

The Negotiator: Interactive Hostage-Taking Training Simulation

Pierre-Benjamin Monaco¹^a, Per Backlund²^b and Stéphane Gobron³^c

¹Master of Software Engineering, University of Applied Sciences and Arts Western Switzerland (HES-SO), Avenue de Provence 7, Lausanne, Switzerland

²Department of Game Technologies, School of Computer Science, University of Skövde, Skövde, Sweden

³HE-Arc School of Engineering, University of Applied Sciences and Arts Western Switzerland (HES-SO), Espace de l'Europe 11, Neuchâtel, Switzerland

Keywords: Mobile Phone, Natural Language Processing, Hostage-Taking, 3D, Immersion, Negotiation, Decision-Making, Human-Machine Interface, Virtual Reality, Multimodal Systems.

Abstract: High-stakes professions like negotiators and pilots utilize simulations for risk-free training, enhancing skills and decision-making. Traditional simulations, while effective, are resource-intensive. Computer simulations offer a scalable alternative but lack realism and sensory engagement, impacting immersion. This study explores mobile phones' role in simulation-based learning, using a 3D hostage-taking scenario to improve immersion through realistic interactions. The simulation integrates a detailed environment and interactive elements, and demonstrate the potential of mobile technology to enhance training in critical fields by making simulations more lifelike and engaging. This paper is presented alongside a twin paper, both part of the same project Master final project. The second paper introduces a different version of the simulation using a Large Language Model (like ChatGPT) for generating dialogue freely and interactively. It also discusses the results of a study comparing immersion levels between the two versions.

1 INTRODUCTION

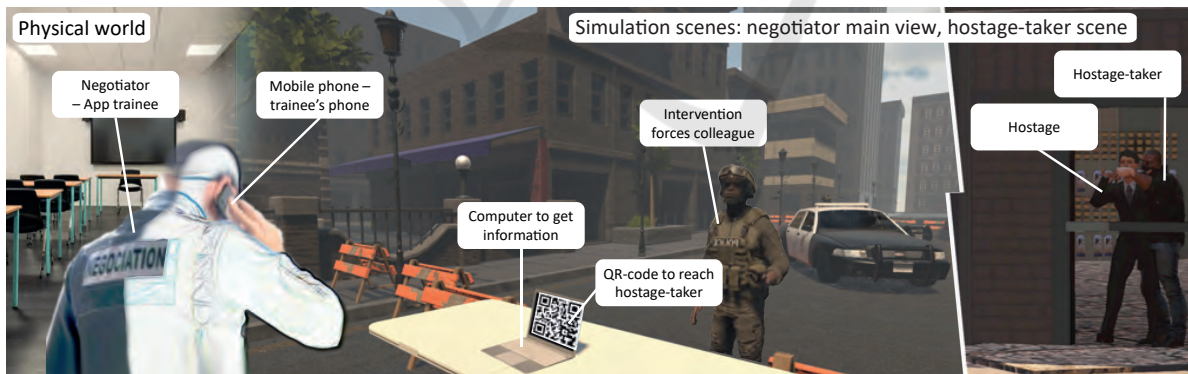





Figure 1: Overview of the negotiation simulation: the user on the left is in the physical world (e.g. a classroom) with (1) a computer with which he uses the application and (2) his cell phone to converse with the simulated hostage-taker.

High-risk professions, especially in the fields of healthcare, aviation, and security, do not allow for errors. The consequences of such errors can be extremely serious and sometimes even lead to the death

of one or more individuals. The question then arises: “Is it possible to train these skills without risking the lives of the people involved?”. The answer is simple: “Simulation”. Training simulations are effective tools for experiencing complex situations and are widely used in the aforementioned fields (Bienstock and Heuer, 2022), as well as many others (Valiente

^a <https://orcid.org/0000-0002-4487-8195>

^b <https://orcid.org/0000-0001-9287-9507>

^c <https://orcid.org/0000-0003-3796-6928>

et al., 2019). In a simulation, it is possible to make mistakes without suffering the sometimes disastrous consequences that would result. Based on the concept of Experimental learning, educational or training simulations allow for the integration of behaviors requiring quick and automatic actions and/or reflections through experience and repetition (Salas et al., 2009). Today, a surgeon can practice making incisions on a realistic mannequin, a firefighter can train to enter a burning house and find victims amid the smoke, and a pilot can practice landing a commercial airplane in disastrous weather conditions with engines failing. These situations are practiced using more or less digitized simulations. The closer a simulation is to reality, the more effective it is as a training tool (Bunker, 1978). Fidelity can be distinguished into two types: sensory fidelity and cognitive fidelity (Hochmitz and Yuviler-Gavish, 2011). Depending on the educational/training objectives, it is important to consider both aspects. The effort required to increase realism generally follows an exponential trend, and depending on available resources, sometimes choices must be made: “How much is too much?”. The educational effectiveness of a simulation is strongly linked to the level of immersion of the participants, and the realism of a simulation contributes greatly to this (MacLean et al., 2019). Non-digitized or minimally digitized simulations often require the engagement of many resources, human, financial, and temporal. The preparation of a simulation can take years, and the implementation of them is expensive, especially in the case of a full-scale simulation (Simpson and Oser, 2003). On the other hand, a digital simulation can be just as expensive to develop but proportionally requires fewer resources for implementation because it involves a program that can be repeated indefinitely, like a video game.

This paper explores the novel possibility of using mobile phones as an interaction tool with virtual characters in a training simulation. This simulation is set in a hostage-taking scenario where the participant is a police negotiator who must interact with the hostage-taker. In this type of scenario, cognitive fidelity is primarily sought; the dialogue must be coherent. However, the sensory component should not be overlooked, as learning is always more effective when multiple senses are stimulated (Baines, 2008). The question is whether it is possible to simulate the natural use of a mobile phone and also whether this contributes to realism and ultimately, immersion. Traditionally, a negotiation simulation or more broadly, dialogue with a virtual character relies on dialogue trees (Lobo et al., 2022). These simulations typically offer a simple “chat” interface where the participant

can read the reactions of the simulated character and respond by clicking among various choices available to them (Bel-Enguix et al., 2009). A simulation of this type was conducted before this project, where the mobile phone was already used as an interface (Monaco et al., 2023). The use of it seemed to increase immersion when the virtual character’s phrases were prerecorded and played on the phone. The realism of the situation was lost right after hearing the character’s speech because to respond, one had to click buttons on the phone screen instead of speaking. Building on the insight of using phones more naturally, this research project emerged. The findings are shared across two twin papers. This paper focuses on the design and implementation of a negotiation simulation that innovatively employs mobile phones for natural interaction with a virtual hostage-taker (Figure 1). The companion paper delves into a new dialogue generation system using LLMs (like ChatGPT) and compares immersion levels between versions entitled “Interactive storytelling apps - increasing immersion and realism with artificial intelligence?” (Monaco et al., 2024).

2 CONCEPTION

The development of this simulation begins with identifying the critical components necessary for its creation. Such simulations require a specific area, varying from an indoor space to an outdoor environment, designed to replicate the real-world scenario being simulated. Moreover, the involvement of actors and specialists, who assume various roles during the exercise, is essential for the simulation’s authenticity. Furthermore, detailed briefings and debriefings, executed by supervisors prior to and following the simulation, are fundamental for preparing participants and offering them constructive feedback.

To economically replicate this environment, one can employ video game development tools to create a lifelike 3D landscape. The briefing phase can be facilitated by a Non-Player Character (NPC) within the virtual environment. The process of debriefing can also be programmed and incorporated in a similar fashion, although this aspect is deliberately not included in the scope of this project. The primary objective of this study is to investigate the integration of mobile phones as interactive devices, rather than producing a market-ready product. It is essential for the hostage-taker simulation to serve dual functions: as an entity within the 3D world and as an interactive dialogue agent. Consequently, this requires the development of an immersive 3D application designed to run on a computer.

In this simulation, participants take on the role of negotiators, responsible for engaging in dialogue with the hostage-taker through phone communication. For a fully immersive experience, participants would use their own mobile devices to call an actor representing the hostage-taker. Consequently, the development of a smartphone application is required to replicate the interfaces and capabilities of a standard telephone, including the functionality to record phone numbers and place calls. This application must be seamlessly integrated with the desktop software that manages the simulation's flow and triggers specific scenarios based on the interactions with the hostage-taker, such as releasing hostages, disarming the hostage-taker, the hostage-taker's surrender, or calling in police forces.

To enable voice communication between the participant and the hostage-taker, the use of Natural Language Processing (NLP) technologies needed. As these technologies are largely founded on Machine Learning (ML) frameworks, the incorporation of a third software module to handle the simulation's "smart" features becomes necessary. Consequently, constructing this simulation demands a layered application architecture that orchestrate the three elements mentioned earlier (Figure 2).

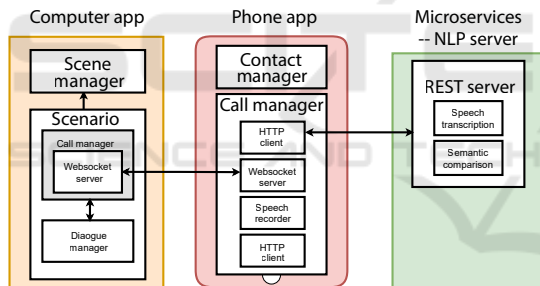


Figure 2: Negotiator app: fundamental concepts and technologies. Interactions between entities is describes in Figure 4.

2.1 Computer App

The computer application should feature three scenes: an introduction scene in 2D displaying the simulations's name and logo; a main scene where a police officer briefs the negotiator and provides textual information; and three dialogue parts where the negotiator contacts the hostage-taker, calms the situation, and resolves it. The negotiator receives updates on their job at the start of each part from the briefing officer. After the final conversation, the simulation concludes with a scene displaying newspaper headlines providing feedback on the ending.

2.2 Phone App

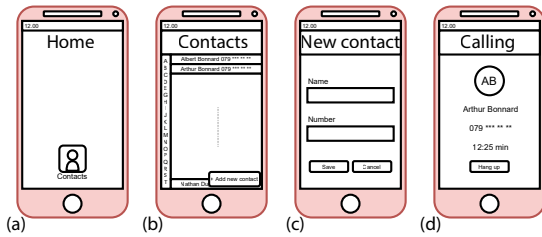


Figure 3: Interfaces of the phone app: (a) home screen, (b) contacts list, (c) menu to add a contact, and (d) phone call. Final renderings are presented in Figure 12.

The phone app mimics a current smartphone's interface, featuring a professional-looking home screen with essential features for the negotiator, such as a contacts app and a phone call function. The app consists of four key screens (Figure 3), designed to be simple for realism, ease of navigation. Avoiding unnecessary features decreases the risk of bugs and allows the participant to focus only on the task they need to accomplish.

2.3 Natural Language Processing

To enable users of the app to talk with the hostage-taker, their spoken words must first be converted into text using Speech-to-Text (STT) technology. Following this, the text is analyzed and matched against pre-determined options by employing Word Embeddings with text similarity techniques.

The conversation follows a tree-like structure (Cohen et al., 2014), offering the negotiator various pre-set responses. To help the participant, hints are provided on screen.

2.4 Call Procedure

When a call starts (Figure 4), the computer app picks a dialogue tree. It then sends the hostage-taker's pre-

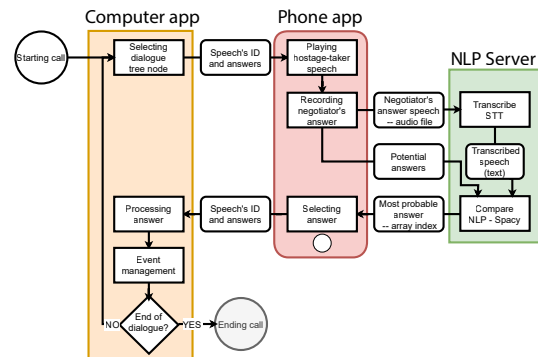


Figure 4: Flowchart of the process for natural interaction over the phone.

recorded speech and response choices to the phone app. The phone app plays this speech and records the negotiator's reply, sending it to the NLP server. The server determines the reply that semantically matches the negotiator's speech the best, it sends this choice back to the phone app, which then passes it on to the computer app. The computer app responds by moving to the next part of the dialogue tree based on this choice. This process repeats, moving through the dialogue tree until it reaches a point with no further options, at which time the computer app ends the call.

3 IMPLEMENTATION

3.1 Computer App



Figure 5: Main scene viewed by the negotiator.

3D World – The setting for the hostage scenario was crafted using the Unity game engine, establishing the main scene as a conventional crime scene featuring police cars and officers positioned behind barricades. Participants, assuming the role of a negotiator, are placed in a street section cordoned off by security barriers, immersing them directly into the hostage situation's environment. The negotiator has the ability to move within a limited area, leading them towards a table equipped with a laptop. Upon reaching the computer, the simulation officially kicks off as a SWAT officer arrives to brief the negotiator on the situation and outline their responsibilities (Figure 5).

The environment is divided into three sections: a building interior where the hostage situation unfolds, an intervention area in front of the building, and a navigation zone behind security barriers (Figure 6). ProBuilder and ProGrid tools in Unity were used to create this setup, providing urban development tools

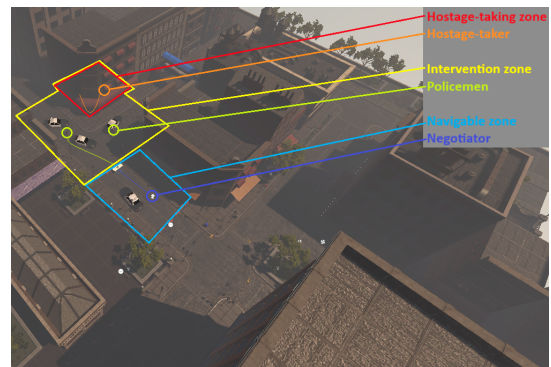


Figure 6: A view of the main scene from the top with the different zones and actors of the world.

and prefabricated components. The hostage area features the hostage-taker and hostages, with the former capable of animated movements, including approaching the entrance to surrender or disarm. Hostages can also flee the building. The intervention area includes two animated police officers, one of whom periodically offers verbal guidance to the negotiator. The navigation zone allows movement for the negotiator character, controlled from a first-person perspective, and contains a table with a laptop. A blue circle marks the simulation trigger area in front of the table, initiating the simulation when the negotiator enters it. Background sounds like police sirens and urban ambience deepen immersion.

Animations – The actions of both the police officer and the hostage-taker were recorded using motion capture technology, utilizing the Mocap Fusion software alongside equipment such as the HP Reverb G2 headset, Index controllers, and Vive trackers. The dialogue for the police officer was recorded in advance, and a lip-sync system was employed to synchronize the facial animations with the speech. The animation sequences are orchestrated through a state machine, allowing the scenario manager to regulate the character animations by initiating state transitions.

Starting of the Simulation – When the game begins, the computer application launches a local WebSocket (WS) service implemented with the help of the WebSocketSharp.Standard NuGet package. To kick off the simulation, the negotiator needs to move to a position in front of the table in front of the intervention zone. The intervention officer then approaches and asks the negotiator to scan a QR code displayed on the computer screen in front of them (Figure 7). This action serves as an “authentication” process with their service phone. This action establishes a connection between the phone and the computer application and initiates the briefing phase. The officer provides details about the suspect and prepares the participant for the call to the hostage-taker.



Figure 7: From left to right: QR code given by the policeman; view from the cellphone at the starting of the app.

Interactive Elements – The Heads-Up Display (HUD) is a see-through overlay on the screen that presents information to the participant (Figure 8). Instructions about controls and required actions are shown during the simulation. The HUD also displays suggested phrases for the participant to use during the conversation and badges that represents the progression of the simulation.



Figure 8: Heads-Up Display providing useful information for the user.

The negotiator can access the hostage-taker’s profile (Figure 9) that includes initial information compiled by the intervention team, such as a summary of the situation, the hostage-taker’s name, age, social status, spouse’s name, and phone number. This provides the negotiator with a preliminary understanding of the scenario and supplies a contact number for reaching out to the hostage-taker.

Situation	Notes
<p>A man holds 4 people hostage in a store on Third Street. He appears to be armed and threatens to attack the hostages. His motive is unclear.</p> <p>Suspect details : Name: Samuel Rodriguez Gender: Male Age : 35 Contact: 079 786 45 35 Married to : Laura Rodriguez</p>	<p>Write your notes here...</p>

Figure 9: File of the suspect where user can receive hints and take notes.

After reaching the second phase of the simulation, the laptop initially used to show the QR code (Figure 7) can also be used to gather more information about the hostage-taker. Participants can access the

police database to search by the hostage-taker’s name or other associated individuals. Additionally, they can explore a fictional social network called “SelfBook” to gain insights into the personality and background of the person they are negotiating with.

Interacting with the computer brings up a full-screen interface with a search bar designed for looking up police records. A navigation bar with tabs labeled “police files” and “SelfBook” lets users toggle between different pages. These pages are created in advance, saved as jpeg images, and integrated into the simulation, allowing for a realistic browsing experience without real-time internet connectivity (Figure 10).



Figure 10: Left: Police search and police file. Right: Social network search and social network wall.

Scenario and Dialogues – The ScenarioManager class oversees the simulation, serving as a bridge between the DialogueManager class and managing the simulation’s state variables. These variables track the status of the hostages, the hostage-taker, the negotiator, and the current phase of the dialogue.

A dialogue tree editor has been created, enabling crafting and adjusting dialogues. Nodes represent potential statements by the negotiator and replies from the hostage-taker, interconnected to provide response options influencing the operation’s success.

Following each dialogue segment, the Scenario Manager transitions to the next phase, with the intervention officer updating the negotiator. After the concluding dialogue, the Scenario Manager evaluates the negotiation’s effectiveness, leading to four possible outcomes: hostage-taker surrender, police assault without casualties, police assault resulting in the hostage-taker’s death, or a police assault with multiple casualties.

3.2 End Scene

The simulation concludes with a focus on a table displaying the next day’s newspaper, which reports on the hostage scenario. The final section of the article outlines the resolution of the event (Figure 11). To

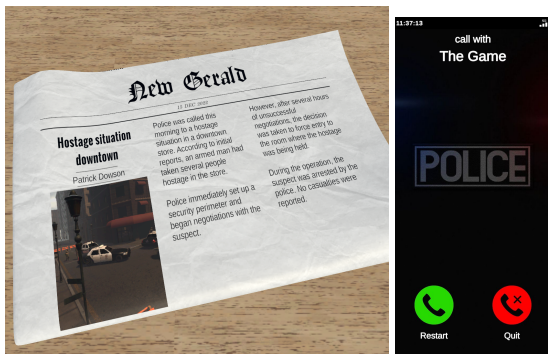


Figure 11: From left to right: The end scene explaining the end of the situation and the end screen on the cellphone.

preserve the simulation’s realism from start to finish, a decision was made against assigning a numerical score, reflecting the real-world context where negotiators do not receive quantitative evaluations of their performance. At this point, a message indicating the end of the simulation is transmitted to the phone app, and the mobile phone screen presents options for the user to either restart the simulation or exit it.

3.3 Phone App

Similar to the computer application, the phone app was also developed using Unity. This decision was taken early in the prototype development phase to investigate Unity’s capabilities for mobile app creation. The design of the app mimics a compact, secure operating system that could be utilized by law enforcement agencies.

QR Code – To establish a connection between the phone app and the computer app, the phone app requires the computer’s address. This is achieved by scanning the QR code (Figure 7), through which the phone app saves the address, connects to the computer app, and sends a message specifying the version to be used and to initiate the simulation. The ZXing library was used to decode the QR code, and the NuGet package WebSocketSharp. Standard was used to implement the WS client.

Main Menu, Contacts and Call – After establishing the connection, the phone app transitions to the main phone scene, designed to resemble a simplified mobile operating system tailored for law enforcement use, featuring a Contacts application. The interface is styled with a distinctive “police” motif as the backdrop, enhancing the device’s professional appearance. A header displays a live clock, a padlock symbol to signify system security, and a signal strength indicator to inform about the connection’s quality. The primary interface showcases the Contacts app, facilitating the saving of phone numbers and initiating calls –

Figure 12(a).

The design of the Contacts menu is intentionally straightforward, equipped with a green button for new contact entries and a red cross for navigating back to the home screen – Figure 12(b). Activating the add contact feature unveils input fields for the contact’s name and number, alongside a save option – Figure 12(c).

While on a call, the screen displays the contact’s name with a prominent button at the bottom to hang up – Figure 12(d). To enrich the call’s authenticity, background sounds such as footsteps, door movements, and muffled conversations are integrated, adding depth to the simulation experience.

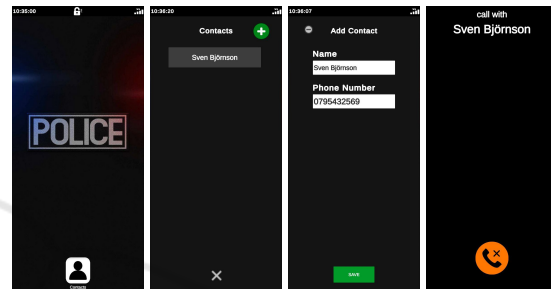


Figure 12: From left to right: (a) Main screen, (b) Contacts, (c) Filling new contact, and (d) Call.

3.4 Microservices

To manage the Machine Learning (ML) aspects of the simulation, a Python REST server has been established. This server launches concurrently with the computer application, ensuring seamless integration between the components. For verbal interactions with the hostage-taker, two key models are employed: a Speech-to-Text (STT) transcriber for converting spoken words into written text, and a semantic comparator to analyze and match the dialogue’s content.

Speech Transcription (STT) – Azure Speech Services was selected as transcriber for its good timing performances. While local models could potentially offer superior performance on computers equipped with higher processing capabilities, the cloud-based solution is chosen for this project. It ensures uniform performance across different computers running the simulation. However, note that a minimum internet connection speed of 10Mb/s is necessary to prevent any performance issues related to uploading audio files to the cloud.

Word Embeddings with Text Similarity – The negotiator’s spoken words are matched against possible replies within the dialogue tree. SpaCy is an open-source Natural Language Processing (NLP) tool that employs pretrained transformer models like

BERT (Devlin et al., 2018). It is designed to work effectively in both CPU and GPU setups, supporting platforms such as TensorFlow or PyTorch. In this project, spaCy is chosen as a local library for its ease of use and low demand on computational resources. The longest time taken to compare sentences averages at 0.02 seconds, which aligns with the simulation's requirement for real-time processing.

4 PERFORMANCES

The application's performance was analyzed in terms of timings. The figure below (Figure 13) illustrates the speech processing sequence of the application. The "End of Speech Detection" phase is also included in the total processing time but does not involve AI services. The sentence comparison step is not represented as the time for this step is negligible (0.03 sec).

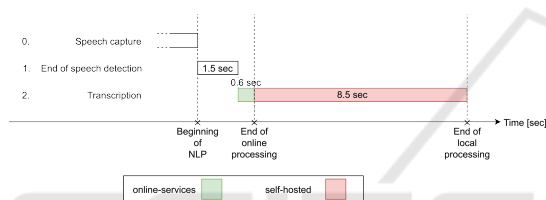


Figure 13: Timings sequence comparison between self-hosted and online-services solution.

Depending on the sound level of the environment in which the simulation is played or the volume level thereof, the end of speech is sometimes not detected. Another issue identified after implementing the final product is an empty transcription. Azure Speech Services has two modes of operation: stream and one-shot. The one-shot mode was used, and if it does not detect speech in the first two seconds of the audio file, it considers it as empty. This proved to be problematic because participants, sometimes not feeling comfortable or needing time to think about what to say to the hostage-taker, saw all their responses considered null. For further work, the stream mode is recommended.

5 DISCUSSION

The development and deployment of the simulation, with its integration of a 3D environment and natural language processing dialogues, require a comprehensive approach that goes beyond the scope of conventional computer science. The simulation's intricate framework, which includes a mobile app, a Python server for NLP functionalities, and a detailed 3D world, introduces various technical and design

hurdles. While this complexity enhances the user experience, it also complicates distribution due to the elaborate setup and deployment processes involved.

Incorporating a 3D element not only boosts the simulation's realism and visual appeal but also imposes certain limitations, such as the demand for substantial computing power and restricted content creation options for facilitators. The Unity game engine plays a pivotal role in creating the 3D setting and managing the simulation's logic, striking a balance between technological innovation and usability. Nonetheless, the requirement for a powerful graphics processor highlights the importance of considering the simulation's compatibility and accessibility across different platforms. The use of a VR headset was not considered for this simulation as it would interfere with the normal usage of the cellphone. Moreover, the aim of this project is also to demonstrate the use of the cellphone as an immersive VR tool.

Opting for a local network solution over real phone number interactions addresses some practical challenges related to incorporating mobile phone features. This approach simplifies the technological requirements by leveraging WebSockets but also points to the need for improvements in remote connectivity and user experience.

The deployment of an NLP pipeline, including speech-to-text transcription and text comparison, showcases the application of advanced machine learning technologies. However, the substantial processing power these transformer-based models demand presents a bottleneck in performance, particularly on less equipped devices. Resorting to cloud-based services to bypass these hardware limitations raises issues about model control, the reliability of external services, and dependence on network quality, despite offering faster processing times overall.

6 CONCLUSION

This project resulted in the creation of a hostage-taking simulation prototype for Negotiator training including a 3D simulation, a phone application that allows for natural dialogue with the hostage taker, and a Python server providing Natural Language Processing (NLP) services. The implementation of this simulation showcases the complexities and challenges inherent in creating an immersive and technologically advanced training tool. This research has shown that it is possible to use a smartphone as an interactive tool for naturally (verbally) conversing with an autonomous agent in a simulation. This innovation paves the way for using the phone not only in the

specific context of police negotiation but for all negotiation situations such as: Business and Commerce, Diplomacy and International Relations, Law and Mediation, Human Resources, Healthcare, Entertainment and Media, Politics, Education, Environmental Conservation. Beyond negotiation, all computer simulations or video games involving autonomous agents can benefit from this research by including the phone as an interaction tool in their toolkit. To take this a step further, we have developed an LLM (AI) model for the autonomous agent and user tests – see twin paper (Monaco et al., 2024).

Limitations in terms of timing, particularly during speech capture and the transcription of the participant's speech are still to be overcome. In the current state, the performance of standard computers does not allow the use of local models and to have a smooth and uninterrupted discussion, which is why online services had to be used. This choice can pose a problem in the case of simulation exchanges containing sensitive or confidential personal data. The use of external services also poses issues in terms of availability and bandwidth. Further research into the use of NLP models on smartphones could yield results that overcome these limitations; the limiting factor would undoubtedly be the quality of the phone itself.

ACKNOWLEDGEMENTS

We are deeply thankful to the University of Applied Sciences and Arts of Western Switzerland (HES-SO) and the University of Skövde (HIS) for their generous provision of essential equipment and invaluable knowledge, which greatly contributed to the success of this project. We extend our heartfelt appreciation to Diego Villagrasa and Dylan Canton for their invaluable assistance during the implementation process, their expertise and dedication significantly enhanced the project's development. Additionally, we express our gratitude to Colin Lavanchy for lending his voice to portray the hostage-taker, enriching the simulation with authenticity and depth.

REFERENCES

- Baines, L. (2008). *A Teacher's guide to multisensory learning: Improving literacy by engaging the senses*. ASCD.
- Bel-Enguix, G., Dediú, A. H., and Jiménez-López, M. D. (2009). *A Dialogue-Based Interaction System for Human-Computer Interfaces*, pages 55–65. Springer Berlin Heidelberg, Berlin, Heidelberg.
- Bienstock, J. and Heuer, A. (2022). A review on the evolution of simulation-based training to help build a safer future. *Medicine*, 101:e29503.
- Bunker, W. M. (1978). Training effectiveness versus simulation realism. In Beiser, L., editor, *Visual Simulation and Image Realism I*, pages 76–82.
- Cohen, A., Gottifredi, S., García, A. J., and Simari, G. R. (2014). A survey of different approaches to support in argumentation systems. *The Knowledge Engineering Review*, 29:513–550.
- Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *North American Chapter of the Association for Computational Linguistics*.
- Hochmitz, I. and Yuviler-Gavish, N. (2011). Physical fidelity versus cognitive fidelity training in procedural skills acquisition. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 53:489–501.
- Lobo, I., Rato, D., Prada, R., and Dignum, F. (2022). *Socially Aware Interactions: From Dialogue Trees to Natural Language Dialogue Systems*, pages 124–140. Springer International Publishing.
- MacLean, S., Geddes, F., Kelly, M., and Della, P. (2019). Realism and presence in simulation: Nursing student perceptions and learning outcomes. *Journal of Nursing Education*, 58:330–338.
- Monaco, P.-B., Backlund, P., and Gobron, S. (2024). Interactive storytelling apps - increasing immersion and realism with artificial intelligence? In *14th International Conference on Simulation and Modeling Methodologies, Technologies and Applications (SIMULTECH 2024)*. SCITEPRESS.
- Monaco, P.-B., Villagrasa, D., and Canton, D. (2023). The negotiator. In *Proceedings of the 8th International Conference on Gamification & Serious Games (GSGS'23)*, pages 94–97. HE-Arc, HES-SO.
- Salas, E., Rosen, M. A., Held, J. D., and Weissmuller, J. J. (2009). Performance measurement in simulation-based training. *Simulation & Gaming*, 40:328–376.
- Simpson, H. and Oser, R. L. (2003). Evaluating large-scale training simulations. *Military Psychology*, 15:25–40.
- Valiente, D., Payá, L., de Ávila, S. F., Ferrer, J.-C., Cebollada, S., and Reinoso, Ó. (2019). Active learning program supported by online simulation applet in engineering education. In *SIMULTECH*, pages 121–128.