Enlarging a Smartphone with AR to Create a Handheld VESAD (Virtually Extended Screen-Aligned Display)

Erwan Normand*
École de technologie supérieure

Michael J. McGuffin†
École de technologie supérieure

Abstract

We investigate using augmented reality to extend the screen of a smartphone beyond its physical limits with a virtual surface that is co-planar with the phone and that follows as the phone is moved. We call this extension a VESAD, or Virtually Extended Screen-Aligned Display. We illustrate and describe several ways that a VESAD could be used to complement the physical screen of a phone, and describe two novel interaction techniques: one where the user performs a quick rotation of the phone to switch the information shown in the VESAD, and another called "slide-and-hang" whereby the user can detach a VESAD and leave it hanging in mid-air, using the phone to establish the initial position and orientation of the virtual window. We also report an experiment that compared three interfaces used for an abstract classification task: the first using only a smartphone, the second using the phone for input but with a VESAD for output, and the third where the user performed input in mid-air on the VESAD (as detected by a Leap Motion). The second user interface was found to be superior in time and selection count (a metric of mistakes committed by users) and was also subjectively preferred over the other two interfaces. This demonstrates the added value of a VESAD for output over a phone's physical screen, and also demonstrates that input on the phone's screen was better than input in mid-air in our experiment.

Index Terms: H.5.2 [User Interfaces]: User Interfaces—Graphical user interfaces (GUI); I.3.6 [Computer Graphics]: Methodology and Techniques—Interaction techniques

1 Introduction

Smartphones with larger screens, of roughly 5 inches or more, have become increasingly popular [1], approaching the upper size limit of what can be comfortably held and operated with one hand. Nevertheless, mobile users would benefit from increased real-estate for accessing large numbers of apps, browsing through many photo thumbnails or files, or viewing large maps and high resolution photos, since eye motions are faster than scrolling with a finger.

At the same time, augmented reality (AR) head-mounted displays (HMDs) allow a user to be surrounded by large virtual information surfaces [12]. However, providing input by pointing with bare hands in mid-air suffers from tremor, arm fatigue [20, 21], a lack of physical feedback [30], and a lower Fitts index of performance [44] as compared to pointing on a physical screen with fingers that are stabilized by a smartphone.

We investigate a combination of these two technologies, with AR displaying a Virtually Extended Screen-Aligned Display (VESAD) that is centered on, and co-planar with, a smartphone (Fig. 1). This can provide a user with the same real-estate as a desktop monitor, but in a mobile usage scenario. The user continues to benefit from the reliable, stable, and precise input on the phone's physical screen, even using multitouch input as desired. No special interaction techniques are required for repositioning the virtual extension, since it is implicitly and continuously repositioned to follow the physical phone.

Figure 1: The extended space around the phone, displayed using augmented reality, is aligned to be co-planar with the phone's physical screen. The extended space around a phone could display additional screens and running applications that the user can switch to (mock-up).

We describe a prototype system with a wide ($> 90°$) field-of-view AR HMD, and present scenarios (Section 3) for use of smartphones with VESADs, including two novel interaction techniques. We also report an experiment (Section 5) comparing three modes of use (smartphone only, input on smartphone and output with VESAD, and input and output on the VESAD) for an abstract classification task introduced by Liu et al. [27]. Our results found that the second

Figure 2: A simulated 3rd-person view illustrating our experiment in the *MidAirInArOut* condition.

*e-mail: normand.erwan@protonmail.com
†e-mail: michael.mcguffin@etsmtl.ca

[1] For recent history, see https://en.wikipedia.org/wiki/Phablet

of these interfaces was superior in speed, rate of mistakes, and subjective preference. This demonstrates that the VESAD significantly enhances the phone's physical screen for output, but that the input in our experimental task was better on the phone's screen than in mid-air on the VESAD.

2 Background

Overcoming the limits of a physical screen is a theme found in much previous literature, and has been especially popular for mobile devices or other small screens.

One category of approaches involves adding no extra display hardware. A large literature exists around the idea of showing both a zoomed-out contextual view and a zoomed-in detailed of data [9]. There are also techniques for displaying information on a screen about objects located outside the screen's field-of-view, both for small screens [7] and for HMDs [16, 25, 32]. Detecting input gestures *around* a screen [22, 41, 45] is another way to enhance a small display area.

A second category of work involves adding hardware to extend a screen's physical limits for output. This has been done for small screens using projectors [24], for small screens using AR HMDs [15, 31], for small screens with a secondary screen [5, 33, 37], for desktop-sized displays using a projector [3, 23], for HMDs using LEDs [40], and for AR HMDs using projectors [4].

Also in this second category is previous work that has combined a hand-held device with a projector to provide a secondary display [19,39], however this required a physical surface on which to project, and the secondary display was not aligned with the device, inducing a division of attention between the two displays.

There is also much previous literature on large displays [1, 6, 10, 17, 27, 28, 43] and studies of limited field-of-view (FOV) [25] and peripheral vision [18]. The most relevant of these studies to our own work is Liu et al. [27] who compared a large wall-sized display where the user performed physical navigation (walking in front of the display) versus a desktop display where the user performed virtual navigation (panning and zooming with a mouse). Their study found that virtual navigation was faster for easier tasks, but that with more difficult tasks, physical navigation was superior. Regarding our current work, we note that a smartphone with a VESAD affords both virtual navigation through explicit input, as well as physical navigation by moving the user's arm sideways or backward or forward, however such arm motions are much easier and faster than walking. The results of our study, presented later, also found differences in the amount of virtual and physical navigation performed depending on the user interface technique in use.

Recently, there has been interest in virtual displays shown with AR that are connected to or aligned with a physical screen (MultiFi [15], Gluey [31], as well as the menus displayed outside, and aligned with, the hand-held devices in [26, 38]). The most relevant previous work is by Grubert et al. [15] who describe multi-fidelity (MultiFi) displays, combining AR HMDs and the physical displays on a smartphone and/or smartwatch. Their work identifies several design factors, such as the *alignment mode* which can be side-by-side (the HMD and physical displays are not spatially coordinated), body-aligned (the physical screen is a peephole [14,42]), or device-aligned (what we call a VESAD in our work). In this sense, a VESAD is a special case of MultiFi. Grubert et al. also discuss two input modalities: spatial pointing (moving the physical screen through space like a peephole), and touch input on the physical screen. Our present work extends this by also considering direct input in the space surrounding the smartphone, using a Leap Motion as input device, and we evaluate this experimentally. We also contribute new example designs and interaction techniques (Section 3).

3 Design Concept

In general, a VESAD could be used to extend any physical screen, including desktop displays or large displays. The current work specifically focuses on *handheld* VESADs, i.e., used with smartphones.

A smartphone screen held in the user's hand provides a higher angular resolution and enables more stable input than is possible with a state-of-the-art AR HMD. However, an HMD can display information covering a wider FOV than a hand-held phone. We identified several ways these two technologies could complement each other, with mock-ups to illustrate certain designs ideas.

- The VESAD can display alternative versions of what the phone already displays. For example, the phone and VESAD could display multiple phone-sized screen-fulls containing sets of app launcher icons and/or running apps (Fig. 1) and/or web browser tabs. Fig. 1 also indicates how the VESAD could display notifications with details (for example, the sender's name and subject of incoming messages, descriptions of upcoming appointments, and a chart of battery level over time).

- The VESAD can provide information that refers to the content on the phone. The content on the phone might be a user interface or a document, while the VESAD displays tooltips, annotations, or callouts (Fig. 3).

- The VESAD can show an extended view of content already on the phone. Fig. 4 shows this for a geographic map.

- The VESAD could display some content that is controlled with a UI on the phone. For example, the phone might show a gallery of thumbnails, where selecting one thumbnail causes an enlarged photo to be displayed on the VESAD (Fig. 6). As another example, the phone might display video and audio player controls, while the VESAD shows the video playback. A third example of this would be a text chat or video conferencing application, where the phone displays a list of contacts and options to invite people to conversations, while the VESAD displays a history of the text messages exchanged and/or videos of participants.

- The phone and VESAD could also be used for overview + details or focus + context visualizations of data [8, 9]. In such cases, either display modality could be used to show extra detail. For example, in Fig. 6, the gallery of thumbnails on the phone serves as a kind of overview, with the VESAD showing a detailed (zoomed in) view of one selected photo. However, these roles could be swapped to allow for more thumbnails to be visible at once (in the VESAD) while also displaying a single photo with greater pixel density (on the phone).

In addition to personal applications such as those just illustrated, VESADs could have industrial applications: a worker in a warehouse searching for an item, or a supervisor of a construction site, or an assembly-line repair technician looking for a machine that isn't working, could all benefit from a large, interactive, mobile display area showing an overview of their physical workplace.

We also imagined novel interaction techniques that could be convenient with such a hardware combination:

- We envision the VESAD normally following the phone and remaining co-planar to it. However, a simple wrist motion could be useful for quickly changing the orientation of the phone, triggering a temporary change in the VESAD's output, switching to another view. Fig. 5 shows the user flipping the phone $\approx 90^\circ$ around a vertical axis to switch to alternative apps and launcher icons. The wrist motion could be detected by the

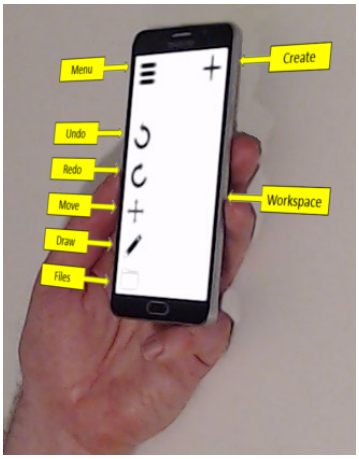


Figure 3: The extended space around a phone could be used to display tooltips for the current application (mock-up).

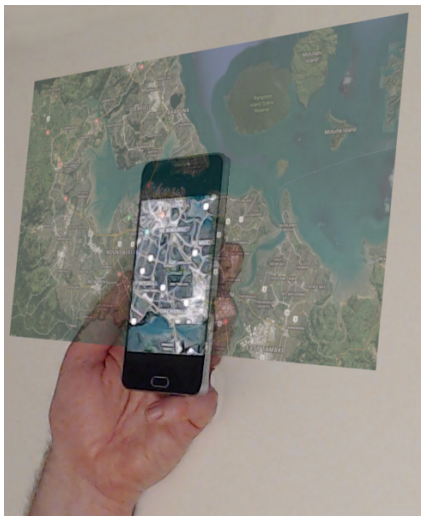


Figure 4: A map browser, with the view on the phone extending beyond the phone's physical limits (mock-up).

phone's inertial measurement unit (IMU), or be triggered by the angle between the phone's screen and the HMD's forward vector, and may depend on where the user is looking (e.g., a user looking away from the phone may be trying to avoid occlusion from a VESAD).

- A strength of the VESAD is that it implicitly follows the phone, remaining close to the user. However, in some cases, the user may want to "pin" virtual windows to a wall, a table, or suspend them in midair. We propose allowing the user to detach a VESAD from their phone and suspend it in a fixed position in world space. Fig. 7 shows how a "slide-to-hang" gesture could be used to convert two VESADs into such virtual windows. This gesture might be done with the phone on or near a wall or table, to "pin" windows there. It might also be done around a user's current position, to surround themself with windows suspended in 3D. Note that the HoloLens allows users to pin windows to flat surfaces and then subsequently reposition or rotate them, which requires several actions. By comparison, our proposed "slide-to-hang" gesture establishes the position, orientation, and detaching of a virtual window in a single action.

Figure 5: The hand rotates ≈90° to reveal an alternative view for quickly switching to other apps (mock-up).

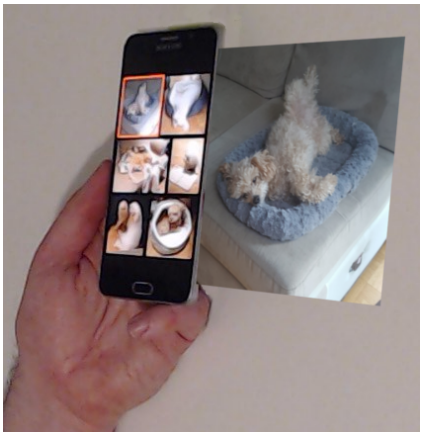


Figure 6: A photo gallery browser, where the extended space is used to view an enlarged version of a selected photo (mock-up).

4 Implementation

To prototype a VESAD, we considered using the Microsoft HoloLens, a state-of-the-art commercial *see-through* AR HMD with excellent latency and angular resolution. Unfortunately, its FOV is ≈32° horizontally per eye. Assuming a smartphone viewed at a distance of 34 cm ≈ 13.4 inches [2], this FOV covers 7.7 inches, which is insufficient for a VESAD of much interest around a 5-inch diagonal smartphone.

To achieve a wider FOV with commercial hardware, we implemented our prototype using *video pass-through* AR with hardware similar to AR-Rift [29, 35]. For the HMD, we used an Oculus Development Kit 2 (DK2) virtual reality headset, with 960 px × 1080 px per eye, 94° horizontally × 106° vertically per eye (95° horizontally for both eyes combined), yielding an angular resolution of 10.2 pixels per degree. The DK2 was fitted with an OVRVision Pro [2], providing stereo front-facing fisheye cameras at 960 px × 950 px per eye, 100° horizontally × 98° vertically per eye, or 9.6 pixels per degree.

The image displayed by the headset was updated at ≈30 FPS, with latency of ≈250 ms. The "undistortion" of the fisheye camera images was the most time-consuming computation. The headset was

[2] http://ovrvision.com

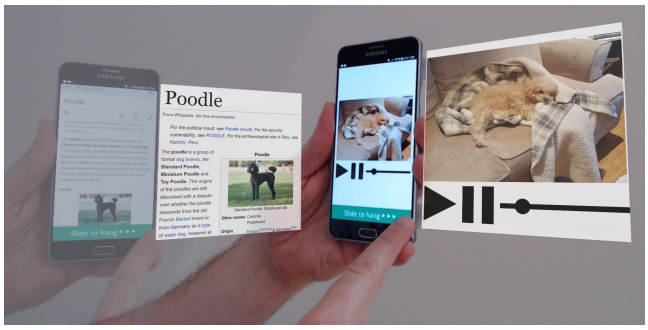


Figure 7: A "slide-to-hang" feature, allowing the user to slide the app in the phone toward the right, causing it to become suspended in 3D and detached from the phone, as if hanging in space. On the left, the user first slides out a webpage, then on the right, the user slides out a video player. This feature would allow a user to position apps around them in 3D, or onto physical surfaces such as walls and tables (mock-up).

connected to a desktop PC with an Intel Core i5 7400 CPU (4 cores at 3.0 GHz), 8 GB DDR4 RAM, NVIDIA GeForce GTX 1060 GPU with 6 GB memory, running Microsoft Windows 10.

For the smartphone, we used a Xiaomi Redmi Note 4 running Android 7 with a 1920 px × 1080 px, 5.5 inch diagonal screen, with a density of 401 pixels per inch (i.e., each pixel is 0.063 mm) (by comparison, the first iPhone marketed with a "Retina Display", the iPhone 4, featured 326 pixels per inch). Assuming an average viewing distance of 34 cm ≈ 13.4 inches [2], the phone's screen has an angular resolution of $401/\arctan(1/13.4) = 94$ pixels per degree when viewed without a headset. However, when viewed through the OVRVision, the angular resolution is reduced to 9.6 pixels per degree, producing an apparent screen resolution of 41 pixels per inch, as if each pixel was 0.62 mm.

Some of the software provided with the OVRVision Pro was rewritten in our lab. This effort resulted in a new open-source library called ArucoUnity [3], which is (i) C# bindings for calling three modules of the open-source C++ library OpenCV: aruco (to detect fiducial markers), calib3d and ccalib (to calibrate and undistort regular webcams and wide field-of-view fisheye cameras), and (ii) C# scripts developed on top of the bindings to facilitate camera calibration and marker tracking from the Unity editor (a 3D game engine using C# as its scripting language) [4].

ArucoUnity is currently the only free and open-source AR library for Unity/C# that supports both monoscopic and stereoscopic cameras with regular or fisheye lenses and that makes OpenCV 3.3's recent functionality available. Other C# libraries are either closed-source, or don't support fisheye cameras, or have not been updated to use OpenCV 3.3, or are not compatible with Unity.

Communication between the phone and the headset's computer was over a USB cable using Unet, the built-in networking service of Unity.

5 Experiment

Our experiment investigates (1) whether the VESAD provides any advantage over displaying output only on a phone, and (2) whether it is better to have input only on the phone, or to perform input in mid-air on the VESAD's plane. Input on the phone has the advantage that the physical surface of the phone stabilizes the finger, making input more precise, enabling reliable detection of touch events, and where the Fitts' index of performance is theoretically higher because

[3] https://github.com/NormandErwan/ArucoUnity

[4] See our tutorials: https://normanderwan.github.io/ArucoUnity

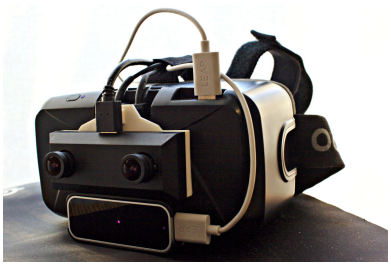


Figure 8: Oculus DK2 headset with OVRVision cameras for video pass-through augmented reality, and with a Leap Motion to detect fingers.

Figure 9: Smartphone fitted with ArUco markers.

the fingers, rather than the arms, are involved. However, mid-air input is more direct, possibly more intuitive, and may benefit from the user performing coordinated, simultaneous motions of both arms (e.g., moving hands in opposite directions so that the non-dominant hand "pulls" a target closer to the dominant hand that is reaching toward the target).

5.1 Task

Several tasks could have been used. We considered performing a Fitts-style pointing task [34] where the user must navigate to different targets and select them; interpreting a map of a road network [3] or circuit diagram to find connections or pathways; or implementing a multi-tasking scenario where the user must switch between, and interact with, several apps [13]. Finally, we decided to use the classification task used by Liu et al. [27], where users must categorize items by moving them into containers. This task is more realistic than a pure pointing task (but also includes pointing as a subtask); it cannot be automated like path-finding in a road network could be; and it is easier to implement and easier to control than a multi-tasking scenario with many apps.

Liu et al.'s classification task requires users to group abstract items of the same type by moving items into containers with matching items. At the start of each trial, the user is presented with a grid like the one in Fig. 10, where a small number of items are mis-classified and highlighted in red. The user must move these red items into their correct containers (i.e., into containers that contain a majority of matching items) to complete the trial. In a real-world scenario, the judgment of which items actually match could correspond to any criterion evaluated by the user, and could require the user to zoom in on each item to examine it in detail. This is operationalized in the experimental task by labeling each item with a small single-letter label, and by requiring users to group together items with the same label. Furthermore, to focus the task on correctly categorizing items, any mis-classified items are highlighted in red. In a real-world scenario, such highlighting would of course be absent, however having it in the experiment reduces the variance of measured variables within each condition, increasing statistical power.

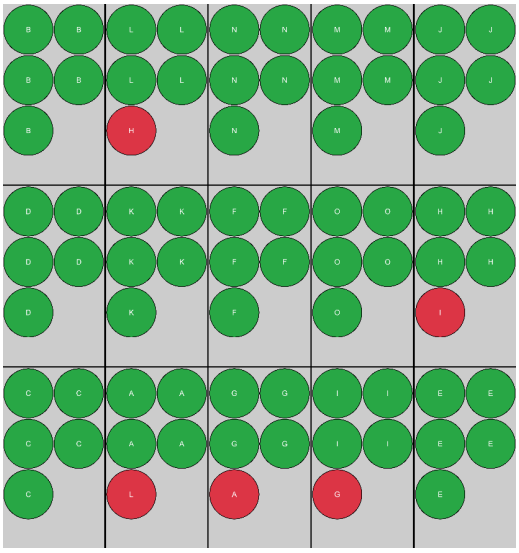


Figure 10: The experimental task grid, in the TEXTSIZE=*Big* condition.

Our grid was a 3×5 matrix of rectangular containers (Fig. 10). Each container holds a maximum of 6 circular items. The grid contents are randomly generated each time a trial begins: Each container is assigned 5 items (of which at least 4 have matching labels), and 5 items in the grid are initially in the wrong container. There are 15 containers × 5 items/container = 75 items in the entire grid.

A unique single-letter label, from A to O, is assigned to each container and its correctly-classified items. Each item can be correctly classified in only one container. The label of an item is displayed at its center. To simplify the searching subtask, similarly to Liu et al., we colored correctly classified items in green, mis-classified items in red, and the currently selected item in blue (Fig. 11). To know the label associated with a container, one must read the letter of any green item in the container.

The goal of the task is to classify all the red items in their matching container to "make everything green" [27].

5.2 Design

Our experiment manipulated the following independent variables:

- The TECHNIQUE could be *PhoneOnly* (input and output on the phone, with no VESAD), *PhoneInArOut* (input on the phone, but output on the phone and on a VESAD), and *MidAirInArOut* (input in mid-air, sensed with a Leap Motion, and output on the phone and on a VESAD);
- The TEXTSIZE of the labels on the items could be *Big* or *Small*;
- The average DISTANCE between initially mis-classified items and their matching containers could be *Near* or *Far*.

The TECHNIQUE, DISTANCE, and TEXTSIZE variables were all within subjects (i.e. each user experienced all levels of each of these variables). Since TECHNIQUE was the main variable of interest, the order of techniques was counterbalanced with a 3×3 Latin square. DISTANCE and TEXTSIZE had a fixed ordering from easier to more difficult. In total, there were 12 participants × 3 techniques × 2 distances × 2 text sizes × 2 repetitions = 288 trials.

We followed Dragicevic's advice ("Tip 9: As many observations as participants" for each condition [11]) and averaged each pair of repetitions to obtain 144 observations for the analysis we present later.

The ORDERING of techniques was an additional independent variable with 3 levels, and was of course between subjects (i.e., each user only experienced one ordering of the Latin square): *Group1* started with the *PhoneOnly* condition, *Group2* with *PhoneInArOut*, and *Group3* with *MidAirInArOut*.

In all conditions of the experiment, the user wears the same AR headset, to maintain the same headset weight, feedback latency, and angular resolution across conditions, even when the user does not benefit from any augmented reality.

5.3 Difficulty

Task difficulty is controlled with the TEXTSIZE and DISTANCE factors. At the start of each trial, the grid is zoomed such that each container has the size of the phone screen (68 mm × 121 mm), and the diameter of items is half of a container's width (34 mm × 34 mm). The font size of the item's letter is 12 pt (4.22 mm × 4.22 mm, 12% of the item diameter) for the *Big* text condition, and 10 pt (3.51 mm × 3.51 mm, 10% of the item diameter) for the *Small* text condition. (Assuming the viewing distance of 13.4 inches stated in Section 4, this means that each letter of *Big* text was 6.6 px high when viewed through the headset, if the grid was zoomed such that one container filled the phone's screen). We generated random initial layouts and chose ones who distances matched the conditions reported by Liu et al. [27]: the average distance (measured in containers) between a mis-classified item and its container is between 1.25 and 1.45 for the *Near* distance condition and between 2.5 and 2.7 for the *Far* distance condition.

A pilot study with 3 users involved a larger number of initially mis-classified items as well as smaller text labels, however this proved to be excessively difficult and time consuming for users, especially in the *MidAirInArOut* condition. The smaller text size was thus increased, and the number of mis-classified items reduced to 5 items per trial, so that users could complete the entire experiment in under one hour.

5.4 Interaction Details

In all three techniques, the user performed pick-and-drop actions to move mis-classified items into their correct container. Each pick-and-drop action was initiated by selecting an item and later selecting the container to drop it in. The user could also pan and zoom the entire grid.

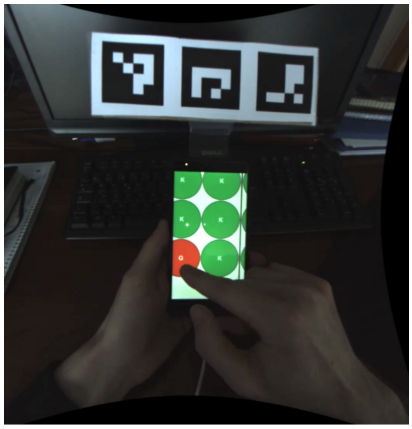
Each trial required the user to correctly classify all 5 items that were initially mis-classified. When an item was dropped to an incorrect container, it was counted as an error, but users still had to successfully classify all 5 items.

With the *PhoneOnly* and *PhoneInArOut* techniques, selecting an item or a container was done with a tap on the phone's screen that had to last less than 500 ms; panning was done by pressing and dragging a single finger; and zooming was done with two fingers using the status quo pinch-to-zoom gesture.

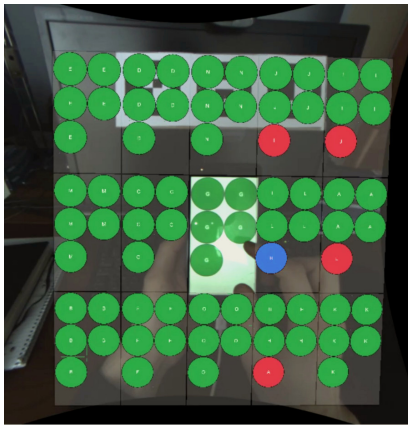
Also, with *PhoneInArOut*, the user directly pointed at targets, but this could only be done on the portion of the grid visible within the phone's screen. To select a target item or target container outside the phone's screen, the user first had to pan the target into the phone's screen (or zoom out enough to make the target visible on the phone's screen).

In all techniques, panning responded with a 1:1 gain, as if the grid was a sheet of paper moving under the friction of the finger tip.

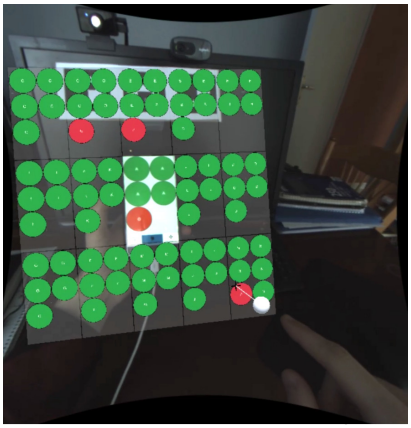
With the *MidAirInArOut* condition, zoom could not be done with two fingers as the positions of two fingers in mid-air could not be reliably detected. Instead, 3 modal buttons were displayed along the bottom of the phone's screen (Fig. 11 (c)). Users were instructed to press these buttons with the thumb of their non-dominant hand (the hand holding the phone) to switch to Select, Pan, or Zoom mode.



(a) *PhoneOnly* condition.



(b) *PhoneInArOut* condition.



(c) *MidAirInArOut* condition.

Figure 11: The experimental task grid for the three techniques. In the *PhoneInArOut* condition, an item is selected (in blue), and we see that the user's hand is partially occluded by the items displayed on the VESAD, but not occluded by the phone's display. In the *MidAirInArOut* condition, a white sphere indicates the finger tips position (as sensed with a Leap Motion) and a white line segment and black cross show the projected finger position on the VESAD. The line and the cross turn blue when the sphere touches the VESAD's plane.

After switching to any of these modes, the index of the the user's dominant hand could then touch down on the VESAD's virtual plane to intersect with it. Selecting an item or a container required switching to Select mode with the non-dominant hand, and then performing a "long press" (longer than 500 ms) with the dominant hand on the VESAD. Panning or zooming required switching to Pan or Zoom mode with the non-dominant hand, and then pressing and dragging with the dominant hand on the VESAD's plane. Zooming was centered on the center of the phone, not of the grid, as the index dragged in and out from the phone to zoom.

Note that, in the *MidAirInArOut* condition, the modal buttons displayed along the bottom of the phone's screen were small enough that a complete container with 6 items was still visible with no occlusion on the phone's screen.

In the *PhoneInArOut* and *MidAirInArOut* conditions, the virtual content in the VESAD was rendered on top of the frames captured by the HMD's front-facing cameras. This resulted in incorrect depth occlusion cues, namely, the dominant hand was occluded by the items which are theoretically *behind* the hand (Fig. 11 (c)). We mitigated this problem in two ways: first, the empty space within each container was transparent and the items were semi-transparent to allow the hand to remain partially visible, and second, the projected position of the dominant hand's finger on the VESAD was shown with a line segment and a black cross. None of the participants complained about the occlusion of the hand by the VESAD. This is somewhat like [29, Fig. 3C] where participants of a pilot study preferred a reconstructed hand drawn semi-transparently over virtual objects as much as correct occlusion of the hand with virtual objects.

5.5 Participants

We recruited 12 volunteers (3 females), aged from 18 to 49 (two above 25). Twelve have normal or corrected vision, and one self-reported to be color blind but was able to distinguish all the task elements. Ten were right-handed and 2 left-handed; we asked each participant to hold the phone in their non-dominant hand. Eight had already used a VR HMD, and 1 also an AR HMD. Eight regularly use 3D software and all use a computer daily.

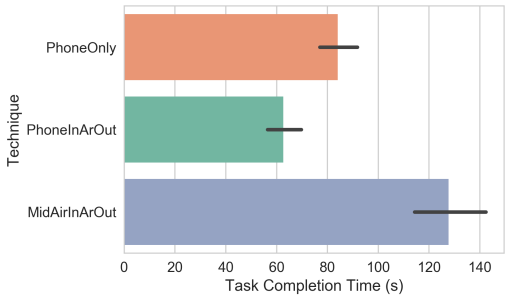


Figure 12: The geometric mean of Task Completion Time (TCT) for each technique. Error bars show the 95% confidence intervals.

5.6 Results

5.6.1 Task Completion Time

We first transformed the Task Completion Time (TCT) with a log transform to approximate a normal distribution [11]. We confirmed normality with Shapiro-Wilk tests ($p > 0.05$ for each technique) and equal variances with a Levene test ($p > 0.05$).

We then analyzed the transformed TCT with a multifactorial ANOVA, which found a significant effect due to TECHNIQUE ($F_{2,22} = 62.2$, $p = 2 \times 10^{-19}$), ORDERING ($F_{2,22} = 2.1$, $p = 5 \times 10^{-5}$), and their interaction TECHNIQUE $\times$ ORDERING ($F_{4,44} = 3.1$, $p = 1 \times 10^{-5}$), but no effect due to DISTANCE nor TEXTSIZE ($p > 0.05$).

A pairwise t-test (Benjamini-Hochberg correction) compared the TCT for the 3 techniques, and found them to be all significantly different ($p < 0.0001$ for all pairs). We computed means and 95% CI from the transformed TCT and antilogged them (Fig. 12). The fastest technique is *PhoneInArOut*, which is 22 s (33% faster) than *PhoneOnly*, which is itself 49 s (36% faster) than *MidAirInArOut*.

5.6.2 Errors and Selections

Two kinds of mistakes could happen during trials. First, a user could select an item and drop it in an incorrect container. This was counted toward the number of ERRORS for the trial. Second, a user could select one item, and while searching for the container to drop it in,

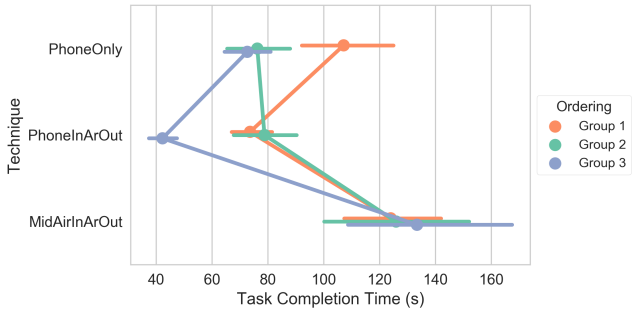


Figure 13: Ordering has an effect on TCT. Error bars are 95% CI.

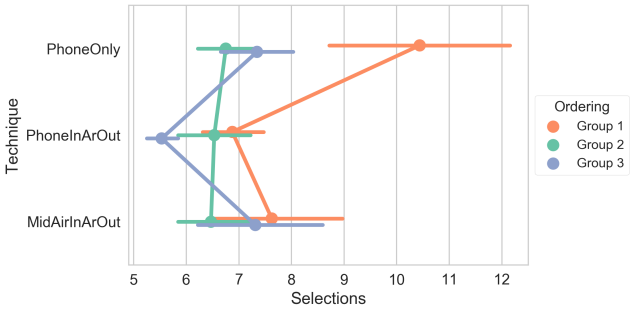


Figure 15: Ordering had an effect on selections. Error bars are 95% CI.

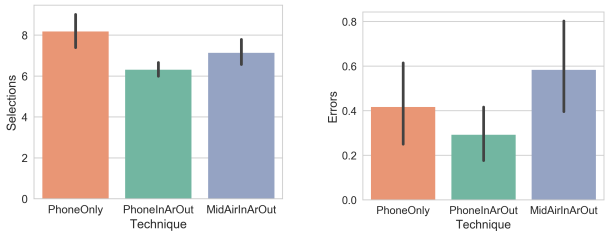


Figure 14: Left: Selections per trial, which include successfully moved items, as well as canceled selections and errors. Right: Errors per trial, which are unsuccessfully moved items. In both charts, a lower number is better. Error bars are 95% CI.

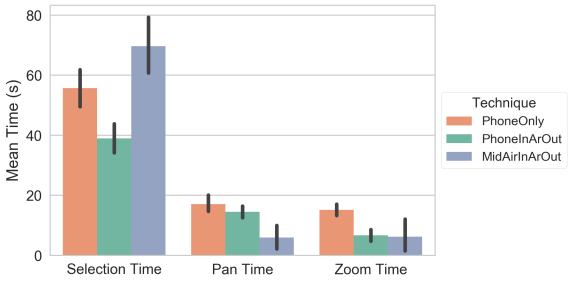


Figure 16: The total time (with 95% CI error bars) spent per trial with an item selected, or panning, or zooming. Note that these times could overlap.

they might decide to select a different item (possibly because they forgot the letter of the first item), causing their previous selection to be canceled. This was not counted toward ERRORS, but was counted toward the total number of SELECTIONS. Because there were 5 mis-classified items to move during each trial, SELECTIONS was at least 5 in each trial, and in general, SELECTIONS = 5 + ERRORS + (number of times the user canceled a selection). Hence, ERRORS is a more conservative measure of classification errors, whereas SELECTIONS includes both kinds of mistakes.

We analyzed both SELECTIONS and ERRORS, using Kruskal-Wallis tests (Benjamini-Hochberg correction) to check for effects due to TECHNIQUE, TEXTSIZE, DISTANCE, and ORDERING. We found that TECHNIQUE ($p = 0.004$) and ORDERING ($p = 0.01$) had significant effects on SELECTIONS, but only ORDERING ($p = 0.01$) had a significant effect on ERRORS. We found no effect for the other independent variables.

To further analyze the effect of TECHNIQUE on SELECTIONS, we used a pairwise Mann-Whitney test (Benjamini-Hochberg correction). There were significant differences for the {*PhoneOnly*, *PhoneInArOut*} ($p = 0.0006$) and {*PhoneOnly*, *MidAirInArOut*} ($p = 0.03$) pairs, but not for the {*PhoneInArOut*, *MidAirInArOut*} ($p = 0.06$) pair.

The poor performance of *PhoneOnly* in terms of SELECTIONS agrees with our observation that, when using *PhoneOnly*, users sometimes forgot what they had selected or changed their mind during the drop operation, increasing the number of selections.

At the same time, the poor performance of *MidAirInArOut* in terms of ERRORS agrees with our observation that, with this technique, users sometimes dropped items in the wrong container, not voluntarily but because they were too zoomed out and/or the sensing limitations of the Leap Motion made it difficult to successfully aim at targets, especially when arms were crossed (i.e., the user was trying to drop an item on the opposite side of the arm holding the phone).

Finally, breaking SELECTIONS down by ORDERING (Fig. 15), it appears that users committed more mistakes in the *PhoneOnly* condition when that was the first condition they experienced (*Group1*), suggesting that this technique was initially more prone to mistakes, possibly due to the limited screen size.

5.6.3 Operations within each trial

To gain a deeper understanding of the source of differences between the techniques, for each trial, we recorded the number of times that the user performed a pan or zoom, and the total time spent on these operations, as well as the total physical distance (as measured projected onto the VESAD plane) traveled by the dominant hand's index finger (the Leap Motion was used to measure these distances, regardless of which TECHNIQUE was in use). We similarly quantified the moments when an item was selected, in terms of time and distance, to quantify how much the user was performing pick-and-drop actions. We also recorded how much head motion was performed, by finding the distance moved by the head (in any direction) with respect to the phone, from frame to frame, and adding up all these displacements. The data are shown from Fig. 16 to Fig. 18.

A first question we consider is: how much virtual navigation (panning and zooming) versus physical navigation (moving the phone with respect to the head) was performed by users in each condition? Virtual navigation was more prominent in the first two techniques. Fig. 17 shows that the number of pan and zoom operations for *PhoneOnly* and *PhoneInArOut* was more than 5 per trial, implying that each item to be classified involved, on average, more than one pan and one zoom operation. With the *MidAirInArOut* technique, the count is lower, however Fig. 18 (lower left) shows that this technique involved more *physical* navigation as measured by head-phone motion. This can be explained by two observations made during the experiment. First, users often performed physical navigation with *MidAirInArOut* to bring target containers closer to their dominant

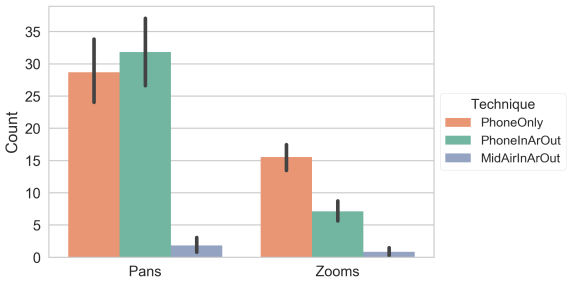


Figure 17: The mean (with 95% CI error bars) number of pans or zooms per trial.

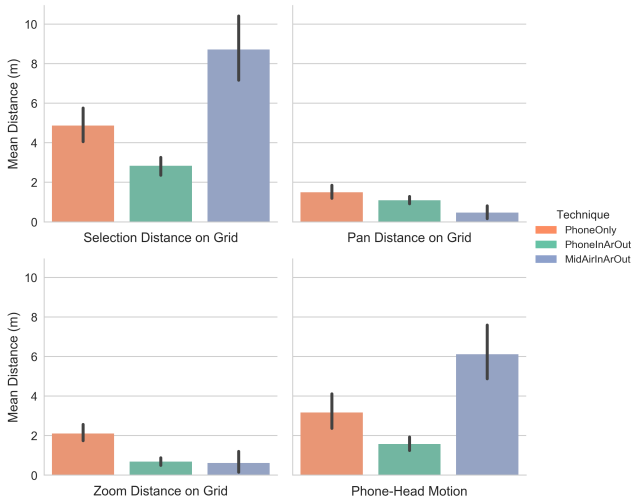


Figure 18: The total distance dragged or moved while (upper left) an item was selected, (upper right) panning, (lower left) zooming, and (lower right) total distance moved by the head with respect to the phone. All quantities are per trial. Error bars are 95% CI.

hand. Second, users also found it difficult to perform virtual navigation with *MidAirInArOut* due to somewhat unreliable sensing with the Leap Motion.

A second question is: What was the main weakness of the *MidAirInArOut* technique? This technique performed well, compared to the others, in terms of panning and zooming, requiring few of these operations as measured by distance, time, and count. However, *MidAirInArOut* also has the worst selection distance Fig. 18 (upper left), indicating the users moved their dominant more when performing pick-and-drop operations to move items to containers. This is an inherent problem with direct input techniques: the user's arm must perform larger motions to touch content directly. User feedback indicates that motions requiring them to cross there arms, to drop an item on the other side of the non-dominant arm, were tiring or awkward. An added problem is the lack of a physical surface that is touched to stabilize the finger's motion during dragging and to provide non-visual feedback. The use of a Leap Motion was also not as reliable for sensing finger position as the phone's touchscreen.

A third question: what explains the smaller TCT with the *PhoneInArOut* technique compared to *PhoneOnly*? Fig. 16 shows that *PhoneInArOut* required somewhat less time for zooming and slightly less for panning, but the differences are insufficient to explain the difference in total time. We also see that *PhoneInArOut* resulted in much less selection time than *PhoneOnly*. Since most of the difference in time was not spent panning nor zooming, we

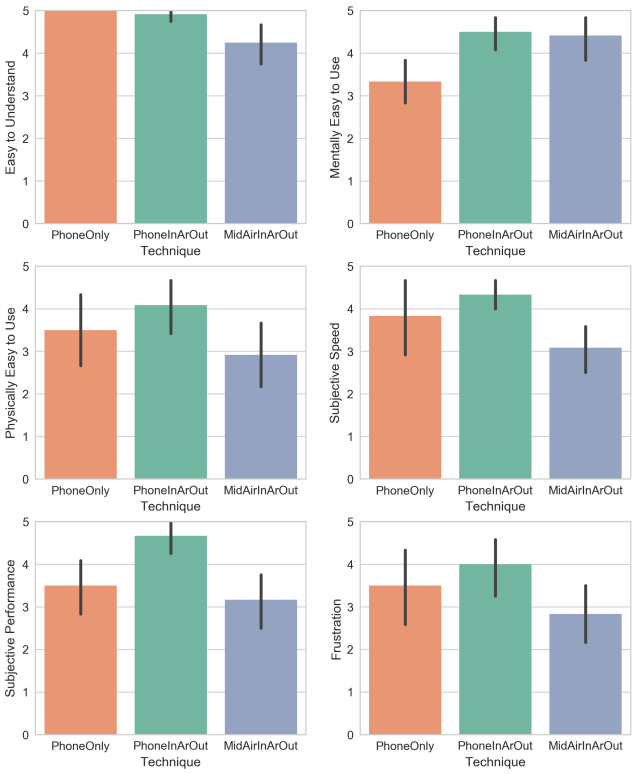


Figure 19: The average Likert score with bootstrapped 95% CI error bars given to each technique according to six criteria. In all cases, a larger number is better. For example, a higher Frustration score means "less frustrating".

suspect it was simply easier to find each target container in the *PhoneInArOut* because of the increased display area.

5.6.4 Subjective Measures

Users rated the three techniques according to 6 criteria on Likert scales and also gave a ranking for overall Preference. Fig. 19 and Fig. 20 present the average score for each criteria or the average preference, both with bootstrap 95% confidence intervals (CI) as error bars.

A Kruskal-Wallis test (Benjamini-Hochberg correction) on each criterion checked for statistical significant differences due to TECHNIQUE. Five of the criteria were significantly affected by TECHNIQUE: Easy to Understand ($p = 0.007$), Mentally Easy to Use ($p = 0.007$), Subjective Speed ($p = 0.04$), Subjective Performance ($p = 0.006$), and overall Preference ($p = 0.006$), whereas Physically Easy to Use and Frustration were not significantly affected.

We then used pairwise Mann-Whitney test (with Benjamini-Hochberg correction) for the significant criteria, finding the following:

- Easy to Understand: *PhoneOnly* is significantly better than *MidAirInArOut* ($p = 0.009$). This is not surprising, since the *PhoneOnly* technique is more familiar to users.
- Mentally Easy to Use: *PhoneOnly* is significantly worse than *PhoneInArOut* ($p = 0.005$) and *MidAirInArOut* ($p = 0.01$). This may be due to the limited screen size of the phone.
- Subjective Speed: *PhoneInArOut* was judged to be significantly faster than *MidAirInArOut* ($p = 0.05$).

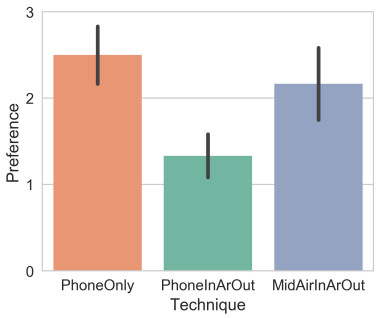


Figure 20: The average ranking, or "preferences" of users, of each technique with bootstrapped 95% CI error bars. Lower is better.

- Subjective Performance: *PhoneInArOut* is significantly better than *PhoneOnly* ($p = 0.005$) and *MidAirInArOut* ($p = 0.004$). Performance was defined for the user as taking into account speed and errors.
- Overall Preference: *PhoneInArOut* is significantly preferred over *PhoneOnly* ($p = 0.004$) and over *MidAirInArOut* ($p = 0.01$).

Users were also asked their opinion on various concept applications for VESADs. More than half the participants thought the extended map application (Fig. 4) would be useful, and all of them thought the multi-tasking support (Fig. 1) would be valuable.

6 Discussion

Although users preferred the *PhoneInArOut* technique overall, some participants felt that *MidAirInArOut* is the technique with the most potential because it allows directly selecting items. We observed users spontaneously reaching out to touch items on the VESAD before being prompted to do so. Several users wished that they could pan and zoom in the *MidAirInArOut* technique without having to switch modes through a menu, similar to pinch-to-zoom on status quo phone interfaces. Unfortunately, despite our use of off-the-shelf sensing technology (Leap Motion), the sensing of the finger position was not always reliable. Better hand-tracking techniques [36] could improve this in the future.

The physical navigation performed by users in the *MidAirInArOut* technique was often to bring a distant target (e.g., located in a corner) closer to the dominant hand. This suggests that targets for the input on the VESAD should not be placed too far from the phone, and this also recalls the recommendation by Ens et al. [13] that windows suspended in front of the user should be given an equidistant, spherical layout.

In Grubert et al.'s experimental comparison [15], the only device-aligned condition that was evaluated involved a smartwatch with a 40×35 mm touchscreen, and the experiment found that this device-aligned condition (called "SWRef") was not significantly better than the smartwatch alone ("SW") in terms of either time or errors. One inconvenience discussed with the SWRef condition was the difference in focal distance for the HMD (≈300cm) and smartwatch (≈40cm). In our current work, our device-aligned VESAD involves the much larger physical screen of a smartphone, and the *PhoneInArOut* version of it was found to be superior to the smartphone alone (*PhoneOnly*). Unlike Grubert et al., our experiment maintained a constant focal distance, headset weight, feedback latency, and angular resolution across conditions by having users always wear the same video pass-through HMD. This has the disadvantage of penalizing the *PhoneOnly* condition by reducing the smartphone's normal display resolution and performance, however the same "penalty" applies across all conditions in our experiment, and we anticipate future AR HMDs will be much lighter and higher performance. If we had compared a phone-only condition without HMD against VESAD-based techniques with HMD, we would not be able to tell how much of the difference between conditions was due to the VESAD, and how much was due to headset weight or display performance.

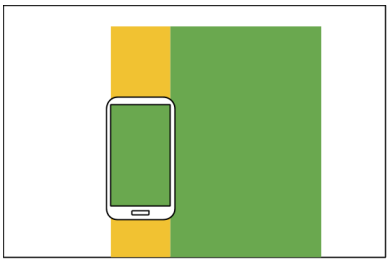


Figure 21: For a right-handed user holding the phone in their left hand, the green zones are suggested as the best ones for input and output, the orange zones are suggested as less convenient but still acceptable for input and output, and the surrounding white zone is suggested for any additional output.

6.1 Design implications

Based on our observations of users and their feedback, we recommend avoiding placing VESAD input elements in areas that require crossing arms. We suggest the zones in Fig. 21 as a guide for laying out user interfaces. These zones avoid crossing of arms during input by the right hand, and the narrow margin below the phone avoids interference with the user's body when the phone is held in a comfortably low position. Of course, the layout would need to be flipped if held in the right hand of a left-handed user. Furthermore, since users sometimes prefer or need to operate their phone with a single hand, there should be a way to access all input elements on the phone's physical screen, either by scrolling through input elements or by switching to a single-handed mode.

We used the layout guide in Fig. 21 to redesign one of our earlier mock-ups (Fig. 22). As on the original mock-up, the hand is rotating the phone ≈90° to access alternative screens, which now show up on the side of the user's right (dominant) hand to facilitate selection of an alternative screen. Notifications appear above the screen, since they are for output only.

Finally, we propose the following guidelines for future VESAD designs:

- Avoid placing input targets on the VESAD far from the phone.
- Avoid interactions or layouts that require users to cross their arms (i.e., where the dominant hand must reach over to the opposite side of the non-dominant hand).
- Allow the VESAD layout to be flipped according to the handedness of the user.
- Anticipate single-handed use scenarios. It should be possible to access all essential input options from the phone's screen, e.g., using a finger or thumb of the hand holding the device.
- Allow both physical and virtual navigation.

7 Limitations

One limitation of VESADs is the requirement of an HMD. Furthermore, to enable a usefully large FOV, we used a video pass-through HMD. Unfortunately, this resulted in feedback latency and reduced angular resolution. In the future, an optical see-through HMD with

Figure 22: Redesign of Fig. 5 based on the zones in Fig. 21 (mock-up).

a wide FOV should be used to eliminate such problems, and we anticipate such hardware eventually becoming widely available, lightweight and convenient to wear. In contrast, smartphones already have mature display hardware, with pixel density often surpassing the human eye's resolving power.

Because we were more interested in the benefits of VESADs with mature HMDs of the near future, our experiment maintained a constant headset weight, feedback latency, angular resolution, and focal distance across conditions by having users always wear the same HMD, even when only interacting with the smartphone. It is possible that eliminating latency or increasing angular resolution would change the results we found, however there are almost certainly tasks that will still benefit from the increased display space offered by a VESAD.

We also only compared techniques for one abstract task. Follow-up work could evaluate VESADs using a variety of realistic tasks, and also compare with peephole displays [42].

The Leap Motion did not allow us to reliably detect 2 fingers for mid-air pinch-to-zoom gestures in the *MidAirInArOut* condition. It is possible that improvements in mid-air finger tracking (for either 1 or 2 fingers) will yield better performance of the *MidAirInArOut* technique, in part by reducing the need for modal buttons.

8 Conclusion

Our experiment in Section 5 found that the *PhoneInArOut* condition was significantly faster and resulted in a significantly smaller selection count (the less conservative of our two metrics of mistakes), and was also significantly preferred over the two other techniques.

Section 3 also presented mock-ups of multiple uses of VESADs, as well as two novel interaction techniques: quickly rotating the phone to switch the VESAD to alternative information (Fig. 5) and the "slide-and-hang" interaction technique (Fig. 7).

9 Future Work

Future work could include improvements to the *MidAirInArOut* technique with better sensing of mid-air finger positions, adding haptic feedback with a hand-worn device to let the user know when they are in contact with the VESAD plane, or comparing direct touch with allowing the user to point at locations on the VESAD using the HMD's view vector (similar to how the HoloLens allows users to aim at targets). A variant of the *PhoneInArOut* technique could also be developed where the phone acts like a touchpad that can reach the entire surface of the VESAD using an appropriate gain or cursor acceleration function.

Acknowledgments

This work was funded by NSERC. Thanks to Alexandre Millette, Alicia Servera, Dylan McGuffin, and Kamea McGuffin for ideas and for help creating mock-ups.

References

[1] C. Andrews, A. Endert, and C. North. Space to think: large high-resolution displays for sensemaking. In *Proc. ACM Conf. Human Factors in Computing Systems (CHI)*, pp. 55–64, 2010.

[2] Y. Bababekova, M. Rosenfield, J. E. Hue, and R. R. Huang. Font size and viewing distance of handheld smart phones. *Optometry and Vision Science*, 88(7):795–797, 2011.

[3] P. Baudisch, N. Good, V. Bellotti, and P. Schraedley. Keeping things in context: a comparative evaluation of focus plus context screens, overviews, and zooming. In *Proc. ACM Conf. Human Factors in Computing Systems (CHI)*, pp. 259–266, 2002.

[4] H. Benko, E. Ofek, F. Zheng, and A. D. Wilson. FoveAR: Combining an optically see-through near-eye display with projector-based spatial augmented reality. In *Proc. ACM Symposium on User Interface Software and Technology (UIST)*, pp. 129–135, 2015.

[5] L. Besançon, P. Issartel, M. Ammi, and T. Isenberg. Hybrid tactile/tangible interaction for 3D data exploration. *IEEE Transactions on Visualization and Computer Graphics (TVCG)*, 23(1):881–890, 2017.

[6] X. Bi and R. Balakrishnan. Comparing usage of a large high-resolution display to single or dual desktop displays for daily work. In *Proc. ACM Conf. Human Factors in Computing Systems (CHI)*, pp. 1005–1014, 2009.

[7] S. Burigat and L. Chittaro. Visualizing references to off-screen content on mobile devices: A comparison of arrows, wedge, and overview+detail. *Interacting with Computers*, 23(2):156–166, 2011.

[8] S. Burigat and L. Chittaro. On the effectiveness of overview+detail visualization on mobile devices. *Personal and ubiquitous computing*, 17(2):371–385, 2013.

[9] A. Cockburn, A. Karlson, and B. B. Bederson. A review of overview+detail, zooming, and focus+context interfaces. *ACM Computing Surveys (CSUR)*, 41(1):2, 2009.

[10] M. Czerwinski, G. Smith, T. Regan, B. Meyers, G. G. Robertson, and G. K. Starkweather. Toward characterizing the productivity benefits of very large displays. In *Interact*, vol. 3, pp. 9–16, 2003.

[11] P. Dragicevic. Fair statistical communication in HCI. In *Modern Statistical Methods for HCI*, pp. 291–330. Springer, 2016.

[12] B. Ens, J. D. Hincapié-Ramos, and P. Irani. Ethereal planes: a design framework for 2d information space in 3d mixed reality environments. In *Proc. ACM Symposium on Spatial User Interaction (SUI)*, pp. 2–12, 2014.

[13] B. M. Ens, R. Finnegan, and P. P. Irani. The personal cockpit: a spatial interface for effective task switching on head-worn displays. In *Proc. ACM Conf. Human Factors in Computing Systems (CHI)*, pp. 3171–3180, 2014.

[14] G. W. Fitzmaurice. Situated information spaces and spatially aware palmtop computers. *Communications of the ACM (CACM)*, 36(7):39–49, 1993.

[15] J. Grubert, M. Heinisch, A. Quigley, and D. Schmalstieg. MultiFi: Multi fidelity interaction with displays on and around the body. In *Proc. ACM Conf. Human Factors in Computing Systems (CHI)*, pp. 3933–3942, 2015.

[16] U. Gruenefeld, D. Ennenga, A. E. Ali, W. Heuten, and S. Boll. EyeSee360: designing a visualization technique for out-of-view objects in head-mounted augmented reality. In *Proc. ACM Symposium on Spatial User Interaction (SUI)*, pp. 109–118, 2017.

[17] Y. Guiard, M. Beaudouin-Lafon, J. Bastin, D. Pasveer, and S. Zhai. View size and pointing difficulty in multi-scale navigation. In *Proc. Advanced Visual Interfaces (AVI)*, pp. 117–124. ACM, 2004.

[18] C. Gutwin, A. Cockburn, and A. Coveney. Peripheral popout: The influence of visual angle and stimulus intensity on popout effects. In

Proc. ACM Conf. Human Factors in Computing Systems (CHI), pp. 208–219, 2017.

[19] A. Hang, E. Rukzio, and A. Greaves. Projector phone: a study of using mobile phones with integrated projector for interaction with maps. In *Proc. International Conf. Human-Computer Interaction with Mobile Devices and Services (MobileHCI)*, pp. 207–216. ACM, 2008.

[20] J. D. Hincapié-Ramos, X. Guo, P. Moghadasian, and P. Irani. Consumed endurance: a metric to quantify arm fatigue of mid-air interactions. In *Proc. ACM Conf. Human Factors in Computing Systems (CHI)*, pp. 1063–1072, 2014.

[21] S. Jang, W. Stuerzlinger, S. Ambike, and K. Ramani. Modeling cumulative arm fatigue in mid-air interaction based on perceived exertion and kinetics of arm motion. In *Proc. ACM Conf. Human Factors in Computing Systems (CHI)*, pp. 3328–3339, 2017.

[22] B. Jones, R. Sodhi, D. Forsyth, B. Bailey, and G. Maciocci. Around device interaction for multiscale navigation. In *Proc. International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI)*, pp. 83–92. ACM, 2012.

[23] B. R. Jones, H. Benko, E. Ofek, and A. D. Wilson. IllumiRoom: peripheral projected illusions for interactive experiences. In *Proc. ACM Conf. Human Factors in Computing Systems (CHI)*, pp. 869–878, 2013.

[24] S. K. Kane, D. Avrahami, J. O. Wobbrock, B. Harrison, A. D. Rea, M. Philipose, and A. LaMarca. Bonfire: a nomadic system for hybrid laptop-tabletop interaction. In *Proc. ACM Symposium on User Interface Software and Technology (UIST)*, pp. 129–138, 2009.

[25] N. Kishishita, K. Kiyokawa, J. Orlosky, T. Mashita, H. Takemura, and E. Kruijff. Analysing the effects of a wide field of view augmented reality display on search performance in divided attention tasks. In *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pp. 177–186, 2014.

[26] H. Lee, D. Kim, and W. Woo. Graphical menus using a mobile phone for wearable AR systems. In *International Symposium on Ubiquitous Virtual Reality (ISUVR)*, pp. 55–58. IEEE, 2011.

[27] C. Liu, O. Chapuis, M. Beaudouin-Lafon, E. Lecolinet, and W. E. Mackay. Effects of display size and navigation type on a classification task. In *Proc. ACM Conf. Human Factors in Computing Systems (CHI)*, pp. 4147–4156, 2014.

[28] T. Ni, D. A. Bowman, and J. Chen. Increased display size and resolution improve task performance in information-rich virtual environments. In *Proc. Graphics Interface (GI)*, pp. 139–146. Canadian Information Processing Society, 2006.

[29] T. Piumsomboon, D. Altimira, H. Kim, A. Clark, G. Lee, and M. Billinghurst. Grasp-shell vs gesture-speech: A comparison of direct and indirect natural interaction techniques in augmented reality. In *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pp. 73–82, 2014.

[30] G. Robles-De-La-Torre. The importance of the sense of touch in virtual and real environments. *IEEE Multimedia*, 13(3):24–30, 2006.

[31] M. Serrano, B. Ens, X.-D. Yang, and P. Irani. Gluey: Developing a head-worn display interface to unify the interaction experience in distributed display environments. In *Proc. International Conf. Human-Computer Interaction with Mobile Devices and Services (MobileHCI)*, pp. 161–171. ACM, 2015.

[32] T. Siu and V. Herskovic. SidebARs: improving awareness of off-screen elements in mobile augmented reality. In *Proc. Chilean Conference on Human-Computer Interaction*, pp. 36–41. ACM, 2013.

[33] H. Sollich, U. von Zadow, T. Pietzsch, P. Tomancak, and R. Dachselt. Exploring time-dependent scientific data using spatially aware mobiles and large displays. In *Proc. ACM International Conference on Interactive Surfaces and Spaces (ISS)*, pp. 349–354, 2016.

[34] R. W. Soukoreff and I. S. MacKenzie. Towards a standard for pointing device evaluation, perspectives on 27 years of Fitts' law research in hci. *International Journal of Human-Computer Studies*, 61(6):751–789, 2004.

[35] W. Steptoe, S. Julier, and A. Steed. Presence and discernability in conventional and non-photorealistic immersive augmented reality. In *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pp. 213–218, 2014.

[36] J. Taylor, L. Bordeaux, T. Cashman, B. Corish, C. Keskin, T. Sharp, E. Soto, D. Sweeney, J. Valentin, B. Luff, et al. Efficient and precise interactive hand tracking through joint, continuous optimization of pose and correspondences. *ACM Transactions on Graphics (TOG)*, 35(4):143, 2016.

[37] D. Wenig, J. Schöning, A. Olwal, M. Oben, and R. Malaka. WatchThru: Expanding smartwatch displays with mid-air visuals and wrist-worn augmented reality. In *Proc. ACM Conf. Human Factors in Computing Systems (CHI)*, pp. 716–721, 2017.

[38] S. White, D. Feng, and S. Feiner. Interaction and presentation techniques for shake menus in tangible augmented reality. In *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pp. 39–48, 2009.

[39] C. Winkler, M. Löchtefeld, D. Dobbelstein, A. Krüger, and E. Rukzio. SurfacePhone: a mobile projection device for single-and multiuser everywhere tabletop interaction. In *Proc. ACM Conf. Human Factors in Computing Systems (CHI)*, pp. 3513–3522, 2014.

[40] R. Xiao and H. Benko. Augmenting the field-of-view of head-mounted displays with sparse peripheral displays. In *Proc. ACM Conf. Human Factors in Computing Systems (CHI)*, pp. 1221–1232, 2016.

[41] R. Xiao, G. Lew, J. Marsanico, D. Hariharan, S. Hudson, and C. Harrison. Toffee: enabling ad hoc, around-device interaction with acoustic time-of-arrival correlation. In *Proc. International Conf. Human-Computer Interaction with Mobile Devices and Services (MobileHCI)*, pp. 67–76. ACM, 2014.

[42] K.-P. Yee. Peephole displays: pen interaction on spatially aware handheld computers. In *Proc. ACM Conf. Human Factors in Computing Systems (CHI)*, pp. 1–8, 2003.

[43] B. Yost, Y. Haciahmetoglu, and C. North. Beyond visual acuity: the perceptual scalability of information visualizations for large displays. In *Proc. ACM Conf. Human Factors in Computing Systems (CHI)*, pp. 101–110, 2007.

[44] S. Zhai, P. Milgram, and W. Buxton. The influence of muscle groups on performance of multiple degree-of-freedom input. In *Proc. ACM Conf. Human Factors in Computing Systems (CHI)*, pp. 308–315, 1996.

[45] C. Zhao, K.-Y. Chen, M. T. I. Aumi, S. Patel, and M. S. Reynolds. SideSwipe: detecting in-air gestures around mobile devices using actual GSM signal. In *Proc. ACM Symposium on User Interface Software and Technology (UIST)*, pp. 527–534, 2014.