AR-enhanced Widgets for Smartphone-centric Interaction

Eugénie Brasier, Emmanuel Pietriga, Caroline Appert

AR-enhanced Widgets for Smartphone-centric Interaction

Eugénie Brasier
Emmanuel Pietriga
Caroline Appert
eugenie.brasier@inria.fr
emmanuel.pietriga@inria.fr
caroline.appert@universite-paris-saclay.fr
Université Paris-Saclay, CNRS, Inria, LISN
Orsay, France

ABSTRACT

We contribute a detailed investigation of AR-enhanced widgets for smartphones, where AR technology is not only used to offload widgets from the phone to the air around it, but to give users more control on input precision as well. Such widgets have the obvious benefit of freeing up screen real-estate on the phone, but their other potential benefits remain largely theoretical. Their limitations are not well understood, most particularly in terms of input performance. We compare different AR-enhanced widget designs against their state-of-the-art touch-only counterparts with a series of exploratory studies in which participants had to perform three tasks: command trigger, parameter value adjustment, and precise 2D selection. We then derive guidelines from our empirical observations.

CCS CONCEPTS

• **Human-centered computing** → **Interaction techniques**; **Empirical studies in HCI**; **Mixed / augmented reality**; **Mobile phones**.

KEYWORDS

augmented reality; mobile phone; widget control; multi-device mobile setup

1 INTRODUCTION

Augmented Reality Head-Mounted Displays (ARHMDs) have capabilities that distinguish them from other mobile devices: they render stereoscopic imagery seamlessly superimposed on what users see in their field of view; and they enable direct manipulation of this imagery via freehand gestures coupled with head- and gaze-tracking. As such, ARHMDs have much potential to complement smartphones and smartwatches, which render imagery on a much smaller physical surface but with a much higher display quality, while also enabling tactile input. Combined together, these different devices are shaping a future where users interact with *Mobile Multi-Device Environments* [17]. In such environments, users can select the best device for a task based on its unique strengths, but they should also be able to use those devices together as one single, powerful and seamless interactive system.

The combination of ARHMDs with other personal devices has opened up a large design space. Of particular interest is the combination of smartphones and ARHMDs, which can be made to work in tandem, as explored at length in the BISHARE design space [44]. For instance, smartphones can be used as high-precision input devices to manipulate spatial AR content (*e.g.*, [31]), while ARHMDs can enlarge a smartphone's display capacity (*e.g.*, [15, 33]) or show 3D content in the air. These are only two examples from a much richer design space, in which many possibilities remain to be explored.

In this paper, we study a suite of AR-enhancements for phone-centric interaction. Acknowledging the strong *legacy bias* associated with smartphones, which makes users resort to well-known interaction styles whenever possible [6, 44], AR-enhanced widgets are augmentations of those traditionally found on this type of device. We make use of AR capabilities to distribute output by offloading widgets "in the air" around the phone,[1] as well as to distribute input between touch and mid-air gestures. Both distribution strategies free up precious screen real-estate on the phone, and limit situations where content gets occluded as well, since users interact partly "around the device" [24, 25]. Additional input degrees of freedom may also help users perform tasks that require precision.

As pointed out in the recent literature, such phone-centric distributed UIs have great *potential*. But not much can be asserted beyond that. While the concept has been briefly discussed among a wealth of other possible AR+phone combinations (*e.g.*, BISHARE [44], VESAD [33], MultiFi [15]), it remains at the stage of rough sketches only, sometimes not actually prototyped. In this paper, we examine a sample of interactions that is representative of UI controls on smartphones (buttons, sliders, and 2D selections), reconsidering their design in the context of an AR+Phone set up. We adopt a UI distribution strategy that consists of offloading widgets in the air when applicable, and making indirect control possible through movements in the air (as a complement or replacement to touch). Figure 1 illustrates the potential advantages of such an approach: 1) more widgets always visible in the air and more space on the phone's screen to display content in high resolution, and 2) indirect control with hand movements in the air to avoid potential occlusion and to enable higher-precision control. We discuss design considerations about applying such a strategy to phone widgets, and report

https://doi.org/10.1145/3447526.3472019

[1] The concept of such widgets actually floating in the air around the phone has been popularized in science fiction shows such as, *e.g.*, *The Expanse*. Querying the Web for `"the expanse hand terminal"` yields artists' renderings that accurately illustrate the idea.

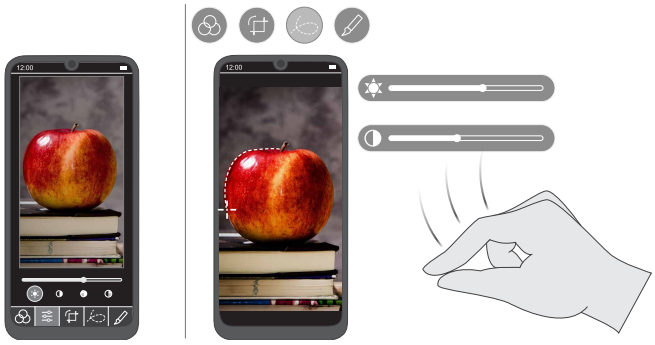


Figure 1: Offloading controls of an image editing application running on a smartphone. Left) UI Widgets take significant screen real-estate when displayed on the phone; and precise selections are difficult with touch input because of finger occlusion. Right) Default UI distribution: 1) *output*: widgets are displayed in the air, freeing screen real-estate on the phone to show more content; and 2) *input*: the pointer can be controlled indirectly with movements performed in the air, causing less occlusion of the image being edited.

on a series of exploratory studies to identify viable designs and estimate their performance.

Our contributions are:

(1) a series of detailed designs for AR-enhanced phone widgets;
(2) an exploratory study of these widgets compared to their state-of-the-art, touch-only counterparts.

2 RELATED WORK

Phone-centric distributed UIs constitute only two cells in a larger design space of *bidirectional interactions between smartphones and head-mounted augmented reality* [44], which is itself only one item in the *cross-device interaction taxonomy* [6]. We refer the reader to these two recently-published and complete articles, and only focus here on work that directly relates to distributed UIs in a mobile context: interacting above and around handheld or wearable devices; augmenting mobile 2D input; multi-device mobile systems involving augmented reality technology.

2.1 Interacting Above and Around Mobile Surfaces

While direct touch remains a very effective way to interact with a mobile device, bare-hand interaction around the device has several interesting properties. It enables a greater physical input volume beyond the small touchscreen [8, 19, 20, 24], sometimes with higher expressiveness [11, 18]. It partially addresses problems of content occlusion [8, 18, 20, 24]. It provides possibilities to lay out widgets beyond the touchscreen [26], or organize multiple pieces of content relative to the user's body and quickly access them [10, 21].

One of the challenges is to accurately track the fingers relative to the device and to segment gestures. While many systems use external motion tracking systems for prototyping purposes [19, 21, 24], some others actually try to embed the tracking solution in the mobile setup. The early SideSight system [8] used IR proximity sensors mounted on the device, tracking finger gestures performed on the surface upon which the device was laid. Skin Buttons [26] also use IR sensors to detect taps on icons projected from a smartwatch on the surrounding skin. Air+Touch [11] uses a miniature depth camera attached to the phone. Similar to SideSight, GlassHands [18] recognizes surface gestures around the phone, but achieves this without additional sensors, by tracking the reflection of the user's phone and hands in her sunglasses with the phone's front-facing camera. Ether-Toolbars [36] explore a similar approach, but uses a reflective surface mounted above the camera. Tracking can also rely on magnetic sensing. Ketabdar *et al.* [25] detect changes in the magnetic field around the phone produced by the circular or linear movements of a finger ring. Nenya [4] also uses a ring, but detects twisting and sliding gestures, which are mapped to selection and click, respectively. Abracadabra [20] has users wear a magnet below their finger to interact around a smartwatch.

One downside of mid-air around device interaction is fatigue [22]. Jones *et al.* [24] report that participants performed equally well with mid-air and with touch interactions, and that they found mid-air interactions enjoyable but tiring as well. Participants also raised concerns about arm fatigue with AirPanes [21]. This suggests that the use of mid-air gestures should be restricted to occasional, non-lasting interactions that complement touch interactions. Air+Touch [11] explores such an approach, interweaving touch events and short in-air gestures. Empirical data about user performance and fatigue is not reported, unfortunately.

2.2 Augmenting Mobile 2D Input

Because we augment 2D widgets, our work also relates to mobile interaction techniques that add more input degrees of freedom to trigger commands or adjust parameter values. Grossman *et al.* track a pen in close proximity to a portable screen, letting users invoke so-called Hover Widgets [14] to trigger commands with small 2D gestures performed *in-situ*. The Hover Cursor [34] tracks fingers hovering the phone's screen, in this case to facilitate the selection of small targets by offsetting the pointer, thus avoiding occlusion of the target of interest. Earlier, Shift [42] had explored a complementary approach to the well-known fat-finger problem, offsetting a copy of the area under the finger when in contact with the screen to keep that region visible and enable precise selection.

The above techniques can be seen as augmenting the capabilities of the screen itself. Other techniques have investigated a more indirect way of augmenting 2D widgets, using the mobile device's side buttons. SidePress [40] uses two buttons on one of the phone's sides, that can sense different pressure levels. The buttons can be used for any bidirectional selection and navigation task. Going further, the Power-up Button [39] combines pressure and proximity sensors to let users control 2D widgets with the thumb holding the device.

2.3 Multi-device Mobile Systems using AR

AR-enhanced widgets for phone-centric interaction are part of the broader category of Distributed User Interfaces in which the system is aware of the relative spatial location of all devices involved: smartphones, tablets, and combinations thereof [16, 27, 35]. These often focus on use cases where one or more users temporarily combine several devices in a stationary context, typically laying them out on a table. Gluey [37] is a conceptual interactive system where devices, which can be in different locations, are combined thanks to an ARHMD, using the latter for tracking and as a means to transfer content between devices.

ARHMDs also enable distributed user interfaces adapted to a mobile context. As mentioned earlier, ARHMDs and mobile touchscreens have very different capabilities, that can complement one another effectively when made to work in tandem, as observed empirically with MultiFi [15]. The BISHARE design space [44] organizes smartphone+ARHMD combinations into two broad categories: *Phone-centric* and *HMD-centric*. The latter category covers cases where the handheld device enhances spatial interaction with AR content. For instance, the handheld device becomes an alternative or a complement to bare-hand input for AR/VR content manipulation [7, 41] and navigation [29], with applications to, *e.g.*, CAD [30] and gaming [31]. The former category rather includes cases where the HMD is used to enhance interaction with the handheld device. Our AR-enhanced widgets fall under this category, and more specifically in subcategories *phone-centric distributed UI* (D2P) and *phone-centric distributed input* (D1P).

Such phone-centric combinations, in which the ARHMD is mostly used to add more input and output capabilities to the phone, have gained attention in the HCI literature. VESAD [33] is an example of augmenting a phone's visual output. The technique virtually extends the phone's screen by displaying additional 2D content in AR around the phone, keeping that content spatially aligned in the same plane. A VESAD proved more efficient than a phone only on a manual classification task. However, Eiberger *et al.* [13] report that the combination of a physical display with AR is not always beneficial. In their study about the integration of information from different depth layers, participants had to perform a visual search task using a combination of HMD and smartwatch, or using an HMD only. Their performance was significantly lower in the HMD+smartwatch case, in terms of both speed and error.

In the above examples, the AR imagery consists of an expansion of what is shown on the mobile device. But AR can also show complementary information of a different nature, as illustrated by WatchThru [43] and mobile true-3D displays [38]. MultiFi [15] goes one step beyond, making interactive widgets run across devices and ARHMD, adapting both their input and output based on the respective fidelity of devices involved: screen resolution and physical size, direct *vs.* indirect input. For instance, the smartphone's high-resolution screen can be used as the primary display for an item in focus, while the ARHMD will provide lower-resolution information about surrounding items in the same list. Or the phone's touchscreen can be turned into a fullscreen keyboard, the edited text being shown in mid-air. MultiFi essentially focuses on how to make widgets span devices with different input/output characteristics. Interaction remains limited to basic touch and spatial pointing. Finally, VESAD can also be used to show additional windows or widgets around the phone. The authors briefly suggest this possibility [33], but report neither on the prototyping nor the evaluation of such AR-based enhancement techniques.

Extending interactive surfaces to the air can involve input (*e.g.*, mid-air movements), output (*e.g.*, AR imagery) or both – though they have so far been mostly studied separately. Such strategies are promising research avenues. However, the actual, quantifiable pros and cons of applying these extension strategies to distribute UI phone controls in an actual ARHMD+phone set-up have not been explored. We discuss their design, implementation and performance aspects individually and, where possible, in combination with a representative set of phone controls.

3 AR-ENHANCED WIDGETS: DESIGN CONSIDERATIONS

We first identified three low-level, generic interactions[2] typically performed using widgets that could be at least partially offloaded to the air, either their input, their output, or both: command trigger/item selection; parameter value adjustment; and precise 2D selection. We then prototyped different AR-enhanced techniques, discarding those that we perceived as clearly ineffective. Among the retained techniques, some raised specific questions or required some parameters to be fine-tuned empirically, in which case we ran preliminary experiments to inform these choices. This section reports on the main considerations involved in making design and implementation decisions.

3.1 Phone-centric Interaction Space

One of the first decisions we had to make was to bound the input space around the phone. In HMD-centric distributed UIs [44], the phone acts as a controller for AR content that can be anywhere in the user's physical environment. It can be tethered to the phone, or free-floating. In phone-centric distributed UIs, the phone holds the primary content. We thus focus on the direct surroundings of the phone, considering widgets and input areas tethered to the phone.

We adopted the rule of thumb that interaction should take place *not further than 10cm away from the phone*. We found that this distance allowed for comfortable posture and movements, especially when the user's upper arms are along their trunk. It avoids large arm movements in the air, which can be both tiring [22] and

[2]Navigation interactions (scrolling, panning, zooming) represent another category, that usually does not involve widgets on a smartphone and has been investigated elsewhere (see, *e.g.*, [24]).

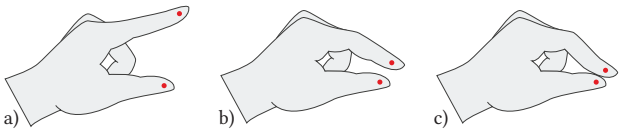


Figure 2: The interpretation of users' hand movements in the air depends on the distance Δ_{TI} between the thumb's tip and the index finger's tip. a) `Out of Range` (Δ_{TI} > 4cm): hand movements have no effect; b) `Tracking` (2cm < Δ_{TI} < 4cm): hand movements control which widget is in focus; c) `Dragging` (Δ_{TI} < 2cm): hand movements actually control the widget in focus.

socially awkward [1]. It also avoids large eye movements and limits problems of divided attention between the phone and the AR display.

3.2 Legacy Bias

Past experience with interactive systems influences the way users apprehend a novel system, making them expect the novel system to work the same way as the systems they are familiar with. Brudy *et al.* [6] warn interaction designers about this so-called *legacy bias* [32] in the context of cross-device interaction. This led us to consider how people interact with smartphones, and find ways to *augment* their experience rather than radically change it. The three generic interactions we identified have a straightforward correspondence with UI widgets on smartphones:

- command trigger and item selection are typically achieved with buttons, checkboxes, or menus;
- parameter value adjustments are typically achieved with sliders;
- precise 2D selection is achieved by pointing at a precise location inside widgets such as images, text areas, maps.

Keeping the issue of legacy bias in mind, we favor mid-air interactions which are as close as possible to their touch-only counterparts, striving for a unified set of interaction techniques across modalities. In our design, offloading widgets from the phone to the air does not noticeably change their appearance. Interaction with these widgets is directly inspired by the paradigm developed for the Microsoft Hololens 2 for hand input. Users *touch* discrete widgets with their index tip, and perform continuous input by adopting a *pinch* posture and moving their hand (air-tap-and-hold). These can be seen as the mid-air counterparts to *tap* and *slide* interactions on the phone's screen.

3.3 Three-state Mid-air Input

Touch input is direct, with focus implicitly given to the widget being touched. Mid-air input is more indirect. It calls for a three-state model that will enable the selection of a widget before it gets manipulated. Figure 2 illustrates the different hand postures that we use to distinguish the three states from Buxton's model of graphical input [9]: `Out of Range`, `Tracking` and `Dragging`. This is a refined version of the air-tap-and-hold interaction implemented on Microsoft HoloLens devices, where the system only supports two states based on hand posture: *pinch* for `Dragging` and *release* for `Out of Range`. We introduce an intermediate posture, *hover*, to enable the third state (`Tracking`) triggered when the thumb and index fingers are close to – but not touching – one another (Figure 2-b).

When adopting a *pinch* or *hover* hand posture, users control an invisible pointer located halfway in-between the thumb's and index finger's tips. The *hover* posture is used to select which widget has focus. When the pointer enters a widget, that widget gets highlighted with a blue halo. If the user brings his fingers closer to actually adopt a *pinch* posture, the widget will then own focus and subsequent hand movements will control it. A widget that already has focus can be controlled by initiating an air-tap-and-hold interaction either on the widget itself or anywhere in empty space.

This third state also enables us to provide users with a straightforward way to customize how the UI is distributed around and on the phone. An air-click event[3] immediately followed by an air-tap-and-hold interaction[4] actually grabs the highlighted widget. Users are then free to drop it wherever they want by releasing their *pinch* posture. This works both for widgets displayed on the phone and widgets that are already in the air.

3.4 Mid-air Input Accuracy

The three-state model above addresses the issue of expressive power, but it does not address that of input accuracy. While indirect mid-air input can contribute to improving accuracy (less occlusion, larger widgets), the absence of physical support and haptic feedback mean that movements are typically less precise (because of, *e.g.*, hand tremor). Furthermore, from a design perspective, movements in the air need to be segmented in order to isolate control gestures from other movements. With the three-state model, we rely on the *pinch* posture for segmenting continuous input from other movements. But releasing control necessarily entails relaxing this *pinch* posture. As opposed to lifting up a finger from a touchscreen, relaxing a posture is a movement, which thus cannot be detected immediately. The system needs to collect a sufficient portion of the relaxing movement before actually releasing control, which inevitably causes an involuntary movement of the cursor. However, since cursor control is indirect with mid-air input, we can account for such inaccuracy by adjusting the CD gain. Setting it to a large-enough value, involuntary movements will not cause the cursor to move.

We ran a preliminary experiment involving six participants to quantify the amplitude of such involuntary movements, which we identify as $\Delta_{release}$. Participants controlled a cursor on the phone with mid-air movements. They had to put it precisely on a very small target (1.5mm), and then switch from a *pinch* to a *release* posture. We measured $\Delta_{release}$, the offset between the target position and the cursor's actual position at the time the system detects this change of hand posture. The design, procedure and results are detailed in supplemental material. Averaging all collected values, we obtain $\Delta_{release} = 1.4 \pm 2.2 < 4mm$.

We use this 4-mm maximum bound for $\Delta_{release}$ in the design of interactions that involve mid-air input by enabling users to adjust the CD gain up to $4mm : cursorUnit$ so that the inaccuracy inherent

[3] An air-tap event followed by an air-release event less than 500ms later.
[4] The air-tap event should happen less than 500ms after the last air-click event.

to the segmentation of air movements will be compensated for, at least when the CD gain is maximum.

3.5 Effective use of the Air as a Display

The above experiment was coupled with another one aimed at gathering empirical data about users' widget placement preferences. Indeed, widgets can be placed anywhere in the air around the phone. But some sides will generate occlusion, or require adopting uncomfortable postures, depending on which hand is holding the phone. In this second preliminary experiment, participants had to interact with buttons and sliders displayed in the air on each of the phone's four sides. The design, procedure and results are detailed in supplemental material.

Overall, a majority of participants (5) ranked the right side as their preferred location for interacting with *Sliders*, the top side coming second. Results were more nuanced for *Buttons*. Half of the participants ranked the top side first, and the other half the right side. The area below the phone requires users to adopt uncomfortable postures, at least with the limited vertical field of view of current ARHMDs. Our findings are consistent with the recommendations from Normand and McGuffin [33], who report that some study participants had trouble interacting with the AR content around the phone when they had to cross their arms (for instance when they had to interact with virtual elements on the phone's left side using their right hand).

4 AR-ENHANCED WIDGETS: DESIGN AND COMPARATIVE EVALUATION

We designed AR-enhanced widgets that cover a representative sample of phone interactions: *Command trigger* (E_1), *Parameter adjustment* (E_2), and *2D selection* (E_3). For each of these interactions, we designed several variations of the corresponding widget: one that offloads the widget to the air, one that extends the input space to the air, one that does both. Specific design choices were guided by the considerations detailed in the above section. We systematically compared those designs against state-of-the-art touch-only input techniques in an exploratory study.

We conducted one study per interaction. The three studies were conducted sequentially, always in this order. Participants had to take a break of at least 15 minutes between them. During these breaks, they were asked to rate each technique along all six NASA-TLX dimensions, and then to rank the techniques according to their preferences.

Participants and Apparatus. 12 participants (5 women, 7 men, 32 ± 11.1 year-old on average), all right-handed, took part in the study.[5] They were seated on a chair without armrest. We used a Samsung Galaxy A50 and a Microsoft HoloLens 2. The two devices communicate on a dedicated wireless network using TCP. We track the phone's location with Vuforia,[6] using three markers offset by 8cm from the phone's top edge, as illustrated in Figure 4.

Study data are available as supplemental material online at https://ilda.saclay.inria.fr/ARwidgets4phones/

4.1 Experiment E_1: Command Trigger

In this first experiment, participants are asked to push buttons, either on the phone or in the air around the phone. We describe techniques based on where the output (O) and input (I) take place, respectively. Technique *Screen - Screen*, our baseline, displays buttons on the phone and expects touch input. Technique *Air - Air* displays buttons in the air and expects direct input in the same place.[7]

Offloading buttons to the air has the benefit of freeing screen real-estate on the phone. But the lack of haptic feedback with air buttons will likely make input more difficult. Furthermore, many users have a lot of prior experience with traditional touch buttons. However, we do not know how much this impacts user performance with air buttons.

Task. Three circular buttons, horizontally aligned, with a spacing of 3mm between them, are displayed either 5mm below the screen's top edge (*Screen - Screen*), or 5mm above it (*Air - Air*). Button diameter varies according to task Difficulty: 2cm (*Low*), 1.5cm (*Medium*) or 1cm (*High*). When a button gets highlighted, participants have to select it as fast and as accurately as possible. Buttons in a sequence are always highlighted in the same order: left, right, middle.

Design. We follow a within-subject design with two factors: Technique ∈ {*Screen - Screen*, *Air - Air*} and Difficulty ∈ {*Low*, *Medium*, *High*}. Trials are blocked by Technique. Block presentation order is counterbalanced across participants with a Latin Square. Each Technique block consists of 12 trials, *i.e.*, four replications per Technique × Difficulty. The presentation order of those 12 trials is counterbalanced across blocks and participants with a Latin Square. Each block starts with three practice trials presented in order of increasing difficulty.

Results. Our main measure is the *Time* between two button activations. We filter out interactions with the first (leftmost) button in a sequence as there is no control over the initial position of the participant's hand. We observe a large difference between techniques, with average time 0.4±0.25s for *Screen - Screen* and ∈ 1.2±0.8s for *Air - Air*.

Before conducting any analysis, we check that the collected data follow a normal distribution with a Shapiro-Wilk test. As we observe a violation of normality in one of the conditions (*Screen - Screen* × *Medium*, $p < 0.05$), we log-transform our measure data to avoid having a strong violation of the normality test.[8] An analysis of variance (anova) of Technique × Difficulty on *Time* reveals a

[5] **About COVID-19 and experiments**: all experiments reported in this article were run during the summer of 2020. Throughout all studies, we recruited participants from a very restricted pool of 12 volunteers: six relatives of the first author who live in the same house, and three volunteer co-workers. The sanitary protocol involves hand disinfection, device cleaning, wearing hair protection under the ARHMD and a FFP2 safety mask.

[6] https://developer.vuforia.com

[7] A button is a graphical component that implements the `HandInteractionTouch` script from the Microsoft Mixed Reality Toolkit (MRTK). To trigger an event, the index finger's tip must enter and then leave the component.

[8] A quick sanity check shows that significant effects and differences are overall the same when running the anova and post-hoc tests on the non-corrected *Time* measure. The log correction tends to make differences more nuanced but, overall, the general observations remain the same.

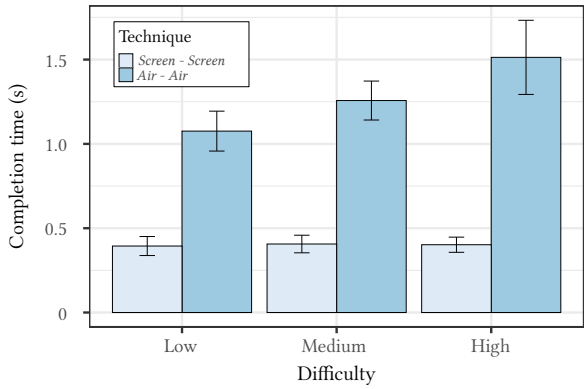


Figure 3: Command Trigger experiment (E_1): completion time by Difficulty × Technique. Error bars represent 95% confidence intervals.

simple effect of both Technique ($F_{1,11} = 181$, $p = 0.001$, $\eta^2_G = 0.9$) and Difficulty ($F_{2,22} = 12$, $p = 0.001$, $\eta^2_G = 0.13$). *Screen - Screen* is significantly faster than *Air - Air* (0.4s vs 1.28s on average), and the three levels of difficulty are all different from each other (p's < 0.05).

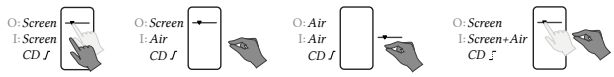
Comparing the techniques' relative performance across participants also reveals a high variability with, *e.g.*, one participant performing 4.82 times better with *Screen - Screen* than with *Air - Air*, and another one 1.88 times only. Interestingly, the participant whose relative performance difference between techniques is the smallest has strong experience with AR headsets. This suggests that users might be able to trigger buttons in the air relatively quickly, but this seems to require some training, and even then performance with on-screen buttons remains significantly better.

Qualitative assessments based on NASA-TLX scales are in line with quantitative observations: average grades on all scales are in favor of *Screen - Screen*. Wilcoxon signed-rank tests reveal that all these differences between techniques are significant (p's < 0.05 and $Z's \in [-2.6, -2.4]$), with the exception of the *frustration* scale ($p = 0.6$). Two participants found interacting in the air *fun* (2 participants), but participants mostly mentioned the advantages of interacting on screen, such as *high reliability* (6 participants) and *haptic feedback* (5). They also reported that touch requires *low amplitude movements* (8) compared to air input, which requires traversing the button and then exiting it with movements that are large-enough to be tracked reliably. Finally, only two participants preferred *Air - Air* over *Screen - Screen*.

Overall, *Screen - Screen* outperforms *Air - Air* in terms of both completion time and comfort. These results are not in favor of offloading phone buttons for activating discrete commands.

4.2 Experiment E_2: Parameter Adjustment

In the second experiment, participants had to select a numerical value using a slider. We include high precision tasks which involve value ranges that exceed typical smartphone screen resolution – causing quantization problems [2] – and users' motor abilities in terms of fine-grained control. The touch-only baseline, *Screen - Screen* (*CD* ⌐), implements OrthoZoom control [3]. This enables participants to perform very precise adjustments, with sub-pixel accuracy. We compare this baseline to three AR-enhanced slider designs (see also Figure 4) that vary in where output (O) and input (I) take place, and in how users control input precision (CD):



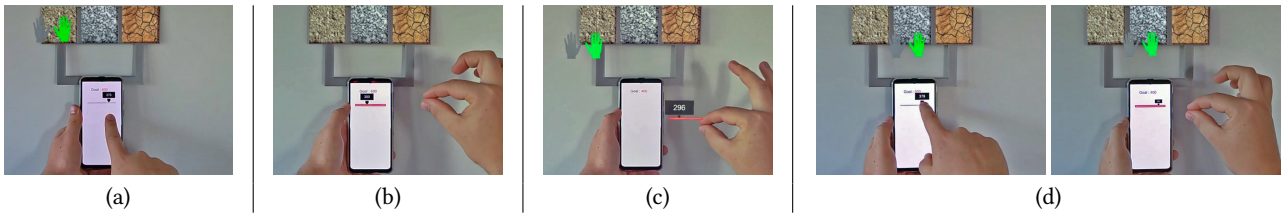
Screen - Screen **(***CD* ⌐**).** Users interact with OrthoZoom sliders as they do with traditional touch sliders. The only difference is that input precision increases progressively (⌐) as the orthogonal distance between the slider and the finger increases. CD gain is set to 1:1 when the finger is less than 2cm away from the slider's track, and linearly increases up to 1:$sliderUnit_{mm}$ at 10cm from the slider, with $sliderUnit_{mm}$ being the size of a slider unit in mm. Above this maximal distance, a 1mm-displacement thus corresponds to a 1-unit adjustment on the slider.

Screen - Air **(***CD* ⌐**).** The slider is still displayed on the phone, but input takes place in the air. The slider is manipulated with air-tap-and-hold interactions (Section 3.3). As with OrthoZoom, input precision is a function of the distance between the fingers' location when the interaction started and the current fingers' location. But in this case, the distance considered is different: precision increases progressively (⌐) as the hand moves away from the phone's plane towards the user's head. CD gain is set to 1:1 when that distance is short (<2cm), and linearly increases up to 4: $sliderUnit_{mm}$ when it reaches 10cm. Above this 10-cm distance, the hand movement must be of at least 4mm to change the slider's value. As discussed in Section 3.4, this gives the tolerance required to enable users to release mid-air pinch postures safely without changing the slider's value. The idea of gaining precision for sliders and scrollbars as the distance in the air increases was mentioned in previous work about tabletop interaction [11, 28]. But to our knowledge, it has not actually been tested. Harrison and Hudson [20] explore a similar idea for radial selection on a smartwatch, but do not report how CD gain is adjusted, and study their technique in a very different situation, where participants used an external hardware button attached to a table to validate selections.

Air - Air **(***CD* ⌐**).** The slider is displayed in the air, on the right-hand side of the phone. Interaction is exactly the same as with *Screen - Air* (*CD* ⌐).

Screen - Screen+Air **(***CD* ⌐**).** The slider is displayed on the phone. Input is hybrid, taking place both on the screen and in the air. Users perform coarse adjustments with slide interactions. They can also perform fine adjustments with air-tap-and-hold interactions. There are only two levels of precision (symbol ⌐ means *dual-precision*). CD gain is set to 1:1 for finger slides on the touch-screen, and to 4: $sliderUnit_{mm}$ for movements in the air.

Design properties. Each of these four designs has its own advantages relative to the others, but it is difficult to infer their overall performance ranking. First, screen input benefits from haptic feedback, and from the strong experience users have acquired with touch devices. This might be an advantage in favor of *Screen - Screen* (*CD* ⌐). Second, some designs are consistent regarding where input and output take place, while others are not. Consistency might be preferable and thus be an advantage for *Screen - Screen* (*CD* ⌐) and *Air - Air* (*CD* ⌐). But, at the same time, their control requires



(a) (b) (c) (d)

Figure 4: The four techniques tested in Experiment E_2: a) *Screen - Screen* (*CD* ⎍), b) *Screen - Air* (*CD* ⎍), c) *Air - Air* (*CD* ⎍), d) *Screen - Screen+Air* (*CD* ⎎), performing a coarse selection on the touchscreen (left), then a precise adjustment in the air (right). The two hand icons provide basic feedback about HoloLens left & right hand tracking status and never interfere with widgets displayed in the air.

continuously adjusting two degrees of freedom (slider value, CD gain). They might thus be more cognitively demanding than a clear distinction between only two levels of precision (coarse and fine). In that respect, *Screen - Air* (*CD* ⎍) might have some advantages over the other three techniques. In the end, these pros and cons make it difficult to know the cost of offloading input, output, or both in the air for slider control.

Task. A 5cm horizontal slider is displayed on the phone for all *Screen - ∗* designs (11cm above the bottom edge of the screen) and in the air for the *Air - Air* design (offset 2cm from the phone's right edge). Depending on task Difficulty, the slider has 100 (*Low*), 500 (*Medium*) or 1000 (*High*) units. The value to select, Goal, is displayed in the upper part of the phone. Participants have to select the Goal as fast as possible. The trial ends as soon as participants select the goal value (no finger on screen, hand in *release* posture). The slider knob's size depends on the current CD gain (half its default size at maximum precision), providing feedback to users about input precision.

Design. We follow a within-subject design with two primary factors: Technique ∈ {*Screen - Screen* (*CD* ⎍), *Screen - Air* (*CD* ⎍), *Air - Air* (*CD* ⎍), *Screen - Screen+Air* (*CD* ⎎)} and Difficulty ∈ {*Low*, *Medium*, *High*}. Trials are blocked by Technique. Block presentation order is counterbalanced across participants with a Latin Square. We introduce a Goal factor that can take four possible values for ecological purposes. It corresponds to the value participants are instructed to select. Each Technique block consists of 12 trials, *i.e.*, one per Technique × Difficulty× Goal. Presentation order of trials within a block is counterbalanced across blocks and participants with a Latin Square. Each block starts with three practice trials, presented in order of increasing difficulty.

Results. We first check the normality of the collected task completion *Time* measures with Shapiro-Wilk tests. We observe that the data are not normally distributed under a few conditions (*Screen - Screen+Air* (*CD* ⎎) × *Low*: $p < 0.001$, *Screen - Screen+Air* (*CD* ⎎) × *Medium*: $p < 0.05$ and *Screen - Screen* (*CD* ⎍) × *Medium*: $p < 0.05$). Applying a log-transformation solves the issue. We thus perform our analyses on a log-transformed time measure. An anova of Technique and Difficulty on *Time* reveals a simple effect of both Technique ($F_{3,33} = 23$, $p = 0.001$, $\eta^2_G = 0.5$) and Difficulty ($F_{2,22} = 118$, $p = 0.001$, $\eta^2_G = 0.6$). Unsurprisingly, all levels of Difficulty significantly differ from each other: completion time increases as the task gets more difficult (Figure 5-a). Regarding differences between techniques, pairwise t-tests reveal that all techniques significantly differ from each other with all p's < 0.01, except for *Screen - Screen+Air* (*CD* ⎎) and *Screen - Screen* (*CD* ⎍), which have a p-value close to 0.05 ($p = 0.04$).

Sorting techniques by performance, we get: *Screen - Screen+Air* (*CD* ⎎) (6.25s), *Screen - Screen* (*CD* ⎍) (6.65s), *Air - Air* (*CD* ⎍) (8.64s), *Screen - Air* (*CD* ⎍) (11.1s). Compared to experiment E_1, the drop in performance when offloading input and output in the air is much lower with sliders than with buttons. Interestingly, *Screen - Screen+Air* (*CD* ⎎) actually performs slightly better than *Screen - Screen* (*CD* ⎍), the OrthoZoom touch-only baseline. This suggests that air control can even be better than touch for indirect control. However, this is only the case when air control is implemented according to a dual-precision strategy. On the opposite, continuous precision level control is less efficient in the air than on screen.

Qualitative assessment. We perform Wilcoxon signed-rank tests to compare pairs of techniques along each of the NASA-TLX scales. The difference between *Screen - Air* (*CD* ⎍) and *Air - Air* (*CD* ⎍) is systematically significant, except for *physical demand*. Participants' comments reflect this preference for *Air - Air* (*CD* ⎍) over *Screen - Air* (*CD* ⎍): six of them spontaneously said that they *like* interacting in the air; and five of them reported having issues of *divided attention* when output is on the phone and input in the air, *i.e.*, when there is a mismatch between where input and where output take place. *Screen - Air* (*CD* ⎍) and *Screen - Screen* (*CD* ⎍) also differ significantly, with ratings systematically in favor of *Screen - Screen* (*CD* ⎍) on all scales except *mental demand*. Prior experience actually using an equivalent of *Screen - Screen* (*CD* ⎍) was reported by three participants and might explain this.

Screen - Air (*CD* ⎍) and *Screen - Screen+Air* (*CD* ⎎) also differ significantly along all scales but *physical demand* and *effort*. Users found it easier to switch between two levels of precision with *Screen - Screen+Air* (*CD* ⎎), as opposed to the continuous precision adjustments of the other (*CD* ⎍). The former strategy (*CD* ⎎) also avoids some usability problems. One issue raised by four participants about continuous CD gain adjustment (*CD* ⎍) relates to clutching. When releasing a pinch posture in mid-air to immediately initiate a new one (*e.g.*, to avoid leaving the tracking volume or to adopt a more comfortable position), CD gain is reset to its default value. These participants commented that they would have preferred the system to set the CD gain to its last value before release instead. Implementing such behavior is not trivial

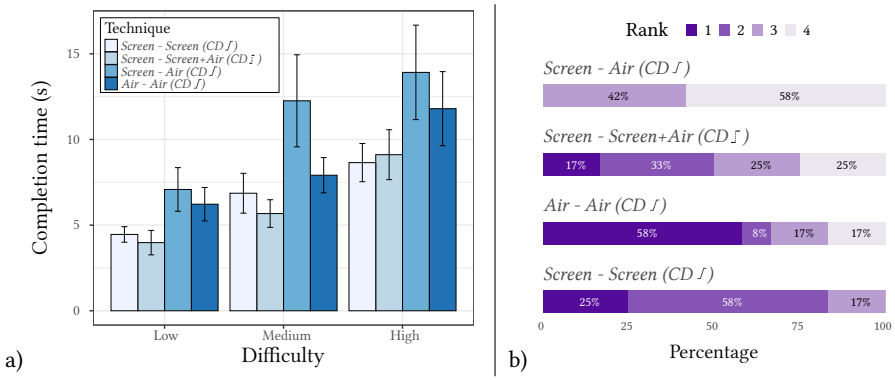


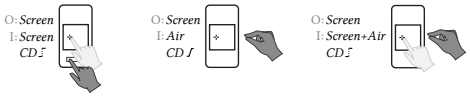
Figure 5: Parameter Adjustment experiment E_2. a) Completion time by Difficulty × Technique. Error bars represent 95% confidence intervals. b) Participants' preferences. Each bar represents the rank distribution for one technique.

however, as it requires distinguishing between clutch actions and actual successive-but-distinct uses of the same widget (for which the current behavior makes more sense). It also raises usability issues, such as the higher chance for the phone to *"be in the way"* of the user's hand if resuming slider manipulation too close to the screen. Interestingly, clutching issues are not observed only for air control. Three participants reported sometimes reaching the phone's left or right edge when seeking a high level of precision with OrthoZoom, which forced them to clutch (*i.e.*, releasing control and resuming on the slider's track itself).

Finally, seven participants reported that the switching cost between touch and air is high. Overall, participants' ranking of techniques is not completely in line with actual performance. As Figure 5-b shows, preferences are split and the fastest techniques are not systematically the preferred ones. In particular, *Screen - Screen+Air* (*CD*) performed best in terms of completion time, but only 17% of participants ranked it first. The novelty effect probably played in favor of *Air - Air* (*CD*), which got ranked first by more than half of the participants. On the contrary, there is a clear consensus against *Screen - Air* (*CD*).

4.3 Experiment E_3: 2D Selection

In the third experiment, participants have to point at a precise location in a 2D workspace. Output (O) necessarily takes place on the phone, as the display resolution of current ARHMDs is too low to render 2D targets small-enough for our purposes. We thus focus on where input (I) takes place. Mid-air input has the potential to offer an occlusion-free alternative to touch for precise pointing tasks such as, *e.g.*, positioning a caret in a text or delineating a region in an image or a map. But it will also likely suffer from significant precision issues. We compare input performed in the air to a touch-only technique and to a hybrid technique:



Screen - Screen (*CD*) serves as our baseline condition. Direct touch in the 2D canvas lets users position the pointer coarsely. They can then control it indirectly but much more precisely (higher CD gain) by initiating a slide gesture from a smaller rectangular area at the bottom of the screen (2cm above the edge). This is a direct adaptation of a technique available on Android and iOS smartphones for caret positioning, invoked by dwelling on the soft keyboard's space bar or by applying a higher force level anywhere on the keyboard (with pressure-sensitive screens).[9] Our adaptation behaves differently though, changing the CD gain to enable very high precision pointing. The original technique found on most smartphones keeps a 1:1 CD gain when entering indirect pointing mode.[10] Thus, it does address finger occlusion issues, but it does not enable high-precision pointing. Our implementation is a dual-precision version (*CD*) which sets CD gain to 1:$precision_{mm}$ when entering indirect mode, where $precision_{mm}$ represents the size of the smallest element that users might want to acquire. In this experiment, we set it to 0.5mm, meaning that a 1mm finger displacement moves the pointer by 0.5mm.

Screen - Air (*CD*) is similar to *Screen - Air* (*CD*) for sliders (in Experiment E_2), but enables 2D selection. Users control the cursor with air-tap-and-hold interactions. Precision gets higher as the hand moves away from the start position of the air-tap-and-hold gesture. The CD gain is set to 1:1 when the distance is low (<2cm), and progressively increases up to 4: $precision_{mm}$ at a distance of 10cm and above. In our study, users are able to point with sub-millimeter precision ($precision_{mm}$ = 0.5mm). This technique implements indirect 2D input with movements in the air [5]. It is conceptually close to the 1-Button-Simultaneous technique from Jones *et al.*'s study [24], which compared different designs for pan & zoom navigation using mid-air gestures around the phone. Although the three input dimensions are not exactly the same in the two studies, their technique and ours are both based on integral and continuous input in a 3D mid-air volume, with movements in the x-y plane for controlling a 2D space (pan *vs.* cursor position) and movements along the z-axis for *scaling* this 2D input (zoom *vs.* CD gain).

[9] We set the dimensions of this area according to the dimensions of the space bar on the phone used for the experiment (31.77mm×7.75mm), offsetting it 1cm upward to make room for downward movements.

[10] Some smartphones implement a transfer function that makes the CD gain decrease as the finger accelerates to enable coarse adjustments from this mode.

Screen - Screen+Air **(*CD*)** is similar to *Screen - Screen+Air* (*CD*) for sliders, but enables 2D selection. Users perform coarse adjustments with direct touch input, and fine-grained adjustments with air-tap-and-hold interactions. Precision-level control is dual (*CD*), with the CD gain set to 1:1 for touch input on the phone's screen, and to 4: $precision_{mm}$ for mid-air input.

Design properties. Again, each of these designs have supposed advantages, but it is difficult to anticipate their comparative performance. The difference between *Screen - Screen* (*CD*) and *Screen - Screen+Air* (*CD*) mostly lies in the input modality. *Screen - Screen* (*CD*) might benefit from touch input properties such as haptic feedback and prior experience. But at the same time, in the above-mentioned study by Jones *et al.* [24] the 1-Button-Simultaneous – which resembles *Screen - Air* (*CD*) – performed as well as multi-touch gestures did. This latter result suggests that offloading the control in the air with a continuous control strategy might not have a high cost. The dual-precision control in *Screen - Screen+Air* (*CD*) could make this cost even lower considering how it improved air control for parameter adjustment in experiment E_2.

Task. Participants have to successively acquire eight circular targets laid out on the phone's screen following the ISO 9241-9 standard [23]. Depending on task Difficulty, target diameter is 4mm (*Low*), 2mm (*Medium*) or 1mm (*High*). The next target to acquire is colored black, while other targets are colored grey. As targets can be very small, the background turns green when the crosshair cursor is inside the target. As soon as participants select the black target (no finger on screen, hand in *release* posture), the next target turns black. Cursor size is inversely proportional to the current CD gain, providing participants with a rough indication of its value, as in Experiment E_2.

Design. We follow a within-subject design with two primary factors: Technique ∈ {*Screen - Screen* (*CD*), *Screen - Screen+Air* (*CD*) and *Screen - Air* (*CD*)} and Difficulty ∈ {*Low*, *Medium*, *High*}. Trials are blocked by Technique. Block presentation order is counterbalanced across participants with a Latin Square. Each trial is a series of eight pointing tasks, and is replicated twice. A Technique block consists of 6 trials (*i.e.*, 6×8 pointing tasks). Presentation order for trials within a block is counterbalanced across blocks and participants with a Latin Square. Each block starts with three practice trials, presented in order of increasing difficulty.

Results. Before running analyses, we filter out the first pointing task in each series of eight, as the initial cursor position is not controlled when a series starts. We additionally remove one outlier that corresponds to a trial during which the HoloLens actually stopped working for a few seconds. Shapiro-Wilk tests do not reveal any violation of normality. An anova of Technique and Difficulty on *Time* reveals a simple effect of both Technique ($F_{2,22} = 42$, $p = 0.001$, $\eta^2_G = 0.65$) and Difficulty ($F_{2,22} = 39$, $p = 0.001$, $\eta^2_G = 0.41$), as well as an interaction effect ($F_{4,44} = 6$, $p = 0.001$, $\eta^2_G = 0.14$). As illustrated in Figure 6, the difference between *Screen - Screen* (*CD*) and *Screen - Screen+Air* (*CD*) gets larger as difficulty increases, and is not even significant when Difficulty = *Low* ($p = 0.7$). The third design, *Screen - Air* (*CD*), performed very poorly in comparison with the other two. Sorting techniques by performance, we get: *Screen - Screen* (*CD*) (1.67s), *Screen - Screen+Air* (*CD*) (2.53s), *Screen - Air* (*CD*) (6.18s). Offloading input to the air has a performance cost. But this cost is limited if air control is implemented with a dual-precision mode, and could be counterbalanced by the advantages that air control has over touch: no occlusion, and no need to reserve space on screen for toggling between precision modes.

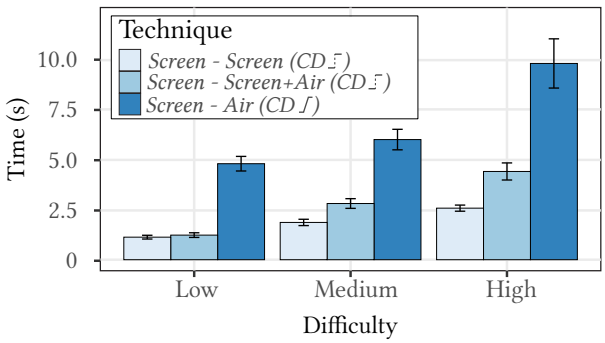


Figure 6: 2D Selection experiment E_3. Completion time by Difficulty × Technique. Error bars represent the 95% confidence interval.

Qualitative assessment. Quantitative measures match participants' qualitative evaluation of techniques, in which they systematically rated *Screen - Air* (*CD*) as significantly worse than the other two along all NASA-TLX scales, with the exception of *Screen - Screen+Air* (*CD*) for *Mental Demand* ($p = 0.07$). This poor performance of *Screen - Air* (*CD*) is consistent with the findings from Experiment E_2. It again proved difficult to control position and input precision with a movement in the air while the output was on phone. Interestingly, this is different from what Jones *et al.* [24] observed. Several elements might be at play. First, their technique was used for multi-scale navigation, which affects the whole graphical scene rather than just the cursor's position and size. The stronger visual feedback associated with their task might be more effective for continuous control. Second, zoom control is relative in their case, while precision control is absolute with *Screen - Air* (*CD*). As discussed in E_2, implementing relative control when clutching is not trivial, a problem which does not exist in the case of multi-scale navigation. One last difference is about the delimiter for mid-air gestures. Jones *et al.* rely on a physical button on the phone, which might be faster than the *pinch* posture that we use. However, participants in their study also complained about this physical button delimiter which they found uncomfortable to use.

Finally, participants' preferences are distributed across *Screen - Screen* (*CD*) and *Screen - Screen+Air* (*CD*) with eight participants ranking *Screen - Screen* (*CD*) first, the other four preferring *Screen - Screen+Air* (*CD*). They consistently ranked *Screen - Air* (*CD*) third.

5 SUMMARY OF FINDINGS

We compared different distributed UIs for three tasks that are representative of touch interaction with smartphones. While we observe an input performance drop with air controls in many cases, this

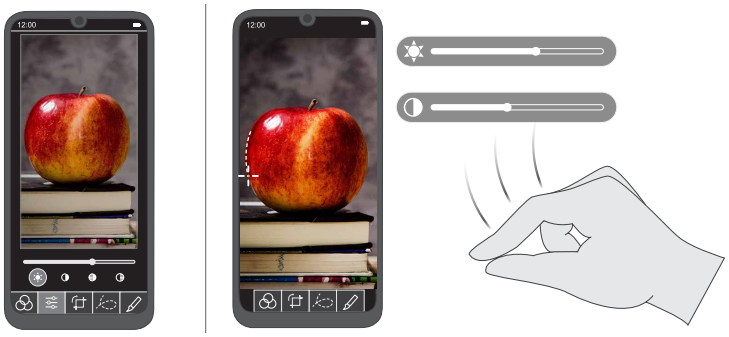


Figure 7: Redesign of the image editing application from Figure 1 following the guidelines derived from our empirical results. Buttons remain on the smartphone. Sliders for parameter adjustments performed frequently get offloaded. 2D selection can be performed directly on the phone, or indirectly in the air (dual-precision mode) if occlusion is an issue.

drop is compensated for in certain situations since offloading widgets to the air has interesting properties. Widgets displayed in the air are readily accessible when they would otherwise be several steps away in the hierarchy. Some phone screen real-estate gets freed up to show more content. We derive the following guidelines from these empirical results.

First, *discrete controls (*e.g.*, buttons, checkboxes) should not be offloaded to the air.* Users have much trouble activating discrete controls with air gestures, at least with the default MRTK behavior used on the HoloLens 2.

Second, *dual-precision air control (CD ⌐̈) can effectively complement touch for parameter adjustments and 2D selections.* Our observations reveal that users' performance and preferences are split between touch and dual-precision air control for on-screen widgets. Participants were faster in experiment E_2 with dual-precision air control than with touch control for sliders on screen. In experiment E_3, they performed equally well with touch and air controls (both dual-precision) for low-difficulty cursor positioning (4-mm precision). Air control performance degraded at higher difficulty, but the technique remains interesting nevertheless. First because it supports a three-state model of input which gives interaction designers more expressive power. But also because it is indirect and thus avoids visual occlusion issues – which were not operationalized in our task. Occlusion can be partly addressed in a touch-only context by featuring an area on the phone's screen to trigger indirect control as our baseline condition does in experiment E_3, but this takes some screen real-estate. Thus, as touch and air controls can coexist, there are only benefits in supporting both.

Finally, *frequently-used sliders can be good candidates for offloading to the air.* While participants were slower with air sliders than they were with screen sliders, the performance drop of *Air - Air* (*CD* ⌐) illustrated in Figure 5 (about 1-to-2 seconds) can be considered reasonable when put in perspective. Touch sliders are faster, but they consume a lot of screen space and accessing them often involves navigating menus (Figure 1-(a)). Air sliders are slower, but can be accessed very quickly without navigating the menu hierarchy. In addition, they do not generate any visual occlusion, as opposed to high-precision sliders such as OrthoZoom which extends control space beyond the widget's visual footprint. Interaction designers should thus consider this trade-off when making a choice between touch and air sliders for parameter adjustments.

Figure 7 illustrates how these guidelines can be applied to the design of our image editing example from Figure 1. The toolbar remains on the phone's screen as offloading buttons in the air would cause too much of a drop in performance. Frequently-used sliders, which were accessible after navigating a menu, are offloaded in the air and thus brought to the application's top layer. The second button in the toolbar ( ), which was used to invoke those sliders, is thus no longer necessary. This frees a bit more space in the phone's toolbar to accommodate additional controls or enlarge other buttons. Finally, 2D selection can be performed directly on the phone, or indirectly in the air (dual-precision mode) if occlusion is an issue.

6 LIMITATIONS AND FUTURE WORK

Studies as ours necessarily depend on the characteristics of the apparatus. Even using a state-of-the-art ARHMD, there are limitations, related primarily to the field of view and tracking accuracy. Our study gives a sense of the comparative performance of different widget designs with *current* AR technology. As this technology improves, comparative performance results might evolve. Novel sensing and display capabilities might also enable enhanced widget designs. Beyond initial empirical results, our contribution also includes a series of AR-enhanced widget designs, along with a set of tasks representative of phone interactions to test them. These can serve as a basis for future studies.

The observed performance drop of air control relative to touch should be interpreted keeping in mind that we compared AR-enhanced techniques to OrthoZoom and to a dual-precision touch pointer. These state-of-the-art touch-only techniques are hard to beat, even more so given the experience people have accumulated with touch interfaces over the years. Several participants reported that they found touch input more reliable, but also that they were feeling increasingly efficient with air input as they were proceeding with the experiment. It would be interesting to conduct longer studies in order to quantify learning aspects and identify performance envelopes [12]. Follow-up studies should also consider more elaborate tasks to evaluate the impact of visual occlusion and the

overhead caused by having to navigate in the widget hierarchy in touch-only conditions.

Because of the covid-19 pandemic, the studies were run with a restricted pool of participants, including the authors themselves. Although this is unusual, we believe that our results have reasonable validity. First, running our statistical analyses without the authors does not change the outcome. Second, our contribution is an exploration of various AR-enhanced widget designs. There is no notion of *the authors' solution* and thus no bias in favor or against any specific design. Our pool also includes both participants with much ARHMD experience and participants who had never even seen one before. Despite this great variability, we were able to observe significant – and sometimes quite large – effects. However, we insist that our results remain exploratory, and should not be generalized without further investigation.

Finally, there are likely opportunities to improve air controls in terms of interaction design. For example, we used the default MRTK behavior for air buttons, but we thought about other mechanisms to activate them. For instance, by crossing them with finger movements coplanar to the phone, whose edges would provide haptic feedback. While participants performed better with sliders displayed on the phone but controlled in the air than with sliders fully in the air, several of them indicated that they preferred the latter. They did not like the input-output mismatch of the former. Adapting a dual-precision strategy for sliders fully in the air could also improve performance.

ACKNOWLEDGMENTS

We sincerely thank the participants who took part in our experiments.

REFERENCES

[1] David Ahlström, Khalad Hasan, and Pourang Irani. 2014. Are You Comfortable Doing That? Acceptance Studies of around-Device Gestures in and for Public Settings. In *Proceedings of the 16th International Conference on Human-Computer Interaction with Mobile Devices and Services* (Toronto, ON, Canada) *(MobileHCI '14)*. Association for Computing Machinery, New York, NY, USA, 193–202. https://doi.org/10.1145/2628363.2628381

[2] Caroline Appert, Olivier Chapuis, and Emmanuel Pietriga. 2010. High-Precision Magnification Lenses. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Atlanta, Georgia, USA) *(CHI '10)*. Association for Computing Machinery, New York, NY, USA, 273–282. https://doi.org/10.1145/1753326.1753366

[3] Caroline Appert and Jean-Daniel Fekete. 2006. OrthoZoom Scroller: 1D Multi-Scale Navigation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Montréal, Québec, Canada) *(CHI '06)*. Association for Computing Machinery, New York, NY, USA, 21–30. https://doi.org/10.1145/1124772.1124776

[4] Daniel Ashbrook, Patrick Baudisch, and Sean White. 2011. Nenya: Subtle and Eyes-Free Mobile Input with a Magnetically-Tracked Finger Ring. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Vancouver, BC, Canada) *(CHI '11)*. Association for Computing Machinery, New York, NY, USA, 2043–2046. https://doi.org/10.1145/1978942.1979238

[5] Eugenie Brasier, Olivier Chapuis, Nicolas Ferey, Jeanne Vezien, and Caroline Appert. 2020. ARPads: Mid-air Indirect Input for Augmented Reality. In *2020 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. 332–343. https://doi.org/10.1109/ISMAR50242.2020.00060

[6] Frederik Brudy, Christian Holz, Roman Rädle, Chi-Jui Wu, Steven Houben, Clemens Nylandsted Klokmose, and Nicolai Marquardt. 2019. Cross-Device Taxonomy: Survey, Opportunities and Challenges of Interactions Spanning Across Multiple Devices. In *Proceedings of the CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland Uk) *(CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–28. https://doi.org/10.1145/3290605.3300792

[7] Rahul Budhiraja, Gun. A. Lee, and Mark Billinghurst. 2013. Using a HHD with a HMD for mobile AR interaction. In *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. 1–6. https://doi.org/10.1109/ISMAR.2013.6671837

[8] Alex Butler, Shahram Izadi, and Steve Hodges. 2008. SideSight: Multi-"touch" Interaction around Small Devices. In *Proceedings of the 21st Annual ACM Symposium on User Interface Software and Technology* (Monterey, CA, USA) *(UIST '08)*. Association for Computing Machinery, New York, NY, USA, 201–204. https://doi.org/10.1145/1449715.1449746

[9] William Buxton. 1990. A Three-State Model of Graphical Input. In *Proceedings of the IFIP TC13 Third Interational Conference on Human-Computer Interaction (INTERACT '90)*. North-Holland Publishing Co., NLD, 449–456.

[10] Xiang "Anthony" Chen, Nicolai Marquardt, Anthony Tang, Sebastian Boring, and Saul Greenberg. 2012. Extending a Mobile Device's Interaction Space through Body-Centric Interaction. In *Proceedings of the 14th International Conference on Human-Computer Interaction with Mobile Devices and Services* (San Francisco, California, USA) *(MobileHCI '12)*. Association for Computing Machinery, New York, NY, USA, 151–160. https://doi.org/10.1145/2371574.2371599

[11] Xiang "Anthony" Chen, Julia Schwarz, Chris Harrison, Jennifer Mankoff, and Scott E. Hudson. 2014. Air+touch: Interweaving Touch & in-Air Gestures. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology* (Honolulu, Hawaii, USA) *(UIST '14)*. Association for Computing Machinery, New York, NY, USA, 519–525. https://doi.org/10.1145/2642918.2647392

[12] John Dudley, Hrovje Benko, Daniel Wigdor, and Per Ola Kristensson. 2019. Performance Envelopes of Virtual Keyboard Text Input Strategies in Virtual Reality. In *Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR '19)*. IEEE, 289–300. https://doi.org/10.1109/ISMAR.2019.00027

[13] Anna Eiberger, Per Ola Kristensson, Susanne Mayr, Matthias Kranz, and Jens Grubert. 2019. Effects of Depth Layer Switching between an Optical See-Through Head-Mounted Display and a Body-Proximate Display. In *Symposium on Spatial User Interaction* (New Orleans, LA, USA) *(SUI '19)*. Association for Computing Machinery, New York, NY, USA, Article 15, 9 pages. https://doi.org/10.1145/3357251.3357588

[14] Tovi Grossman, Ken Hinckley, Patrick Baudisch, Maneesh Agrawala, and Ravin Balakrishnan. 2006. Hover Widgets: Using the Tracking State to Extend the Capabilities of Pen-Operated Devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Montréal, Québec, Canada) *(CHI '06)*. Association for Computing Machinery, New York, NY, USA, 861–870. https://doi.org/10.1145/1124772.1124898

[15] Jens Grubert, Matthias Heinisch, Aaron Quigley, and Dieter Schmalstieg. 2015. MultiFi: Multi Fidelity Interaction with Displays On and Around the Body. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (Seoul, Republic of Korea) *(CHI '15)*. Association for Computing Machinery, New York, NY, USA, 3933–3942. https://doi.org/10.1145/2702123.2702331

[16] Jens Grubert and Matthias Kranz. 2017. HeadPhones: Ad Hoc Mobile Multi-Display Environments through Head Tracking. In *Proceedings of the CHI Conference on Human Factors in Computing Systems* (Denver, Colorado, USA) *(CHI '17)*. Association for Computing Machinery, New York, NY, USA, 3966–3971. https://doi.org/10.1145/3025453.3025533

[17] Jens Grubert, Matthias Kranz, and Aaron Quigley. 2016. Challenges in mobile multi-device ecosystems. *mUX: The Journal of Mobile User Experience* 5, 1 (2016), 5. https://doi.org/10.1186/s13678-016-0007-y

[18] Jens Grubert, Eyal Ofek, Michel Pahud, Matthias Kranz, and Dieter Schmalstieg. 2016. GlassHands: Interaction Around Unmodified Mobile Devices Using Sunglasses. In *Proceedings of the ACM International Conference on Interactive Surfaces and Spaces* (Niagara Falls, Ontario, Canada) *(ISS '16)*. Association for Computing Machinery, New York, NY, USA, 215–224. https://doi.org/10.1145/2992154.2992162

[19] Jaehyun Han, Sunggeun Ahn, Keunwoo Park, and Geehyuk Lee. 2017. Designing Touch Gestures Using the Space around the Smartwatch as Continuous Input Space. In *Proceedings of the ACM International Conference on Interactive Surfaces and Spaces* (Brighton, United Kingdom) *(ISS '17)*. Association for Computing Machinery, New York, NY, USA, 210–219. https://doi.org/10.1145/3132272.3134134

[20] Chris Harrison and Scott E. Hudson. 2009. Abracadabra: Wireless, High-Precision, and Unpowered Finger Input for Very Small Mobile Devices. In *Proceedings of the 22nd Annual ACM Symposium on User Interface Software and Technology* (Victoria, BC, Canada) *(UIST '09)*. Association for Computing Machinery, New York, NY, USA, 121–124. https://doi.org/10.1145/1622176.1622199

[21] Khalad Hasan, David Ahlström, Junhyeok Kim, and Pourang Irani. 2017. AirPanes: Two-Handed Around-Device Interaction for Pane Switching on Smartphones. In *Proceedings of the CHI Conference on Human Factors in Computing Systems* (Denver, Colorado, USA) *(CHI '17)*. Association for Computing Machinery, New York, NY, USA, 679–691. https://doi.org/10.1145/3025453.3026029

[22] Juan David Hincapié-Ramos, Xiang Guo, Paymahn Moghadasian, and Pourang Irani. 2014. Consumed Endurance: A Metric to Quantify Arm Fatigue of Mid-Air Interactions. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Toronto, Ontario, Canada) *(CHI '14)*. Association for Computing Machinery, New York, NY, USA, 1063–1072. https://doi.org/10.1145/2556288.2557130

[23] ISO. 2000. 9241-9 Ergonomic requirements for office work with visual display terminals (VDTs)-Part 9: Requirements for non-keyboard input devices. *International Organization for Standardization* (2000).

[24] Brett Jones, Rajinder Sodhi, David Forsyth, Brian Bailey, and Giuliano Maciocci. 2012. Around Device Interaction for Multiscale Navigation. In *Proceedings of the 14th International Conference on Human-Computer Interaction with Mobile Devices and Services* (San Francisco, California, USA) *(MobileHCI '12)*. Association for Computing Machinery, New York, NY, USA, 83–92. https://doi.org/10.1145/2371574.2371589

[25] Hamed Ketabdar, Mehran Roshandel, and Kamer Ali Yüksel. 2010. Towards Using Embedded Magnetic Field Sensor for around Mobile Device 3D Interaction. In *Proceedings of the 12th International Conference on Human Computer Interaction with Mobile Devices and Services* (Lisbon, Portugal) *(MobileHCI '10)*. Association for Computing Machinery, New York, NY, USA, 153–156. https://doi.org/10.1145/1851600.1851626

[26] Gierad Laput, Robert Xiao, Xiang "Anthony" Chen, Scott E. Hudson, and Chris Harrison. 2014. Skin Buttons: Cheap, Small, Low-Powered and Clickable Fixed-Icon Laser Projectors. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology* (Honolulu, Hawaii, USA) *(UIST '14)*. Association for Computing Machinery, New York, NY, USA, 389–394. https://doi.org/10.1145/2642918.2647356

[27] Nicolai Marquardt, Frederik Brudy, Can Liu, Ben Bengler, and Christian Holz. 2018. SurfaceConstellations: A Modular Hardware Platform for Ad-Hoc Reconfigurable Cross-Device Workspaces. In *Proceedings of the CHI Conference on Human Factors in Computing Systems* (Montreal QC, Canada) *(CHI '18)*. Association for Computing Machinery, New York, NY, USA, 1–14. https://doi.org/10.1145/3173574.3173928

[28] Nicolai Marquardt, Ricardo Jota, Saul Greenberg, and Joaquim A. Jorge. 2011. The Continuous Interaction Space: Interaction Techniques Unifying Touch and Gesture on and above a Digital Surface. In *Human-Computer Interaction – INTERACT 2011*, Pedro Campos, Nicholas Graham, Joaquim Jorge, Nuno Nunes, Philippe Palanque, and Marco Winckler (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 461–476. https://doi.org/10.1007/978-3-642-23765-2_32

[29] Tim Menzner, Travis Gesslein, Alexander Otte, and Jens Grubert. 2020. Above Surface Interaction for Multiscale Navigation in Mobile Virtual Reality. In *2020 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. 372–381. https://doi.org/10.1109/VR46266.2020.00057

[30] Alexandre Millette and Michael J. McGuffin. 2016. DualCAD: Integrating Augmented Reality with a Desktop GUI and Smartphone Interaction. In *IEEE International Symposium on Mixed and Augmented Reality (ISMAR-Adjunct)*. 21–26. https://doi.org/10.1109/ISMAR-Adjunct.2016.0030

[31] Peter Mohr, Markus Tatzgern, Tobias Langlotz, Andreas Lang, Dieter Schmalstieg, and Denis Kalkofen. 2019. TrackCap: Enabling Smartphones for 3D Interaction on Mobile Head-Mounted Displays. In *Proceedings of the CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland Uk) *(CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–11. https://doi.org/10.1145/3290605.3300815

[32] Meredith Ringel Morris, Andreea Danielescu, Steven Drucker, Danyel Fisher, Bongshin Lee, Jacob O Wobbrock, et al. 2014. Reducing legacy bias in gesture elicitation studies. *interactions* 21, 3 (2014), 40–45. https://doi.org/10.1145/2591689

[33] Erwan Normand and Michael J. McGuffin. 2018. Enlarging a Smartphone with AR to Create a Handheld VESAD (Virtually Extended Screen-Aligned Display). In *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. 123–133. https://doi.org/10.1109/ISMAR.2018.00043

[34] Anna Ostberg and Nada Matic. 2015. Hover Cursor: Improving Touchscreen Acquisition Of Small Targets With Hover-Enabled Pre-Selection. In *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems* (Seoul, Republic of Korea) *(CHI EA '15)*. Association for Computing Machinery, New York, NY, USA, 1723–1728. https://doi.org/10.1145/2702613.2732903

[35] Roman Rädle, Hans-Christian Jetter, Nicolai Marquardt, Harald Reiterer, and Yvonne Rogers. 2014. HuddleLamp: Spatially-Aware Mobile Displays for Ad-Hoc Around-the-Table Collaboration. In *Proceedings of the Ninth ACM International Conference on Interactive Tabletops and Surfaces* (Dresden, Germany) *(ITS '14)*. Association for Computing Machinery, New York, NY, USA, 45–54. https://doi.org/10.1145/2669485.2669500

[36] Hanae Rateau, Yosra Rekik, Edward Lank, and Laurent Grisoni. 2018. EtherToolbars: Evaluating Off-Screen Toolbars for Mobile Interaction. In *23rd International Conference on Intelligent User Interfaces* (Tokyo, Japan) *(IUI '18)*. Association for Computing Machinery, New York, NY, USA, 487–495. https://doi.org/10.1145/3172944.3172978

[37] Marcos Serrano, Barrett Ens, Xing-Dong Yang, and Pourang Irani. 2015. Gluey: Developing a Head-Worn Display Interface to Unify the Interaction Experience in Distributed Display Environments. In *Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services* (Copenhagen, Denmark) *(MobileHCI '15)*. Association for Computing Machinery, New York, NY, USA, 161–171. https://doi.org/10.1145/2785830.2785838

[38] Marcos Serrano, Dale Hildebrandt, Sriram Subramanian, and Pourang Irani. 2014. Identifying Suitable Projection Parameters and Display Configurations for Mobile True-3D Displays. In *Proceedings of the 16th International Conference on Human-Computer Interaction with Mobile Devices & Services* (Toronto, ON, Canada) *(MobileHCI '14)*. Association for Computing Machinery, New York, NY, USA, 135–143. https://doi.org/10.1145/2628363.2628375

[39] Daniel Spelmezan, Caroline Appert, Olivier Chapuis, and Emmanuel Pietriga. 2013. Controlling Widgets with One Power-up Button. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology* (St. Andrews, Scotland, United Kingdom) *(UIST '13)*. Association for Computing Machinery, New York, NY, USA, 71–74. https://doi.org/10.1145/2501988.2502025

[40] Daniel Spelmezan, Caroline Appert, Olivier Chapuis, and Emmanuel Pietriga. 2013. Side Pressure for Bidirectional Navigation on Small Devices. In *Proceedings of the 15th International Conference on Human-Computer Interaction with Mobile Devices and Services* (Munich, Germany) *(MobileHCI '13)*. Association for Computing Machinery, New York, NY, USA, 11–20. https://doi.org/10.1145/2493190.2493199

[41] Hemant Bhaskar Surale, Aakar Gupta, Mark Hancock, and Daniel Vogel. 2019. TabletInVR: Exploring the Design Space for Using a Multi-Touch Tablet in Virtual Reality. In *Proceedings of the CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland Uk) *(CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3290605.3300243

[42] Daniel Vogel and Patrick Baudisch. 2007. Shift: A Technique for Operating Pen-Based Interfaces Using Touch. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (San Jose, California, USA) *(CHI '07)*. Association for Computing Machinery, New York, NY, USA, 657–666. https://doi.org/10.1145/1240624.1240727

[43] Dirk Wenig, Johannes Schöning, Alex Olwal, Mathias Oben, and Rainer Malaka. 2017. WatchThru: Expanding Smartwatch Displays with Mid-Air Visuals and Wrist-Worn Augmented Reality. In *Proceedings of the CHI Conference on Human Factors in Computing Systems* (Denver, Colorado, USA) *(CHI '17)*. Association for Computing Machinery, New York, NY, USA, 716–721. https://doi.org/10.1145/3025453.3025852

[44] Fengyuan Zhu and Tovi Grossman. 2020. BISHARE: Exploring Bidirectional Interactions Between Smartphones and Head-Mounted Augmented Reality. In *Proceedings of the CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–14. https://doi.org/10.1145/3313831.3376233