# Towards natural language understanding for intuitive interactions in XR using large language models

Kevin Skyba*
University of Applied Sciences Emden/Leer

Thies Pfeiffer†
University of Applied Sciences Emden/Leer

## Abstract

This paper presents a voice assistance system for extended reality (XR) applications based on large language models (LLMs). The aim is to create an intuitive and natural interface between users and virtual environments that goes beyond traditional, predefined voice commands. An architecture is presented that integrates LLMs as embodied agents in XR environments and utilizes their natural language understanding and contextual reasoning capabilities. The system interprets complex spatial instructions and translates them into concrete actions in the virtual environment. The performance of the system is evaluated in XR scenarios including object manipulation, navigation and complex spatial transformations. The results show promising performance in simple tasks, but also reveal challenges in processing complex spatial concepts. This work contributes to the improvement of user interaction in XR environments and opens up new possibilities for the integration of LLMs in XR environments.

**Index Terms:** Voice-based interaction, Mixed reality, Large language models

## 1 Introduction

Recent technical advances in the field of Large Language Models (LLMs), such as OpenAI's GPT models, have the potential to improve the way in which systems understand and process language by showing emergent behaviors such as reasoning, solving mathematical issues, or interacting with the world and with humans [7, 6].

In the context of XR applications, LLMs may enable more natural and intuitive user interactions. Instead of relying on a restricted, pre-defined set of commands to express their intentions, which requires users to learn how to use the system a priori, users can communicate their intentions in a natural way in their own words. Moreover, LLMs show emergent behaviors, which allow them to interpret, draw conclusions, and even plan independently within certain limits [7, 34, 17]. This flexibility may increase accessibility of XR applications for casual users, users with little to none domain specific knowledge (e.g. in training applications), or people with cognitive impairments. LLMs can help to correctly interpret the intended interaction despite unclear voice commands by adding other information as context [31].

The main contribution of this paper is an approach which defines an interface between the user, the virtual environment (VE), and the LLM. The architecture of an embodied agent is able to reason logically about input from defined interfaces and perform actions on the users behalf within the virtual environment (see Figure 1). A first study demonstrates the performance of the implemented prototype using user generated commands collected in an online survey.

The presented approach may be applied in a variety of XR applications. Examples are games, in which players navigate using voice commands and interact with virtual objects, or training applications, in which users are guided through complex scenes and can explore certain tasks using voice commands. In particular for training and education, this technology may enhance existing approaches by allowing users in early learning stages, in which the vocabulary of the domain is not yet developed, to express themselves in their own words and learn about the domain specific vocabulary on-the-job by the assistant's feedback. In addition, the system can also be used in rehabilitation or remote control of devices, where it can enable users with limited mobility to perform tasks in the virtual environment using voice commands. This means that the voice assistant can help to improve the accessibility and user-friendliness of XR applications and enable more people to use them.

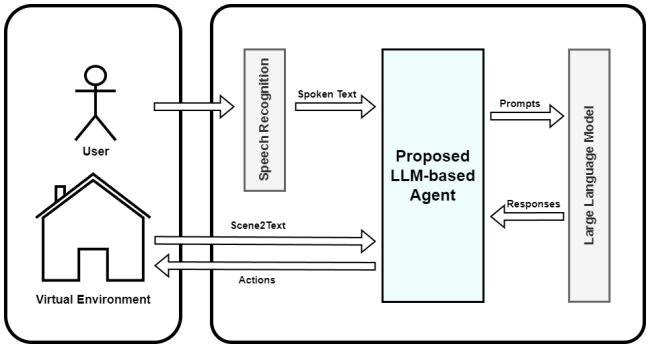


Figure 1: High-level system architecture of the proposed assistance system.

## 2 Related Work

This paper builds on several areas of research dealing with language interaction in virtual environments, the integration of LLMs into interactive systems, and the reasoning capabilities of AI models. Before going into details, it is important to understand the different types of interaction in virtual environments.

In XR applications, there are three main types of interaction: selection, manipulation, and locomotion. While an XR application usually requires all three, the approach considered here, for a starter, focuses primarily on locomotion. Voice-based locomotion as co-existing alternative to conventional interaction techniques (e.g. controller-based) may be particularly useful in situations where the user's hands are otherwise occupied, such as when carrying virtual objects or performing complex manual tasks.

In the following we thus focus on locomotion, however, the presented work is part of a larger study which also addressed selection and manipulation tasks.

### 2.1 Voice-Based-Locomotion

Using language for control and interaction in VEs has been subject of research for a long time. Early work focused mainly on predefined commands or restricted vocabularies [12, 14, 4].

Hombeck et al. [14] did a comprehensive study on three different speech-based locomotion techniques. All of them were triggered by

*e-mail: kskyba@acm.org
†e-mail: thies.pfeiffer@hs-emden-leer.de

| | Level 1<br>Point & Teleport | Level 2<br>Predefined Voice Commands | Level 3<br>LLM |
|---|---|---|---|
| **Physical Input** | User Controlled | Optional | Optional |
| **Obstacle Avoidance** | User Controlled | User Controlled | Semi-Automatic |
| **Task Decomposition** | User Controlled | User Controlled | Semi-Automatic |
| **Contextual Understanding** | None | User Controlled | Semi-Automatic |
| **Adaption to User's Language** | None | User Controlled | Automatic |
| **Cognitive Assistance (Using Memory)** | None | User Controlled | Automatic |
| **Linguistic Repairs & Error Correction** | None | User Controlled | Automatic |

Figure 2: Compared to direct, controller-based interaction, LLMs may support users on multiple levels of interaction.

specific keywords. Their results show that voice-based locomotion techniques may compete with conventional controller-based methods in terms of ease of use and efficiency.

Beyond monomodal approaches, a multimodal interface may provide additional benefits, as has been shown by Sin et al. [30], who investigated the combination of speech recognition and hand tracking in virtual reality. They proposed a hybrid approach that combines speech recognition with gesture recognition to increase the accuracy of identifying user intentions.

### 2.2 Embedding LLMs Into Virtual Environments

The integration of LLMs in VEs is a relatively new field of research that has gained in importance in the past year. Recently, Biggie et al. [3] described NavCon, an approach that utilizes modern computer vision, classical robotic planning algorithms, and the reasoning capabilities of LLMs to perform zero-shot navigation using natural language in different environments. They showed how to combine the strengths of LLMs in natural language and context processing for orientation and decision making in VEs.

Huang et al. [17] combined two LLM chains in an innovative way to control robots in a simulated environment. The first chain is responsible for interpreting human instructions, while the second chain acts as an "embodied agent" and plans as well as executes the actual actions. Their approach show parallels to the concept of an LLM-based embodied agent in VEs presented in the work at hand.

A fundamental contribution to the integration of LLMs in VEs has further been made by Huang et al. [16]. They investigated whether the world knowledge contained in LLMs can be used to act in interactive environments. Using a VirtualHome environment, they showed that LLMs, such as GPT-3 and Codex, are capable of converting high-level natural language instructions into executable action sequences without the need for additional training. This work demonstrates the potential of LLMs for planning, decision making, and acting in VEs.

VOYAGER by Wang et al. [32] is an autonomously acting and planning Minecraft agent that uses LLMs to make decisions. The agent can solve complex tasks in the Minecraft world by translating natural language instructions into concrete actions. This work demonstrates the potential of LLMs for autonomous action in VEs.

Despite the promising progress enabled by LLMs, there are still numerous challenges in integrating LLMs into interactive systems. Mirchandani et al. [22] discussed the limitations of LLMs as general pattern recognition engines and emphasized the need to integrate domain-specific knowledge for specific applications.

Liu et al. [21] investigated the „lost in the middle" problem of LLMs, where information in the middle of long contexts is processed less effectively. This is particularly relevant for the development of systems with long-term memory, as proposed in this paper.

### 2.3 Reasoning Capabilities of LLMs

The ability of LLMs to reason is a central aspect of their application in complex interaction scenarios. However, it is more of an emergent behavior and not build in from ground up. Different strategies have been found to guide the LLM in the reasoning process. Wei et al. [35], for example, developed the „chain-of-thought" (CoT) approach, which encourages LLMs to break problems down and solve them step by step. This technique is effective in improving the performance of LLMs in complex reasoning tasks.

Kojima et al. [18] extended this approach to zero-shot prompting: they showed that LLMs are able to solve complex reasoning tasks without specific training or examples, which significantly increases the flexibility and applicability of these models.

Yao et al. [37] developed the „ReAct" approach, which combines reasoning and acting in LLMs by enabling them to switch between thinking and acting steps, which is particularly relevant for use in interactive environments.

There are numerous papers which benchmark LLMs regarding their capability to „reason" [15, 10, 9]. Spatial reasoning is a current and challenging field of research with many open questions. While LLMs are able to handle complex text-based tasks, they still show weaknesses in the processing and interpretation of three-dimensional spatial information. The available benchmarks for spatial reasoning show that current LLM performances are not yet very satisfactory [19, 23]. While many advanced techniques for improving spatial reasoning exist [36, 29, 2], the approach presented in this paper shows that a basic functionality for spatial reasoning with LLMs is possible. A major advantage of the presented system is its modularity and extensibility. This makes it possible to seamlessly integrate future advances and new techniques in the field of spatial reasoning without having to fundamentally change the basic architecture of the system. Thus, the proposed approach provides a flexible platform that can keep pace with the progress of research and continuously improve spatial reasoning in practical applications.

## 3 Method

The focus of this paper is on the development and evaluation of a voice assistance system for XR applications using LLMs. The presented system aims to create a natural and intuitive interface between users and VEs that allows users to express their action intentions in a more flexible and natural way than classic command and control techniques.

### 3.1 Overview

The main objectives of this paper include several aspects:

An architecture that effectively integrates LLMs into XR environments, taking advantage of the strengths of these models in terms of language understanding and contextual reasoning. For this purpose, a generic approach was developed to enable the LLM to interpret complex spatial instructions, using spatial knowledge, and long- as well as short-term memory in the reasoning process, to finally derive concrete actions that are executed in the VE. To achieve this, an „embodied agent" approach was implemented enabling the LLM to act from the perspective of the user embedded in the VE.

A robust mechanism for representing and processing spatial information, enabling the LLM to effectively „understand" the 3D environment.

A memory system that manages both short-term and long-term information to enable contextualized and consistent interactions over extended periods of time.

An evaluation of the approach in different XR scenarios to analyze its performance and limitations.

### 3.2 Selected Use Cases with Focus on Locomotion

In the target use cases, interactions are initiated by the users using natural language, who are describing what they want to do. For

locomotion, we have identified the following three types of navigation:

1. **Egocentric relative locomotion** describes navigation relative to the user's current position and perspective of view. Example: „Move 2 meters forward."
2. **Point-of-Interest-based locomotion** means navigation to an object or reference point that can be identified by the user. Example: „Move to the window."
3. **Sequential locomotion** refers to navigation by means of a sequence of several movement instructions. Example: "Move to the car and turn by 180 degrees."

It is important to note that all these types of locomotion could be context-dependent, particularly with respect to the dynamic environment and past actions. This context-dependency adds an additional layer of complexity to navigation tasks in XR environments. This includes environmental context, which includes the current state of the virtual environment, as well as the historical context, which references past actions, objects, or locations. Also, task specific context can be provided to the LLM so that user instructions can be interpreted based on the current task or goal of the user.

### 3.3 The Co-Embodied Agent Architecture

A central element of this method is the „co-embodied agent" [13]: an assistive agent is co-embodied with the user in an avatar. This agent provides natural language understanding (using the LLM) and is able to implement user instructions and plans into actions of the avatar, by this mediating the interaction between the user and the virtual world. These actions can range from movements of the avatar itself to manipulation of objects. Technically, the agent receives instructions through structured prompts and can access a predefined library of functions that trigger specific actions in the virtual environment. While the study of Fribourg et al. [13] showed mixed results when two humans were sharing the same avatar, in the present case the user is authoritative and the agent compliant to the user's intentions. Mismatches of intentions and actions, which are a main cause of problems when two beings share one avatar, are thus only an issue when the agent misunderstands the communicated intentions and reparation processes are necessary.

One of the frameworks for implementing embodied agents with LLMs is the so-called ReAct approach by Yao et al [37]. ReAct („Reason" + „Acting") aims to integrate the ability of LLMs to reason, plan, and act in one approach. Approaches such as CoT are only suitable for helping the LLM to reason. Other approaches, such as ChatGPT's approach of giving the LLM access to the Internet via a plugin, only provide the LLM with the ability to act. Yao et al. [37] have shown that both approaches combined lead to better performance when solving tasks.

The agent can be seen as a dynamic prompt chain, where each single message inside the chain is being dynamically manipulated and injected by the agent's runtime (see Figure 3). The static system prompt (1) consists of general instructions, introducing the LLM to its task and the prompting mode that will be introduced in the next step. After that, there is another static prompt called ReAct-Prompt (2), which introduces the Chain-of-Thought (CoT) [35] principle as well as the ReAct method as the framework for running the thought- and act-process. Even though Kojima et al. [18] have showed that CoT can work with zero-shot-prompting, our own experiments showed that one-shot-prompting (4) led to better results. In order to improve the reasoning capabilities of the LLM, self-consistency by Wang et al. [33] can be employed to replace the greedy strategy of running the „think"-loop just once by running it multiple times in parallel and choosing the most common action. In the following, the individual modules contributing to the chain composing the prompt are elaborated in more details.

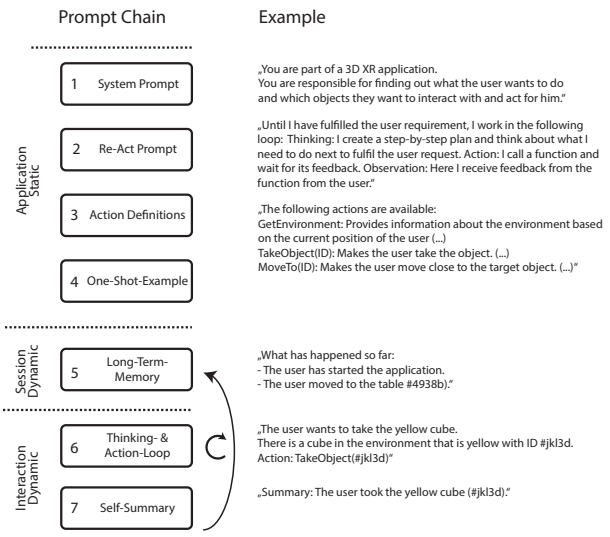


Figure 3: The composition of the prompt chain architecture of the agent. Application static parts are created once, session dynamic parts are updated after each user request, and interaction dynamic parts are updated while processing one request.

### 3.4 Spatial Understanding

The ability to understand spatial relationships and contexts is important in order to integrate LLMs into XR environments. There are various approaches to implement spatial understanding into LLMs. Since LLMs are made for processing text sequences, the challenge lies in bringing three dimensional information from the virtual environment into a textual representation, that LLMs can interpret. It is important to pre-process this information in a way that maintains:

**Scalability** The approach must be able to be used in scenes with potentially many objects and complex spatial relationships. This typically conflicts with the token length of the potential input of a LLM prompt.

**Performance** It must process the information in a reasonable amount of time.

**Context Sensitivity** It is important that there is some kind of filtering so that, ideally, the LLM does only get information that is required for the current task.

**Perspective** Since the LLM is acting from the point of view of the user, the approach has to memorize the current perspective the user is having on the scene.

**Generalizability** The user should not need to learn specific keywords for objects. Instead, the LLM should use its language capabilities and world knowledge to successfully reason about what object the user is referring to.

It is important to note that the component for spatial understanding is interchangeable and in practice will depend on the application domain. The approach chosen here only demonstrates the general feasibility. Further work should replace the component with a more robust and efficient method.

This paper presents an approach that aims to address these challenges in a basic way and effectively bridges the gap between the three-dimensional nature of XR environments and the text-based processing of LLMs. This approach is based on a textually structured transformation of spatial information that enables

LLMs to understand and process complex spatial relationships. Similar approaches have been successfully demonstrated in other fields [11, 39, 1, 8, 20].

For creating a textual spatial representation, a basic scene-to-text technique based on a 3D scenegraph representation is used which first identifies all visible objects from the perspective of the virtual camera, following a two-stage approach: Firstly, the dot product is used to filter out objects that are behind the camera. Raycasting is then used to check whether the remaining objects are actually visible or are obscured by other objects. This method minimizes the number of computationally intensive raycasts and efficiently filters all objects that are visible to the user.

Once the visible objects have been identified, a spatial graph is generated that represents the topological arrangement of the objects in space. Nodes thereby represent the objects whereas edges define the relationships between them. The DBSCAN algorithm is used to cluster all objects. After that, the nodes are connected by edges that represent the spatial relationships between the objects. These relationships are identified by orientation vectors in combination with relative positions and describe positions such as „to the right of" „to the left of", „above", and „below". Depth-based relations such as „behind" or „in front of" are currently not used in order to reduce complexity and thus token length of the textual representation, but are easy to implement likewise, if required. Instead, the distance of each object to the camera is calculated so that the depth information can be reconstructed, if needed. This assumes that users would rather talk about objects in relation to each other in an orthogonal manner, which depends on the application domain.

Each object in the Spatial Graph is annotated with extensive metadata that provides the LLM with contextual and semantic information. This metadata includes a unique identification number (ID), a general name, and a list of attributes that describe additional properties of the object, such as color, material, or other relevant characteristics. Also, invisible virtual objects called „Point-of-Interests" are introduced to be able to mark certain abstract locations, such as rooms or landmarks, without the need of actual 3D geometries. This information is essential to enable the LLM to provide a precise and comprehensible description of the environment. The presented approach requires attribution of metadata to objects, either through manual annotation or automated processes. Methods for the automatic generation of metadata are multimodal models like CLIP (Contrastive Language-Image Pre-training) by OpenAI [26] or object detection algorithms such as YOLO (You Only Look Once) [28]. These approaches could be used to extract semantic information from the rendered image, however, doing so at runtime increases the total system latency significantly. Multimodal large language models such as GPT4-o by OpenAI [25] could be used in the future to allow the model to use the visual data as part of the user prompt. Whether this is a viable alternative to the Spatial Graph remains to be evaluated, as the multimodal-based automatism might be more difficult to tune for specific application domains.

The final step of the Scene-To-Text generation is to export the generated spatial graph into a textual description. This description lists all clusters and the objects they contain, including their IDs, names, attributes, and distances to the camera. In addition, all relations between the objects are described in a structured form in order to provide the LLM with a complete picture of the scene. For this, a rudimentary domain specific language (DSL) has been developed, as depicted in Figure 4. This DSL is explained to the LLM as part of the system prompts as well as being part of the one-shot sample.

That textual scene graph, generated by the time of the user's initial input, is injected into the prompt chain by default as part of the thinking-loop (6), so that there are no redundant think steps regarding any „GetEnvironment()" function call.

For more complex environments, a hierarchical structure in the

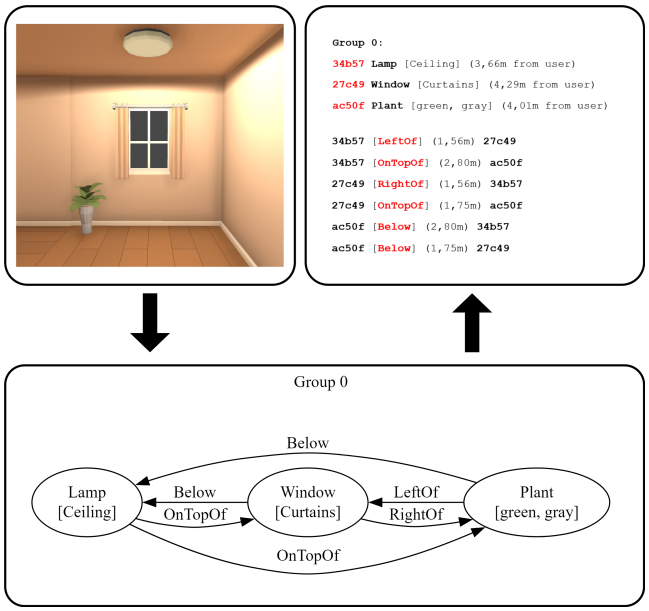


Figure 4: Top Left: Current scene; Bottom: Representation as Spatial Graph after clustering and identification of spatial relations; Top Right: Fragment of the prompt generated for the LLM.

Spatial Graph has been implemented. Objects can be summarized in higher-level groups, which allows scenes to be serialized to text at different levels of detail. This is particularly useful for processing large or complex XR environments.

### 3.5 Memory using „Self-Summary"

To ensure the integrity of the interaction, an approach was implemented that tracks user initiated changes in the virtual environment textually. The LLM not only receives information about the current state via the context, but also about relevant changes since the last interaction. This enables a logical understanding of the scene across time and interactions.

For this, the agent relies on two levels of memory, short-term-memory (STM) and long-term-memory (LTM). The STM, or „working memory", is being implemented solely by using the ReAct framework and keeping all prompts in the chain which are relevant for the current interaction or „think"-loop (6).

In order to keep the chain short, and to only keep important events, that STM is being cleaned up when an interaction finishes. It is then being replaced by a summary which the LLM is producing itself at the end of an interaction. Through this concept of „self-summary" (7), the STM is independently integrated into the LTM by the LLM itself. LLMs are very effective at summarizing things [38]. The instruction „I'll summarize at the end what has happened since the last user request.", combined with an one-shot example in the prompt triggers this summary automatically. The assumption here is that the summarized interaction is sufficient to extract enough information for possible references in future interactions. The thought processes in the STM are no longer relevant for later references. It is only important what the user wanted and what the system made of it.

### 3.6 Linguistic Repairs

To enable the system to resolve ambiguities and uncertainties, „linguistic repairs" allow the system to clarify the content and understand the correct intention through conversational feedback. By giving the agent the ability to call a „GetFeedback()" action, which is implemented to show the user the response of the LLM and to

allow the user to record a response, the agent can use clarifying queries („Do you mean the left cube on the table?"), solve ambiguities („I can see multiple cubes, do you mean the red or the blue one?"), gather context information („Based on the previous interactions, do you want to teleport to the window in the first floor?"), or be fault tolerant towards linguistic inaccuracies („You said you want to take the block. Are you referring to the cube on the table?").

Linguistic repairs allow the system to be more robust, improve the UX, and allow for more human-like behavior, which is characterized by underspecified instructions, which are developed within a negotiation of meaning in an iterative manner. Linguistic repairs have to be implemented using prompt engineering to make sure the LLM is using these feedback mechanisms in a reasonable way.

### 3.7 Multimodality

The system is designed to integrate other modalities in addition to spatial representation. This includes, for example, information from gesture recognition or eye tracking. This multimodal integration enables the LLM to develop a more complete understanding of the user's intention. By integrating gaze and gesture directions, it is possible to resolve unclear instructions that potentially refer to several possible objects, thus making the system more error-tolerant.

Multimodality can be implemented by adding the relevant information to the DSL of the spatial graph. In this case, objects that the user is currently pointing at or looking at could be added with an additional entry such as "The user is currently looking at ac50f". Prompt engineering can be used to instruct the LLM to process this information as required as part of the reasoning process.

However, it should be ensured that different modalities do not contradict each other. In addition, a mechanism should be implemented that evaluates and weights the various input modalities, and only communicates them to the LLM if relevant. When the objects to be considered for selection can be narrowed down from one modality, the spatial graph can be pruned accordingly. Ideally, an additional modality can be used to single out the object for selection and thus reduce the DSL to a single statement in relation to the user intention. In the study prototype, multimodality is not realized.

### 3.8 Implementation

The concept presented was implemented as a modular framework in Unity, a widely used development environment for XR applications. The implementation includes several main components that together form a customizable system for voice-based interaction and locomotion in virtual environments. For the translation of spoken words into text, any speech-to-text-blackbox can be used. In our prototype, we have used Whisper by OpenAI [27] via the public API. The spoken instruction is then passed as text input to the agent. That agent is using a LLM blackbox (GPT-4 by OpenAI [24]) which it feeds a chain of prompts to. That prompt chain is being dynamically created and manipulated by the runtime, depending on the users' actions, the LLMs responses, and the VE.

The system is based on a modular architecture that enables a clear separation of responsibilities while ensuring close integration of the various components. The main components are:

- SpatialObject and SpatialObjectBehaviour
- SpatialGraph and SpatialGraphFactory
- LocomotionAssistant

The 'SpatialObject' class serves as the core component for the representation of spatial objects within the system. Important attributes of this class are

- A unique ID
- The name of the object
- A list of attributes
- Information about tangibility
- Spatial properties such as position, rotation and bounding box

The 'SpatialObjectBehaviour' class acts as a wrapper for the 'SpatialObject' class and extends it with Unity-specific functionalities through the inheritance of 'MonoBehaviour'. This architecture makes it possible to make the 'SpatialObject' instances configurable in Unity scenes while maintaining a certain engine agnosticism.

The 'SpatialGraph' structure serves as a data structure for representing the spatial relationships between 'SpatialObject's in a 3D scene. It consists of a list of clusters, where each cluster in turn contains a list of 'SpatialGraphNode' instances.

The 'SpatialGraphFactory' class is responsible for the creation of 'SpatialGraph' instances. It implements the following steps:

1. Filtering of visible objects by vector calculations and raycasting
2. Cluster formation using the DBSCAN algorithm
3. Creation of the actual 'SpatialGraph' with nodes for each 'SpatialObject' and edges based on spatial relationships

The 'LocomotionAssistant' class coordinates the interaction between the user, the LLM and the virtual environment. Core functions include:

- Management of message lists for the context and state of the interaction
- Implementation of an asynchronous "think" loop (ReAct) for processing user requests
- Communication with the external LLM service (in this case OpenAI's GPT-4)
- Parsing and execution of generated actions

The class implements various prompting techniques such as Chain-of-Thought and ReAct.

GPT-4 was used as the LLM via the OpenAI API. Communication takes place via HTTP requests, with the generated responses being returned in the form of structured text. The prompts for the LLM are generated dynamically to ensure the effectiveness of the Chain-of-Thought and ReAct approach.

Whisper, also from OpenAI, was used for speech recognition. Integration takes place via a simple HTTP-based interface that converts spoken language into written text with relatively short latency (1 to 2 seconds, depending on the input length).

A user request is processed as following:

1. The user enters a voice command, which is converted to text by Whisper.
2. The 'LocomotionAssistant' initiates a „think" loop in which it repeatedly calls the LLM with the current prompt chain.
3. The LLM analyses the request and the environmental context (provided by the 'SpatialGraph') and generates a response that contains either further considerations or an action.
4. If an action is required, it is executed via the 'LocomotionAssistant' component in the virtual environment.
5. After the interaction is completed, a summary is created and integrated into the long-term memory.

Several actions are provided to the LLM for execution within the virtual environment as part of the action definitions (4). For locomotion there are these actions: 'MoveTo(ID)', 'MoveBy(right, forward)', 'RotateTo(ID)' and 'RotateBy(y)'. The LLM can use these actions to move and orientate itself relative to an object as well as in absolute terms. For object manipulation, the actions 'TakeObject(ID)', 'PlaceObject(ID)', 'RotateObject(ID, X, Y, Z)' and 'ScaleObject(ID, factor)' are available to fully manipulate objects according to LaViola et al. [5]. The virtual environment also provides the LLM with two functions for querying the environment. The 'GetEnvironment()' function generates the DSL of the SceneGraph and is always automatically inserted into the prompt at the start of an interaction. The 'Raycast(direction)' method allows the LLM to identify the first object that was hit by a raycast.

## 4 EXPERIMENTS

To evaluate the effectiveness of the developed voice assistance system, several experiments were conducted. These experiments aimed to test different aspects of interaction in virtual environments and to evaluate the system's ability to interpret and execute natural language commands.

### 4.1 Concept

To ensure a complete and robust evaluation of the developed voice assistance system, several key objectives were defined to analyze the performance of the system in different XR contexts. These objectives include:

1. The evaluation of the system's ability to understand and implement natural language instructions.
2. The investigation of the robustness against different formulations and degrees of complexity of instructions as well as the comparison of the system performance with instructions addressed to humans or to a computer system.
3. Identification of strengths and weaknesses of the system in different interaction scenarios.
4. Analyzing the efficiency and accuracy of the system when executing user instructions
5. Analyzing the system's ability to process and use contextual information.

To achieve these goals, three different scenarios were developed to cover different aspects of XR interaction. The first scenario, the „dice" scenario, focused on object manipulation. In a virtual environment with a table and four colored cubes, participants were asked to give instructions to stack the cubes in a specific order. This scenario presented challenges in terms of precise object identification and the interpretation of spatial instructions.

The second scenario, „Corridor", focused on navigation and locomotion. Here, the participants had to formulate instructions to navigate through three consecutive corridors with a decreasing number of landmarks. This scenario tested the system's ability to interpret complex navigation instructions and utilize landmarks.

The third scenario, „aircraft", focused on complex object manipulation. In a virtual room with a hovering aircraft model, the participants had to give instructions for rotating and scaling the model. This scenario required a special understanding and ability to implement complex spatial transformations. It also examines the linguistic variability in relation to the formulation of rotation instructions.

In order to obtain realistic and diverse user input for the evaluation, an online survey was conducted with 40 participants aged between the ages of 18 and 40. The participants were presented with videos and images of the three scenarios and asked to formulate voice commands in text form. A distinction was made between instructions to a human and to a computer system. Users were asked to formulate instructions specifically towards another human being and, in a second iteration, specifically towards a computer system. This methodology aimed at generating a diverse data set of natural language instructions and to analyze differences in the formulation between addressing humans and computers as embodied agents. This approach also helped the participants to avoid any bias.

A test environment was designed based on the data collected. The scenarios were implemented in Unity and the language assistance system was integrated into the test environment. The test procedure included the initialization of the scenario, the input of a natural language instruction, the processing by the system, the execution of the interpreted commands and the automatic recording and evaluation of the final result. For each scenario, the automated tests looked like this:

1. Input: Natural language commands from the online survey data set
2. Processing: Run through the language assistance system
3. Execution: Running the actions of the interpreted commands in the virtual environment
4. Evaluation: Automatic evaluation of the final result based on scenario-specific criteria

Both quantitative and qualitative metrics were defined to evaluate system performance. Quantitative metrics included the success rate, the execution time and the number of actions required. Qualitative criteria included the accuracy of object identification, the correctness of spatial interpretation and the appropriateness of the selected actions of the LLM. In addition, scenario-specific evaluation criteria were defined, such as the correct sequence of stacked cubes in the „cube" scenario or the precision of rotation and scaling in the „aircraft" scenario.

The data analysis consisted of both statistical and qualitative methods. Statistical analyzes included the calculation of means and standard deviations, linguistic analyses to compare human and computer addressing performance, and analyses of variance to examine differences in performance between scenarios. Qualitative analyzes included the categorization of error types, content analyses of the language assistance system output and linguistic analyses of user input.

When designing the experiments, limitations such as the limited sample size of the online survey and the focus on specific XR scenarios were recognized and documented.

By combining different scenarios, collecting realistic user input, and defining clear evaluation criteria, a solid basis was created for the evaluation of the system, which was intended to show both the strengths and the potential for improvement of the system.

### 4.2 Results

The quantitative analysis showed clear differences in the success rates between the various scenarios. The „cube" scenario, which focused on object manipulation, had the highest success rates with 88.46% for instructions formulated towards humans and 73.91% for instructions formulated towards a computer system. These results suggest that the system performs robustly in simple manipulation tasks, although the slightly lower rate for instructions directed towards a computer system indicates potential for improvement in the interpretation of technical formulations.

In contrast, the „corridor" scenario, which tested navigation and locomotion, showed significantly lower values with a success rate

of 35.29% for both instruction types. This discrepancy may be due to the higher complexity of the navigation tasks and the methodology of the study. As the user instructions were recorded in advance and then played back asynchronously, errors that occurred in the second corridor, for example, could not be corrected. As a result, the entire interaction was categorized as a failure, even if only a partial step was not carried out correctly. The fact that the success rates for human- and computer-directed instructions were identical suggests that the difficulties lie more in the basic processing of complex spatial concepts than in the adaptation to different formulation styles. It is worth noting that 64.64% of failed interactions did successfully pass the first corridor and failures only manifested along the further sequence of actions. The failures can be classified into four distinct categories.

Firstly, object recognition errors occurred, with the LLM failing to identify the correctly referenced objects for both interaction recipients in one single case each.

Secondly, navigation problems were identified, which manifested themselves in a total of eight cases. These problems were due to difficulties in resolving the target position through Unity's navigation mesh, which was caused by an error in the level design in Unity. Technically, this means that the fault was per design and is not a principle problem, however, this also emphasized the sensitivity of the approach to a proper design, which might require a higher modeling quality than for standard navigation techniques and thus will increase the design time of virtual levels.

The third category involved errors in relative movement. Here, in three cases for each of the two interaction recipients, the LLM attempted to navigate the user to the end of the corridor through a sequence of small, 1m-long steps. This procedure ultimately led to an error due to the maximum permissible number of instructions being exceeded. This issue can be solved using prompt engineering.

Finally, in one case of instructions directed to humans, it was observed that the LLM was unable to correctly interpret and indicate the intended direction of rotation, since the user was specifying a rotation by 90° without specifying the direction.

These findings point to specific challenges that need to be considered when developing and improving LLM-based navigation systems in VEs. In particular, abilities to correctly recognize objects, to effectively use the provided actions for locomotion, and to precisely interpret directional instructions are critical aspects that require further optimization.

To summarize, the developed voice assistance system shows promising performance in basic object manipulations and locomotion, but has clear weaknesses in complex spatial tasks and multi-step navigation instructions. The results emphasize the need to improve the spatial reasoning of the system. Future improvements could focus on the integration of advanced spatial models and the development of techniques for more effective processing and storage of contextual information, so that the LLM has more specific contextual information available for each interaction.

It is noteworthy that the end-to-end latency of the system is currently still beyond what is acceptable for direct control-based interactions with VR. This is primarily due to the runtime of the LLMs and speech recognition services, which in this case were online versions, coming along with higher latencies. However, current developments in this field of LLMs already show significant speed improvements and we expect this to be less of an issue in the near future.

## 5 Conclusion

This paper shows how a large language model can be used to implement a voice-based assistance system in XR environments. The presented method demonstrates the potential for natural language user interactions without relying on predefined voice commands. The core contributions of this paper include an architecture that integrates LLMs as embodied agents in XR environments and utilizes their speech understanding and contextual reasoning capabilities, a simple method for representing spatial information as a textual scene graph that enables LLMs to understand and process complex spatial relationships, the implementation of a ,,self-summary" approach to managing short- and long-term memory that enables contextual and consistent interactions over extended periods of time, and a comprehensive evaluation of the system in various XR scenarios that demonstrates both the potential and the challenges of the approach.

The results show that the system performs promisingly in basic object manipulation and navigation tasks. However, weaknesses were evident in more complex spatial tasks, particularly multi-step navigation instructions and complex rotations. These findings emphasize the need for further research to improve the spatial reasoning of LLMs in XR contexts.

Future work should focus on improving spatial representation and reasoning, possibly by integrating advanced spatial models or better architectures, optimizing context processing to extract and use relevant information more efficiently, extending the multimodal capabilities of the system, to seamlessly integrate gestures, gaze direction, and other input modalities, investigating methods to improve resilience to different formulations and levels of abstraction in user instructions, and exploring techniques to reduce latency and improve the real-time capability of the system. Despite the challenges identified, this work demonstrates the significant potential of LLMs to improve user interaction in XR environments. The presented approach opens up new possibilities for intuitive, context-sensitive and adaptive assistance systems in virtual and augmented realities. This system can benefit from all advances in large language models and therefore can become better by just connecting it to a more optimized model. With further progress in this area, a future can be expected in which interaction with complex virtual environments becomes as natural and intuitive as communicating with a human assistant.

## References


[1] G. Abrami, A. Henlein, A. Kett, and A. Mehler. Text2scenevr: Generating hypertexts with vannotator as a pre-processing step for text2scene systems. In *Proceedings of the 31st ACM Conference on Hypertext and Social Media*, HT '20, p. 177–186. Association for Computing Machinery, New York, NY, USA, 2020. doi: 10.1145/3372923.3404791 4

[2] B. bench authors. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models, 2023. 2

[3] H. Biggie, A. N. Mopidevi, D. Woods, and C. Heckman. Tell me where to go: A composable framework for context-aware embodied robot navigation. In *7th Annual Conference on Robot Learning*, 2023. 2

[4] R. A. Bolt. "put-that-there": Voice and gesture at the graphics interface. In *Proceedings of the 7th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '80, p. 262–270. Association for Computing Machinery, New York, NY, USA, 1980. doi: 10.1145/800250.807503 1

[5] D. A. Bowman, E. Kruijff, J. J. LaViola, and I. Poupyrev. *3D User Interfaces: Theory and Practice*. Addison Wesley Longman Publishing Co., Inc., USA, 2004. 6

[6] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, eds., *Advances in Neural Information Processing Systems*, vol. 33, pp. 1877–1901. Curran Associates, Inc., 2020. 1

[7] S. Bubeck, V. Chandrasekaran, R. Eldan, J. Gehrke, E. Horvitz, E. Kamar, P. Lee, Y. T. Lee, Y. Li, S. Lundberg, H. Nori, H. Palangi, M. T.

Ribeiro, and Y. Zhang. Sparks of artificial general intelligence: Early experiments with gpt-4, 2023. arxiv: 2303.12712v5, preprint. 1

[8] A. Chang, M. Savva, and C. D. Manning. Learning spatial knowledge for text to 3D scene generation. In A. Moschitti, B. Pang, and W. Daelemans, eds., *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 2028–2038. Association for Computational Linguistics, Doha, Qatar, Oct. 2014. doi: 10.3115/v1/D14-1217 4

[9] G. Dagan, F. Keller, and A. Lascarides. Dynamic planning with a llm, 2023. arxiv: 2308.06391v1, preprint. 2

[10] X. Daull, P. Bellot, E. Bruno, V. Martin, and E. Murisasco. Complex qa and language models hybrid architectures, survey, 2023. arxiv: 2302.09051v4, preprint. 2

[11] Y. Ding, X. Zhang, C. Paxton, and S. Zhang. Task and motion planning with large language models for object rearrangement, 2023. arxiv: 2303.06247v4, preprint. 4

[12] A. Ferracani, M. Faustino, G. Giannini, L. Landucci, and A. Del Bimbo. Natural experiences in museums through virtual reality and voice commands. pp. 1233–1234, 10 2017. doi: 10.1145/3123266.3127916 1

[13] R. Fribourg, N. Ogawa, L. Hoyet, F. Argelaguet Sanz, T. Narumi, M. Hirose, and A. Lécuyer. Virtual Co-Embodiment: Evaluation of the Sense of Agency while Sharing the Control of a Virtual Body among Two Individuals. *IEEE Transactions on Visualization and Computer Graphics*, June 2020. doi: 10.1109/TVCG.2020.2999197 3

[14] J. Hombeck, H. Voigt, T. Heggemann, R. R. Datta, and K. Lawonn. Tell me where to go: Voice-controlled hands-free locomotion for virtual reality systems. In *2023 IEEE Conference Virtual Reality and 3D User Interfaces (VR)*, pp. 123–134, 2023. doi: 10.1109/VR55154.2023.00028 1

[15] J. Huang and K. C.-C. Chang. Towards reasoning in large language models: A survey. In A. Rogers, J. Boyd-Graber, and N. Okazaki, eds., *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 1049–1065. Association for Computational Linguistics, Toronto, Canada, July 2023. doi: 10.18653/v1/2023.findings-acl.67 2

[16] W. Huang, P. Abbeel, D. Pathak, and I. Mordatch. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents, 2022. arxiv: 2201.07207v2, preprint. 2

[17] W. Huang, F. Xia, T. Xiao, H. Chan, J. Liang, P. Florence, A. Zeng, J. Tompson, I. Mordatch, Y. Chebotar, P. Sermanet, N. Brown, T. Jackson, L. Luu, S. Levine, K. Hausman, and B. Ichter. Inner monologue: Embodied reasoning through planning with language models, 2022. arxiv: 2207.05608v1, preprint. 1, 2

[18] T. Kojima, S. S. Gu, M. Reid, Y. Matsuo, and Y. Iwasawa. Large language models are zero-shot reasoners, 2024. 2, 3

[19] F. Li, D. C. Hogg, and A. G. Cohn. Advancing spatial reasoning in large language models: An in-depth evaluation and enhancement using the stepgame benchmark, 2024. arxiv: 2401.03991v1, preprint. 2

[20] B. Liu, Y. Jiang, X. Zhang, Q. Liu, S. Zhang, J. Biswas, and P. Stone. Llm+p: Empowering large language models with optimal planning proficiency, 2023. arxiv: 2304.11477v3, preprint. 4

[21] N. F. Liu, K. Lin, J. Hewitt, A. Paranjape, M. Bevilacqua, F. Petroni, and P. Liang. Lost in the middle: How language models use long contexts, 2023. arxiv: 2307.03172v3, preprint. 2

[22] S. Mirchandani, F. Xia, P. Florence, B. Ichter, D. Driess, M. G. Arenas, K. Rao, D. Sadigh, and A. Zeng. Large language models as general pattern machines, 2023. arxiv: 2307.04721v2, preprint. 2

[23] R. Mirzaee, H. Rajaby Faghihi, Q. Ning, and P. Kordjamshidi. SPARTQA: A textual question answering benchmark for spatial reasoning. In K. Toutanova, A. Rumshisky, L. Zettlemoyer, D. Hakkani-Tur, I. Beltagy, S. Bethard, R. Cotterell, T. Chakraborty, and Y. Zhou, eds., *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 4582–4598. Association for Computational Linguistics, Online, June 2021. doi: 10.18653/v1/2021.naacl-main.364 2

[24] OpenAI. Gpt-4 technical report, 2024. arxiv: 2303.08774v6, preprint. 5

[25] OpenAI. Hello gpt-4o. https://openai.com/index/hello-gpt-4o/, 2024. [Accessed 08-07-2024]. 4

[26] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever. Learning transferable visual models from natural language supervision. In M. Meila and T. Zhang, eds., *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, vol. 139 of *Proceedings of Machine Learning Research*, pp. 8748–8763. PMLR, 2021. 4

[27] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever. Robust speech recognition via large-scale weak supervision. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org, 2023. 5

[28] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 779–788, 2016. doi: 10.1109/CVPR.2016.91 4

[29] Z. Shi, Q. Zhang, and A. Lipani. Stepgame: A new benchmark for robust multi-hop spatial reasoning in texts. In *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, pp. 11321–11329, Jun. 2022. doi: 10.1609/aaai.v36i10.21383 2

[30] J. Sin and C. Munteanu. Let's go there: Voice and pointing together in vr. pp. 1–3, 10 2020. doi: 10.1145/3406324.3410537 2

[31] S. Valencia, R. Cave, K. Kallarackal, K. Seaver, M. Terry, and S. K. Kane. "the less i type, the better": How ai language models can enhance or impede communication for aac users. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, CHI '23. Association for Computing Machinery, New York, NY, USA, 2023. doi: 10.1145/3544548.3581560 1

[32] G. Wang, Y. Xie, Y. Jiang, A. Mandlekar, C. Xiao, Y. Zhu, L. Fan, and A. Anandkumar. Voyager: An open-ended embodied agent with large language models, 2023. arxiv: 2305.16291v2, preprint. 2

[33] X. Wang, J. Wei, D. Schuurmans, Q. Le, E. Chi, S. Narang, A. Chowdhery, and D. Zhou. Self-consistency improves chain of thought reasoning in language models, 2023. arxiv: 2203.11171v4, preprint. 3

[34] T. Webb, K. J. Holyoak, and H. Lu. Emergent analogical reasoning in large language models, 2023. 1

[35] J. Wei, X. Wang, D. Schuurmans, M. Bosma, B. Ichter, F. Xia, E. H. Chi, Q. V. Le, and D. Zhou. Chain-of-thought prompting elicits reasoning in large language models. 2024. 2, 3

[36] W. Wu, S. Mao, Y. Zhang, Y. Xia, L. Dong, L. Cui, and F. Wei. Mind's eye of llms: Visualization-of-thought elicits spatial reasoning in large language models, 2024. arxiv: 2404.03622v2, preprint. 2

[37] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, and Y. Cao. React: Synergizing reasoning and acting in language models, 2023. arXiv:2210.03629, v3, preprint. 2, 3

[38] T. Zhang, F. Ladhak, E. Durmus, P. Liang, K. McKeown, and T. B. Hashimoto. Benchmarking Large Language Models for News Summarization. *Transactions of the Association for Computational Linguistics*, 12:39–57, 01 2024. doi: 10.1162/tacl_a_00632 4

[39] Y. Zhao, H. Fei, W. Ji, J. Wei, M. Zhang, M. Zhang, and T.-S. Chua. Generating visual spatial description via holistic 3D scene understanding. In A. Rogers, J. Boyd-Graber, and N. Okazaki, eds., *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7960–7977. Association for Computational Linguistics, Toronto, Canada, July 2023. doi: 10.18653/v1/2023.acl-long.442 4