Enhancing Augmented VR Interaction via Egocentric Scene Analysis

YANG TIAN, The Chinese University of Hong Kong
CHI-WING FU*, The Chinese University of Hong Kong, and Key Laboratory of Computer Vision and Virtual Reality Technology, Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences
SHENGDONG ZHAO, National University of Singapore
RUIHUI LI, The Chinese University of Hong Kong
XIAO TANG, The Chinese University of Hong Kong
XIAOWEI HU, The Chinese University of Hong Kong
PHENG-ANN HENG, The Chinese University of Hong Kong

Augmented virtual reality (AVR) takes portions of the physical world into the VR world to enable VR users to access physical objects. State-of-the-art solutions mainly focus on extracting and showing physical objects in the VR world. In this work, we go beyond previous solutions and propose a novel approach to realize AVR. We first analyze the physical environment in the user's egocentric view through depth sensing and deep learning, then acquire the layout and geometry of the surrounding objects, and further explore their affordances. Based on the above information, we create visual guidance (*hollowed guiding path*) and hybrid user interfaces (*augmented physical notepad*, *LR finger slider*, and *LRRL finger slider*) to augment the AVR interaction. Empirical evaluations showed that the participants responded positively to our AVR techniques.

CCS Concepts: • **Human-centered computing → Virtual reality**; **Interactive systems and tools**; *Human computer interaction (HCI)*; *Interaction paradigms*;

Additional Key Words and Phrases: virtual reality, augmented VR interaction, egocentric view, visual guidance, visual tool, depth sensing, scene analysis, deep learning

ACM Reference Format:
Yang Tian, Chi-Wing Fu, Shengdong Zhao, Ruihui Li, Xiao Tang, Xiaowei Hu, and Pheng-Ann Heng. 2019. Enhancing Augmented VR Interaction via Egocentric Scene Analysis. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 3, 3, Article 105 (September 2019), 24 pages. https://doi.org/10.1145/3351263

1 INTRODUCTION

When immersed in the VR world, users cannot easily access physical objects as they normally do because of the visual disconnection from the physical world. Besides the brute-force approach of taking off the headset, users

*corresponding author

Authors' addresses: Yang Tian, ytiancuhk@gmail.com, The Chinese University of Hong Kong; Chi-Wing Fu, cwfu@cse.cuhk.edu.hk, The Chinese University of Hong Kong, and Key Laboratory of Computer Vision and Virtual Reality Technology, Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences; Shengdong Zhao, zhaosd@comp.nus.edu.sg, National University of Singapore; Ruihui Li, RuihuiLi.Lee@gmail.com, The Chinese University of Hong Kong; Xiao Tang, xtang@cse.cuhk.edu.hk, The Chinese University of Hong Kong; Xiaowei Hu, xwhu@cse.cuhk.edu.hk, The Chinese University of Hong Kong; Pheng-Ann Heng, pheng@cse.cuhk.edu.hk, The Chinese University of Hong Kong.

2474-9567/2019/9-ART105 $15.00
https://doi.org/10.1145/3351263

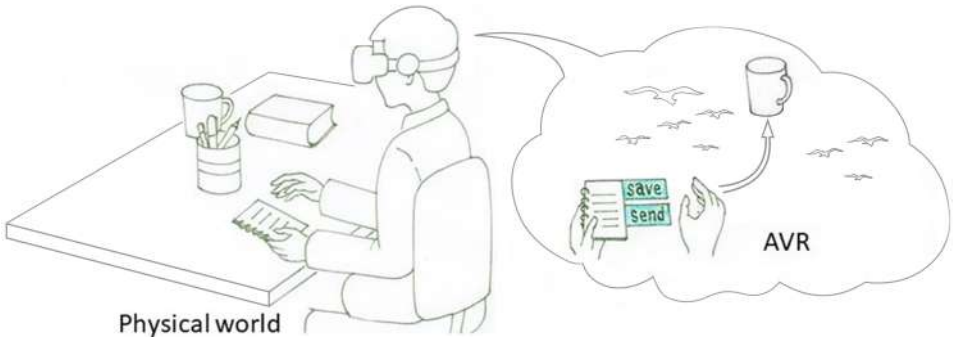

Fig. 1. In our approach, relevant physical objects (e.g., a cup and a notepad) are automatically detected and incorporated into the VR world. Furthermore, a design of visual guidance (the guiding path in front of the user's hand) helps the user conveniently interact with the physical cup without severely breaking the immersive experience. Virtual buttons along an edge of the notepad enrich its function in the AVR environment. See the implementations of the visual guidance and the *augmented physical notepad* in Figures 6 and 9.

may use a button on the headset to teleport between the physical world and the VR world. To enable the users to see physical elements while immersed in the VR world, augmented virtual reality (AVR) augments the virtual reality by overlaying portions of the physical world on top of the VR world.

Several existing solutions [3, 4, 22, 24, 25, 35] have been proposed to realize AVR. The solutions mainly focused on selectively extracting and displaying physical objects as overlays. However, they did not analyze the layout of the physical environment in the user's egocentric view or make use of the geometry of the objects in the physical environment. Both the layout and geometry are beneficial to enhance/enrich the interaction in the AVR environment.

Motivated by the recent success in real-time object detection (e.g., YOLO [29]) and scene understanding (e.g., SemanticFusion [21]) through deep learning and depth sensing, our vision in this work is to leverage these techniques to understand the physical environment around the VR user to acquire the information including the layout of the physical environment and the geometry of the physical objects in it. Then we apply the layout information and the affordances of the physical objects based on their geometry information to enhance/enrich the AVR interaction. An affordance determines how an object could possibly be used [27].

In this paper, we present a new approach to realize augmented virtual reality (AVR). Compared with previous AVR solutions, our approach is able to extract the context information (layout and affordance) semantically. Based on the context information, *our approach not only enhances the AVR users' experience via visual guidance when the users access and move the physical objects, but also creates hybrid user interfaces, which combine physical objects and virtual elements to enhance or enrich the AVR interaction;* see Figure 1.

Under our new approach, we have the following contributions:

- a prototype system, which is able to acquire the above context information in real-time;
- *hollowed guiding path*, a visual guidance design, to help a VR user conveniently interact with a target physical object without severely breaking the immersive experience, especially when the user is immersed in virtual environments with a lot of dynamic contents;
- *augmented physical notepad*, which is augmented from a physical notepad with virtual buttons, to enable the VR users to not only write as they do in the real world, but also enjoy novel interactions via the buttons;
- *LR finger slider*, a physical slider, which is augmented from a physical desk rim with a slider widget, to facilitate the VR users to comfortably adjust parameters with haptic feedback; and

- *LRRL finger slider*, which is also augmented from a physical desk rim, to facilitate the VR users to comfortably adjust parameters with haptic feedback, but with doubled accuracy.

2 RELATED WORK

In this section, we review works on AVR techniques, their applications, and some related topics.

2.1 AVR Techniques

To realize AVR, early works put an RGB camera on the front side of the VR headset, extracted physical objects in the camera view using some simple video-keying techniques, and then overlaid the result on top of the VR world. A pioneer work was done by Metzger et al. [24] who used a luminance keyer to segment the physical objects out of a uniformly-lit background. Bruder et al. [2] used a skin detection method to segment users' hands, adjusted their luminance and overlaid them above the VR world. Later, Bruder et al. [3] applied a chroma keyer to segment objects against a uniformly-painted background. Hence, both the user's hands and physical tools could be seen in their virtual architectural exploration application. Clearly, video-keying solutions are relatively easy to implement for supporting AVR. However, the solutions cannot be applied to general environments in practice.

Later, the research community started to explore the use of consumer-grade depth sensors attached on the front side of VR headsets for supporting AVR. Suma et al. [36] enabled a VR user to see a physical person in front by analyzing the depth image and extracting 3D points associated with the person. Tecchia et al. [37] reconstructed and showed the 3D meshes of the user's hands in the VR world. In addition, they attached colored rings on the user's fingers for tracking them. Besides, Nahon et al. [26] used a fixed Kinect sensor* exterior to the VR user to capture 3D point clouds of the physical world and to enable the user to see physical elements.

Besides the techniques to segment objects and human bodies, McGill et al. [22] proposed an engagement-dependent method, which selectively incorporates relevant portions (objects and persons) of the physical world based on whether the user engages with them or not. Budhiraja et al. [4] found that showing the extracted hands and objects as well as edges detected in the RGB camera view was favored by most users. Though these works enable VR users to access physical objects, they mainly focus on how to select and display physical objects in the VR world. Through depth sensing and deep learning, our approach goes beyond the previous works to analyze the surrounding physical environment around the user to extract the layout information of the surrounding physical environment and the geometry information of the physical objects in it. Enabled by the layout information and the affordances of the physical objects based on their geometry, we can create visual guidance and hybrid user interfaces to enhance or enrich the AVR interaction; see results in Sections 4, 5, and 6. We are not aware of any previous works on AVR that explore egocentric physical scene analysis and provide visual guidance and hybrid user interfaces.

2.2 AVR Applications

AVR techniques have been applied in various areas such as remote collaboration, 360 video viewing, inputs in VR, etc. Gao et al. [10] developed a remote collaboration system that renders the point clouds of the local worker's surrounding physical environment in his/her egocentric view together with the remote helper's hands on both the local and remote sides. Thus, it facilitates the remote helper to understand the spatial relationships on the local side. The local worker is well guided by the helper's hands during the physical tasks. Later, they further reconstructed the local worker's workspace before a task starts to enable the remote helper to freely explore the local workspace from a free viewpoint [11]. To support long-term use, they replaced the heavy VR headsets tethered to PCs with a lightweight self-contained headset on the remote side and an Android-based smartphone or AR display on the local side [9]. Lee et al. [16] proposed a conceptual design of applying the AVR

*Microsoft's Kinect sensor: https://en.wikipedia.org/wiki/Kinect .

techniques to 360° panorama movies. They emphasized the user embodiment, transitions between the real and virtual spaces, and interactivity in this design. Also, they implemented a proof-of-concept system that blends the VR user's hands over the movie scene and enhances the transition between the real and virtual spaces via a head shaking gesture. Khan et al. [14] incorporated the VR user's hands tracked by a Leap Motion sensor into the VR space, where the users could watch the 360 videos while interacting with the virtual objects relevant to the videos. There were several works on how to facilitate VR users to effectively provide inputs by incorporating keyboards [18, 39] and smartphones [7, 15] in the VR world.

2.3 Mapping the Physical and Virtual Spaces

Instead of incorporating the physical elements with their original appearance into the VR space, some works build mappings between the physical elements and their virtual counterparts. Simeone et al. [32] explored how much the designers, who substitute the physical world with virtual counterparts, can magnify the mismatch between the physical objects and their virtual proxies before breaking the VR illusion. They created a multi-layer model of substitution, which demonstrates the increasing levels of mismatch between the physical objects and their virtual proxies. Sra et al. [34] created interactive VR environments by capturing indoor scenes in 3D, detecting obstacles, and determining the walkable areas for users. Then the VR users could freely walk in a physical space where the physical objects were replaced by virtual counterparts. Estrada et al. [12] proposed to allow VR users to substitute the physical objects with the help of a recommender system to enhance the VR interaction experience. Azmandian et al. [1] developed a passive haptics repurposing framework, with which a single physical prop provides passive haptics for multiple virtual objects. Using this framework, they introduced three approaches for haptic retargeting: (i) manipulating the virtual representation of the person's body; (ii) manipulating the virtual world coordinate system; and (iii) manipulating both of them. Cheng et al. [5] made users to reconfigure and actuate passive props to create constantly-varying virtual worlds or dynamic haptic effects. They presented two passive props in this work: (i) a foldable board, which can be reconfigured by the user to match various virtual objects in the VR space, e.g., suitcase, fuse box, railing, etc.; and (ii) a volleyball suspended from the ceiling, which can be animated to produce lively tactile feedback.

2.4 Visual Tools in AR/VR

Next, we review AR/VR visual tools that are relevant to our hybrid user interfaces. Works [6, 28] explored notepads in VR. Poupyrev et al. [28] used a spatially-tracked pressure-sensitive tablet and a physical pen to support various interactions in VR, e.g., taking/modifying notes, flipping/tearing pages, etc. Clergeaud et al. [6] extended the notepad in the work of Poupyrev et al. [28] by attaching the physical notepad to the ceiling with a rope and a pulley. While these works allow the VR users to take notes, they do not naturally make use of the affordance of a physical notepad for determining a 3D interactive plane in the virtual environment. With our *augmented physical notepad*, a VR user can not only take notes as they do in the physical world but also interact with the virtual buttons added along a notepad edge. Mendes et al. [23] proposed four mid-air and one multi-touch based approaches to manipulate 3D virtual objects for stereoscopic interactive desks, where the user can manipulate a virtual object by using one finger to touch it on the interactive desk and another finger to move on the desk along a direction indicated by a widget. Sousa et al. [33] used a multitouch frame on a regular desk to let the VR user perform a slider-like up and down (or left and right) movement in the frame to control the visual brightness and the volume slicing. Compared with these works, our *finger slider*s, augmented from a desk rim with slider widget(s), facilitate the VR users to adjust parameters comfortably with tactile feedback and two accuracy modes.

Fig. 2. Our hardware system setup. Note the transformations (M's) among various components in the system.

3 PROTOTYPE SYSTEM

3.1 Hardware Setup

Figure 2 shows our hardware setup, which consists of (i) an HTC Vive VR headset[†] (with a 612×460 RGB camera on its front side); (ii) the HTC Vive Lighthouse base stations for tracking the headset (tracking area: $4m \times 3m$); and (iii) an Intel RealSense SR300 depth sensor[‡] (with a 640×480 depth camera and a 1920×1080 RGB camera), which is rigidly mounted on the front side of the headset using a vehicle-mounted cellphone holder. We denote the 3D transformations between the coordinate systems of various components as a family of abbreviations started with M, i.e., M_{zcam2rgb}, M_{rgb2fm}, M_{fm2hrgb}, and $M_{\text{hrgb2hcen}}$; see Figure 2. In addition, our computer provides the computing ability with an Intel Xeon E5-1630 v3 CPU and an NVIDIA Titan X GPU.

3.2 Software Setup

Our system is built upon the following SDKs or software tools.

- The OpenVR SDK [31] for integrating the HTC Vive system into our prototype system.
- The Intel RealSense SDK [30] for acquiring the color and depth images captured from the depth sensor, mapping the pixels in the camera view to obtain a colored 3D point cloud, etc.
- The ARUCO marker module [19] for estimating the poses of the fiducial marker relative to the RGB cameras (rgb and hrgb in Figure 2) during the offline calibration.
- The PCL library [17] for point cloud processing.
- Tiny YOLO [29] for real-time object detection in the user's view.
- The iFLYTEK voice recognition SDK [13] for the user to voice out the intention of accessing physical objects.

3.3 Offline Preprocessing

There are two main tasks in the offline preprocessing:

Task 1: find the 3D transformation from depth camera to VR headset (denoted as $M_{zcam2hcen}$). Using the Lighthouse base stations, we can obtain the location and orientation of the VR headset relative to the two stations in the physical space. Hence, we can track in real-time the user's head pose in the VR space (virtual world coordinate system), where the headset center in the VR space corresponds to the center point of the headset's front side (denoted by hcen); see Figure 2 (right). On the other hand, the depth camera (denoted by zcam) on the headset acquires 3D points in front of the user. Since the 3D point coordinates are relative to zcam, we have to transform

[†]HTC Vive: https://www.vive.com/us/product/vive-virtual-reality-system/ .
[‡]Intel RealSense SR300 depth sensor: https://software.intel.com/en-us/realsense/sr300 .

them from the zcam space to the VR space, so that we can correctly render the points in the user's egocentric view in VR.

- First, we lookup the transformation from zcam to rgb, i.e., M_{zcam2rgb}, which is a constant matrix, from the Intel RealSense SDK; see Figure 2 (right).
- Second, we put a fiducial marker (denoted by fm) in front of the headset (see Figure 2 (left)) to find the transformation from rgb (the RGB camera on the depth sensor) to fm (i.e., M_{rgb2fm}) and also the transformation from hrgb (the RGB camera on the VR headset) to fm (i.e., M_{hrgb2fm}). Obtaining a series of M_{rgb2fm} and M_{hrgb2fm} over time, we can then fit and determine

$$M_{\text{rgb2hrgb}} \;=\; M_{\text{rgb2fm}} \times M^{-1}_{\text{hrgb2fm}} \,. \tag{1}$$

- Third, we lookup the transformation from hrgb to hcen, i.e., $M_{\text{hrgb2hcen}}$, which is a constant matrix, from the OpenVR SDK; see Figure 2 (right).
- Lastly, by using the above matrices, we can find the transformation from zcam to hcen:

$$M_{\text{zcam2hcen}} \;=\; M_{\text{zcam2rgb}} \times M_{\text{rgb2hrgb}} \times M_{\text{hrgb2hcen}} \,. \tag{2}$$

Task 2: prepare a deep network. We prepared a deep neural network based on Tiny YOLO [29] for object detection and localization in the RGB images in the user's view. In detail, we trained the Tiny YOLO network on datasets collected and labeled by ourselves. For the *hollowed guiding path* in Section 4, we collected and labeled 1,470 images of an office desk scene. The object labels in our dataset included cup, mouse, tea box, keyboard, hand, cellphone, etc. For the *augmented physical notepad* in Section 5, we collected 825 images with labels on pen tip and notepad. For the *finger sliders* in Section 6, we collected 929 images of index fingers. The images were all captured by the RGB camera (rgb) on the depth sensor in the user's view. The training took around two days on an NVidia Titan X GPU.

3.4 Online Egocentric Scene Analysis

Apart from the offline preprocessing, we need to analyze the egocentric scene to extract the surrounding context information. Such an analysis has to complete in real-time for supporting the AVR interaction, so we have to keep performance in mind when designing the analysis process. Below are the key steps in our analysis:

- First, we use the trained deep network to recognize objects and find the object label and 2D bounding box associated with each object in the RGB image of the user's view. The process takes only ~5ms to complete, so it is sufficient to support interactive performance.
- Second, for each recognized object, we use the detected 2D bounding box to segment out the object on both the RGB image and depth image; see the top row of Figure 3 (left and middle).
- Third, we transform the 3D points captured by the zcam to the virtual world coordinate system by using $M_{\text{zcam2hcen}} \times M_{\text{hcen2VR}}$, where M_{hcen2VR} is the transformation from hcen to the virtual world coordinate system; and M_{hcen2VR} can be looked up from the OpenVR SDK. Right now, the resulting point cloud may include some outliers, e.g., points on background; for real-time performance, we simply remove irrelevant points that are too far from the user by distance thresholding.
- Fourth, we try to fit horizontal planes for desks in front of the user. For real-time performance, we downsample the input (dense) point cloud from 640×480 to 106×80, compute a histogram on the vertical components of the points, and fit horizontal planes over the points using the RANSAC algorithm [§]; see the bottom row of Figure 3 for a running example. Further, we use the detected planes to refine the point cloud of each object by filtering out points on the desks; see the top row of Figure 3 (right). After the refinement, we compute the point cloud

[§]RANSAC: https://en.wikipedia.org/wiki/Random_sample_consensus .

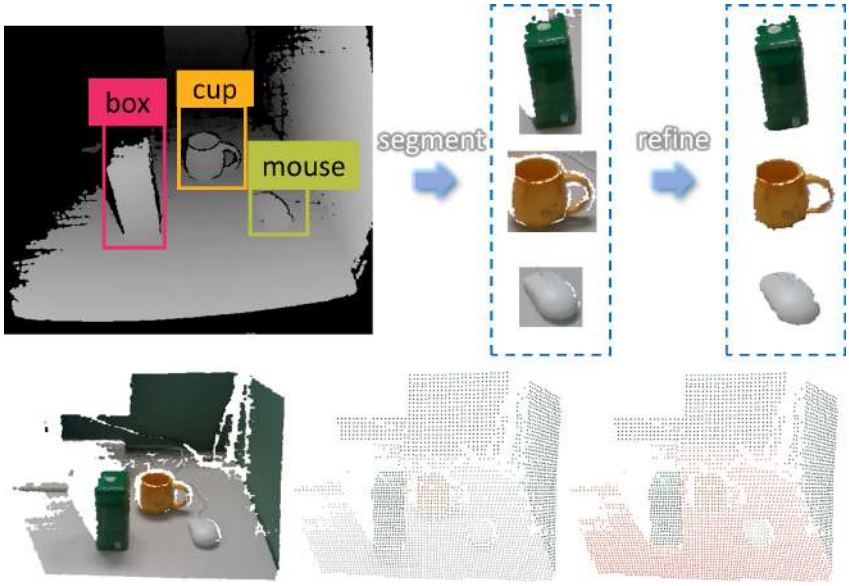

Fig. 3. Top row: the framework of our online egocentric scene analysis. Bottom row: a running example for fitting a desk plane (marked by red color).

centroid for each recognized object and take it as the 3D object center in the VR space. This step completes in only ~3ms.

- In addition, to support the *augmented physical notepad*, we further extract the upper and right borders on the recognized notepad (see Figure 10 (a)) and employ such information to augment the notepad with virtual buttons in the VR view. For the *finger slider*s, we further extract the rim of the detected desk (see Figure 12) for aligning the interactive sliders. Details will be given in Sections 5 and 6.

The capturing latency of the Intel RealSense SR300 depth sensor was found to be ~100ms with the help of the latency tool in the librealsense SDK [38]. The latency for processing the egocentric scenes of the visual guidance, *augmented physical notepad*, and *finger slider* usage scenarios is ~20ms, ~19ms, and ~17ms, respectively. Note also that to efficiently render each recognized object in the virtual world, we form an image-based mesh for each object and use OpenGL shaders to render the objects on the GPU. The latency for rendering the AVR environments is ~11ms. The latency of the HTC Vive headset is ~11ms (90Hz). Overall, the latency of the prototype system is ~141ms. To facilitate the users to trigger the system to incorporate physical elements into the VR space in the usage scenario of the visual guidance, we developed a voice recognition module based on the iFLYTEK SDK [13] to record the user's voice input from the microphone and translate it into a list of words. If the word list contains the name of any labeled object in our deep learning database, our system will regard this object as the one with which the user wants to interact.

4 VISUAL GUIDANCE

Adding physical overlays in the VR environment enables the AVR user to interact with the physical objects (physical interaction), and in general, the more the physical context the better we should enhance the performance of the physical interaction. However, the physical overlays would inevitably break the immersive experience to some extent. Thus, we want to figure out "Is there a method for incorporating physical objects that provides convenient physical interaction without severely breaking the VR immersion?" Particularly, we hypothesize that

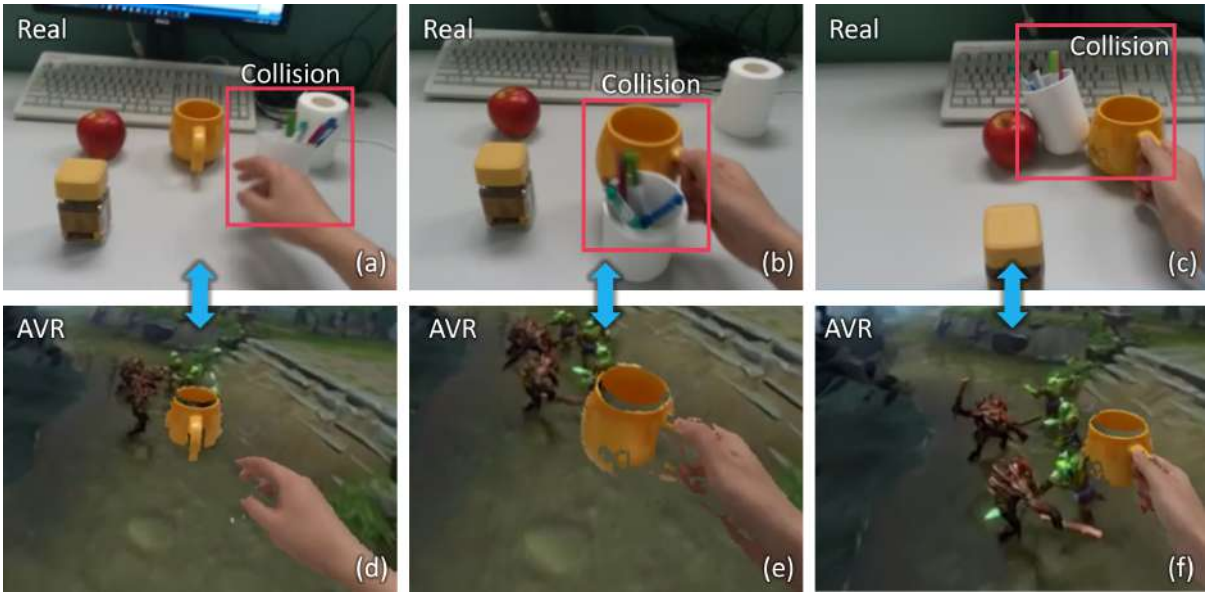

Fig. 4. AVR environments under the *Target-Hand* (*TH*) incorporation method (i.e., only incorporating the target physical object and the user's hand into the virtual environment). Although the immersive experience is relatively good, the interaction with the physical object(s) prones to collisions with the unintended objects.

balancing the physical interaction convenience and the VR immersion is more favored by users immersed in virtual environments with a lot of dynamic contents (e.g., a VR movie with rich plots), since these users may easily miss the dynamic plots if there are excessive physical overlays, while the users immersed in the virtual environments with little dynamic contents may not experience such a problem.

With the above question and hypothesis, we started our exploration with a pilot study.

4.1 Pilot Study for Design Exploration

The purpose of this pilot study is to explore how to design a physical object incorporation method that balances the physical interaction convenience and the VR immersion. We evaluated four methods for incorporating physical overlays into the virtual environment in different levels: (i) *Target-Hand* (*TH*): the target object and the user's hand; see Figure 4; (ii) *Around-Hand* (*AH*): the physical area around the user's hand; (iii) *Partial*: all possible interactive physical objects in the user's egocentric view and the user's hand (see Figure 5); and (iv) *Full*: the full reality in the user's egocentric view. Among the four methods, (ii) (iii) (iv) are proposed by McGill et al. [22]. We also considered two types of virtual environments: (i) *static*: a static virtual environment, which is a skybox with a scenery cubemap texture; and (ii) *dynamic*: a virtual environment, which is a 360 video containing rich plots.

We recruited five participants from the campus aged between 22 and 26 (two females and three males). Two of them had experience with VR. The participants were asked to sit next to a desk on which there were a target physical object (a cup) and four unintended physical objects; see Figure 5 (a). The procedure was as follows: (i) the participant wore a VR headset and watched a virtual environment for 30 seconds; (ii) he/she made the physical objects appear in the virtual environment by voice command (the voice command module is introduced at the end of Section 3); (iii) the participant grabbed the target object, fetched it close to him/herself, and put it back; and (iv) once the participant put the target object back and near to its original position, the physical overlays

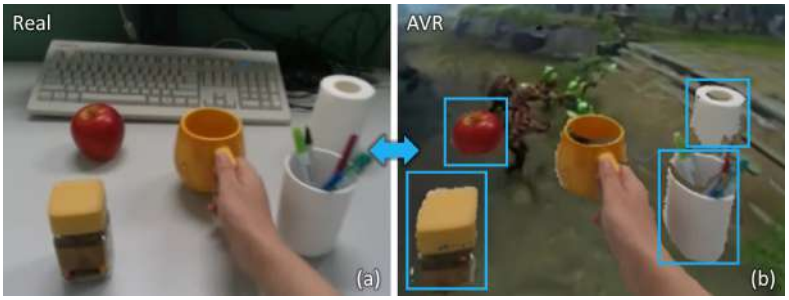

Fig. 5. AVR environment under the *Partial* incorporation method. (a) A real scene. (b) The corresponding AVR scene. Regions marked by the colored boxes are occluded by the unintended objects. The interaction with the target physical object is relatively convenient. However, a large portion of the virtual contents is occluded by the unintended objects.

disappeared. Each participant repeated this procedure eight times to cover all combinations of the incorporation method and the virtual environment type. At last, we asked the participants to comment on their experience. The summary of the participants' comments is as follows:

(i) Participants cared about the occlusion caused by the physical overlays above the *dynamic* virtual environment very much. They commented that balancing the physical interaction convenience and VR immersion was more important in the *dynamic* virtual environments than in the *static* ones. The reason is exactly the missing dynamic plots problem as described in the hypothesis.
(ii) All participants gave negative feedbacks to the *Full* method. They mentioned that the full reality in the user's egocentric view incorporated into the virtual environment destroyed the immersive experience, especially in the *dynamic* virtual environment.
(iii) All participants gave negative feedbacks to the *AH* method. They came up with the issue that they always took too much time to search for the target object because they had no clue about its position. A participant described his experience by "searching the target object in the darkness with a search light."
(iv) All participants thought that the *TH* method brought the best immersive experience among the four methods. However, three of them complained that the collisions happened between the hand/target object and the unintended objects, and they did not know where to put the target object after use.
(v) For the *Partial* method, all participants reported that this method provided them with a relatively convenient way to interact with the physical objects almost as they did in the real world. However, they felt that the physical overlays under the *Partial* method still occluded a large portion of the virtual environment.

Based on participants' comments, we drew the following conclusions: (i) our hypothesis raised at the beginning of this section was verified; thus, we only used a *dynamic* virtual environment in the main experiment; (ii) the *Full* method was not practical because it destroyed the VR immersion; (iii) the *AH* method was also not practical because users did not know the location of the target object; and (iv) both the *TH* and *Partial* methods had pros and cons. Motivated by these conclusions, we finalized our design of the incorporation method by trying to adopt the advantages of both the *TH* method (relatively good VR immersion) and the *Partial* method (relatively convenient physical interaction): based on the *TH* method, we further incorporate a hollowed path which brings in only limited occlusion above the virtual environment. We hoped that this path is able to make up the absence of the unintended objects displayed in the *Partial* method, which actually provides visual guidance for convenient physical interaction. We denote this method as the *Target-Hand-Path* (*THP*) method (see Figure 6).

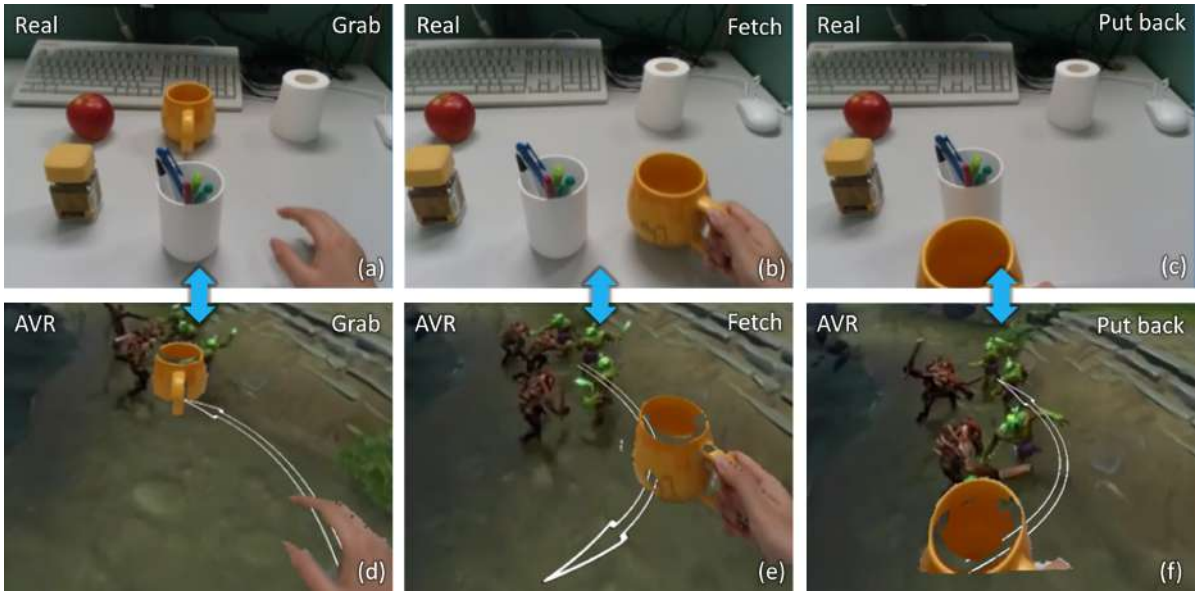

Fig. 6. Illustrating the *Target-Hand-Path* (*THP*) incorporation method (the system only incorporates the target object (the cup), the user's hand and a hollowed guiding path into the virtual environment). The hollowed guiding path helps the AVR user to avoid collision with a physical obstacle located just in front of his/her body during the "grab", "fetch" and "put back" steps. In addition, only the edges of the guiding path bring in very limited occlusion above the virtual contents.

4.2 Main Experiment

We conducted an experiment to explore whether the *THP* method inherits the advantages of both the *TH* and *Partial* methods or not.

Participants. We recruited 21 participants from the campus (9 females and 12 males) aged between 20 and 29. Among them, six participants had experience with VR before. They are all right-handed.

Developing apparatus for the THP method. Immersed in a dynamic virtual environment, the user may issue a voice command to our prototype system, i.e., by saying out the name of the object that he/she wants to interact with. Our system further uses the trained deep network to try to locate the associated object in the user's view and then renders a hollowed path on the desk plane to guide the user to grab the object. Figure 6 illustrates the interaction procedure. To do so, we first estimate a Bezier curve from the right front side of the user to the target object on the desk plane. The first and second control points are calculated according to the positions of the user's head and the unintended object. The third one is the position of the target object. After that, we construct a curved band using the Bezier curve as the centerline. At last, we map a texture of a hollowed path with an arrow end to the band. When the user moves the target object close to him/herself for use or puts it back to the original position, the target object/user's hand may also collide with some unintended objects (see Figure 4 (b)&(c)). Our system thus renders hollowed paths suitable for these situations accordingly using the above method; see Figures 6 (e)&(f).

Task. A trial began as soon as a participant tapped the blue cube (Figure 7 (b)) on his/her right-hand side. After it disappeared, the participant immediately reached out his/her right hand to grab the handle of a physical cup and fetched it to the position of the red cube just in front of him/her (Figure 7 (b)). After the red cube disappeared,

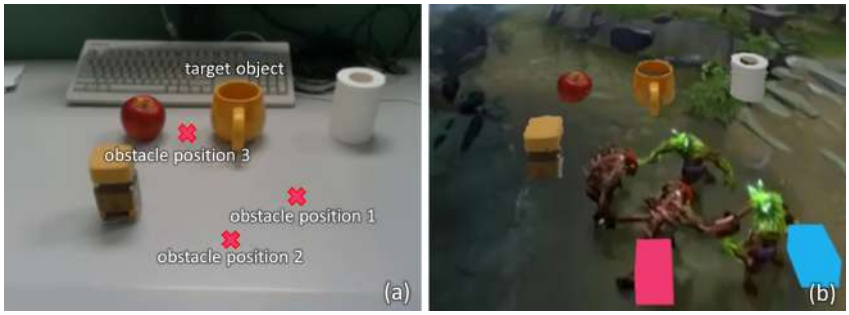

Fig. 7. (a) The physical scene of the experimental setting for the visual guidance main experiment. (b) The egocentric view of a participant who will start a trial.

the participant was required to put the cup back to its original position. If the cup was placed more than 6 cm away from the original position, the participant would be required to repeat the trial.

Experimental design. This experiment investigated the effects of the physical object *incorporation method* (*THP*, *TH* and *Partial*) and the *obstacle position* on the *performance time*, the *number of collisions* which happen during the trials, and the participants' subjective ratings. We designed three levels for the factor of *obstacle position* (marked by the red crosses in Figure 7 (a)): (i) the middle point of the line segment, which links the positions of the blue cube and the initial position of the target object; (ii) a position on the line segment, which links the red cube and the initial position of the target object; and (iii) a position on the left side of the initial position of the target object and just next to it. The initial position of the target object was fixed in the main experiment. Each participant performed the task described previously under the three incorporation methods. The order of the incorporation methods was counter-balanced across the participants using a balanced Latin square design. In order to simulate the real desktop scenarios, we put another three interactive objects at three fixed positions.

For each incorporation method, we used a random ordering of the three levels of the *obstacle position*, so that the participants could not easily anticipate the obstacle position. For each level of the *obstacle position*, participants performed 3 trials. At the start of the task under each new incorporation method, we gave participants a 6 minutes' practice session. Then the participants conducted the formal trials. After finishing the experiment with one incorporation method, participants took a 3 minutes' rest. After the formal trials, we asked participants to rate on three statements (1: fully disagree, 5: fully agree): (i) The immersive experience is good; (ii) It is convenient to interact with the target physical object; and (iii) I prefer this method the most. At last, we asked the participants to comment on their experience in this experiment. We recorded their voice for further analysis. It took approximately 40 minutes to complete the whole experiment for each participant.

Thus our design (excluding practice trials) has a total of:
21 participants ×
3 methods (*Target-Hand-Path (THP), Target-Hand (TH), Partial*) ×
3 obstacle positions ×
3 trials for each obstacle position
= 567 trials.

Parametric dependent variables were the *performance time* and the *number of collisions*. *Performance time* measured the interval between the disappearance of the blue cube and the moment that the participants put the cup back successfully. *Number of collisions* meant the number of collisions that happened between the hand/target object and any unintended objects. It was counted by the experimenter.

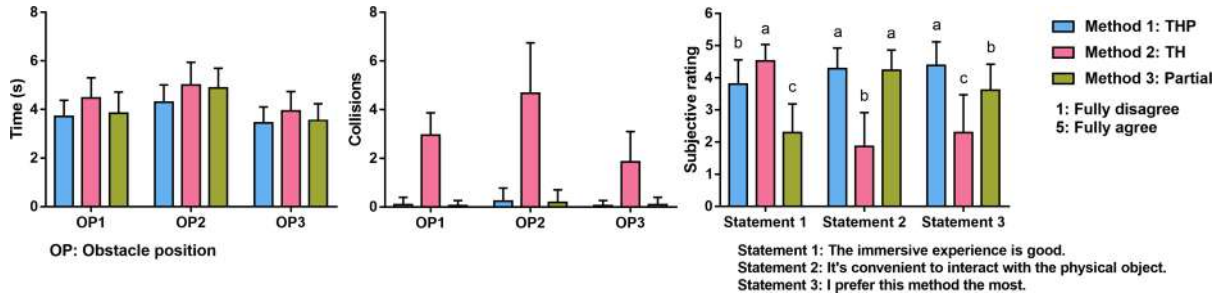

Fig. 8. The result of the visual guidance main experiment. *THP*: *Target-Hand-Path* (see Figure 6); *TH*: *Target-Hand* (see Figure 4); and *Partial* (see Figure 5).

Quantitative Results

Firstly, we did a normality check on the datasets of the *performance time* and *number of collisions* using the Shapiro-Wilk method. The result showed that for every combination of independent variables (*incorporation method* and *obstacle position*), the datasets were normally distributed. Then we performed the two-way repeated-measure ANOVA with post hoc Bonferroni's multiple comparisons test for the datasets. For the participants' ratings on the above statements, we conducted the Friedman tests with post hoc Wilcoxon Signed Rank tests applying Bonferroni correction. Figure 8 shows the means and standard deviations for the dependent variables.

Performance time: The results showed a significant interaction between the *incorporation method* and the *obstacle position* ($F_{4,80} = 3.299$, $p = 0.0148$). Significant difference between different incorporation methods was found ($F_{2,40} = 12.4$, $p < 0.0001$). The *performance time* of the *TH* method was significantly longer than that of the *THP* ($p < 0.0001$) and the *Partial* ($p = 0.0178$) methods. There was no significant difference between the *THP* and *Partial* methods for the *performance time* ($p = 0.1412$). *The results implied that the physical interaction convenience level of the* THP *method was the same as that of the* Partial *method, which provides convenient physical interaction (as reported by the participants in the pilot study). The results also implied that the physical interaction under the* TH *method is not as convenient as that of the other two methods.*

Number of collisions: The results showed a significant interaction between the *incorporation method* and the *obstacle position* ($F_{4,80} = 16.59$, $p < 0.0001$). Significant difference between different incorporation methods was found ($F_{2,40} = 218.7$, $p < 0.0001$). Furthermore, the *number of collisions* of the *TH* method was significantly larger than that of the *THP* ($p < 0.0001$) and the *Partial* ($p < 0.0001$) methods. There was no significant difference between the *THP* and *Partial* methods for the *number of collisions* ($p > 0.9999$). *The results again implied that the physical interaction convenience of both the* THP *and* Partial *methods was better than that of the* TH *method because of the significantly lower collision numbers.*

Immersive experience: The difference between repeated measures on *immersive experience* was found significant ($\chi^2(2) = 35.412$, $p < 0.0001$). Post hoc pairwise comparisons showed that the VR immersion level of the *THP* method was between that of the *TH* ($Z = -2.972$, $p = 0.009$) and the *Partial* ($Z = -3.804$, $p = 0.000429$) methods. The immersive experience of the *TH* method was significantly better than that of the *Partial* method ($Z = -3.971$, $p = 0.000213$).

Interaction convenience: The difference between repeated measures on *interaction convenience* was found significant ($\chi^2(2) = 30.097$, $p < 0.0001$). Furthermore, the interaction convenience level of the *TH* method was significantly worse than that of the *THP* ($Z = -3.792$, $p = 0.000447$) and the *Partial* ($Z = -3.976$, $p = 0.00021$) methods. There was no significant difference between the *THP* and *Partial* methods for the *interaction convenience* ($Z = -0.243$, $p = 1.0$).

Overall preference: The difference between repeated measures on *overall preference* was found significant ($\chi^2(2) = 26.911, p < 0.0001$). Post hoc pairwise comparisons showed that (i) the participants significantly preferred the *THP* method the most against the *TH* ($Z = -3.527, p = 0.001263$) and the *Partial* ($Z = -3.447, p = 0.003$) methods, and (ii) the participants significantly preferred the *Partial* method to the *TH* method ($Z = -2.537$, $p = 0.033$).

Qualitative results

The TH *method.* Participants commented that although the *TH* method provided the best VR immersion among the three methods, it was not acceptable for practical use because it was too inconvenient to provide no information about the physical environment layout at all. Participant 11 (P11) said, "*It's good to see that only my hand and the target object being incorporated. In this way, I can almost fully understand what's going on in the virtual environment. However, I don't know whether there are other objects in front of my hand or not when I move my hand. I feel insecure because my hand may get hurt or I may knock down fragile or valuable things.*" P15 also said, "*To avoid colliding with unintended objects, I have to carefully probe the physical environment slowly.*"

The Partial *method.* Participants stated that although the physical interaction under the *Partial* method was much more convenient than that of the *TH method*, they were still not satisfied with the VR immersion level of this method. P13 said, *"I can conveniently interact with the physical object. But the additional physical objects bring in more occlusion above the virtual environment, which looks very messy and distracting."* P5 also said, "*I missed some key plots in the virtual environment because of the large occlusion of the physical overlays.*"

The THP *method.* Almost all participants (20/21) expressed their preference for the *THP* method. As P20 stated, "*I like the* hollowed guiding path *very much. It effectively helps me avoid collisions with unintended objects. On the other hand, it does not occlude too much portion of the virtual environment. I can still see through it to well understand the plots in the virtual environment.*" P10 stressed, "*Only the edges of the* hollowed guiding path *brings in a little occlusion. I don't think it's a big deal.*" Several participants (9/21) mentioned that the *hollowed guiding path* also reduced their workload. For instance, P3 stated, "*I think that the* hollowed guiding path *is so cool because I just need to follow its lead without thinking.*" In addition, some participants (7/21) mentioned that the *hollowed guiding path* also effectively helped them easily put the target object back to its original position. For example, P2 said, "*I can easily put back the target object by putting it near the arrow tip of the path. Under other methods, even the* Partial *method, I often hesitated to put it back.*"

Visual Guidance Experiment Summary

Both the quantitative and qualitative results consistently answered the question raised at the beginning of this section: the *THP* method was able to provide convenient physical interaction without severely breaking the VR immersion. For physical interaction convenience, there was no statistical difference between the *THP* method and the *Partial method* under which users interact with physical objects almost like they do in the real world. For VR immersion, the statistical analysis on the subjective ratings of the participants showed that the VR immersion level of the *THP* method was better than that of the *Partial* method (which was not satisfying for participants), and worse than that of the *TH* method.

5 AUGMENTED PHYSICAL NOTEPAD

Some existing works [6, 28] have enabled a VR user to take notes with a physical notepad and pen, while the user is immersed in virtual environments, e.g., VR courses, VR tele-conference, VR brainstorming, etc. However, they did not explore the affordance of the physical notepad to determine an interactive 3D plane to enrich the interaction with the notepad. We go beyond the existing works to augment a physical notepad with virtual buttons arranged on the 3D plane extended from the notepad (see Figure 9). With the virtual buttons, the user can

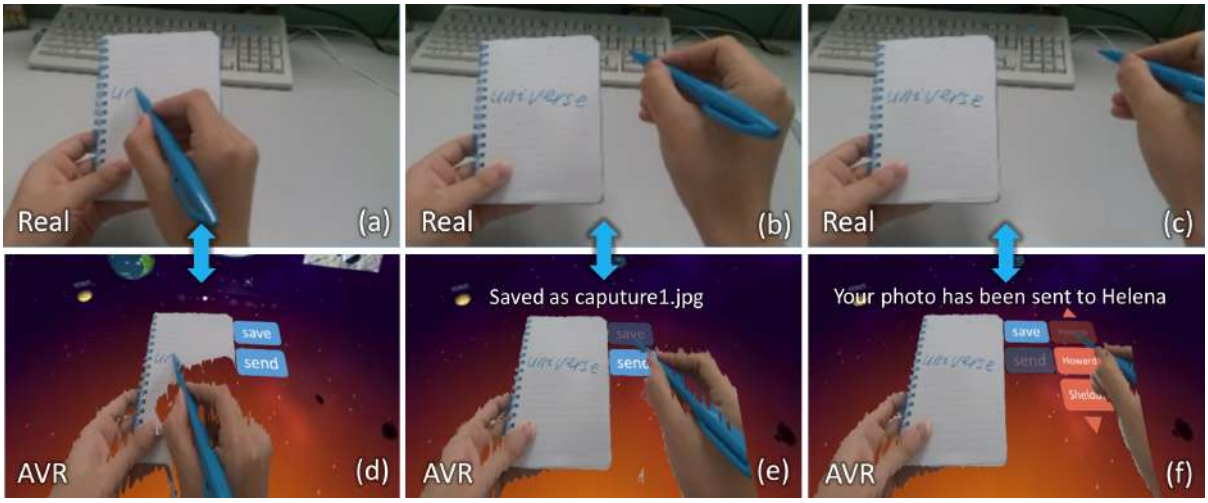

Fig. 9. Illustrating the *augmented physical notepad*. Top row: the user's physical actions captured by the RGB camera of the depth sensor attached on the VR headset. Bottom row: the AVR scenes in the user's egocentric view.

capture the notepad contents and share the contents with others conveniently. We denote such kind of notepad as the "*augmented physical notepad*."

5.1 Implementation

Based on our prototype system introduced in Section 3, we locate the 3D geometry of the notepad plane in the physical world relative to the user, and further locate the notepad plane in the virtual world and construct the virtual buttons attached to the notepad on the extended notepad plane; see Figure 10 (a) for the details. Furthermore, our system tracks the 3D position of the pen tip in the physical world and maps the 3D position to the virtual world. By doing so, it can detect if the pen clicks on any virtual button and perform the user-selected action accordingly, e.g., "save" to capture the notes and the virtual scene and "send" to share the captured contents with others; see Figures 9 (e)&(f). Note that our system provides some visual feedback. It darkens the button being selected by the user. After clicking the "save" button, users can see the captured photo in front of them; see the supplementary video.

5.2 User Study

We evaluated our *augmented physical notepad* by comparing it with an "*augmented virtual notepad*" (See Figure 10 (b)), i.e., both the buttons and the notepad are virtual objects.

Participants. We recruited 16 participants from the campus (8 females and 8 males) aged between 20 and 28. Among them, ten had experience with VR, and all are right-handed.

Constructing the augmented virtual notepad. We render a quadrangle in front of the participant in the virtual environment as the virtual notepad. It floats just above a physical desk for the participant to rest his/her elbows on the desk to write on the virtual notepad using the pinched index and thumb finger tips. We attach a Leap Motion device on the front side of the VR headset for tracking the position of the pinched finger tips. The strokes

Fig. 10. (a) The construction of the virtual buttons on the *augmented physical notepad* (APN). A corner (the green dot) and two edge points (red dots) are detected and mapped into 3D space for notepad plane tracking. (b) A user is taking notes with the *augmented virtual notepad* (AVN). (c) The result of the user study for the *augmented notepad*s.

are rendered according to the moving path of the pinched finger tips. Virtual buttons that correspond to the "save" and "send" functions are arranged in the same way for the *augmented physical notepad*.

Task. Each participant was immersed in a VR course and was required to take notes about the course contents using the *augmented physical notepad* or the *augmented virtual notepad*. After that, he/she was required to save his/her notes and to share it with others using the virtual buttons.

Experimental design. This experiment investigated the effects of the *augmented notepad* designs (*augmented physical notepad* and *augmented virtual notepad*) on the *user preference*. Each participant performed the task described previously using the two *augmented notepad* designs, respectively. The order of the *augmented notepad* designs was counter-balanced across the participants. After the tasks, the participants were required to rate their preference for each *augmented notepad* design using a five-point Likert scale. At last, the experimenter interviewed the participants to comment on their experience in this experiment. The experimenter recorded their voice for further analysis. The participants completed the whole experiment in approximately 25 minutes.

Quantitative Results

We conducted the Wilcoxon Signed Rank test on the dataset of the *user preference*. Figure 10 (c) shows the means and standard deviations of the user preference scores. We found that the participants significantly preferred the *augmented physical notepad* to the *augmented virtual notepad* ($Z = -3.355$, $p = 0.001$).

Qualitative Results

All participants commented that writing on the *augmented physical notepad* was much better than that on the *augmented virtual notepad* because (i) they were used to write with physical notepads; and (ii) it was difficult to write in mid-air. P1 said that "*To write every single stroke with the* augmented virtual notepad *involves three steps: first, pinch my two fingers; second, write; and third, stretch out my fingers. The process is complex and tiring when writing many words. As a result, it is difficult to pay attention to the course and taking notes simultaneously.*" P5 also said, "*I often forgot to stretch out my fingers when I finished one stroke. Thus, I drew a lot of unwanted connected strokes. And my writing on the* augmented virtual notepad *is much more ugly than that on the* augmented physical notepad." P2 said that "*There is no tactile feedback when I write with the* augmented virtual notepad. *It is very difficult for me to keep my finger tips on the virtual notepad.*" However, not everything about the *augmented physical notepad* is perfect. Participants reported a noticeable latency of the prototype system: "*When I wrote with the* augmented physical notepad*, I felt that the strokes were lagging behind my pen tip a little (P3).*" However, the

impact of the latency did not outweigh the difficulty introduced by the *augmented virtual notepad*. As mentioned by P3: "*The latency did not affect my experience too much. I still think that the* augmented physical notepad *is better than the virtual one because of the tactile feedback.*" In addition, P6 and P13 commented that the *augmented virtual notepad* is still useful when there are no physical notepad and pen around the user.

All participants favored the idea of augmenting a notepad with more relevant functions via virtual buttons. P3's comment is representative among participants, "*It is common to see functions like "save" and "send" that are relevant to writing, which can be triggered through buttons on the screen of a tablet. But this time in VR, I can use a daily life notepad which is much cheaper than a tablet to enjoy the same functions. It is definitely a good idea.*" In addition, participants found the layout of virtual buttons well-designed. As mentioned by P16, "I find it very convenient to use the virtual buttons along the right edge of the notepads because after I finish writing, my hand will be almost above the right edge of the notepads."

Augmented Physical Notepad Experiment Summary and Discussion
Participants responded positively to the *augmented physical notepad* as it combines the tactile feedback of the physical objects with the convenience of additional, dynamic digital capabilities based on the physical affordance to provide a better overall user experience. This is achieved regardless of a noticeable drawback of the AVR systems, i.e., the latency of visual feedback mainly caused by the processing time required by the attached sensors (RGB(D) cameras) for capturing the image data and the transmission time from the sensor to the PC via USB cables. This shows that in the testing scenario, the benefit of having the affordance of a physical notepad outweighs the drawback of certain amount of latency. We certainly cannot expect that applying AVR only brings benefits and no drawbacks, but with the advance of technology, we are optimistic that the latency issue will be reduced, and the benefit of being able to leverage the affordances of the physical objects in virtual environments will be further appreciated by the users in the future.

6 FINGER SLIDER

Sliders are commonly-used GUI widgets in traditional PC applications for continuously adjusting parameters. In the VR space, users also need to use sliders in various scenarios, e.g., changing the progress or audio volume of a VR movie, adjusting the properties of a photo in an illumination-controlled VR space, etc. However, state-of-the-art VR systems (e.g., HTC Vive, Oculus Rift, etc.) mainly adopt joysticks to manipulate sliders that float in mid-air in the VR space, which is not convenient for users. The reason is that the users have to probe the surrounding physical environment to grab and use the joystick, or deliberately hold it from the very beginning.

Motivated by the affordance of a desk rim for sliding a finger straightly, we propose to augment a physical desk rim (if available, in front of the user) into a physical slider by aligning a slider widget above the desk rim to facilitate the VR users for continuous adjustment with tactile feedback. Then the user can rest a finger on the desk rim and slide the finger to manipulate the slider widget ("*finger slider*").

6.1 Design Consideration

The design of the *finger slider* needs to take the following factors into consideration:

VR immersion. In order to maximize the VR immersion, the *finger slider* should be visible in the VR space only when it is needed. To make it visible, the user just needs to lift up one of his/her index fingers.

User comfort. The length and position of the slider widget should be carefully determined to be ergonomically accepted by the users. According to the RULA ergonomic evaluation metrics [20], the physical load is small when "the user's lower arm is not working across the midline of the body or out to the side." Thus, the length of the slider widget should be half of the average shoulder width of the users. Given that the average shoulder widths of American women and men are ~36.7 cm and ~41.1 cm, respectively [8], the length of the slider widget should be 0.5×36.7 cm, i.e., 18.35 cm. For right-handed users, the left end of the slider widget should be at the intersection

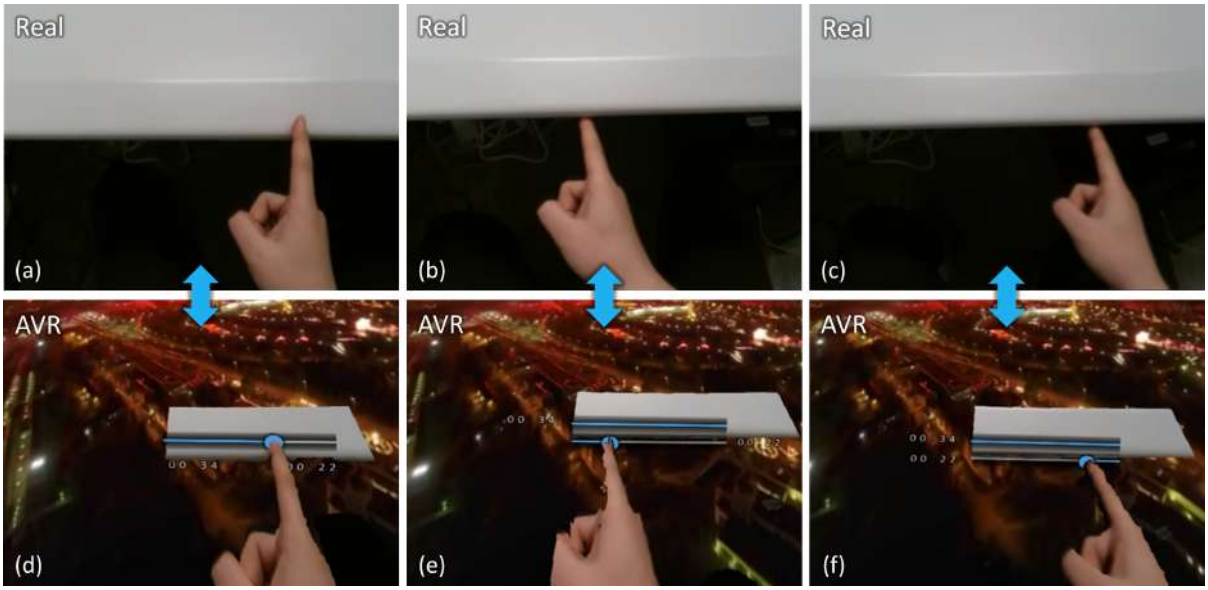

Fig. 11. (a) & (d): The *Left-Right (LR) finger slider*. (b) & (e): The *Left-Right-Left-Right (LRLR) finger slider*. (c) & (f): The *Left-Right-Right-Left (LRRL) finger slider*.

of the desk rim line and the midline of the user's body. For left-handed users, the right end of the slider widget should be at this intersection. In addition, we recommend the users to rest one lateral side of the index finger instead of the index finger belly on the desk rim, because the users would not twist their lower arms in this way (see Figure 11).

Control Accuracy. The length of our slider widget (18.35 cm) may not provide satisfying control accuracy when the parameter range is relatively large. Our vision to relieve this problem is to align another 18.35 cm long slider widget below the desk rim and take both the upper and lower widgets to control the parameter. In this way, the users can still slide their fingers comfortably with tactile feedback, but with doubled slider widget length (36.7 cm), i.e., doubled the control precision. We denote this kind of slider as the "*doubled finger slider*."

Mapping doubled finger slider *and parameter range*. For the upper widget of the *doubled finger slider*, the left and right ends should be mapped to the minimum and middle values of the parameter range, respectively, so that the parameter-increase direction is consistent with the traditional sliders. According to different methods of mapping the lower widget of the *doubled finger slider* and the second half of the parameter range, we denote two *doubled finger slider* designs as: (i) *Left-Right-Left-Right (LRLR) finger slider*: mapping the left and right ends of the lower widget to the middle and maximum values, respectively; and (ii) *Left-Right-Right-Left (LRRL) finger slider*: mapping the right and left ends of the lower widget to the middle and maximum values, respectively.

In addition, we denote the *finger slider* consisting of only the upper widget as the *Left-Right (LR) finger slider*. See the three designs of the *finger slider* in Figure 11.

6.2 User Study

Although the control accuracy of the *doubled finger slider*s (the *LRLR* and *LRRL finger slider*s) is twice of that of the basic *LR finger slider*, we are not sure "whether the lower widget of the *doubled finger slider*s adds too much burden to the user or not." Furthermore, although the parameter-increase directions of the two widgets of

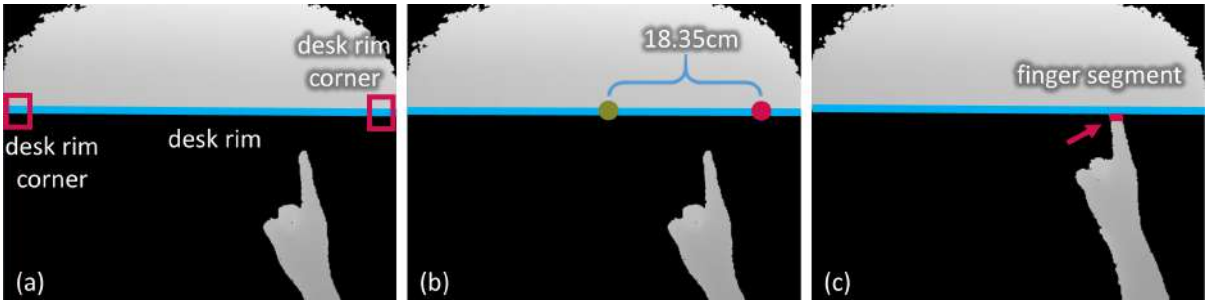

Fig. 12. (a) & (b) Procedure to construct the *finger sliders*. (c) Track the 3D position of an index finger.

the *LRLR finger slider* are consistent with those of the traditional sliders, the *LRLR finger slider* cannot support continuous adjustment around the middle value of the parameter range (i.e., the user has to move the finger between the right and left ends of the upper and lower widgets); although the *LRRL finger slider* enables the user to do that by simply moving the finger to the other side of the desk rim, the parameter-increase direction of the lower widget is contrary to that of the traditional sliders, which may not be accepted by the users. Thus, we are not clear that "which *doubled finger slider* do users prefer?"

With the above two questions, we evaluated the three designs of the *finger slider* (*LR finger slider, LRLR finger slider, and LRRL finger slider*) under different parameter ranges.

Participants. We recruited 15 participants from the campus (7 females and 8 males) aged between 22 and 27. Among them, seven participants had experience with VR. They are all right-handed.

Developing apparatus. First, we use the deep network in our system to detect the user's lifted finger. Second, our system looks for a desk rim in front of the user (if any) and constructs a slider widget in the 3D virtual space. Figures 12 (a)&(b) illustrate the procedure: (i) find the 3D line segment of the desk rim in the user's view (see the blue line in Figure 12 (a)); (ii) compute the line's direction and centroid in 3D; (iii) construct a virtual 3D line segment (18.35 cm long), set its location according to which hand the user lifts up (see Figure 12 (b)), and lift the line segment up above the desk surface (for the *LR finger slider*); for the *doubled finger sliders*, we perform a similar procedure for the lower slider widget. Third, our system checks whether the index finger is in the slider or not. To do so, we examine the depth image captured by the depth sensor and locate the line intersection between the index finger and the desk rim in the user's view. Then, we find a line of pixels out of the desk rim but belonging to the index finger; see the red line segment in Figure 12 (c), and then extract the depth values of these pixels. Next, our system locates the centroid of the finger segment in the 3D virtual space and takes it to control the slider at interactive speed.

Task. A participant was firstly immersed in a 360 video and asked to freely use one of the designs of the *finger slider* (*LR finger slider*, *LRLR finger slider*, or *LRRL finger slider*) to manipulate the progress of this video for six minutes. The manipulation consisted of fast rewinding/forwarding, randomly browsing across the whole video, etc. After that, the participant was immersed in a new 360 video. He/she was required to use the *finger slider* to find a target scene. Then, the participant was immersed in another new 360 video. He/she was again required to find the same target scene as the one in the previous video.

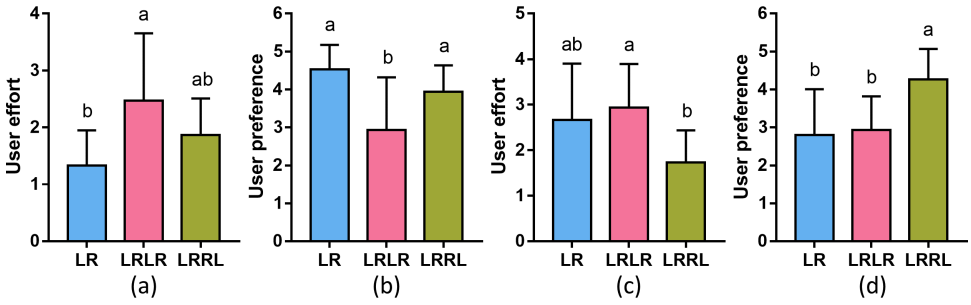

Fig. 13. The experimental results for the *Left-Right (LR) finger slider*, the *Left-Right-Left-Right (LRLR) finger slider*, and the *Left-Right-Right-Left (LRRL) finger slider*. (a) & (b): the results under short parameter range (15 seconds). (c) & (d): the experimental results under long parameter range (135 seconds).

Experimental design. This experiment investigated the effects of the three *finger slider* designs (*LR finger slider*, *LRLR finger slider*, or *LRRL finger slider*) and the *parameter range* on the *user effort* and the *user preference* for the *finger slider* designs. Each participant performed the task described previously using the three *finger slider* designs, respectively. The order of the *finger slider* designs was counter-balanced across the participants using a balanced Latin square design. The time durations (parameter ranges) of the two videos consisting the target scene were 15 seconds and 135 seconds, respectively. The target scene lasted for around two seconds. The order of the two videos was also counter-balanced across the participants. After the tasks, the participants were required to rate their efforts to complete the tasks and their preferences on the *finger slider*s using a five-point Likert scale. At last, the experimenter asked the participants to comment on their experience in this experiment. The experimenter recorded their voice for further analysis. Participants completed the whole experiment in approximately 25 minutes.

Quantitative results.
We conducted Friedman tests with post hoc Wilcoxon Signed Rank tests on datasets of *user effort* and *user preference* under small and large parameter ranges (15 seconds or 135 seconds), respectively. We applied Bonferroni correction in post hoc pairwise comparisons. Figure 13 shows the means and standard deviations of the results.

User effort under small parameter range. The difference between repeated measures on the *user effort* was found significant ($\chi^2(2) = 12.047$, $p = 0.002$). The user effort of the *LR finger slider* was significantly smaller than that of the *LRLR finger slider* ($Z = -2.709$, $p = 0.021$). There was marginally significant difference between the user effort of the *LR* and *LRRL finger slider*s ($Z = -2.309$, $p = 0.063$). There was also marginally significant difference between the user effort of the *LRLR* and *LRRL finger slider*s ($Z = -2.165$, $p = 0.09$). *The results implied that the lower widget of the* LRLR finger slider *with the left-to-right parameter-increase direction might add too much burden to the participants.*

User preference under small parameter range. The difference between the repeated measures on *user preference* was found significant ($\chi^2(2) = 18.304$, $p < 0.001$). Participants significantly preferred the *LR finger slider* against the *LRLR finger slider* ($Z = -3.093$, $p = 0.006$). There was marginally significant difference between the user preference of the *LR* and *LRRL finger slider*s ($Z = -2.324$, $p = 0.06$). For the two *doubled finger slider*s, participants significantly preferred the *LRRL finger slider* to the *LRLR finger slider* ($Z = -2.724$, $p = 0.018$). *The results indicated that, for a relatively small parameter range, (i) the control accuracy of the* LR finger slider *might be enough for the participants; and (ii) the* doubled finger slider*s might not be necessary in this situation.*

User effort under large parameter range. The difference between repeated measures on *user effort* was found significant ($\chi^2(2) = 7.704$, $p = 0.021$). The user effort of the *LRRL finger slider* was significantly smaller

than that of the *LRLR finger slider* ($Z = -2.946$, $p = 0.009$). There was no significant difference between the user effort of the *LRRL* and *LR finger slider*s ($Z = -2.019$, $p = 0.129$). There was also no significant difference between the user effort of the *LRLR* and *LR finger slider*s ($Z = -0.618$, $p = 1.0$). *The results suggested again that the lower widget of the* LRLR finger slider *with the left-to-right parameter-increase direction might add too much burden to the participants.*

User preference under large parameter range. The difference between the repeated measures on the *user preference* was found significant ($\chi^2(2) = 15.346$, $p < 0.001$). Participants significantly preferred the *LRRL finger slider* against the *LR* ($Z = -2.654$, $p = 0.024$) and the *LRLR* ($Z = -3.126$, $p = 0.006$) *finger slider*s. There was no significant difference between the user preference of the *LRLR* and the *LR finger slider*s ($Z = -0.576$, $p = 1.0$). *The results implied that, for a relatively large parameter range, (i) the control accuracy of the* LR finger slider *might become not enough; and (ii) participants might accept the right-to-left parameter-increase direction of the lower widget of the* LRRL finger slider.

Qualitative Results

Small parameter range. All participants preferred the *LR finger slider* the most because it was simple, intuitive, and accurate enough when they looked for the target scene in the short video. P1 said that "*For the short video, the control accuracy of the* LR finger slider *is enough. It is not necessary to use two slider widgets to control the progress.*" Similarly, P2 also said that "*For the short video, I can slide my finger across relatively short distance to achieve my goal well. Why bother using more complex interfaces?*"

Large parameter range. For finding the target scene in the long video, almost all participants (14/15) preferred the *LRRL finger slider* against the other two *finger slider*s. There were two main reasons: (i) when the proportion of the target parameter range to the total parameter range was relatively low, the participants liked the enhanced control accuracy of the *doubled finger slider*s; and (ii) compared with the *LRLR finger slider*, controlling with the *LRRL finger slider* was continuous and fluent. P6 commented that "*With the* LR finger slider*, I skipped the frames of the target scene easily, while the* LRRL *and* LRLR finger slider*s provided much better accuracy. Controlling with* LRRL finger slider *is continuous because I just need to slide my finger along two sides of the desk rim clockwise or anticlockwise. I hate the* LRLR finger slider *because I have to move my finger across the length of a slider widget to use another widget when I need to adjust the progress around the middle of the whole video.*" P9 also said, "*With the* LRRL finger slider*, I can keep my finger on the two sides of the desk rim all the way, while with the* LRLR finger slider*, I have to move my finger away from the desk rim when I need to switch to another slider widget.*" Only one participant insisted that she still preferred the *LR finger slider* because she liked the traditional controlling way. When asked about whether they accept the right-to-left parameter-increase direction of the lower widget of the *LRRL finger slider* or not, almost all participants (14/15) accepted. For example, P12 said that "*In nature, the controlling way of the* LRRL finger slider *is sliding a finger clockwise or anticlockwise on the two sides of the desk rim. I certainly accept this way because we are used to many clockwise or anticlockwise operations in daily life.*" P14 also noticed some additional benefits of the *LRRL finger slider*, "*After I quickly browse the whole video, I can browse it again from the beginning very conveniently because I just need to move my finger from the bottom side to the upper side of the desk rim. I can also conveniently access the other half of the video randomly by moving my finger to the other side of the desk rim.*" P12 suggested that "*I think that you should make the users easily know how the* LRRL finger slider *works, or they don't know they should slide the finger clockwise or anticlockwise.*"

Finger Slider Experiment Summary

The purpose of this experiment is not to investigate whether or not the physical affordance can help with digital interaction (this has been proven by the *augmented physical notepad* experiment), but rather, how to design physcially augmented digital widgets. The results indicated that the design of the *finger slider* depends on the range of values in which the slider needs to control: if the parameter range was relatively small (15 seconds video

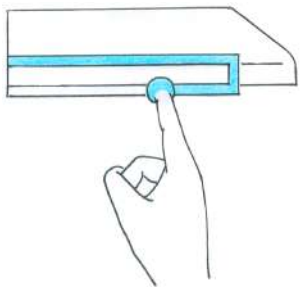

Fig. 14. To make the *Left-Right-Right-Left* (*LRRL*) *finger slider* more intuitive to use, we will add a short vertical slider widget to join the two right ends of the upper and lower widgets.

clip), the participants preferred the *LR finger slider* the most; and if the parameter range was relatively large (135 seconds video clip), the participants preferred the *LRRL finger slider* the most.

Although we tried to make the *LRRL finger slider* intuitive to use by rendering the elapsed and remaining times on the left side of the two left ends of the widgets (see Figure 11 (f)), a participant (P12) still thought that it is not enough. We plan to solve this issue by adding a short vertical slider widget that joins the two right ends of the two widgets. See Figure 14.

7 OVERALL DISCUSSION

The goal of our experiments is to explore how the semantic context information of the physical objects surrounding the VR users can augment the AVR interaction. The participants between the different experiments were partially the same. Among all the participants (37 participants) involved in the experiments, two of them were recruited in all the three experiments; three of them were recruited in both the visual guidance and *finger slider* experiments; one of them was recruited in both the visual guidance and *augmented physical notepad* experiments; seven of them were recruited in both the *finger slider* and *augmented physical notepad* experiments; and twenty-four participants were only recruited in one of the three experiments. The visual guidance experiment demonstrated that a visual guidance based on the layout information was able to balance the physical interaction convenience and the VR immersion. The *augmented physical notepad* experiment showed that the physical affordance helped to create a hybrid user interface that outperformed a fully virtual one. The *finger slider* experiment told us that digital widgets could be physically augmented based on the physical affordance to provide better parameter adjustment experience in AVR. From the experiments, we observed some common insights about how we should make use of the physical affordance and object layout to create a better AVR environment in general:

Using physical affordances in AVR. Previous works [3, 4, 6, 22, 24, 25, 28, 35] only provide basic interactions with the incorporated objects, e.g., using the notepad only for writing in AVR. Physical affordances in this work serve as the medium to reasonably combine daily-life physical objects and virtual elements together closely to create more AVR interaction possibilities (e.g., *augmented physical notepad*) and physically augmented digital widgets (e.g., *finger sliders*). There are various more kinds of physical affordances to explore. For example, cylindrical objects afford rotating them around their centerlines. Thus they can be augmented with virtual scale marks to become knobs in AVR. The two sides of the desk rim also afford encoding two states, e.g., on/off. Users may tap on the upper side to start/resume a function or tap on the bottom side to close/pause the function. We believe that our new approach creates new forms of AVR environment where the virtual reality and the portions of the reality are better mixed.

Extracting geometry properties from depth regions to support physical affordances. To leverage the physical affordances of objects in the AVR environments, it is important that such affordances can be recognized.

We believe that important physical affordance information can be retrieved from 3D geometry properties of physical objects. To extract the 3D geometry properties, we first adopt a neural network to semantically detect the depth regions of the physical objects surrounding the user in the depth images (see the top row (left) in Figure 3). Naively, we can extract geometry properties from the point clouds converted from the depth regions. However, this method is difficult because the point clouds do not contain structured information. Instead, we directly extract the geometry properties of the physical objects from their depth regions and then map the geometry properties into 3D space. For example, three key points on two edges of a notepad are extracted from the depth region of the notepad. We then map the three points into 3D space and derive the 3D directions of the two notepad edges. See Figures 10 (a) for illustration. In addition, another benefit of semantically detecting the depth regions is that we can derive the real-time 3D layout of the egocentric physical environment. With the layout, our work is able to go beyond the previous AVR works (which mainly focus on rendering the physical objects as overlays) to estimate the interaction status and plan suitable visual guidance for the user.

Balancing physical interaction convenience and VR immersion. When leveraging physical affordances, it is important to consider clarity and efficiency. One can incorporate everything in the user's egocentric view to facilitate the users to grab, fetch or put back the target object. However, doing so can reduce immersion as physical overlays can occlude a large portion of the virtual environment, and make it less efficient to interact with the dynamic virtual contents. We believe that it is important to extract only the most essential information from the physical environment so that interaction and immersion can be optimally balanced.

In the visual guidance experiment, we try to approach this optimal balance by proposing the Target-Hand-Path (*THP*) method. The *THP* method is able to "simplify" the AVR user's egocentric physical environment by introducing only minimum information necessary for user interaction: the target object, the user's hand, and a hollowed guiding path. With this simplification, we demonstrated that we can achieve a better balance between the physical interaction convenience and VR immersion. We believe that researchers and designers of AVR applications can benefit from this approach to create better user experience in the future.

8 CONCLUSION AND FUTURE WORK

We go beyond previous methods to propose a new approach to realize augmented virtual reality. By adopting deep learning and depth sensing for egocentric physical scene analysis in real-time, our method is able to semantically understand the context information that includes not only the layout of the physical objects surrounding the user but also their affordances.

Given the layout of the surrounding objects, our approach is able to enhance the AVR user experience when the users access and move physical objects with the help of visual guidance, e.g., the *hollowed guiding path* helps a VR user conveniently interact with a target physical object without severely breaking the immersive experience, especially when the user is immersed in virtual environments with a lot of dynamic contents. With the affordances of the surrounding objects, our approach is able to create hybrid user interfaces to enhance the AVR interaction, e.g., the *augmented physical notepad* enables the VR users to not only write as they do in the real world, but also perform more convenient and novel interactions via the virtual buttons; the *finger slider*s facilitate the VR users to continuously adjust parameters comfortably with haptic feedback and two accuracy modes.

We plan to scale our approach and techniques in the future. As discussed in Section 7, our approach is able to "simplify" the AVR user's egocentric physical environment on the layout level. In the future, we will explore the effects of simplifying or abstracting the egocentric physical environment on the object level, i.e., preserving the interactive parts of a physical object and replacing the rest with contour lines. By doing so, we may further enhance the immersive experience in the AVR environments and find other benefits. We also want to explore hybrid interfaces based on physical affordance further. We can arrange more types of virtual elements on the interactive 3D plane extended from the physical notepad, e.g., adding a "*finger slider*" along the bottom edge of

the notepad and using the pen instead of a finger to manipulate it. By doing so, our *augmented physical notepad* may be applied to more usage scenarios. We can also fold a virtual widget into multiple (more than two) segments and align them with the two sides of the rims of smaller objects such as books and cellphones. In this way, we can attach our *finger slider*s on almost all kinds of physical objects with straight edges.

ACKNOWLEDGMENTS

This work is supported by research grants from the Research Grants Council of Hong Kong Special Administrative Region, under Project No. CUHK14225616 and Project No. CUHK14201918, Shenzhen Science and Technology Program (No. JCYJ20170413162617606), and SoC robotics initiative C-251-000-065-001. We thank the reviewers cordially for their valuable comments, Shuxia Zhang for drawing Figures 1 and 14, Zouyan He for creating the texture of the hollowed path with an arrow, and Yuming Bai for implementing the fully virtual version of the *augmented notepad*.

REFERENCES

[1] Mahdi Azmandian, Mark Hancock, Hrvoje Benko, Eyal Ofek, and Andrew D. Wilson. 2016. Haptic Retargeting: Dynamic Repurposing of Passive Haptics for Enhanced Virtual Reality Experiences. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 1968–1979. https://doi.org/10.1145/2858036.2858226

[2] G. Bruder, F. Steinicke, K. Rothaus, and K. Hinrichs. 2009. Enhancing Presence in Head-Mounted Display Environments by Visual Body Feedback Using Head-Mounted Cameras. In *2009 International Conference on CyberWorlds*. 43–50. https://doi.org/10.1109/CW.2009.39

[3] Gerd Bruder, Frank Steinicke, Dimitar Valkov, and Klaus Hinrichs. 2010. Augmented virtual studio for architectural exploration. In *Proc. of Virtual Reality International Conference (VRIC 2010)*. 43–50.

[4] Pulkit Budhiraja, Rajinder Sodhi, Brett Jones, Kevin Karsch, Brian Bailey, and David Forsyth. 2015. Where's My Drink? Enabling Peripheral Real World Interactions While Using HMDs. *arXiv preprint arXiv:1502.04744* (2015).

[5] Lung-Pan Cheng, Li Chang, Sebastian Marwecki, and Patrick Baudisch. 2018. iTurk: Turning Passive Haptics into Active Haptics by Making Users Reconfigure Props in Virtual Reality. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. ACM, New York, NY, USA, Article 89, 10 pages. https://doi.org/10.1145/3173574.3173663

[6] D. Clergeaud and P. Guitton. 2017. Design of an annotation system for taking notes in virtual reality. In *2017 3DTV Conference: The True Vision - Capture, Transmission and Display of 3D Video (3DTV-CON)*. 1–4. https://doi.org/10.1109/3DTV.2017.8280398

[7] A. P. Desai, L. Peña-Castillo, and O. Meruvia-Pastor. 2017. A Window to Your Smartphone: Exploring Interaction and Communication in Immersive VR with Augmented Virtuality. In *2017 14th Conference on Computer and Robot Vision (CRV)*. 217–224. https://doi.org/10.1109/CRV.2017.16

[8] Data from centers for disease control and prevention. 2019. https://www.cdc.gov/nchs/data/series/sr_11/sr11_249.pdf [Online; accessed 27-March-2019].

[9] Lei Gao, Huidong Bai, Weiping He, Mark Billinghurst, and Robert W. Lindeman. 2018. Real-time Visual Representations for Mobile Mixed Reality Remote Collaboration. In *SIGGRAPH Asia 2018 Virtual & Augmented Reality (SA '18)*. ACM, New York, NY, USA, Article 15, 2 pages. https://doi.org/10.1145/3275495.3275515

[10] Lei Gao, Huidong Bai, Gun Lee, and Mark Billinghurst. 2016. An Oriented Point-cloud View for MR Remote Collaboration. In *SIGGRAPH ASIA 2016 Mobile Graphics and Interactive Applications (SA '16)*. ACM, New York, NY, USA, Article 8, 4 pages. https://doi.org/10.1145/2999508.2999531

[11] Lei Gao, Huidong Bai, Thammathip Piumsomboon, G Lee, Robert W Lindeman, and Mark Billinghurst. 2017. Real-time visual representations for mixed reality remote collaboration. In *International Conference on Artificial Reality and Telexistence*. 87–95.

[12] J. Garcia Estrada and A. L. Simeone. 2017. Recommender system for physical object substitution in VR. In *2017 IEEE Virtual Reality (VR)*. 359–360. https://doi.org/10.1109/VR.2017.7892325

[13] iFLYTEK voice recognition SDK. 2019. http://global.xfyun.cn/ [Online; accessed 25-February-2019].

[14] Humayun Khan, Gun Lee, Simon Hoermann, Rory Clifford, Mark Billinghurst, and Robert W Lindeman. 2017. Evaluating the Effects of Hand-gesture-based Interaction with Virtual Content in a 360 Movie. In *International Conference on Artificial Reality and Telexistence*. 63–70.

[15] Youngwon R. Kim and Gerard J. Kim. 2016. HoVR-type: Smartphone As a Typing Interface in VR Using Hovering. In *Proceedings of the 22Nd ACM Conference on Virtual Reality Software and Technology (VRST '16)*. ACM, New York, NY, USA, 333–334. https://doi.org/10.1145/2993369.2996330

[16] G. A. Lee, J. Chen, M. Billinghurst, and R. Lindeman. 2016. Enhancing Immersive Cinematic Experience with Augmented Virtuality. In *2016 IEEE International Symposium on Mixed and Augmented Reality (ISMAR-Adjunct)*. 115–116. https://doi.org/10.1109/ISMAR-

Adjunct.2016.0054
[17] Point Cloud Library. 2019. https://pointclouds.org/ [Online; accessed 27-April-2019].
[18] Jia-Wei Lin, Ping-Hsuan Han, Jiun-Yu Lee, Yang-Sheng Chen, Ting-Wei Chang, Kuan-Wen Chen, and Yi-Ping Hung. 2017. Visualizing the Keyboard in Virtual Reality for Enhancing Immersive Experience. In *ACM SIGGRAPH 2017 Posters (SIGGRAPH '17)*. ACM, New York, NY, USA, Article 35, 2 pages. https://doi.org/10.1145/3102163.3102175
[19] ARUCO Marker. 2019. https://docs.opencv.org/3.1.0/d5/dae/tutorial_aruco_detection.html [Online; accessed 27-April-2019].
[20] Lynn Mcatamney and E. Nigel Corlett. 1993. RULA: A Survey Method for the Investigation of Work-Related Upper Limb Disorders. *Applied ergonomics* 24 (05 1993), 91–99. https://doi.org/10.1016/0003-6870(93)90080-S
[21] J. McCormac, A. Handa, A. Davison, and S. Leutenegger. 2017. SemanticFusion: Dense 3D semantic mapping with convolutional neural networks. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*. 4628–4635. https://doi.org/10.1109/ICRA.2017.7989538
[22] Mark McGill, Daniel Boland, Roderick Murray-Smith, and Stephen Brewster. 2015. A Dose of Reality: Overcoming Usability Challenges in VR Head-Mounted Displays. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI '15)*. ACM, New York, NY, USA, 2143–2152. https://doi.org/10.1145/2702123.2702382
[23] D. Mendes, F. Fonseca, B. Araùjo, A. Ferreira, and J. Jorge. 2014. Mid-air interactions above stereoscopic interactive tables. In *2014 IEEE Symposium on 3D User Interfaces (3DUI)*. 3–10. https://doi.org/10.1109/3DUI.2014.6798833
[24] P. J. Metzger. 1993. Adding reality to the virtual. In *Proceedings of IEEE Virtual Reality Annual International Symposium*. 7–13. https://doi.org/10.1109/VRAIS.1993.380805
[25] Paul Milgram and Herman Colquhoun. 1999. A taxonomy of real and virtual world display integration. *Mixed reality: Merging real and virtual worlds* 1 (1999), 1–26.
[26] D. Nahon, G. Subileau, and B. Capel. 2015. "Never Blind VR" enhancing the virtual reality headset experience with augmented virtuality. In *2015 IEEE Virtual Reality (VR)*. 347–348. https://doi.org/10.1109/VR.2015.7223438
[27] Don Norman. 2013. *The design of everyday things: Revised and expanded edition*. Basic books.
[28] I. Poupyrev, N. Tomokazu, and S. Weghorst. 1998. Virtual Notepad: handwriting in immersive VR. In *Proceedings. IEEE 1998 Virtual Reality Annual International Symposium (Cat. No.98CB36180)*. 126–132. https://doi.org/10.1109/VRAIS.1998.658467
[29] J. Redmon and A. Farhadi. 2017. YOLO9000: Better, Faster, Stronger. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 6517–6525. https://doi.org/10.1109/CVPR.2017.690
[30] Intel RealSense SDK. 2019. https://software.intel.com/en-us/realsense-sdk-windows-eol [Online; accessed 27-April-2019].
[31] OpenVR SDK. 2019. https://github.com/ValveSoftware/openvr [Online; accessed 27-April-2019].
[32] Adalberto L. Simeone, Eduardo Velloso, and Hans Gellersen. 2015. Substitutional Reality: Using the Physical Environment to Design Virtual Reality Experiences. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI '15)*. ACM, New York, NY, USA, 3307–3316. https://doi.org/10.1145/2702123.2702389
[33] Maurício Sousa, Daniel Mendes, Soraia Paulo, Nuno Matela, Joaquim Jorge, and Daniel Simões Lopes. 2017. VRRRRoom: Virtual Reality for Radiologists in the Reading Room. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. ACM, New York, NY, USA, 4057–4062. https://doi.org/10.1145/3025453.3025566
[34] Misha Sra, Sergio Garrido-Jurado, Chris Schmandt, and Pattie Maes. 2016. Procedurally Generated Virtual Reality from 3D Reconstructed Physical Space. In *Proceedings of the 22Nd ACM Conference on Virtual Reality Software and Technology (VRST '16)*. ACM, New York, NY, USA, 191–200. https://doi.org/10.1145/2993369.2993372
[35] F. Steinicke, G. Bruder, K. Rothaus, and K. Hinrichs. 2009. Poster: A virtual body for augmented virtuality by chroma-keying of egocentric videos. In *2009 IEEE Symposium on 3D User Interfaces*. 125–126. https://doi.org/10.1109/3DUI.2009.4811218
[36] E. A. Suma, D. M. Krum, and M. Bolas. 2011. Sharing space in mixed and virtual reality environments using a low-cost depth sensor. In *2011 IEEE International Symposium on VR Innovation*. 349–350. https://doi.org/10.1109/ISVRI.2011.5759673
[37] Franco Tecchia, Giovanni Avveduto, Raffaello Brondi, Marcello Carrozzino, Massimo Bergamasco, and Leila Alem. 2014. I'm in VR!: Using Your Own Hands in a Fully Immersive MR System. In *Proceedings of the 20th ACM Symposium on Virtual Reality Software and Technology (VRST '14)*. ACM, New York, NY, USA, 73–76. https://doi.org/10.1145/2671015.2671123
[38] Latency tool from the librealsense SDK. 2019. https://github.com/IntelRealSense/librealsense/tree/master/wrappers/opencv/latency-tool [Online; accessed 27-April-2019].
[39] James Walker, Bochao Li, Keith Vertanen, and Scott Kuhl. 2017. Efficient Typing on a Visually Occluded Physical Keyboard. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. ACM, New York, NY, USA, 5457–5461. https://doi.org/10.1145/3025453.3025783