VOICE TRAINING AS A KEY COMPETENCE FOR STUDENTS IN TEACHER TRAINING – BENEFITTING FROM A VIRTUAL REALITY CLASSROOM IN HIGHER EDUCATION. PART 2 - TECHNICAL IMPLEMENTATION

Dr. Ulrike Nespital [1]
Gerald Czerney [2]
[1, 2] Justus Liebig University, Giessen, Germany

ABSTRACT

The project "Making voice and Presence a Virtual Realistic Experience" is funded by “HessenHub - Network digital university teaching Hessen”. At the Center for Foreign Language and Occupational Competencies (German: ZfbK) at the University of Giessen, a teaching concept for student teacher training was developed that includes both the learning and use of a physiologically sound voice and offers transfer to professional practice. For this purpose, a virtual classroom featuring avatars of noisy school children was developed. The teaching concept includes exercises on breathing, posture, articulation and physiological voice enhancement at the syllable, word, sentence and text levels, up to spontaneous speech. Utilizing virtual reality (VR) headsets, student teachers practice these exercises in a realistic classroom environment under the guidance of a speech scientist. The ultimate objective is to enable future teachers to naturally and physiologically modulate their voice in professional settings.

Technical implementation relied on contemporary, agile software development methodologies. After extensive online research, specific software tools were chosen for successful project execution. This paper discusses both the technical approach and the software techniques and tools employed.

Keywords: *voice training, virtual reality headset, technical implementation*

INTRODUCTION

The objective of this article is to further explain the methodology employed for the technical implementation of the project titled “Voice Training for Student Teachers Using VR Headsets.” It also describes the implemented software components and outlines the project’s step-by-step realization and progressive value-added stages.

Furthermore, this article presents the fundamentals of virtual reality (hereafter abbreviated as VR) and the technical background relevant to the project. Agile software engineering methods were adopted, with Ian

Sommerville's "Software Engineering," 10th edition (2018), serving as the foundational text [1].

Dr. Nespital evaluated various VR headsets to select appropriate hardware for the project. The choice fell on Pico Neo 3 Pro [4]. Decisive criteria in favour of the unit were:

- lightweight construction of the device
- capability to run standalone applications and no need for physical connections like cables to a remote host

The VR headset operates on an Android operating system, commonly used by various smartphone manufacturers, which allows for convenient software installation via APK files.

SOFTWARE ENGINEERING INTRODUCTION

Internet research indicated the absence of any existing software applications meeting the project's specific requirements, necessitating bespoke software development. Accordingly, a software development project was planned.

As Sommerville notes in his 2018 publication, software systems, being abstract and intangible, can quickly become complex, difficult to navigate, and hence challenging to maintain. To manage this complexity, the use of structured software engineering methods is essential [1].

The core activities include [1]:

- Software specification: Definitions of the software requirement and the framework for its deployment.
- Software development: Building the software using one or more software tools.
- Software validation: Ensuring that the created software corresponds to the requirements and functions error-free.
- Software maintenance: Further development and refactoring (revision of the product) of the software in order to adapt it to changing requirements.

The following product characteristics were considered when planning the software project [1]:

- Acceptability: The user of the software must be able to operate and use it comfortably and easily.
- Reliability: The software must be created to be reliable, information and operationally secure.

- Efficiency: The software shall be designed to be resource efficient in terms of system resources such as microprocessor utilization and memory usage.
- Maintainability: The software shall be developed in such a way that changes can be easily inserted, and the created software can be easily imported into the target hardware.

SELECTION OF THE PROCEDURE MODEL

Various methods and paradigms exist for software development, often described in procedure models. Sommerville discusses several such models, two of which were relevant for our project [1]:

- The Waterfall Model: Encompasses sequential phases such as software specification, development, validation, and enhancement.
- Incremental Development: Iteratively runs through the steps of software specification, development, and validation until an initial product version is satisfactory.

In the Waterfall model, each phase must be completed before the next one begins. However, in practicality, these phases often overlap. Incremental development allows for greater flexibility, letting the project adapt to changes in requirements. Given the project's initial lack of detailed requirements, the incremental approach was adopted to allow for rapid and flexible responses to any changes [1].

In incremental development [1], the stages of software specification, software development, and software validation are executed in similar fashion to traditional models. However, the focus here is on defining small, manageable software packages that are individually specified, developed, and validated. Following each validation step, the project's requirements are re-evaluated based on the current state of the software, allowing for necessary adjustments

Since only rough requirements were available in this software project, the incremental approach was chosen in order to react as quickly and flexibly as possible to changes in requirements.

SOFTWARE SPECIFICATION

The project aimed to create a virtual world featuring a classroom environment with avatars that can move and produce sound. From this overarching goal, the following functional requirements were distilled for software implementation:

- Virtual classroom featuring school children avatars
- Activation buttons for avatars
- Sound files and avatar movements triggered upon activation

- VR headset view projection via an external projector
- Reusability of existing development environments and assets
- Application of simple software development methods
- Scalable avatar control
- Quick VR headset update capabilities

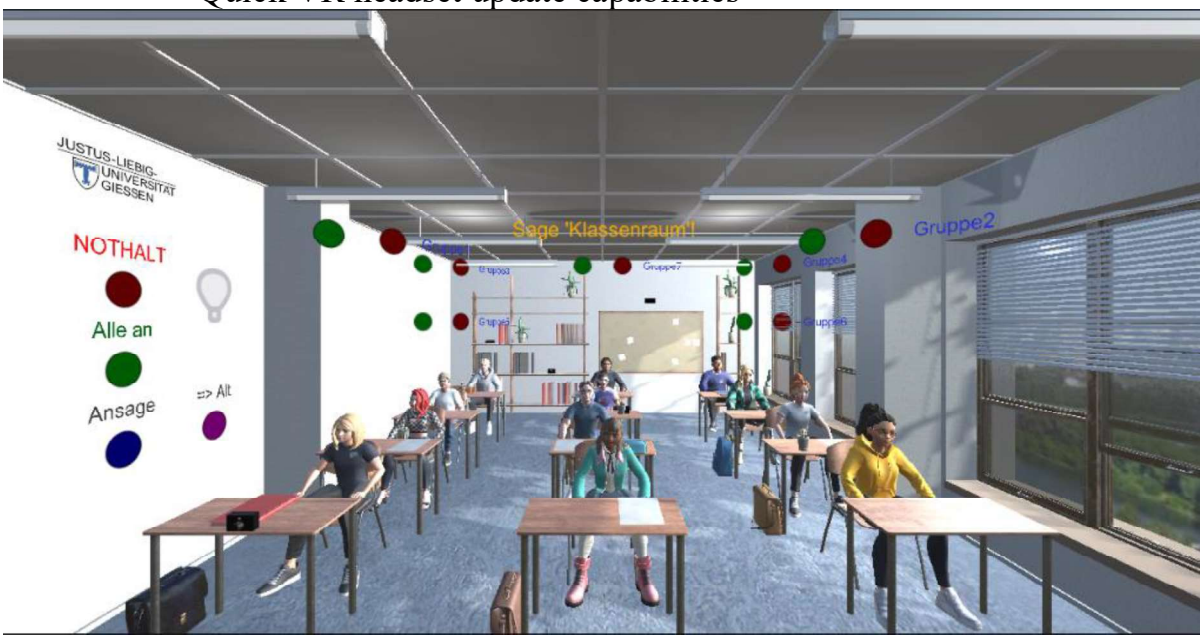

Fig. 1. *Virtual classroom with avatars and control buttons*

Product characteristics should be considered as follows:

Acceptability:

- User-friendly buttons should facilitate avatar activation and deactivation.
- Avatars must appear lifelike.

Reliability:

- The software should operate without errors.

Efficiency:

- Seamless performance is essential - without the user noticing judder or anything similar in the VR world.
- Assets and software development tools should be used that incur as few costs (licensing costs) as possible or are reusable.

Maintainability:

- Changes should be straightforward to implement.
- Audio files should be easily integrated.
- New versions of the software application should be easy to install on the target hardware (VR headset) without much effort.

SOFTWARE DEVELOPMENT AND SOFTWARE COMPONENTS

The conception phase took up a significant amount of time, primarily because the development team consisted of a single individual. As a result, the responsibility for the feasibility study could not be distributed. Several development tools were evaluated with the support of various vendors to determine their suitability for developing the Virtual Reality (VR) environments needed for the seminars. The feasibility study concluded that Unity VR was the most appropriate tool for the task. Its ease of learning, coupled with its flexible graphical and script-based development interfaces, made it an ideal choice. Additionally, Unity VR allows easy integration of various assets like avatars, their animations, and sound files—all crucial components for our software solution [7] [2].

Fig. 2. *Unity's real-time 3D development engine*

For the VR world's foundation, pre-developed assets like a classroom setting were purchased from Unity's asset store [7]. These assets included a classroom environment complete with tables, chairs, and other class-specific utensils.

For the creation of the avatars, the online tool Readyplayer Me [6] was best suited for our purposes. It is very easy to use, has many options to equip the avatars with many different accessories. You can customize the avatars' face shape, hair colors, hairstyles and much more. Additionally, it allows for the importation of real-person portraits to generate avatars. The created avatars are fully compatible with the Unity VR development environment and can be easily integrated therein [7].

For the avatars to move, they must be combined with software extensions called animations. For linking the avatars with already available animations, the

tool [3] was used. This platform offers a substantial library of animations that can be associated with avatar-specific files previously imported into the tool. Once modified, these files are integrated back into the Unity VR environment and can be paired with one or multiple avatars [7].

For the auditory aspect, sound files were recorded using resources from the private environment of the authors. These sound files could be easily integrated into the VR development environment.

A crucial aspect of the feasibility study was the licensing of components. While many vendors impose fees for commercial, educational, or personal use of their products, the majority of the tool providers referenced in this project were amenable to granting complimentary usage licenses. This generosity was largely because the software was intended for non-commercial educational purposes within a university setting. The sole exception was the animation vendor, who imposed a monthly licensing fee. The virtual reality (VR) world assets were purchased outright, thus eliminating any concerns over licensing for those particular components. As for the audio files, we ensured compliance with legal requirements by obtaining signed consent statements from each respective speaker.

The initial implementation of the software solution equipped participants with a dynamic training environment. In this virtual setting, users could interact with various groups of avatars modeled after noisy school children. These avatar groups were linked to different sources of loud noises, allowing participants to adjust the noise intensity in a scalable manner. Initially, participants engaged with quieter avatar groups to establish a baseline for vocal training. As the training progressed, they had the option to introduce increasingly noisy avatar groups, adding layers of complexity to the vocal challenges. This approach intensified the training experience over the course of the seminar. Additionally, participants had the flexibility to turn off individual avatar groups or all of them simultaneously via designated "stop" buttons.

Expansion stage 1

To enhance the realism of the virtual environment, we activated a feature known as "spatial audio," which is natively supported by the Unity development environment [7]. Once connected to the sound sources within the environment, this feature dynamically modulates audio levels, making sounds louder as one approaches and quieter upon moving away.

Expansion stage 2

Feedback collected from student teachers after the first semester revealed a key shortcoming: although the seminar enabled them to practice vocal modulation

against noisy avatars, the lack of reactive behavior from these avatars led to an unsatisfactory experience.

To address this issue, we sought a speech recognition solution that could offer reliable command recognition without significantly taxing the system's computational resources. Our research identified "Rhino" from Picovoice Inc [5] as the most fitting choice. The company provided easily comprehensible C# script examples, which were rapidly integrated into the Unity development environment for testing [7].

We customized these scripts to meet our specific requirements. Now, participants can issue vocal commands to the noisy avatars, prompting them to quiet down. This enhancement has resulted in a more engaging and efficient training environment for voice modulation.

TECHNICAL LIMITATIONS

During the development and testing phases, we observed that the software's performance decreased as the number of avatars in the virtual environment increased, manifesting as black areas within the user's field of vision. Consultation with Unity's support team [7] revealed that the microcontroller within the VR headset reached its performance limits under these conditions. Subsequent tests indicated that a maximum of approximately 20 avatars could coexist within the same VR environment without performance degradation.

The application largely functions without an internet connection, save for a brief moment required for the voice recognition software, Rhino, to validate its authentication code with Picovoice's server [5].

CONCLUSION

Every project typically begins with a vision—a goal or an expectation to be fulfilled. In this case, the vision was to create a virtual environment where student teachers could train their voices under expert guidance. Thanks to close collaboration between Dr. Ulrike Nespital, the seminar leader, and Gerald Czerney, the software engineer, we succeeded in bringing this vision to life.

Extensive online research led to efficient and user-friendly tools, which significantly simplified the technical execution of the project. Among these, the incorporation of the voice recognition system marked a significant advancement, enabling student teachers to gain instant feedback by silencing noisy avatars. It is crucial to note, however, that expert evaluation from speech scientists remains indispensable as the system does not assess vocal quality.

The software solution has garnered positive acclaim for its robust performance, stability, and immersive experience, as evidenced by numerous

commendations received from seminar participants (see Nespital/Czerney article from the NORDSCI 2023 Conference).

Looking ahead, a promising avenue for further development would be to implement an algorithm capable of evaluating whether the voice is being used in a manner that minimizes strain, thereby adding another layer of sophistication to the training environment.

REFERENCES

[1] Sommerville, I. Software Engineering, 10th edition Germany, 2018;

[2] Theis, T., Einstieg in Unity, Germany, 2017

[3] https://www.mixamo.com/#/

[4] https://www.picoxr.com/de/products/neo3-pro-eye

[5] https://picovoice.ai/

[6] https://readyplayer.me/de/avatar

[7] https://unity.com