# A Modular and Efficient Framework for the Development of Large Language Model-Based Virtual Humans: An Educational Scenario

Michele Giordano[1][0009−0008−9375−5424], Daniele Berardini[3][0000−0001−7009−6317], Emanuele Frontoni[2][0000−0002−8893−9244], Primo Zingaretti[3][0000−0002−5709−2159], and Lorenzo Stacchio[2][0000−0002−9341−7651]

[1] eCampus University, Facoltà di Ingegneria, Italy
giordano.michele@gmail.com
[2] University of Macerata, Department of Political Sciences, Communication and International Relations, Macerata, Italy
{emanuele.frontoni, lorenzo.stacchio}@unimc.it
[3] Università Politecnica delle Marche, Department of Information Engineering (DII), Ancona, Italy
{d.berardini, p.zingaretti}@staff.univpm.it

**Abstract.** The integration of Large Language Models into virtual Human systems opens new avenues for creating interactive, intelligent agents capable of natural and personalized human-computer communication. However, the real-time generation and deployment of such avatars remain computationally demanding and often lack modularity or adaptability. In this paper, we propose an efficient and scalable framework for creating LLM-driven virtual humans that balances performance, responsiveness, and expressiveness. Our architecture combines lightweight dialogue management with multimodal synchronization pipelines to support speech and facial animation. The framework includes an optimization layer that enables on-device deployment without compromising interactivity. We demonstrate the effectiveness of our approach by deploying our system into several lightweight devices, showing improvements in latency and adaptability to user input. This work sets the stage for broader use of intelligent avatars in domains such as education, entertainment, and customer support.

**Keywords:** Virtual Humans · Large Language Models · Human-In-The-Loop · Metaverse

## 1 Introduction

As the vision of the metaverse moves from concept to implementation, the demand for lifelike, intelligent digital agents that can populate shared virtual spaces is rapidly increasing [6, 15]. These agents are expected not only to communicate naturally with users, but also generate dynamic, multimodal, and emotionally resonant conversations and interactions [17]. Within this emerging landscape, Artificial Intelligence (AI) serves as the foundational paradigm, enabling machines

to perceive, reason, and interact across digital and physical boundaries [14, 3, 11]. Among AI technologies, Large Language Models (LLMs) have emerged as key enablers of this transformation, offering state-of-the-art capabilities in understanding and generating human-like dialogue [7]. Additionally, LLMs convey vast world knowledge, which can also be complemented with external databases through the integration of Retrieval-Augmented Generation (RAG), which allows LLMs to reference external knowledge dynamically, enhancing their grounding in real-world facts and preventing hallucinations [1, 16]. Considering this, the integration of LLMs into virtual humans systems offers transformative opportunities for creating interactive, intelligent avatars capable of natural and personalized communication, for different use cases and actors [17]. These range from Education [18, 12], to Healthcare and Therapy [9], Entertainment and Social Companionship [17, 10], and also Cultural Heritage and Museums [20]. Despite these advances, existing systems often lack modularity, scalability, or adaptability to real-time deployment on resource-constrained devices [16]. This observation motivates our study, which is organized around a core research question: *Is it possible to design a scalable and modular architecture that simultaneously supports LLM-driven dialogue management, synchronized speech synthesis, and facial animation, while avoiding latency-inducing cascaded pipelines?*

To address such limitations, we propose a novel framework that combines: (i) A lightweight dialogue-management module that interfaces with an LLM generating multimodal outputs; (ii) A synchronized speech and facial animation pipeline with a 3D character; (iii) A modular graphical User Interface to deploy and chat with this avatar. To evaluate our approach, we evaluate our system requirements and performance on consumer devices, measuring latency and responsiveness. We considered our conversation use case, which is the education of Computer Science History. Our results demonstrate significant gains over baseline systems, supporting the feasibility of deploying intelligent avatars in domains of education.

The rest of this paper is structured as follows. In Section 2, we review the state of the art in LLM-driven virtual agents, with a focus on their deployment in immersive environments and educational technologies. Section 3 describes our proposed framework, outlining the architectural design, modular components, and implementation strategies for achieving low-latency, on-device execution. Section 4 presents the evaluation of our system, including performance benchmarks, qualitative user interaction scenarios, and a discussion on deployment constraints. Finally, Section 5 concludes the paper by summarizing our contributions and outlining directions for future research.

## 2 Related Works

### 2.1 LLM-Driven Agents in Virtual Reality

The integration of LLMs into virtual humans and environment systems offers transformative opportunities for creating interactive, intelligent agents capable of natural and personalized communication [8, 12].

For example, [19] developed an open-domain avatar chatbot embedded in a VR environment, leveraging an LLM for dialogue generation while meeting challenges in multimodal synchronization and response latency. From pioneering works like the mentioned one, a growing body of work has explored the deployment of LLM-based agents in immersive VR environments. In [17], authors introduced an LLM-driven agent embedded in VRChat capable of simulating human interaction by combining GPT-4 responses with memory and retrieval modules. Their system focused on optimizing the context size used in the prompt to balance realism and computational load. On a similar line [13] implemented a VR environment with multiple voice-driven avatars powered by a locally deployed LLM, coupled with automatic speech recognition (ASR), Text-to-Speech (TTS), and lip-syncing. Their pilot study examines different avatar status indicators (e.g., "Thinking" lights, loading bars), yielding key design insights into enhancing responsiveness and perceived realism. More recently, [12] presents a novel framework to enable users to seamlessly switch between open-ended conversation and domain-specific knowledge via RAG through natural interactions with AI-driven avatars. Considering the efficiency of such approaches, to the best of our knowledge, only [16] studied an edge-optimized framework for low-latency LLM inference, demonstrating that substantial performance and energy gains are possible on lightweight devices without sacrificing model capacity.

Different from the works mentioned here, we introduce a modular, efficient, and open-source pipeline that could be used to deploy LLM-driven 3D avatars in different digital environments, including Extended Reality ones.

### 2.2 LLM Virtual Humans in Education

Different works analyzed the effectiveness of LLM-driven Virtual Humans [4, 17, 22, 20, 5]. For instance, [4] discussed how LLM-based embodied educational avatars could improve educational settings, including personalized instruction, adaptive feedback, and collaborative learning support. They also suggested that multimodal LLM inputs (e.g., image, audio, video) should be adopted. On this line, authors of [5] explored the effectiveness of ChatGTP-driven VR in facilitating machine learning education through an avatar that provides real-time assistance and uses LLMs to personalize learning paths based on various sensor data from VR. They observed that while both learning modes supported learning effectively, personalization significantly improved learning outcomes. On a similar line [22] proposed a novel system providing embodied AI-Guided Interactive digital teachers for education, which integrates an LLM-based chatbot employing RAG to organize and retrieve useful educational documents for the LLM. They also created an animatable 3D avatar powered by text-to-speech and audio-to-motion models to provide students with interactive conversation experiences. Authors of [21] specifically investigate how LLM-powered chatbots and avatar guides impact user engagement, experiential quality, and learning outcomes within virtual museum environments. They designed and implemented two key avatars: a text-based chatbot interface and a more immersive MetaHuman-style avatar guide, both utilizing LLMs to dynamically interact with museum visitors

in a VR setting. A controlled user study was conducted, and findings indicate that the avatar guide produced significantly higher engagement and perceived realism than the text-only chatbot. Our work builds upon these contributions by proposing a modular and efficient framework designed for scalability and on-device deployment, taking as a use case the development of a computer science history virtual teacher.

## 3 System Architecture

We here describe the system architecture of the designed framework and its implementation. Our system matches a client-server architecture, which is particularly suitable for our approach to balance different computational demands. Our client application is designed to run on resource-constrained devices such as a smartphone or immersive headset and is responsible for rendering the interface, managing user interaction, and animating the virtual humans. Conversely, the server hosts the main logic and so the computationally intensive modules executing it on high-performance hardware. It is also worth noticing that isolating these components enhances both modularity and scalability. Our framework is visually depicted in Figure 1.

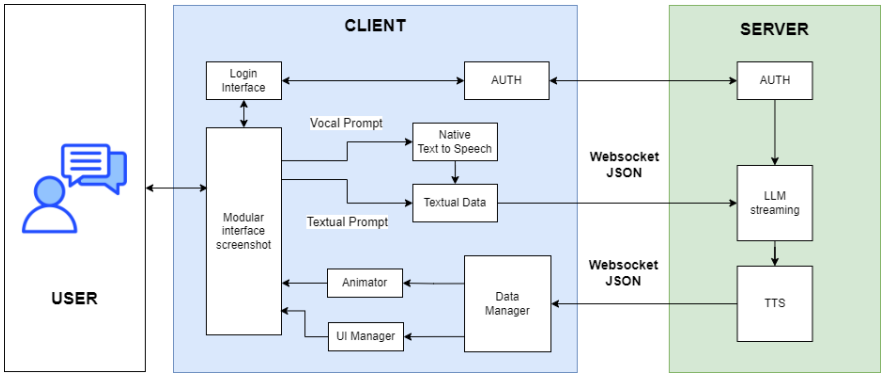



Fig. 1: System architecture of our multimodal LLM-driven virtual humans framework

The proposed architecture is designed to support a real-time interactive experience with virtual humans through multimodal input and output channels, and a graphical User Interface. The flow begins with the user providing a prompt (textual or audio) via a mobile or XR device. This input is managed by the client application, developed in Unity, which handles user interface rendering, animation, and prompt dispatching.

The client transmits the user's input to the server, which is implemented in Python and structured into independent services for speech recognition, language model inference, and speech synthesis. Depending on the input type, the

server performs automatic speech recognition or directly processes the text. An LLM then generates a textual response, which is optionally refined. Then, the generated text is fed to a text-to-speech model to generate the corresponding audio. The resulting multimodal response (composed of both text and audio) is sent back to the client interface. The client then displays the textual explanation and animates the virtual teacher in sync with the speech, ensuring a coherent and immersive user experience. This modular and asynchronous architecture enables low-latency communication and facilitates easy integration of additional components, such as specialized processors for emotion, language filtering, or content personalization.

### 3.1 Internal asynchronous modules

The client-server architecture has been structured around the principle of modularity and responsibility separation, enabling a modular and testable organization of the processing pipeline. Both the client and server implementations feature internal asynchronous modules, called Virtual Roles, each one responsible for a specific task, mirroring the organizational structure of a small audio-visual production company. On the client side, distinct modules manage authentication, network transmission, message ordering, and the final rendering of the avatar's speech animations, lip sync, and UI updates. These components operate independently but in a coordinated sequence to ensure low-latency interaction, synchronized audiovisual playback, and fault-tolerant communication. On the server side, the architecture mirrors a similar compartmentalization: specific modules handle tasks such as server-side authentication, text generation and refinement, speech synthesis, and response delivery. Each unit is designed to perform a narrowly defined function and communicates asynchronously with others through lightweight queues and shared memory spaces. This functional decomposition, implemented respectively in C# and Python, is supported by an Extended Finite State Machine system with debug interfaces that run on all target devices, allowing the minimization of interdependencies, facilitating debugging, and supporting future extensions. By decoupling content creation from formatting and transmission, the architecture remains scalable and adaptable to different deployment contexts and computational constraints.

### 3.2 Client Implementation

The client, developed in Unity 6, handles the rendering and animation of the virtual humans, as well as the user interface and the orchestration of user inputs and responses. As described in Figure 1, it is composed of four main components:

– Native Text to Speech: Converts spoken user prompts into text. We decided to adopt, per device, its native TTS for several reasons: (i) on-device synthesis eliminates network latency responses enabling truly real-time, interactive feedback. (ii) Client-side processing substantially reduces server

load and infrastructure cost by leveraging users' hardware instead of managing backend compute and bandwidth. Third, privacy is inherently stronger: user utterances and generated speech remain on the device, preventing audio transmission to remote servers, an outcome consistent with privacy-by-design principles.
- Data Manager: Sends and receives data in JSON format, ensures integrity and the correct order of paragraphs, and orchestrates input/output flow between components.
- UI Manager: Manages the graphical interface, generating elements at runtime, sends input prompts to the server, and receives multimodal outputs.
- Animator: Triggers facial and body animations synchronized with the received audio and text. In particular, we integrated the animations of the virtual humans into Unity3D as a rigged FBX model with blend shapes, with pre-calculated lighting to optimize visual rendering without burdening performance.

### 3.3 Server Implementation

The server's architecture design follows modular principles, with each module implementing one step in the generation process. The server includes the following functional components:

- The AUTH component manages authentication and encryption. The security WSS layer is applied during the initial connection to reduce latency.
- The LLM Streaming component exploits incrementally generated text using a preloaded Large Language Model, producing output paragraph by paragraph (streaming approach) to reduce latency.
- The Text Revision quickly processes each paragraph to improve clarity and formatting, optimizing it for speech synthesis.
- Then, the TTS converts the revised text into high-quality audio using a preloaded speech synthesis model. It uses a custom vocabulary and a crafted reference voice sample processed through voice cloning techniques. This creates a voice that shows key human speech traits. The audio is then encoded into a compressed format for efficient delivery.
- Finally, both generated text and audio are transmitted to the client, in a structured JSON object, including metadata such as message ID, position, and timestamp, through a persistent WebSocket connection.

To summarize, the design of each component aims at providing low latency, even on limited local hardware. It is worth mentioning that the proposed architecture is fully modular and can run on any platform that exposes a TTS service, without depending on outside services.

## 4 Results

In this Section, we first describe how we implemented the different components of our architecture (Section 4.1) and report the results obtained in an experimental

setting to demonstrate the general efficiency of the implemented system (Section 4.2).

### 4.1 System implementation

We here describe the implementation choices provided to implement our framework, which basical usage is depicted in Figure 2.

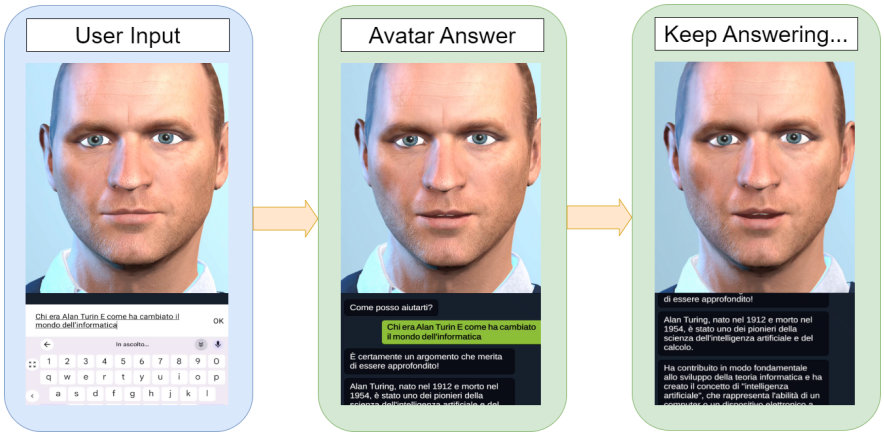



Fig. 2: Illustration of the interaction pipeline. The left panel ("User Input") shows the user posing a question to the virtual humans via text input (in this case, displayed on a smartphone interface). The right panel ("Avatar Answer") shows the avatar generating a verbal response in real-time.

*Selected Hardware* For the implementation of our system, we adopted a minimal hardware configuration required to achieve full real-time performance. We so executed our system and perform the experiments that follows in a local machine equipped with an AMD Ryzen 7 5700X CPU, 32 GB DDR4 RAM, NVMe SSD storage, and a PCIe 4.0 compatible motherboard. At the core of this setup is the NVIDIA RTX 3060 with 12 GB of VRAM, selected as the critical component capable of keeping both the LLM and TTS models fully resident in memory and performing low-latency inference through CUDA and tensor core acceleration. This local machine allows the entire server-side pipeline to run locally, without reliance on cloud infrastructure. The Unity-based client application runs separately on target devices such as desktop and Android systems, which maintain a lightweight runtime profile thanks to the complete offloading of computational tasks to the server.

*3D model and Scene Design* The 3D virtual teacher was implemented in Unity3D using a fully rigged character model with facial blend shapes, high-resolution

textures, and a minimal static environment designed for real-time performance. The scene includes only the character and essential lighting, allowing the use of a relatively high polygon budget without compromising frame rates. A fixed frontal camera framing supports consistent eye contact and simplifies interaction management. Lip sync is achieved through real-time audio signal analysis and activation of predefined blend shapes, enabling synchronized mouth movements with low computational cost. Minor idle animations and facial micro-movements enhance expressiveness while maintaining responsiveness across target platforms. Although the current scene is intentionally minimal, it can be augmented with additional complexity when needed, for example, to support immersive AR, VR or mixed reality interactions, or to accommodate specific accessibility requirements for users with impairments.

*Large Language Model* The selection of the text generation model was guided by extensive testing of open-source architectures compatible with the available VRAM. For this reason, we adopted the 3B parameter version of Hermes [2], derived from Meta's LLaMA 3.2, was adopted after initial experiments with larger models to improve the overall system efficiency without saturating the available GPU VRAM. To support progressive token-level streaming and reduce memory footprint, the selected model was converted from its original safe-tensors format to the more efficient gguf format, optimized for local inference through *lama cpp* library [4]. The inference was performed with a streaming decoding strategy, setting a low temperature of 0.20, top-p sampling with a threshold of 0.92, and a top-k limit of 40. Additionally, minor penalties were introduced to reduce repetitiveness (repeat penalty = 1.20) and balance lexical diversity (presence penalty = 0.15; frequency penalty = 0.10), and avoid too long statements (token limit = 64). The system prompt was structured to include strict task-oriented constraints. These guided the model to respond in Italian using an academic and formal register, avoiding hallucinations and enforcing factual accuracy, especially within the historical domain of informatics. The prompt also imposed syntactic rules, such as avoiding initials with dots and limiting the output to a maximum of 200 words, promoting compact and precise content delivery. This is depicted in Figure 3.

*Text to Speech* For speech synthesis, the project adopted a custom fork of the open-source CoquiTTS framework [5]. On top of CoquiTTS, the voice of the virtual teacher was crafted through an iterative process combining targeted audio sampling, signal enhancement, and voice cloning, also controlling utterances under tonal consistency and phonetic clarity. This approach enabled the creation of modular vocal timbre without requiring professional-grade recording sessions, remaining compatible with open-source tools and standard local hardware.

[4] https://github.com/abetlen/llama-cpp-python
[5] https://github.com/coqui-ai/TTS

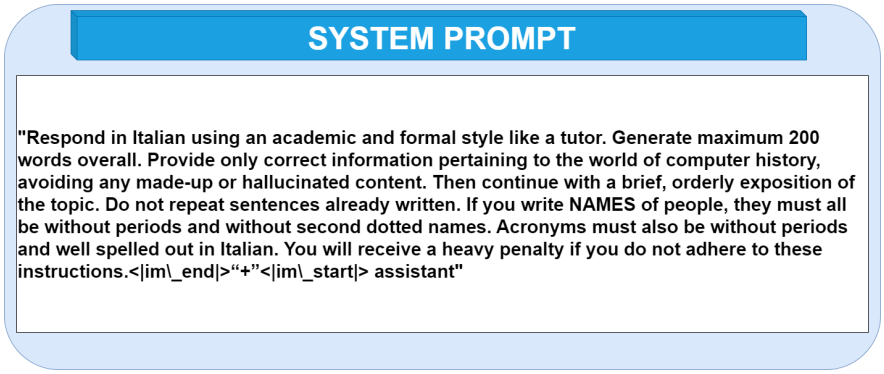



Fig. 3: Adopted System Prompt.

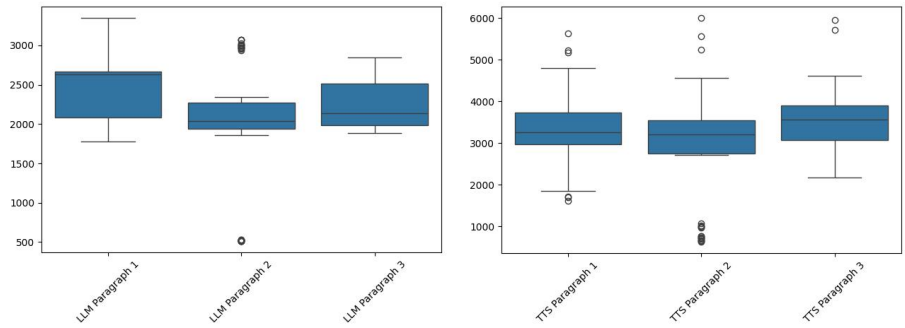
### 4.2 Experimental Setting

To validate the runtime performance of our pipeline, we measured the inference latency for both the LLM-based paragraph generation and the subsequent TTS synthesis. We selected five representative prompts covering different linguistic structures and semantic contents. Each prompt triggered the generation of three distinct paragraphs using the Hermes-3B model running locally via llama.cpp, and their respective audio renderings were synthesized through our custom CoquiTTS-based engine. All measurements were conducted on the same hardware configuration described in Section 4.1, namely a local machine equipped with an AMD Ryzen 7 5700X CPU and an NVIDIA RTX 3060 GPU with 12 GB VRAM, ensuring consistent conditions without reliance on cloud services. To assess system latency in realistic conditions, we measured it at the paragraph level using five question prompts, each generating three paragraphs. For each paragraph, we recorded the combined time of LLM inference and TTS synthesis, as audio was streamed to the client immediately upon readiness. This approach mirrors the system's real-time behavior and minimizes perceived latency, ensuring smooth and natural interactions.

All the obtained measurements, reported in Table 1, were collected on the local hardware described in Section 4.1 and generated by fixing the random seed, ensuring consistency and reproducibility of the results. As depicted in Table 1, the adopted LLM exhibits inference times that are generally low, with mean values ranging from $\sim 500$ ms to $\sim 3000$ ms, depending on the prompt and paragraph. Prompt 5, Paragraph 2 exposes a low generation time of 518.84 ms, given by a less complex generation path induced by that specific prompt. Those values are, in general, smaller than the TTS model times, which range from $\sim 2000$ ms to $\sim 4000$ ms. This confirms that the synthesis step is the dominant contributor to the overall response time. To have a general view of the performance, we depict the boxplots of all the generation times, aggregated by prompts, in Figure 4. Those values further confirm the aforementioned considerations.

| Prompt ID | Paragraph 1 | Paragraph 2 | Paragraph 3 |
|---|---|---|---|
| Prompt 1 | LLM: 2097.90 (49.45)<br>TTS: 3165.26 (180.64) | LLM: 2054.60 (47.28)<br>TTS: 2928.37 (162.63) | LLM: 2625.43 (64.38)<br>TTS: 4016.29 (244.52) |
| Prompt 2 | LLM: 1964.98 (91.71)<br>TTS: 2062.84 (439.23) | LLM: 1977.22 (72.50)<br>TTS: 3605.12 (757.04) | LLM: 1930.69 (75.02)<br>TTS: 3728.32 (874.20) |
| Prompt 3 | LLM: 2615.27 (90.12)<br>TTS: 3200.11 (510.94) | LLM: 2220.24 (147.33)<br>TTS: 3778.15 (812.59) | LLM: 2053.01 (137.70)<br>TTS: 2617.19 (444.68) |
| Prompt 4 | LLM: 3002.00 (176.89)<br>TTS: 4392.92 (546.18) | LLM: 3000.59 (42.72)<br>TTS: 3581.59 (248.89) | LLM: 2488.44 (36.46)<br>TTS: 3794.39 (237.59) |
| Prompt 5 | LLM: 2652.78 (32.06)<br>TTS: 3433.40 (215.17) | LLM: 518.84 (5.15)<br>TTS: 806.14 (156.61) | LLM: 2137.81 (15.34)<br>TTS: 3356.97 (436.93) |

Table 1: Runtime mean (std) in milliseconds for each paragraph's generation (LLM) and synthesis (TTS), across five prompts.




(a) Distribution of LLM Paragraph Times (b) Distribution of TTS Paragraph Times

Fig. 4: Boxplot comparison of generation (LLM) and synthesis (TTS) times across paragraphs.

## 5 Discussions and Conclusion

We presented a modular and efficient framework for deploying LLM-driven virtual humans on resource-constrained devices. By integrating a lightweight dialogue-management layer with a synchronized speech-and-facial-animation pipeline, our architecture achieves significant reductions in end-to-end latency. Our quantitative evaluation of the LLM and TTS modules demonstrates that response times per paragraph remain within a practical window for near real-time interaction, ensuring natural communication with the virtual avatar. Addressing this, we plan to conduct controlled studies to assess communicative and pedagogical effectiveness, user engagement, and learning impact. Moreover, we did not include a direct empirical comparison with SOTA approaches. For future works, we plan to conduct controlled studies to assess communicative and pedagogical effectiveness, user engagement, and learning impact against SOTA approaches. Addi-

tionally, we planned to investigate: (i) the adoption of efficient RAG Paradigms, to allow avatars to dynamically access and ground their responses on external knowledge bases; (ii) the usage of efficient Multimodal LLM architectures to assess their performance, inference costs, and energy consumption; (iii) conducting a large-scale, and user-centered studies to provide insights into perceived naturalness, engagement levels, and learning outcomes. To conclude, this work laid the groundwork for modular and efficient applications of intelligent avatars across education, customer support, and entertainment scenarios, where real-time responsiveness and avatar customization are critical aspects.

**Acknowledgments.** This work has been co-funded by the European Union under the "AGRITECH EU - Digital agriculture for sustainable development" founded in the Digital Europe Programme, Project 101123258, and CTE Square Pesaro CUP D74J22000930008FSC, MISE 2014-2020.

## References

1. Arslan, M., Ghanem, H., Munawar, S., Cruz, C.: A survey on rag with llms. Procedia Computer Science **246**, 3781–3790 (2024)
2. Ayed, F., Maatouk, A., Piovesan, N., De Domenico, A., Debbah, M., Luo, Z.Q.: Hermes: A large language model framework on the journey to autonomous networks. arXiv preprint arXiv:2411.06490 (2024)
3. Cascarano, P., Franchini, G., Porta, F., Sebastiani, A.: On the First-Order Optimization Methods in Deep Image Prior. Journal of Verification, Validation and Uncertainty Quantification **7**(4), 041002 (2022)
4. Fink, M.C., Robinson, S.A., Ertl, B.: Ai-based avatars are changing the way we learn and teach: benefits and challenges. In: Frontiers in Education. vol. 9, p. 1416307. Frontiers Media SA (2024)
5. Gao, H., Xie, Y., Kasneci, E.: Pervrml: Chatgpt-driven personalized vr environments for machine learning education. International Journal of Human–Computer Interaction pp. 1–15 (2025)
6. Gatto, L., Gaglio, G.F., Augello, A., Caggianese, G., Gallo, L., La Cascia, M.: Met-iquette: enabling virtual agents to have a social compliant behavior in the metaverse. In: 2022 16th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS). pp. 394–401. IEEE (2022)
7. Hadi, M.U., Qureshi, R., Shah, A., Irfan, M., Zafar, A., Shaikh, M.B., Akhtar, N., Wu, J., Mirjalili, S., et al.: A survey on large language models: Applications, challenges, limitations, and practical usage. Authorea Preprints (2023)
8. John, K.S., Roy, G.A., S, B.P.: Llm based 3d avatar assistant. In: 2024 1st International Conference on Trends in Engineering Systems and Technologies (ICTEST). IEEE (2024)
9. Kenny, P.G., Parsons, T.D., Geer, A., ORCiD, I., Parsons, T.D.: Virtual standardized llm-ai patients for clinical practice. Annual Review of Cybertherapy And Telemedicine 2024 p. 177 (2024)
10. Kimara, E., Oguntoye, K.S., Sun, J.: Personaai: Leveraging retrieval-augmented generation and personalized context for ai-driven digital avatars. arXiv preprint arXiv:2503.15489 (2025)

11. Loli Piccolomini, E., Gandolfi, S., Poluzzi, L., Tavasci, L., Cascarano, P., Pascucci, A.: Recurrent neural networks applied to gnss time series for denoising and prediction. In: 26th International Symposium on Temporal Representation and Reasoning (TIME 2019). pp. 10–1. Schloss Dagstuhl–Leibniz-Zentrum für Informatik (2019)
12. Marquardt, A., Golchinfar, D., Vaziri, D.: Ragatar: Enhancing llm-driven avatars with rag for knowledge-adaptive conversations in virtual reality. In: 2025 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW). pp. 1604–1605. IEEE (2025)
13. Maslych, M., Pumarada, C., Ghasemaghaei, A., LaViola Jr, J.J.: Takeaways from applying llm capabilities to multiple conversational avatars in a vr pilot study. arXiv preprint arXiv:2501.00168 (2024)
14. Reiners, D., Davahli, M.R., Karwowski, W., Cruz-Neira, C.: The combination of artificial intelligence and extended reality: A systematic review. Frontiers in Virtual Reality **2**, 721933 (2021)
15. Schmidt, S., Köysürenbars, I., Steinicke, F.: Frankenstein's monster in the metaverse: User interaction with customized virtual agents. IEEE Transactions on Visualization and Computer Graphics (2024)
16. Tian, C., Qin, X., Tam, K., Li, L., Wang, Z., Zhao, Y., Zhang, M., Xu, C.: Clone: Customizing llms for efficient latency-aware inference at the edge. arXiv preprint arXiv:2506.02847 (2025)
17. Wan, H., Zhang, J., Suria, A.A., Yao, B., Wang, D., Coady, Y., Prpa, M.: Building llm-based ai agents in social virtual reality. In: Extended Abstracts of the CHI Conference on Human Factors in Computing Systems. pp. 1–7 (2024)
18. Xu, T., Zhang, Y., Chu, Z., Wang, S., Wen, Q.: Ai-driven virtual teacher for enhanced educational efficiency: Leveraging large pretrain models for autonomous error analysis and correction. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 39, pp. 28801–28809 (2025)
19. Yamazaki, T., Mizumoto, T., Yoshikawa, K., Ohagi, M., Kawamoto, T., Sato, T.: An open-domain avatar chatbot by exploiting a large language model. In: Proceedings of the 24th Annual Meeting of the Special Interest Group on Discourse and Dialogue. Association for Computational Linguistics, Prague, Czechia (2023)
20. Zhang, S., Ma, M., Li, Y., Man, K.L., Smith, J., Yue, Y.: The effects of llm-empowered chatbots and avatar guides on the engagement, experience, and learning in virtual museums. International Journal of Human–Computer Interaction pp. 1–13 (2025)
21. Zhang, Y., Zhao, S., Tian, X., Sun, H.: Design and development of "virtual ai teacher" system based on nlp. In: 2023 11th International Conference on Information and Education Technology (ICIET). pp. 141–145. IEEE (2023)
22. Zhao, Z., Yin, Z., Sun, J., Hui, P.: Embodied ai-guided interactive digital teachers for education. In: SIGGRAPH Asia 2024 Educator's Forum. pp. 1–8 (2024)