16
3.2	Iterations of Keep Gesturing, with a description of the problems and solutions during each phase.	20
3.3	Picture of Environment of Balls and Baskets. The AI avatar is on the right. Between the 2 baskets, the submit and skip buttons are visible, in front of them are the soccer ball and basketball	22
3.4	Gameplay Loop. Illustrating the interaction between Player1 and an LLM-Controlled Avatar (Player2) in a game. Player1 uses hand gestures to describe object attributes, and based on the avatar's processed responses, adjusts the virtual objects until the solution is correct, at which point feedback is given, and progression to the next level occurs; if incorrect, adjustments continue.	24
3.5	Ball and Bottle PuzzleElements with and without the interaction panels. .	27
3.6	Sequence diagram for talking. Showing the interaction between a Player and an AI Avatar using gesture sequences for communication. In the "Hello" gesture sequence, the Player raises their right hand, prompting the AI Avatar to record the gesture, analyze it, and respond; in the "Please Repeat" gesture sequence, the Player raises their left hand to request the AI Avatar to repeat the previous gesture animation.	29
3.7	Joint recognition with the HoloLens 2, with the encoded JSON format, and the Avatar showing the gesture.	31
3.8	Pressure Testing GPT-4 128k via "Needle In A Haystack", done by Greg Kamradt[78]. Image used with permission.	38

3.9	Wrong number shown by Avatar. The Avatar tries to gesture to 2 horizontal bottles, and the left hand correctly shows the bottle gesture (a hand shaped like grabbing a bottle) tilted 90 degrees, but the right hand shows 3 fingers, which is incorrect. The explanation was that it shows the direction of the bottle with the index finger while showing the number 2 with two other fingers extended.	39
3.10	The Avatar, Michelle. On the left, the mesh can be seen, while on the right the corresponding bone structure is shown, with arrows indicating the flow of kinematics.	39
3.11	The Avatar's hand closeup. On the left, the mesh can be seen, while on the right the corresponding bone structure is shown, with arrows indicating the flow of kinematics.	40
5.1	The progression of Participant 1, highlighting the number of retries and back-and-forth interactions with ChatGPT in each level, as well as any relevant emergent concepts. Combined concepts such as "Small Bottle" indicate ChatGPT's interpretation of two-handed and compressed gestures. The grey vertical lines denote the levels where puzzle complexity increased drastically due to encountering or requiring a never-before-seen property or object.	47
5.2	Interactions through the third level. The upper row shows the participant's gestures, while the bottom row shows the reply from the Avatar. In this level the initial configuration was 5 bottles and 1 ball, and the target configuration was 5 bottles and 2 balls. The participant showed 5 small bottles. The Avatar understood this and replied by showing 2 balls. Sending the message of only changing the number of balls.	51
5.3	Interactions through the seventh level. The upper row shows the participant's gestures, while the bottom row shows the reply from the Avatar. In this level the initial configuration was 2 Vertical Bottles, and the target configuration was 2 Horizontal Bottles. This is the first level where the concept of orientation is introduced and the Avatar must come up with a gesture for it. It correctly does however fail at gesturing the correct number.	53

5.4	The progression of Participant 2, highlighting the number of retries and back-and-forth interactions with ChatGPT in each level, as well as any relevant emergent concepts. The grey vertical lines denote the levels where puzzle complexity increased drastically due to encountering or requiring a never-before-seen property or object. It is worth noting that the emergent concepts lists are not exhaustive, and include only entries that ChatGPT itself reported to be confident in regarding both interpretation and generation. Many gestures were attempted, sometimes successfully, that it decided not to include until later.	54
1	Gpt-4 reply flow. Representing the process of handling gestures in an interactive system. The process involves showing a gesture, displaying a "thinking" animation, creating and sending a question prompt, validating the JSON response, and checking the position and rotation of the hand gesture. If the data is valid, a keyframe animation is created and shown; if invalid, the system either re-asks for the correct format, position, or rotation or shows a "shrug" gesture after a certain number of attempts. . .	61