Designing a better action menu for a mixed reality/virtual reality collaboration app

Ethan Guo Relevant courses: CS 349, CS 449, CS 488

Background

Virtual reality, or VR — the idea of using technology to completely immerse yourself in a virtual world — has been dreamt about in science fiction since the 1930s, but only in the last few years has VR technology started to appear on the consumer market, in the form of head-mounted display (HMD) devices such as the Oculus Quest.

Even more recently, a new class of HMD devices have emerged that feature a transparent display, allowing them to superimpose graphics over your field of vision. When combined with spatial tracking, this enables "holograms" (yes, that is the technical term) — virtual objects and scenes that can be anchored to your real-world environment. This technology, which blends the virtual and real world, is known as mixed reality (MR). The Microsoft HoloLens was the first MR platform widely available to consumers, followed by next-generation MR devices such as the HoloLens 2 and the Magic Leap.

Spatial is a cross-platform MR/VR application that allows people to collaborate in a shared virtual or holographic room, or "space," regardless of where you are in the world. Users are represented by a life-like 3D avatar and can freely move around this space, add sticky notes or freeform 3D drawings, and upload content such as 3D models, images and PDFs into the space. These objects are shared with all users in the room, so everyone can see and interact with the same content.

During my recent internship at Spatial, I was tasked with the major project of redesigning and rebuilding the "dock", the central action menu from which all of the above features (and more) are selected or performed. The previous dock was visually outdated, unintuitive to use, and not discoverable; the code for it was also poorly structured and difficult to modify or maintain, so a complete rewrite was in order.

The basic visual design of this new dock was already set, but I worked with the designers to solve some unique UX and interaction design problems that only manifested in the realm of MR/VR. Here we will explore one such design problem which showcases some of the key technical differences and challenges that come with designing user experiences in MR/VR.

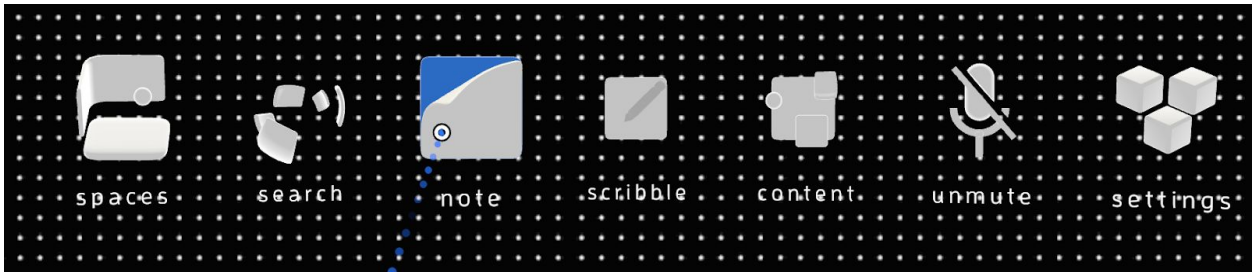

Figure 1. The old dock — too cluttered, and inconsistent with the rest of the app's visual style.

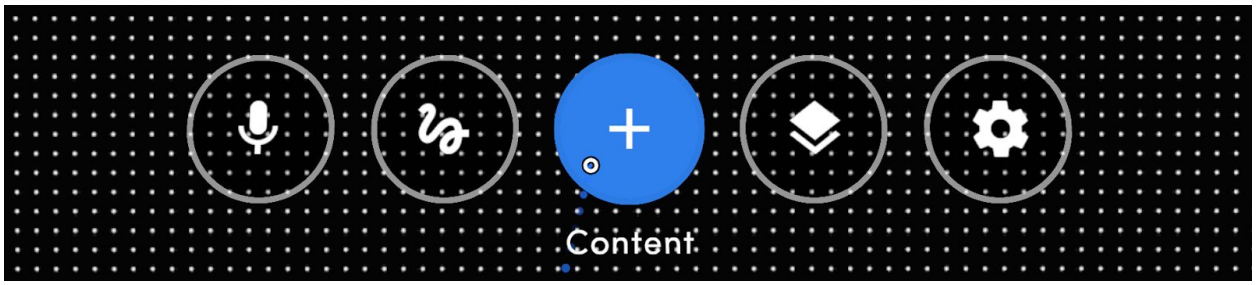

Figure 2. The new dock — sleek, discoverable, and on brand!

The problem

One of the first design challenges we had to tackle when designing the new dock was how the user would access (**invoke**) the dock.

The first and biggest requirement was the chosen invocation method had to be **discoverable**. The old dock was distinctly not discoverable — the only hint to the user that it even exists is a toast message that appears briefly when they enter a room for the first time, instructing them to "double air-tap open the dock." The double-tap is not a standard nor intuitive gesture, so if the user misses that toast message or forgets the gesture that opens the dock, they will be stuck without access to any of the app's basic functionality.

Now, this would be a rather trivial problem to solve in most traditional applications — a simple gear icon or hamburger menu displayed one corner of the screen would do the trick. **However, a major constraint when designing for MR/VR is that we cannot use fixed screen-space UI elements.** Since MR/VR involves a stereoscopic display that shows a slightly different view to each eye to enable depth perception, HUD-style graphics cannot simply be rendered as an overlay over the rest of the scene. As the Oculus design guide describes, this can cause visual discomfort and confusion if scene elements are meant to be closer than the HUD, yet the HUD is still rendered "in front" of the scene elements. Additionally, HUD elements can be very uncomfortable to look at if they are placed towards

the edges or corners of the screen, because you cannot simply turn your head to bring them into the centre of your view — the display is attached to your head!

Two more design constraints were that we must **follow each platform's standards and conventions for interaction design**, and that our app's user experience should be mostly **consistent across all platforms**. Unfortunately, as you will see in the next section, these two requirements were sometimes irreconcilable because each platform had very different input patterns, so it was decided that the former took precedence over the latter.

Finally, the chosen interaction method should also be **easy and comfortable to use**, as well as **not significantly obstructing any existing functionality** such as interacting with content in the space. These were measured by conducting informal internal usability tests with sample sizes of up to 20.

Interaction types across different platforms

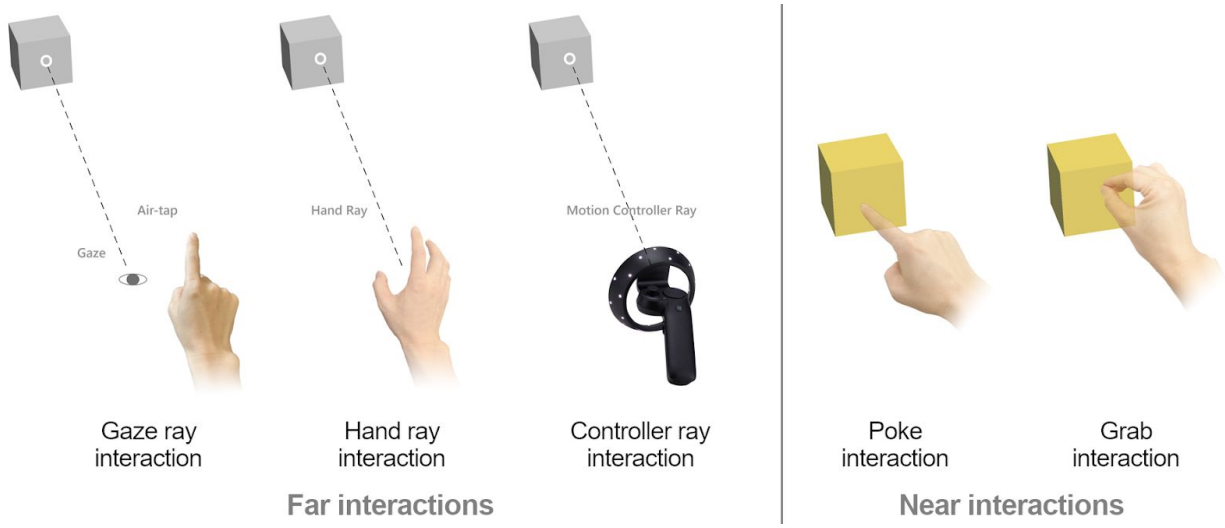

Figure 3. Illustration of all available interaction types.

The Microsoft **HoloLens 1** was one of the earliest MR headsets available. It utilizes basic hand tracking for user input. This detects hand gestures, such as a pinch gesture called an **air-tap** which serves as its basic "click" interaction. The target of the click is determined by shooting a ray forward from the exact centre point of your field of view (indicated by a small circular reticle). This type of interaction is known as **gaze ray interaction**.

The next-generation **HoloLens 2** was capable of much more precise hand tracking which allowed for much more intuitive and versatile input and interaction types. We can divide these available interaction types into two categories:

First, **near interactions**, where you can directly interact with objects using your hands. For example, you could push a holographic button with your finger (a **poke interaction**) or close your fist around a holographic object to drag it around (a **grab interaction**). Second, **far interactions**, which allow you to interact with objects at a distance by pointing at them. The HoloLens 1 gaze ray interaction is one type of far interaction. The HoloLens 2 uses a similar air-tap "click" gesture as on the HoloLens 1, but it improves upon this by making use of a **hand ray** (indicated by a line originating from your hand, like a laser pointer) to allow you to use your hand to point at the object you want to select or interact with, rather than having to turn your head to gaze directly at it. We will call this interaction type a **hand ray interaction**.

The **Magic Leap**, another MR headset, uses a handheld controller instead of hand tracking. This interaction type is similar to the HoloLens 2's hand ray interaction, but the ray originates from the controller instead, and it uses a button on the controller to "click" instead of the air-tap gesture. We will call this a **controller ray interaction**. In theory, it is possible to also track the Magic Leap's controller position to simulate near interaction, but that went against design conventions on this platform — and per the stated requirements, platform-specific conventions take priority over cross-platform consistency.

Finally, we have our only VR platform, the **Oculus Quest**. The Quest also used handheld controllers, but these controllers were represented as virtual hands in VR (since unlike in MR, you can't see your real hands). A surprising consequence of this is that the Quest is almost identical to the HoloLens 2 in terms of interaction design — the same near interactions could be used (the controller had a "grip" trigger which substituted the fist gesture for grab interactions), and the controller ray was functionally equivalent to the HoloLens 2's hand ray.

	Far interactions			**Near interactions**	
Platform	Gaze ray	Hand ray	Controller ray	Poke	Grab
HoloLens 1	✔				
HoloLens 2		✔		✔	✔
Magic Leap			✔		
Oculus Quest			✔	✔	✔

Figure 4. Summary of supported interaction types by platform.

The solution

Our solution for a discoverable dock invocation mechanism is to display a persistent "minimized" dock that would always hover near the resting position of the user's hands, and which the user could interact with to expand it into the full dock. Since we cannot use fixed screen-space elements in MR/VR, this minimized dock is treated as a scene object, and would float and follow you in a kinematically accurate way if you turn your head or move around your space. This visual minimized dock served two main purposes:

First, it allows a user who has not used the app before to discover that the dock exists in the first place, as well as how to use it. To achieve this second part, we added a simple "hint text" that will appear on top of the minimized dock when you hover a cursor (the gaze reticle, or a hand or controller ray) on it, or if a tracked hand or controller moves close enough (an "activation radius" for near interactions).

Second, it serves as a handle or target for the invocation interactions. The specific interaction methods would depend on the platform it's running on.

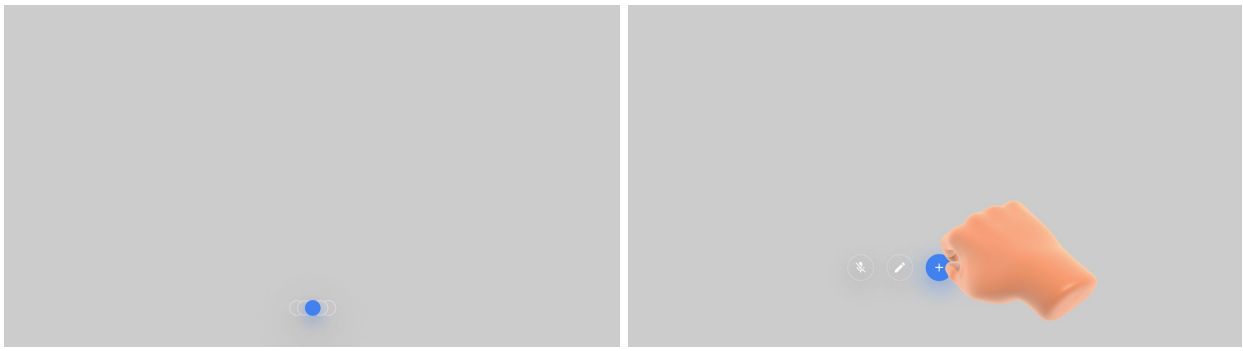

Figure 5. Left: *the minimized dock (simulated perceived size relative to field of view).*
Right: *the dock after being grabbed and dragged up to open it.*

On **HoloLens 2 and Oculus Quest**, the minimized dock only responds to near interactions, either by poking it like a physical button to invoke the dock, or by grabbing it and pulling it up like a drawer. For the grab interaction, the minimized dock visual would move to follow your hand as if you were pulling out a drawer by the handle, and the dock would open when you release it.

Both types of near interaction were supported because it was found that each was suited to different users and use cases. When in a sitting position, for example, the drag interaction is easier to perform than the poke interaction because you would have to move your hand in an unnatural direction and bring it too close to your body to order to begin a

poke interaction (which has to be started from the front of the interactive object). Meanwhile, the poke interaction is an easier gesture to learn and therefore first-time users would have more success using poke. We also conducted an informal internal survey on personal preference between using poke vs. using grab to invoke the dock, and the results were split almost exactly equally.

We do not use far interaction on HoloLens 2 and Oculus Quest — enabling far interaction would require making the dock be a valid target for hand or controller raycasts, which could block those rays and obstruct interacting with content in the room. We conducted an extensive usability test on this and determined that enabling far interaction is a significant obstruction (more than half of subjects reported their far interaction rays being blocked unexpectedly by the minimized dock), and that the near interactions were sufficient.

Since near interaction was not supported on **HoloLens 1 and Magic Leap**, we use far interactions which correspond to the poke and grab interactions. The minimized dock can be clicked via far interaction the same way as a regular button to invoke the dock (analogous to the poke interaction, which is also used for buttons). It can also be dragged by holding down the air-tap gesture or controller button while moving upwards (this is the standard way to drag virtual objects on these platforms, analogous to the grab gesture).

Unlike in the previous case, this drag gesture did not directly improve usability or comfort (since far interaction raycasting does not depend on hand position, so it doesn't necessarily allow for a more comfortable hand position), but it was kept for cross-platform consistency. But other than consistency for the sake of consistency, why else would it be desirable to support the drag interaction (and the associated visual feedback) on these platforms? Well, consider for instance a HoloLens 2 user switching to Magic Leap, who is accustomed to using the drag gesture. If out of habit they tried to drag the dock upwards using the Magic Leap controller, but we didn't support the drag gesture on Magic Leap, then the dock wouldn't move when they try to drag it up and they will get a false impression that their click didn't register properly — when in fact it did register the start of their click, it was just waiting for the click to be released.

The final constraint that we need to satisfy on these 2 devices is not obstructing interacting with other content. On the Magic Leap this is not an issue thanks to its large field of view (30° vertically), but it does present a challenge on the HoloLens 1 due to its very small field of view (only 17° vertically) and lack of flexibility for targeting (due to the gaze ray being fixed in the centre of your field of view). This was solved by lowering the position at which the dock appears to hover, so that it only appears when looking at a steep downward angle, which is unlikely to be encountered when interacting with room content.

Altogether, this new solution for the dock invocation mechanism fulfills all of its constraints and represents a vast improvement in user experience over the old dock. It introduces **discoverability** by making use of visual hints designed appropriately for MR/VR. It respects interaction design conventions for each platform, making the dock interactions much more **intuitive** and easy to remember. It **unifies** the user experience between different platforms by making use of a consistent visual element, and uses similar or analogous interaction methods between platforms that don't support the same interaction types. Finally, the comfort and ease of use of this solution was confirmed through multiple rounds of internal usability testing.

	Interactions used for dock invocation			
Platform	Far interaction click	Far interaction drag	Poke	Grab
HoloLens 1	✔	✔		
HoloLens 2			✔	✔
Magic Leap	✔	✔		
Oculus Quest			✔	✔

Figure 6. Matrix of interaction types used for dock invocation, by platform. Notice how this is less irregular than the matrix of all available interaction types in Figure 4.

Impact

When I first took on this project, I understood that it would be one of the highest visibility things I'd get to work on at Spatial, as the central menu bar for our app and the entry point for most actions taken in a Spatial space. But shortly after I started work on this project, the stakes were unexpectedly raised even higher.

As a collaboration app whose primary purpose is bringing people together from a distance, Spatial has picked up a ton of momentum and interest in the past 2 months during the COVID-19 pandemic, as many people have been forced to work from home and distance themselves from friends, coworkers, clients, etc. Because of this, less than a week after I started work on this project, the company made a decision to accelerate our efforts towards our next major release version and make it available to anyone for free — a stark contrast from our previous tightly-controlled enterprise sales channel model, with demos guided by our own sales people. Without these guided demos and support channels, the

public launch thus demanded a new level of polish, robustness and ease-of-use. Therefore, the dock rewrite was more crucial than ever, to ensure our core features were easily discoverable and accessible without needing guidance.

Spatial prides itself in good UX and good design — both co-founders are acclaimed UX designers in their own right — and the dock had long stood as a glaring (and ugly) reminder of the design and UX work that we've been long putting off. This refresh of one of the most prominent user interface elements made a huge difference to the perception of our brand. This was evidenced by the fact that the redesigned dock was featured as one of the **headlining features in the release notes** for our newly released update, and was even included in the press release slide deck for our open public release.

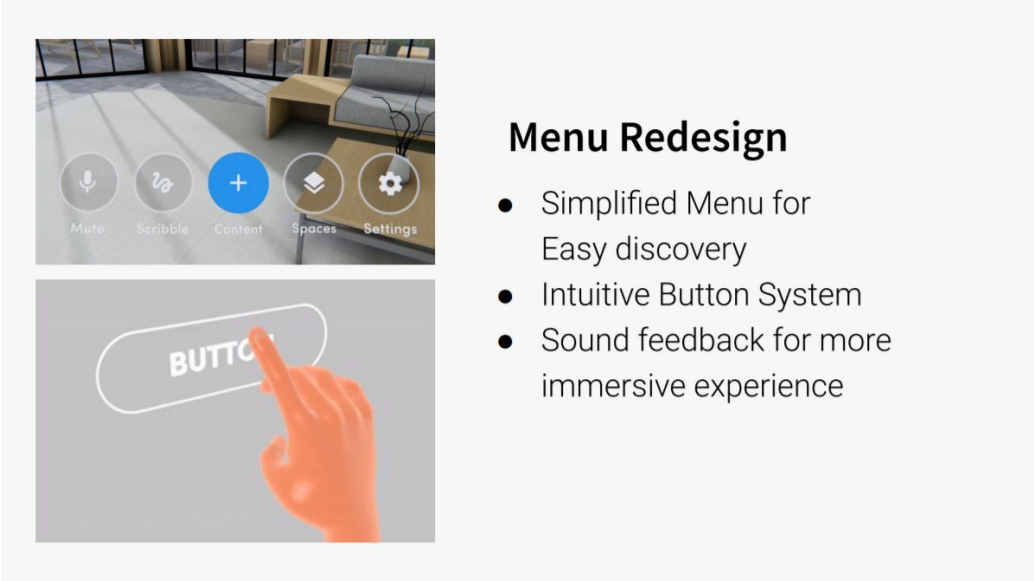

Figure 7. Slide 5 of 7 of our press release slide deck featured the redesigned dock.

But most importantly, when the feedback started rolling in from external users, it was overwhelmingly positive as well, and validated our belief that this greatly improved the user experience through discoverability and intuitiveness:

- When we sent the new update to the review team at Magic Leap, the dock was the first thing they noticed: "I love the redesigned menus! Much more intuitive."
- Multiple enterprise customers sent us feedback about the new update being "more intuitive", "easy for onboarding", and "easy to use".
- On the TWiT Podcast, Devindra Hardawar of Engadget (correctly) asserted that the biggest focus of this update was "onboarding ... they're teaching people how to use this platform," and he chose to use to illustrate that by talking about the dock!

Takeaways

The biggest thing I've gotten out of this co-op term at Spatial has been a new fascination with UX design and interaction design. Over the course of this project and co-op term, I found that the MR/VR world — and the 3D world in general — repeatedly challenged my assumptions about fundamental things in design like how a button works, what app navigation means, and how to present information. I've discovered that, as emerging fields, MR and VR currently provide a sandbox with relatively few rules and an opportunity to work on fascinating new design challenges that have never been tackled before.