

---

SAARLAND UNIVERSITY

Faculty of Mathematics and Computer Science  
Department of Computer Science  
Master's Thesis



---

# Large Language Model Applications in a Mobile Augmented Reality Tourism Scenario

Submitted by:  
Alice Virginia Chase  
June 19, 2024

Reviewers:  
Prof. Dr. Antonio Krüger  
Dr. Alassane Ndiaye

---

## **Erklärung**

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

## **Statement**

I hereby confirm that I have written this thesis on my own and that I have not used any other media or materials than the ones referred to in this thesis

## **Einverständniserklärung**

Ich bin damit einverstanden, dass meine (bestandene) Arbeit in beiden Versionen in die Bibliothek der Informatik aufgenommen und damit veröffentlicht wird.

## **Declaration of Consent**

I agree to make both versions of my thesis (with a passing grade) accessible to the public by having them added to the library of the Computer Science Department.

Saarbrücken, \_\_\_\_\_  
19.06.2024  
(Datum/Date)

*Alice V. Chase*  
\_\_\_\_\_  
(Unterschrift/Signature)

## **Abstract**

Large language models such as ChatGPT have exploded in popularity with many domains adapting them to their use as they can be used to create chatbots that provide natural, human-like responses to a variety of topics. However, these models often suffer from hallucinations and fine-tuning the weights to a particular domain may be expensive. As part of the 5G-SLS project we adapt the GPT-4 model, the successor to the model used by ChatGPT, to a tourism domain by supplying it with extra information obtained from Wikipedia and documents from the Saarlouis museum to act as a custom knowledge base. This knowledge base, along with 3D reconstructions of historical buildings deployed on a Hololens2, allows users to get detailed and accurate information quickly and reliably through interaction with GPT-4 as their virtual tour guide. In addition, the application contains speech-to-text capabilities and spatial information in order to let users interact with the chatbot in a natural manner and orient themselves relative to the virtual models. To better understand the chatbot performance, a German language user study was conducted in which participants used the model to find answers to questions about Saarlouis' history and architecture.

---

# Contents

<b>1</b>	<b>Introduction and Problem Statement</b>	<b>1</b>
<b>2</b>	<b>Related Work</b>	<b>3</b>
2.1	Mobile Augmented Reality and 5G . . . . .	3
2.2	Embeddings and Information Retrieval . . . . .	4
2.3	Large Language Models and OpenAI . . . . .	6
2.4	Speech-to-Text and Text-to-Speech . . . . .	10
2.5	Prompt Engineering . . . . .	12
2.6	Referring Expressions . . . . .	14
<b>3</b>	<b>Implementation</b>	<b>17</b>
3.1	LangChain Library . . . . .	19
3.2	Custom Knowledge Base . . . . .	20
3.3	LLM Parameters and Prompting . . . . .	24
3.3.1	Prompts . . . . .	25
3.4	Spatial Awareness . . . . .	26
3.5	Speech-to-Text and Text-to-Speech . . . . .	28
<b>4</b>	<b>User Study</b>	<b>30</b>
4.1	Participants . . . . .	31
4.2	Setup and Design . . . . .	31
4.3	Results . . . . .	36
<b>5</b>	<b>Discussion</b>	<b>42</b>
5.1	User Experience . . . . .	42
5.2	Speech Recognition . . . . .	43
5.3	Queries, Responses and Database Usage . . . . .	43
5.4	Hallucinations . . . . .	48

<b>6 Conclusion and Future Work</b>	<b>49</b>
6.1 Findings and Contributions . . . . .	49
6.2 Future Work . . . . .	50
<b>Bibliography</b>	<b>52</b>
<b>Appendix</b>	<b>57</b>

---

---

# **Chapter 1**

## **Introduction and Problem Statement**

Saarlouis, a district town in Saarland, Germany, named after Louis XIV of France and originally built as a fortress, is aiming to enhance living and working conditions by implementing a 'smart city strategy'. This initiative, supported by the Federal Ministry of Digital and Transport since November 2021, focuses on the "5G-SLS" [12] project whose goal is to develop and test use cases based on the new 5G mobile communications standard. These use cases, categorized into energy, tourism and transport, aim to offer residents and visitors a firsthand experience of new technologies, contributing to local infrastructure improvement. The project involves collaboration among Stadtwerke Saarlouis GmbH, the district town of Saarlouis, the German Research Center for Artificial Intelligence (DFKI), and Saarland University of Applied Sciences (HTW Saar), bringing together many fields such as energy supply, augmented and mixed reality, tourism, and transport planning.

As part of the tourism sector, to provide tourists and visitors with a more immersive experience, the application SaAR-Louis was conceived as a novel approach to AR/MR touristic applications [12]. Unlike many traditional touristic information apps, which predominantly rely on smartphones, this project uses virtual reality headsets. In recent years, Augmented and Mixed Reality (AR/MR) technologies can offer visitors more immersive and unique educational experiences, changing the way tourists engage with historical sites and cities, fueled by presenting new opportunities to interact with virtual elements that blend into the physical world. While Saarlouis has a unique history with its french and german influences, World War II and other incidents left significant parts of the city in ruins, and the subsequent reconstruction efforts led to changes in its historical structures.

Thus, this thesis introduces a chat-based feature into the application introduced in "SaAR-

Louis: Augmented and Mixed Reality Touristic Applications" with the goal of offering a more immersive way for tourists to explore the city's unique history. Together with 3D reconstructions deployed on a Hololens2, this creates a prototype which takes users on a smart tour of the city.

The chat features make use of transformer-based large language models which have become state of the art for language generation due to their ability to provide natural, human-like responses on a variety of topics. This thesis adapts GPT-4, the successor to the model used by ChatGPT, to our tourism domain by supplying it with extra information obtained from Wikipedia and documents from the Saarlouis museum to act as a custom knowledge base.

In the second chapter this thesis explores the current mixed-reality(MR) landscape in mobile settings before going into various machine learning and natural language processing (NLP) techniques. The third chapter presents the current implementation and how it brings various NLP techniques together, supplied with information from the augmented reality(AR) device, to create a smart chatbot in the tourism domain. Finally, we address the ongoing challenges and future of the project.

This thesis is applied research that primarily explores how to mitigate large language models' (LLM) issues with producing incorrect information in domain-specific tasks and the practical application of a complex, low latency chat system that can be employed on a mobile device utilizing 5G communications. The thesis also conducts a small exploratory user study to test the current prototype, explained in chapter 4 through 5.

Thus, the contributions of the thesis can be listed as follows:

1. Adapt GPT-4 to a tourism domain with accurate, factual recall about Saarlouis and its history using a custom database and referring expressions to orient the user
2. Interact with GPT-4 via speech recognition deployed on a Hololens2 device in a mixed-reality setting
3. Evaluate the prototype in a small, German-language user study

---

---

# **Chapter 2**

## **Related Work**

In this chapter we briefly review similar tourism related projects and 5G architecture. Next, we describe how text is represented by computers followed by a description of the transformer models which power large language models. This is followed by a description of text-to-speech and speech-to-text methods and interaction and description information necessary for using LLMs effectively.

### **2.1 Mobile Augmented Reality and 5G**

Mobile Augmented Reality (MAR) applications are applications which superimpose computer-generated images over a user's actual view. These can be run on mobile devices such as smartphones or wearables. This allows for more immersive and informative experiences through multiple modalities provided by the MAR technology such as images, text, audio, and videos [30]. Increasingly this technology is being used in museums, including ones pertaining to art, natural history and science and technology [13].

A project by Koo et. al [24] created a prototype MAR tour guide for a world heritage cultural site while keeping in mind important strategies and factors such as usability, gamification, and visual appeal to enhance the cultural site's historical, cultural, and architectural themes. Incorporated AR elements included navigation, architectural visualization and games; a user evaluation found that the overall results of the application were positive. The project was developed in Unity and deployed on a Samsung Galaxy S9 and the only data transmission was GPS usage [24].

Another tour-based mobile application with AR features for visualisation of the house

---

of Olbrich in Darmstadt, Germany relied on wireless communication and outsourced the AR processing tasks to a cloud service and also had higher latency between their client and server (2-3 seconds) [22]. Thus, because the AR technology of this device was only available when there was a network connection, in future applications the 5G system supporting MAR communication should provide smooth transitions from one base station to another so the impact on the MAR functionality isn't noticeable to the user [30].

MAR devices have many requirements: ultra-low latency, extremely high bandwidth and massive connectivity. Non-3GPP access methods such as Wi-Fi is not a good choice for these types of applications due to handover, reliability and performance issues, thus the future of MAR will involve direct 5G connectivity [30].

## 2.2 Embeddings and Information Retrieval

An essential component to understanding how computers process and extract meaning from text is with embeddings. Word embeddings are real-valued vector representations of text that can capture things like the context of a word in a document, semantic and syntactic similarity, relation with other words, and more [5].

To create a word embedding the text data is tokenized, where tokens are units consisting of characters, words, or subwords. When tokens frequently appear close to one another in text, the values of their vectors will also be similar to each other and we can calculate their distance from each other in their embedding space by techniques such as cosine similarity or dot product. Cosine similarity measures the cosine of the angle between the two vectors.

$$\cos(\mathbf{t}, \mathbf{e}) = \frac{\mathbf{t}\mathbf{e}}{\|\mathbf{t}\| \|\mathbf{e}\|} = \frac{\sum_{i=1}^n t_i e_i}{\sqrt{\sum_{i=1}^n (t_i)^2} \sqrt{\sum_{i=1}^n (e_i)^2}} \quad (2.1)$$

Note that the cosine similarity does not take into account vector magnitude - this is shown in the denominator of the function which scales the numerator to a number between -1 and +1. Smaller angles between vectors lead to larger cosine values (closer to +1), indicating greater similarity. The dot product is just the numerator of the cosine similarity and is computationally less expensive to compute. The dot product is more commonly used when calculating similarity for attention in transformer models [51] whereas cosine similarity is often used to measure document similarity because we want to compare how similar two or more documents are regardless of their length.

By doing these similarity calculations, we can capture aspects of the words' relationships. It is important to note that how vectors relate to each other in vector space is context-dependent. For example, if word embeddings are made using poem data, words that rhyme such as 'cat' and 'hat' will likely appear closer in vector space, whereas 'dog' may

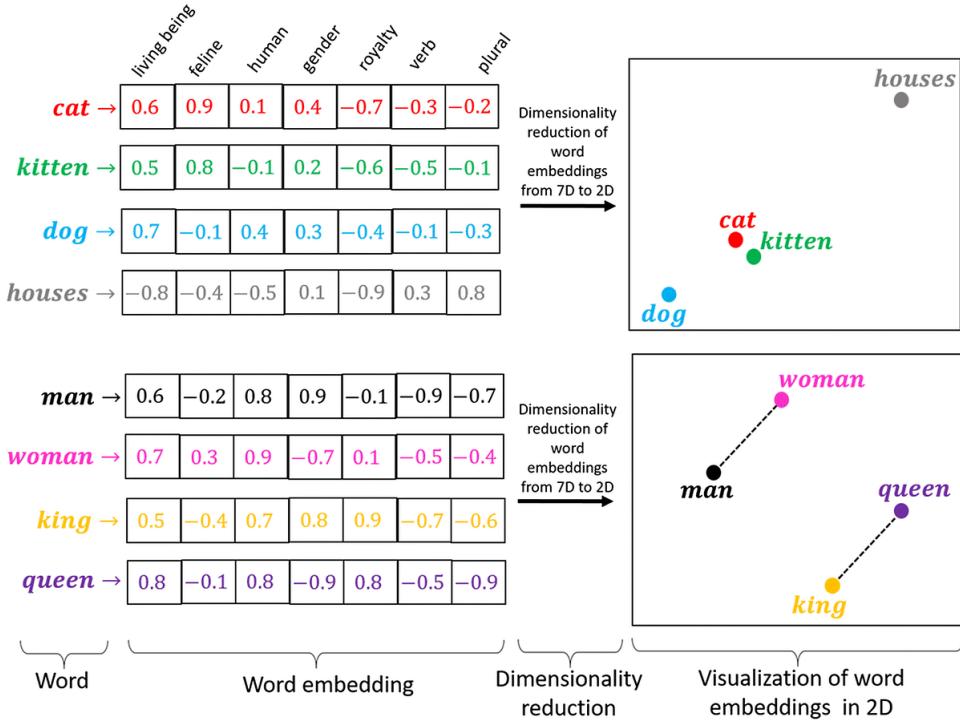


Figure 2.1: Vector representation and visualization [18].

appear further away from ‘cat’, but if we trained it on a text about domesticated animals, ‘cat’ and ‘dog’ vector values will be more similar. Figure 2.1 presents a visual of this word embedding process and their representation in 2D space.

However, embeddings don’t always have to be at the word-level. Embeddings can be generated for larger pieces of text, where the documents containing context information are chunked, or split up into smaller sections of text. This reduces noise allowing the search algorithm to better find relevant data [25]. The chunks consist of a sliding window over a specified amount of tokens that usually contain an amount of overlapping tokens so important context isn’t lost. The embeddings may simply be saved in a file or, for a larger dataset, a vector database is a better choice.

Embeddings work well for information retrieval from a vector-based knowledge base (KB) in the context of a closed domain Q&A system such as ours [39]. At runtime, a query is given to the system (in our case a question that the user wants to ask the chatbot), which is encoded and compared against the database embeddings using a similarity search. They are ranked according to their similarity and then the top-n documents are retrieved - database embeddings are tied back to their original object (the textual documents) with an ID, usually in the form of a dictionary/hash table [39] and sent along to the next step in the pipeline (such as the final LLM call) in text form along with the original user query.

## 2.3 Large Language Models and OpenAI

Large language models are pre-trained, transformer-based models with parameters in the hundreds of billions, making them capable of tasks such as in-context learning, instruction following and step-by-step reasoning that are not found in models with fewer parameters [53].

ChatGPT, released in November 2022, is a type of large language model optimized for chat. GPT stands for Generative Pre-trained Transformer. Transformer architectures rely on the concept of self-attention, which was a paradigm shift for processing sequential data [51]. Prior to transformer architecture, precursors such as Recurrent Neural Networks (RNN) [38], Long Short-Term Memory (LSTM) [20] and Convolutional Neural Networks (CNN) [27] failed to capture long-range context dependencies, or do so in a manner that is too computationally expensive and very difficult to scale.

Unlike sequence-to-sequence models such as RNN or LSTM, transformers do not rely on past hidden states to capture dependencies with previous words. Instead, self-attention is used to compute similarity scores between words in a sentence and is the main reason why transformers do not suffer from long term dependency issues. Attention ‘heads’ are present in each encoder and decoder in the stack. The encoder-decoder architecture is a typical neural generation design used for transforming one sequence into another (such as machine translation or question-answering), where the encoder compresses the input sequence into a fixed-size context vector, and the decoder uses this vector to generate the output sequence [6]. It is commonly applied in tasks like machine translation, text summarization, and image captioning. Another innovation proposed in the original transformer architecture are positional embeddings which encode information related to the specific position of a token in a sentence. Positional embeddings are made by mapping each index of a token to a unique vector by using the sine and cosine functions with varying wavelengths. The positional embeddings are the same dimension as the word embeddings and their values are added element-wise to the word embeddings before feeding them into the model. Figure 2.2 shows the architecture of the transformer model.

The idea of self-attention is for each step in the decoder to have direct access to the inputs and calculate how similar each word is to all the words in the sentence, including itself. This captures contextual information such as pronoun resolution. If you take an example sentence, "The day was so hot, I drank some cold water and it was refreshing," the word 'water' should be more relevant for 'it' than the noun 'day' and the attention mechanism captures this relationship. This process draws inspiration from information retrieval in the form of keys, values, and queries which are generated by passing the embedded input plus positional encoding through three separate linear layers each with different weights as shown in Figure 2.3. The output of an attention block is a weighted average

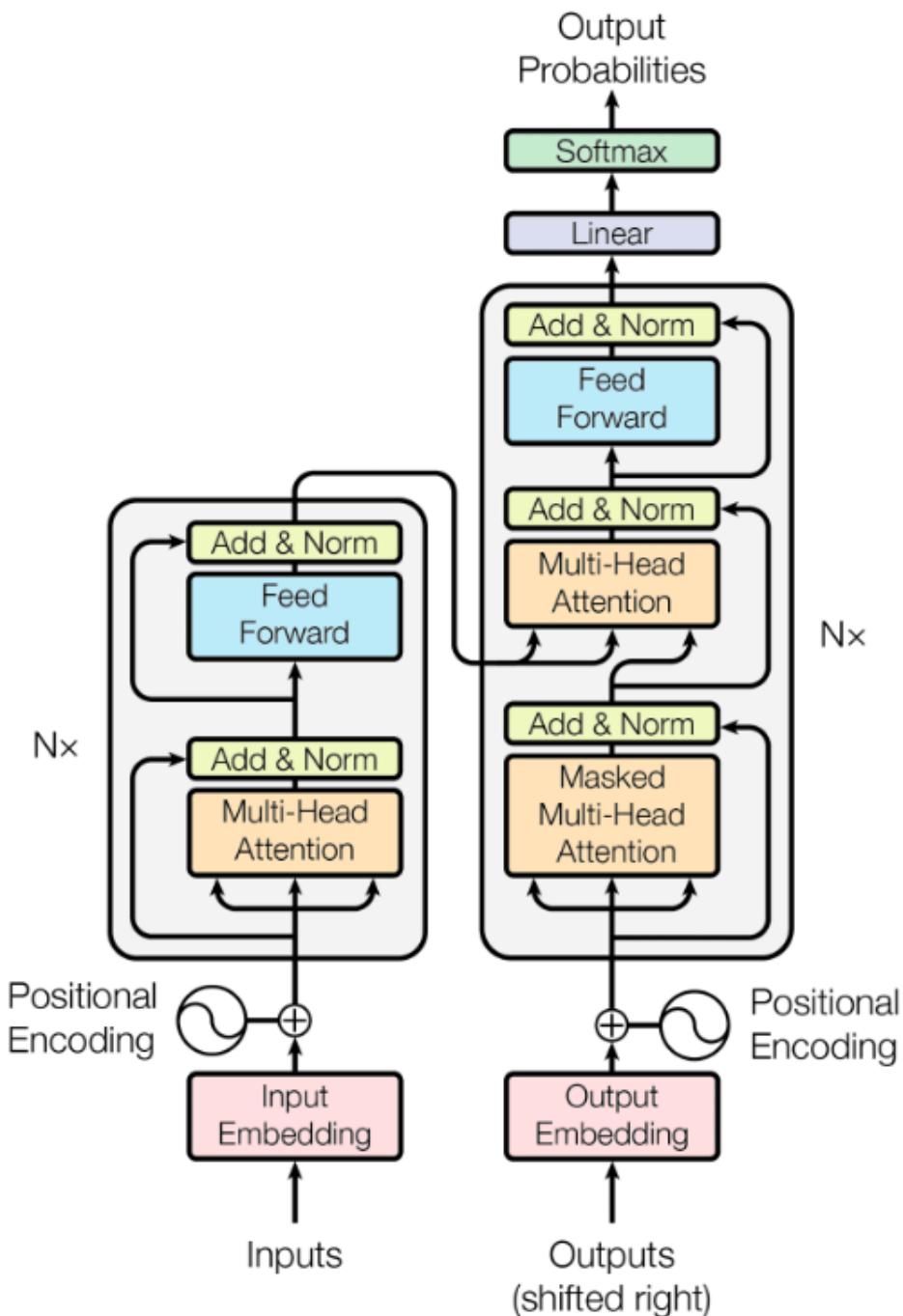


Figure 2.2: The original transformer architecture taken from [51].

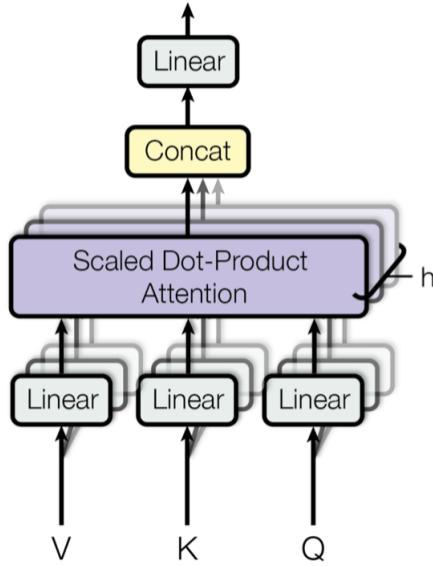


Figure 2.3: Attention head [51].

of values in a dictionary of key-value pairs, represented by the following formula:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (2.2)$$

Where  $Q$  is the query,  $K$  is the key and  $V$  is the vector. Note that the  $K, V, Q$  are matrices, consisting of individual vectors. For example,  $V$  is a matrix where each row is a word embedding. In the numerator of the attention function shown in Equation 2.2, the queries are compared to the keys using the dot product and then scaled by square rooting the dimension of  $K$ ,  $d_k$ . However the dimension for keys, queries, and values are usually equal for simplicity and efficiency. Next, softmax is applied which gives a probability distribution for what ‘percent’ of the encoded input word we should use when decoding.

Now we have a matrix which has rows equal to the number of input queries and columns equal to the number of input keys where each element is scaled to be between 0 and 1. Last, this matrix and  $V$  are multiplied to produce a matrix containing the weighted average of the rows in  $V$ , where the number of rows are equal to the number of queries. Thus, each row represents the importance of different words in the sequence for the specific query word.

Having multi-headed attention modules allows the model to become more expressive with different attention heads learning different linguistic phenomenon. In Figure 2.2 there are three places in which attention heads are used. Once in the encoder and twice in the decoder. The attention in the encoder is when the source sequence pays attention to itself. Similarly, attention in the decoder is the target sequence paying attention to

---

itself and the final encoder-decoder attention in the decoder is the target sequence pays attention to the source sequence. The masked decoder (target sequence paying attention to itself) refers to hiding right side tokens since text is read left to right, the model should not have access to the future tokens in order to learn how to predict them.

Described above is the original Transformer model as in [51]. Since then, many adapted the system with modifications. The GPT series, of which ChatGPT is part of, are decoder-only models [34]. GPT-2, an early model in the series, made adjustments to initialization, layer normalization, larger batch and context size, and a reversible tokenizer [45]. GPT-3 introduced further modifications most notably by scaling up the number of layers, their dimensions and training data [10]. OpenAI’s ChatGPT, GPT-3.5-turbo, and GPT-3 have 175 billion parameters [48] while GPT-4, released in March 2023, likely has a magnitude of order more and multimodal capabilities (image input), but the architectural details have not been publicly released. GPT-3.5 accepts only text input.

Model Name	Release Date	Parameters	Improvements
GPT-1	2018	117M	Generates short sequences of text given prompt
GPT-2	2019	1.5B	Human-like responses and cohesive short paragraphs
GPT-3	2020	175B	Human-like, natural sounding text as well as code and other genres, longer context
InstructGPT	2022	1.3B	Understands and generates text in response to instructions and structured data.
GPT-3.5	2022	175B	Fine-tuning with reinforcement learning with human feedback, longer context
GPT-4	2023	>175B	Multimodal, longer context

Table 2.1: Summary of GPT-series models.

Although scaling up a model’s size and training data was one of the biggest innovations and reasons for success in the GPT series, reinforcement learning from human feedback (RLHF) which is used to fine tune the models to better align with human intent, is what helps ChatGPT be so engaging to its users. For comparison, human evaluation on InstructGPT was more favorable which had 1.3 billion parameters and used RLHF, compared to GPT-3 which had 175 billion and did not employ RLHF [43].

Though these models have seen an explosion of popularity, it is important to note that the models are not fixed and are periodically updated but the source code is not available to users. Research has shown that for the same input prompts in domains such as solving math problems, answering sensitive/dangerous questions, code generation and visual reasoning that both GPT-3.5 and GPT-4 models’ answers change significantly in their

---

accuracy over a period of time as short as a couple months and that GPT-4 is not always better than its predecessor [11].

While attention-based architectures and scaling parameters into the billions along with RLHF allow LLMs to be very powerful in a diverse array of tasks, severe problems remain in their specificity and controllability which is leading to the rise of importance in prompts and prompt engineering through which people interact with LLMs [52].

## 2.4 Speech-to-Text and Text-to-Speech

As speech is the most natural mode of communication for people and it would be difficult to use a virtual keyboard in a Hololens application, integrating speech interaction is an essential component of the project. Speech can easily be recorded on the Hololens using the built-in microphones but a model is needed to convert the users' utterance to text. Speech-to-text (STT) which converts a sound wave to a textual transcription is also referred as automatic speech recognition (ASR). There are many factors that can affect the quality of speech recognition such as noise in the environment or linguistic ones like the model's grammatical and semantic robustness. Other aspects include sociolinguistic and personal variables like the speed at which a person speaks, accent, dialect and more [9]. State-of-the-art techniques in deep learning have vastly improved these systems. On the same hand, deep learning techniques often require large amounts of data to perform well and minority or underrepresented languages do not have these resources. Currently this thesis only concerns itself with German and English languages, of which there is an abundance of high quality data and STT and TTS systems compared to minority or low-resource languages such as the Western Germanic language Gronings or the Polynesian language Besemah [7].

STT or ASR systems are normally evaluated with two metrics: speed and accuracy. Speed is largely hardware dependent and word error rate (WER) or word recognition rate are used for the accuracy, which is a ratio of the errors in a transcript to the total words spoken [36]. This can be represented by the formula:

$$WER = \frac{S + D + I}{N} \quad (2.3)$$

where S is the number of substitutions (a word is misrecognized or replaced by a different one), D (words that are not recognized at all by the system and not added to the sequence) is the number of deletions, I is the number of insertions (ASR systems inserts words that weren't spoken) and N is the number of words in the reference.

Vocabulary size of an ASR model has a large impact because it can affect the speed, accuracy, and complexity of the system. A larger vocabulary may reduce the accuracy due to having more similar sounding words [36]. As with LLMs, attention-based encoder-

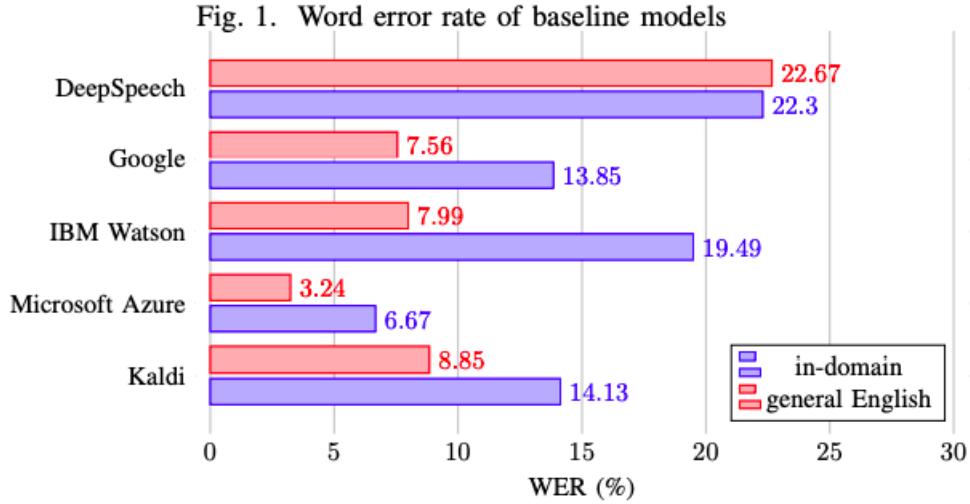


Figure 2.4: Evaluation of different baseline models, showing Azure with lowest WER. In-domain data is for commands for a robot such as ‘Move to home position’[4].

decoder architectures are now the state-of-the-art, replacing previous techniques such as Hidden Markov Models or sequence-to-sequence ones like RNNs.

Despite these good results, streaming scenarios are made difficult with this architecture leading them to not be widely used in practice as this architecture requires it to see the entire utterance before processing can start. This is because attention is computed between all elements in the sequence. Yet, streaming requires a real-time or extremely low-latency. This is important in our project because we want the conversation to feel as natural as possible for the users.

There have been proposed modifications to make this more feasible. One solution is to manage the timing delays in the encoder architecture, in which the extent of the future context in the input sequence is constrained to a predetermined size, known as restricted or time-restricted self-attention which was previously employed in hybrid HMM-based ASR systems [41]. Another approach by Tsunoo et. al [50] created a blockwise synchronous beam search algorithm that relies on blockwise processing of the encoder for streaming. During the beam search feature blocks were synchronously encoded through a block boundary detection technique. This method assesses the reliability of each predicted hypothesis by considering end-of-sequence markers and repeated tokens within the hypothesis.

For our project we use Microsoft’s Azure API, however the exact architecture is unknown due to it being closed source. A study evaluating different ASR systems for a service robot showed that Azure works best out-of-the-box compared to other API services and open source ASR services [4].

Likewise, Text-to-Speech (TTS) or speech synthesis has also immensely benefited from transformers [46] and is the opposite direction of STT - it generates a spectrogram from text to create speech that is not just intelligible but also natural. Thus, textual features need to be converted to acoustic ones [17].

## 2.5 Prompt Engineering

A prompt is a set of instructions, examples, or input texts to customize the language model's behavior and influence the generation of responses. It provides context information for the interaction and specifies what information is important [52][21]. Thus, prompt engineering is the process of designing and optimizing prompts, whether manually or automatically, to get the best results from a language model [52].

Language models can be sensitive to prompts and sometimes the issue isn't that the language model doesn't contain the relevant knowledge that the prompt is asking for but rather the prompt must be worded in a certain way in order to retrieve it [21]. Mining and paraphrasing-based methods can be used to automatically generate different prompts that are semantically similar to one another and when used to query a model can see higher accuracy than a single manually created prompt [21]. However, more robust LLM's should be able to handle different prompts but still return similar results.

Thus, if LLMs retain factual knowledge present in the training data which is accessible with the right prompts, this makes them have high potential to operate as open-domain question answering (QA) systems that do not require fine-tuning [44]. In the context of QA systems, LLMs have unified the architecture for solving QA tasks with different formats via prompting methods; LLMs can infer the desired format based on the query [33][23]. Previously, the QA task format was built into the model architecture itself such as whether it was extractive, abstractive, multiple-choice, or Yes/No. See Figure 2.4 below for different examples of different types of QA tasks.

With a conversational QA chatbot the user is not limited to the type of questions they can ask, thus generating appropriate answers to any question is useful but abstractive tasks are the most relevant where the answer to a query  $Q$  requires the model to generate text that is not a word-for-word copy-pasted substring from the context  $C$  [23].

To make a more interesting chatbot that does not regurgitate facts verbatim requires incorporating friendliness and enthusiasm into the responses and chatbot personality can also be shaped by prompts. A study [47] to see how reliably a personality can be implemented with LLM's used PaLM generative models and personality research from the field of psychology to isolate traits and adjectives that are good personality descriptors. This information was incorporated into prompts to create various personalities in a chatbot. For example, to elicit a personality with high extraversion a prompt such as: "PersonaChat description I'm extraverted, energetic, talkative, bold, active, assertive,

### Extractive [SQuAD]

**Question:** At what speed did the turbine operate?

**Context:** (Nikola\_Tesla) On his 50th birthday in 1906, Tesla demonstrated his 200 horsepower (150 kilowatts) 16,000 rpm bladeless turbine. ...

**Gold answer:** 16,000 rpm

### Abstractive [NarrativeQA]

**Question:** What does a drink from narcissus's spring cause the drinker to do?

**Context:** Mercury has awakened Echo, who weeps for Narcissus, and states that a drink from Narcissus's spring causes the drinkers to "Grow dotingly enamored of themselves." ...

**Gold answer:** fall in love with themselves

### Multiple-Choice [ARC-challenge]

**Question:** What does photosynthesis produce that helps plants grow?

**Candidate Answers:** (A) water (B) oxygen (C) protein (D) sugar

**Gold answer:** sugar

### Yes/No [BoolQ]

**Question:** Was America the first country to have a president?

**Context:** (President) The first usage of the word president to denote the highest official in a government was during the Commonwealth of England ...

**Gold answer:** no

Figure 2.5: Types of QA tasks [23].

and adventurous," is appropriate [47]. The study found that the LLMs can have robust personalities that are expressed in downstream tasks and when answering questions on psychometric personality tests, the output cannot be distinguished from a human one and that model size also reflects how well personality nuances are expressed.

The general process of improving the quality and accuracy of LLM responses by supplementing it with external knowledge is called retrieval-augmented generation (RAG) [28], the vectorization process as described in Section 2.2 is merely one way of doing it.

Although having our own database allows us to check whether the LLM responses are accurate, there is still the issue that even if the LLM can handle long contexts, whether this context is actually beneficial to performance. While augmenting a prompt often improves model performance [33], a study by Liu et. al [32] shows that LLM performance may degrade significantly when the position of the relevant information is changed in the input when dealing with long contexts. They found performance is often lowest when the key information is in the middle of long input contexts. In Figure 2.6 shown below, there is a U-shaped pattern in which the key information near the beginning or end of the input yields the highest accuracy. When the pertinent information is at the start or end of the context, the model performs better due to primacy and recency biases. Thus, accessing information from the middle of the context leads to a notable decline in performance. In addition, concatenating text and making longer prompts will also result in a longer processing time and this should be factored in when a low latency application is the goal. Thus, when inserting custom data into the prompt as retrieved by the RAG system, the amount of data retrieved and the prompt structure should be carefully designed so there is minimal information loss.

Finally, it is important to note the security concerns posed by LLMs. An ongoing problem in LLM usage is the creation of prompts in order to get malicious output, such as creating convincing fake news or impersonating individuals [35]. This can pose severe consequences in terms of spreading harmful misinformation or data privacy. Thus, a system using an LLM must be designed so that it both protects its users and is protected from them; people must have their data be secure and likewise not be able to break the model to produce malicious output.

## 2.6 Referring Expressions

Referring expressions (RE) are words or phrases which describe a domain object, often by utilizing unique attributes or relationships to other objects within the environment [31]. Leveraging contextual information and handling misinterpretations in a dynamic environment are open challenges in the referring expression generation (REG) domain [14]. To create as unambiguous REs as possible, it is important to incorporate spatial references ('furthest to the right', 'near the back', 'closest', etc.) from a specific perspective

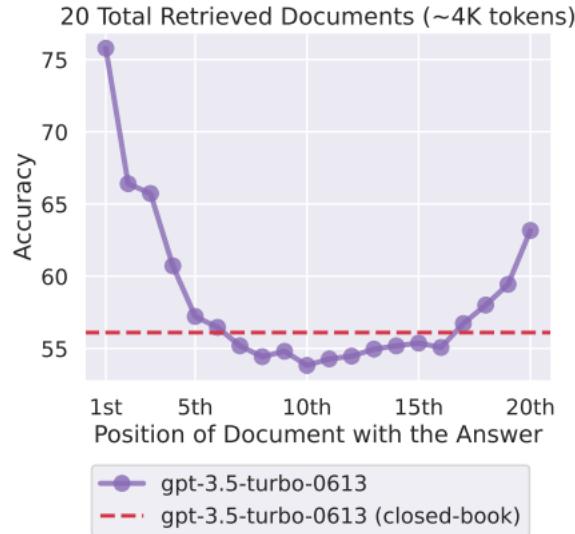


Figure 2.6: U-Shaped performance pattern (from [32]. Location of crucial information appears in the language model’s input affects its performance. Closed book refers to predicting without documents.

(i.e, the user’s) [29]. Using anchoring points and camera positioning available by the Hololens, the perspective is more straightforward than in other RE tasks. Given that a Hololens has a narrower field of vision (FoV) than humans, a horizontal FoV of 43° and a vertical of 29°, compared to the average person’s FoV of 130°–135° vertically and 200–220° horizontally, a previous study found that users found directional information useful when wearing a Hololens1, which has a more limited FoV than Hololens2 [19].

The majority of prior referring expression generation work has been done using rule-based methods. However, while these methods have seen great success [15], templates, rules, or other hard-coded thresholds make it difficult to generalize to other environments but in a closed domain this isn’t an issue.

The general procedure for rule-based REG approaches often first realizes the information as predicate logic formulae which are then converted to natural language using templates [26].

Rule-based techniques in relevant domains include taking sensor data as input and outputting a textual description referring to geographic locations, as seen in Figure 2.7 [40]. The first step involves creating specific rules according to a general template:

$$\text{SensorType}(x, y) \wedge C_1 \wedge \dots \wedge C_n \rightarrow \text{Template}(x, y) \quad (2.4)$$

Where  $x$  is the sensor variable,  $y$  is its type followed by a set of conditions  $C_i$ . Conditions evaluate values over features which may be according to a threshold, a comparison, or other criteria based on the application. Finally,  $\text{Template}(x, y)$  refers to the description

Number	Type of Measure	Latitude	Longitude	Generated Description
1	Water level	41.9697456359863	2.41436839103699	Sau reservoir
2	Rainfall	36.9890556335449	-2.35419869422913	Alhamilla Mountains
3	Water flow	38.4831809997559	0.796071231365204	River Vinalopó at Elda
4	Rainfall	40.034252166748	3.62156891822815	Aranjuez
5	Water flow	38.2390022277832	1.42935407161713	River Segura at Cieza
6	Rainfall	38.2390022277832	-1.96505701541901	San Miguel Mountains
7	Water level	38.3932685852051	-2.20556235313416	Fuensanta reservoir
8	Rainfall	38.7551918029785	-3.40385103225708	Valdepeñas

Figure 2.7: Sensor data to textual description from [40].

of sensor  $x$  is the textual description  $y$ . This can be used to then create fine-grained rules:

$$\begin{aligned} & \text{SensorType}(s, \text{WaterFlow}) \wedge \text{Feature}(s, \text{River}, x) \\ & \wedge \text{Feature}(s, \text{PopulatedPlace}, y) \rightarrow \text{Template}(s, [x, "at", y]) \end{aligned} \quad (2.5)$$

Which can be read as "If sensor  $s$  is a water flow sensor and  $s$  has the geographic features  $\text{River}$  and  $\text{PopulatedPlace}$  evaluated to some values  $x$  and  $y$ , then the natural language description for that sensor is the string  $x + \text{at} + y$ " [40].

---

---

# Chapter 3

## Implementation

The previous chapter describes the current technologies, their limitations and abilities. This chapter puts this information to practical use in order to optimize and augment GPT-4 to create a low latency chatbot that can answer questions accurately in our tourism domain while running on a mobile Microsoft Hololens2 device.

First an overview is given of the architecture of the larger SaAR-Louis application in which the chatbot feature and its components will be integrated.

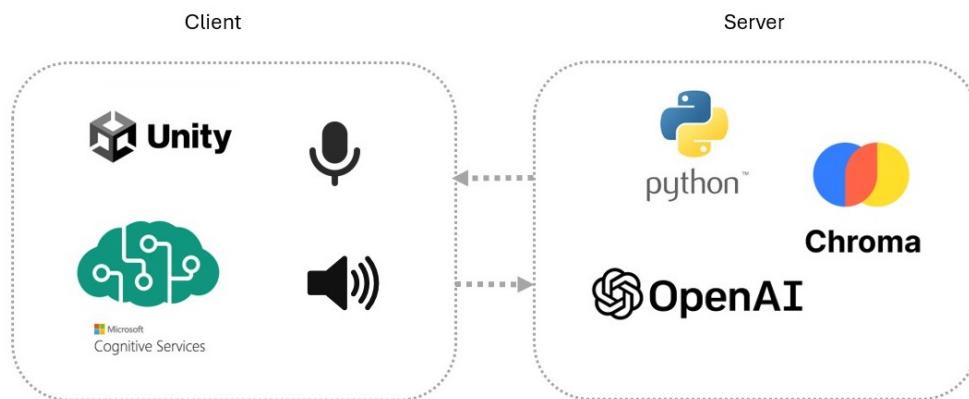


Figure 3.1: Tech stack of the chatbot feature.

Figure 3.1 shows the basic tech stack of the application which consists of a client compo-

ment deployed on the Hololens2 device and a server component which has the chatbot functionalities.

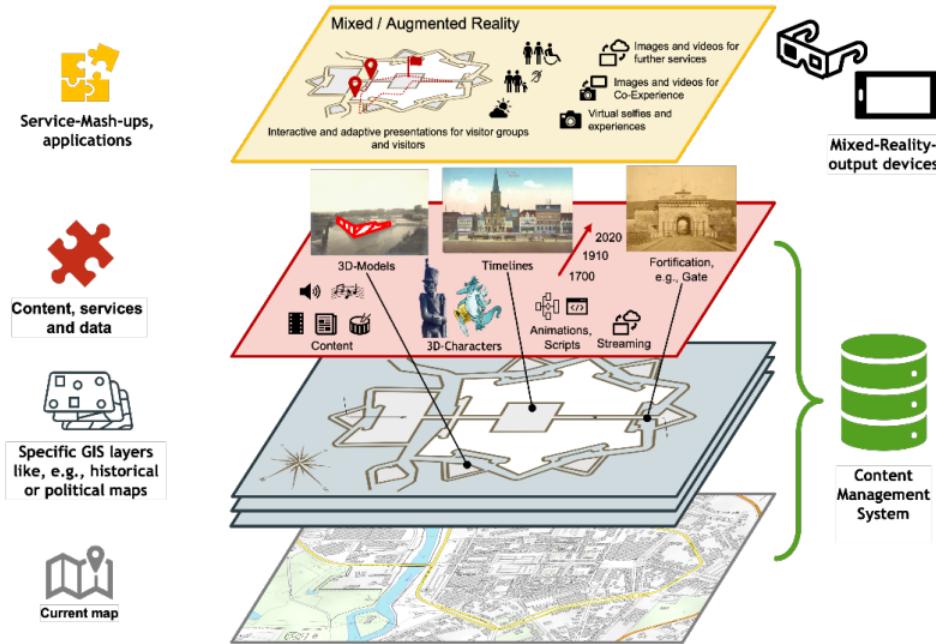


Figure 3.2: Technical architecture of the SaAR-Louis platform [12].

In the top layer, 'Service-Mash-ups, Applications', an interactive interface will be displayed to the user on the VR headset allowing them to ask questions and learn about the historical buildings of Saarlouis, presented as 3D models. As shown in the second layer of Figure 3.2, 'Content, services and data', the chatbot will have access to voice and speech capabilities, a database that is accessed and searched via embeddings containing information about Saarlouis, streaming (between the Hololens and the server) and the coordinates of the 3D models within the environment. The chatbot feature should rely on all of these layers but only the first two layers are realized in the current prototype. Because there are no GPS sensors on the Hololens itself, detailed geographic information isn't readily available without external APIs and spatial information is hard-coded using the location of the 3D models in relation to the user. Due to the limited resources of the Hololens2 headset, many of the articles in layer 2 are not stored on the Hololens2 itself but streamed or called from APIs such as the speech services, language model and database.

The overall pipeline is as follows, beginning with the creation of the database so the pdf documents containing information about Saarlouis can be put into a format that is accessible and fast for the LLM:

1. Parse unstructured documents (PDFs) into readable text.
2. Divide the extracted text into smaller chunks with some overlap in the tokens.
3. Convert the text chunks into embeddings using OpenAI embedding model.
4. Store the embeddings and documents in a vector database using Chroma.
5. Azure Speech-to-Text transcribes users' query which is then streamed from the Hololens to the Python server.
6. The query is converted to an embedding and the LLM decides whether database should be queried.
7. If the database is used, the cosine similarity between the query and the document chunks are calculated and the top-k results are returned.
8. The returned documents are inserted into the prompt and GPT-4 generates the response.
9. The GPT-4 response is streamed back to the Hololens2 and converted to speech using Azure Text-to-Speech.

The following sections go into more detail about the architecture of this pipeline. The current implementation is written in both Python3 (server) and C# (client) as shown in Figure 3.1 and allows two-way communication for sending and receiving data via a UDP socket [16].

### 3.1 LangChain Library

With the advent of LLMs, a new open-source framework, LangChain [49], has been introduced in order to streamline the process of applications that use them. LangChain provides easy to implement functionality in the form of agents (utilizing a language model to determine which action(s) to take) and tools (define the specific actions that an agent can perform) and chains (customizable use case-specific pipelines) [49]. Agents are the core architecture to process input and reason while tools are external resources such as APIs or even other LLMs. This allows us to abstract away from using a specific language model or database service. As there is much competition around LLMs right now, a better choice may be available in the future or a paid vector API service for the database may be more effective in later iterations. When using LangChain for creating the application the only thing that needs to be changed in the code are the API key(s) and the parameter specifying which model/service to use.

There are various pipelines available for QA-specific systems in LangChain and this thesis uses the 'ConversationalRetrievalQA' which preserves chat history when interacting with

an LLM. Furthermore, for every user query, it utilizes the query and the conversation history to create a standalone question. This is necessary in order to create an embedding for the query which can then be compared to the document embeddings. The top relevant  $k$  documents and the query are passed to the LLM to return a response. For condensing chat history and the current query, a faster and simpler model (i.e GPT-3.5-turbo) may be used and then a more expensive model (GPT-4) for answering the query itself.

Because low latency is also a priority in the application we must stream back the tokens to Unity as we receive them. This requires the streaming function in the LangChain library to be overwritten with a custom class to send the tokens via the socket connection. Visually, this also gives the effect of having the tokens appear one by one to the user.

## 3.2 Custom Knowledge Base

While we can probe the LLM's own knowledge to answer queries, LLMs often hallucinate facts. Here is sample output from an early prototype using GPT-3.5-turbo demonstrating this. The highlighted text contains hallucinated information:

Query: "Who are famous people from Saarlouis? Reply in 70 words or less."

Response: "Saarlouis has been home to several notable figures throughout history. One of the most famous is Michel Ney, a French military commander who was born in Saarlouis in 1769. He fought in the Napoleonic Wars and was eventually executed for treason. Other notable figures include the German writer and journalist, Hermann Kinder, and the German footballer, Kevin Trapp, who was born in Merzig but grew up in Saarlouis. Additionally, Saarlouis is the birthplace of several musicians, including the composer and pianist, Hermann Zilcher."

Because fine-tuning the actual weights of the model is not often practical and the parameters discussed in the previous section are insufficient for our scenario, many businesses and applications have implemented methods to supplement queries with context drawn from a custom, local knowledge base. Converting the documents to embeddings and storing them in a vector database is done as a pre-processing step and only needs to be done once. Individual documents can be modified or added later on if necessary without needing to rewrite the entire database. Then, the user query can be used to extract information from the database to find the documents that answer it.

To build the database, using LangChain library components, the raw text is extracted from the Saarlouis pdf documents. No text cleaning was done on the extracted strings as GPT-4 is also trained on unstructured text data so even if there are some formatting errors or miscellaneous characters within the text, the model should still be able to parse the text.

To store this unstructured textual data, first the documents must be split into smaller pieces, or chunks, as described in Chapter 2. The size for the chunks of text was 250 tokens with an overlap of 50. This number was chosen rather arbitrarily and gave decent

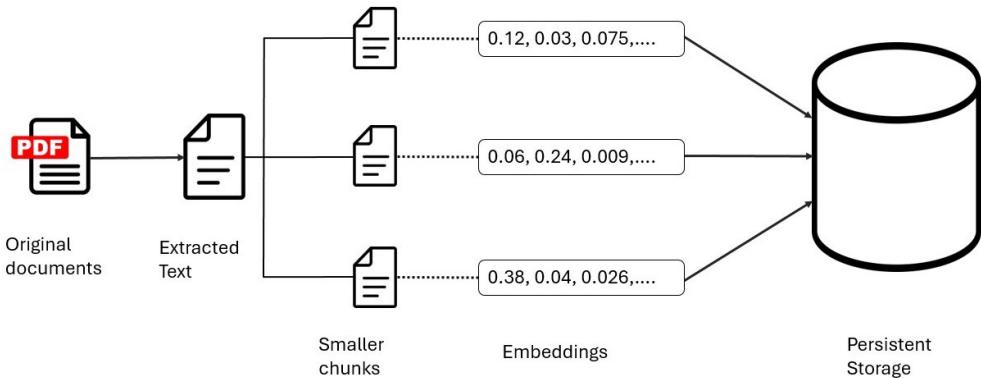


Figure 3.3: Database creation.

results but could experiment with in future studies to see if there is a more optimal chunk size. Most likely the chunk size would be larger in a more extensive database. Then, unique IDs are used to map the embeddings back to the documents along with optional metadata, for example with a dictionary object. The process is shown in Figure 3.3.

This thesis uses the open-source embedding database Chroma [1]. Like LangChain, Chroma offers high level abstraction, making it easy to implement. There is some important behind-the-scenes work the database does to actually make the storage and retrieval of these embeddings effective. This is because it is not efficient to simply store these high dimensional embeddings and perform linear search at time complexity  $O(N)$  to get the closest match to our query vector with our similarity metric. Chroma and other vector databases index the vectors by approximate nearest neighbor methods which is a way of organizing the text embeddings to facilitate faster search by only having to go through a subset of data, which is necessary in real-time applications. A common ANN method which Chroma also employs is Hierarchical Navigable Small World (HNSW) [37]. This algorithm organizes the data into a hierarchical graph-based structure where each layer in the hierarchy is a proximity graph that represents a different subset of the data [37]. Then the chosen similarity metric is used to navigate the graph. By default Chroma uses cosine similarity, this can be modified to other distance metrics but cosine similarity is fine for this use case.

Chroma has various functions to query and filter the database (with metadata) and can retrieve not only the relevant documents but view associated metadata and the distance scores. Vectors that are similar to each other will be closer in the vector space and hence lower scores represent higher similarity. For example, the query of 'Deutsches Tor' with a  $k = 2$  where  $k$  is the number of documents to return, returns the following information:

1. **Document:** Verbrauchspulvermagazine eingerichtet. Die beiden prachtvollen Toranlagen

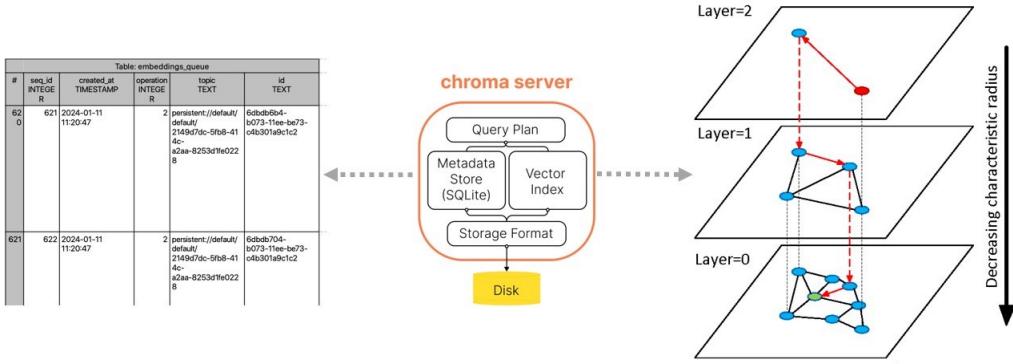


Figure 3.4: Architecture of Chroma database [1], with a snapshot of a table of meta data from the created SQLite3 file on the left and the structure of a HNSW graph index structure on the right [37].

*werden umgebaut und verstärkt, zunächst das Deutsche Tor im Jahr 1824, das Französische Tor folgt in den Jahren 1828/29. Die barocken Schmuckfassaden weichen einer sachlich, schlichten Gestaltung. Die Torhäuser werden bis auf die Höhe des Walls abgetragen. Gleichzeitig mit dem Umbau werden in den Kurtinen beiderseits des Deutschen Tores die Kasematten...*

**metadata:** 'page': 8, 'source': 'docs/Modul-2-2.2-Preuß-Festung-SLS.pdf'

**Score:** 0.26499634981155396

2. **Document:** *Absingen des Te Deums der preußische Adler an der Kommandantur angebracht.[17] Preußen baute die von Frankreich errichteten Festungswerke weiter aus und legten unter anderem umfangreiche Kasematten an. Im Prümer Zeughaustrum vom 18. Mai 1849 hatten sich demokratisch gesinnte Anhänger der Revolution von 1848 bewaffnet, um die Reichsverfassungskampagne militärisch zu unterstützen. Die Aktion in dem Eifelstädtchen Prüm war – wie der Iserlohner Aufstand und die weiteren Maiaufstände in der Rheinprovinz und anderen Teilen Preußens – eine Folge der Politik König Friedrich Wilhelms IV. von Preußen, der im...*'

**metadata:** 'page': 7, 'source': 'docs/Saarlouis-1.pdf'

**Score:** 0.34101513028144836

Finally, the database can be added to the application by a LangChain retriever tool. The agent uses the description parameter of the tool to decide if it is relevant for the current input. Thus, the tool is not necessarily used every time the model is called which helps speed up the response time if the LLM can answer the query based on previous conversation history or from its own world knowledge. For example, in the beginning the user may introduce themselves to the chatbot. Responding with a greeting and the user's name would not require a database search as the user's name is in the chat history

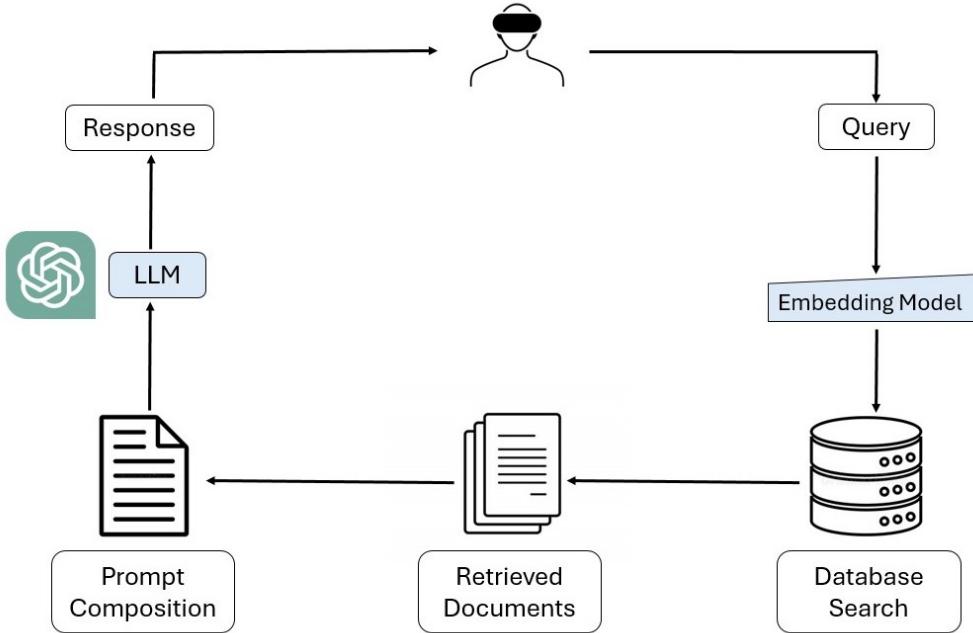


Figure 3.5: Query-response loop

and tool description is ‘Database containing information about Saarlouis’, so the tool would not be useful nor invoked.

The query is taken into context with the chat history is condensed and converted into a singular embedding to compare against the embeddings in the database. The top  $k$  documents are retrieved and inserted into the prompt along with the original user and any other variable (i.e chat history). This is sent to GPT-4 which returns our final response.

To summarize, the user query is transformed into an embedding with the same technique as for the text documents. A similarity search between query vector and the document vectors is done via cosine similarity. Then, when the top- $k$  vectors are retrieved, their textual counterparts are retrieved from the database and inserted into the prompt as context along with original user query. The complete process is shown in Figure 3.5.

The next part of the process is to write a prompt where all necessary variables are included and utilized effectively.

### 3.3 LLM Parameters and Prompting

LLMs are black box but there are still ways we can manage the input to tailor the LLMs' output, namely through the prompt and other parameters.

Early iterations of the prototype were developed with GPT-3.5-turbo as GPT-4 wasn't yet released and some examples in this section use GPT-3.5-turbo. However, for a conversational text-based QA system they have the same parameters we can alter to affect the performance.

For ChatGPT the important parameters are:

- Temperature: Controls the randomness of the model, and has a range between 0 and 1. A temperature of 0 is mostly deterministic while 1 is highly variable.
- Max tokens: The length of the output. According to OpenAI, 1,500 words are approximately 2048 tokens.
- Messages: This is an array of message objects which is context or the chat history. A message object contains a role and content. There are three sub-types of roles.
  - system: Sets up the context and behavior of the assistant.
  - user: Provides instructions to the assistant and is usually generated by the end user. However, initial prompts can be given beforehand.
  - assistant: Information in advance so that the API gives us the response we expect from the API.

Below is an example of the parameters in sample code of an early test using the GPT-turbo-3.5 model without LangChain abstractions to get a better idea of how OpenAI's GPT works. The order of the message object array with the sub-types will be [*system, user, assistant, user, assistant...*] because the first time the API is called the chatlog is empty, so it is initialized with the prompt instructing it to behave as a tour guide as well as brief context information about Saarlouis. Every time the API is called thereafter, both the user input and the response are appended to the array of message objects.

```

def generate_response(question, chat_log=None):
    if chat_log is None or not chat_log:
        chat_log = [
            {'role': 'system',
             'content': "You are a tour guide in Saarlouis. Do not mention you
                        are an AI language model.
                        Keep your answers precise
                        and to the point. Do not
                        give extra information the
                        tourist didn't ask about.
                        For context, Saarlouis is
                        town in Saarland, Germany,
                        capital of the district of
                        Saarlouis."},
        ]
    chat_log.append({'role': 'user', 'content': question+" Reply in 50 words or
                                              less."})
    response = completion.create(model='gpt-3.5-turbo',
                                  temperature = 0.3,
                                  max_tokens = 120,
                                  messages=chat_log)
    answer = response.choices[0]['message']['content']
    chat_log.append({'role': 'assistant', 'content': answer})
    return answer, chat_log

```

Unlike the web-browser version, the API call requires the user to update and provide the previous conversation to it for context every time. However, this array of message objects counts towards the token limit so with longer conversations it is necessary to pop messages off of the array. In addition, we provide text information from documents containing information about Saarlouis in order to get factually accurate responses. There needs to be a balance between context and token limit for optimal results. The amount of tokens allowed in memory is sometimes increased with model updates and also depends on which model is used (i.e GPT-4 has a longer memory). In LangChain we can also set these same parameters and specify which most recent  $n$  messages to include in the chat history parameter.

### 3.3.1 Prompts

So far we have covered how LangChain is used to streamline the vectorization of queries and LLM calls. We've also created a database to retrieve useful information and established the parameters available to us to guide the LLM to our use case. Now comes the actual prompt for the final LLM call to give the user a response and the integration of the database tool and parameters into it.

The prompt is composed of two parts, a prefix and suffix and later joined together with other necessary variables such as the database tool and the chat history. The prefix explains the personality type the chatbot should have, the tasks, and the tool(s). The suffix briefly reiterates a couple important points as with longer prompts sometimes the model forgets information when it falls in the middle of the prompt. The suffix

also includes the non-tool variables such as chat history, the user's own query, and the agent scratchpad. The agent scratchpad loads the tools' descriptions and stores intermediate values/steps used in the chain so the agent can use them properly and have continuity. There is also a parameter in which we can choose how many messages from the conversation history we want to save. In the scope of the tasks designed for the user study, not much conversational history is mandatory but in a more refined application we perhaps would want it much longer. Another option would be to create a tool for chat history as well so that we are saving the conversation either locally or on a server and do searches on it against some criteria such as if the user back to a question or response far back in the user-agent interaction.

```

89 # create llm
90 llm = OpenAI(
91     temperature=0,
92     model_name='gpt-4',
93     streaming=True,
94     verbose=True
95 )
96
97 # prompt
98 PREFIX = """Sie sind ein freundlicher, einladender und enthusiastischer Reiseleiter in Saarlouis, Saarland, Deutschland. \
99 | Seien Sie fröhlich und hilfsbereit, aber geben Sie bitte keine konkreten Empfehlungen zu Hotels und Restaurants. \
100 |
101 Manchmal erhalten Sie anstelle einer Frage eine Richtungsangabe, wo sich ein wichtiges historisches Gebäude befindet, \
102 wie zum Beispiel: 'Vor Ihnen ist der Große Markt', 'Rechts/Links von Ihnen...' 'Hinter Ihnen...'. \
103 Behandeln Sie in diesem Fall die Anfrage nicht als eine Anfrage des Benutzers, sondern eher als einen Auslöser, \
104 um kurze historische Informationen über das spezifische Gebäude bereitzustellen. \
105 Halten Sie Ihre Informationen kurz und Sie MÜSSEN diese Richtungsinformationen am Anfang Ihrer Antwort angeben, \
106 damit der Benutzer weiß, wo er suchen muss. Zum Beispiel: 'Vor Ihnen ist...'.
107
108 Wenn die Datenbank keine relevanten Informationen enthält, \
109 können Sie auf der Grundlage Ihres Weltwissens eine rationale Antwort geben. \
110 Wenn Sie die Antwort nicht kennen, sagen Sie einfach, dass Sie es nicht wissen, \
111 und versuchen Sie nicht, eine Antwort zu erfinden. Halten Sie Ihre Antwort kurz und präzise. \
112
113 Sie haben Zugriff auf folgende Werkzeugen:"""
114
115 SUFFIX = """Beginnen! Denken Sie daran, als begeisterter Reiseleiter für Saarlouis zu antworten. Antworten Sie auf Deutsch.
116
117
118 Vorheriger Gesprächsverlauf:
119 {chat_history}
120
121 Neue Frage: {input}
122 {agent_scratchpad}"""
123

```

Figure 3.6: LLM parameters and prompt for the user study.

The English translation of the prompt can be found in the appendix.

### 3.4 Spatial Awareness

The last information needed to supplement our call to GPT-4 are spatial references as described in Section 2.5. When a user is within a certain distance threshold to a 3D-model, information about the model and where the user should direct their attention to is triggered. For example 'if you look to your left', or 'behind you', or 'in front of you'. These expressions are chosen from calculations using the vectors of the user's gaze, which is represented by the main forward-facing camera mounted on the Hololens headset, and the anchor points of the models. Once the user is within a certain distance

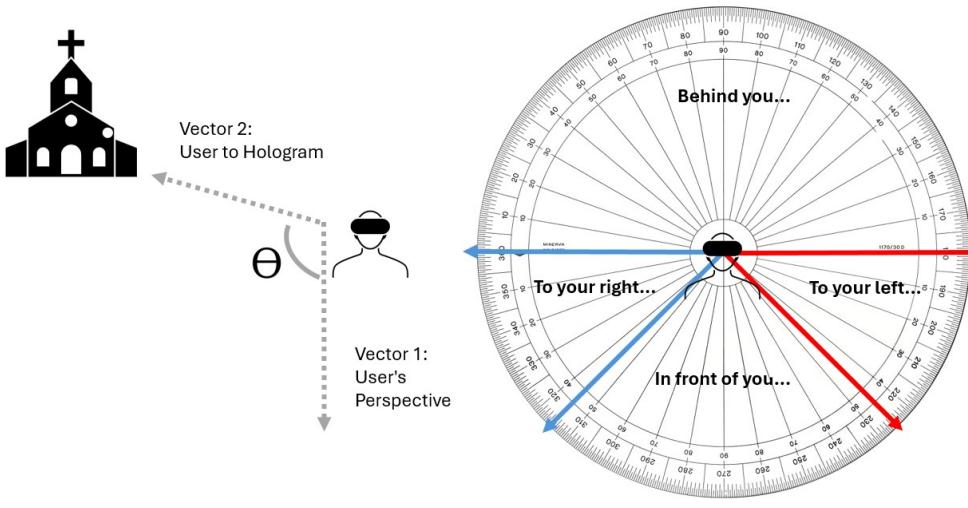


Figure 3.7: Referring expressions generated by the user's distance and direction to a model of interest.

threshold of the hologram (i.e 1 meter) we can measure the angle from the user to the object on the x-axis and give an expression based on a hard-coded threshold. Using this information we can determine the angle thresholds we want to give the referring expression directions. This is shown in Figure 3.7.

As noted for word embeddings, the dot product is used to measure how similar words may be and for spatial applications it tells us which direction the user is looking in relation to the object of interest. Thus, using the dot product and x-coordinates of the models we can tell if the model is out of frame and if the user should look right or left.

There are also eye tracking features available which may be used as it can detect when the user's gaze hits an object of interest but for the user study scenario a custom, simpler script was written by calculating the above vectors.

As shown in Figure 3.6, the angles correspond to textual spatial information which is sent automatically to the chatbot without the user voicing a query to deliver them context information once they reach a certain hard-coded distance threshold of a model. The downside to this is that the user cannot utilize the chatbot while this query is being processed. While the rule-based approach is satisfactory for simple instructions based on the user's positioning, a more interesting and complex solution is to somehow integrate larger spatial understanding into the chatbot's knowledge in the form of a tool or another add-on, thus having a more 'end-to-end' model (though less controllable) rather than a machine learning tool relying on various discrete rule based modules. In such a scenario the user could simply walk up to a building and ask "Which building is this?" and get

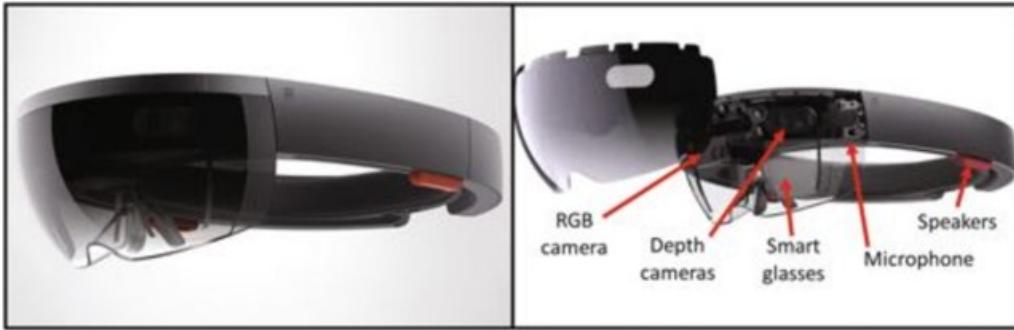


Figure 3.8: Hololens2 hardware overview [8].

the answer rather than accidentally triggering information they don't want or having to wait to use the chatbot feature.

Unfortunately, as mentioned there are no GPS sensors on the Hololens2 nor other VR headsets on the market. However, geolocation may be fetched from the 'Windows.Devices.Geolocation' namespace APIs [2]. In a mobile scenario location accuracy is often a problem though 5G capabilities may mitigate this. Another option is to stream the data from a nearby bluetooth device.

A further extension of the spatial awareness system should include the information shown in the bottom two GIS layers of Figure 3.2. This could include information such as giving directional information on more than one building at a time with street names, current construction and traffic warnings, and detecting if other buildings or objects are occluding the point of interest from the user's view, akin to a map navigation system like Google maps. It may be necessary to employ a web-scraping step that regularly compiles the newest relevant information about Saarlouis to store in the database. The application should have the geographic layout of both old Saarlouis and current Saarlouis. The components may be incorporated as other tools for the LangChain agent to use.

### 3.5 Speech-to-Text and Text-to-Speech

Hololens and other mixed reality technologies contain hardware for speech and audio input and having a voice input query to audio output response flow is important for our scenario. The Hololens2 is equipped with a 5-channel microphone array and spatial audio speakers [3] shown in Figure 3.8. While the Hololens2 has some built-in voice recognition capabilities, they are limited to short phrases and specific commands. Because we want to create a scenario where the user can talk naturally with a voice assistant we utilize Microsoft's Azure Speech-to-Text API to transcribe user's speech. This returns a string which we can then use as the user query to send to the chatbot. Azure is closed source so

no details can be provided on the architecture or datasets used to train the model. Azure allows its users to customize the model by adding domain-specific vocabulary to a list. As soon as the speech input has been recorded it is streamed to the Python server where it is incorporated into the prompt and sent to the LLM.

The TTS component is more complicated because the off-the-shelf TTS service from Azure requires a complete input sequence in order to synthesize the speech. Although the tokens are streamed back from the server one-by-one we can't directly input each token for synthesis as OpenAI tokens are not necessarily full words but word chunks. Thus, we need to process the incoming text stream response and find the word boundaries denoted by spaces and punctuation and feed these into the model. Too small chunks would result in unnatural sounding speech but too large chunks may result in delays.

---

---

## **Chapter 4**

### **User Study**

Due to the source documents in the database being in German, the best way to test the prototype is with German queries to mitigate confounding variables that might arise when comparing an embedded English query against the German document embeddings as well as GPT translations. However it may be noted that both Microsoft Cognitive Services and GPT-4 have multilingual capabilities and the project can easily be adapted to other languages. It is possible to have a prompt in English and request GPT-4 to respond in a different language. Though, to ensure the most reliable performance, all prompts and questions were translated into German. In order to expand this thesis into other languages the knowledge database should be professionally translated into the desired languages.

The user study seeks to get information for the following points:

1. General opinion of the current prototype whether the voice input correctly understands their question, and if the chatbot gives satisfying and factually accurate output.
2. If the voice input correctly understands proper nouns related to our project ('Saarlouis', 'Ravelin', 'Saarland', etc.). If misheard, how does the chatbot handle/correct these instances?
3. See if chatbot can refer to past questions and resolve pronouns. I.e. "What is this building?" versus "What is the St. Ludwig's Church?"
4. Have the chatbot direct participants with directional/spatial information. "To your left..."

5. Database usage and hallucination cases
6. Cases where the chatbot fails to adhere to the prompt

Several points are evaluated post-study by checking if the answers given to the user are factually accurate and examining which documents the model pulled from the knowledge base, if at all.

All user input and model output was saved out to text files during the user study, including some internal states unseen to the user, allowing us to examine how the system condenses prompts and when it decides to query the database.

## 4.1 Participants

Participants must have very good knowledge of English and German. There were no other limiting factors.

Eleven users participated in the study in total. All were native German speakers between the ages of 22 and 59 with a median age of 29. Only one participant had never used a VR headset, four had only used one very minimally and the rest had multiple experiences with VR technology in both application development and usage.

## 4.2 Setup and Design

The participants filled out the standard consent form and were given a summary of the applications and their tasks in English. The users did a short tutorial application in order to get used to basic Hololens2 and application functions such as pressing buttons and issuing speech commands '*Aufnahme*', '*Weiter*' ('Record', 'Next'). After this the main application was started where users visited the three models displayed at the edges of the lab space to get information about them by interacting with the chatbot.

Once the users are satisfied by the elicited information from the chatbot they can move to the next task by clicking a 'next button' on the task menu window. After the final task is completed they press 'finish' and the headset was removed. This was followed by a post-survey where they answered questions about their opinions of the user experience of the application as well as how friendly and informative they found the answers to be and whether or not they experienced trouble with the speech recognition system. All documents mentioned above can be found in the appendix.

The tutorial application involved practicing speech commands as well as adjusting the position of the chat window via gesture in order to more smoothly navigate the main application. If users struggled with the tutorial application they were allowed to repeat it to become more comfortable with the setup. Figure 4.1 is a screenshot of the tutorial

interface in Unity. The actual tutorial application was deployed on the Hololens2 like the main application.

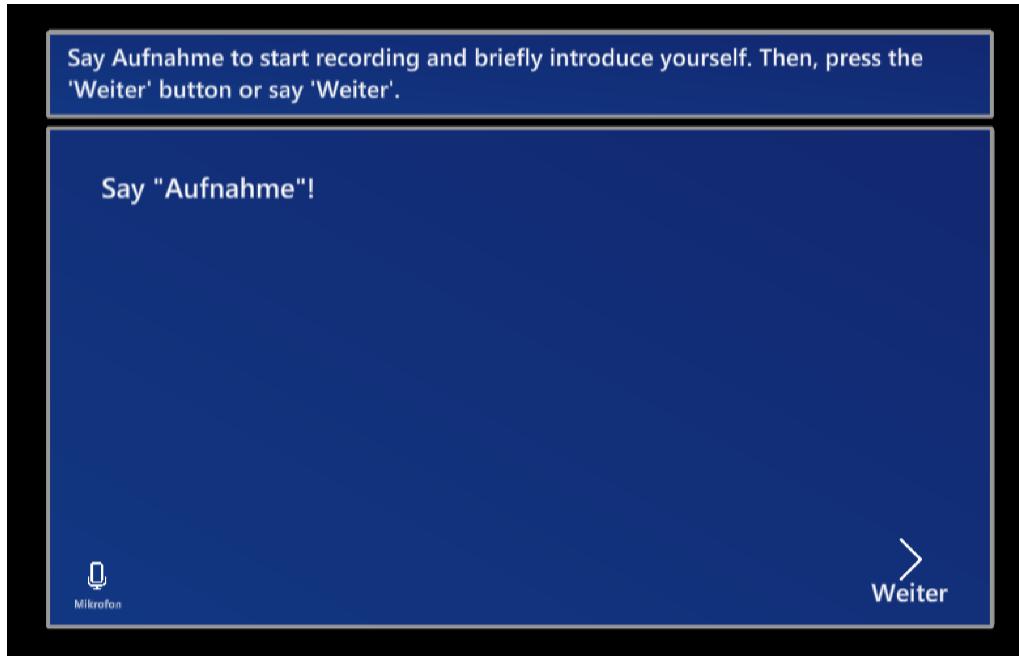


Figure 4.1: The tutorial familiarizes the user with the command for recording 'Aufnahme' or its corresponding button 'Mikrofon' and for navigating to the next task(s) ('Weiter').

The user study had participants interact with the German language chatbot in a toy application that contained shrunk down models of three historical Saarlouis buildings: St. Ludwig's Church (St. Ludwig), the French Gate (Französisches Tor), and the German Gate (Deutsches Tor). Unity layouts of the models are shown in Figure 4.2 and Figure 4.3. The experiment was conducted in a lab that consisted of a roughly 4m x 6m space as shown in Figure 4.4.

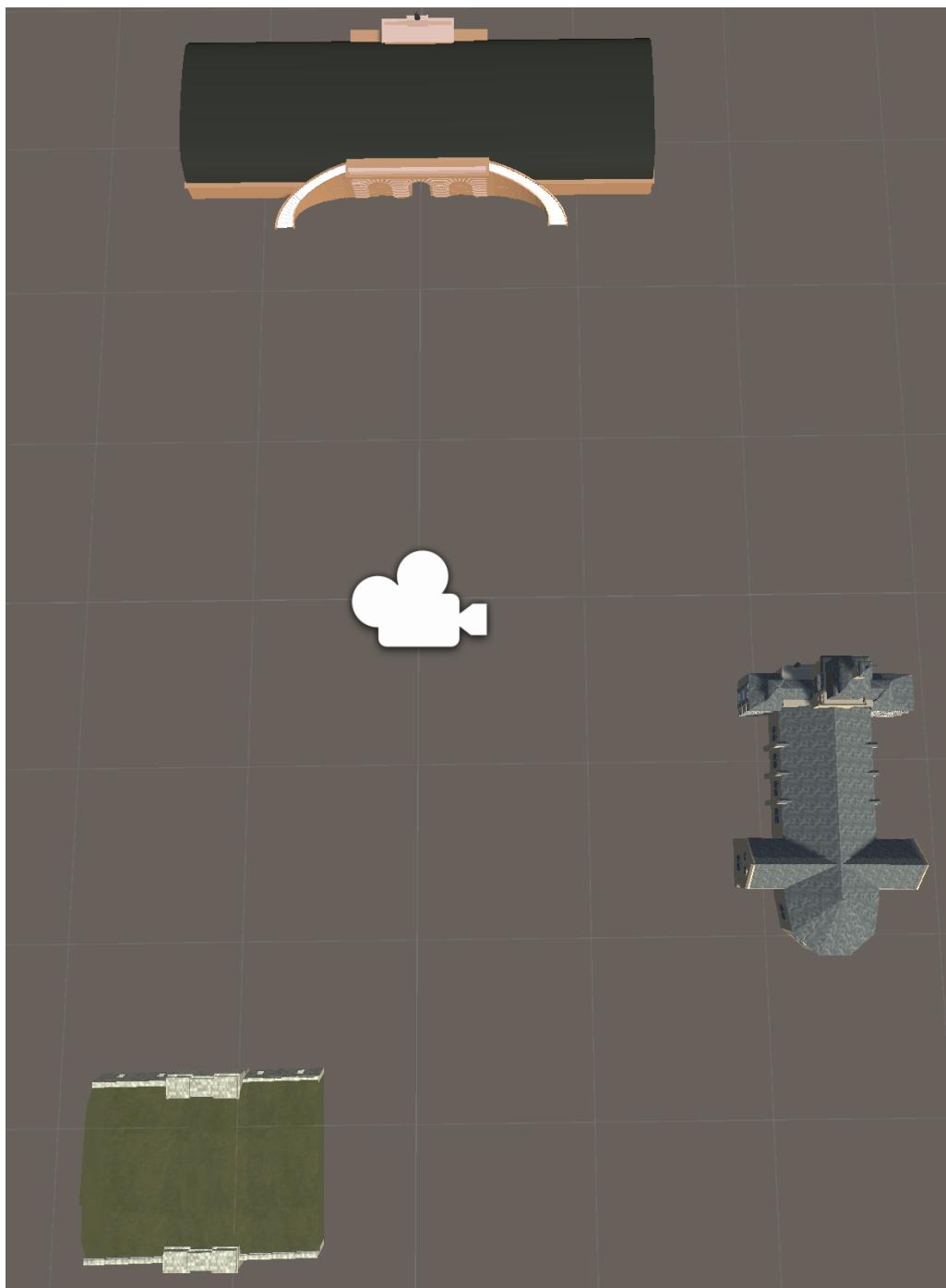


Figure 4.2: User study layout in the Unity 3D scene.  
Top: (French Gate) Französisches Tor, Right: St. Ludwig's Church (St. Ludwig), Bottom:  
German Gate (Deutsches Tor), Camera: User starting point.

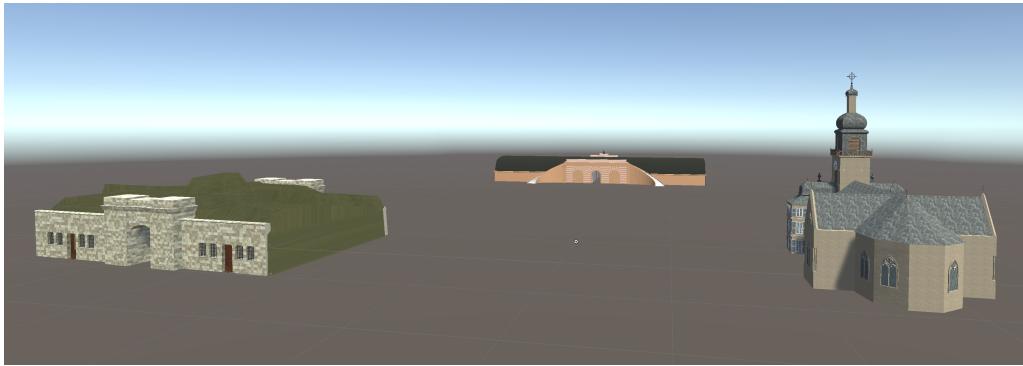


Figure 4.3: Another view of the 3D models in Unity.  
Left: German Gate (Deutsches Tor), Center: French Gate (Französisches Tor), Right: St. Ludwig's Church (St. Ludwig)



Figure 4.4: Deployment of the application in the lab space when deployed on the Hololens2.  
The red box shows the starting point of the user where they begin facing the French Gate so the models load in the space consistently across participants as there were no anchor points to fix the models, thus their position is based relative to the Hololens2 headset camera.

Each building came with a different task, which is shown in the Table 4.1.

The tasks were designed to be somewhat open-ended with the goal of eliciting different questions from users that should still get the same answers from the chatbot. To keep it

Building	Task (German)	Task (English)
Französisches Tor	Fragen Sie nach dem Zustand des Gebäudes.	Ask about the current condition of this building.
Deutsches Tor	Fragen Sie nach der Rolle des Gebäudes während der preußischen Zeit.	Ask about the role of this building during the Prussian period.
St. Ludwig	Fragen Sie über die Architektur oder nach dem Architekten.	Ask about the architecture or the architect.

Table 4.1: User tasks.

consistent all tasks used the same verb 'Fragen' (to ask). As pronoun resolution was also a variable of interest, the building names were not mentioned in the tasks in the hope that it would elicit general nouns and pronouns to refer to the building by the user and that the chatbot could understand what is being referred to based on chat history.

Before asking questions related to the task, users were told to approach a building of their choosing. The chat window appears based on a hard-coded distance threshold (approximately .85 meters) when the user is close enough to a model and this triggers directional and summarized historical information about the building such as in Figure 4.5.

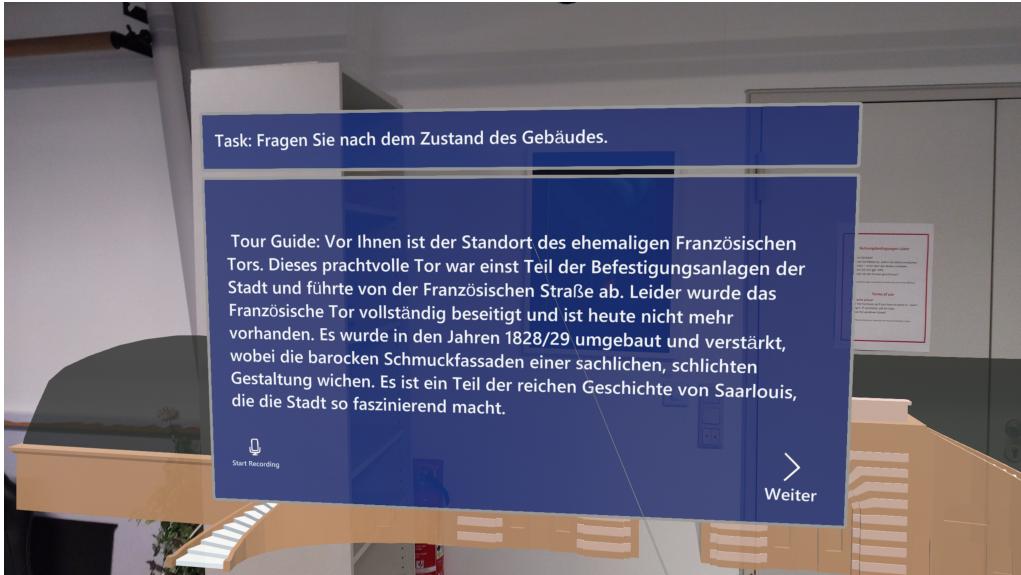


Figure 4.5: Example of the preliminary information given about the Französisches Tor.

After the user reads this information they may ask a question and follow up questions to the task. There was no limit on the amount of questions they could ask. Sometimes the information regarding the task was already given in the automated summary. Users

were still instructed to follow the task whether this information was provided. The user's transcribed queries and the chatbot's internal states and responses were saved to individual log files for each participant.

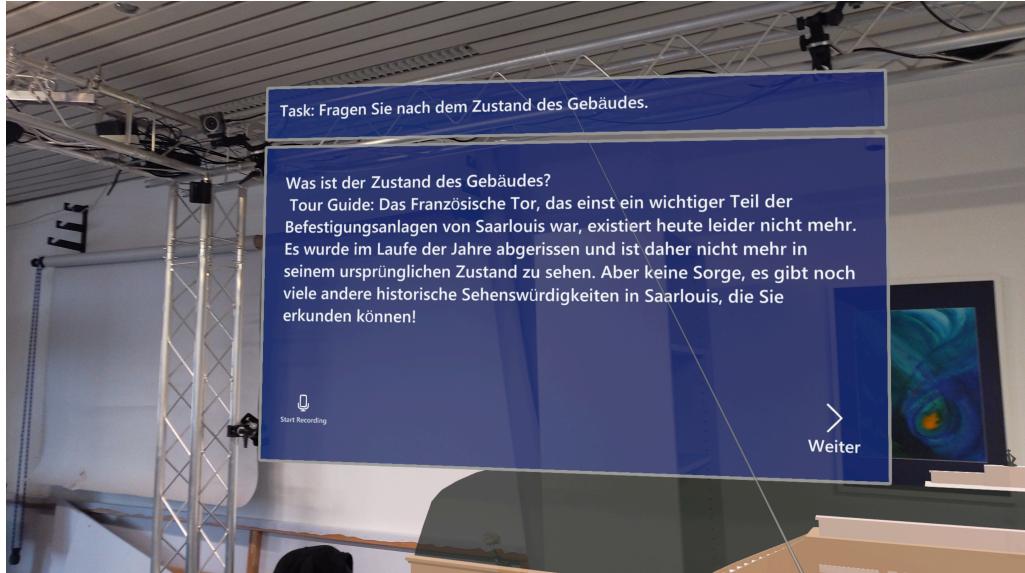


Figure 4.6: User query and chatbot answer example.

### 4.3 Results

Usability and functionality of the application was assessed using a five-point Likert scale with one being 'Strongly Disagree' and five being 'Strongly Agree'. Table 4.2 below summarizes the Likert scale questions.

Variable	Mean	SD	Mode
I found the headset comfortable to wear.	4.36	0.77	5
I could perform all the functions I desired.	3.81	1.24	5
I found the chatbot's responses informative.	4.54	0.43	5
The chatbot's responses were fast.	4.18	0.51	4
I found the chatbot to have a friendly personality.	3.63	0.59	3
The chatbot's responses were grammatically correct.	4.9	0.08	5

Table 4.2: Descriptive statistics over post-user study survey.

In addition, users were asked how many times, if any, they had to repeat themselves with the choices being 'never', 'once or twice', and 'three times or more'. The majority, 45.5% responded 'never', 4 participants (36.4%) responded 'once or twice' and 2 participants (18.2%) responded 'three times or more'. This is shown in Figure 4.7.

#### I had to repeat myself to be understood by the speech recognition system.

11 responses

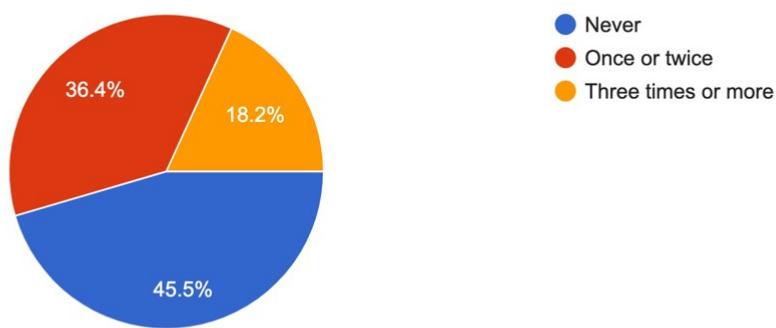


Figure 4.7: Amount of repetitions for the ASR system.

In a follow up free response question they were asked to specify which word(s) were misunderstood. Finally, there was a free response field in which users could give feedback, concerns, or criticism.

Post-study the output logs from users were analyzed and queries were grouped into two categories: database tool used and no database used. User queries that were not a question (for example a user's personal opinion or a statement) were removed from the dataset. The chatbot's response to user queries were then coded as being accurate, partially accurate, not accurate, or not available. Not available includes instances where the chatbot was unable to answer the given query but also expressed this, i.e admitted it didn't have enough information to give an answer. Times when the user went off-topic or gave non-questions/non-commands to the chatbot (such as expressing their opinion or commenting on something) as well as accidental recordings were removed from the dataset. Responses where the chatbot did not adhere to the prompt such as 'breaking character' were also noted but not marked as inaccurate unless the response contains

false information.

1. Correct = All information in the chatbot response was factually accurate.
2. Not Accurate = Answer is completely wrong or irrelevant to the given user query/input.
3. Not Available = The chatbot states it didn't have enough information to answer the query.
4. Other = Answers that were a mix of above and/or the information couldn't be verified for factual accuracy.

It is also important to note that while there is a substantial amount of answers labelled 'NA', this can be considered a positive because minimizing hallucinations is more important than giving a false answer and shows that the LangChain agent created can recognize when there is not sufficient information available to respond to a query. Moreover, the amount of NA's can easily be reduced by adding more documents to the database with the needed information. The coding is qualitative rather than quantitative due to the observational nature of the study so it is difficult to give concrete percentages on 'how accurate' the chatbot is. Thus, the data annotation is designed to give better insight into what works well and what went wrong and why. A future study with multiple annotators and specific criteria can examine accuracy in a more quantitative nature by having fixed questions compared to a gold standard.

Table 4.3 contains an example of each code.

Query	Answer	Label
Wie viele Fenster hat die Kirche?	Es tut mir leid, aber ich konnte keine spezifischen Informationen darüber finden, wie viele Fenster die St. Ludwig in Saarlouis hat. Es ist jedoch bekannt, dass die Kirche ein beeindruckendes Beispiel für moderne Kirchenarchitektur ist und viele einzigartige architektonische Merkmale aufweist.	NA
Ich würde gerne etwas über die Architektur dieses Gebäudes wissen.	Die Kirche ist ein beeindruckendes Beispiel für moderne Kirchenarchitektur. Das heutige Kirchenschiff wurde vom renommierten Architekten Gottfried Böhm entworfen und in Sichtbeton ausgeführt. Böhm ist bekannt für seine skulpturale Architektur und die Verwendung von Beton als Hauptmaterial. Die Kirche hat eine ungewöhnliche Form, die an ein Schiff erinnert, und ist innen mit modernen Kunstwerken geschmückt. Der Altar, der Tabernakel und das Taufbecken wurden ebenfalls von Böhm entworfen. Die ursprüngliche Barockkirche wurde im 19. und 20. Jahrhundert stark umgebaut, aber einige Elemente, wie der Turm, sind noch erhalten.	P
Welche Rolle hat das Gebäude in der preußischen Zeit gespielt?	Das Deutsche Tor spielte eine wichtige Rolle in der preußischen Zeit. Als Saarlouis am 1. Dezember 1815 an Preußen übergeben wurde, befanden sich die Festungsanlagen, einschließlich des Deutschen Tors, in einem sehr schlechten Zustand. Die Preußen erkannten jedoch die strategische Bedeutung der Festung und setzten sie in den folgenden Jahrzehnten mit großem Aufwand instand, modernisierten sie und erweiterten sie. So rückten beispielsweise preußische Soldaten durch das Deutsche Tor in die Festungsstadt ein. Es diente weiterhin als wichtiger Zugangspunkt zur Stadt und war ein zentraler Teil der preußischen Verteidigungsstrategie.	Y
Was findet man heute in dem Gebäude?	Heute wird das Deutsche Tor vielseitig genutzt. Es beherbergt verschiedene Unternehmen und dient als Veranstaltungsort für kulturelle Events. Die genaue Nutzung kann sich jedoch im Laufe der Zeit ändern. In der Vergangenheit war es zum Beispiel einmal Standort einer Lebensmittelgroßhandlung. Es ist ein lebendiger Teil der Stadt Saarlouis, der seine historische Bedeutung bewahrt hat, während er sich den modernen Anforderungen anpasst.	N

Table 4.3: User output examples of the labelled data.

The reasons for the labels in Table 4.3 are as follows:

**NA:** Chatbot apologizes for not knowing information and gives some generic information related to user query.

**P:** The information returned about Vauban is accurate but it doesn't answer the user query and there is a church architect (of the more recent brutalist version) mentioned in the documents that the LLM failed to retrieve: Gottfried Böhm. It also mentions it 'searched through documents' which goes against the rules specified in the prompt.

**Y:** Answered question accurately using documents from database.

**N:** This answer refers to a different building, Barraks X, not the Deutsches Tor which is mostly destroyed and now the remaining is a tourist attraction.

Overall there were 133 queries and responses including the initial context with spatial information given to the user when they first approach a building which triggers with a hard-coded query rather than a user-given one. Out of the 172 responses, the database was invoked 75 times (56%) and 58 times (44%) without. On average, users asked roughly 3 questions per model.

The pie charts in Figure 4.8 show the proportions of categories for responses, divided into whether the database was called by the LLM or not.

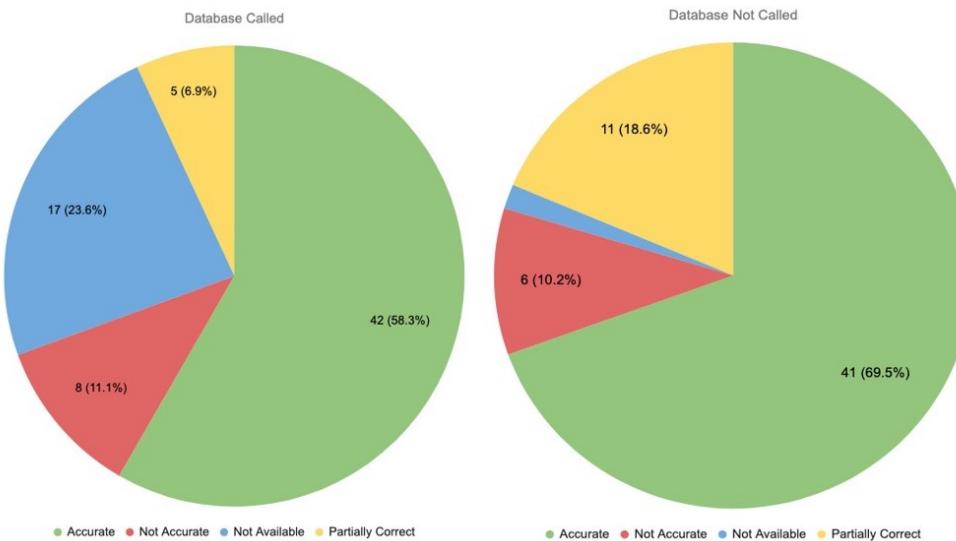


Figure 4.8: Pie charts showing the labels for chatbot responses with and without database usage.

When using the database, more answers were labelled as 'Not Available' and less as 'Partially Correct' compared to responses that did not use the database. The queries

answered without the use of the database had more responses labelled as accurate but the amount of responses of not making use of the database is less (58 versus 75). Only 1 query in the 'Database Not Called' was labelled as being 'Not Available'.

---

---

# Chapter 5

## Discussion

This chapter analyzes the results of the user study to examine what went wrong and identify possible ways to modify the implementation to improve results in the future. The discussion begins with the user's preferences and feedback and then moves on to more technical workings of the chatbot itself.

### 5.1 User Experience

The feedback and descriptive statistics as shown in Table 4.1 portray a largely positive user experience. Most of the users' criticisms had to do with the challenges of using a Hololens2 headset and the chat window obscuring the view of the building models. From Table 4.1 it is clear that the lowest score was the chatbot's personality/friendliness but no users provided more information of their rating for this feature so it may be something to examine in a further study. However, users also became more comfortable with the speech commands and navigation throughout the study and the TTS component was implemented post-study, thus in future iterations the chat window would be replaced by an audio response, and remain an optional visual addition. This feature may help alleviate issues of discomfort and model visibility.

The free response question at the end of the survey for users contained praise such as:

1. *Interesting application to learn more about old buildings. It was easy to use and navigate. The answers were most of the times helpful and if there was no answer it communicated it very well.*

However, there were some criticisms:

1. *Sometimes the chatbot seemed to information dump a bit. It gave a lot of additional information each time I asked a question. I don't necessarily think that's a bad thing but I imagine that it could be annoying to some users.*
2. *The bot is very good feature if you have not a lot further information. But if you have insight information and ask more detailed the bot's answer was not correct.*

While useful at the same time criticism is subject; one user would like more information while the other found it to be too much. One solution to adapt the application to each user's preference is to have the amount of information returned by the chatbot a feature the user may set and which will be realized within the prompt by an additional line telling approximately how many words to return per response or with descriptors such as 'the user desires informative, detailed responses' versus 'Keep your responses short'. Additionally, users may also voice these preferences in their own queries.

## 5.2 Speech Recognition

Most troubles encountered by participants with the ASR system had to do with the speech commands in order to begin recording, '*Aufnahme*', or navigate the windows '*Weiter*'. This may have happened because participants who were unfamiliar with VR systems and their commands were overemphasizing the words instead of speaking naturally. Although there is an indication on the screen after the speech command has been said (chat window displays the phrase 'Recording...say something!') sometimes participants hesitated or spoke too slowly for the ASR system to record their entire query. However, most participants improved with the speech commands throughout the use of the application and most of these errors can be attributed to unfamiliarity with the system. There were a few instances of proper nouns that were incorrectly transcribed such as '*Pritzker-Preis*' (Pritzker prize) and '*Lacroix*'. The two other failures of the ASR system were '*befand*' (to be located) recognized as '*fand*' (found) and '*Stil*' (style) as '*stiehl*' (steal).

The unrecognized but relevant words such as '*Lacroix*' may simply be added to the custom phrase list for the Azure speech model to improve the recognition in the future.

## 5.3 Queries, Responses and Database Usage

Once again, it is important to reiterate that ultimately GPT is a black-box and only so much can be controlled which will always give room for error even if the correct information is available in the database. This is particularly true for low-latency applications that may sacrifice a decrease in accuracy for speed, such as the use of vector indexing to

search just a subset of the data rather than performing linear search every time to get the true embeddings that are most similar to the query.

The inherent fluctuations can be seen in Table 5.1, where there was the same query across three different users yielding different sources retrieved from the database and responses from the chatbot. For two of the users the database was queried while another was not queried at all. Interestingly, the chatbot response without the database call was accurate, as well as the second query with a database invocation, while the third query failed completely.

This is also a good example of how quickly the quality of GPT may change as the ninth participant (inaccurate answer) was tested the same day as two other participants who were earlier in the day and their same questions were accurate whereas it failed for the ninth participant.

The reason for the inaccuracy has to do with the fact that the database embeddings are fixed and do not change unless the database is deleted and recreated. However, OpenAI makes constant tweaks and changes to the models and thus the database embeddings can become ‘out-of-sync’ to the ones in the model. One method to fix this is to schedule regular updates to the database embeddings so they better represent the current state. As the current database is small, it would even be possible to do this every time the program is started though but otherwise regular intervals are more realistic. It should be noted that the study duration lasted two weeks and the database was updated right before beginning the study and not updated during this period.

The chunks of database text themselves can also be modified instead of stripped directly from the source documents with no further processing. In a few instances information was retrieved from the database in which the text contained facts about a building but the name of the building itself wasn’t mentioned. This lead to the LLM’s (false) assumption that it was the building the user was currently asking about as seen in Table 5.2.

Chunk size may be increased to capture more information. Another option is to create a summary of the database document using an LLM prior to embedding and add this summary to each chunk so it is seen more as ‘part of the whole’. The title of the main document may also be added to the beginning of the chunk. Chunks may be divided not on size but based on content subsections within a document, preserving their titles/headings as metadata. It may be desirable to have a list of the most relevant entities at the beginning of a chunk by using named-entity recognition (NER) technique pre-processing. Another option is to manually compose hypothetical queries and embed them along the documents that would answer the question. Doing some text processing before embedding the chunked text may also help provide more accuracy. Unfortunately there is no universal best retrieval-augmented generation approach as this area is an emerging technology and much more research is needed.

Regardless, a combination of the above techniques may be especially useful for build-

User Query	Database Query	Database Sources	Response	Accurate
Wie ist der Zustand des Gebäudes?	-	-	Leider existiert das Französische Tor heute nicht mehr. Es wurde im Laufe der Jahre abgerissen und ist nicht mehr Teil der Stadtsilhouette von Saarlouis. Aber keine Sorge, es gibt noch viele andere historische Sehenswürdigkeiten in Saarlouis, die Sie erkunden können!	Yes
Wie ist der Zustand des Gebäudes?	Zustand des Französischen Tors	Saarlouis-1.pdf, page 14	Das Französische Tor, wie es ursprünglich gebaut wurde, existiert heute leider nicht mehr in seiner ursprünglichen Form. Es wurde im Laufe der Jahre vollständig beseitigt. Aber keine Sorge, es gibt noch viele andere historische Sehenswürdigkeiten in Saarlouis, die Sie erkunden können!	Yes
Wie ist der Zustand des Gebäudes?	Zustand des Französischen Tors	Modul-2-2.1-Franz-Festung-SLS.pdf, page 6	Das Französische Tor ist in einem guten Zustand. Es wurde in den Jahren 1828/29 umgebaut und verstärkt. Die ursprünglichen barocken Schmuckfassaden wurden durch eine sachliche, schlichte Gestaltung ersetzt. Die Torhäuser wurden bis auf die Höhe des Walls abgetragen. Es ist also gut erhalten und ein beeindruckendes Zeugnis der Festungsarchitektur.	No

Table 5.1: The information in the database is actually referring to Barracks X (a different historical building in Saarlouis) and not the German Gate.

User Query	Database Query	Database Sources	Response	Accurate
Was findet man heute in dem Gebäude?	Deutsches Tor Nutzung heute	Saarlouis-1.pdf, page 16	Heute wird das Deutsche Tor vielseitig genutzt. Es beherbergt verschiedene Unternehmen und dient als Veranstaltungsort für kulturelle Events. Die genaue Nutzung kann sich jedoch im Laufe der Zeit ändern. In der Vergangenheit war es zum Beispiel einmal Standort einer Lebensmittelgroßhandlung. Es ist ein lebendiger Teil der Stadt Saarlouis, der seine historische Bedeutung bewahrt hat, während er sich den modernen Anforderungen anpasst.	No

Table 5.2: Example of retrieving incorrect documents from database.

ings such as St. Ludwig's Kirche which has seen many reconstructions from different architectures and styles over throughout history. Results from the user study showed responses that sometimes only mentioned the original St. Ludwig's Kirche, the current one, or mixing up facts between the two. For every text chunk that discusses the church we could specify via additional text which version of the church it is, i.e 'original Baroque building in 17th century', 'Neo-Gothic building by Carl Friedrich Müller and Vincenz Statz in 19th century', 'Current brutalist building by Gottfried Böhm in 20th century.'

However, as mentioned previously it is important that the returned chunks do not go over the context token limit of the LLM. Either way regular database maintenance is a key component to generating accurate responses.

Another interesting thing to note is how the LLM is able to resolve referring expressions when the user is vague by integrating the chat history into a reformulated query (via LLM call) that can then be embedded and compared against the database. The reformulated query is not shown to the user but is an internal state that was printed out into the logger files for the purpose of the user study. An example of the noun phrase resolution is given below in Figure 5.1 where the user does not specify building name in their query. The LLM accurately infers they are talking about the French gate and inserts this into the reformulated query. Documents from the database are retrieved and all of this information is sent to the main prompt to give the user the chatbot response. Note that in this example the exact answer was not available in the LLM knowledge base or the

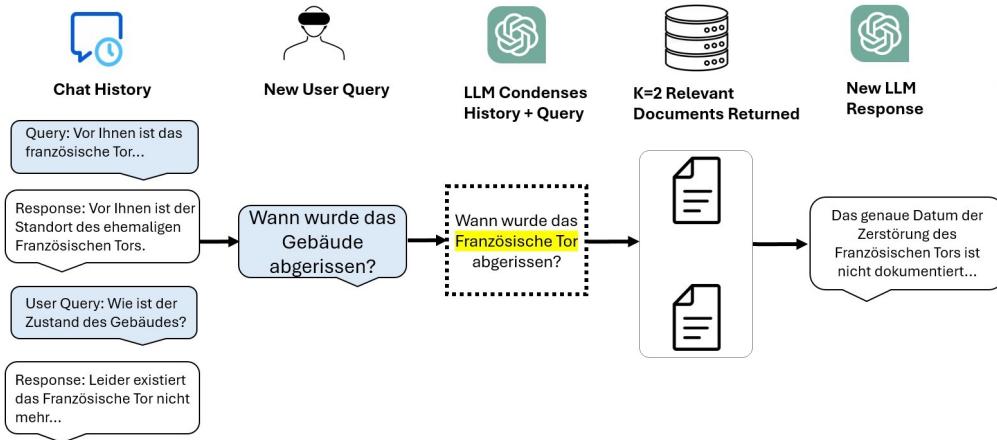


Figure 5.1: Chat history and new query condensing to get new query to search database.

custom database and the result was coded as 'Not Available'.

The proportion of responses labelled accurate for chatbot responses that used the database versus those that didn't also contains differences. As shown previously in Figure 4.8 in the previous section, a higher percentage of responses that didn't use the database were labelled as 'Accurate' compared to those that incorporated information from the database (69.5% for no database call versus 58.3% for responses with a database call). However, there was also a higher proportion of answers that were only 'partially correct' for responses with no database call. Responses that made use of the database also had a much higher rate of 'not available' answers where the chatbot failed to find information to answer the user's query. Though it may seem strange that there were more fully accurate answers without database usage, the LLM still had access to chat history and many of the queries' answers could be found in previous responses that involved querying the database. It seems that the LLM is fairly successful at inferring whether it has enough information to answer the question and when it needs to make use of other resources. Also, the higher percentage of 'not available' responses in ones that used the database is not necessarily a failure on the part of the chatbot as it is also able to correctly infer that it doesn't have the answer and can impart this to the user instead of feeding them false information. Again, expanding the database will likely resolve many issues.

Last, the referring expressions were always accurate and due to the nature of the space the users always approached models head on resulting in the expression 'Vor Ihnen...' (in front of you). This was expected but the other expressions referring to different directions would be useful in a more 'real-world' scenario where the user is further away or their view is obstructed by other objects.

## 5.4 Hallucinations

The most common hallucinations seem to be queries related to the Deutsche Tor and dates. A few times the chatbot stated that the Deutsche Tor building now houses Saarlouis' Municipal museum (Städtische Museum von Saarlouis). However, this is false and in fact the museum is located in what was once military barracks. A few other times it mixes up the Deutsche Tor with the gate of the same name in the nearby town of Metz in France. The chatbot also gave a few varying dates of when the construction of the Saarlouis fortress was completed, either stating 1680-1690 or 1680-1685. Both of these time spans are reasonable but no documents in the database give concrete dates as to when the construction was complete. An article online by the Saarland local government states that the last bastion of the fortress was built in 1686. Besides adding the missing information to the database, another possible solution is to specify that the chatbot can only give dates if this information has been retrieved from the database. For issues related to the Deutsche Tor adding 'Saarlouis' to the building name string such as 'Saarlouis' Deutsches Tor' or 'The Deutsche Tor in Saarlouis' may help.

The LLM also seemed to make the same amount of hallucinations whether the database was called or not. While sometimes misinformation in responses where the database was used was due to the retrieved text chunk not containing enough context information to distinguish which building was being talked about, there were other hallucinations which gave more misinformation:

### Hallucination using Database

**Query:** Wer war der Architekt?

**Response:** Der Architekt der St. Ludwig Kirche in Saarlouis war Jean Dumas, ein talentierter Baumeister aus der Barockzeit. Er hat maßgeblich dazu beigetragen, dass die Kirche heute als ein Meisterwerk der barocken Architektur gilt.

### Hallucination without Database

**Query:** Zu welchem Baustil gehört die Kirche?

**Response:** Die St. Ludwig Kirche in Saarlouis ist ein prächtiges Beispiel für den Barockstil. Dieser Stil, der im 17. und 18. Jahrhundert vorherrschte, ist bekannt für seine üppigen Ornamente, geschwungenen Formen und dramatischen Effekte. Die Kirche selbst wurde von dem Architekten Sebastian Sturm entworfen und ist ein beeindruckendes Zeugnis der barocken Architektur.

Neither of these names (Jean Dumas, Sebastian Sturm) are mentioned in the database, nor did online searches yield any architectures/builders with these names so it can be concluded that they are hallucinated by the LLM. This likely occurred due to an internal update to GPT-4 while keeping the same embeddings in the database, causing large differences in the resulting embeddings and their comparisons.

---

---

# **Chapter 6**

## **Conclusion and Future Work**

The goal of the thesis was to develop a prototype for a low-latency chatbot utilizing GPT-4 that could be deployed on a Hololens2 and make use of a custom database. This final chapter summarizes the findings from the prototype and user study and lays out suggestions for future steps to improve the application.

### **6.1 Findings and Contributions**

The main purpose of this work was to adapt GPT-4 to a tourism domain concerning the historical buildings of Saarlouis. To accomplish this a chatbot utilizing OpenAI APIs and LangChain was developed in Python along with a custom database from museum and wikipedia documents with Chroma. Unity was used to develop the user-side of the application including speech recognition and text-to-speech via Azure APIs as well as calculating distance to building models in order to give spatial references to the user.

The application was deployed on a Hololens2 with the Python server containing the chatbot and database functionalities being hosted on a separate device. The prototype was then tested in a German-language user study with eleven participants. Participants had to approach three virtual models (Französisches Tor, St. Ludwig's Kirche, and Deutsches Tor) and complete tasks to receive specific information from the chatbot about these buildings. They were also allowed to ask follow up questions of their choosing as long as it was related to the building and its history.

Results were largely positive with users finding the application easy to navigate and the response time of the chatbot was fast. Most of the answers were accurate or partially accurate and in many situations the chatbot could identify when it didn't have sufficient

information to answer a query and would express this to the user.

The prototype thus lays the basis for the development of a more sophisticated chatbot to be employed in an actual tourism scenario.

## 6.2 Future Work

The user study worked well in a small, indoor space with miniature versions of the models. In reality the VR headsets would be worn by tourists who walk around Saarlouis to view it how it was in the past [12].

A main factor to transition from indoor, controlled scenarios to outdoor space is to integrate GPS information into the application. This allows the possibility of more specific referring expressions such as whether a model is partially obscured by other nearby buildings, addressing multiple locations in once, and exact coordinates of important locations. There could also be information of traffic or construction in the city available. Currently, there is little research or support for accurate GPS for VR devices without relying on a companion device like a smartphone or API calls. VR headeset are limited in their storage and battery life so all development has to proceed with this in mind, hence the reliance on external APIs and servers to do heavy processing which raises issues of latency.

Transformers are also improving all the time and during this thesis there were many iterations to make use of the newest available technologies. At the time of writing, GPT-4o has been released [42] which accepts as input and can output text, audio, and images as well as a combination thereof. This could have multiple uses in a multi-modal application such as ours, opening up the possibility for features such as the user taking a screen shot of a building and having the chatbot analyze and answer questions about it. OpenAI has also released ways to fine-tune the model to one's use case and further research in retrieval augmentation generation would also boost accuracy of the chatbot responses.

At the moment there seems to be no general consensus or highly accredited publications on the best methods for document chunking and embedding techniques for vector databases. This is likely application-specific anyways, so a separate experiment or study could be done testing the accuracy of different chunking and retrieval methods as described in previous chapters.

Other ways to polish the prototype involve using an API that allows text-to-speech streaming so the user doesn't have to rely on reading text displayed on the VR headset which may be straining or wait for the response to reach completion before hearing it. Making the communication with the chatbot as natural and easy as possible is important for it to be engaging with users. In addition, personalization for each user may also help with user satisfaction. This may include a tool that takes into account the user's age,

background and length of response preference and of course extending the applications to other languages.

Finally, there is a rise in concern and mistrust regarding AI-powered applications as much as there is interest. Thus, data privacy and security issues should carefully be taken into account at every stage of the development process.

---

# Bibliography

- [1] chroma. <https://github.com/chroma-core/chroma>, 2023.
- [2] Windows Universal Samples. <https://github.com/microsoft/Windows-universal-samples>, 2023.
- [3] Hololens2 Hardware Description. <https://www.microsoft.com/en-us/hololens/hardware#document-experiences>, 2024.
- [4] ALIBEGOVIĆ, BESIM; PRILJAČA, NASER; KIMMEL, MELANIE AND SCHULTALBERS, MATTHIAS. Speech Recognition System for a Service Robot - A Performance Evaluation. In *2020 16th International Conference on Control, Automation, Robotics and Vision (ICARCV)* (2020), pp. 1171–1176.
- [5] ALMEIDA, FELIPE AND XEXÉO, GERALDO. Word Embeddings: A Survey. *CoRR abs/1901.09069* (2019).
- [6] BAHDANAU, DZMITRY; CHO, KYUNGHYUN AND BENGIO, YOSHUA. Neural Machine Translation by Jointly Learning to Align and Translate, 2016.
- [7] BARTELDS, MARTIJN; SAN, NAY; McDONNELL, BRADLEY; JURAFSKY, DAN AND WIELING, MARTIJN. Making More of Little Data: Improving Low-Resource Automatic Speech Recognition Using Data Augmentation, 2023.
- [8] BEITZEL, STEVE; DYKSTRA, JOSIAH; TOLIVER, PAUL; AND YOUZWAK, JASON. Exploring 3D Cybersecurity Visualization with the Microsoft HoloLens. pp. 197–207.
- [9] BENZEGHIBA, MOHAMMED FAOUZI; MORI, RENATO DE; DEROO, OLIVER; DUPONT, STEPHANE AND ET. AL. Automatic Speech Recognition and Speech Variability: A Review. *Speech Communication* 49, 10 (2007), pp. 763–786. Intrinsic Speech Variations.
- [10] BROWN, TOM B.; MANN, BENJAMIN; RYDER, NICK; SUBBIAH, MELANIE; KAPLAN, JARED; DHARIWAL, PRAFULLA; NEELAKANTAN, ARVIND; SHYAM, PRANAV; SASTRY, GIRISH; ASKELL, AMANDA; AGARWAL, SANDHINI; HERBERT-VOSS, ARIEL; KRUEGER, GRETCHEN; HENIGHAN, TOM; CHILD, REWON; RAMESH, ADITYA; ZIEGLER, DANIEL M.; WU, JEFFREY; WINTER, CLEMENS; HESSE, CHRISTOPHER; CHEN, MARK; SIGLER, ERIC; LITWIN, MATEUSZ; GRAY, SCOTT; CHESS, BENJAMIN; CLARK, JACK; BERNER, CHRISTOPHER; McCANDLISH, SAM; RADFORD, ALEC; SUTSKEVER, ILYA AND AMODEI, DARIO. Language Models are Few-Shot Learners. *CoRR abs/2005.14165* (2020).
- [11] CHEN, LINGJIAO; ZAHARIA, MATEI AND ZOU, JAMES. How is ChatGPT’s Behavior Changing Over Time?, 2023.

- [12] DERU, MATTHIEU; CHIKOBAVA, MARGARITA; POLLER, PETER; HAUCK, CHRISTIAN; KLIMENKO, ALBERT; SCHMEYER, THOMAS; KIEFER, GIAN-LUCA; BAUS, JÖRG; BRANDHERM, BORIS; NDIAYE, ALESSANE; LOEW, BENEDIKT; RUPP, HORST AND BRAUN, CHRISTIAN. *The Virtual Revival of the City of Saarlouis via a Multimodal Augmented/Mixed Reality Touristic HoloLens App.* 2023, pp. 15–28.
- [13] DING, MANDY. Augmented Reality in Museums. *Museums Augmented Reality-A Collection of Essays From the Arts Management and Technology Lboratory* (2017), pp. 1–15.
- [14] DOĞAN, FETHİYE IRMAK AND LEITE, IOLANDA. Open Challenges on Generating Referring Expressions for Human-Robot Interaction. *CoRR abs/2104.09193* (2021).
- [15] DOĞAN, FETHİYE IRMAK; KALKAN, SINAN AND LEITE, IOLANDA. Learning to Generate Unambiguous Spatial Referring Expressions for Real-World Environments. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2019), pp. 4992–4999.
- [16] ELASHRY, YOUSSEF. Python-Unity-Socket-Communication. <https://github.com/Siliconifier/Python-Unity-Socket-Communication>, 2022.
- [17] GALLEGOS, PILAR OPLUSTIL; O'MAHONY, JOHANNAH AND KING, SIMON. Comparing Acoustic and Textual Representations of Previous Linguistic Context for Improving Text-to-Speech. In *Proc. 11th ISCA Speech Synth. Workshop* (2021), pp. 205–210.
- [18] GUATAM, HARIOM. Word Embeddings: Basics. <https://medium.com/@hari4om/word-embedding-d816f643140>, 2020.
- [19] HAMMADY, RAMY; MA, MINHUA AND STRATHEARN, CARL. User Experience Design for Mixed Reality: A Case Study of HoloLens in Museum. *International Journal of Technology Marketing* 13 (2019).
- [20] HOCHREITER, SEPP AND SCHMIDHUBER, JÜRGEN. Long Short-Term Memory. *Neural Computation* 9, 8 (1997), pp. 1735–1780.
- [21] JIANG, ZHENGBAO; XU, FRANK F.; ARAKI, JUN AND NEUBIG, GRAHAM. How Can We Know What Language Models Know?, 2020.
- [22] KEIL, JENS; ZOLLNER, MICHAEL; BECKER, MARIO; WIENTAPPER, FOLKER; ENGELKE, TIMO AND WUEST, HARALD. The House of Olbrich — An Augmented Reality Tour through Architectural History. In *2011 IEEE International Symposium on Mixed and Augmented Reality - Arts, Media, and Humanities* (2011), pp. 15–18.
- [23] KHASHABI, DANIEL; KHOT, TUSHAR; SABHARWAL, ASHISH; TAFJORD, OYVIND; CLARK, PETER AND HAJISHIRZI, HANNANEH. UnifiedQA: Crossing Format Boundaries With a Single QA System. *CoRR abs/2005.00700* (2020).
- [24] KOO, SEUNGBUM; KIM, JINYOUNG; KIM, CHANGHYUK; KIM, JEEYEOP AND CHA, HEE SUNG. Development of an Augmented Reality Tour Guide for a Cultural Heritage Site. *J. Comput. Cult. Herit.* 12, 4 (2019).
- [25] KUDO, TAKU AND MATSUMOTO, YUJI. Chunking with Support Vector Machines. In *Second Meeting of the North American Chapter of the Association for Computational Linguistics* (2001).

- [26] KUNZE, LARS; WILLIAMS, T.; HAWES, NICK AND SCHEUTZ, MATTHIAS. Spatial Referring Expression Generation for HRI: Algorithms and Evaluation Framework. In *AAAI Fall Symposia* (2017).
- [27] LE CUN, YANN; BOTTOU, LEON; BENGIO, YOSHUA AND HAFFNER, PATRICK. Gradient-Based Learning Applied to Document Recognition. *Proceedings of the IEEE* 86, 11 (1998), pp. 2278–2324.
- [28] LEWIS, PATRICK; PEREZ, ETHAN; PIKTUS, ALEKSANDRA; PETRONI, FABIO; KARPUKHIN, VLADIMIR; GOYAL, NAMAN; KÜTTLER, HEINRICH; LEWIS, MIKE; TAU YIH, WEN; ROCKTÄSCHEL, TIM; RIEDEL, SEBASTIAN AND KIELA, DOUWE. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks, 2021.
- [29] LI, SHEN; SCALISE, ROSARIO; ADMONI, HENNY; ROSENTHAL, STEPHANIE AND SRINIVASA, SIDDHARTHA S. Spatial References and Perspective in Natural Language Instructions for Collaborative Manipulation. In *2016 25th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)* (2016), pp. 44–51.
- [30] LI, SHEN; SCALISE, ROSARIO; ADMONI, HENNY; ROSENTHAL, STEPHANIE AND SRINIVASA, SIDDHARTHA S. A Survey on Mobile Augmented Reality With 5G Mobile Edge Computing: Architectures, Applications, and Technical Aspects. *IEEE Communications Surveys Tutorials* 23, 2 (2021), pp. 1160–1192.
- [31] LIU, JINGYU; WANG, LIANG AND YANG, MING-HSUAN. Referring Expression Generation and Comprehension via Attributes. In *2017 IEEE International Conference on Computer Vision (ICCV)* (2017), pp. 4866–4874.
- [32] LIU, NELSON F.; LIN, KEVIN; HEWITT, JOHN; PARANJAPE, ASHWIN; BEVILACQUA, MICHELE; PETRONI, FABIO AND LIANG, PERCY. Lost in the Middle: How Language Models Use Long Contexts, 2023.
- [33] LIU, PENGFEI; YUAN, WEIZHE; FU, JINLAN; JIANG, ZHENGBAO; HAYASHI, HIROAKI AND NEUBIG, GRAHAM. Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing, 2021.
- [34] LIU, PETER J.; SALEH, MOHAMMAD; POT, ETIENNE; GOODRICH, BEN; SEPASSI, RYAN; KAISER, LUKASZ AND SHAZEE, NOAM. Generating Wikipedia by Summarizing Long Sequences, 2018.
- [35] LIU, YI; DENG, GELEI; XU, ZHENGZI; LI, YUEKANG; ZHENG, YAOWEN; ZHANG, YING; ZHAO, LIDA; ZHANG, TIANWEI AND LIU, YANG. Jailbreaking ChatGPT via Prompt Engineering: An Empirical Study, 2023.
- [36] MALIK, MISHAIM; AND KHAWAR MEHMOOD, MUHAMMAD KAMRAN MALIK AND MAKHDOOM, IMRAN. Automatic Speech Recognition: A Survey. In *Multimed Tools Appl* 80 (2021), p. 9411–9457.
- [37] MALKOV, YURY A. AND YASHUNIN, DMITRY A. Efficient and robust approximate nearest neighbor search using Hierarchical Navigable Small World graphs. *CoRR abs/1603.09320* (2016).
- [38] MEDSKER, LARRY AND LAKHMI, JAIN C. *Recurrent neural networks: Design and Applications*. CRC press, 1999.

- [39] MILLER, ALEXANDER; FISCH, ADAM; DODGE, JESSE; KARIMI, AMIR-HOSSEIN; BORDES, ANTOINE AND WESTON, JASON. Key-Value Memory Networks for Directly Reading Documents, 2016.
- [40] MOLINA, MARTIN; SANCHEZ-SORIANO, JAVIER AND CORCHO, OSCAR. Using Open Geographic Data to Generate Natural Language Descriptions for Hydrological Sensor Networks. *Sensors* 15, 7 (2015), pp. 16009–16026.
- [41] MORITZ, NIKO; Hori, TAKAAKI AND ROUX, JONATHAN LE. Streaming Automatic Speech Recognition with the Transformer Model. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (2020), pp. 6074–6078.
- [42] OPENAI. Hello GPT-4o. <https://openai.com/index/hello-gpt-4o/>, 2024.
- [43] OUYANG, LONG; WU, JEFF; JIANG, XU; ALMEIDA, DIOGO; WAINWRIGHT, CARROLL L.; MISHKIN, PAMELA; ZHANG, CHONG; AGARWAL, SANDHINI; SLAMA, KATARINA; RAY, ALEX; SCHULMAN, JOHN; HILTON, JACOB; KELTON, FRASER; MILLER, LUKE; SIMENS, MADDIE; ASKELL, AMANDA; WELINDER, PETER; CRISTIANO, PAUL; LEIKE, JAN AND LOWE, RYAN. Training Language Models to Follow Instructions with Human Feedback, 2022.
- [44] PETRONI, FABIO; ROCKTÄSCHEL, TIM; LEWIS, PATRICK; BAKHTIN, ANTON; WU, YUXIANG; MILLER, ALEXANDER H. AND RIEDEL, SEBASTIAN. Language Models as Knowledge Bases? *CoRR abs/1909.01066* (2019).
- [45] RADFORD, ALEC; WU, JEFFREY; CHILD, REWON; LUAN, DAVID; AMODEI, DARIO AND SUTSKEVER, ILYA. Language Models are Unsupervised Multitask Learners. *OpenAI blog* 1, 8 (2019), p. 9.
- [46] REN, YI; RUAN, YANGJUN; TAN, XU; QIN, TAO; ZHAO, SHENG; ZHAO, ZHOU AND LIU, TIE-YAN. FastSpeech: Fast, Robust and Controllable Text to Speech. In *Advances in Neural Information Processing Systems* (2019), vol. 32, Curran Associates, Inc.
- [47] SAFDARI, MUSTAFA; SERAPIO-GARCÍA, GREG; CREPY, CLÉMENT; FITZ, STEPHEN; ROMERO, PETER; SUN, LUNING; ABDULHAI, MARWA; FAUST, ALEKSANDRA AND MATARIĆ, MAJA. Personality Traits in Large Language Models, 2023.
- [48] TAMKIN, ALEX; BRUNDAGE, MILES; CLARK, JACK AND GANGULI, DEEP. Understanding the Capabilities, Limitations, and Societal Impact of Large Language Models, 2021.
- [49] TOPSAKAL, OGUZHAN AND AKINCI, T. CETIN. Creating Large Language Model Applications Utilizing LangChain: A Primer on Developing LLM Apps Fast. *International Conference on Applied Engineering and Natural Sciences* 1 (2023), pp. 1050–1056.
- [50] TSUNOO, EMIRU; KASHIWAGI, YOSUKE AND WATANABE, SHINJI. Streaming Transformer ASR With Blockwise Synchronous Beam Search. In *2021 IEEE Spoken Language Technology Workshop (SLT)* (2021), pp. 22–29.
- [51] VASWANI, ASHISH; SHAZEE, NOAM; PARMAR, NIKI; USZKOREIT, JAKOB; JONES, LLION; GOMEZ, AIDAN N.; KAISER, LUKASZ AND POLOSUKHIN, ILLIA. Attention Is All You Need, 2017.

- [52] WHITE, JULES; FU, QUCHEN; HAYS, SAM; SANDBORN, MICHAEL; OLEA, CARLOS; GILBERT, HENRY; ELNASHAR, ASHRAF; SPENCER-SMITH, JESSE AND SCHMIDT, DOUGLAS C. A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT, 2023.
- [53] ZHAO, WAYNE XIN; ZHOU, KUN; LI, JUNYI; TANG, TIANYI; WANG, XIAOLEI; HOU, YUPENG; MIN, YINGQIAN; ZHANG, BEICHEN; ZHANG, JUNJIE; DONG, ZICAN; DU, YIFAN; YANG, CHEN; CHEN, YUSHUO; CHEN, ZHIPENG; JIANG, JINHAO; REN, RUIYANG; LI, YIFAN; TANG, XINYU; LIU, ZIKANG; LIU, PEIYU; NIE, JIAN-YUN AND WEN, JI-RONG. A Survey of Large Language Models, 2023.

## Appendix

### GPT-4 Prompt English Translation of Figure 3.6

PREFIX = """You are a friendly, welcoming and enthusiastic tour guide in Saarlouis, Saarland, Germany.

Be cheerful and helpful, but please do not give specific recommendations about hotels and restaurants.

Sometimes, instead of a question, you will be given a direction to where an important historical building is, such as: 'In front of you is the Grosser Markt', 'To your right/left...' 'Behind you...'. In this case, do not treat the request as a request from the user, but rather as a trigger to provide brief historical information about the specific building.

Keep your information brief and you MUST provide this direction information at the beginning of your answer so the user knows where to look. For example: 'In front of you is...'. If the database does not contain relevant information, you can provide a rational answer based on your world knowledge. If you don't know the answer, just say you don't know and don't try to make up an answer. Keep your answer short and precise.

You have access to the following tools:"""

SUFFIX = """Get started! Remember to answer as an enthusiastic Saarlouis tour guide. Answer in German.

Previous conversation history: {chat\_history}

New question: {input} {agent\_scratchpad}"""

DFKI GmbH, Campus D3 2, Stuhlsatzenhausweg 3, 66123 Saarbrücken

Alice Chase  
Student Research Assistant, DFKI GmbH  
Campus D3 2 Stuhlsatzenhausweg 3  
D-66123 Saarbrücken, Germany



19. January 2024

DFKI Ethics Committee  
Saarland Informatics Campus  
Campus D3 2  
Stuhlsatzenhausweg 3  
66123 Saarbrücken

[www.dfki.de](http://www.dfki.de)

## Ethics Approval

Dear Alice Chase,

your study: “Large Language Model Applications in a Mobile Augmented Reality Tourism Scenario” submitted on 10.01.2024 has been approved by the DFKI Ethics Board.

Sincerely,

A handwritten signature in black ink that reads "Iris Merget".

Iris Merget  
Head of DFKI Ethics Committee

**Iris Merget**

Head of the DFKI Ethics Committee

+49 (0)681 85775-5344  
[iris.merget@dfki.de](mailto:iris.merget@dfki.de)

Deutsches Forschungszentrum für Künstliche Intelligenz GmbH (DFKI)

**Firmensitz**  
Kaiserslautern

**Weitere Standorte und Betriebsstätten:**  
Berlin, Bremen, Darmstadt, Lübeck,  
Oldenburg, Osnabrück, Saarbrücken,  
Trier

**Geschäftsführung**  
Prof. Dr. Antonio Krüger (Vorsitzender)  
Helmut Ditzer

**Vorsitzender des Aufsichtsrats**  
Dr. Ferri Abolhassan

Amtsgericht Kaiserslautern

HRB 2313  
UST-ID-Nummer DE 148 646 973  
Steuernummer 19/672/50006

Sparkasse Kaiserslautern  
IBAN DE30 5405 0220 0028 0004 79  
BIC/SWIFT MALADE51KLK



# Informed Consent of Participation

You are invited to participate in the user study **Large Language Model Applications In A Mobile AR Tourism Scenario** initiated and conducted by Alice Chase. The research is supervised by Alice Chase. Please note:

- Your participation is entirely voluntary and can be withdrawn at any time
- The user study will last approximately 30 minutes
- We will record personal demographics (age, gender, etc.)
- We will record audio, record data of your eye movements, and take notes during the user study.
- All records and data will be subject to standard data use policies
- Repeated participation in the study is not permitted

The alternative to participation in this study is to choose not to participate. If you have any questions or complaints about the whole informed consent process of this research study or your rights as a human research subject, please contact Alice Chase (E-Mail: alice.chase@dfki.de) and Alice Chase. You should carefully read the information below. Please take the time you need to read the consent form.

## 1. Purpose and Goal of this Research

Evaluate the performance of a LLM adapted to a tourism domain. Get feedback on the chatbot application and test its capabilities with users. Your participation will help us achieve this goal. The results of this research may be presented at scientific or professional meetings or published in scientific proceedings and journals.

## 2. Participation and Compensation

Your participation in this user study is completely voluntary. You will be one of approximately 10 people being tested for this research. You will receive no compensation for your participation. You may withdraw and discontinue participation at any time without penalty. If you decline to participate or withdraw from the user study, no one on the campus will be told. You can still demand a certificate of participation. You can deny answering questions if you feel uncomfortable in any way. The investigator may withdraw you from this research if continued participation will not meet the study goals or affect your well-being.

## 3. Procedure

After confirming the informed consent the procedure is as follows:

1. Participant gets overview of study and fills out pre-survey.
2. Participant puts on headset and does a tutorial application to get used to the technology (VR).
3. Participant wears headset and completes main study application.
4. Participant fills out exit survey.

The complete procedure of this user study will last approximately 30 minutes.

## 4. Risks and Benefits

There are no risks associated with this user study. Discomforts or inconveniences will be minor and are not likely to happen. If any discomforts become a problem, you may discontinue your participation. In order to minimize any risk of infection, hygiene regulations of the DFKI apply and must be followed. Any violations of the hygiene regulations or house rules of this institution can mean immediate termination of the study. If you get injured as a direct result of participation in this research, please reach out to the principal investigator. Enrolled students are automatically insured against the consequences of accidents through statutory accident insurance and with private liability insurance in case of any damages. You will not directly benefit through participation in this user study. We hope that the information obtained from your participation may help to bring forward the research in this field. The confirmation of participation in this study can be obtained directly from the researchers.

## 5. Data Protection and Confidentiality

We are planning to publish our results from this and other sessions in scientific articles or other media. These publications will neither include your name nor cannot be associated with your identity. Any demographic information will be published anonymized and in aggregated form. Contact details (such as e-mails) can be used to track potential infection chains or to send you further details about the research. Your contact details will not be passed on to other third parties. Any data or information obtained in this user study will be treated confidentially, will be saved encrypted, and cannot be viewed by anyone outside this research project unless we have you sign a separate permission form allowing us to use them. All data you provide in this user study will be subject of the General Data Protection Regulation (GDPR) of the European Union (EU) and treated in compliance with the GDPR. Faculty and administrators from the campus will not have access to raw data or transcripts. This precaution will prevent your individual comments from having any negative repercussions. Access to the raw interview transcript and transcribed observation protocol will be limited to the authors of this research, academic colleagues, and researchers with whom he might collaborate as part of the research process. Any interview content or direct quotations from the interview, that are made available through academic publications or other academic outlets will be anonymized so that you cannot be identified. During the study, we log experimental data, record audio, record data of your eye movements, and take notes during the user study. Raw data and material will be retained securely and in compliance with the GDPR, for no longer than necessary or if you contact the researchers to destroy or delete them immediately. As with any publication or online-related activity, the risk of a breach of confidentiality or anonymity is always possible. According to the GDPR, the researchers will inform the participant if a breach of confidential data was detected.

## 6. Identification of Investigators

If you have any questions or concerns about the research, please feel free to contact:

Alice Chase  
Student Researcher  
alice.chase@dfki.de

Alice Chase  
Principal Investigator  
Trippstadter Str. 122  
67663 Kaiserslautern, Germany  
alice.chase@dfki.de

## 7. Informed Consent and Agreement

This consent form will be retained securely and in compliance with the GDPR for no longer than necessary.

- I understand the explanation provided to me. I understand and will follow the hygiene rules of the institution. I understand that this declaration of consent is revocable at any time. I have been given a copy of this form. I have had all my questions answered to my satisfaction, and I voluntarily agree to participate in this user study.
- I agree that the researchers will record audio, record data of my eye movements, and take notes during the user study. I understand that all data will be treated confidentially and in compliance with the GDPR. I understand that the material will be anonymized and cannot be associated with my name. I understand that full anonymity cannot be guaranteed and a breach of confidentiality is always possible. From the consent of publication, I cannot derive any rights (such as any explicit acknowledgment, financial benefit, or co-authorship). I understand that the material can be published worldwide and may be the subject of a press release linked to social media or other promotional activities. Before publication, I can revoke my consent at any time. Once the material has been committed to publication it will not be possible to revoke the consent.

---

Printed Name of Subject

---

Signature of Subject

---

Location, Date