TransceiVR: Bridging Asymmetrical Communication Between VR Users and External Collaborators

Balasaravanan Thoravi Kumaravel[1], **Cuong Nguyen**[2], **Stephen DiVerdi**[2], **Björn Hartmann**[1]
[1]UC Berkeley, Berkeley, CA, USA
[2]Adobe Research, San Francisco, CA, USA
{bala,bjoern}@eecs.berkeley.edu, {cunguyen,diverdi}@adobe.com

ABSTRACT
Virtual Reality (VR) users often need to work with other users, who observe them outside of VR using an external display. Communication between them is difficult; the VR user cannot see the external user's gestures, and the external user cannot see VR scene elements outside of the VR user's view. We carried out formative interviews with experts to understand these asymmetrical interactions and identify their goals and challenges. From this, we identify high-level system design goals to facilitate asymmetrical interactions and a corresponding space of implementation approaches based on the level of programmatic access to a VR application. We present TransceiVR, a system that utilizes VR platform APIs to enable asymmetric communication interfaces for third-party applications without requiring source code access. TransceiVR allows external users to explore the VR scene spatially or temporally, to annotate elements in the VR scene at correct depths, and to discuss via a shared static virtual display. An initial co-located user evaluation with 10 pairs shows that our system makes asymmetric collaborations in VR more effective and successful in terms of task time, error rate, and task load index. An informal evaluation with a remote expert gives additional insight on utility of features for real world tasks.

Author Keywords
Virtual Reality; Asymmetric Interaction; Collaboration

CCS Concepts
•**Human-centered computing** → **Virtual reality; Collaborative interaction;**

INTRODUCTION
Virtual Reality (VR) has gone from research technology to a popular user platform with a growing ecosystem of applications, games, and media. However, isolation of a VR user from their peers is a major problem hindering its adoption [26, 27]. While some VR applications are designed to be collaborative, most are conceived as single-user experiences. Our work focuses on the social interactions that emerge from single-user VR applications. Collaborative use of single-user desktop applications has been well studied, e.g. in "social learning" [46], "occasional meetings" [38], pair programming [19], and game streaming [30, 43, 50, 54]. In many of these, application state and the participants' speech and body languages are mutually visible to everyone, or *symmetric*. For VR applications, similar social interactions can also emerge between a person wearing a VR headset and other people being *external* (i.e. not in VR). E.g., a VR artist may want to help teach a new VR user how to paint and sculpt in VR. This communication is *asymmetric* because the VR and external users do not share the same display and input capabilities. The VR user's view is blocked by the headset so they cannot see the external user, and the external user does not have full visibility of the VR scene. This creates a communication barrier between the users.

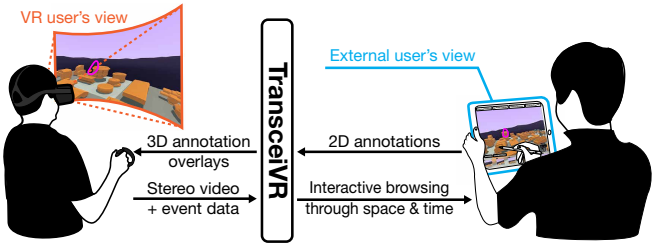

Figure 1: Asymmetric VR interaction using TransceiVR: An external user with tablet can annotate images from the VR user's application; annotations are projected back into VR at the appropriate depth.

Some VR systems can mirror the display output to a screen, but such a *VR mirror* only provides a partial and unstable view of the VR user's actions. It can be difficult for the external user to interact with the VR user, who is in a 3D space, solely using the *VR mirror*. Another solution is to design VR applications specifically for collaboration, either entirely in VR [22, 44, 47] or through custom asymmetric interactions [27, 28]. However, single-user applications are unlikely to be rebuilt for multi-user purposes unless there are strong business needs [17]. Well-known desktop tools such as Photoshop and Blender have not yet introduced real-time social features despite users teaching, guiding, and collaborating with these tools.

We conduct formative interviews with expert VR users who regularly participate in asymmetric VR interactions. We distill our findings into design goals that inform the development of software for asymmetric communication between VR users and external users. These high-level design goals focus on enriching the means by which information regarding the VR environment is delivered to the external user, as well as allowing them to relay information to the VR user.

UIST '20, October 20–23, 2020, Virtual Event, USA

ACM ISBN 978-1-4503-7514-6/20/10 ...$15.00.
DOI: http://dx.doi.org/10.1145/3379337.3415827

Developing these design goals into features require certain access to the underlying VR software and hardware stacks (Fig. 2). Our formative interviews reveal that asymmetric communication can often take place in closed-source VR applications (e.g., during onboarding or testing a new application). Thus we focus our effort on information collected at the VR platform level, without requiring source code access. These data include the VR stereo camera feed and the VR hardware readings, and the ability to inject overlays into VR.

To facilitate asymmetric communication, our TransceiVR system mirrors the VR user's feed onto a touchscreen tablet, so that the external user can sketch annotations and interact directly over this mirrored video. To properly position the annotations of the external user in the VR scene, we apply an optical flow-based technique on the mirrored video to estimate a depth map of the VR scene. TransceiVR also supports annotating recent temporal and spatial history, so the external user can discuss scene elements outside of VR user's view; these static annotated views can be shared between the users and used like a sketch board. TransceiVR also provides an interface for the external users to highlight specific buttons on the VR controller. This feature enables fine-grained discussions around the controller which are common in onboarding activities. Finally, the external user can also share a live camera feed of themselves to the VR user.

We conducted a preliminary evaluation of TransceiVR with 10 co-located user pairs to see how it affects the efficiency of asymmetrical VR interactions. We found that with TransceiVR, users completed a VR assembly task faster, with fewer errors, lower perceived workload, and felt it was easier to understand and communicate with their partner. TransceiVR also supports remote operation through any screen sharing and control software. We carried out a remote expert evaluation, in which they taught a novice using TransceiVR. From this, we gained feedback on the different components of TransceiVR.

There are three main contributions of this work. First, we report our findings of the problems in asymmetrical VR interaction through formative interviews and our design goals. Second, we present a design space of implementation approaches, and the TransceiVR system which realized these design goals for arbitrary existing applications. Finally, a user study of the system that shows an improved efficiency in communication in asymmetric interaction scenarios.

RELATED WORK

Prior research spans asymmetric interaction; multi-user VR and remote collaboration more broadly; as well as techniques for retrofitting existing applications to offer new functionality.

Asymmetric VR Interaction systems

Some games employ Asymmetric VR interactions, where one player operates in VR and one or more other players operate outside VR with additional accessories such as cheat sheets or controllers, with which they can indicate and perform actions in the game environment. These games may be cooperative, in which the players work together, e.g. *Black Hat Cooperative* [1], *Eye in the Sky* [2] or competitive, where they play against one another e.g. *Panoptic* [8], and *Nemesis perspective* [3].

Beyond games, recent research [23, 27, 28, 62] has identified the isolating nature of existing VR applications, and proposed techniques to address this problem. These works provide interesting insights into asymmetric VR interactions, but differ from the scope of TransceiVR in that they assume these interactions are predominantly spatial. Also, they focus on custom written applications where collaboration is key to the user experience. TransceiVR focuses on improving the asymmetric interactions in existing VR applications, that were not necessarily designed for multi-user experience. ShareVR [27] uses spatial augmented reality (SAR) projections [14], mobile displays, and tracked spaces to display the virtual world onto the physical world and vice-versa. FaceDisplay [28] has three touch displays that are directly mounted on the VR headset, which act as viewports and a means for the external user to interact with the VR user and play a VR game. TeleSight [23] uses an instrumented robotic head that mimics the VR user's HMD pose, to achieve a similar interaction. Sometimes these systems require an instrumented space with a rig of depth cameras and projectors. Von Willich et al.'s work renders different representations of a passerby into a custom VR app and compares them [62]. RealityCheck [34] composites 3D renderings of the user's physical surroundings into a VR application, and projects the VR environment into physical space in an application-agnostic manner. In general, these works try to reduce the asymmetry by mapping the virtual environment onto a real environment, and vice versa. Our work starts from perspective of enabling external collaborators to communicate with VR users without requiring additional tracking or projection infrastructure – this also enables our technique to be used remotely.

Multi-user VR experiences

Multi-user interaction and collaboration in VR has been a focus for several decades [18], and continues to be actively investigated. Systems such as CollaVR [47], SpaceTime [63], and Virtual Space [44] demonstrate the value of view and object sharing, and annotations to increase shared context in synchronous VR collaboration scenarios. These systems demonstrate how effective collaboration can be supported if it is designed into the core of each application. TransceiVR adopts screen sharing and annotation interactions from this prior work, but shows how such functionality can be retrofitted. A second line of work focuses on asynchronous guidance and tutorial systems in VR. Most closely related to our work, TutoriVR [61] introduces application-agnostic overlays to deliver previously recorded tutorial information and UI to a VR user. We also use the raw stereo-view feed from VR, and use similar pipeline to render information into VR. However, because TransceiVR focuses on synchronous interaction, we further process the stereo feed using real-time computer vision algorithms to extract depth maps of the VR scene that can then be used to correctly position annotations of external users. Overall, these works validate the need for collaboration in VR systems and promise increased productivity and social engagement. TransceiVR strives to achieve these goals while relaxing the constraint that all users need to be in VR, as well as not requiring the features to be built in to the app ground-up.

Remote guidance and collaboration for physical tasks

Prior research on remote guidance and collaboration for physical tasks generally involve a remote user guiding and collaborating with a local user. The remote user, usually has access to a 2D video feed [42], and more recently 360 video and 3D point cloud feeds [39, 59, 60], or a virtual replica [48] of the local user operating in their physical environment. They guide them through an audio channel along with the aid of other peripherals such as laser pointers [10, 29] or augmented reality devices [35, 41]. These works have demonstrated the value of having tools and interaction techniques such as annotations [29, 49], sharing hand gestures [11, 36, 40] and trade-offs between different viewpoints [24, 57]. Asymmetric VR interaction provides an interesting parallel to these scenarios, where the elements—VR user, the VR environment, the external spectator and the *VR mirror*—are correspondingly analogous to the local user, their physical environment, remote user, and the 2D video feed they view. This analogy points us towards potentially useful interaction techniques introduced by these works such as environment annotations, live sharing video streams and content, and enabling independent spatial exploration of the remote user's environment, which may be adapted to work in VR. However, there are fundamental differences between the real and virtual environments. Unlike the real-world, the elements in a virtual world commonly have dynamic and digital behaviour, e.g. objects and menus can spawn and disappear anywhere, and the scale of objects and the world can change. These differences break the interaction techniques and introduce new, interesting problems in VR.

Retrofitting software applications

TransceiVR retrofits existing single-user VR applications with new communication and collaborative features. Retrofitting frequently uses a combination of available platform APIs *underneath* a closed source application (e.g., using UI toolkit overloading [21] or accessibility APIs) and reverse engineering approaches to extract information where APIs are unavailable. The value of retrofitting and reverse engineering has been well established in the HCI research community. As Cheng et al. [17] write: "*Mission critical applications and legacy systems may be difficult to revise and rebuild, and yet it is sometimes desirable to retrofit their user interfaces with new collaborative features without modifying and recompiling the original code.*" Computer vision-based reverse engineering approaches have been used to enhance desktop software with new interaction techniques [20], automate GUI tasks [64], extract reusable data from rendered information visualizations [52], and improve the usability of video tutorials [51]. Most commercial video conferencing tools today include the ability for a remote party to control single user software on someone else's computer using screen sharing and input event injection, and RealityCheck [34], SeeingVR [65] and TutoriVR [61] highlight the value of application-independent compositing of information into VR. We build on these retrofitting approaches with a focus on facilitating efficient asymmetric communication in existing VR applications.

Formative Interviews

To understand user's needs and challenges of asymmetrical VR interactions, we interviewed five (5) expert users: a VR user experience (UX) designer, a VR UX researcher, a graduate student VR researcher, a VR engineer, and a 360° filmmaker. Each had encountered various asymmetrical scenarios at their work, including conducting VR user studies, collecting feedback on new VR user interfaces, testing new VR prototypes, and reviewing 360° films. Besides these primary activities, they have also engaged in other asymmetrical VR interactions during their non-work times such as demoing VR to friends and playing and viewing VR games. The interview started with the experts giving an overview of their experience with VR. Then they discussed their primary asymmetrical VR activity, using a verbal step-wise walk through of it. When appropriate, some used prior video captures, or enacted it, to clarify details. Our questions focused on goals of the interaction, how it was carried out including its limitations and challenges, the roles of the two users, and their social dynamics. Interviews concluded with open-ended remarks from the experts.

All participants reported using a *VR mirror* as the only way to share display for grounding discussion. *VR mirror* is a standard feature supported in most commercial platforms. It displays what the VR user is seeing on an external display such as a TV screen. However, through the interviews, we identified a number of problems of this technique towards supporting asymmetrical VR communication.

A constantly moving first person view

The video feed in *VR mirror* is very jittery and all our users indicated that following a VR activity this way can be uncomfortable. VR experiences are immersive and often induce frequent, large head motion, leading to unstable video feeds. In VR, this is specifically aggravated due to its reduced field of view, which increases the head movement as well as the time taken by the VR users to locate virtual elements [45].

More issues arise when an external user needs to refer to an object in the VR scene. Since the *VR mirror* shows the VR user's live view, the desired object is visible only when the VR user views it. Otherwise, it will not be visible to the external user. Worse, with external users nearby, when discussing a scene element, the VR user may turn towards them, thereby changing views more frequently.

The difficulties of talking about VR scene elements

A typical VR scene contains dynamic events and objects, many of which are alien concepts in real world (e.g. an interface floating in mid-air). Thus, it is often difficult for the two users to talk about or refer to elements in the scene. We identified four unique problems:

1. Transient elements: Many objects in VR may be seen for only a short period of time. Examples include user interface menus and game objects. The transient nature of these VR elements makes it difficult for the external user to refer to them. For example, an external user may want to instruct a novice VR user on how to choose a new brush style in Tilt Brush [25, 61]. The VR user may make mistakes like pointing at the wrong menus or selecting the wrong brush buttons. Remedying such errors is usually slow: either the VR user has to repeat the actions or the external user has to verbally describe those actions for the VR user to redo.

2. VR Controller elements: Hand controllers are the primary interaction mechanism in VR. As noted in TutoriVR [61], naming and design conventions for VR inputs are not yet standardized. For instance, *grip*, *squeeze*, *secondary trigger* and *fist* were four different terms used by the experts in our formative study to refer to the same button in two different VR platforms (Oculus vs. Vive). One may verbally describe the button by its shape and position. But this often fails in VR applications that do not render a virtual depiction of the hand controllers [9]. In such cases, the external user may resort to instructing the VR user on specific finger(s) to hold down.

3. Gestural elements: A common hassle mentioned by experts is when the external user has to instruct gestural movements to the VR user. These are essential in many VR applications [56] and may be used for common spatial actions like painting, aiming, or teleportation. For these, our experts reported using verbal description of something the VR user might have prior knowledge about (e.g., "move your hand like a sine curve" or "picture yourself as *Ironman* having boosters in your hands and try to push to fly"). Alternatively, they may hold VR user's hand to physically guide movement. But they remarked that this is sometimes infeasible, since it may break social boundaries and make either of them uncomfortable.

4. Directional & attentional elements: It is common for the external user to direct the VR user's attention toward a particular scene area. Some examples include "*move this box to this position*" or "*pay attention to that region*." But these directions are not effective because the VR user does not see the external user's pointing gestures. Understanding a peer's activity is a key requirement for effective shared conversation [55]. Our experts reported workarounds based on verbal egocentric references such as '*Turn to your 3'o clock*" or "*look to your left*". But they found it to be cognitively demanding because they have to constantly take the changing perspective of a VR user. They can also give inaccurate, guessed directions to objects that are not immediately visible to them. Remedying these errors leads to lengthy back and forth conversation between the two users. Elements having distinct shape, color, and size, can be used as landmarks for guidance, but may fail for VR scenes that are too cluttered or too bland.

High-level design goals

We distill feedback from the formative interviews into a set of high-level design goals:

DG1 - Static and stable VR view: Enable external users to quickly access, on demand, a static, and a stabilized visual of the VR environment. This will allow the external user to have a more detailed and unhurried look at the feed rather than a constantly moving view that is hard to keep track of.

DG2 - Support independent exploration: Allow external users to independently explore the different views of the scene despite where and what the VR user is currently looking at. Additionally, allow external users to independently access the recent past, without relying on the VR user to repeat actions.

DG3 - Augment conversation with spatial referencing: Enable external users to quickly and directly [37, 53] refer to scene elements in the VR view. The external user should be able to talk about 3D objects, interfaces, interaction mechanics, and the spatiality of the scene as if they are directly engaging with the VR experience.

DG4 - Support multi-modal context sharing: Provide communication beyond words, such as body language, actions, sketches, and scene contexts that are being observed by either the external or the VR user. These elements would provide a shared context and conversational grounding for both parties. Additionally, this could also help in elevating the feeling of co-presence amongst the users.

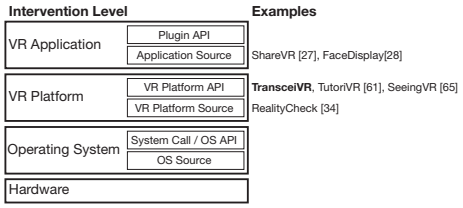

Figure 2: Implementation approaches can make interventions at different levels of the VR software stack.

A DESIGN SPACE OF IMPLEMENTATION APPROACHES

Our formative interviews revealed that participants had collaborative conversations about VR software that they or their organization had written and had source-code-level access to; as well as applications that were only available as closed source binaries (such as Blocks[4], or Quill[9]). To be broadly useful, we describe strategies to augment both open and closed-source applications. There are a number of different levels of programmatic access to a VR application (see Fig. 2):

1. VR Application source code access: For applications with source-level access, support for asymmetric communication can be included in the application, either by rewriting the application, or, more easily, by including a module that encapsulates collaboration functionality prior to compilation. For instance, one can incorporate the external user in the VR environment, by placing additional virtual cameras that can be controlled by and rendered to the external user. ShareVR [27] and FaceDisplay [28] follow this model.

2. VR application plug-in access: VR application developers may also choose to expose plug-in access in lieu of the source code, that can allow third party developers to implement similar features. While in desktop softwares, application-specific plug-in APIs are common (e.g., in Adobe Creative Suite, CAD softwares), we are unaware of any VR application that currently allows this.

3. VR Platform API access: For closed source applications that run on top of a VR platform, e.g., SteamVR, the API of this layer can be used to extract relevant runtime information and inject graphics into the running application (e.g., TutoriVR [61]). Major platforms currently allow API-level access to video feeds of both eyes, poses of VR hardware (headsets, controllers, trackers), and the ability to inject graphical overlays independently to each of the eyes. Such access makes it possible to render custom 3D elements in the VR scene of a closed application. Some projects also have modified and recompiled the platform API/Dll to add additional functionality (e.g., OpenVR in RealityCheck [34]).

4. Operating System (OS) level: An application might also operate at the operating system level; however, calls are likely

too low-level. As with VR plug-ins, we are not aware of projects that operate directly at this level. TransceiVR is built on the platform API level of access, which strikes a balance between universal applicability (it supports any application running on the platform API) and richness of application state that can be sensed and modified (video feeds and pose data, but no scene graph or application semantics).

TransceiVR SYSTEM

Using the design goals (DG) as guidelines, we developed TransceiVR to facilitate efficient communication between a VR user and an external user. An external user can view the scene of the VR user on a tablet device, and detach from the *VR mirror* paradigm by freezing any view (DG 1 - stable VR view) browsing previous views, either in space (*Angle frames*) or time (*Time frames*) (DG2 - independent exploration). They can annotate the scene on their 2D screen; annotations are then rendered at the correct depth in the VR user's 3D scene; (DG 3 - spatial referencing) or they can be shown as a *picture-in-picture* overlay (DG 4 - multi-modal context sharing). External users can indicate controller actions that a VR user should take in their interface; these are rendered on the virtual controller models in VR (DG 3). Finally, the VR user can also see a webcam video feed of the external user (DG 4).

To implement these functions, TransceiVR runs as an application overlay on the SteamVR platform. A number of VR systems such as Oculus, Vive and Windows Mixed Reality either support SteamVR or offer similar APIs. The platform API includes access to stereo-view feed of the VR user and the ability to inject 2D overlays and interfaces into the VR scene. Since these are platform level-operations, it allows TransceiVR to operate in an application-agnostic manner. The left and right eye feed, along with head tracking are used to compute a depth map of the scene and corresponding 3D coordinates of the different scene elements (in the frame of reference of the VR user's room). These are used to compute position of new 3D overlays, which are rendered using the platform's VR overlay injection interface. Fig. 3 gives an overview of these operations in the TransceiVR system.

External tablet interface

External users browse the VR scene and provide input through a tablet. The tablet is connected as a multi-touch external display to the PC on which the VR application is being run. Thus, our current implementation is incompatible with standalone VR headsets. The tablet is the primary interface for the external user, and accepts input using either a mouse or a stylus. We use an Apple Pen and iPad in our implementation. Note that, while we discuss TransceiVR in the context of collocated interactions, the system is equally functional with a remote external user who can access the interface via remote desktop software, coupled with a voice channel. By default, the tablet interface mirrors what the VR users sees in their environment.

Annotations

Our formative interviews indicate that referring to VR scene elements is difficult, requiring the external user to use detailed verbal instructions to direct the gaze and attention of the VR user. TransceiVR overcomes this hurdle by allowing the external user to directly draw into the VR scene (Fig. 4 (5)).

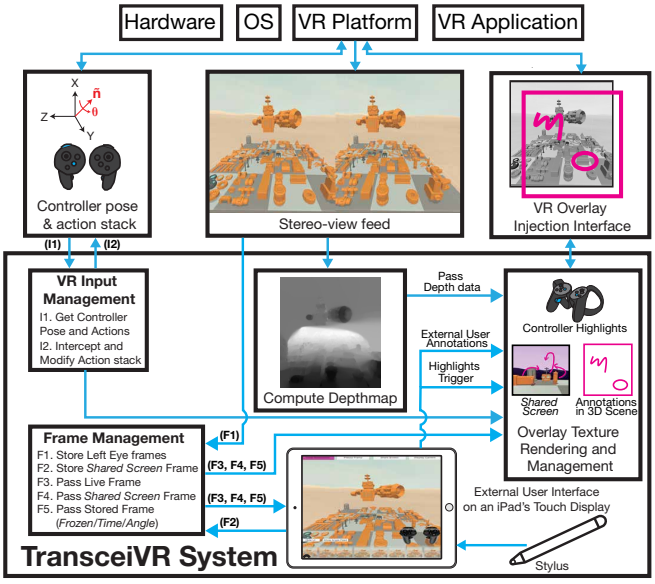

Figure 3: TransceiVR intervenes at the VR platform level, and assumes access to controller pose, actions and a stereo view feed. It uses the stereo view to compute a depth map of the scene. VR controller pose and actions are intercepted to enable the VR user's interactions with TransceiVR. The tablet interface is used by the external user to annotate, share screens, access frames and trigger highlights. All these data are used to render corresponding information as overlays in the VR scene, through the VR platform's overlay injection pipeline.

The tablet interface detects sketches and renders them in the VR scene at the appropriate depth and orientation, making it look like they are a part of the scene. This leverages findings from prior work [12, 13, 58] that suggest that, flat tablet surfaces provide an efficient interface to sketch elements into a VR scene. However, there are issues with this approach: 1) Continuous motion of the VR user's view makes it difficult to sketch over, and 2) The sketches are done on a 2D surface, but need to be rendered in the VR scene at the correct 3D position.

TransceiVR solves the first issue by temporarily freezing the mirrored view if the external user begins sketching on the tablet. This allows them to view and annotate a static, *frozen frame*. The second issue is less trivial to solve. In contrast to prior work where such interfaces were part of custom built applications, TransceiVR is designed to work in an application-agnostic manner with existing VR applications. These applications, do not provide access to the depth of elements in the scene, hence TransceiVR lacks direct access to it. Prior application-agnostic systems such as TutoriVR [61] have simply rendered the stereo-feed as a *3D TV display* in VR scene, without computing numerical depth values of points in the scene. In this work, we estimate scene depth using binocular disparity, and create a real-time depth map of the VR scene. We then use it to render annotations at the correct depth.

Depth map generation

From the computer vision literature for stereo images [33], we know that for a stereo camera setup containing two identical cameras that are separated only along the x-axis of the image plane, there exists a relationship between disparity d of two pixels p_1 and p_2 in the two images corresponding to the same world point P, the distance between the two cameras C, the focal length f, and the real-world depth Z of the point P

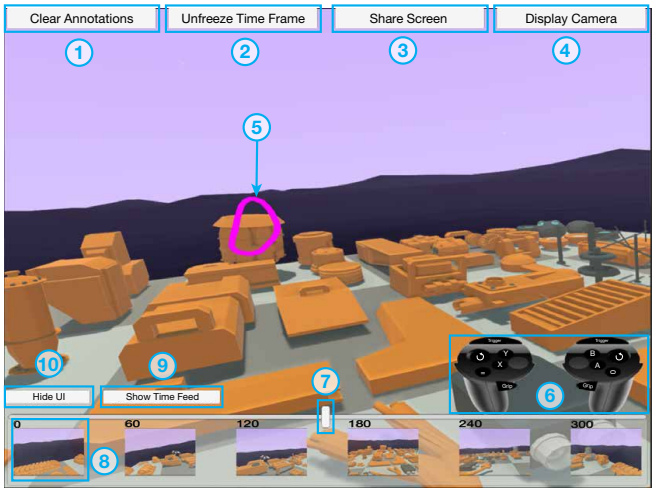

Figure 4: The iPad interface's elements; (1)-(4), (9), (10) - UI Buttons; (5) - A user drawn annotation; (6) - *Controller Panel*; (7) - VR user yaw direction indicator; (8) - *Angle frame* at 0° yaw. Note: UI Text emphasized for clarity.

through the following equation.

$$d = \frac{Cf}{Z} = Ap_x + B = \frac{K}{Z} \tag{1}$$

In our prototype system, the Oculus Rift renders with an asymmetric FOV [6], and in such a case, disparity d is a linear function of x-component of optical flow vectors $\mathbf{p}$ computed between left and right eye images (Fig. 5(R)). We use OpenCV's implementation [7] of the Brox optical flow algorithm [16] that can be executed on a GPU in real-time. This function varies depending on the specific projection matrix used for rendering the VR scene on a specific headset. Where $K = Cf$, and A and B are constants that vary depending on the specific VR hardware being used as well as the resolution of the VR feed that is being used for computing the optical flow. By applying appropriate transforms Eq. 1 can be rewritten with different constants as, Eq. 2. Where $p_{x\infty}$ is the x-component of the flow vector for a pixel at distance $Z \to \infty$.

$$Z = 10^{C_1} 10^{-M_1 \log(p_x - p_{x\infty})} \tag{2}$$

In order for TransceiVR to automatically compute the constants, $p_{x\infty}$ and M_1 and C_1 for different hardware, we created a calibration program that overlays a 2×2 grid of black and white squares at various fixed distances and positions in the world as shown in Fig. 5 (L), and use it to fit Eq. 2. We estimate $p_{x\infty}$ as the flow vector's x-component of an overlay placed at 100m which is treated as an infinite distance. Ideally, we should expect $M_1 = -1$ and $C_1 = 1$, but in practice deviations might occur due to the discretization of pixels and other factors. For our setup, parameter values were $M_1 = -0.9968046$, $C_1 = 1.119834$. Note that the optical flow vector values near the left edge of the image are not accurate—these are regions of the left eye's feed that are not seen by the right eye and are extrapolated by the Brox algorithm to obtain the flow vectors. Rendering annotations at correct depth instantly conveys the location of the desired point to the VR user, and ensures that the VR user sees the correct location even when they move around and change perspective. If an annotation is made outside the current FOV of the VR user, a 3D direction arrow appears in the scene and guides the VR user's gaze towards the direction

Figure 5: (L) The stereo feed of the calibration process being performed in SteamVR Home. (R) The normalized map of the flow vector x-components. The black box shows the region of a single calibration square across which the flow vector values are averaged.

of annotation until it appears in their FOV (Fig. 6(R)). When the annotations are no longer relevant, they can be cleared using a button in the tablet interface (Fig. 4 (1)).

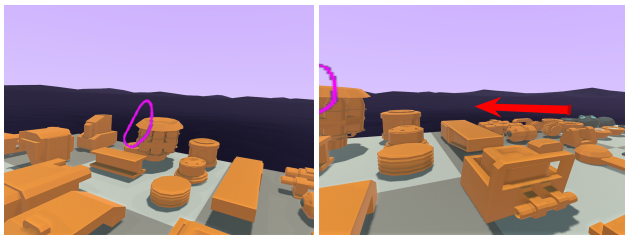

Figure 6: (L) A rendered annotation in the VR environment. (R) A 3D directional arrow indicator (red) appears when a newly created annotation is outside the field of view.

Actions with Controller

Sometimes the external user has to instruct the VR user about specific buttons and associated actions on the VR controller. This is crucial and common, when an expert user trains a novice VR user to use an application. This occurs in public demos, testing of early stage prototypes, workplaces, and home environments, where a frequent VR user gets an interested person to try it. Pointing to buttons of VR controllers is a special case of pointing to virtual objects in the VR scene, but it has some unique characteristics that require additional consideration: controllers are a primary interaction mechanism in VR applications and are almost always in motion. As identified by TutoriVR [61], they may not be present in the VR user's video feed either because they are outside the field of view or because the application does not render a virtual model of the controllers. In TransceiVR, a *Controller panel* (Fig. 4 (6)) is used for referring to controller buttons. The external user taps the button(s) of interest in the panel and colored blinking highlights appear over the location of the controller buttons in the VR scene that visualize these buttons to the VR user. In applications that do not render any virtual controllers, TransceiVR renders a proxy model in lieu of it.

Spatial and Temporal Exploration

In asymmetrical interactions, external users lack the ability to view and refer to virtual objects outside the VR user's current field of view. Additionally, referring to dynamic elements are tough due to their reduced temporal persistence in the VR feed. To aid with these, TransceiVR captures and stores the frames of the VR feed from the recent past (2 mins, one frame per second). External users can then visit a prior frame to examine it, or to discuss a virtual element that has vanished.

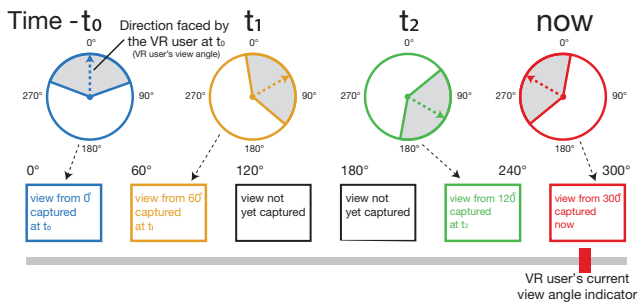

Figure 7: *Angle frames* (or view) are captured and updated when the VR user turns their head around and cross certain fixed angles

We term these frames as *Time frames*. TransceiVR also allows accessing frames based on VR headset's orientation angle (yaw angle), referred to as *Angle frames* (see Fig. 7). This allows external users to view and interact with last seen frames at specific view angles, and enables external users to get a glimpse of the VR environment around the VR user, annotate and refer to objects that are outside VR user's FOV. *Angle view* also supports egocentric directional guidance by providing the external user with a direction indicator(see Fig. 4 (7)). The indicator marks VR user's current view angle on an angle scale that is placed above the *Angle frames*. It provides a direct mapping for the external user, who can use this to guide VR user's relative movements, e.g. "Turn left/right," or more precisely as, "Turn about 30° to your right." Users can toggle between the two types of frames (Fig. 4 (9)).

Share screen

The communication between the two users may involve more than just directional references and pointing to objects. Users need to discuss, plan and execute actions in the VR scene. TransceiVR supports this requirement by allowing each user to share a static frame of the VR user's feed. Either of the users can share a copy of a static frame of the VR scene to the other user. The VR user invokes it using the hand controllers (Joystick button), while the external user can press a button on the tablet interface(Fig. 4 (3)). When shared, the frame appears on the the tablet interface for the external user and as an overlay display in the VR user's scene. The VR user can place it at a convenient location in the scene. The *shared screen* contains the annotations made by the external user and a cursor for the VR user. These can be seen by both users. While a frame is being shared, the live feed of the VR user is also shown as an inset in the tablet interface. The external user can choose to share a static *Frozen frame* from live feed, an *Angle frame* or a *Time frame*. Applications of the *share screen* include discussing actions, complex strokes or gestures. An example of using this to discuss tasks is seen in Fig. 8.

Viewing the External Environment

In longer interactions, experts have reported that the VR user can lose awareness of the external user and the environment. Prior work such as RealityCheck [34] focus on this issue from the viewpoint of safety and ability to interact with the physical environment. We are more concerned with increasing the interaction efficiency and the ability to communicate complex information such as gestures and body language, as well as the need to increase social engagement among the interacting users. TransceiVR allows the VR user to look at the external

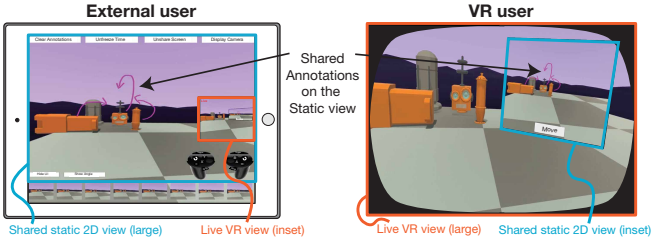

Figure 8: Example use of *share screen* to discuss and plan tasks - External user interface having an shared static view with annotations, having an inset of the live VR view (L) and VR user's feed with the movable display of the shared static view with annotations (R). Note: UI Text emphasized for clarity

user through a live camera feed that is rendered on an overlay display. This is accessed and viewed in a similar manner to the *share screen* feature (see Fig. 4 (3)). While other prior solutions like overlaying the 3D physical environment in the VR scene can be employed, these techniques require access to the VR application, and/or require sophisticated hardware setups that make them impractical for many users. *Share screen* and camera feed allow for higher fidelity information transfer, and can enhance the social engagement between users.

USER STUDY

We conducted a co-located user study to investigate if TransceiVR can effectively aid users in a scenario where an external user is guiding a VR user with an application designed for single-user usage. We compared the task completion time, error rate, and other subjective metrics in a within-subjects design comparing TransceiVR to the baseline condition—standard *VR mirror*.

Procedure

We recruited 20 users (8 female, 12 male, age range 18-57 years) in 10 pairs (VR user and external user), using university mailing lists. Within external users three had no prior experience with VR, while the rest had used it a few times in total. In the case of VR users, four had used VR occasionally (a couple of times per month), while the rest used VR regularly (multiple times a week). We used a within-subjects study design for the evaluation and counter balanced the order of conditions and tasks. The study duration was 60 minutes. Each user was compensated with a $25 Amazon gift card. Participants used the *Blocks* [4] 3D-modeling application to perform a robot assembly task, using a pre-existing set of building blocks (see Fig. 9a). They were asked to select and manipulate them, to assemble a given robot design (see Fig. 9b,c). We sourced over 100 parts for the task from Jarlan Perez's collection for *Bots with Blocks Challenge* [5] to closely match an existing task done using Google Blocks.

Participants were given an initial 5 minute training on usage of VR and safety instructions on sharing the physical space, and then a 5-8 minute training on the usage of the *Blocks* application and the TransceiVR interface. Pairs then completed the two robot design tasks, with each task in a different condition. Time limit per task was 12 minutes. The external user guided the VR user to assemble the robot. To position the external user as an expert, we provided the user with an assembly manual. Thus, to complete the task, the two users had to actively communicate. We added an additional more challenging sub-task towards end of the assembly, in which

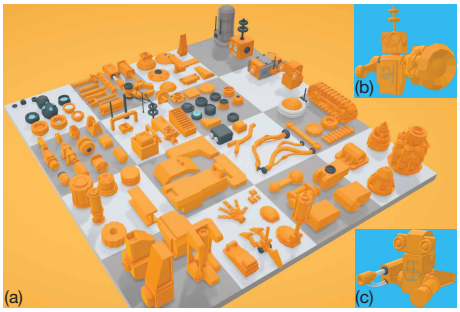

Figure 9: a) The set of building blocks provided to users for the tasks; b) and c) Final output of tasks

the VR user had to draw a specific foreign character (Tamil language) on the robot. None of our users knew the language. This sub-task simulates a scenario where the two users have to work together on something both of them have no prior expertise on.

Measures

We measured task success, task completion time and error rate (number of incorrect elements in the finished robot). After each task, participants rated their subjective experience on 5-point Likert scales. A first set of questions asked the user about the ease of understanding (for VR users) or conveying (for external users), the different aspects of communication (referred objects, where to look, conversational grounding, indicated controller buttons, complex strokes) (C1-C5 - see Fig. 12, top). Then they rated their level of agreement with a set of summary statements. (S1-S5 - see Fig. 12, bottom).

Participants also responded to the NASA-TLX instrument [32] after each condition that measured user's perceived workload, and the SUS [15] questionnaire that measured the usability. After the entire study, participants were asked to rate the usefulness of the different features of the system towards carrying out an efficient conversation with their partner. At the end, they provided open-ended feedback of the experience.

RESULTS

Pairs using TransceiVR outperformed pairs in the baseline condition in task success, task completion time, and error rate.

Task Success: When a pair completes all steps of the task correctly within the given time, the task performance is marked as a success. There was a significant difference in task success between TransceiVR (10 of 10 pairs succeeded) and the baseline (only 4 of 10 pairs) ($p<.05$, Fisher's exact test).

Task Completion Time: All pairs of participants took less time to complete tasks with TransceiVR compared to the baseline (incomplete tasks were stopped after the task time limit of 12 mins or 720s and counted as such). There was a significant difference in task completion time between TransceiVR (M=516.8s, SD=108.6061) and the baseline condition(M=678.1s, SD = 71.56) (paired sample t-test $t(9)=4.38$, $p<0.01$,$d=1.38$) - see Fig. 10.a.

Error Rate: Each task's robot assembly had 7 major parts; we counted the number of incorrect parts in the final output. There were significantly fewer errors in the TransceiVR condition (M=0.5, SD = 0.71) than in the baseline condition (M=1.8,SD=0.79) (paired sample t-test $t(9)=4.99$, $p<0.01$,$d=1.58$) - see Fig. 10.b. These errors were mostly due to placing parts in incorrect orientations, picking up wrong parts and incorrect strokes.

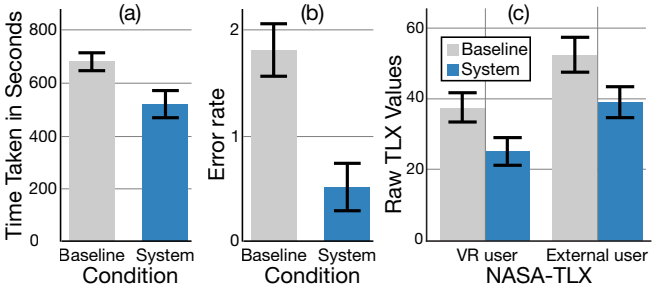

Figure 10: Quantitative measures (a,b): a) Time taken to finish, and b) no. of errors in final output (lower is better); Qualitative measure - c) Raw NASA-TLX for external and VR users (lower is better). Error bars show standard error, $SE = \sigma/\sqrt{n}$.

Subjective Ratings: Besides the differences in quantitative data, there were also statistically significant differences in qualitative data. Participant's ratings of perceived workload as measured by NASA-TLX was lower for TransceiVR (external user - M=39, SD = 13.97; VR user- M=25, SD = 12.57) than for the baseline condition for both users (external user - M=52.49, SD = 15.55; VR user- M=37.5, SD = 13.21) (paired sample t-test for external user - $t(9) = 3.71$, $p<0.01$, $d=1.17$; VR user - $t(9) = 3.45$, $p<0.01$, $d=1.09$) see Fig. 10.c.

The median pooled likert-scale ratings for questions C1-C5 about the ease of understanding by the VR user of different elements involved in communication was same in system condition (Median = 4) compared to baseline condition (Median = 4). However, a Wilcoxon signed-rank test still shows a significant difference ($W = 428$, $Z = 3.06$, $p<0.01$, $r = 0.43$) because the distribution skews more positive for TransceiVR. Similar analysis for the ease of conveying these information for external users, has a higher median for our system condition (Median = 4) compared to the baseline condition (Median = 3) and the Wilcoxon signed-rank test shows that there is a significant difference ($W = 613.5$, $Z = 4.56$, $p<0.01$, $r = 0.64$). These findings are further supported by participant's likert scale response to their agreement to statements S1-S5 mentioned earlier (see Fig. 11.b,c).

In the post-study questionnaire, participants rated the usefulness of individual components of TransceiVR. The highest-scoring component were the annotation system (Median=5) followed by the *Share Screen* (Median=4) and *Controller Panel* (Median=3). Verbal communication that complemented TransceiVR was also rated high (Median=5). Besides being useful, scores of the SUS questionnaire indicated that our system was usable by the VR user (M=75.5, SD=16.7) and the external user (M=69, SD=16.63). Corresponding SUS scores were less for the baseline for both the VR user (M=67, SD=20.13) and the external user (M=54.75, SD=14.6). A significant difference was found only for the external user (paired sample t-test - $t(9) = 2.3056$, $p<0.05$, $d = 0.73$).

Informal Expert Feedback

To gain additional qualitative insight into how TransceiVR might be used in a more realistic context, we tested TransceiVR with a VR expert in a remote setting and col-

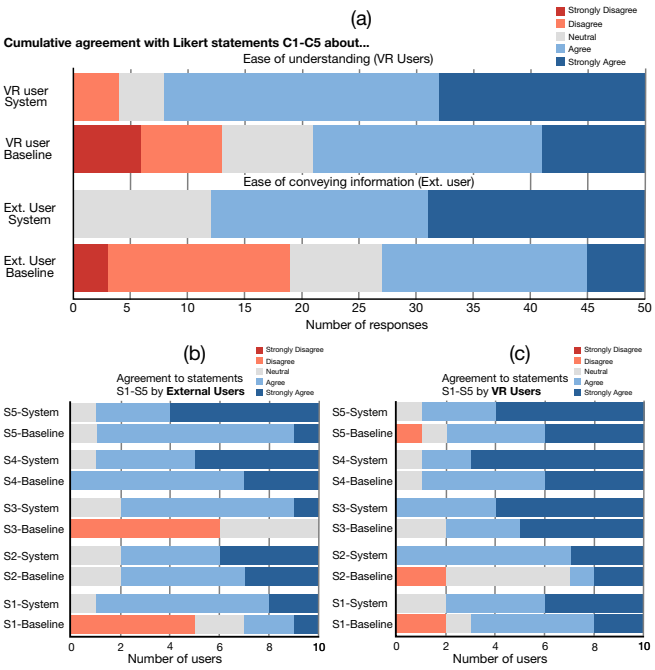

Figure 11: Qualitative measures. a) VR user ratings for ease of understanding and external user's rating for ease of conveying the five different elements involved in communication (higher is better) ; C1-5 (Ref Fig. 12). b) Likert Scale agreement to S1-5 (Ref Fig. 12) by external user and c) by VR user (higher is better).

lected informal feedback. The expert is a well-known VR artist whose work is featured in the Tiltbrush application, and leads a prominent VR art group. They taught Tiltbrush to a first-time VR user. The pair was asked to use TransceiVR over video conferencing (*Zoom*), with the expert operating our tablet interface which has screen sharing, input sharing, and audio. After receiving an initial orientation with TransceiVR, the expert spent 60 minutes to teach the novice basic commands, tools, and elementary 3D painting techniques in Tiltbrush.

Overall, the expert found that TransceiVR helped decrease their dependence on potentially ambiguous and problematic verbal descriptions compared to the status quo *VR mirror*: *"The fact that I had a real technical aid, along with it, I felt less beholden to my words because words can be really problematic, especially when trying to communicate within VR and in terms of instruction."*

The expert primarily relied on two features: the *controller panel*, and *share screen*. With the *controller panel*, the expert indicated to the novice, which buttons to press on each controller to invoke desired functions. This was essential as the novice had no prior experience with VR controllers. The expert used annotations on *shared screen* as the primary method of instruction. While annotating the VR scene directly seemed exciting to the expert, Tiltbrush starts with a blank canvas and thus has almost no objects in the environment to anchor annotations to. In addition, the expert preferred the instructions to appear alongside the novice in VR, rather than directly in the workspace, which might hinder novice's actions.

Finally, the expert also used *Time frames* and static *Frozen Frames*. Many operations in Tiltbrush require users to choose options in a menu. To help the novice with this task, the expert

	Likert Scale Statements for VR User: It was easy to understand each of the following....
	For External User: It was easy to convey each of the following to the VR user....
C1	Pointing out or referring to objects
C2	Directing where to look or turn
C3	Conversational grounding for actions and instructions
C4	Referring to controller buttons
C5	Communicate complex strokes and symbols
	Common Likert Scale Statements
S1	I was able to carry out efficient communication with my partner
S2	I found it easy to understand my partner's actions/words/instructions
S3	My partner found it easy to understand my actions/words/instructions
S4	The experience was engaging
S5	The experience was fun

Figure 12: The set of Likert Scale statements used. Questions S2 and S3 were borrowed from Harms et al.'s questionnaire[31] that measures perceived message understanding in a communication.

frequently selected specific *Time frames* where the menu options were present and screen-shared these frames with the VR user. While *Angle frames* was not used in this task, the expert commented that they could be useful in tasks that require guiding users towards specific spatial locations in a well-structured and populated environment. Besides the largely positive experience with the system, the expert suggested improvements to TransceiVR, mostly focused on improving the UI that could enhance the user experience - such as better feedback for transition across modes and visual distinction between different feature elements, being able to control timing of controller highlights to show the difference between hold, taps and double taps, and also providing haptic feedback on controllers when a button is referenced on it. However, in summary, they found the offered interactions "crucial" and mentioned that a tool like TransceiVR could open up new avenues of remote tutoring, guidance as well as "co-creation" in VR applications.

DISCUSSION

Overall, the quantitative results from our co-located study showed that TransceiVR can effectively aid users in an asymmetrical VR-external collaboration scenario. Pairs that used TransceiVR were able to complete tasks faster, and with better performance. When asked to rate the overall experience, eighteen our of twenty of our participants agreed that TransceiVR made the communication easy and efficient (Fig.11.b,c, S1 and S2). It shows that supporting effective communication was the key factor that led to TransceiVR's success. This finding is important because TransceiVR ran on top of an existing VR application that was designed only for single-user usage, and yet our participants were able to communicate and collaborate over it with ease. Participants found the experience to be fun and engaging (Fig.11.b,c S4 and S5) with lower mental workload (Fig.10.c). These findings confirmed the feasibility of our approach and contributed a new use case (e.g., asymmetrical communication) to the growing body of work on retrofitting closed-source VR applications [34, 61, 65].

Looking at individual features of TransceiVR and how they were used in the study, TransceiVR's annotation system received the highest rating from participants. The annotation system as a whole consists of annotating over *Time frames*, *Angle frames* and the *Frozen frame* of a live feed. These annotations are then rendered in the VR environment at the corresponding 3D position. In the assembly task of the user study, the external user, being the only one who is aware of how the desired part looks, needed to first locate it in the VR

scene. To do that, they used the *Angle Frames* feature and the live video feed to explore the scene on their own. Once they located the desired part, they annotated directly over it. When the annotation was not in the VR user's view, TransceiVR guided the VR user towards it using the 3D directional arrow (Fig.6). Thus, the combination of these tools augmented the external user's ability to understand the VR scene and to convey gestural and directional instructions. These added benefits facilitated more effective communication compared to the baseline system. Without TransceiVR, users reported it was challenging to observe the shaky VR feed, and resorted to using verbal communication, which often created confusion between them. We did not see any effect of the external user's local audio on user study task performance. This might be due to the goal-oriented and time-constrained nature of the assembly task. Also, the task in the user study, and the expert evaluation did not entail the use of external user's camera feed.

The next highly rated feature of the user study is the *Share Screen*. We designed this feature to complement the annotation system in situations where placing annotations directly on the environment of the VR user might be too distracting or create unnecessary clutters. With *Share Screen*, the external user can send a static image of the VR scene with annotations on that image. The static form of this feature allowed the VR user to place it at a convenient spot for their own reference as they continue with their own work. We observed heavy usage of *Shared Screen* during the character drawing step of the co-located user study, since directly annotating in the environment could get in the way of the VR user's drawing.

Another interesting use of TransceiVR was to combine *Shared Screen* and *Time Frames*. We observed this user pattern in the remote expert evaluation. In contrast to Blocks, the Tiltbrush environment of the expert evaluation was rather bland and did not have much scene landmarks for the in-environment annotations. Moreover, the VR painting lesson required the novice user to interact with highly dynamic and transient interface elements like menus, buttons, and strokes. In such scenarios, *Time Frames* complemented the use of *Shared Screen* because it can "freeze" these aforementioned dynamic elements. And then the expert user can use *Shared Screen* to communicate instructions to the novice user. As a result, the expert frequently used these two features to mark on menu options and on 3D strokes made by the novice user. The former helped the novice trigger the right tool, the latter helped improve the novice's stroke quality and techniques.

The least used and rated feature in the user study was the *Controller panel*. A reason for this is that, during the onboarding stage of the user study, the VR users received a controller orientation to use the *Blocks* application. In the expert evaluation, we observed heavy usage of this feature by the remote expert because the painting lesson required the novice user to learn a lot of different Tiltbrush menus, buttons and advanced shortcuts throughout the lesson.

We see that different features of TransceiVR have different tradeoffs. In our experience performing these evaluations and using TransceiVR ourselves, we realized that no feature by itself, is sufficient. Rather, a combination of features complementing each other in function is required, and this combination varies based on the characteristics of the VR scene and the type of information that needs to be communicated in it.

LIMITATIONS AND FUTURE WORK

A key interaction in TransceiVR is the annotation system. Due to limitations of the current SteamVR overlay injection pipeline, only 2D overlays (at arbitrary 3D pose) are supported. This prevents real-time rendering of 3D annotations in the VR scene. Moreover, TransceiVR does not track, when users in the VR scene teleport or change its scaling since these are application level changes that are not passed on to the VR platform. TransceiVR's annotations are rendered according to the system-level world space coordinates, and these go out of place when such interactions occur. To better handle such situations and increase the richness of annotations we recommend that VR platform APIs allow for rendering of dynamic 3D content, and provide access to information regarding the scale and pose of the main camera rendering the VR scene.

The experience of using TransceiVR remotely also raises several questions on how systems like TransceiVR should tailor functionality to specifically support remote asymmetric interaction. Through conducting several additional remote sessions in which an author assumed the role of an expert in VR communicating remote novices outside of VR, we uncovered several avenues for future work. Firstly, tools will have to intelligently make tradeoffs between latency, image resolution and framerate to work in realistic settings. Second, we have not yet investigated spatial audio; audio localization could help external users make statements such as "Turn towards me". Finally, we have thus far only focused on dyadic interactions, but remote access also enables larger groups of participants. This introduces new dynamics and challenges of control sharing across the different users.

CONCLUSION

In this paper, we identified challenges and goals of asymmetric interactions between VR users and external collaborators. We formulated design goals for systems focused on improving such interactions and presented a design space of implementation approaches. We chose a common scenario in the space, and built TransceiVR, a system that reduces communication barriers between collaborating VR and external users. We believe that important asymmetric VR tasks such as on-boarding, instruction, guidance, and co-creation in VR can all be effectively supported by TransceiVR. While our study focused on co-located user, we also tested these interactions remotely. Our initial experiences with remote asymmetric VR guidance point out important open areas for future work. We hope our findings can inspire future extension to multi-user settings, and richer multi-modal interactions.

ACKNOWLEDGEMENT

We thank the experts, study participants and the VR artist for their time. We thank Aiswarya Goutham Gouthaman, and Stephanie Claudino Daffara for their help with pilot testing; and, Richard Lin and Nived Rajaraman for their help with video assets. The robot parts used in the work were sourced from "Robot Kit v.1" by Jarlan Perez[5] (CC-BY 2.0 license). This work was supported in part by a Berkeley FHL Vive Center for Enhanced Reality seed grant and an Adobe gift.

References

[1] 2016. Black Hat Cooperative Game. (2016). https://www.teamfuturegames.com/

[2] 2017. Eye in the Sky - VinLia Games. (2017). https://www.vinliagames.com/

[3] 2017. Nemesis Perspective. (2017). https://www.evocatgames.com/nemesis-perspective/

[4] 2019. Blocks by Google. (2019). https://vr.google.com/blocks/

[5] 2019. Bots with Blocks Challenge. (2019). https://medium.com/@JarlanPerez/bots-with-blocks-challenge-50f3e1dde810

[6] 2019. Oculus Asymmetric Field of View. (2019). https://developer.oculus.com/documentation/quest/latest/concepts/unity-asymmetric-fov-faq/

[7] 2019. OpenCV Brox Optical Flow. (2019). https://docs.opencv.org/3.4.2/d7/d18/classcv_1_1cuda_1_1BroxOpticalFlow.html

[8] 2019. Panoptic Asymmetrical VR game. (2019). http://panopticgame.com/

[9] 2020. Quill. (2020). https://quill.fb.com/

[10] Matt Adcock and Chris Gunn. 2010. Annotating with 'Sticky' Light for Remote Guidance. In *ACM SIGGRAPH ASIA 2010 Posters (SA '10)*. ACM, New York, NY, USA, Article 62, 1 pages. DOI: http://dx.doi.org/10.1145/1900354.1900423

[11] Leila Alem and Jane Li. 2011. A Study of Gestures in a Video-mediated Collaborative Assembly Task. *Adv. in Hum.-Comp. Int.* 2011, Article 1 (Jan. 2011), 7 pages. DOI:http://dx.doi.org/10.1155/2011/987830

[12] Rahul Arora, Rubaiat Habib Kazi, Tovi Grossman, George Fitzmaurice, and Karan Singh. 2018. SymbiosisSketch: Combining 2D & 3D Sketching for Designing Detailed 3D Objects in Situ. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. ACM, New York, NY, USA, Article 185, 15 pages. DOI: http://dx.doi.org/10.1145/3173574.3173759

[13] Rahul Arora, Rubaiat Habib Kazi, Fraser Anderson, Tovi Grossman, Karan Singh, and George Fitzmaurice. 2017. Experimental Evaluation of Sketching on Surfaces in VR. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. ACM, New York, NY, USA, 5643–5654. DOI: http://dx.doi.org/10.1145/3025453.3025474

[14] Oliver Bimber and Ramesh Raskar. 2005. *Spatial Augmented Reality: Merging Real and Virtual Worlds*. A. K. Peters, Ltd., Natick, MA, USA.

[15] John Brooke and others. 1996. SUS-A quick and dirty usability scale. *Usability evaluation in industry* 189, 194 (1996), 4–7.

[16] Thomas Brox, Andrés Bruhn, Nils Papenberg, and Joachim Weickert. 2004. High Accuracy Optical Flow Estimation Based on a Theory for Warping. In *Computer Vision - ECCV 2004*, Tomás Pajdla and Jiří Matas (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 25–36.

[17] Li-Te Cheng, Steven L. Rohall, John Patterson, Steven Ross, and Susanne Hupfer. 2004. Retrofitting Collaboration into UIs with Aspects. In *Proceedings of the 2004 ACM Conference on Computer Supported Cooperative Work (CSCW '04)*. Association for Computing Machinery, New York, NY, USA, 25–28. DOI:http://dx.doi.org/10.1145/1031607.1031612

[18] Elizabeth F Churchill and Dave Snowdon. 1998. Collaborative virtual environments: an introductory review of issues and systems. *Virtual Reality* 3, 1 (1998), 3–15.

[19] Sarah D'Angelo and Andrew Begel. 2017. Improving Communication Between Pair Programmers Using Shared Gaze Awareness. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. ACM, New York, NY, USA, 6245–6290. DOI: http://dx.doi.org/10.1145/3025453.3025573

[20] Morgan Dixon and James Fogarty. 2010. Prefab: implementing advanced behaviors using pixel-based reverse engineering of interface structure. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 1525–1534.

[21] James R. Eagan, Michel Beaudouin-Lafon, and Wendy E. Mackay. 2011. Cracking the Cocoa Nut: User Interface Programming at Runtime. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology (UIST '11)*. Association for Computing Machinery, New York, NY, USA, 225–234. DOI:http://dx.doi.org/10.1145/2047196.2047226

[22] Mike Fraser, Steve Benford, Jon Hindmarsh, and Christian Heath. 1999. Supporting Awareness and Interaction Through Collaborative Virtual Interfaces. In *Proceedings of the 12th Annual ACM Symposium on User Interface Software and Technology (UIST '99)*. ACM, New York, NY, USA, 27–36. DOI: http://dx.doi.org/10.1145/320719.322580

[23] Taichi Furukawa, Daisuke Yamamoto, Moe Sugawa, Roshan Peiris, and Kouta Minamizawa. 2019. TeleSight: Enabling Asymmetric Collaboration in VR Between HMD User and Non-HMD Users. In *ACM SIGGRAPH 2019 Emerging Technologies (SIGGRAPH '19)*. ACM, New York, NY, USA, Article 26, 2 pages. DOI: http://dx.doi.org/10.1145/3305367.3335040

[24] Susan R. Fussell, Leslie D. Setlock, and Robert E. Kraut. 2003. Effects of Head-mounted and Scene-oriented Video Systems on Remote Collaboration on Physical Tasks. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '03)*. ACM, New York, NY, USA, 513–520. DOI: http://dx.doi.org/10.1145/642611.642701

[25] Google. 2019. TiltBrush. (2019). http://www.tiltbrush.com/ Visited 19-Sep-2019.

[26] Jan Gugenheimer, Christian Mai, Mark McGill, Julie Williamson, Frank Steinicke, and Ken Perlin. 2019. Challenges Using Head-Mounted Displays in Shared and Social Spaces. In *Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems (CHI EA '19)*. ACM, New York, NY, USA, Article W19, 8 pages. DOI: http://dx.doi.org/10.1145/3290607.3299028

[27] Jan Gugenheimer, Evgeny Stemasov, Julian Frommel, and Enrico Rukzio. 2017. ShareVR: Enabling Co-Located Experiences for Virtual Reality Between HMD and Non-HMD Users. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. ACM, New York, NY, USA, 4021–4033. DOI: http://dx.doi.org/10.1145/3025453.3025683

[28] Jan Gugenheimer, Evgeny Stemasov, Harpreet Sareen, and Enrico Rukzio. 2018. FaceDisplay: Towards Asymmetric Multi-User Interaction for Nomadic Virtual Reality. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. ACM, New York, NY, USA, Article 54, 13 pages. DOI: http://dx.doi.org/10.1145/3173574.3173628

[29] Pavel Gurevich, Joel Lanir, Benjamin Cohen, and Ran Stone. 2012. TeleAdvisor: A Versatile Augmented Reality Tool for Remote Assistance. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '12)*. ACM, New York, NY, USA, 619–622. DOI: http://dx.doi.org/10.1145/2207676.2207763

[30] William A. Hamilton, Oliver Garretson, and Andruid Kerne. 2014. Streaming on Twitch: Fostering Participatory Communities of Play Within Live Mixed Media. In *Proceedings of the 32Nd Annual ACM Conference on Human Factors in Computing Systems (CHI '14)*. ACM, New York, NY, USA, 1315–1324. DOI:http://dx.doi.org/10.1145/2556288.2557048

[31] Chad Harms and Frank Biocca. 2004. Internal consistency and reliability of the networked minds measure of social presence. (2004).

[32] Sandra G Hart and Lowell E Staveland. 1988. Development of NASA-TLX (Task Load Index): Results of empirical and theoretical research. In *Advances in psychology*. Vol. 52. Elsevier, 139–183.

[33] Richard Hartley and Andrew Zisserman. 2003. *Multiple View Geometry in Computer Vision* (2 ed.). Cambridge University Press, New York, NY, USA.

[34] Jeremy Hartmann, Christian Holz, Eyal Ofek, and Andrew D. Wilson. 2019. RealityCheck: Blending Virtual Environments with Situated Physical Reality. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19)*. ACM, New York, NY, USA, Article 347, 12 pages. DOI: http://dx.doi.org/10.1145/3290605.3300577

[35] Weidong Huang, Leila Alem, and Franco Tecchia. 2013. HandsIn3D: Augmenting the Shared 3D Visual Space with Unmediated Hand Gestures. In *SIGGRAPH Asia 2013 Emerging Technologies (SA '13)*. ACM, New York, NY, USA, Article 10, 3 pages. DOI: http://dx.doi.org/10.1145/2542284.2542294

[36] Weidong Huang, Mark Billinghurst, Leila Alem, and Seungwon Kim. 2018. HandsInTouch: Sharing Gestures in Remote Collaboration. In *Proceedings of the 30th Australian Conference on Computer-Human Interaction (OzCHI '18)*. ACM, New York, NY, USA, 396–400. DOI:http://dx.doi.org/10.1145/3292147.3292177

[37] Edwin L Hutchins, James D Hollan, and Donald A Norman. 1985. Direct manipulation interfaces. *Human-computer interaction* 1, 4 (1985), 311–338.

[38] Shahram Izadi, Harry Brignull, Tom Rodden, Yvonne Rogers, and Mia Underwood. 2003. Dynamo: A Public Interactive Surface Supporting the Cooperative Sharing and Exchange of Media. In *Proceedings of the 16th Annual ACM Symposium on User Interface Software and Technology (UIST '03)*. Association for Computing Machinery, New York, NY, USA, 159–168. DOI: http://dx.doi.org/10.1145/964696.964714

[39] Shunichi Kasahara and Jun Rekimoto. 2014. JackIn: Integrating First-person View with Out-of-body Vision Generation for Human-human Augmentation. In *Proceedings of the 5th Augmented Human International Conference (AH '14)*. ACM, New York, NY, USA, Article 46, 8 pages. DOI: http://dx.doi.org/10.1145/2582051.2582097

[40] David Kirk and Danae Stanton Fraser. 2006. Comparing Remote Gesture Technologies for Supporting Collaborative Physical Tasks. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '06)*. ACM, New York, NY, USA, 1191–1200. DOI: http://dx.doi.org/10.1145/1124772.1124951

[41] Jan Kolkmeier, Emiel Harmsen, Sander Giesselink, Dennis Reidsma, Mariët Theune, and Dirk Heylen. 2018. With a Little Help from a Holographic Friend: The OpenIMPRESS Mixed Reality Telepresence Toolkit for Remote Collaboration Systems. In *Proceedings of the 24th ACM Symposium on Virtual Reality Software and Technology (VRST '18)*. ACM, New York, NY, USA, Article 26, 11 pages. DOI: http://dx.doi.org/10.1145/3281505.3281542

[42] Hideaki Kuzuoka. 1992. Spatial Workspace Collaboration: A SharedView Video Support System for Remote Collaboration Capability. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '92)*. ACM, New York, NY, USA, 533–540. DOI: http://dx.doi.org/10.1145/142750.142980

[43] Pascal Lessel, Alexander Vielhauer, and Antonio Krüger. 2017. Expanding Video Game Live-Streams with Enhanced Communication Channels: A Case Study. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. ACM, New

York, NY, USA, 1571–1576. DOI: http://dx.doi.org/10.1145/3025453.3025708

[44] Sebastian Marwecki, Maximilian Brehm, Lukas Wagner, Lung-Pan Cheng, Florian 'Floyd' Mueller, and Patrick Baudisch. 2018. VirtualSpace - Overloading Physical Space with Multiple Virtual Reality Users. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. ACM, New York, NY, USA, Article 241, 10 pages. DOI: http://dx.doi.org/10.1145/3173574.3173815

[45] Michael Venturino Maxwell J. Wells. 1990. Performance and head movements using a helmet-mounted display with different sized fields-of-view. *Optical Engineering* 29, 8 (1990), 870 – 877 – 8. DOI:http://dx.doi.org/10.1117/12.55672

[46] Emerson Murphy-Hill, Da Young Lee, Gail C. Murphy, and Joanna Mcgrenere. 2015. How Do Users Discover New Tools in Software Development and Beyond? *Comput. Supported Coop. Work* 24, 5 (Oct. 2015), 389–422. DOI: http://dx.doi.org/10.1007/s10606-015-9230-9

[47] Cuong Nguyen, Stephen DiVerdi, Aaron Hertzmann, and Feng Liu. 2017. CollaVR: Collaborative In-Headset Review for VR Video. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology (UIST '17)*. ACM, New York, NY, USA, 267–277. DOI: http://dx.doi.org/10.1145/3126594.3126659

[48] Ohan Oda, Carmine Elvezio, Mengu Sukan, Steven Feiner, and Barbara Tversky. 2015. Virtual Replicas for Remote Assistance in Virtual and Augmented Reality. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology (UIST '15)*. Association for Computing Machinery, New York, NY, USA, 405–415. DOI: http://dx.doi.org/10.1145/2807442.2807497

[49] Doug Palmer, Matt Adcock, Jocelyn Smith, Matthew Hutchins, Chris Gunn, Duncan Stevenson, and Ken Taylor. 2007. Annotating with Light for Remote Guidance. In *Proceedings of the 19th Australasian Conference on Computer-Human Interaction: Entertaining User Interfaces (OZCHI '07)*. ACM, New York, NY, USA, 103–110. DOI: http://dx.doi.org/10.1145/1324892.1324911

[50] Karine Pires and Gwendal Simon. 2015. YouTube Live and Twitch: A Tour of User-generated Live Streaming Systems. In *Proceedings of the 6th ACM Multimedia Systems Conference (MMSys '15)*. ACM, New York, NY, USA, 225–230. DOI: http://dx.doi.org/10.1145/2713168.2713195

[51] Suporn Pongnumkul, Mira Dontcheva, Wilmot Li, Jue Wang, Lubomir Bourdev, Shai Avidan, and Michael F. Cohen. 2011. Pause-and-Play: Automatically Linking Screencast Video Tutorials with Applications. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology (UIST '11)*. Association for Computing Machinery, New York, NY, USA, 135–144. DOI: http://dx.doi.org/10.1145/2047196.2047213

[52] Manolis Savva, Nicholas Kong, Arti Chhajta, Li Fei-Fei, Maneesh Agrawala, and Jeffrey Heer. 2011. ReVision: Automated Classification, Analysis and Redesign of Chart Images. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology (UIST '11)*. ACM, New York, NY, USA, 393–402. DOI: http://dx.doi.org/10.1145/2047196.2047247

[53] Ben Shneiderman. 1997. Direct Manipulation for Comprehensible, Predictable and Controllable User Interfaces. In *Proceedings of the 2Nd International Conference on Intelligent User Interfaces (IUI '97)*. ACM, New York, NY, USA, 33–39. DOI: http://dx.doi.org/10.1145/238218.238281

[54] Thomas Smith, Marianna Obrist, and Peter Wright. 2013. Live-streaming Changes the (Video) Game. In *Proceedings of the 11th European Conference on Interactive TV and Video (EuroITV '13)*. ACM, New York, NY, USA, 131–138. DOI: http://dx.doi.org/10.1145/2465958.2465971

[55] David N. Snowdon, Elizabeth F. Churchill, and Alan J. Munro. 2001. Collaborative Virtual Environments: Digital Spaces and Places for CSCW: An Introduction. In *Collaborative Virtual Environments*, Elizabeth F. Churchill, David N. Snowdon, and Alan J. Monro (Eds.). Springer, London, Chapter 1, 3–17.

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

[57] Maximilian Speicher, Jingchen Cao, Ao Yu, Haihua Zhang, and Michael Nebeling. 2018. 360Anywhere: Mobile Ad-hoc Collaboration in Any Environment Using 360 Video and Augmented Reality. *Proc. ACM Hum.-Comput. Interact.* 2, EICS, Article 9 (June 2018), 20 pages. DOI:http://dx.doi.org/10.1145/3229091

[58] 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 2019 CHI Conference on Human Factors in Computing Systems (CHI '19)*. ACM, New York, NY, USA, Article 13, 13 pages. DOI: http://dx.doi.org/10.1145/3290605.3300243

[59] Theophilus Teo, Louise Lawrence, Gun A. Lee, Mark Billinghurst, and Matt Adcock. 2019. Mixed Reality Remote Collaboration Combining 360 Video and 3D Reconstruction. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19)*. ACM, New York, NY, USA, Article 201, 14 pages. DOI:http://dx.doi.org/10.1145/3290605.3300431

[60] Balasaravanan Thoravi Kumaravel, Fraser Anderson, George Fitzmaurice, Bjoern Hartmann, and Tovi

Grossman. 2019a. Loki: Facilitating Remote Instruction of Physical Tasks Using Bi-Directional Mixed-Reality Telepresence. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology (UIST '19)*. Association for Computing Machinery, New York, NY, USA, 161–174. DOI: http://dx.doi.org/10.1145/3332165.3347872

[61] Balasaravanan Thoravi Kumaravel, Cuong Nguyen, Stephen DiVerdi, and Björn Hartmann. 2019b. TutoriVR: A Video-Based Tutorial System for Design Applications in Virtual Reality. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19)*. ACM, New York, NY, USA, Article 284, 12 pages. DOI:http://dx.doi.org/10.1145/3290605.3300514

[62] Julius von Willich, Markus Funk, Florian Müller, Karola Marky, Jan Riemann, and Max Mühlhäuser. 2019. You Invaded My Tracking Space! Using Augmented Virtuality for Spotting Passersby in Room-Scale Virtual Reality. In *Proceedings of the 2019 on Designing Interactive Systems Conference (DIS '19)*. ACM, New York, NY, USA, 487–496. DOI: http://dx.doi.org/10.1145/3322276.3322334

[63] Haijun Xia, Sebastian Herscher, Ken Perlin, and Daniel Wigdor. 2018. Spacetime: Enabling Fluid Individual and Collaborative Editing in Virtual Reality. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology (UIST '18)*. ACM, New York, NY, USA, 853–866. DOI: http://dx.doi.org/10.1145/3242587.3242597

[64] Tom Yeh, Tsung-Hsiang Chang, and Robert C Miller. 2009. Sikuli: using GUI screenshots for search and automation. In *Proceedings of the 22nd annual ACM symposium on User interface software and technology*. 183–192.

[65] Yuhang Zhao, Edward Cutrell, Christian Holz, Meredith Ringel Morris, Eyal Ofek, and Andrew D. Wilson. 2019. SeeingVR: A Set of Tools to Make Virtual Reality More Accessible to People with Low Vision. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–14. DOI:http://dx.doi.org/10.1145/3290605.3300341