ViRgilites: Multilevel Feedforward for Multimodal Interaction in VR

VALENTINO ARTIZZU, Dept. of Mathematics and Computer Science, University of Cagliari, Italy
KRIS LUYTEN, UHasselt - Flanders Make, Belgium
GUSTAVO ROVELO RUIZ, UHasselt - Flanders Make, Belgium
LUCIO DAVIDE SPANO, Dept. of Mathematics and Computer Science, University of Cagliari, Italy

Navigating the interaction landscape of Virtual Reality (VR) and Augmented Reality (AR) presents significant complexities due to the plethora of available input hardware and interaction modalities, compounded by spatially diverse visual interfaces. Such complexities elevate the likelihood of user errors, necessitating frequent backtracking. To address this, we introduce ViRgilites, a virtual guidance framework that delivers multi-level feedforward information covering the available interaction techniques as well as the future possibilities to interact with virtual objects, anticipating the interaction effects and how they fit with the overall user's goal. ViRgilites is engineered to facilitate task execution, empowering users to make informed decisions about action methodologies and alternative courses of action. This paper presents the architecture and functionality of ViRgilites and demonstrates its efficacy through evaluation with a formative user study.

CCS Concepts: • **Human-centered computing** → **Virtual reality**; **Graphical user interfaces**; **User interface programming**; **User interface toolkits**; • **Applied computing** → **Education**.

Additional Key Words and Phrases: virtual reality, toolkit, meta-design, vocational education, feedforward, task

ACM Reference Format:
Valentino Artizzu, Kris Luyten, Gustavo Rovelo Ruiz, and Lucio Davide Spano. 2024. ViRgilites: Multilevel Feedforward for Multimodal Interaction in VR. *Proc. ACM Hum.-Comput. Interact.* 8, EICS, Article 247 (June 2024), 24 pages. https://doi.org/10.1145/3658645

1 INTRODUCTION

In recent years, there has been a raising interest in the adoption of Virtual Reality (VR) headsets. These types of devices are more accessible and increasingly easier to use. This trend, also supported by the idea of the Metaverse (pushed forward from leading industries like Meta) may also lead to a renewed interest from old and new developers in using this technology in a wide variety of application domains. VR can be used for, amongst other, entertainment, training and education, and interaction with digital twins. However, interaction in VR remains challenging, especially for occasional users because of the different modalities that can be used and combination of direct and indirect interaction techniques, and the lack of visibility and discoverability of supported interactions compared to a GUI that has standardised widgets and common interaction devices.

Authors' Contact Information: Valentino Artizzu, Dept. of Mathematics and Computer Science, University of Cagliari, Cagliari, Italy, valentino.artizzu@unica.it; Kris Luyten, UHasselt - Flanders Make, Expertise Center for Digital Media, Diepenbeek, Belgium, kris.luyten@uhasselt.be; Gustavo Rovelo Ruiz, UHasselt - Flanders Make, Expertise Center for Digital Media, Diepenbeek, Belgium, gustavo.roveloruiz@uhasselt.be; Lucio Davide Spano, davide.spano@unica.it, Dept. of Mathematics and Computer Science, University of Cagliari, Cagliari, Italy.

ACM 2573-0142/2024/6-ART247
https://doi.org/10.1145/3658645

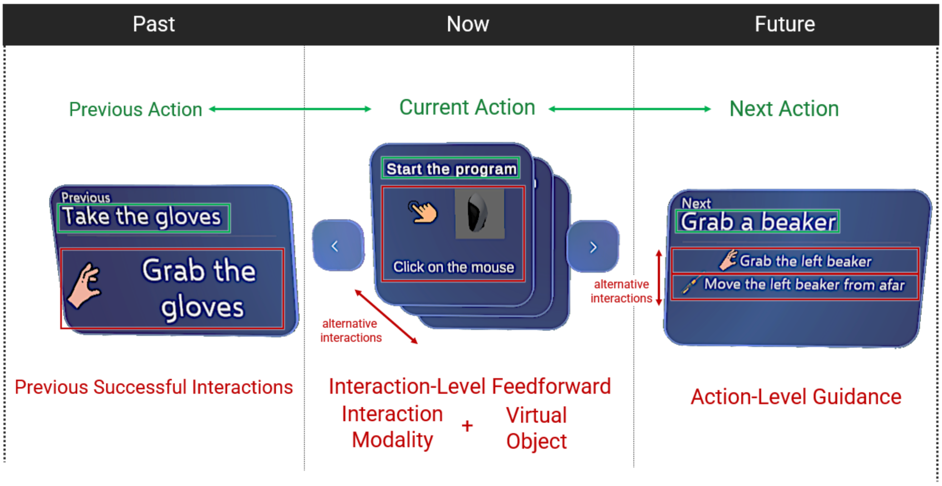

Fig. 1. The anatomy of ViRgilites: from feedback of past actions and interactions, over the current action and interactions at hand, to guidance toward the next action and interactions. In ViRgilites, the virtual object depicted alongside the interaction, provides a shortcut to navigate to the object in the virtual environment.

One solution can be the use of feedforward for showing the modalities to use, in order to achieve completion of an action. Indeed, feedforward "informs the user about what the result of his action will be" [15]. In this case the action would be described and, as a result, the interaction would be complete. Muresan et al. [34] propose using feedforward to clearly communicate the action possibilities in VR, presenting a design space and a set of guidelines to implement efficient feedforward in VR user interfaces. This finding is consistent with the reference framework that was introduced by Vermeulen et al. [50], who defined feedforward as a cognitive affordance that is understandable through a well-defined sensory affordance (such as a readable, descriptive label or the object's physical shape) and reveals the functional affordance (the system function) coupled to a physical affordance (the action possibility). Our approach, ViRgilites, introduces multiple levels of feedforward, is applicable for goal-driven environments and supports multiple user tasks. With ViRgilites, interactions are not isolated actions on virtual objects, interactions –either single or more complex sequences– are essential parts that contribute toward completing a task [54].

The two levels of feedforward integrated into ViRgilites are *interaction level feedforward* and *action level guidance*. While interaction level feedforward helps users select and perform interactions with virtual objects, the action level guidance helps users recognise and be aware of the context in which the interactions are performed. To show the relevance of these two levels, we consider use cases in the off-site vocational training domain. The application of virtual reality technologies in this setting gives the opportunity to safely develop the necessary skills for performing the required tasks while also letting the trainees learn from their errors that, especially in dangerous jobs, can risk their lives.

There are many possible ways to interact with a virtual object in a virtual environment;

- Multiple modalities can be used, such as gestures, speech command, touch, gaze, and many more. Most if not all headsets support various modalities for interaction.
- The interaction devices that are available across VR (and AR) setups and headsets differ. Even when offering the same modality, there can be subtle differences in, for example, latency, level of detail and precision of interaction detection.

- The interaction with an object in a 3D environment is affected by the physical distance between the user and the target object. Some modalities are more appropriate in a near interaction mode, while other offer the advantage of supporting the interaction without requesting the user to navigate the 3D scene. In addition, the effectiveness of the available interaction modalities are influenced by the interaction context (e.g., voice commands are affected by the environment noise). Finally, switching from a modality to another may require effort for the user.

In this paper, we present ViRgilites, a system aiming to guide users in performing multimodal interactions in a VR environment. ViRgilites is a meta-level user interface [13], meant to support the user to explore, discover and perform interactions with virtual objects. Missing visibility of possible interactions, the lack of standardised, widely accepted interactions with 3D objects and the uncertainty of the outcome of executed interactions in virtual, mixed and augmented reality environments make these very challenging environments to support efficient interactions and avoid erroneous user actions. Our approach informs users about the possibilities to execute tasks in goal-driven virtual environments, improves the discoverability of tasks supported by the virtual environment and provides feedforward on what follows when a selected interaction has been performed. We included an approach to automatically recommend and prioritize interaction modalities depending on the application domain, the context of use and the predefined preferences of the designer. We validated ViRgilites in a formative study using two real-world use cases, focusing on gathering user feedback on the design and anatomy of the meta-user interface.

2 MULTI-LEVEL INTERACTION GUIDANCE FOR VIRTUAL REALITY

We distinguish between tasks, actions and interactions. *Tasks* are sequences of actions that a user has to complete for reaching a desired goal[1] An *action* represents a step towards the completion of a task. Considering the different modalities usually available in XR environments, the user can select among different *interactions* to perform the same action. Therefore, for each action, we have a set of interactions that are equivalent in terms of effects.

The interaction level feedforward, as also shown in Figure 1, presents the various options to perform an interaction to the users, ViRgilites provides designers and developers with the freedom to provide one or more interaction techniques within a virtual environment, while minimising the cognitive effort for users to explore, discover and use possible interactions.

Multiple sensory affordances convey information about the available interactions on a given object to trigger the same system function (i.e., complete the same action). A simple yet informative example is designing an interface suggesting how to grab an object to move it elsewhere. If we are designing for VR devices supporting hand tracking, we may want to support such operation at design time through a grabbing action. In addition, we also want to support the selection from a distance, using the laser pointing technique for reducing the environment navigation. In such a setting, we need to reveal the same functional affordance (the position change) using two sensory affordances, one for communicating the grab action and one for the laser-pointing. The physical affordance (i.e., the object shape) should communicate that we may grab the object, but it does not communicate how to laser-point it, since usually the shape does not change according to the interaction modality.

The action level guidance is included in ViRgilites to ensure users are aware how their actions contribute to completing a task and reaching a goal. In this sense, a series of actions is like an operation procedure that users have to perform. For ViRgilites to be useful, such operation

---

[1] +In our examples, we consider simplified scenarios where a training procedure requires completing a single case. In the general case, a procedure may require the completion of more than one task.

procedures tasks that consists of multiple actions need to be defined for a virtual environment, as happens in [27, 54]. This ensures actions have a context and purpose, and are not isolated with respect to the other actions performed with the virtual objects. It also helps the user to become aware and knowledgeable on the various tasks that are supported in the virtual environment.

Such multi-level interaction support allows users to understand which interactions are available in the simulation and to anticipate their effects. Suppose we model a VR task's solution through the classic human problem-solving theory by Newell and Simon [35]. In that case, ViRgilites allows users to understand 1) which operators are available and 2) which is the next state in the problem space reachable by applying each operator. This provides information on the possible changes at an *action* level. However, users also require information on the outcome at a *task* level, i.e., whether or not the new state gets closer to the desired world configuration or which one among the possible new states is the closest to the user's goal (or sub-goal). Considering a training scenario, the feedforward support should communicate how a given interaction contributes to the task's progress, helping the trainee towards the goal of the exercise.

Another relevant aspect of our work is the supporting a mixed initiative process for selecting the interaction modality. While there are different options for triggering a function, some are more effective than others in different contexts. For instance, a vocal modality should be avoided in a noisy environment, while hand-based interactions should not be available if users have their hands full. On the one hand, XR applications have different means for sensing the context, thus the UI may react to changes and adapt the interaction by proposing the most suitable modality given the sensed data. We discuss a generic prioritisation method for controlling this aspect during the UI development. On the other hand, users must have the final word on the selection, so the UI must contain means for overriding the system choice if needed.

3 RELATED WORK

3.1 Feedforward in Virtual Reality

Feedforward is a relatively unexplored concept that it goes on the other side of the temporal spectrum with the concept of feedback. Indeed, we can define feedforward as "what the result of their action will be" [50], while feedback refers to the ability to provide information about what an action did (or is doing). As extensively described by Vermeulen et al. [50], feedforward is grounded on the ability to couple available actions and the possible reactions in terms of time, location, direction, dynamics, modality and expression [51] and it provides information on what actions are available and how to perform them (*inherent*), the features they activate or their purpose (*functional*), or it may supplement the action possibilities through labels, icons and other UI elements (*augmented*).

In the literature, we can find different research work covering both theoretical aspects of feedforward [7, 36, 38, 50, 51], and practical applications of feedforward techniques for 2D interfaces [5, 12, 20, 48]. Considering 3D interaction, Muresan et al. [34] provide a design space on the feedforward design and modelling through a comprehensive analysis of the techniques available for VR, and they illustrate how to design feedforward for specific VR interactions. The design space covers three key stages of feedforward in VR (*triggering, previewing actions and outcomes* and *exiting the preview*) and, for each stage, it provides a list of parameters the designer can set for identifying the right technique to use in the considered application. Our work leverages the categorisation for identifying techniques to be applied in training procedures in XR and supporting procedural adaptation from a simple text-based procedure definition. Our goal is to provide automatic feedforward support rather than exploring the design space.

The application of feedforward techniques in Virtual Reality environments usually covers the guidance for motor tasks, i.e., showing how to perform movements for completing an interaction. For instance, Fennedy et al. [16] proposed to use a 3D version of the *Octopocus* technique [5] showing a dynamic trace of the available interactive gestures and a label providing information on the effect. *Just Follow Me* shows the motion to be completed to the user as a superimposed ghost that anticipates the movements. Similarly, Lilija et al. [29] limit the ghosting technique to the representation of the virtual hand. *LightGuide* [44] uses superimposed arrows for suggesting the movement direction. Besides the guidance for motor tasks, traces and arrows have also been used for suggesting the interaction targets, i.e., 3D objects the user can interact with [3]. ViRgilites uses arrows and traces to guide the user towards the interactive object in the scene required for completing the current action. It also allows an automatic teleportation towards the target when far away.

Other studies focused on particular aspects and desired effects requiring the design of a feedforward support. For instance, Chauvergne et al. [10] surveyed the techniques applied during the onboarding in VR applications (i.e., the phase where the application teaches the user how to interact with it). Barathi et al. [4] studied the effect of the interactive feedforward for enhancing the user's physical performance through a VR video game. We focus on a particular set of VR environments supporting goal-driven tasks, and we generate the feedforward to facilitate the interaction by exploiting the description of the procedure steps. Such environments are often dedicated to "Vocational training", a term referring to "education programmes that are designed for learners to acquire the knowledge, skills and competencies specific to a particular occupation, trade, or class of occupations or trades" [18]. Previous research has demonstrated the effectiveness of immersive learning environments for vocational training both using VR [2] and AR [11], in different domains such as welding [47] and construction management [42]. However, these existing systems do not provide learners with the means to discover and select the appropriate interactions and modalities for completing the tasks. ViRgilites eases the introduction of such guidance, considering the designer's preferences, the context of use and, above all, the learner's choice.

3.2 Multimodal Interaction in Virtual Reality

The most relevant trait of the interaction in Virtual Reality is the supported level of immersion in a synthetic environment. While other types of media target the dominant visual and auditory channel [8, 45], VR integrates visual, auditory, haptic and, in the most recent work, even olfactory and gustatory [33, 43] channels. We refer to Martin et al. [31] for an in-depth survey on the effects of multimodality in VR on perceived realism, user attention and performance.

This work focuses on the different modalities available for interacting with virtual objects in VR. Techniques relying on the visual channel are the most researched in the literature [28, 40, 41, 46], and they are quite established and standardised in professional toolkits for VR development. For instance, we adopted the Mixed Reality Toolkit (MRTK) by Microsoft [32], which offers support for effectively implementing far interactions (based on hand or remote-based raycasting), near interactions (based on poke interactions with nearby objects), object manipulation, locomotion and navigation techniques based on keyboard, controllers combined with the head-mounted display input. Even though these are standardised techniques, not all the objects in a VR environment respond to all the interactions. In addition, we cannot assume in VR the same level of familiarity as in mobile or desktop applications, and a learning phase is usually required [10]. In such a context, interfaces supporting feedforward techniques are needed to guide the user towards proper interactions.

Besides techniques relying on the visual channel, VR hardware supports gesture-based input through remotes or using additional devices such as the Leap Motion or the Microsoft Kinect [23, 53, 55]. Another source of input related to the movement of body parts is eye tracking, which is increasingly integrated into VR headsets and allows for pointing through the user's gaze [9, 22, 24, 56]. In different toolkits (e.g., MRTK [32]), gaze pointing is usually supported also by head orientation. Finally, voice input allows users to interact with VR environments through natural language sentences or keyword-based utterances [6, 17, 21].

Our work contributes to enhancing the usability of goal-based VR environments by generating feedforward that guides users in selecting the appropriate interaction considering all the available modalities.

3.3 Plastic User Interfaces

We found inspiration in the field of plastic user interfaces, "the capacity of a user interface to withstand variations of both the system's physical characteristics and the environment while preserving usability" [49], since we aim to provide a tailored guidance interface according to a 3D environment and the tasks that are supported in that environment. For example, 3DPlasticToolkit [26] is a toolkit that leverages plasticity to create adaptive 3D user interfaces. It utilises three models to represent the context of use, including hardware, task, and user configurations. The adaptation process is based on a scoring algorithm that considers the context of use to create the most appropriate 3D user interface. We share with this project the idea of using the user environment as a judgement point for deciding which element to present to the user, and the idea of using a scoring mechanism for evaluating the best outcome for a given scenario.

Lacoche et al. present a plastic interaction technique model to use and create interaction techniques that will fit to the needed tasks of a 3D application and to the input and output devices available [25]. In ViRgilites we aim to make this type of work more directly accessible by the end-users, and guide users in what interaction techniques are available and even allows users to control what interaction technique is used.

4 THE DESIGN OF VIRGILITES

The method we have followed for designing the interface of ViRgilites is the following: first, we reviewed the literature related to existing solutions for these problems [7, 34, 36, 38, 50, 51]; then we organized a design workshop with 7 experts where various design alternatives were proposed. These alternatives were discussed and resulted in an initial prototype design for ViRgilites. A functional implementation of the design was realised, and further iterations were made on the design to obtain a workable prototype. For these iterations, we always took into account the basic design principles: visibility, affordances, mappings, metaphors, consistency, feedback, and constraints [37]. In the following paragraphs, we will summarize how the design workshop results influenced our design choices.

Interaction Alternatives. During the design workshop, one of the aspects that challenged the participants was how to represent alternative ways of reaching the same goal. In addition, XR interaction environments support different modalities for performing the same functional action (e.g., grab and laser-pointing selection). All participants agreed that the guidance interface must represent such alternatives. Some distinguished between those that originated by the procedure and those generated by the interaction modality. We eventually used the same representation for both sources, but we made such a decision passing through different iterations that separated the modalities from the alternative steps in the procedure.

Alternative Selection. This design choice concerns who is responsible for selecting alternatives for completing the current task. Some participants would opt for a completely adaptive solution that assigns this concern to the system, others for an adaptable interface that gives complete control to the user. The discussion on this point led to a mixed solution: the system selects the alternative it considers the best, but the user can override the selection at any time. In ViRgilites, the system proposes an alternative according to a prioritisation function, and users can change this proposal whenever they like.

Feedforward Levels. When discussing which information to include in the interface, the proposals covered both feedforward information on the outcome of the current interaction (*interaction-level* feedforward) and how it contributes to reaching the current goal (*action guidance* level). The guidance is peculiar of goal-oriented interactions, where the system can establish in advance the user's goal (e.g., a correct final configuration in a training environment).

Direct vs Indirect affordances. There are many alternatives for designing both the functional (interaction outcome) and physical affordances (how to act) in XR environments, as summarised in [34]. The most relevant aspect of our work is distinguishing between a *direct* or *indirect* representation. The former relies on a preview simulating the targets, the actions and the outcomes in the XR environment. The latter relies on abstractions, such as explanatory text or icons. During the workshop, participants highlighted that it is possible to provide a generic direct representation of the action for performing an interaction (e.g., grabbing, pointing, looking at an object, etc.). However, the same does not apply to the action outcome, whose direct representation requires a customised design for each step. This limits the design of a generic guidance interface, which was the focus of the workshop. Therefore, they opt for indirect approaches which allow the generation of the affordances using structured information. We eventually selected a text-based format, but it is possible to select other media. In addition, we selected to use the same representation (indirect) for both affordances to concentrate the information on the guidance panel.

Feedback. Besides the notification of the correct execution of an interaction, participants agreed on including a feedback element in the guidance interface, to show the history of the previously executed actions in the procedure. An interesting discussion among the participants highlighted the trade-off between the number of previous steps maintained in the UI and the space required for displaying the information. On the one hand, increasing the number of previous steps helps users remember the previous actions leading to the current state. In a training context, this may affect the learning outcome. On the other hand, the visualisation of many steps may visually interfere with the surrounding environment, occluding the user's view. Participants concluded that keeping going beyond the previous action would cost too much UI space.

Future Steps. Including information about the next steps in the guidance UI poses a design problem similar to maintaining the information about the previous interactions. How many steps should be included in the visualisation? However, while past steps represent a linear path, the action in the future may include alternatives. In this case, the design choice concerns i) how many next steps should be included, ii) how to represent the alternatives and iii) how to select and/or filter them for the presentation. We tried different solutions during the iterations, representing the alternatives for the next steps in a more compact visualisation than the current interaction and using the same prioritisation function for sorting them.

5 ANATOMY OF VIRGILITES

The design of ViRgilites consists of three parts, as outlined in Figure 2:

the left panel (Figure 2-Pa) contains information on the previously completed action and the interaction selected for completing it,

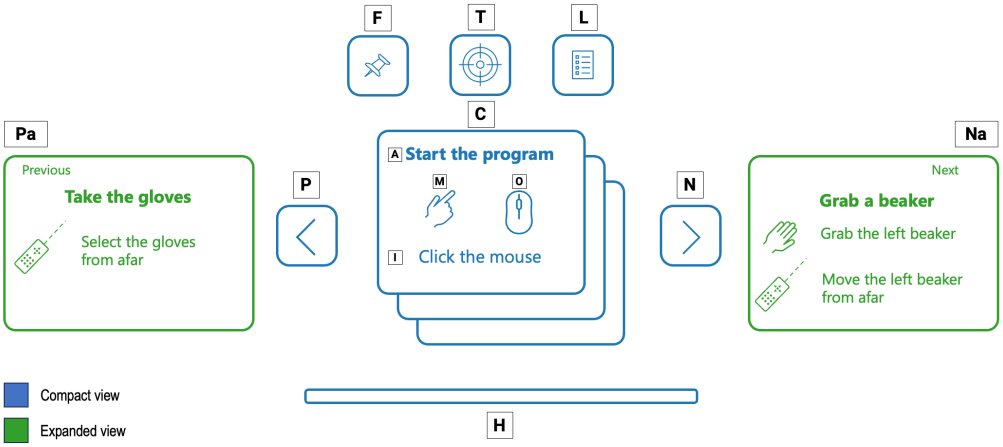

Fig. 2. Organisation of the ViRgilites guidance interface. It contains three panels: the middle (C) containing information on how to execute the current action, the left (Pa) on the previous successful action (and interaction), while the right on the next possible action and the associated interactions (Na). Users browse the available interactions (if more than one) using two dedicated buttons (P and N). The top-left push button activates or deactivates the pinned mode (F), the top-center (T) activates the target object highlighting, while the top-right (L) activates the compact (blue part) or expanded (blue plus green) view mode. The bottom handle (H) enables the reposition of the user interface in the virtual space.

the middle and main panel (Figure 2-C) displays how to execute the current action and presents alternative interactions that have the same result, and

the right panel (Figure 2-Na) shows information about the next action that will follow when the current one is completed, and the corresponding supported interactions.

ViRgilites presents the current context of a task by showing the latest previous interaction thus informing the user what was done to get to the current state, and by presenting what the next actions will be when the current one is completed successfully. Notice the next action (right panel) can change while exploring the options for current action (middle panel), thus providing feedforward on what can or should be done when completing the current action in a particular way. For example, consider the task of moving a box from one shelf to another. If the user is considering a grab action the box from the first shelf using the virtual hand modality, the next action of the same task is navigating the VR environment until reaching the second shelf, which will appear in the right panel (Figure 2-Na). An equivalent action in a modality supporting selection from afar (e.g., laser pointing) does not need an intermediate navigation, the user can directly drop the box from afar, and the action in the right panel will be updated accordingly. Section 6.2 describes our approach to calculating the order in which the suggestions are displayed. The middle panel represents the alternative interactions to reach the same result as a stack of cards (Figure 2-C). The user can navigate through this stack and explore and select different ways to complete the current actions (Figure 2-P and N). The panel on the left (Figure 2-Pa) contains feedback informing the user on the last completed action. ViRgilites uses it both as a task completion notification, moving a card from the central to the left to represent the progression and as a support for helping users recall past actions. We limited the number of past actions to one. The card representation is mostly the same as used in the central part, with the differences of the removal of the interacted object and the rearrangement of the information of the card.

The panel on the right (Figure 2-Na) contains further feedforward information related to the overall procedure goal. It shows a preview of the next steps in the procedure, provided that the user completes the current action using the modality selected in the card deck. Therefore, the content in the right panel *depends* on the selection in the card deck. The representation in the next action panel is slightly different from the others. First, it groups equivalent alternatives for executing the same action in a single group. The equivalent effect is the title of a list of modalities and operating suggestions. Second, it lists the various alternative modalities in a list representing branches in the procedure. We use the prioritisation function to order the available interactions in the card (see Section 6.2). In summary, the next action panel offers insight into the next step considering the current modality choice. In this way, the user can decide how to continue, considering the current action and the next step.

5.1 Exploring and Selecting the Current Interaction

The cards are sorted using the adaptive prioritisation function we discuss in Section 6.2, to help the user quickly find the most appropriate way to complete the current action. While the system controls the ordering of the alternatives at the beginning of each action, users are in charge of selecting the appropriate modality: they can keep the modality suggested by ViRgilites, or they can browse the card stack (Figure 2-C) until they find the most appropriate for them. Each card contains four pieces of information. The first is the goal of the current action (Figure 2-A), which anticipates the consequence of the interaction in the procedure completion context. The second is an icon representing the modality associated with the current card (Figure 2-M), together with another icon that represents the target object (Figure 2-O), which is a snapshot of the 3D object in the environment. The fourth is a label describing how to operate the interaction using the considered modality (Figure 2-I). The cards in the central part of the interface list the possible actions for continuing the procedure. They may be equivalent ways for activating the same function depicted as cards having the same title and target object but different modalities and operating suggestions. They may also represent branches in the procedure, i.e., entry points for different paths (sequences of tasks) leading to the same overall goal.

5.2 ViRgilites Placement

The pin push button in the middle part (Figure 2-F) allows users to control the position of the guidance interface in the 3D environment. By default, the pin button is not active, and when activated, the panel “follows” the users changing its position and orientation that guarantees a comfortable interaction in the user’s field of view[2]. ViRgilites freezes its current position and orientation in the 3D environment when the pin button is pushed once again. Regardless the state of pin button, the user can move the interface around using the handlebar (Figure 2-H) at the bottom. Users can then interact with the XR environment without the panel occlusion. Instead, the list button (Figure 2-L) hides or shows the left and right panels, switching between a compact and an extended version of the guidance interface.

5.3 Navigation aids

Navigating smoothly through a virtual environment is crucial for a user-friendly experience, especially when the user must interact with dispersed virtual objects. Our system addresses this by incorporating specific navigation features geared toward user-object spatial interactions. Interacting with the icon of the virtual objects in the top card in the central panel (Figure2-T), prompts the system to guide the user directly to a virtual object’s location. This guidance can manifest in various

---

[2] please see https://learn.microsoft.com/en-us/windows/mixed-reality/design/billboarding-and-tag-along

forms, tailored to the most probable way the user will interact with the object. If the object is within reach and the user is expected to simulate real-world interaction, an arrow and line will appear, directing them toward the object until the action is completed. Conversely, if close proximity to the object is required –like touching or grabbing it– and it's too far away, the system will automatically teleport the user closer. These added features augment the existing navigation options provided by the standard VR setup.

6 META-LEVEL INTERACTIONS: TRIGGERS, PROGRESS AND ADAPTATION

In the previous section we introduced the concept of multi-level feedforward for Extended Reality environments. ViRgilites can provide users with full flexibility to decide how they want to interact. They can navigate through the different options and pick the one that best suit their needs or preferences for completing tasks.

6.1 Triggering the interaction guidance

The effectiveness of a guidance system hinges not just on the information it provides, but also how and when that information appears [34]. An essential aspect of its design is figuring out the right moments to activate and show the guidance. We call this a 'meta-interaction' because it shapes the way users interact with particular parts of the interface that guide the user on how to interact with the system [13].

One of such triggers to start ViRgilites is a change in *context* [14]. A change in context could imply that some ways of interacting in the virtual environment might become more or less effective, thus should be taken into account by the guidance system. In our work, we have exploited the ideas of sensing and modelling context for reacting to its changes, since it is a widely investigated topic, and there are different dimensions relevant for adaptive behaviours such as user characteristics [39], physical elements [57] or social context [30]. In some models, the task itself is part of the context [52]. In our work, changes in the context result in a different prioritisation of the steps in the training procedure described in Section 6.2. As a result, the guidance system may suggest another modality for completing the same task or an alternative path for achieving the same goal. For instance, consider a scenario where a user has the option to either utter a voice command or employ tactile interaction with an object. The optimal interaction modality depends on the ambient noise level. In a high-noise context, tactile interaction would be prioritised as the most efficacious mode of interaction. Conversely, in a low-noise environment where the user must navigate between distant objects, voice commands would be promoted as the most efficacious mode.

6.2 Adaptation and Prioritisation of the Steps

The middle pane of ViRgilites presents a stack of cards, containing all alternative interactions that can be performed at the current time, and that all have the same result. The suggested interaction is selected according to a priority function. We rank these alternatives using a prioritisation function that takes into account two classes of parameters: the requirements from the application domain and the context in which the interactions take place. Then, the best interaction is suggested to the user, through the user interface.

We represent a procedure as a set of interactions I, and each interaction $i \in I$ is associated with an interaction modality $m(i)$. Therefore, prioritising an interaction also prioritises the associated modality. Each function defines a set of parameters, and we represent them with the vector $\mathbf{v}_i$. We consider another vector $\mathbf{w}_{m(i)}$ of the same length for defining the weight (i.e., the importance) of each parameter for the given interaction modality. Both vectors contain values between zero and one, and the weights sum to one. Once the parameters have been identified, the priority of an action is simply the dot product $p_i = \mathbf{w}_{m(i)} \cdot \mathbf{v}_i$. We do not calculate the priority of all possible actions at each step, but we use the definition of the procedure for obtaining, given an interaction i, the set of its possible subsequent interactions $nxt(i)$ and the set of its alternatives $opt(i)$. ($nxt : i \longrightarrow B, i \in I, B \subseteq I$), and the set of its alternatives ($opt : i \longrightarrow B, i \in I, B \subseteq I$)

A similar approach is used for the visualisation of the list of the interactions that the user will do on the next step. In this case, the list is sorted following the same algorithm used for the suggestion of the best possible interaction, but uses an heuristic variant to evaluate the rank of each interaction.

We use both *static* and *dynamic* weights in our weights vectors. Static weights represent upfront, predefined values, such as the importance of using a specific modality for the application domain. They encode the designer's knowledge of different aspects of the interaction and domain. Dynamic weights, on the other hand, represent values that can change over time in response to interaction or context-related events, such as the distance between the user and the target object. This can lead to different recommended interaction modalities for the same set of available interactions for the same action, depending on the situation.

To easily add or change the way prioritisation is done, we use a Strategy pattern [19]. This allows using ViRgilites in a wide variety of situations and application domains, and even change the approach for different types of users. The following parameters are considered when recommending interaction modalities:

- **Interaction fidelity** (static, designer-defined). The degree of correspondence between the current action and its real-world counterpart. For instance, a direct manipulation of an oven temperature knob is more faithful than a remote selection.
- **Target Distance** (dynamic, derived from the XR environment). The inverse of the physical distance between the user and the target object of the current action. The designer relies on the modality for establishing the weight (e.g., higher for direct manipulation).
- **Modality Change** (static, designer-defined). It estimates the cost for the user of a possible modality change for executing the next step. Denoting the current interaction as i and having a matrix C containing pre-defined costs for the change of each possible pair of modalities, the parameter value is $1 - \min_{n \in nxt(i)} C_{m(i),m(n)}$.
- **Context Information** (dynamic, derived from the context model). It expresses the feasibility of the current modality and/or action considering the current context model. The definition of this parameter is tightly coupled with the training procedure, and it may include environment sensing (e.g., avoiding voice commands in noisy environments), tracing previous executions of the procedure by the same trainee for identifying preferred interaction modes or prioritising unexplored procedure branches.

For a better understanding of how the algorithm selects an interaction with respect to another, we provide the following example in which we explain how our particular algorithm chooses between different types of interactions with objects. Imagine we have three interactions for selecting the same target object (e.g., collecting a screwdriver from a toolbox).

Grab - a 'grab' interaction
Far - a 'far' interaction (interaction from a distance)
Touch - a 'touch' interaction

In our scenario, the user's next task is to touch another object. For our example's purpose, let's assume:

- *Touch* and *Grab* have high 'interaction fidelity' (quality or realism of the interaction).
- *Far* has low interaction fidelity.

The system will assign 1 as interaction fidelity to *Touch* and *Grab* and 0 to *Far*. *Touch* and *Grab* involve a greater distance to the target than *Far*. The algorithm also considers the 'modality' of the

Table 1. Example of Modality Costs table. The table shows the effort needed from current modality (row) to next modality (column). We evaluated every cell using the same criteria (hence the discrepancy in an inverse reading of the table). Higher values are better (0 -> Very hard to switch between modality, 1 -> no effort on changing).

	Gaze and Commit	**Touch**	**Grab**	**Voice**	**Far Interaction**	**None**
Gaze and Commit	1	0	0.5	0.5	0.5	0
Touch	0.5	1	0.5	0.5	0	0
Grab	0.5	0.5	1	0.5	0	0
Voice	0.5	0.5	0.5	1	0	0.5
Far Interaction	0.5	0	0.5	0.5	1	0.5
None	0.5	0	0.5	0	0.5	1

Table 2. Final Values for score evaluation. The chosen modality is underlined in bold. Context evaluation has been omitted since they scored all 1 and its contribution does not influence the final choice.

	Action Fidelity	**Distance from Target**	**Modality Change Cost**	**Final Value**
Grab	1	0.9	0.5	0.875
Far Interaction	0	1	0	0.5
Touch	**1**	**0.9**	**1**	**1**

interaction (the way the interaction is performed). Since a far interaction (like *Far*) can be done from anywhere in the environment, it scores higher than *Touch* and *Grab* in this aspect. In this case we can assume that *Far* gets 1 where *Touch* and *Grab* get less than 1 (we can say that, in a 10x10m square, the user is 1m away from the target object, so the final value will be 0.9).

The algorithm uses a predefined matrix to calculate the change in modality (the shift from one type of interaction to another, see Table 1). In our case:

- *Touch* scores 1, as we have defined that changing from one touch interaction to another is effortless.
- *Grab* scores 0.5, since from a grab interaction to a touch interaction the effort is minimum.
- *Far* scores 0 because for the user to touch the object he also needs to move and approach the object.

However, since there are no 'context changes' (factors related to the specific situation or environment), all interactions initially receive 1.

Now, assume the designer prioritized interaction fidelity and modality change equally in the algorithm ($w_{fidelity} = w_{modality} = 0.25$ but did not consider target distance ($w_{distance} = 0$) and placed a high emphasis on context information ($w_{context} = 0.5$). Given these settings, the algorithm will ultimately recommend *Touch* as the best choice (*Grab* = 0.875, *Far* = 0.5, *Touch* = 1 - see Table 2).

7 ARCHITECTURE OF THE SYSTEM

This section discusses the architecture of ViRgilites. Unity and the Microsoft MRTK 3 [32] toolkits are used for creating the virtual environment and managing the interactions. ViRgilites to supports the execution of a series of complex interactions in a VR environment without prior experience or knowledge of how the environment works. We validated this by using ViRgilites in two scenarios: one for training users in chemistry lab tasks using nuclear materials, and the other for training users in performing food preparation tasks in the kitchen. Both scenarios require users to execute a

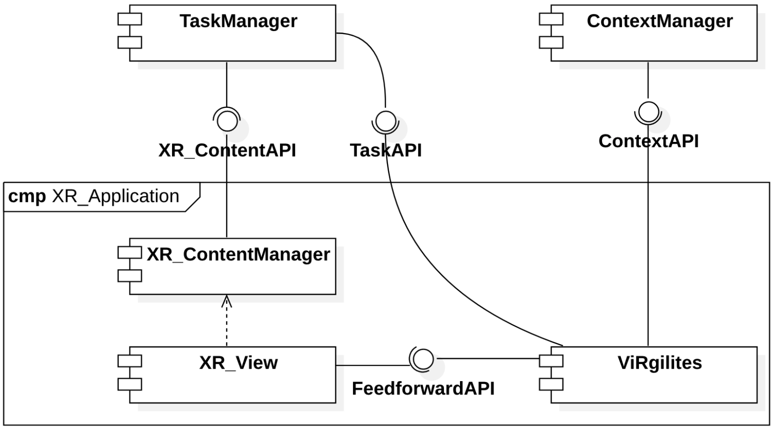

Fig. 3. UML component diagram of the ViRgilites architecture. The *TaskManager* and the *ContextManager* provide the information to the feedforward component (ViRgilites). The *XR_ContentManager* synchronises the state of virtual and real objects that blend in the *XR_View*.

series of tasks that can be done in various ways; users can perform tasks like they would do in real life and train their motor skills too, or users can make use of other, more convenient, interaction modalities so they can focus on the high-level steps that need to be executed.

Figure 3 shows the components included in the prototype. The XR application consists of three main components. The *XR_View* visualises the XR content and interfaces, blending the virtual and real elements and managing the interaction. Such visualisation depends on the current state of the objects in the XR environment (both real and virtual).

An important requirement in a training setting is that the XR environment supports different training procedures, which may be modified or changed according to the varying learning needs. Therefore, hard-coding the procedure management in the XR environment limits its reuse. To overcome this problem, we designed the prototype applying the solution proposed by Artizzu et al. [1]. This solution proposes the reproduction of virtual environments that can support more than one procedure, and by doing so not being hard-coded and locked to a single experience. These environments, called *templates*, then expose the control of managed (e.g.: prepared beforehand in a way that can be retrieved by procedure actions) XR objects through the *XR_ContentAPI*. The advantage of such an organisation is twofold. On the one hand, it allows the modelling of the environment characteristics relevant for goal-oriented applications in XR. On the other hand, it allows the creation of goal-oriented procedures without requiring further builds of the XR application, thus enabling domain experts to author procedures without involving developers.

The *TaskManager* contains the logic for executing a training procedure (i.e. a task). In our implementation, its definition consists of a text file containing the name of the task and a list of *Action* elements containing the following information: i) the position of the action in the procedure; ii) the action description. In turn, each action contains at least one or more *Interactions*, representing alternative ways for completing the same step. An interaction contains the following information: i) two values defining the subject and the target of the action to be performed (e.g.: if the user have to do something to an object called "Pan", then the subject would be "Player" and the target would be "Pan", if it is a collision between two objects, "egg" and "Pan", subject will be "egg" instead); ii) the modality required for performing the interaction; iii) a text describing how to perform the interaction; vi) the array of values described in section 6.2. The *TaskManager* exploits the

XR_ContentAPI for receiving notifications about the current interactions and updating the runtime state of the current task. Given the structure of this component, the prototype can execute different procedures for the same environment by simply loading other definition files.

Besides controlling the procedure progress, the *TaskManager* exposes the procedure state through the *TaskAPI*, which notifies the subscribing components about the completion of an interaction, the history of the previously executed actions and allows forward navigation of the possible options for continuing the procedure execution. The *TaskManager* supports a Unity event publisher that can be triggered for signalling the change of the task triggered by the correct execution of an action. Depending on the events that are triggered and the status of the application, The *TaskManager* can set active tasks, it can check whether tasks were completed, it can check whether a whole procedure was completed, it can provide the list of actions for a given step, and manages the progress of the procedure. ViRgilites exploits the APIs for controlling the information provided in the *XR_View*. As described in Section 4, such presentation depends on the context. Considering the variety of entities and sensing devices contributing to building a context representation, we isolated its management into a dedicated component, again external to the XR Application. This solution allows having a context representation that fits the considered environment. Indeed, some require only simple information on the state of the world, while others may need to model even complex social relationships. The *ContextManager* is responsible for providing an API accessing the context information, hiding the modelling and sensing complexity to its subscribers. In particular, the APIs provide an event publisher that notifies the system of context changes (this event is triggered by the sensors in the user environment - managed by the *ContextManager* - and subscribed by the *TaskManager* in the proposed implementation). The component responsible for feedforward presentation instantiates the user interface depicted in Figure 2, utilizing MRTK 3 as the foundational framework [32]. It employs the prioritization algorithm described in Section 6.2 and interfaces with the *Context API* to query the context of use. The system constantly updates the guide based on both task conditions and the overall context. Finally, it allows the *XR_View* component to control the guidance system through the XR devices. ViRgilites updates the ranking of the "next" panel list using the associated triggered function, and it signals the change itself (mainly used for notifying the user of the update through alerts in the *XR_View*). ViRgilites implements functions for listening to context changes (published by the *ContextManager* and used by prioritisation algorithms that require assessing the context as a parameter) and a function that provides the final suggestion to the middle panel list. The system also takes from MRTK the default player prefab, which includes hand management support[3]. The hands, when used with supported XR devices (e.g.: Meta Quest 2, 3 and Pro) enable the interaction with other objects in the scene using poke, grab, far ray, and gaze pinch interactors. To enable the communication between an MRTK-supported game object (i.e. a Unity GameObject including an MRTK-provided component such as PressableButton, ObjectManipulator or StatefulInteractable Components) and ViRgilites, it is required to insert a script that forwards the MRTK-interaction event towards the ViRgilites component for updating the guidance system.

7.1 Example Scenarios

To demonstrate the system's validity, we developed two simple scenarios, one depicting a waste disposal procedure for nuclear material in a chemistry lab and the other showing how to cook an Italian focaccia in a kitchen environment.

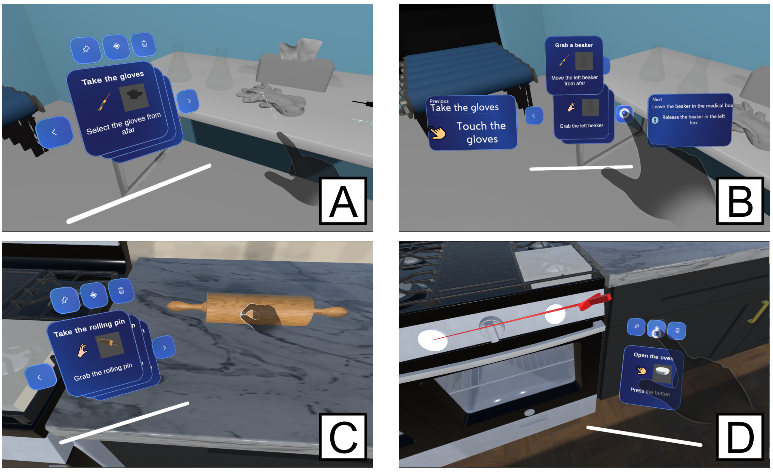

Fig. 4. ViRgilites interactions in the chemistry lab (A, B) and kitchen (C, D) scenarios. Figure A shows the user picking up the gloves from afar. Figure B shows the user changing the interaction with another one. Figure C shows the user grabbing the rolling pin. Figure D shows the user trying to identify which object he should interact with.

7.2 Chemistry Lab Scenario

This scenario is situated in a chemistry lab. To mitigate the risk of contaminating the surroundings or propagating hazardous substances, users are tasked with disposing of a beaker in a designated waste basket. This task requires precision and close adherence to the protocol. The sequence of actions to execute is as follows:

(1) Start the computer program for registering the activities in the lab using the computer mouse in the scene;
(2) Pickup a pair of gloves from a table and put them on;
(3) Request a beaker from outside;
(4) Grab the hazardous waste from the table;
(5) Insert the hazardous waste in the beaker;
(6) Grab the beaker from the table;
(7) Position the beaker into the waste basket

At the beginning of the procedure, the interface initially displays itself in compact mode, directing the user to start by clicking the computer mouse. Initially, there are no alternative interactions. The user then uses standard MRTK gestures (as described in section 7) to teleport close to the computer desk and clicks the mouse. Once the mouse has been clicked, the interface updates to show the next step, which is grabbing some gloves. For this action, the system suggests using a near interaction for picking up the gloves, but after the trainee changes position, the prioritisation algorithm re-evaluates the parameters and the context of the environment, and determines that a far interaction is now more suitable for this step. The interface promptly switches to the correct recommendation when this happens. Once the gloves has been acquired (as seen in Figure 4 - A), the interface updates again, instructing the user to request a beaker from another room to put the hazardous materials inside of it. This time, the user has the choice to either use a voice command

---

[3] https://learn.microsoft.com/en-us/windows/mixed-reality/mrtk-unity/mrtk3-input/packages/input/hand-tracking#hand-controller-prefabs

("Give me the beaker") or touch a button to get it manually. Since the user environment is quiet, there are no context changes related to it being noisy, and because of that, the system suggests using the voice command. The user tells the keyword, and the beaker appears in the scene. Now, they have to put the hazardous waste inside the beaker, which is positioned on the table near the waste baskets, and they can do it either by grabbing, ray selecting or gaze committing to the object. At this point, the user wants to know the next action to decide on the appropriate modality and switches to the extended view of the interface. In the next task panel, the user sees that they need to move the materials in the beaker. Since the user is far from the materials, they select the "gaze and commit" modality, look at the object and make a pinch gesture to start the movement phase; the user then inserts the materials inside the beaker by moving them near the beaker. The next task is to pick up the beaker. Here, the user is presented with the choice of grabbing the beaker, selecting it from a distance or gaze-committing from a distance as well, and the system recommends the far interaction, since according to the prioritisation formula, it is the most suitable interaction modality. Once again, having a look at the extended view of the interface, the panels show that they need to move the beaker into the waste basket. This time, they swap the card and select the grab interaction (Figure 4 - B). Finally, the user's task is to take the beaker to the waste basket, as indicated by the extended view of the interface. When this task is completed, the procedure finishes, displaying the “Procedure complete!” message in the middle panel.

7.3 Kitchen Scenario

A similar process can be illustrated for the kitchen scenario, where the objective of the user is to prepare a focaccia. To achieve this, the user needs to adhere to the subsequent steps:

- Preheat the oven;
- Take the rolling pin;
- Flatten the dough with the rolling pin;
- Take the oil;
- Put the oil on the dough;
- Open the oven door;
- Bring the dough into the oven;
- Close the oven door (for cooking the focaccia);
- Open the oven door (after the focaccia has been cooked);
- Serve the focaccia on a plate;

First, when the user enters the scene, the interface displays the compact view, similar to the previous scenario. The user is tasked to touch the oven button to preheat the oven. There are no alternatives so the user proceeds to click the button. The second step tasks the user to pick up a rolling pin. The recommended modality is to grab the rolling pin remotely. Still, in this case, the user would like a realistic approach and opts for the arrow controls of the interface to select the grab interaction instead. Once the user has done that (Figure 4 - C), the interface updates, illustrating the next action: flattening the dough on the table with the rolling pin, using any available means. In this case, the user has to bring the rolling pin using any possible interaction technique available, but since he already took the rolling pin with a grab interaction he proceeds to complete the action simply by moving the rolling pin to the dough.. Now, since the user is distant from the bottle of oil in the scene, when the task for picking up the oil is presented, the system promptly changes the suggested interaction from grab to far interaction. The user this time is fine with the choice of the system and picks the oil bottle from afar. The subsequent interaction is similar to the flattening of the rolling pin so the user brings the oil bottle to the dough. Now, the system tells the user to open the oven door using a button. Unfortunately, the button icon is similar to the oven on/off button, so when the user presses the original button nothing happens. In order to disambiguate the image, he presses the locator button to visualise the arrow with the line pointing to the correct object. At the same time, since the selected modality is a touch interaction, the user is teleported near the target object to facilitate interaction with it (Figure 4 - D). Once the user clicks on the button, the interface asks the user to bring the focaccia to the oven grate. The user picks the focaccia from afar and places it into the oven. The last actions of the users are to close the oven by touching the correct button, opening it again (for the purpose of the scenario the focaccia is instantly cooked once the oven door closes) and bringing the focaccia into a plate for serving it. Once again, the interface will display the “Procedure Complete” message, for communicating the end of the procedure.

8 FORMATIVE USER STUDY

We conducted a formative user study with the goal of explore potential issues to make ViRgilites more usable and increase the acceptance rate. We invited 10 participants (1 female, 9 male, age range: 23 - 36 years old, Avg.: 26,8, Median: 27, SD: 3.823). In particular, we invited 5 participants with self-declared proficiency in Virtual Reality environments, and 5 participants with self-declared little to no experience. Participation was voluntary and no compensation of any kind has been handed out.

8.1 Method

The evaluation, after collecting the consent of the participants and their demographic data, was structured in four parts.

Tutorial: The participants were introduced to the interface, explaining its goal and its various parts, using a VR test scenario. The participants had to complete a simple task, while using ViRgilites and being guided by the evaluator on how to interpret the language of the interface. The task consisted in selecting three cubes in order (red first, green second, blue as last). Moreover, in a separate area of the same scenario, a set of cubes were available to be freely manipulated, and a nearby panel (with a different layout than the one used by ViRgilites) would show the same icons of the interface of ViRgilites and a text explanation of the performed modality.

Waste Management Scenario. In the second part, the participants performed 4 tasks in a virtual chemistry lab environment, while having the possibility to provide feedback after each task. After each task, the participants were asked about their level of comprehension of the user interface and, if multiple interactions were available for the current action, the understanding of the interaction suggestion based on the context. All ratings used a 5-point Likert scale. The four tasks performed by the participants were the following:

- Task 1 (T1): Perform an action with a single interaction available. The participants needed to touch a mouse on a nearby desk;
- Task 2 (T2): Perform an action with multiple possible interactions. The suggested interaction was the one to perform. The participants needed to perform a far interaction selection of a pair of gloves on a desk;
- Task 3 (T3): Perform an action with multiple possible interactions. The interaction to perform was not the suggested one. The participants needed to perform a grab interaction on a beaker in the same desk of the gloves.
- Task 4 (T4): Perform an action with a single interaction available. The participants needed to bring the beaker taken from the previous task to a nearby medical box. The difference between this task and the first one is that in this task there is no recommended interaction, and the task would have been completed as long as the beaker reached the medical box.

Kitchen Scenario. In the third part, the participants were required to complete a full task from start to finish without interruptions between the actions. The task (T5) to be completed is the same depicted in Section 7.3. Participants were free to complete the actions in the task as they seen fit, it was up to them whether they wanted to change the interactions or keep using the ones suggested by the system. At the end of the task the participants were asked the questions in the previous part once more.

Semi-structured interview. In the last part we collected qualitative comments on the features of the system, any positive and negative comments about it, whether if they would have had ideas for improving the interface, and whether if they felt the level of feedforward implemented in the system would help an user in understanding how to do the task that the system is requiring him to do, and additional comments on the ranking system for the next action.

8.2 Results

8.2.1 *Feedforward Comprehension.*

Over the five tasks, the participants were asked to answer the following question on a Likert scale from 1 to 5: "Did you understand how to do the interaction based on the information in the interface?", where 1 was "Not at all" and 5 was "Completely". This question aimed to gather information on the level of helpfulness the feedforward system ViRgilites currently implements.

As shown in Table 3, T1 scored the lowest score above all (Avg.: 3.8, Median: 3.5, SD: 0.92, Mode: 3, Min: 3, Max: 5), and the same behaviour is reflected if we consider the scores of the participants proficient in VR environments (Avg.: 3.8, Median: 4, SD: 0.84, Mode: 4, Min: 3, Max: 5) and the novice participants (Avg.: 3.8, Median: 3, SD: 1.09, Mode: 3, Min: 3, Max: 5). This can be expected since, even after the training session, we can consider T1 as a period of accustoming to the visual language of the system. The other tasks scored average values more or equal than 4.5 (T2: 4.9, T3: 4.7, T4: 4.5, T5: 4.7), and it is interesting to notice that the average scores for each single task tends to be generally higher with the novice participants with respect to the scores of the proficient ones, with the exception of T5, that shows the opposite trend (T2: 5 vs 4.8, T3: 5 vs 4.4, T4: 4.8 vs 4.2, T5: 4.6 vs 4.8).

8.2.2 *Automatic Interaction Suggestion.*

For inferring whether the participants understood how the Interaction Suggestion system suggested a particular interaction with respect to the other ones we asked them, when the test task had an action with more than one interaction (T2, T3 and T5) the following question: "Based on the hypothetical scenario we gave you, did you understand why the system recommended you that interaction technique?", using the same Likert scale and extremes as described in Section 8.2.1.

Table 4 shows the scores for this question. In this case, they were lower with respect to the feedforward comprehension (T2: 3.9, T3: 3, T5: 3.9) showing nonetheless that there was a general understanding of the feature. In particular, it is interesting to notice that novice participants had a worse understanding of the feature with respect to the proficient ones (T2: 3.6 vs 4.2, T3: 2.4 vs 3.6, T5: 3.8 vs 4) meaning that experienced participants could grasp more the reasoning behind the selection of an interaction over the other available ones.

8.3 Discussion on Feedback

After gathering the experience of the participants we can report on the feedback provided by them.

Helpfulness of the feedforward interface. All participants agreed on the utility of the interface and its feedforward features, when explicitly asked to give an opinion on it. During the test tasks, the participants has been asked on which elements of the UI helped them understand how to do the action. Over the course of the five tasks, the most helpful element was the text (29 times out of 50,

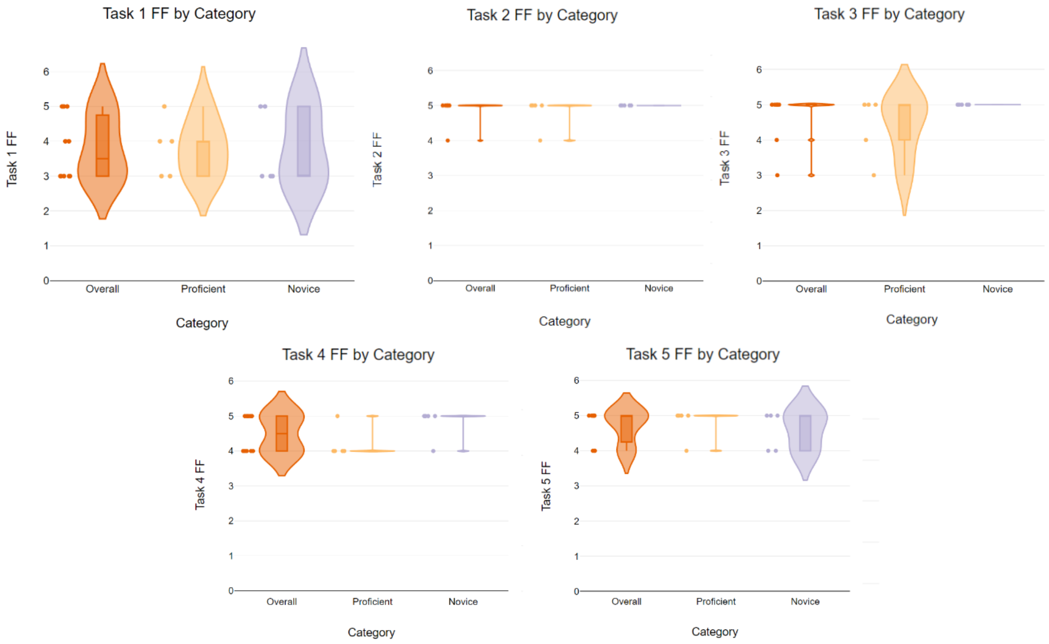

Fig. 5. Violin box plots of the Feedforward comprehension results

Table 3. Evaluation Results for Feedforward comprehension from the participants

		Average	Median	STD	Mode	Min	Max
Task 1	**Overall**	3,8	3,5	0,92	3	3	5
	Only Proficient	3,8	4	0,84	4	3	5
	Only Novices	3,8	3	1,09	3	3	5
Task 2	**Overall**	4,9	5	0,32	5	4	5
	Only Proficient	4,8	5	0,45	5	4	5
	Only Novices	5	5	0	5	5	5
Task 3	**Overall**	4,7	5	0,67	5	3	5
	Only Proficient	4,4	5	0,89	5	3	5
	Only Novices	5	5	0	5	5	5
Task 4	**Overall**	4,5	4,5	0,53	4	4	5
	Only Proficient	4,2	4	0,45	4	4	5
	Only Novices	4,8	5	0,45	5	4	5
Task 5	**Overall**	4,7	5	0,48	5	4	5
	Only Proficient	4,8	5	0,45	5	4	5
	Only Novices	4,6	5	0,55	5	4	5

58%), followed by the interaction technique icon (27 times out of 50, 54%) and the object image (19 times out of 50, 38%).

Utility of the automatic Interaction Suggestion. P6 and P8 appreciated the automatic selection of the interaction, P6 stated that because it removed the need to select the most reasonable one for the action to perform.

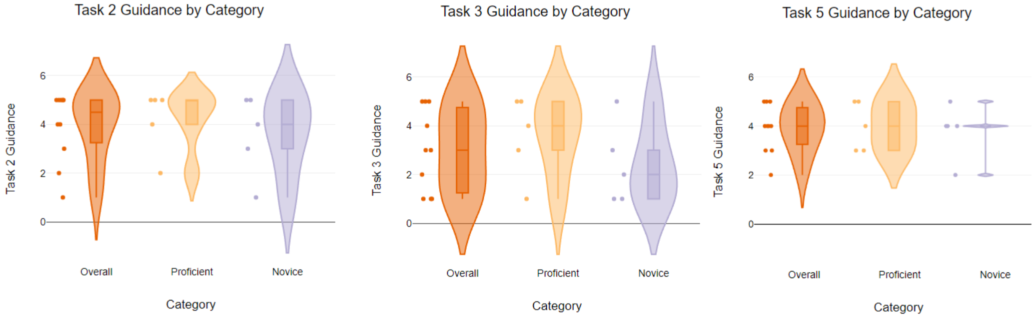

Fig. 6. Violin box plots of the Automatic Interaction Suggestion results

Table 4. Evaluation results of the Automatic Interaction Suggestion from the participants

		Average	Median	STD	Mode	Min	Max
Task 2	**Overall**	3,9	4,5	1,45	5	1	5
	Only Proficient	4,2	5	1,30	5	2	5
	Only Novices	3,6	4	1,67	5	1	5
Task 3	**Overall**	3	3	1,70	1	1	5
	Only Proficient	3,6	4	1,67	5	1	5
	Only Novices	2,4	2	1,67	1	1	5
Task 5	**Overall**	3,9	4	0,99	4	2	5
	Only Proficient	4	4	1	5	3	5
	Only Novices	3,8	4	1,09	4	2	5

Object Locator is useful but improvable. The object locator, in the same questions as above, was less voted as useful (10 times out of 50, 20%), but in the open questions the participants agreed on its usefulness (P1, P2, P6, P9). P6 and P7 denoted that it may have been unnecessary to teleport the user when the locator button is pressed and the interaction technique to perform is either a touch or a grab one, because of the unexpected positioning of the teleportation (P6). P2 and P7 noticed the teleportation being too near to the target object, and P7 suggested to make the teleportation to a nearby position of the target object, instead of the object itself.

Utility of the Next panel A particular pain point that most of the participants (P1 - P5, P7, P9) stated, when asked about positive or negative aspect about the "Next" panel was that they did not feel the need to use it or they just forgot to have it. One participant (P6) appreciated its value and features ("It is very useful to already see the ranking of the modalities beforehand to know what is expected of you next and how it is done "best"). We can infer that the feature, despite not being prominent in the user interface, has potential to be useful in certain situations, but most of the times becomes an accessory information, not vital to the task execution. P5 and P8 suggested, as an improvement for the system, to keep the extended information on by default, and P8 also suggested to make the visualisation of the interface semitransparent, in a way that does not disturb the view of the user, while retaining its functionality.

Improvements to the system. Many participants took the opportunity, when asked, to express ideas for improving the overall experience with the system. In this paper we are going to report the suggestions strictly inherent to the user experience with the interface, improvements of the toolkit itself will be omitted, because out of scope for this research. P2, P4 and P8 suggested a different way to identifying the object, by highlighting it instead of pointing an arrow at it. P4 also suggested to indicate an object being out of sight by using a lateral arrow to the field of view, like it was used by Lallai et al. [27]. P1 suggested to put a sound trigger when the action was successfully performed, instead of relying solely on the user interface. P8 suggested to show only two cards when two interactions were available, because it felt confusing to see the discrepancy between number of cards and number of alternatives. Finally, P7 suggested to keep the position of the interface relative to the position of the user, instead of being absolute with respect to the position in the environment.

9 CONCLUSIONS

In this paper, we introduce ViRgilites, a system designed to help users navigate and interact in a Virtual Reality (VR) setting. Discovering and using interactions in VR applications is often cumbersome, because of the wide variety of possible interactions that have the same outcome in combination with a diversity of implementations of interaction techniques. ViRgilites offers a meta-user interface that supports users to *discover, select and perform* various multimodal interactions. In addition, ViRgilites helps users understand what tasks they can complete, shows how to do them, and presents the user what comes next. It automatically suggests the best way to interact based on the context and designer preferences. We tested ViRgilites in a preliminary study that looked at two real-world scenarios, mainly focusing on user opinions about the design and layout of this new interface. The feedback we gathered demonstrated the usage of ViRgilites for goal-driven VR applications, and will enable us to further improve the design and behaviour of ViRgilites. Finally, we implemented a flexible prioritisation approach that recommends, steers or even constrains users to use the most effective interaction modalities to perform an interaction.

ACKNOWLEDGMENTS

This work is supported by the Italian Ministry and University and Research (MIUR) under the PON program: *“The National Operational Program Research and Innovation”* 2014-2020 (PONRI), the Italian MUR and European Union - NextGenerationEU under grant PRIN 2022 EUD4XR (Grant F53D23004380006), and by the Flemish Government under the *“Onderzoeksprogramma Artificiële Intelligentie (AI) Vlaanderen”* programme.

REFERENCES

[1] Valentino Artizzu, Gianmarco Cherchi, Davide Fara, Vittoria Frau, Riccardo Macis, Luca Pitzalis, Alessandro Tola, Ivan Blecic, and Lucio Davide Spano. 2022. Defining Configurable Virtual Reality Templates for End Users. *Proc. ACM Hum.-Comput. Interact.* 6, EICS, Article 163 (jun 2022), 35 pages. https://doi.org/10.1145/3534517

[2] SK Bailey, Cheryl I Johnson, Bradford L Schroeder, and Matthew D Marraffino. 2017. Using virtual reality for training maintenance procedures. In *Proceedings of the Interservice/Industry Training, Simulation and Education Conference.*

[3] Marc Baloup, Thomas Pietrzak, and Géry Casiez. 2019. RayCursor: A 3D Pointing Facilitation Technique Based on Raycasting. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland Uk) *(CHI ’19)*. Association for Computing Machinery, New York, NY, USA, 1–12. https://doi.org/10.1145/3290605.3300331

[4] Soumya C. Barathi, Daniel J. Finnegan, Matthew Farrow, Alexander Whaley, Pippa Heath, Jude Buckley, Peter W. Dowrick, Burkhard C. Wuensche, James L. J. Bilzon, Eamonn O’Neill, and Christof Lutteroth. 2018. Interactive Feedforward for Improving Performance and Maintaining Intrinsic Motivation in VR Exergaming. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (Montreal QC, Canada) *(CHI ’18)*. Association for Computing Machinery, New York, NY, USA, 1–14. https://doi.org/10.1145/3173574.3173982

[5] Olivier Bau and Wendy E. Mackay. 2008. OctoPocus: A Dynamic Guide for Learning Gesture-Based Command Sets. In *Proceedings of the 21st Annual ACM Symposium on User Interface Software and Technology* (Monterey, CA, USA) *(UIST ’08)*. Association for Computing Machinery, New York, NY, USA, 37–46. https://doi.org/10.1145/1449715.1449724

[6] Mark Billinghurst. 2013. Hands and Speech in Space: Multimodal Interaction with Augmented Reality Interfaces. In *Proceedings of the 15th ACM on International Conference on Multimodal Interaction* (Sydney, Australia) *(ICMI ’13)*. Association for Computing Machinery, New York, NY, USA, 379–380. https://doi.org/10.1145/2522848.2532202

[7] Jeremy Boy, Louis Eveillard, Françoise Detienne, and Jean-Daniel Fekete. 2015. Suggested interactivity: Seeking perceived affordances for information visualization. *IEEE transactions on visualization and computer graphics* 22, 1

(2015), 639–648.

[8] Eric Burns, Sharif Razzaque, Abigail T Panter, Mary C Whitton, Matthew R McCallus, and Frederick P Brooks. 2005. The hand is slower than the eye: A quantitative exploration of visual dominance over proprioception. In *IEEE Proceedings. VR 2005. Virtual Reality, 2005.* IEEE, 3–10.

[9] Alisa Burova, John Mäkelä, Jaakko Hakulinen, Tuuli Keskinen, Hanna Heinonen, Sanni Siltanen, and Markku Turunen. 2020. Utilizing VR and Gaze Tracking to Develop AR Solutions for Industrial Maintenance. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3313831.3376405

[10] Edwige Chauvergne, Martin Hachet, and Arnaud Prouzeau. 2023. User Onboarding in Virtual Reality: An Investigation of Current Practices. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems* (Hamburg, Germany) *(CHI '23)*. Association for Computing Machinery, New York, NY, USA, Article 711, 15 pages. https://doi.org/10.1145/3544548.3581211

[11] Feng-Kuang Chiang, Xiaojing Shang, and Lu Qiao. 2022. Augmented reality in vocational training: A systematic review of research and applications. *Computers in Human Behavior* 129 (2022), 107125. https://doi.org/10.1016/j.chb.2021.107125

[12] Sven Coppers, Kris Luyten, Davy Vanacken, David Navarre, Philippe Palanque, and Christine Gris. 2019. Fortunettes: Feedforward about the Future State of GUI Widgets. 3, EICS, Article 20 (jun 2019), 20 pages. https://doi.org/10.1145/3331162

[13] Joëlle Coutaz. 2006. Meta-User Interfaces for Ambient Spaces. In *Proceedings of the 5th International Conference on Task Models and Diagrams for Users Interface Design* (Hasselt, Belgium) *(TAMODIA'06)*. Springer-Verlag, Berlin, Heidelberg, 1–15.

[14] Anind K Dey, Gregory D Abowd, and Daniel Salber. 2001. A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. *Human–Computer Interaction* 16, 2-4 (2001), 97–166.

[15] Tom Djajadiningrat, Kees Overbeeke, and Stephan Wensveen. 2002. But how, Donald, tell us how? on the creation of meaning in interaction design through feedforward and inherent feedback. In *Proceedings of the 4th Conference on Designing Interactive Systems: Processes, Practices, Methods, and Techniques* (London, England) *(DIS '02)*. Association for Computing Machinery, New York, NY, USA, 285–291. https://doi.org/10.1145/778712.778752

[16] Katherine Fennedy, Jeremy Hartmann, Quentin Roy, Simon Tangi Perrault, and Daniel Vogel. 2021. Octopocus in VR: using a dynamic guide for 3D mid-air gestures in virtual reality. *IEEE Transactions on Visualization and Computer Graphics* 27, 12 (2021), 4425–4438.

[17] Andrea Ferracani, Marco Faustino, Gabriele Xavier Giannini, Lea Landucci, and Alberto Del Bimbo. 2017. Natural experiences in museums through virtual reality and voice commands. In *Proceedings of the 25th ACM international conference on Multimedia*. 1233–1234.

[18] UNESCO Institute for Statistics. 2012. International standard classification of education: ISCED 2011. *Comparative Social Research* 30 (2012).

[19] Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. 1995. *Design patterns: elements of reusable object-oriented software.* Pearson Deutschland GmbH.

[20] Tovi Grossman and George Fitzmaurice. 2010. ToolClips: An Investigation of Contextual Video Assistance for Functionality Understanding. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Atlanta, Georgia, USA) *(CHI '10)*. Association for Computing Machinery, New York, NY, USA, 1515–1524. https://doi.org/10.1145/1753326.1753552

[21] Daniel Hepperle, Yannick Weiß, Andreas Siess, and Matthias Wölfel. 2019. 2D, 3D or speech? A case study on which user interface is preferable for what kind of object interaction in immersive virtual reality. *Computers & Graphics* 82 (2019), 321–331.

[22] Teresa Hirzle, Jan Gugenheimer, Florian Geiselhart, Andreas Bulling, and Enrico Rukzio. 2019. A Design Space for Gaze Interaction on Head-Mounted Displays. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland Uk) *(CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–12. https://doi.org/10.1145/3290605.3300855

[23] Robert J.K. Jacob, Audrey Girouard, Leanne M. Hirshfield, Michael S. Horn, Orit Shaer, Erin Treacy Solovey, and Jamie Zigelbaum. 2008. Reality-Based Interaction: A Framework for Post-WIMP Interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Florence, Italy) *(CHI '08)*. Association for Computing Machinery, New York, NY, USA, 201–210. https://doi.org/10.1145/1357054.1357089

[24] Tiffany C.K. Kwok, Peter Kiefer, Victor R. Schinazi, Benjamin Adams, and Martin Raubal. 2019. Gaze-Guided Narratives: Adapting Audio Guide Content to Gaze in Virtual and Real Environments. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland Uk) *(CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–12. https://doi.org/10.1145/3290605.3300721

[25] Jérémy Lacoche, Thierry Duval, Bruno Arnaldi, Eric Maisel, and Jérome Royan. 2015. Plasticity for 3D user interfaces: new models for devices and interaction techniques. In *Proceedings of the 7th ACM SIGCHI Symposium on Engineering Interactive Computing Systems* (Duisburg, Germany) *(EICS '15)*. Association for Computing Machinery, New York, NY, USA, 28–33. https://doi.org/10.1145/2774225.2775073

[26] Jérémy Lacoche, Thierry Duval, Bruno Arnaldi, Eric Maisel, and Jérôme Royan. 2019. 3DPlasticToolkit: Plasticity for 3D User Interfaces. In *Virtual Reality and Augmented Reality*, Patrick Bourdot, Victoria Interrante, Luciana Nedel, Nadia Magnenat-Thalmann, and Gabriel Zachmann (Eds.). Springer International Publishing, Cham, 62–83.

[27] Giorgia Lallai, Giovanni Loi Zedda, Célia Martinie, Philippe Palanque, Mauro Pisano, and Lucio Davide Spano. 2021. Engineering Task-based Augmented Reality Guidance: Application to the Training of Aircraft Flight Procedures. *Interacting with Computers* 33, 1 (03 2021), 17–39. https://doi.org/10.1093/iwcomp/iwab007 arXiv:https://academic.oup.com/iwc/article-pdf/33/1/17/38495857/iwab007.pdf

[28] Joseph J LaViola Jr, Ernst Kruijff, Ryan P McMahan, Doug Bowman, and Ivan P Poupyrev. 2017. *3D user interfaces: theory and practice*. Addison-Wesley Professional.

[29] Klemen Lilija, Søren Kyllingsbæk, and Kasper Hornbæk. 2021. Correction of Avatar Hand Movements Supports Learning of a Motor Skill. In *2021 IEEE Virtual Reality and 3D User Interfaces (VR)*. 1–8. https://doi.org/10.1109/VR50410.2021.00069

[30] Paul Lukowicz, Sandy Pentland, and Alois Ferscha. 2011. From context awareness to socially aware computing. *IEEE pervasive computing* 11, 1 (2011), 32–41.

[31] Daniel Martin, Sandra Malpica, Diego Gutierrez, Belen Masia, and Ana Serrano. 2022. Multimodality in VR: A Survey. *ACM Comput. Surv.* 54, 10s, Article 216 (sep 2022), 36 pages. https://doi.org/10.1145/3508361

[32] Microsoft. 2022. Mixed Reality Toolkit Unity Plugin. https://github.com/microsoft/MixedRealityToolkit-Unity [Online; accessed 27-July-2023].

[33] Florian 'Floyd' Mueller, Tuomas Kari, Rohit Khot, Zhuying Li, Yan Wang, Yash Mehta, and Peter Arnold. 2018. Towards Experiencing Eating as a Form of Play. In *Proceedings of the 2018 Annual Symposium on Computer-Human Interaction in Play Companion Extended Abstracts* (Melbourne, VIC, Australia) *(CHI PLAY '18 Extended Abstracts)*. Association for Computing Machinery, New York, NY, USA, 559–567. https://doi.org/10.1145/3270316.3271528

[34] Andreea Muresan, Jess McIntosh, and Kasper Hornbæk. 2023. Using Feedforward to Reveal Interaction Possibilities in Virtual Reality. *ACM Trans. Comput.-Hum. Interact.* (jun 2023). https://doi.org/10.1145/3603623 Just Accepted.

[35] Allen Newell and Herbert Alexander Simon. 1972. *Human problem solving*. Prentice-hall Englewood Cliffs, NJ.

[36] Donald A Norman. 1999. Affordance, conventions, and design. *interactions* 6, 3 (1999), 38–43.

[37] Donald A. Norman. 2002. *The Design of Everyday Things*. Basic Books, Inc., USA.

[38] Donald A Norman. 2008. The way I see IT signifiers, not affordances. *interactions* 15, 6 (2008), 18–19.

[39] Charith Perera, Arkady Zaslavsky, Peter Christen, and Dimitrios Georgakopoulos. 2013. Context aware computing for the internet of things: A survey. *IEEE communications surveys & tutorials* 16, 1 (2013), 414–454.

[40] Krzysztof Pietroszek. 2018. *Raycasting in Virtual Reality*. Springer International Publishing, Cham, 1–3. https://doi.org/10.1007/978-3-319-08234-9_180-1

[41] Krzysztof Pietroszek. 2018. *Virtual Hand Metaphor in Virtual Reality*. Springer International Publishing, Cham, 1–3. https://doi.org/10.1007/978-3-319-08234-9_178-1

[42] Amotz Perlman Rafael Sacks and Ronen Barak. 2013. Construction safety training using immersive virtual reality. *Construction Management and Economics* 31, 9 (2013), 1005–1017. https://doi.org/10.1080/01446193.2013.828844

[43] Nimesha Ranasinghe, Pravar Jain, Nguyen Thi Ngoc Tram, Koon Chuan Raymond Koh, David Tolley, Shienny Karwita, Lin Lien-Ya, Yan Liangkun, Kala Shamaiah, Chow Eason Wai Tung, Ching Chiuan Yen, and Ellen Yi-Luen Do. 2018. Season Traveller: Multisensory Narration for Enhancing the Virtual Reality Experience. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (Montreal QC, Canada) *(CHI '18)*. Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3173574.3174151

[44] Rajinder Sodhi, Hrvoje Benko, and Andrew Wilson. 2012. LightGuide: Projected Visualizations for Hand Movement Guidance. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Austin, Texas, USA) *(CHI '12)*. Association for Computing Machinery, New York, NY, USA, 179–188. https://doi.org/10.1145/2207676.2207702

[45] Charles Spence, Jae Lee, and Nathan Van der Stoep. 2020. Responding to sounds from unseen locations: crossmodal attentional orienting in response to sounds presented from the rear. *European Journal of Neuroscience* 51, 5 (2020), 1137–1150.

[46] Anthony Steed, Tuukka M. Takala, Daniel Archer, Wallace Lages, and Robert W. Lindeman. 2021. Directions for 3D User Interface Research from Consumer VR Games. *IEEE Transactions on Visualization and Computer Graphics* 27, 11 (2021), 4171–4182. https://doi.org/10.1109/TVCG.2021.3106431

[47] Richard Stone, Elease McLaurin, Peihan Zhong, and Kristopher Watts. 2013. Full virtual reality vs. integrated virtual reality training in welding. (2013).

[48] Michael Terry and Elizabeth D. Mynatt. 2002. Side Views: Persistent, on-Demand Previews for Open-Ended Tasks. In *Proceedings of the 15th Annual ACM Symposium on User Interface Software and Technology* (Paris, France) *(UIST '02)*.

Association for Computing Machinery, New York, NY, USA, 71–80. https://doi.org/10.1145/571985.571996

[49] David Thevenin and Joëlle Coutaz. 1999. Plasticity of user interfaces: Framework and research agenda.. In *Interact*, Vol. 99. 110–117.

[50] Jo Vermeulen, Kris Luyten, Elise van den Hoven, and Karin Coninx. 2013. Crossing the Bridge over Norman's Gulf of Execution: Revealing Feedforward's True Identity. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Paris, France) *(CHI '13)*. Association for Computing Machinery, New York, NY, USA, 1931–1940. https://doi.org/10.1145/2470654.2466255

[51] S. A. G. Wensveen, J. P. Djajadiningrat, and C. J. Overbeeke. 2004. Interaction Frogger: A Design Framework to Couple Action and Function through Feedback and Feedforward. In *Proceedings of the 5th Conference on Designing Interactive Systems: Processes, Practices, Methods, and Techniques* (Cambridge, MA, USA) *(DIS '04)*. Association for Computing Machinery, New York, NY, USA, 177–184. https://doi.org/10.1145/1013115.1013140

[52] Heli Wigelius and Heli Väätäjä. 2009. Dimensions of context affecting user experience in mobile work. In *Human-Computer Interaction–INTERACT 2009: 12th IFIP TC 13 International Conference, Uppsala, Sweden, August 24-28, 2009, Proceedings, Part II 12*. Springer, 604–617.

[53] Huiyue Wu, Yu Wang, Jiali Qiu, Jiayi Liu, and Xiaolong Zhang. 2019. User-defined gesture interaction for immersive VR shopping applications. *Behaviour & Information Technology* 38, 7 (2019), 726–741.

[54] Hui-Yin Wu, Florent Robert, Théo Fafet, Brice Graulier, Barthelemy Passin-Cauneau, Lucile Sassatelli, and Marco Winckler. 2022. Designing Guided User Tasks in VR Embodied Experiences. *Proc. ACM Hum.-Comput. Interact.* 6, EICS, Article 158 (jun 2022), 24 pages. https://doi.org/10.1145/3532208

[55] LI Yang, Jin Huang, TIAN Feng, WANG Hong-An, and DAI Guo-Zhong. 2019. Gesture interaction in virtual reality. *Virtual Reality & Intelligent Hardware* 1, 1 (2019), 84–112.

[56] Difeng Yu, Xueshi Lu, Rongkai Shi, Hai-Ning Liang, Tilman Dingler, Eduardo Velloso, and Jorge Goncalves. 2021. Gaze-Supported 3D Object Manipulation in Virtual Reality. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (Yokohama, Japan) *(CHI '21)*. Association for Computing Machinery, New York, NY, USA, Article 734, 13 pages. https://doi.org/10.1145/3411764.3445343

[57] Özgür Yürür, Chi Harold Liu, Zhengguo Sheng, Victor CM Leung, Wilfrido Moreno, and Kin K Leung. 2014. Context-awareness for mobile sensing: A survey and future directions. *IEEE Communications Surveys & Tutorials* 18, 1 (2014), 68–93.

First Submitted October 2023; Revised January 2024; Accepted March 2024