SMART VIRTUAL ENVIRONMENT FOR DESIGNERS WITH MOTOR LIMITATIONS

Dr. N. Muni Nordin

ABSTRACT:

Traditional CAD modeling software relies heavily on input devices such as keyboards and mice for the creation of three-dimensional (3D) models, which can be challenging for individuals with motor disabilities. These tasks require fine motor control, which may be difficult for users who experience limited mobility in their limbs—especially their arms and fingers—due to injury or congenital conditions. To address these accessibility barriers, this paper presents a virtual reality (VR)-based interface that enables users with motor disabilities to create simple 3D architectural models. The proposed system leverages head-gaze tracking as a primary input method in the VR environment, allowing users to manipulate and scale 3D objects, such as simplified building models, without needing to use their hands. Additionally, the VR system incorporates head tilting for navigation, with the user seated in a revolving chair, eliminating the need for any limb movement. This approach minimizes physical effort and improves accessibility for individuals with limited motor abilities. Developed using the Unity game engine, the system offers two variations of the VR interface, each with a different button layout for constructing cuboidal volumes that represent architectural structures. The two versions were tested with 32 participants, with performance assessed based on task completion time. In addition to these objective measures, participants also provided feedback on perceived effort and visual clutter. Following these tests, retrospective feedback sessions further informed the evaluation of each interface variant. The results indicated that Variant 1 performed better overall, showing higher usability and lower effort requirements. In addition, the paper proposes a framework for incorporating artificial intelligence (AI) into the VR interface, creating an intelligent and adaptive system that can adjust in real time to users' capabilities and challenges. This AI-driven approach could lead to a more robust, gaze-based VR tool with broader functionality, enabling individuals with motor disabilities to undertake more complex 3D modeling tasks in the future.

Index Terms: *Virtual Reality; design tool; intelligent interface; gaze-based input; artificial intelligence.*

Reference to this paper should be made as follows:

Dr. N. Muni Nordin, (2024), "Smart Virtual Environment for Designers with Motor Limitations", Vol. 12, No. 3, pp. 69-82, DOI 10.30696/IJEEA.XII.III.2024. 69-82.

Biographical notes:

Dr. N. Muni Nordin *is born in Malaysia, done BS in Information Technology and MS in information Technology, done PhD in data security form IUKLM, University Malaysia.*

1. INTRODUCTION

The term "virtual reality" merges two contrasting ideas—“virtual,” meaning artificial, and “reality,” meaning the real world—emphasizing the distinct experience it offers. Virtual Reality (VR) is a human-computer interface that immerses users in a simulated environment, encouraging them to temporarily accept it as real. The VR experience is built around two key principles: immersion and interactivity. Immersion allows users to focus deeply on virtual stimuli, blocking out external distractions, which creates a strong sense of presence in the virtual world. Interactivity, meanwhile, enables users to engage with and influence events in the environment, making the experience more dynamic and responsive. To deliver this experience, VR systems utilize devices such as head-mounted displays (HMDs) that visually immerse users in the virtual setting, hand controllers that allow interaction with objects, and haptic gloves that simulate the sensation of touch, enhancing the realism. These technologies work together to provide a fully immersive and interactive environment that feels convincingly real, offering users a rich, engaging virtual experience.

1.1 Application of VR

Virtual Reality (VR) has become a versatile tool in various fields today, including military and industrial training, as well as medicine. VR is also gaining recognition as an effective tool for rehabilitation, helping individuals with diverse disabilities. It has been successfully applied in the rehabilitation of Parkinson’s disease, stroke, and orthopedic conditions, as well as for balance training and activities of daily living. In these applications, VR-based systems allow users to practice motor skills in a virtual setting, which can then be transferred to real-world environments. Additionally, VR-based rehabilitation has proven to be both safe and effective for children with sensorimotor deficits, providing an engaging medium for therapeutic interventions. Apart from its wide adoption in gaming and entertainment, VR is increasingly being used in creative sectors such as architecture and design. It enables designers to visualize virtual environments and assists in the planning and creation of complex systems like hospitals. In architectural education, VR is an immersive tool for teaching spatial geometry, helping students better understand design concepts. Moreover, VR has been integrated with Building Information Modeling (BIM) to improve design processes, allowing professionals to interact with 3D models and make adjustments before construction begins. For example, a study by Su et al. explores a VR application that combines Geographic Information Systems (GIS) and BIM technology, while also noting the challenges of incorporating VR into architectural design workflows. Similarly, research by Sait et al. highlights the use of VR in designing frameworks for daylight-responsive smart buildings, showcasing its potential in smart architecture. As VR technology advances, its applications continue to expand, making it a powerful tool not only for rehabilitation but also for creative design, enhancing both visualization and real-world implementation across various industries.

1.2 A survey of the literature on gaze-based interfaces

Eye gaze tracking technology allows for the estimation of a user's point of gaze on digital interfaces, capturing their eye movements in relation to head position. Recent advancements have made it possible for users to control graphical user interfaces (GUIs) through their gaze using devices like infrared-based eye trackers, webcams, and head-mounted displays (HMDs). This integration of eye-tracking technology enables effective communication for individuals with severe speech and motor impairments (SSMI), as illustrated in a study by Agarwal et al. (2019), which evaluated webcam-based eye gaze interfaces. Gaze-controlled interfaces have also been explored for scenarios where users' hands are occupied with primary tasks, such as pilots and drivers, allowing them to interact with secondary controls without diverting their attention. For individuals with motor impairments, who cannot utilize peripherals like a mouse, keyboard, or joystick, gaze tracking offers a valuable alternative. In the realm of Virtual Reality (VR), various input modalities linked to HMDs include gaze-based inputs, hand controllers, voice recognition, hand gesture detection, and motion sensors like accelerometers. These input methods significantly enhance the interactive experience within VR environments. Gaze interactions in VR can be based on eye movements or head orientation, each presenting distinct advantages and limitations. Many low-cost VR solutions, such as Google Cardboard, employ simpler input methods like head-gaze or on-device buttons for simulating clicks. Typically, head-gaze input is based on the direction in which the user’s head is pointed rather than tracking pupil movement, which is more common in advanced eye-tracking systems. This type of input is particularly useful for tasks requiring aiming, while selection or triggering tasks often rely on dwell-time methods (where the user’s gaze lingers on an object) or controller clicks. The benefits of eye gaze-based interactions in VR and augmented reality (AR) include improved interaction speed, decreased physical task load, and reduced head movement, especially in interfaces with larger fields of view. A study by Qian and Teather (2017) demonstrated that head-only input methods outperformed eye-only and eye-with-head methods in terms of accuracy and error rates, suggesting that while eye-tracking technology is valuable, head-gaze can offer better control for specific tasks in virtual environments.

Further studies support this notion. Sidenmark and Gellersen (2019) found that combining eye and head tracking provided greater flexibility and control for gaze-based pointing and selection tasks in VR. The integration of these two methods can enhance user precision and adaptability, creating a more seamless and efficient interaction experience. In contrast, ray-casting-based controller interactions—where users direct a virtual ray from their controller to objects—have been shown to offer faster and more intuitive interaction compared to gaze-dwell or gaze-with-trigger methods for object selection. Research indicates that while gaze-based methods can reduce physical effort, they may lead to eye fatigue, particularly during prolonged focus on tasks or in interfaces with smaller targets. Mobile VR platforms, which usually employ head-gaze as the main interaction method, are often more affordable than high-end HMDs with integrated eye-tracking hardware. These mobile-based systems have been applied in various contexts, such as developing memory games that assess user interest and satisfaction through parameters like background color, field-of-view range, feedback systems, and multi-dimensional information transfer. Additionally, mobile VR applications have facilitated virtual shopping experiences by combining head-gaze input with speech recognition for navigation and item selection. Gaze-based interactions are also being investigated for 3D object manipulation in VR. A study by Yu et al. (2021) examined four techniques that combine eye gaze with hand input for manipulating 3D objects, primarily focusing on translation tasks (moving objects through space). The researchers suggest that gaze-based input methods may be particularly effective for interacting with distant objects in larger virtual environments rather than those within arm's reach. They also indicate that head-gaze, a cost-effective alternative to eye tracking, could be utilized for object manipulation, and that new gaze-based methods for scaling and rotating objects should be further explored to enhance interactivity and usability in VR systems. In the context of Augmented Reality (AR), gaze-based object manipulation has also been researched. A study involving the HoloLens AR device indicated that users preferred head-gaze input in conjunction with a clicker device for interacting with virtual objects. This combination provided a more intuitive and efficient experience for tasks involving scaling, rotating, and translating objects in AR, reinforcing the idea that head-gaze input, while not as precise as eye-tracking, is a suitable alternative for manipulating objects in both VR and AR environments. This paper employs head-gaze input for scaling 3D objects in a VR setup with at least three degrees of freedom (DoF), opting for this approach over the use of expensive VR HMDs with integrated eye-tracking systems. This method aligns with the broader trend of exploring simpler yet effective input techniques for users with limited mobility, making VR technology more inclusive and adaptable to a wider range of individuals. As VR and AR technologies continue to advance, further investigations into gaze-based interactions, including both eye-gaze and head-gaze methods, will likely unveil new opportunities to enhance user experience and accessibility in virtual environments.

1.2 Intelligent and Adaptive User Interfaces

The paper also investigates the concept of Intelligent User Interfaces (IUI) to assess the feasibility of developing user-centric Virtual Reality (VR) applications aimed at 3D modeling tasks, particularly for users with limited mobility or motor impairments. Maybury (1998) characterizes IUIs as advanced human-machine interfaces that function to act, represent, and analyze various models related to the user, the domain, the task at hand, discourse, and media. The main objective of IUIs is to enhance the efficiency, effectiveness, and naturalness of human-machine interactions, thereby making technology more accessible to a wider audience. Typically, IUIs adapt their interface content based on the user’s information and contextual environment. Their development is interdisciplinary, incorporating fields such as artificial intelligence (AI), user modeling, psychology, and human-computer interaction (HCI). The terms “adaptive user interface” (AUI) and IUI are frequently used interchangeably, referring to interfaces that automatically modify elements by monitoring user activity and accommodating changes in the user’s skills, preferences, or knowledge over time.

Rissland (1984) asserts that an interface is considered "intelligent" if it can access comprehensive information regarding the user, their tasks, the relevant domain, available tools, interaction modalities, and evaluation metrics for those interactions. Machine learning (ML) techniques are increasingly applied to create IUIs by predicting user models based on past interaction data, even when faced with challenges like incomplete datasets, noisy information, or non-numeric data types. Within the context of VR, various studies have concentrated on predicting user intent for interaction tasks, such as object selection, by analyzing gaze dynamics through logistic regression. Moreover, VR scenarios have been crafted for emergency response training, particularly for professionals like firefighters, while also evaluating Intelligent User Interfaces that deliver task-relevant information through VR headsets. As detailed in sections 3.2 and 3.3 of this paper, a preliminary supervised ML model has been utilized to ascertain the optimal timing for updating a VR application’s interface configuration based on user activity and feedback. This approach not only enhances inclusivity for users with disabilities but also expands the potential for integrating intelligent technologies into everyday tasks.

1.2 Aim of the paper

This paper focuses on developing an intelligent virtual environment that assists individuals with reduced motor abilities in performing 3D modeling tasks with minimal effort. To achieve this goal, two specific objectives have been outlined:

1. **User Interface Variations Development:** The first objective is to design two variations of a user interface (UI) for a preliminary design tool in a virtual reality (VR) environment, aimed at enabling basic 3D object manipulation using head-gaze as the primary input method. The usability of these UI variations will be evaluated through performance metrics, including task completion times, alongside user-reported feedback regarding usability and overall satisfaction.
2. **Methodological Framework for Intelligent Interface:** The second objective involves establishing a methodological framework for an intelligent and adaptive user interface that incorporates artificial intelligence (AI) technologies. This framework will be crafted to effectively meet user needs while seamlessly integrating essential functionalities within the VR application.

To meet these objectives, the paper initially examines the interaction challenges experienced by users with the two VR model variants, each featuring different UI configurations. The insights gathered from this analysis will guide the development of a user-centric intelligent interface that enhances accessibility and usability for individuals with motor impairments.

2. METHODOLOGY

The methodology outlined in this paper, illustrated in Fig. 1, comprises a systematic series of steps aimed at developing and evaluating a gaze-based virtual reality (VR) model for 3D modeling tasks. The process begins with the creation of two distinct variants of the gaze-based VR model, each designed with different button configurations tailored for basic 3D modeling. The next step involves testing these two VR model variants with a group of 32 participants to gather comprehensive feedback and performance data. After the testing phase, the collected data is analyzed in detail, incorporating both performance metrics—such as task completion times—and self-reported metrics that reflect the users’ experiences and perceived effort. A crucial aspect of this methodology is the prediction of the effort exerted by users through the use of an artificial intelligence model, specifically a neural network. This AI model will be integrated into a strategy aimed at developing a responsive VR interface that can adapt to user needs. Ultimately, the paper proposes a comprehensive methodological framework for creating adaptive and intelligent interfaces, grounded in user data and attributes. The following sub-sections will provide detailed information about the two VR model variants, including aspects of gaze detection, button activation, navigation within the virtual environment, and the experimental setup for user testing.

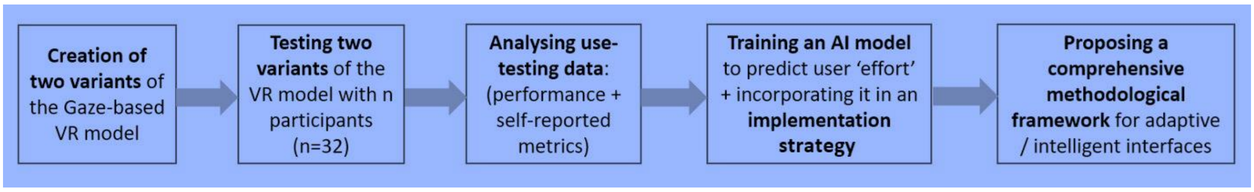
Creation of two variants of the Gaze-based VR model
Testing two variants of the VR model with n participants (n=32)
Analysing use-testing data: (performance + self-reported metrics)
Training an AI model to predict user 'effort' + incorporating it in an implementation strategy
Proposing a comprehensive methodological framework for adaptive / intelligent interfaces

Fig. 1. General methodological representation.

2.1. Development of the VR Model and its Two Iterations

In this article, the terms "VR model" and "VR application" are used interchangeably to describe the proposed gaze-based virtual reality application intended for head-mounted displays (HMDs). The VR environment simulates a conceptual site where users can construct and customize a hypothetical city. This immersive setting consists of two main components: a) **'road'**, which acts as a navigational pathway, allowing users to position themselves to manipulate geometries representing buildings, and b) **'building blocks'**, depicted as cuboids arranged along either side of the road, mimicking real city structures. These building blocks can have their heights adjusted by gazing at specific buttons within the virtual environment.

Two different versions of the VR model, referred to as variant 1 and variant 2, have been developed using the Unity game engine, with variations in the button layouts for height adjustments of the virtual buildings. An Android application has also been created using Unity's build feature, enabling users to access the VR model through Google Cardboard. However, during user testing, a Windows Mixed Reality headset featuring six degrees of freedom was utilized to provide a more immersive experience.

In **variant 1**, the buttons for adjusting the height of the building extrusions—labeled **'Build Up'** and **'Build Down'**—are consistently visible to users in designated zones of the virtual space. These buttons are arranged vertically, with the **'Build Up'** button located above the **'Build Down'** button on the front faces of each building, positioned just below the user's eye level (see Fig. 2a). The buttons are activated when the gaze pointer is within their designated area and deactivate when the gaze pointer moves outside of this zone (as illustrated in Fig. 3a). This design fosters an intuitive interaction model, enabling users to engage with the VR environment seamlessly.

In **variant 2**, the buttons for adjusting the height of building extrusions in the virtual environment are seamlessly integrated into hamburger buttons located on the front faces of each building (refer to Fig. 2b). The main aim of using hamburger buttons is to assess the usability of the gaze-based VR application when multiple functions are contained within a single button, thus maintaining a clean and uncluttered user interface (UI).

These hamburger buttons become active when the user's gaze hovers over their designated area while navigating through the virtual space. Once activated, a secondary button labeled **'Build'** appears, offering users additional options. Gazing at the **'Build'** button triggers the appearance of two tertiary buttons, labeled **'Up'** and **'Down,'** which allow for adjustments to the height of the building extrusions.

After the secondary and tertiary buttons are displayed, they will collapse back into the hamburger button when deactivated, which occurs when the gaze pointer moves away. The hamburger button indicates its active state by changing color when the secondary or tertiary buttons are expanded. To revert the hamburger button to its original state, the user simply needs to gaze at it again.

The sequence of actions related to the activation and deactivation of the hamburger, secondary, and tertiary buttons is depicted in Fig. 4. Additionally, screenshots illustrating the button activation sequence from the user's perspective in variant 2 are provided in Fig. 3b. In this configuration, the tertiary buttons are specifically responsible for altering the building heights in either the upward or downward direction when selected.

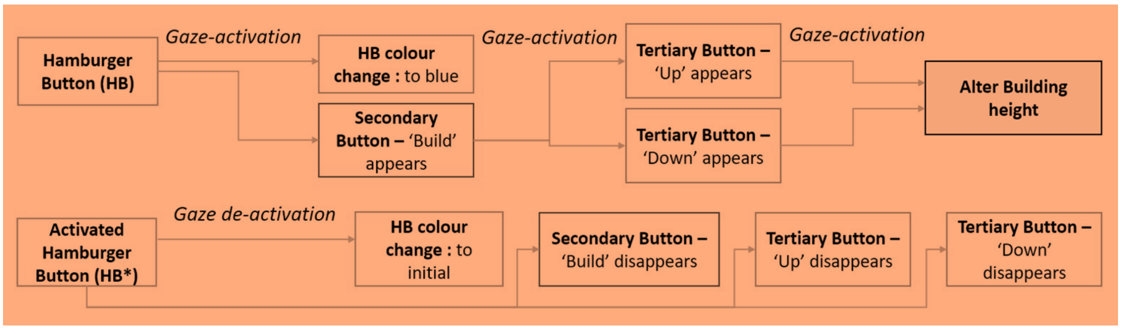
Hamburger Button (HB)
Gaze-activation
HB colour change : to blue
Secondary Button – 'Build' appears
Gaze-activation
Tertiary Button – 'Up' appears
Tertiary Button – 'Down' appears
Gaze-activation
Alter Building height
Activated Hamburger Button (HB*)
Gaze de-activation
HB colour change : to initial
Secondary Button – 'Build' disappears
Tertiary Button – 'Up' disappears
Tertiary Button – 'Down' disappears

Figure 4: Diagram showing how to activate and deactivate the hamburger button in VR model variation 2.

2.2. VR models' gaze detection

To render a scene in real-time within a virtual reality setup, a sequence of matrix transformations (T) is applied to establish the final coordinates of any point P (x, y, z) on the virtual screen. The transformation formulas are outlined below.

$$T = T_{vp}\ T_{can}\ T_{left}\ T_{eye}\ T_{rb} \text{ (for the left eye)} \quad [29]$$

and,

$$T = T_{vp}\ T_{can}\ T_{right}\ T_{eye}\ T_{rb} \text{ (for the right eye)} \ [29]$$

In the virtual reality (VR) environment, the transformation of a point for optimal viewing is represented by T. This is achieved through a series of transformation matrices, each serving distinct roles: T_{vp} handles viewport transformation, T_{can} facilitates canonical view transformation via perspective projection, T_{eye} adjusts the virtual world according to the eye's coordinate system, and T_{left} and T_{right} shift the scene left and right for stereoscopic effects. T_{rb} pertains to rigid body transformations. It's essential to recognize that these matrix operations are non-commutative and performed in a right-to-left order.

A reticle pointer, which aligns with the user's gaze, is positioned at the center of the viewing plane of a virtual camera anchored between the user's eyes. This reticle pointer, designated as *P,* along with other points in the virtual space, undergoes the aforementioned transformations to be rendered accurately on the virtual screens for both the left and right eyes of the VR head-mounted display.

To implement gaze detection within the VR model, the **GvrReticlePointer prefab** from the Google VR SDK for Unity is utilized. In Unity, a prefab acts as a template that contains a GameObject along with all its properties and components, enabling the creation of new instances in a scene. The **GvrReticlePointer**, which creates the reticle pointer based on the user's gaze, is attached as a child of the 'Main Camera' in the Unity VR model, ensuring that the pointer continuously tracks the user's gaze.

Alternatively, the gaze pointer can be constructed using a vector or ray that originates from the 'Main Camera' and points forward. An opaque shader is applied to a sphere object positioned along this vector, improving the visibility and interaction of the gaze pointer. Figure 6 illustrates the schematic for the gaze pointer and its interactions with target buttons, showcasing how users can effectively engage with the virtual environment.

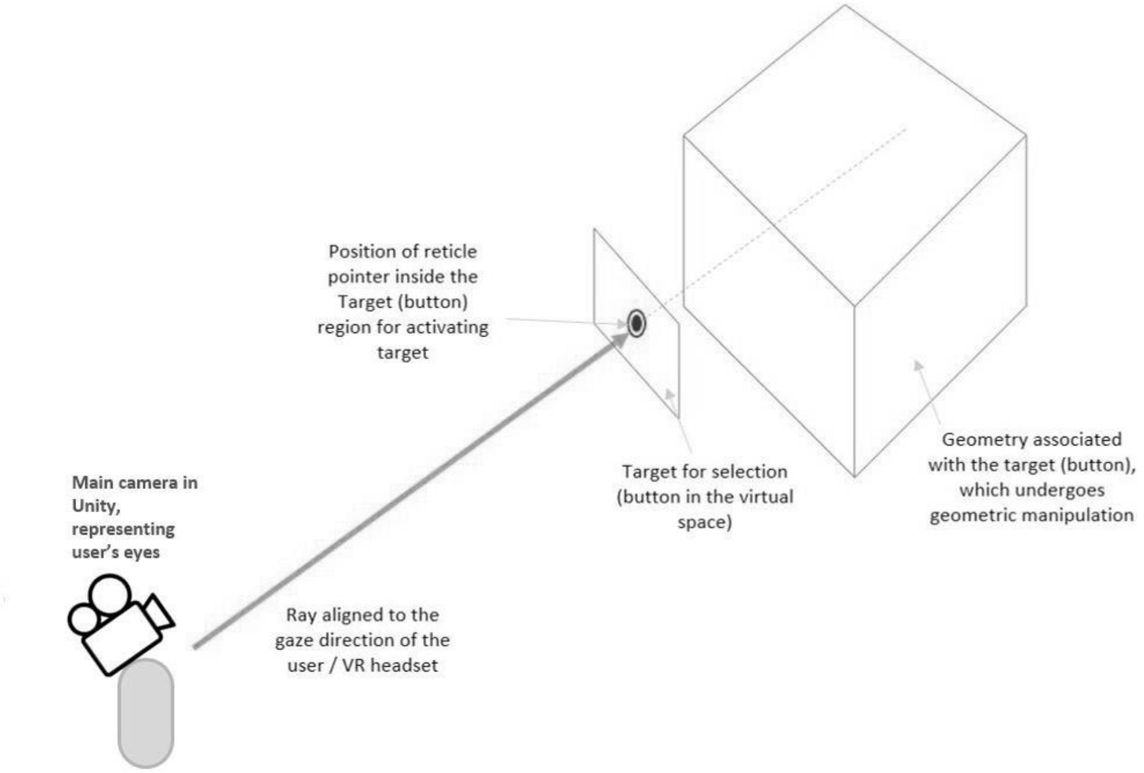
Position of reticle pointer inside the Target (button) region for activating target
Main camera in Unity, representing user's eyes
Target for selection (button in the virtual space)
Geometry associated with the target (button), which undergoes geometric manipulation
Ray aligned to the gaze direction of the user / VR headset

Fig. 2 The gaze/reticle marker points forward along the vector to the "Main Camera."

2.3. The suggested VR model's object handling and button activation

In Unity's Events library, the **Event Trigger** class plays a crucial role in managing user interactions within the virtual reality (VR) environment, particularly for scaling tagged GameObjects that represent buildings. This class enables the execution of functions based on specific user actions, utilizing event types like **'Pointer Enter'** and **'Pointer Exit'** to facilitate these interactions.

In **model variant 1**, the functionality is straightforward. When the **'Pointer Enter'** event for the **'Build Up'** button is triggered—indicating that the gaze pointer has entered the button's area—the associated GameObject (or building) continuously scales up vertically by a designated vector. This visual effect allows the building to appear to grow taller in real-time. Conversely, the **'Pointer Exit'** event signifies that the gaze pointer has moved away from the button, causing the upward scaling of the building to stop. A similar approach is applied to the **'Build Down'** button, where the **'Pointer Enter'** event initiates continuous scaling down of the building, and the **'Pointer Exit'** event halts this process.

In **model variant 2**, the interaction becomes more intricate due to the inclusion of hamburger buttons along with their associated secondary and tertiary buttons. Here, the **'Pointer Enter'** and **'Pointer Exit'** events for each button produce a sequence of interactions that enhance usability. For example, when the gaze pointer hovers over a non-activated hamburger button, it triggers the appearance of a secondary button labeled **'Build'**. The **'Pointer Exit'** event ensures that both the hamburger and secondary buttons remain active, allowing the user to interact with these options without losing their selection.

Subsequently, when the gaze pointer activates the **'Build'** button, it brings forth the tertiary buttons labeled **'Up'** and **'Down'**. Interacting with these tertiary buttons causes the corresponding GameObject to scale up or down vertically, depending on which button is engaged. When the gaze pointer exits these buttons, the scaling operations cease, providing precise control over the building's dimensions.

Moreover, if the gaze pointer enters an activated hamburger button, it returns to its normal state (de-highlighted) and retracts any visible secondary or tertiary buttons from the VR scene. This system of events fosters a responsive and intuitive interface, enhancing user experience and interaction fluidity within the virtual environment, allowing users to manipulate building models effectively and efficiently.

2.4. Using the suggested VR model for navigation

In the VR environment, users navigate forward by tilting their heads downward while wearing the head-mounted device (HMD), exceeding a predetermined threshold angle (see Fig. 7). To implement this feature within the Unity-based VR application, a **Character Controller** component is added to an empty **Game Object**, with the **main camera** positioned as its child. This setup effectively tracks the user's head movements.

A script has been created and attached to the Game Object to facilitate forward movement, allowing smooth translation within the virtual space. The movement is activated when the camera's horizontal rotation falls between **15 degrees** and **90 degrees**, signifying that the user is looking down at an appropriate angle. This intuitive navigation method aligns the user's physical actions with their virtual experience, thereby enhancing immersion in the VR environment.

```
SET walkAngle to 15.0
SET walkSpeed to 3.0
SET cc of type CharacterController FUNCTION Start()
        cc = GetComponentInParent<CharacterController>()
END FUNCTION
FUNCTION Update()
IF rotation of camera angle about x direction IS GREATER THAN walkAngle AND rotation of camera angle about x direction IS LESSER THAN 90
                SET walkForward to true
ELSE
                SET walkForward to false
END IF
IF walkForward IS true
                CREATE a Vector forward in the forward direction of the camera
        MOVE cc in the direction of forward times walkSpeed END IF
END FUNCTION
```

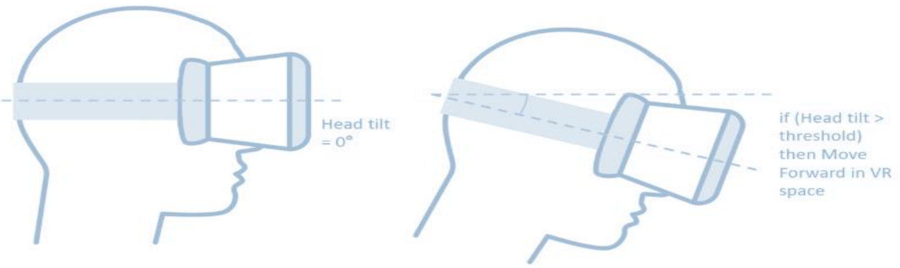

Fig. 7. Head-tilt results for forward movement in the virtual world.

2.5. Experimental setup

A total of 32 participants evaluated each variant of the VR model, ensuring an equal gender ratio and ages between 18 and 46 years. Their experience with computer-aided design (CAD) or 3D modeling software varied from none to 22 years, with an average experience of 5.4 years.

The **Acer Windows Mixed Reality Headset (model AH101-D8EY)** was utilized for the experiments, boasting a **Field of View** of **100°**, a maximum resolution of **2880 x 1440**, and **6 degrees of freedom (DoF)**. This headset was connected to a desktop computer equipped with an **i7-8700K processor**, **16 GB of RAM**, and an **NVIDIA GeForce GTX 1080** GPU with **8 GB** of memory, ensuring high performance during the trials.

Initially, participants were given the opportunity to explore the VR environment and its features within both variants of the model. After becoming comfortable with the VR setup, they were tasked with extruding buildings in a staggered pattern for both application variants. Participants were shown a reference image depicting buildings symmetrically extruded in a staggered arrangement on either side of a virtual road to guide their modeling efforts.

Seated in revolving chairs, participants could rotate and navigate the VR environment while instructed to limit the movement of their upper arms, forearms, and hands during the task. This approach aimed to replicate a scenario where individuals with physical disabilities—particularly those with limited upper limb mobility—would use the proposed VR application for 3D modeling. Participants were allowed to rotate their heads to navigate the VR environment and activate virtual buttons through gaze direction, emphasizing the accessibility features of the VR model and its potential benefits for users with reduced motor abilities.

2.6. Testing the two VR model variations with users

The start and end times for each participant's activities in both VR model variants were carefully recorded and analyzed to assess performance. After completing the modeling task, participants were asked to fill out a brief questionnaire specific to the variant they used. The questionnaire aimed to evaluate user perceptions regarding two main factors: the level of visual clutter in the VR environment and the effort required to perform tasks using the gaze-based VR system.

The perception of visual clutter captures participants' opinions on the design of the user interface, indicating whether they found the environment to be overwhelming or well-organized. The perception of effort reflects how difficult or easy they felt it was to complete tasks within the VR environment. Participants rated each factor on a five-point scale, where '1' indicated a poor perception and '5' indicated an excellent one.

After the survey, participants were invited to share their overall experiences with both model variants, offering feedback, concerns, or suggestions for improvement. This qualitative input is crucial for refining the gaze-based VR application, ensuring that future versions better meet user expectations, improve usability, and create a more satisfying experience.

3. RESULTS AND DISCUSSION

3.1 Findings from user testing two VR model iterations

The study evaluated task completion times and user experiences across two VR model variants, focusing on the time required to complete tasks, perceived visual clutter, and the effort involved in interacting with the virtual environment. For VR model variant 1 (Task 1 or T1), the average task completion time was 78.84 seconds, with a median of 64 seconds. In contrast, variant 2 (Task 2 or T2) showed an average completion time of 93.87 seconds and a median of 78.5 seconds. The differences between the two variants were notable, with a 15.03-second gap in average task completion time and a 14.5-second difference in median time. These results indicate that participants generally took longer to complete the same task in variant 2 compared to variant 1, as depicted in Figure 8a. In addition to measuring task completion times, the study also assessed user perceptions of visual clutter and effort. Participants rated these factors on a five-point scale, where 1 indicated the highest visual clutter or most effort, and 5 represented the least clutter or minimal effort. For visual clutter, Task 1 received an average score of 4, while Task 2 scored 3.78, suggesting that participants perceived more visual clutter in variant 2 than in variant 1 (Figure 8b). Similarly, for effort, Task 1 received an average score of 4.46, while Task 2 scored 3.65, indicating that participants felt they had to exert more effort in variant 2 (Figure 8c). Overall, the results demonstrate that variant 1 provided a better user experience in terms of both task completion time and user-reported metrics. The additional time required in variant 2 can be attributed to the process of triggering tertiary buttons, which only appeared after the activation of hamburger and secondary buttons. In variant 1, participants only needed to activate one set of buttons, which were always visible in the VR scene, simplifying the process and enabling quicker task completion. Although variant 2 was designed with hamburger buttons and nested button structures to reduce visual clutter, the average score for visual clutter was still lower than in variant 1. This outcome could be due to the sequential activation of hamburger buttons and their associated secondary and tertiary buttons, which led to a growing number of buttons on the screen. Since the buttons did not automatically collapse after being used, visual clutter accumulated as the task progressed, especially towards the end.

The increased effort perceived in variant 2 was likely due to the additional head movements required to aim at and select the hamburger, secondary, and tertiary buttons needed to manipulate the building geometries. In variant 1, the fewer buttons and simpler interactions resulted in less physical effort. However, the effort required for navigation via head tilting was similar across both variants, as the scale and terrain of the virtual environments were identical. The study also gathered qualitative feedback from participants after they completed both tasks. Approximately 34% of participants (11 individuals) reported discomfort when selecting buttons located at the extreme edges of their position in the virtual environment. This issue was present in both variants and likely arose because the buttons and their labels were kept the same size regardless of their location in the scene, making distant buttons harder to select. Five participants (16%) found it challenging to interact with the expanding and retracting buttons in variant 2. The dynamic nature of the buttons added complexity, making it harder for some users to complete their tasks. Additionally, four participants (13%) reported difficulty shifting their gaze between the buttons and the tops of the buildings, as the increasing distance between these elements made it harder to maintain focus. As users manipulated the building heights, the distance their gaze had to travel increased, slowing task completion. Three participants (9%) noted accidentally triggering buttons while navigating the environment, which may have contributed to increased effort and task delays. Two participants suggested implementing an auto-retraction feature for variant 2’s buttons, which would automatically collapse the buttons after they were used, thereby minimizing the effort required to manually retract them. In summary, the study revealed that variant 1 offered a more efficient and less cluttered experience than variant 2. The nested buttons in variant 2 added complexity, leading to longer task completion times, more visual clutter towards the end of tasks, and greater perceived effort. User feedback also highlighted key areas for improvement, including the need for larger buttons in peripheral areas and an auto-retraction feature for buttons. These insights offer valuable guidance for improving the VR model, with the goal of enhancing usability, reducing visual clutter, and minimizing the effort required to interact with the virtual environment. Radviz visualization created using Orange data mining software, showcasing user data for tasks T1 and T2 across three key metrics: task completion time, effort applied, and perceived visual clutter. The visualization projects these variables onto a 2-dimensional unit circle, with each variable's values scaled between 0 and 1. The colored regions in the diagram represent the likelihood of data points corresponding to either T1 or T2. Task T1 data points are primarily linked with lower perceived effort, while task T2 data points are more often associated with longer task completion times or reduced visual clutter. This visualization highlights the differences in user experiences between the two tasks based on these performance indicators.

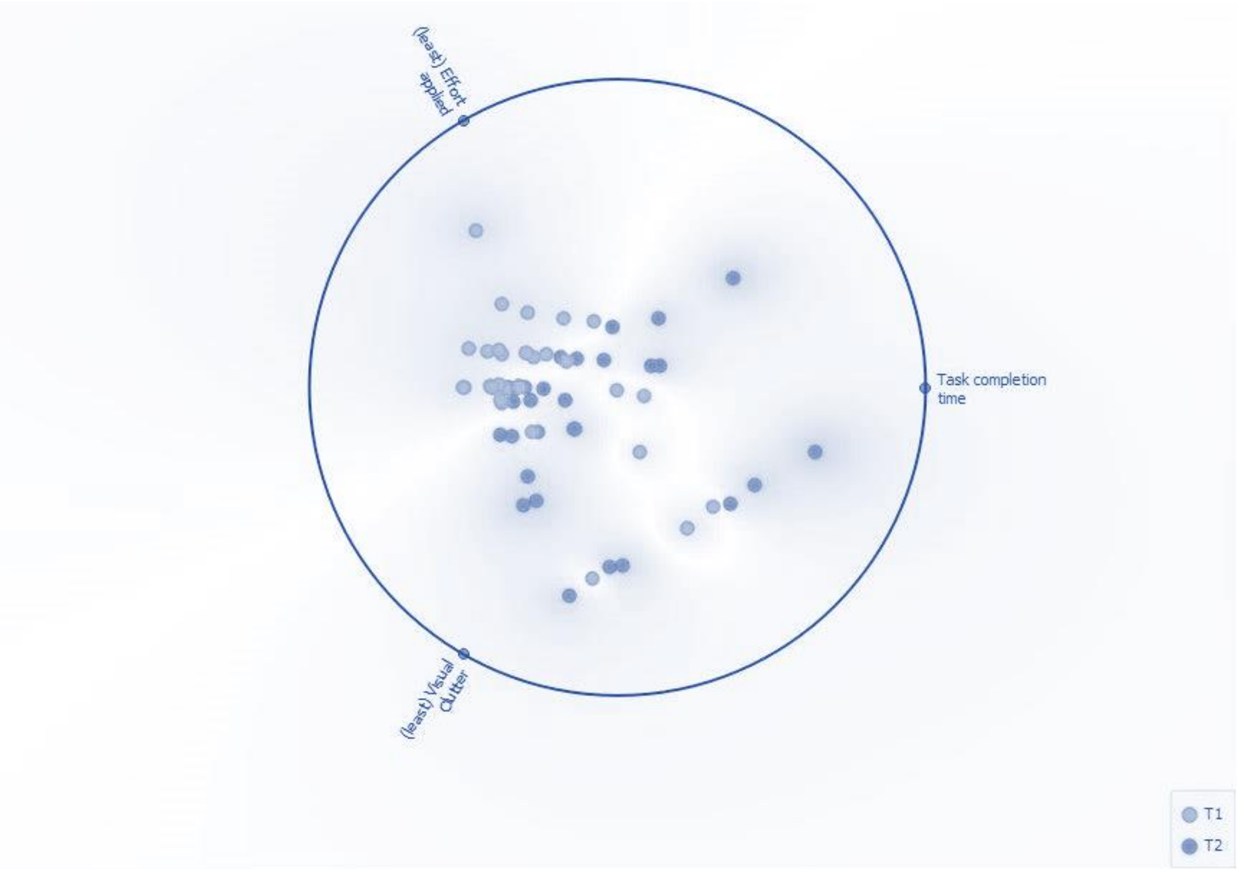

Fig. 9. A Radviz visualisation of the time taken to complete tasks T1 and T2, the amount of effort put in, and the visual clutter.

3.2 A machine learning algorithm to forecast user effort

The perceived effort exerted by users to complete tasks in the VR interface reflects both their physical exertion and any resistance encountered due to the interface's design. A key factor influencing this perception is the allowable rotational movement of the head (about the spinal axis), which should be minimized, particularly for applications intended for motor-impaired users. In this context, the spatial arrangement and location of objects and interactive entities, such as buttons—which partially contribute to the perception of visual clutter—affect how users move their heads to navigate and select within the VR environment. Additionally, longer task completion times suggest that users took more time aiming, selecting, and navigating due to the virtual environment’s design, which could influence their overall perception of effort.

To explore the relationship between task completion times, perceived visual clutter, and user-reported effort, a Neural Network (NN) model was used to predict the level of effort exerted by the participants. Data from 32 participants who performed tasks T1 and T2 were used for this prediction. In the model, the ‘effort applied’ parameter served as the target variable, and it was categorized into two groups: (a) ‘High effort’ for participant scores below 4, and (b) ‘Low effort’ for scores of 4 and 5. The input layer of the NN model was based on participants' scores for visual clutter and task completion times, while the output layer indicated the predicted category for the effort applied (Figure 10a).

The architecture of the NN model included hidden layers in the configuration {30, 20, 10, 3}, with the ReLU (Rectified Linear Unit) activation function, Adam optimizer as the solver, and a regularization parameter (α) set to 0.0001. The model was evaluated using stratified 5-fold cross-validation on the dataset and produced the following performance metrics: a classification accuracy of 0.78, an F1 score of 0.771, a precision of 0.768, and a recall score of 0.781.

Figure 10b presents the confusion matrix of the NN model, which displays the predicted classification results as a percentage of the actual data, demonstrating how well the model could predict the degree of effort applied based on task completion time and perceived visual clutter.

3.3 The VR application's implementation plan

An AI model capable of predicting user effort based on performance metrics and real-time feedback can be highly valuable in determining whether a specific user interface (UI) configuration—such as the arrangement of buttons, menus, and displayed information—should be retained or adjusted. This adaptive system allows for real-time modifications based on the user's needs. Figure 11 illustrates a strategy for implementing this approach in the VR application. If the AI model identifies a 'high' level of effort being exerted by the user, the interface can be updated to reduce strain by altering the layout or functionality. The strategy also envisions dividing complex tasks into sequential sub-tasks, where each sub-task is completed, providing rapid feedback and real-time performance data for the AI model to assess. Alternatively, the model can collect user data during an ongoing task, predicting effort in real-time and enabling interface adjustments while the user is still engaged, thus creating a more responsive and user-centric experience.

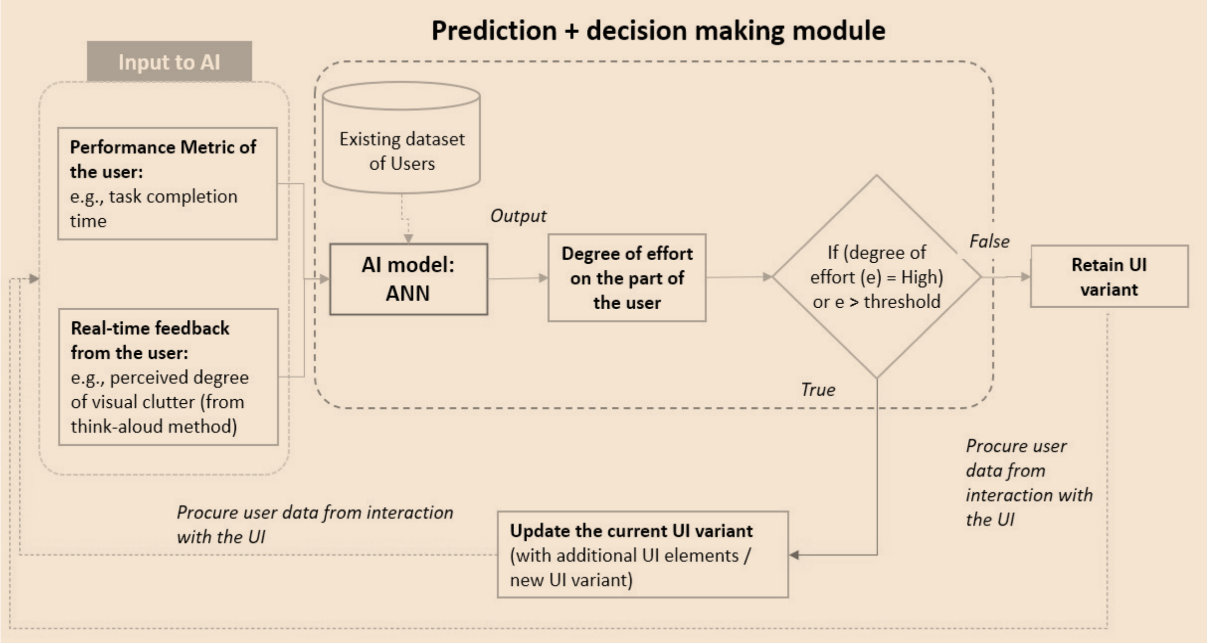
Prediction + decision making module
Input to AI
Performance Metric of the user: e.g., task completion time
Real-time feedback from the user: e.g., perceived degree of visual clutter (from think-aloud method)
Existing dataset of Users
AI model: ANN
Output
Degree of effort on the part of the user
If (degree of effort (e) = High) or e > threshold
False
True
Retain UI variant
Procure user data from interaction with the UI
Update the current UI variant (with additional UI elements / new UI variant)
Procure user data from interaction with the UI

Fig. 3. Neural network (NN)-based implementation technique for effort prediction in the suggested VR application.

In a gaze-based VR system designed for basic 3D modeling, if users encounter longer task completion times or increased visual clutter with a particular interface configuration (such as variant 2), the interface can be switched to a more user-friendly version (like variant 1) that requires less effort across a broader range of users. These variations in the user interface typically involve adjustments to button layouts to improve the gaze input experience. However, gathering real-time feedback from users may disrupt their natural task flow and cause discomfort. To reduce this, feedback collection should be kept to a minimum, alternative methods such as voice input through a think-aloud technique can be used, or user behavior can be observed to avoid the need for direct feedback.

In a supervised learning context, collecting user data from both single and multiple UI configurations is essential to build a comprehensive dataset. Data collection should occur at three critical stages: the task initiation stage, task execution stage, and post-task completion stage. This allows for a complete evaluation of the interface's effectiveness and the overall user experience. To create more user-centric VR applications, including an intelligent future version of the current system, a robust methodological framework is required. This framework should incorporate data from all stages of the user journey, enabling UI adjustments that enhance usability and overall satisfaction.

3.4 An approachable framework for creating user interfaces

A key objective of an intelligent and adaptive user interface (UI) is to be responsive to user needs, allowing it to adjust itself accordingly. Figure 12 illustrates how user data is collected at various stages of interaction—task initiation, task execution, and post-task completion—and integrated into an AI model. This model is essential for generating recommendations to modify the UI, such as defining its initial state, dynamically changing UI elements during tasks, and providing contextual cues to users throughout their tasks. During the task initiation phase, gathering user attributes—such as age, gender, education level, and disability status—is crucial. This information helps in selecting the initial UI configuration tailored to each user. In the task execution phase, two types of data can be collected. The first involves real-time user actions and performance metrics, including eye gaze movements, fixation durations, task completion times, selections, and clicks. The second type is user feedback obtained through think-aloud protocols or brief surveys, allowing users to assess their experience as they engage with the interface. After task completion, the post-task phase focuses on gathering user feedback through comprehensive surveys, retrospective think-aloud sessions, or interviews. This feedback is vital for understanding user satisfaction and overall experience with the UI. By utilizing the extensive data collected across these phases, the AI model can continually adapt the interface to meet users' current needs, from the initial setup to task completion, while ensuring appropriate user permissions are respected. Additionally, this data-driven AI framework can analyze insights from past interactions—whether from a specific user or multiple users engaging with the same interface—to identify patterns and preferences. By leveraging this rich dataset, the AI can recommend the most suitable UI configuration for each user context, thereby enhancing the overall user experience and fostering a more intuitive interaction with the VR system. This ongoing adaptability positions the AI-driven UI as a responsive entity that evolves to meet users' needs and preferences.

4. CONCLUSION

This paper introduces an innovative head gaze-based virtual reality (VR) application designed for basic 3D modeling, featuring a one-dimensional scaling approach for 3D geometries. The application consists of two variations of user interfaces (UIs) that primarily differ in their button layouts. These UI variants enable users to navigate the virtual environment using head rotation and activate both visible buttons and those hidden within a hamburger menu through head-directed actions. To evaluate the effectiveness of these VR variants, a study involving 32 participants was conducted. The analysis focused on user performance and self-reported metrics, providing insights into how different UI configurations influence usability and the overall user experience. Feedback indicated that variant 1 outperformed variant 2 regarding usability, demonstrating its effectiveness in facilitating user interaction within the VR environment. This valuable user feedback laid the groundwork for further development, leading to the creation of an AI model based on neural networks. This AI model was specifically designed to predict the 'effort' exerted by users while interacting with the VR interface. The predictions rely on two key input metrics: task completion time, which serves as a performance metric, and the degree of visual clutter within the UI, assessed through user reports.

REFERENCES

[1] Maskey M., Lowry J., Rodgers J., McConachie H., Parr J.R. Reducing Specific Phobia/Fear in Young People with Autism Spectrum Disorders (ASDs) through a Virtual Reality Environment Intervention. PLoS ONE. 2014;9:e100374. doi: 10.1371/journal.pone.0100374.

[2] Choi J.Y., Yi S., Ao L., Tang X., Xu X., Shim D., Yoo B., Park E.S., Rha D. Virtual reality rehabilitation in children with brain injury: A randomized controlled trial. Dev. Med. Child Neurol. 2020;63:480–487. doi: 10.1111/dmcn.14762.

[3] Qian J., McDonough D.J., Gao Z. The Effectiveness of Virtual Reality Exercise on Individual's Physiological, Psychological and Rehabilitative Outcomes: A Systematic Review. Int. J. Environ. Res. Public Health. 2020;17:4133. doi: 10.3390/ijerph17114133.

[4] Leon, M.; Laing, R.; Malins, J.; Salman, H. Making collaboration work: Application of A conceptual design stages protocol for pre-BIM stages. WIT Trans. Built Environ. 2015, 149, 205–216.

[5] Succar, B. Building information modelling framework: A research and delivery foundation for industry stakeholders. Autom. Constr. 2009, 18, 357–375.

[6] Gross, M.D.; Yi-Luen Do, E.; McCall, R.J.; Citrin, W.V.; Hamill, P.; Warmack, A.; Kuczun, K.S. Collaboration and coordination in architectural design: Approaches to computer mediated team work. Autom. Constr. 1998, 7, 465–473.

[7] Izard S.G., Juanes J.A., García-Peñalvo F.J., Gonçalvez Estella J.M., Ledesma M.J.S., Ruisoto P. Virtual Reality as an Educational and Training Tool for Medicine. J. Med Syst. 2018;42:50. doi: 10.1007/s10916-018-0900-2.

[8] Javaid M., Haleem A. Virtual reality applications toward medical field. Clin. Epidemiol. Glob. Health. 2020;8:600–605. doi: 10.1016/j.cegh.2019.12.010.

[9] Gert Kwakkel, Boudewijn J Kollen, and Hermano I Krebs. 2008. Effects of robotassisted therapy on upper limb recovery after stroke: a systematic review. Neurorehabilitation and neural repair 22, 2 (2008), 111–121.

[10] Philip Lindner. 2021. Better, virtually: the past, present, and future of virtual reality cognitive behavior therapy. International Journal of Cognitive Therapy 14, 1 (2021), 23–46.

[11] Arend, B.: Hey Siri, what can I tell about Sancho Panza in my presentation? Investigating Siri as a virtual assistant in a learning context? pp. 7854–7863 (2018). https://doi.org/10.21125/inted.2018.1874

[12] Baber C.: Developing interactive speech technology. In: Interactive Speech Technology: Human Factors Issues in the Application of Speech Input/Output to Computers. Taylor & Francis, Inc., Bristol (1993)

[13] Buxton B. *Sketching User Experiences: Getting the Design Right and the Right Design.* Morgan Kaufmann; San Francisco, CA, USA: 2007.

[14] Anand S., Sen A.K. *Sustainable Human Development: Concepts and Priorities.* UNDP; New York, NY, USA: 1996. Office of Development Studies Discussion Paper, No. 1.

[15] United Nations Development Programme (Various Years) *Human Development Report.* Oxford University Press; Oxford, UK: 2019.