RealityCheck: Blending Virtual Environments with Situated Physical Reality

Jeremy Hartmann[*]
Microsoft Research
Redmond, WA, USA

Christian Holz
Microsoft Research
Redmond, WA, USA

Eyal Ofek
Microsoft Research
Redmond, WA, USA

Andrew D. Wilson
Microsoft Research
Redmond, WA, USA

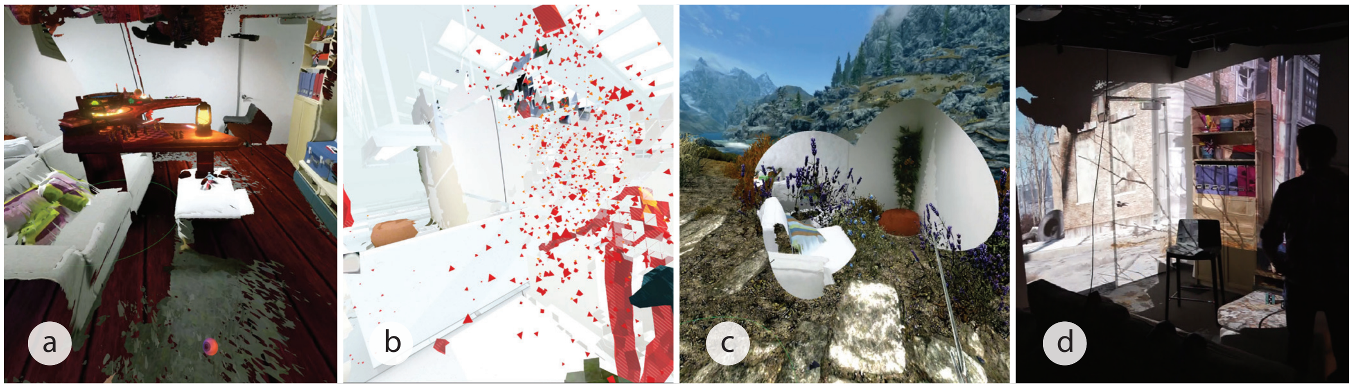

Figure 1: RealityCheck blends the VR player's real world with the virtual world. A real time 3D reconstruction of the player's room is integrated into the VR title's rendering pipeline to allow: a) proper 3D hidden surface removal (*Waltz of the Wizard*), b) collision estimation (*SUPERHOT VR*), c) a flashlight into reality using controllers (*Skyrim VR*), and d) co-located spectatorship through projection mapping (*Fallout 4 VR*).

ABSTRACT

Today's virtual reality (VR) systems offer chaperone rendering techniques that prevent the user from colliding with physical objects. Without a detailed geometric model of the physical world, these techniques offer limited possibility for more advanced compositing between the real world and the virtual. We explore this using a realtime 3D reconstruction of the real world that can be combined with a virtual environment. RealityCheck allows users to freely move, manipulate, observe, and communicate with people and objects situated in their physical space without losing the sense of immersion or presence inside their virtual world. We demonstrate RealityCheck with seven existing VR titles, and describe compositing approaches that address the potential conflicts when rendering the real world and a virtual environment together. A study with frequent VR users demonstrate the affordances provided by our system and how it can be used to enhance current VR experiences.

[*]Also with University of Waterloo.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

CHI 2019, May 4–9, 2019, Glasgow, Scotland UK

© 2019 Copyright held by the owner/author(s). Publication rights licensed to ACM.

ACM ISBN 978-1-4503-5970-2/19/05…$15.00
https://doi.org/10.1145/3290605.3300577

KEYWORDS

virtual reality, depth cameras, 3D compositing

1 INTRODUCTION

Today's virtual reality (VR) systems such as the Oculus Rift, HTC Vive, and Windows Mixed Reality, aim to completely immerse the user in a virtual environment. However, such immersion comes at the cost of the user's awareness of their physical surroundings. Simple tasks, such as picking up small objects, moving within a physical space, or communicating with someone in the room become difficult if not impossible. Current systems render a 3D grid that appears whenever the user comes into close proximity to a predefined boundary. This is seen in both the Oculus Rift's "guardian" and the HTC Vive's "chaperone" systems. Equipped with a color camera, the HTC Vive offers a variant of the grid chaperone where an outline of the real world is rendered (composited) on top of the virtual environment (see Figure 2). Though these approaches help prevent unintended collisions, they employ a very simple 3D model of the room which is assumed to be static, with the floor clear of obstructions such as furniture, people and pets. We speculate that many VR users would often choose some other form of entertainment rather than clear their living room of obstacles. Meanwhile, late 2017 saw what is perhaps the first VR-related death, as reported by TASS: "According to preliminary information, while moving around the apartment in virtual reality glasses, the man tripped and crashed into a glass table, suffered wounds and died on the spot from a loss of blood"[1].

We introduce *RealityCheck*, a system that exploits a real-time 3D reconstruction of the user's environment to enable the combination of the real and virtual worlds that goes beyond state of the art chaperone systems. With *RealityCheck*, the real world is embedded inside the virtual world as if the application rendered the physical world natively within the scene (Figure 1). The live reconstruction of the physical environment is obtained through two different approaches. In the first, we equip the physical environment with eight RGB-D cameras (Microsoft Kinect v2) that are positioned inside a physical environment to reconstruct a geometric representation of the world in real-time. In the second, we equip a RGB-D camera (Intel RealSense) directly onto the HMD and reconstruct a live view of the physical environment from the user's perspective.

RealityCheck modifies the graphics rendering pipeline of existing VR titles that rely on OpenVR. Once reference to the back buffer and z-buffer are acquired, multiple means of blending the real with the virtual are possible. For example, the player's couch may appear around them, correctly occluding virtual objects, allowing the player to safely take a seat during gameplay. Meanwhile, a non-player character (NPC) may correctly appear in front of the player's ottoman as it approaches. Since the rendering of the physical environment is dynamically updated, people or objects placed inside the environment will also appear inside the virtual scene. This allows for ad hoc manipulation of objects and for communicating with someone else in the room without removing the headset.

RealityCheck makes the following contributions:

- A new approach to blending the real world with the virtual world in VR, in which a 3D reconstruction of the real world is composited with the virtual world in the usual graphics rendering pipeline
- A prototype implementation that works with existing VR titles, without modification
- Several variations of the basic compositing technique that explore the interaction between real and virtual world geometry

[1]http://tass.com/society/982465

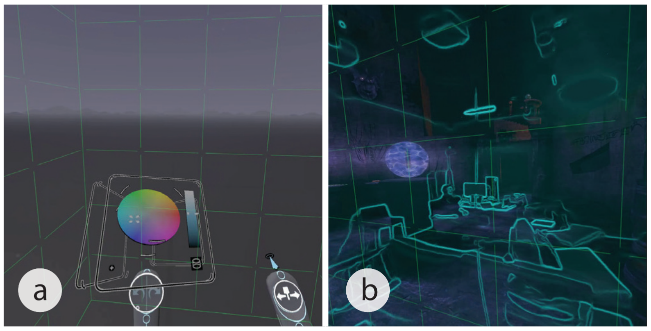

Figure 2: Current chaperone system implemented by HTC Vive: a) grid chaperone (*Tilt Brush*), and b) line overlay chaperone (*Waltz of the Wizard*).

- Demonstration of the approach with multiple hardware configurations: external and internal cameras, and external display (projection mapping) for spectators
- A user study that compares our system to state of the art chaperone techniques

2 RELATED WORK

The main idea of RealityCheck lies in the merging of the real and the virtual. Milgram and Kishino describe various versions of this blending in their virtuality continuum [25]. Many extensions to this have been proposed, usually through the addition of axes orthogonal to the AR-VR continuum that provide new insights into aspects of human computer interaction [11, 20, 24]. A variation on this is *Virtualized Reality* [14], which involves creating a virtual copy of the real world. Our work pushes this further and explores the application of blending in the context of situated physical reality. Here we discuss work in blending in VR that relate to the dimensions of interaction, communication, and object avoidance.

Blending Reality

A number of recent works explore a variety of ways to combine the real and virtual worlds, where it is often advantageous to manipulate either the virtual world or the real world representation. In the reconstruction of the physical space for the purpose of scene generation, Reality Skins [33] uses the real world as a blueprint to build up a virtual scene, matching the real world against predefined geometric objects. Generating procedural worlds is explored by Sra et al. [36], converting large spaces into 3-D environments. Meanwhile, direct manipulation of a scene reconstruction is explored in Remixed Reality [18].

Several works investigate the transition of the user's attention between the real world and the virtual. McGill et al. [23] identified several usability challenges with VR users interacting with object in the real world, such as typing on a real keyboard, or being aware of someone entering the room. They contribute a prototype that uses green screen compositing to blend the keyboard into VR. As in the present work, they demonstrate multiple blending techniques, including the area around the user's hands. In another study, they use a depth camera to segment video of people in the room, which is then composited in VR. ShareVR [8] enables communication and interaction between non-HMD wearing users and a virtual environment. The systems described here offer some affordances related to communication and interaction, however they implement pre-defined experiences and cannot integrate with arbitrary virtual worlds. JackIn Space [16] investigates the transition between third and first person views in the context of telepresence. OneReality [32] describes a holistic design space, blending the user's presence across many levels of virtuality, while Steptoe et al. investigated the effects of non-photorealistic rendering on immersion [37].

The blending of haptics with virtual environments can enhance interaction with a virtual space, making it feel more "real". For examples, the use of real world proxies as substitutes for virtual objects [9, 34]. Haptic redirection is explored in work by Azmandian et al. [1], while Zhoa and Follmer demonstrate continuous retargeting between virtual and real objects [45].

Object Avoidance

One goal of our system is to allow the user to freely move within their virtual environment without fear of unintended collisions with external objects. Current chaperone systems assume that the space is free from obstacles. There will always be some conflict between the physical and virtual world, and several works look at ways to diminish this. For example, Virtual Space [21] creates a set of design guidelines and an API in order to share a physical space with multiple VR users.

Redirected walking is one way to resolve conflict with the physical world [27, 39]. The goal of these systems is to create virtual spaces that feel larger through the subtle manipulation of the virtual scene. Recent work uses saccades to make the user oblivious to these changes [38]. Avoidance of other co-located user is explored by Lacoche et al. [17], where different in-game representations of the player are compared.

Room-scale experiences

Room-scale experiences have been explored in the context of spatial augmented reality (SAR) [12, 13, 28] and in VR [3, 18]. These systems provide the user with custom interactive experiences tailored to a large interactive space. Our system also works at room scale, but seeks to integrate a pre-existing virtual space with the real space.

3 REALITYCHECK

VR systems are able to render highly detailed environments that allow users to explore vast worlds. While VR technology will continue to improve, providing ever more immersive experiences, problems relating to the isolation of the user from the real world will remain. RealityCheck investigates these issues along three dimensions: 1) mitigating risk through increased awareness, 2) communication and spectatorship, and 3) physical manipulation.

Mitigating the Risk of Collision

RealityCheck enables an enhanced chaperone system by integrating parts of the user's real world into the their virtual world in various ways (Section 5). When the blending is based on the distance to the user's head and controllers, a minimal chaperone is possible that only shows the physical environment when the user is at risk of collision to both static and dynamic objects. Current VR environments also use a chaperone system to help guide the user (Figure 2). However, current solutions require a sufficiently large play area (e.g. minimum area for the HTC Vive room-scale experience is $1.5\text{m} \times 2\text{m}$) where the perimeter of the space must be manually defined. While a grid is shown when the user is within a certain distance, any obstacles intruding or within its static and predefined border are not considered. In contrast, our chaperone is dynamic and does not need to be manually defined as the real world is used directly.

Outside Communication

A VR user is physically present but removed from their immediate context, making interpersonal engagement difficult to initiate. RealityCheck provides a mechanism to merge dynamic objects into the virtual context of the user to help eliminate these kind of communication barriers. We use a method of background-subtraction to render salient objects such as a person crossing the room (section 5). Recent works have explored the use of vibro-tactile sensors to alert the VR user of a nearby presence [19], or use 2D blending techniques to bring a stenciled video of a person to the foreground [23]. In contrast, we explore communication under two contexts. First, from the perspective of the VR user, and second, from the perspective of a co-located person. Further, our approach has the advantage of rendering real people in the room as if they were part of the virtual environment, while not requiring the VR application developer to add special support.

Physical Manipulation

We tend to interact with the physical items around us. Research into the manipulation of real objects while in VR has largely come in the forms of haptics [1, 22], substitution [9, 34], or input [15, 23]. Supporting everyday physical

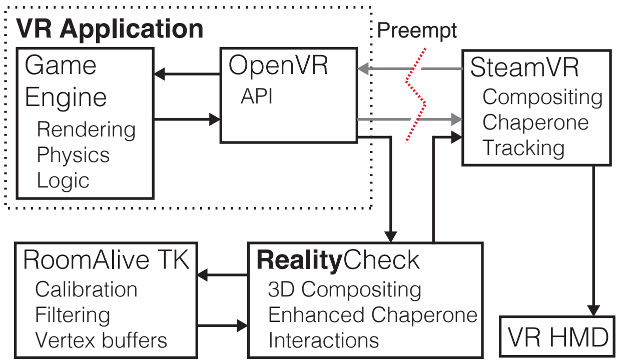

Figure 3: Overview of the RealityCheck system.

interaction around the user has not been fully explored, for example the VR user could simultaneous grab a plate of snacks, a drink, move from their desk, and sit down on a nearby couch. RealityCheck provides mechanisms for these kind of interactions to take place. By bringing in part of the real world at appropriate moments, the user's virtual environment can become easier to use and be more enjoyable.

Overall, RealityCheck looks to address issues of safety, communication, and the physical manipulation of objects by merging the real world inside a virtual one. This allows for proper 3D hidden surface removal, blending, interactions, and integration within a virtual environment. We demonstrate compositing techniques that are agnostic to the underlying game implementation, in fact, no development requirements or API integrations are required to use our system; and showcase their use in the transition between real and virtual worlds.

4 COMPOSITING REAL AND VIRTUAL WORLDS

In describing our system, we take as a starting point the problem of compositing a 3D model of the user's real physical environment into the rendered 3D graphics of a VR application. Such a 3D model may be collected from an array of depth cameras arranged around the user's space ("outside in"), or from a depth camera mounted on the user's head mounted display ("inside out"), for example. In either case, we presume that the 3D model of user's real physical environment has been calibrated to align with the native coordinate system of the VR system: in practice, this can be accomplished by a procedure in which the tracked VR controllers can be located in the reconstructed model of the room.

Given the 3D nature of both the virtual and real worlds, it is natural that a real object should appear in front of any virtual objects that are further away, and vice versa. In a traditional 3D graphics pipeline, occlusion of one object by another is accomplished during rendering by updating the *z-buffer*, a texture which records at each pixel location the distance of the nearest surface rendered thus far. In the rendering process, a given pixel's color is updated only if the currently rendered geometry falls at a point nearer than the value recorded in the z-buffer.

To perform our own rendering of real world geometry in a running VR title, we require access to its final rendered output for each eye, as well as the z-buffer, view and projection matrix used in the rendering process. Additionally, some modifications may make use of other information such as the pose of each VR controller.

Furthermore, we would like to demonstrate the broad applicability of our approach by using it with games that are popular with VR users today. To this end we developed a software framework (Figure 3) that allows the modification of the rendering process of an existing VR application without requiring the application's source code. It uses well-known techniques to replace or "hijack" calls to system APIs involved in rendering and VR compositing. Such techniques are popular with the game hacking and modding community, and have also been a useful tool in building research prototypes on otherwise unmodified software systems [10][31]. We plan to open source this part of the RealityCheck system.

Exploiting OpenVR

OpenVR[2] is an API that VR applications use to communicate with SteamVR[3]. An application obtains HMD and controller pose from SteamVR via OpenVR calls, and provides SteamVR with final rendered frames for each eye by calling `IVRCompositor::Submit`. To add our own graphics to the VR's final rendered graphics, we replace OpenVR's dynamic link library (DLL) with a custom DLL which similarly implement's OpenVR's `Submit` call but includes additional routines to render our real world geometry. This is done by modifying and recompiling OpenVR's open source.

The calls to OpenVR's `Submit` provide a convenient means of injecting our own code to render onto each eye's final output, but it does not provide access to the z-buffer. In a typical frame, the application will call `ID3D11DeviceContext::OMSetRenderTargets` many times throughout its rendering process. Some of these calls will be to set the color backbuffer and z-buffer that we require for our own rendering. To find the application's z-buffer, we first obtain the application's DirectX device by examining a texture passed to `Submit`, and then modify its C++ vtable to intercept all calls to `OMSetRenderTargets`.

Unfortunately there is no direct means to determine which of the render targets provided to `OMSetRenderTargets` is the final rendering and z-buffer we require. Furthermore, the

[2]https://github.com/ValveSoftware/openvr
[3]https://steamcommunity.com/steamvr

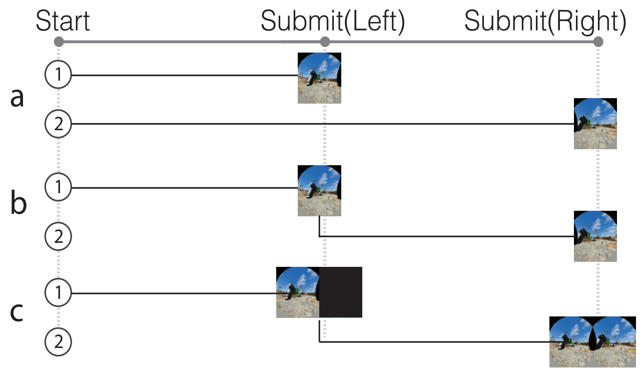

Figure 4: Three likely rendering pipelines used by VR applications. a) single eye textures, b) single texture shared between each eye, and c) stereo texture.

application may use render targets in different ways: 1) each eye may be rendered to separate textures, 2) each eye may be rendered separately but to the same texture which is reused, and 3) both eyes may be rendered into the same texture, side by side (Figure 4). When each eye is rendered separately, there is similarly no direct way to determine whether a given render target corresponds to the left or right eye. When both eyes are rendered into the same texture, there is no direct way to determine if both eyes have been rendered.

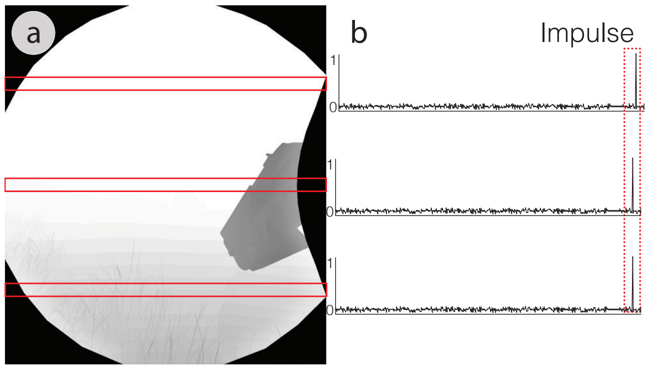

Figure 5: A compute shader analyzes the render target received through `OMSetRenderTargets`. a) the raster lines tested in the analysis, b) the impulse used in eye identification.

We employ a variety of heuristics and image processing techniques to resolve these ambiguities and find our render targets. For example, a compute shader is used to classify candidate z-buffers as left or right eye views based on the stencil pattern (Figure 5). When both eyes are rendered into the same texture, symmetry of the image is a reliable indication of whether both eyes have been rendered. Once determined, references to the correct render targets are cached, and no further analysis is performed. In the case where the application renders eyes separately but reuses the render target, copies are made during rendering.

To perform our rendering, we also require the view and projection matrices used by the VR application in its own rendering of each eye. The view matrix is easily obtained from OpenVR, which provides the position and rotation of the user's HMD. This must be updated every frame to match the user's head pose in game. Obtaining the projection matrix is more difficult. OpenVR provides the projection matrix for each eye via a function call that takes the near and far plane values used by the application. Presently, we determine the application's near and far plane values empirically and retrieve each eye's projection matrix through OpenVR. With the view and projection matrix, we render the real world inside the virtual scene and make a submission on behalf of the game to SteamVR. This allows us to take advantage of advanced post-rendering techniques such as asynchronous reprojection and motion smoothing.

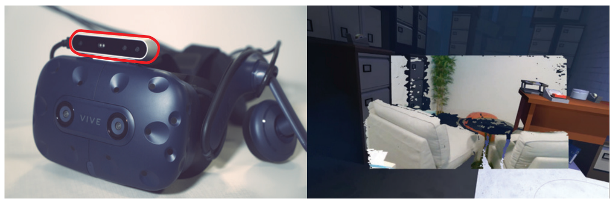

Figure 6: HTC Vive with Intel RealSense highlighted in red (left). The composited depth image with a virtual scene (right).

Data Acquisition and Calibration

The RealityCheck system is agnostic to what can be rendered on a submitted frame. In our implementation, we use the RoomAlive Toolkit [43] for both the room scale reconstruction and for the RealSense[4] head mounted camera (see Figure 6).

Our room scale deployment uses eight Kinect v2 depth cameras calibrated using the RoomAlive [43] calibration procedure, where five projectors display Gray codes onto the physical surfaces to resolve the poses and positions of all projectors and cameras. The depth and color data is compressed using RVL [42] and JPEG compression respectively and sent over a local Ethernet connection to the client. The RoomAlive RealSense server works similarly.

The compressed data is received by the RoomAlive client, in which the depth and color data is decompressed and smoothed with a bi-lateral filter. The depth image is then converted into a DirectX vertex buffer for rendering.

5 COMPOSITING TECHNIQUES

We use the system described in Section 4 to create a variety of game-agnostic compositing techniques that demonstrate the flexibility of our approach across the categories of blending,

[4]https://realsense.intel.com/

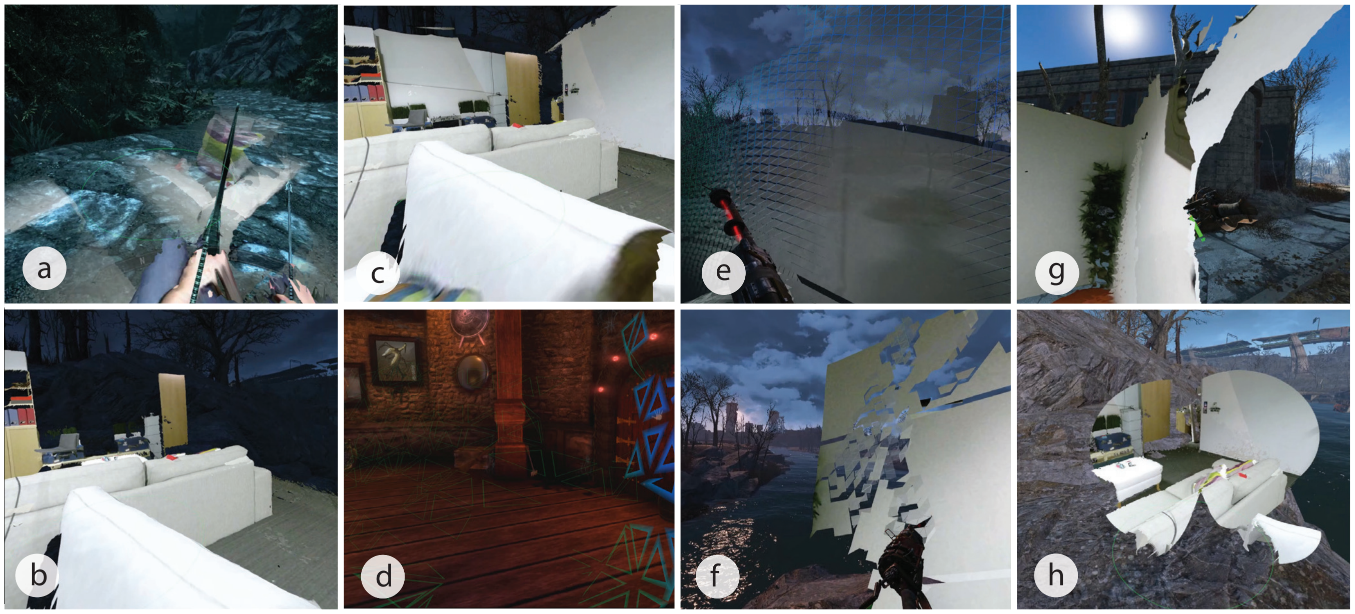

Figure 7: Techniques implemented with the RealityCheck system. a) alpha blending, b) salient objects (no Walls), c) full environment, d) texture abstraction (mesh outlines), e) polygon manipulation (floating polygons), f) collision estimation, g) mesh erasing, h) flashlight into reality.

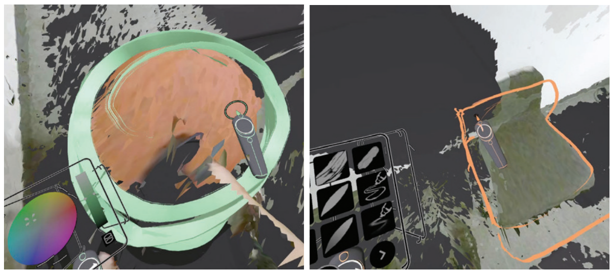
Figure 8: Real world guides: a) an ottoman being used to create a circle, and b) a chair being used as a reference (*Tilt Brush*).

texture and geometry manipulation, and interaction. We test these techniques within seven different VR titles available on Steam: Accounting, Waltz of the Wizard, SUPERHOT VR, Tilt Brush by Google, Blocks, Fallout 4 VR, and Skyrim VR.

Blending

We explore blending in the context of the full environment, only salient objects, and objects that are nearby.

Full environment blending. All available real world geometry is composited into the virtual environment (Figure 7c). While in a room, walls and furniture will occlude everything outside the physical environment. Applications that are more productivity focused may benefit from this level of blending, such as *Tilt Brush* and *Blocks*. Users are able to navigate the real world while simultaneously able to design, draw, or build within it. This offers some interesting affordances, such as using a physical object as a reference while modeling a 3D object (Figure 8).

Salient object blending. Objects that are important to a room's composition, such as furniture, people, or pets (Figure 7b). This level of blending allows a user to freely move around the space with minimal amount of hidden surface removal on the virtual scene, and can be implemented through the removal of predefined objects (i.e. walls) or by showing only objects that have changed from one frame to the next (i.e. background-subtraction). Small tasks, such as manipulating objects on a desk, eating food, or drinking can easily be performed.

Proximity blending. Objects are selectively blended based on proximity to the user's head and hand positions (Figure 7a). This level of blending has minimal impact on the visual coherence of the virtual scene, since only portions of the physical world appears within a certain distance (1m). Proximity blending can act like an advanced chaperone system with the added benefit of also showing dynamic objects (e.g,. people, pets, chairs, etc.).

Texture

Changes to the underlying physical texture can be used to make the rendering of the real world more similar or different than the virtual world (Figure 7d).

Color Transfer. When compositing real world geometry into a virtual world, it may be desirable to light the real world to

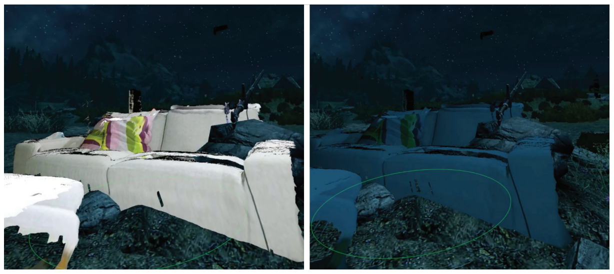
Figure 9: The real world environment before color transfer (left) and after the transfer (right) in the game *Skyrim VR*.

match that of the virtual world (see Figure 9). For example, if the player enters a dark cave, the rendering of their couch should be similarly dark. We use an approach that is both fast and effective in modulating the color of real world geometry to match that of the virtual rendering. We adapt the statistical methods by Reinhard et al. [30] using parallel reduction techniques on a compute shader.

Color statistics are calculated in the CIEL*A*B* color space. The method uses the global illumination (μ) and standard deviation (σ) from a source image (I_s) to transform a target image (I_t) to match the distributions found in each L*A*B* color channel. Every pixel in I_t is scaled by a ratio between the standard deviation of the target (σ_t) and the standard deviation of the source (σ_s), giving:

$$I_t' = \frac{\sigma_t}{\sigma_s}(I_t - \mu_t) + \mu_s \tag{1}$$

This transformation is easily implemented on a compute shader.

Abstraction. Rendering the real world with a very different rendering style can make it clear to the user which parts of the world are real and which are virtual. Rendering the real world as a wireframe or with other stylistic effects (Figure 7d) can also allow the user see through real world objects, which may be important for gameplay.

Geometry

Manipulations of the real world geometry may be useful when creating effects where the physical world appears to react to the virtual. Further, the abstraction of geometry can be used to incorporate artistic renderings of objects that approximate the original [9, 34].

We demonstrate a geometric effect based on the proximity of the user (Figure 7e). As the user moves around the space, the physical environment is reconstructed around them in real time. The user sees individual polygons float down and assemble at their feet, playing with their senses of reality.

In-game collision Estimation. Interaction between the physical world and in-application content is also possible. We use the system implementation in Section 5 to detect intersections between the real world and the virtual scene (Figure 7f).

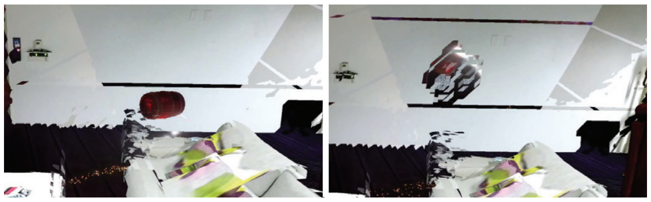
Figure 10: A before (left) and after (right) shot of a barrel colliding with a composited wall (*Waltz of the Wizard*).

For example, when the user shoots an arrow or throws a barrel, the real world will react by breaking apart around the point of impact (Figure 10). These kinds of interactions make the real world seem alive and part of the virtual world.

We estimate collisions between the application and the composited live mesh data by comparing the real world depth of a rendered pixel against the corresponding z-buffer point. Z-buffer values are normalized by the application's projection matrix and may be converted back to world coordinate depths by inverting part of the projection matrix:

$$z' = \frac{nf}{f - zf + zn} \tag{2}$$

where n and f are near and far plane values, respectively. With the z-buffer projected into the system's coordinate space, collision with the applications geometry is approximated by computing the distance between the depth buffer's point and the corresponding point in the live mesh data. The rendering process may use this distance to appropriately modify the rendering of the real world geometry. For example, it may move the real world geometry out of the way.

Interactions

Giving the user full control over aspects of blending, object saliency, or recoloring may be useful when insufficient information is available to infer correct parameters or when greater control over compositing is desired.

Aspects of the real world can be dynamically shown and hidden to the user as they are inside their virtual world, but there are times when giving control over what objects persist and what do not is important. For example, a user may always want to know where their computer desk is within the virtual world. Our system realizes this by allowing users to bring in or remove the real world by drawing over them with their controllers, giving them a granularity of control over what is seen and what is not (Figure 7g)

A flashlight into reality (Figure 7h) can be used as portal into the real world. A user moves the controller, like a flashlight, to reveal sections of the real world. All geometry in the real world that reside within the solid angle emitted from the controller is revealed to the user.

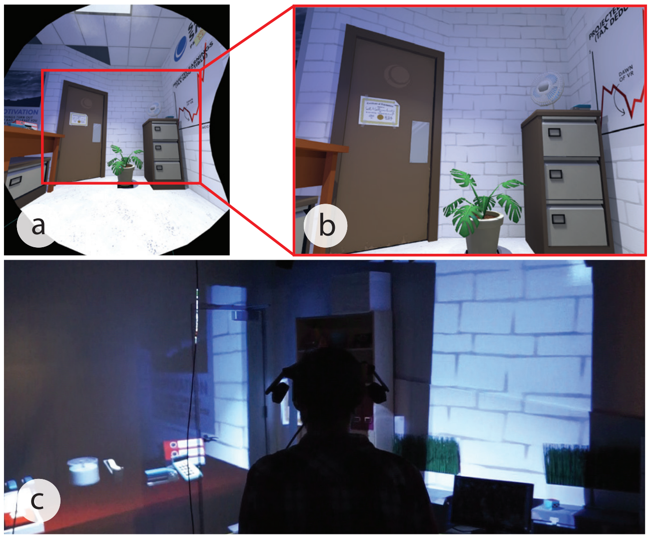

Figure 11: The HMD user's in-game view projection mapped onto physical surfaces from a co-located spectators point of view. a) the left eye view, b) the companion window view, and c) the projection mapped content.

Spectatorship

As part of RealityCheck, we implemented a projection mapping system where a view from the game is projected onto real physical surfaces from the perspective of a co-located viewer (Figure 11). For immersion, it is important that the mapping from the game to the physical surface is spatially stable, where a point in the game corresponds to a single point on the surface during rotation. This is accomplished by obtaining the rotation matrix from the HMD, the head position of the co-located user, and an adjusted field-of-view (FOV) for the projected content.

The HMD's orientation is retrieved from OpenVR. The head position of the co-located user is determined by a mean shift method [6] on the aggregated skeletal data from the eight Kinect sensors in the room.

During a VR session the game displays a companion window on the users desktop (Figure 11b). The window is a subregion of the left-eye texture sent to the HMD (Figure 11a). Since the companion window is free of stencil marks, this is used as the projection mapped content. The FoV for the projection mapped content is then a ratio between the width of the companion window and the width of the left eye texture, ensuring that points in the game correspond to points in the real world.

6 USER EVALUATION

This within-subject experiment evaluates immersion, safety, physical manipulation, and communication between our proposed 3D compositing techniques and the Vive's built in chaperone system. We chose three techniques outlined in

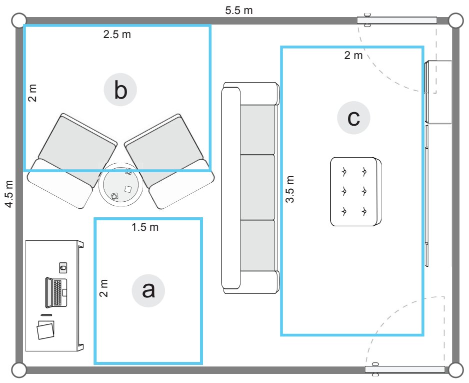

Figure 12: Room layout used in user study. a) office space, b) recreational area, and c) living room area.

Section 5 that are representative points along a continuum of blending (i.e. little to all of reality): 1) full reality (FULL) where all of physical reality is blended, 2) salient objects (SALIENT) where the furniture and dynamic objects are retained, and 3) proximity blending (PROXIMITY) where only a portion of reality around the user is retained.

Each blending level is assigned a virtual environment, physical space, and task in order to replicate realistic VR scenarios. The baseline for comparison is Vive's chaperone grid (GRID) and line overlay (LINE) where the outline of objects are rendered on top of the scene. Ordering of the tasks were counter balanced with a Latin square. In summary, three blending levels (PROXIMITY, SALIENT, and FULL) and two baselines (GRID and LINES).

We recruited 12 participants, ages 28 to 40, 2 female. All participants were right-handed, and most of them use a VR devices a few times a year. The study lasted for approximately 60 minutes: 15 minutes per blending level and an additional 15 minutes of surveys.

The RealityCheck system described in Section 4 is used in a room environment that is approximately 4.5m × 5.5m × 2.5m (Figure 12). The room is split into three sections to replicate common VR configurations. Dimensions were determined by examining Steam's annual hardware surveys. The first room-scale environment (Figure 12a) replicates a home office containing an area of 1.5m × 2m. The factors FULL and GRID are compared. The second replicates a medium recreational area (Figure 12b) containing two sofa chairs and a small table, room-scale dimensions are 2.5m×2m. The factors PROXIMITY and GRID are compared. The third area replicates a large living space (Figure 12c), with a full couch, foot stool, cabinets, and TV stand. Room-scale dimensions are 3.5m × 2m. The factors SALIENT and LINES are compared.

Task and Procedure

We asked participants to play a VR application within each of the three mock environments outlined in previous section.

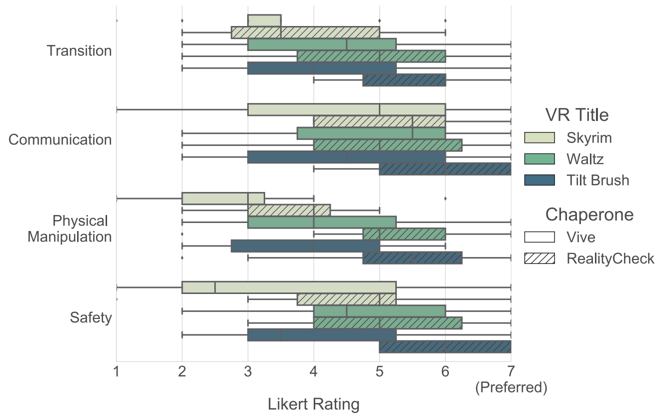

Figure 13: Likert ratings across all participants. A comparison between the Vive's chaperone system (no hatch) and the RealityCheck system (hatch) across three VR titles.

In the mock office environment, the task was to use *Tilt Brush*, a 3D drawing application, to replicate a stool that was located just outside the chaperoned off area. Participants were free to move within the allotted area but not outside it. In the mock recreation room, the task was to play the game *Skyrim VR* by Bethesda Studios and travel from Riverwood to Whiterun, fictional towns within the game. They were asked to switch between a standing and sitting posture every minute. In the mock living space, the task was to explore the world of *Waltz of the Wizard*, an exploration game set inside a fictional wizard's home. At one minute intervals, the participant was asked to pick up a block on the couch and place it on the ottoman in the center of the chaperoned area.

At the beginning of the experiment, participants filled out a short survey regarding their current VR usage. At the end of each VR session, a short 7-point Likert questionnaire (7 = preferred) asked participants to reflect on their experience and the assigned tasks in the categories of safety, physical manipulations, communication, and their transition between the real and virtual worlds. A six part SUS [41] questionnaire followed. Finally, a post experiment survey was conducted asking about their overall experience. A researcher was in the room with the participant for the duration of the study and all instructions were verbally communicated.

Results

Aligned Rank Transform (ART) [44] and post-hoc t-tests with FDR [2] correction are used for all non-parametric data. An overview of the results are outlined in Figure 13. On average, RealityCheck saw an increase in average scores across all game titles in each of the categories. A significant effect on Transitions ($F_{5,55} = 4.83, p < .001$), Physical Manipulation ($F_{5,55} = 4.98, p < .001$), and Safety ($F_{5,55} = 5.11, p < .001$) are observed. There is no effect on Communication.

On Saftey, a post-hoc test shows FULL (5.90) is perceived safer then GRID (3.81) for *Tilt Brush*. On the transitions between the physical and virtual world, FULL for *Tilt Brush* (5.54) is perceived easier then GRID (3.54) for *Skyrim VR*. Finally, on the physical manipulation of real world objects, FULL for *Tilt Brush* is perceived easier then GRID for *Skyrim VR*. No other significance is reported ($p > .082$).

The mean SUS scores are higher for RealityCheck across all games when compared with the baseline chaperones. An ART analysis shows a significant effect ($F_{5,380} = 6.62, p < .001$). Post-hoc t-tests suggest PROXIMITY (4.76) for *Skyrim VR* is perceived as more immersive than GRID (3.47) for *Tilt Brush*. The immersiveness of SALIENT (4.93) for *Waltz the Wizard* is perceived greater than both LINES (4.93) for *Waltz of the Wizard* and GRID (3.47) for *Tilt Brush*. There is no significant result for FULL (4.27) on immersion.

Discussion

We found compelling difference between the three different blending levels and the baseline chaperone. Among all three game titles, Tilt Brush has the largest reported difference across the four questions posed to the user. This is likely due to the contrast between the environment in game and the physical environment merged with it. Surprisingly, there is no significant results for communication, though the mean scores are higher then the baseline.

Participant responses to the the grid and line overlay chaperone showed a generally negative sentiment, stating that grid chaperone "breaks the sense of virtual reality" (P11) and "makes the world less immersive" (P4). On the line overlay participants stated that it "does not work very well for me" (P3) and it "made depth perception slightly more confusing" (P11). In contrast to this, P6 and P8 stated that they preferred the line overlay when compared with the blending techniques. However, none of the participant preferred the grid chaperone and stated that it felt like a "virtual cave" (P5).

On the use of real world blending, participants generally thought is was useful. Stating that it "felt very immersive" and "[s]itting down was definitely more comfortable" (P5). Surprisingly, many participants expressed that the blending techniques seemed more immersive to them over the baseline chaperones, stating it "felt more like a virtual reality" (P12). Though, the blending was sometimes seen as problematic when participants were not able to differentiate between virtual objects and real ones. One participant stated, "there was a bookshelf from a real world, which I thought was a vr bookshelf".

7 LIMITATIONS

While RealityCheck provides new ways to render content within a virtual world, there are still several challenges related to performance, visual fidelity, and OpenVR integration.

Today's VR systems typically aim to render each eye at 90Hz. The 3D compositing and spectatorship components of RealityCheck use the GPU extensively, making this framerate goal more difficult to achieve. Ultimate framerate will be a function of the rendering demands of the VR title, the complexity of the RealityCheck shaders, the number of cameras employed in the system, and, when spectatorship is enabled, the number of projectors. For example, without culling based on visibility, each projector must render all cameras.

The SteamVR title Accounting, for example, easily renders at 90Hz (less than 11.1ms rendering time) with one camera on modern hardware. Our most complex configuration of eight Kinect v2 cameras and five projectors with spectatorship enabled, renders at 15ms, which causes SteamVR to render at 45Hz on Skyrim VR. SteamVR then doubles this framerate with its "motion smoothing" feature, whereby every other frame is synthesized by interpolation. The overall subjective experience is still good.

The visual fidelity of the system depends on the current depth cameras being used, and their calibration and alignment. Incorporating advanced filtering and reconstruction algorithms could help [4, 5, 40], however latency and frame rate need to be considered for a real-time system. The combination of highly accurate but sparse LIDAR sensors with highly dense depth cameras could be an alternative approach to increasing overall quality. Another approach would be to rely on depth cameras to perform coarse hidden surface removal, while otherwise relying on head mounted RGB cameras, such as those available on the HTC Vive, to render the color texture of the real world.

With the room scale deployment of cameras, latency of the acquisition of depth and color data has no discernible impact on the ultimate rendering when the scene is largely composed of stationary objects such as furniture. Meanwhile, the head-mounted version introduces a noticeable latency in rendering of the real world.

Our systems sits on top of OpenVR, which allows us to modify the application's rendering. However, retrieving the applications near and far planes in a generalizable manner remains unsolved. Extending the OpenVR API to include the submission of the near and far planes could be one solution to alleviate this.

8 DISCUSSION AND FUTURE WORK

RealityCheck aims to merge reality with a virtual environment for the purposes of safety, communication, and interaction. Our system works with existing VR applications and enables new ways to engage with the real world while inside the virtual world. RealityCheck is a platform and concept that will enable interesting directions for future work.

Haptics and geometric mappings. Our current implementation of RealityCheck focuses on manipulating the real world geometry to react to changes in the virtual world. Further enhancements may be possible by considering manipulating the virtual geometry as well. Methods that use raw mesh data [7] and semantic scene understanding [35] might be used to modify the virtual world to match the real in a meaningful way. Further, techniques such as change blindness or saccades [38] could be used to make these changes imperceptible to the user. With methods to align the real and virtual, passive haptics could be used to enhance the experience. For example, aligning a in-game wall to match a real physical wall or positioning a couch to align with its in-game counter part.

Pushing this concept further, aspects of the real world could deviate from reality to match the style and tone of the game. Methods like in RealitySkin [33] could be used to create visually compelling scenes or create alternatives to the "Home" application currently used to launch applications.

Treating the game as a depth camera. Currently, we extract the depth and RGB data from the game for use in our compositing techniques. It may be instructive to think of this data as produced by RGB-D camera. Computer vision research suggests many uses of depth cameras, such as in SLAM [26] or using it for in-game object detection and segmentation [29]. These methods could be used for extended types of interaction. For example, reconstructing the game world through SLAM techniques or extracting in-game avatars to superimpose onto a co-located person in the room.

Recommendation. RealityCheck relies on the abilty to obtain the game's color back buffer, z-buffer and view and projection matrices through recompiling the OpenVR DLL and vtable injection on DirectX. Future versions of such APIs could make these components more readily available, encouraging researchers and developers to create novel experiences that expand the state of the art in VR.

9 CONCLUSION

In this paper we presented RealityCheck, a system that builds on top of the current VR rendering pipeline. We demonstrate the capabilities of our system through a number of techniques that integrate the real world inside a virtual scene. A user study further demonstrated its ability to enhance current VR environments. We believe this approach enables applications in safety and awareness as well as creating more meaningful VR experiences. We see this work as the first step towards allowing the seamless transition between two realities.

REFERENCES

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

[2] Yoav Benjamini and Yosef Hochberg. 1995. Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing. *Journal of the Royal Statistical Society. Series B (Methodological)* 57, 1 (1995), 289–300. http://www.jstor.org/stable/2346101

[3] Lung-Pan Cheng, Thijs Roumen, Hannes Rantzsch, Sven Köhler, Patrick Schmidt, Robert Kovacs, Johannes Jasper, Jonas Kemper, and Patrick Baudisch. 2015. TurkDeck: Physical Virtual Reality Based on People. *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology - UIST '15* (2015), 417–426. https://doi.org/10.1145/2807442.2807463

[4] Mingsong Dou, Jonathan Taylor, Pushmeet Kohli, Vladimir Tankovich, Shahram Izadi, Sameh Khamis, Yury Degtyarev, Philip Davidson, Sean Ryan Fanello, Adarsh Kowdle, Sergio Orts Escolano, Christoph Rhemann, and David Kim. 2016. Fusion4D: Real-time Performance Capture of Challenging Scenes. *ACM Transactions on Graphics* 35, 4 (jul 2016), 1–13. https://doi.org/10.1145/2897824.2925969

[5] Ruofei Du, Ming Chuang, Wayne Chang, Hugues Hoppe, and Amitabh Varshney. 2018. Montage4D: Interactive Seamless Fusion of Multiview Video Textures. In *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games - I3D '18*, Vol. 11. ACM Press, New York, New York, USA, 1–11. https://doi.org/10.1145/3190834.3190843

[6] Keinosuke Fukunaga and L. Hostetler. 1975. The estimation of the gradient of a density function, with applications in pattern recognition. *IEEE Transactions on Information Theory* 21, 1 (jan 1975), 32–40. https://doi.org/10.1109/TIT.1975.1055330

[7] Ran Gal, Lior Shapira, Eyal Ofek, and Pushmeet Kohli. 2014. FLARE: Fast layout for augmented reality applications. In *2014 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. IEEE, 207–212. https://doi.org/10.1109/ISMAR.2014.6948429

[8] 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 Press, New York, New York, USA, 4021–4033. https://doi.org/10.1145/3025453.3025683

[9] Anuruddha Hettiarachchi and Daniel Wigdor. 2016. Annexing Reality: Enabling Opportunistic Use of Everyday Objects as Tangible Proxies in Augmented Reality. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems - CHI '16*. ACM Press, New York, New York, USA, 1957–1967. https://doi.org/10.1145/2858036.2858134

[10] Galen Hunt and Doug Brubacher. 1999. Detours: Binary Interception of Win32 Functions. In *Proceedings of the 3rd Conference on USENIX Windows NT Symposium - Volume 3 (WINSYM'99)*. USENIX Association, Berkeley, CA, USA, 14–14. http://dl.acm.org/citation.cfm?id=1268427.1268441

[11] Seokhee Jeon and Seungmoon Choi. 2009. Haptic Augmented Reality: Taxonomy and an Example of Stiffness Modulation. *Presence: Teleoperators and Virtual Environments* 18, 5 (oct 2009), 387–408. https://doi.org/10.1162/pres.18.5.387

[12] Brett Jones, Lior Shapira, Rajinder Sodhi, Michael Murdock, Ravish Mehra, Hrvoje Benko, Andrew Wilson, Eyal Ofek, Blair MacIntyre, and Nikunj Raghuvanshi. 2014. RoomAlive: Magical Experiences Enabled by Scalable, Adaptive Projector-camera Units. *Proceedings of the 27th annual ACM symposium on User interface software and technology - UIST '14* (2014), 637–644. https://doi.org/10.1145/2642918.2647383

[13] Brett R. Jones, Hrvoje Benko, Eyal Ofek, and Andrew D. Wilson. 2013. IllumiRoom. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '13*. ACM Press, New York, New York, USA, 869. https://doi.org/10.1145/2470654.2466112

[14] Takeo Kanade, Peter Rander, and PJ Narayanan. 1997. Virtualized reality: Constructing virtual worlds from real scenes. *IEEE multimedia* 4, 1 (1997), 34–47.

[15] Pascal Knierim, Valentin Schwind, Anna Maria Feit, Florian Nieuwenhuizen, and Niels Henze. 2018. Physical Keyboards in Virtual Reality: Analysis of Typing Performance and Effects of Avatar Hands. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems - CHI '18*. ACM Press, New York, New York, USA, 1–9. https://doi.org/10.1145/3173574.3173919

[16] Ryohei Komiyama, Takashi Miyaki, and Jun Rekimoto. 2017. JackIn Space: Designing a Seamless Transition Between First and Third Person View for Effective Telepresence Collaborations Ryohei. In *Proceedings of the 8th Augmented Human International Conference on - AH '17*. ACM Press, New York, New York, USA, 1–9. https://doi.org/10.1145/3041164.3041183

[17] Jérémy Lacoche, Nico Pallamin, Thomas Boggini, and Jérôme Royan. 2017. Improved Redirection with Distractors: A large-scale-real-walking locomotion interface and its effect on navigation in virtual environments. In *Proceedings of the 23rd ACM Symposium on Virtual Reality Software and Technology - VRST '17*. ACM Press, New York, New York, USA, 1–9. https://doi.org/10.1145/3139131.3139142

[18] David Lindlbauer and Andy D Wilson. 2018. Remixed Reality: Manipulating Space and Time in Augmented Reality. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems - CHI '18*. ACM Press, New York, New York, USA, 1–13. https://doi.org/10.1145/3173574.3173703

[19] Christian Mai, Mariam Hassib, and Ceenu George. 2017. *Like Elephants Do: Sensing Bystanders During HMD Usage*. Technical Report. https://pdfs.semanticscholar.org/5513/8687c17ad46747c95ab33fa9c4851ee2d26e.pdf?

[20] Steve Mann. 1999. Mediated Reality. *Linux J.* 1999, 59es, Article 5 (March 1999). http://dl.acm.org/citation.cfm?id=327697.327702

[21] 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 Press, New York, New York, USA, 1–10. https://doi.org/10.1145/3173574.3173815

[22] John C. McClelland, Robert J Teather, and Audrey Girouard. 2017. Haptobend: Shape-Changing Passive Haptic Feedback in Virtual Reality. In *Proceedings of the 5th Symposium on Spatial User Interaction - SUI '17*. ACM Press, New York, New York, USA, 82–90. https://doi.org/10.1145/3131277.3132179

[23] Mark McGill, Daniel Boland, Roderick Murray-Smith, and Stephen Brewster. 2015. A dose of reality: overcoming usability challenges in vr head-mounted displays. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems - CHI '15*. ACM Press, New York, New York, USA, 2143–2152. https://doi.org/10.1145/2702123.2702382

[24] Paul Milgram and Herman Colquhoun. 1999. A taxonomy of real and virtual world display integration. *Mixed reality: Merging real and virtual worlds* 1 (1999), 1–26.

[25] Paul Milgram and Fumio Kishino. 1994. *A Taxonomy of Mixed Reality Visual Displays Augmented Reality*. Technical Report 12. http://vered.rose.utoronto.ca/people/paul

[26] Raul Mur-Artal and Juan D. Tardos. 2017. ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras. *IEEE Transactions on Robotics* 33, 5 (oct 2017), 1255–1262. https://doi.org/10.1109/TRO.2017.2705103 arXiv:1610.06475

[27] Tabitha C. Peck, Henry Fuchs, and Mary C. Whitton. 2010. Improved Redirection with Distractors: A large-scale-real-walking locomotion interface and its effect on navigation in virtual environments. In *2010 IEEE Virtual Reality Conference (VR)*. IEEE, 35–38. https://doi.org/10.1109/VR.2010.5444816

[28] Julian Petford, Miguel A Nacenta, Carl Gutwin, Joseph Eremondi, and Cody Ede. 2016. The ASPECTA Toolkit: Affordable Full Coverage Displays. In *Proceedings of the 5th ACM International Symposium on Pervasive Displays - PerDis '16*. ACM Press, New York, New York, USA, 87–105. https://doi.org/10.1145/2914920.2915006

[29] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. 2016. You Only Look Once: Unified, Real-Time Object Detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 779–788. https://doi.org/10.1109/CVPR.2016.91 arXiv:1506.02640

[30] Erik Reinhard, M. Adhikhmin, Bruce Gooch, and Peter Shirley. 2001. Color transfer between images. *IEEE Computer Graphics and Applications* 21, 4 (2001), 34–41. https://doi.org/10.1109/38.946629

[31] Jeffrey M. Richter. 1999. *Programming Applications for Microsoft Windows with Cdrom* (4th ed.). Microsoft Press, Redmond, WA, USA.

[32] Joan Sol Roo and Martin Hachet. 2017. One Reality. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology - UIST '17*. ACM Press, New York, New York, USA, 787–795. https://doi.org/10.1145/3126594.3126638

[33] Lior Shapira and Daniel Freedman. 2016. Reality Skins: Creating Immersive and Tactile Virtual Environments. In *2016 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. IEEE, 115–124. https://doi.org/10.1109/ISMAR.2016.23

[34] Adalberto L Simeone. 2015. Substitutional reality: Towards a research agenda. In *2015 IEEE 1st Workshop on Everyday Virtual Reality (WEVR)*. IEEE, 19–22. https://doi.org/10.1109/WEVR.2015.7151690

[35] Shuran Song, Fisher Yu, Andy Zeng, Angel X. Chang, Manolis Savva, and Thomas Funkhouser. 2017. Semantic Scene Completion from a Single Depth Image. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 190–198. https://doi.org/10.1109/CVPR.2017.28 arXiv:1611.08974

[36] Misha Sra, Sergio Garrido-Jurado, and Chris Schmandt. 2016. Procedurally generated virtual reality from 3D reconstructed physical space. In *Proceedings of the 22nd ACM Conference on Virtual Reality Software and Technology - VRST '16*. ACM Press, New York, New York, USA, 191–200. https://doi.org/10.1145/2993369.2993372

[37] William Steptoe, Simon Julier, and Anthony Steed. 2014. Presence and discernability in conventional and non-photorealistic immersive augmented reality. In *2014 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. IEEE, 213–218. https://doi.org/10.1109/ISMAR.2014.6948430

[38] Qi Sun, Arie Kaufman, Anjul Patney, Li-Yi Wei, Omer Shapira, Jingwan Lu, Paul Asente, Suwen Zhu, Morgan Mcguire, and David Luebke. 2018. Towards Virtual Reality Infinite Walking: Dynamic Saccadic Redirection. *ACM Transactions on Graphics* 37, 4 (jul 2018), 1–13. https://doi.org/10.1145/3197517.3201294

[39] Qi Sun, Li-Yi Wei, and Arie Kaufman. 2016. Mapping virtual and physical reality. *ACM Transactions on Graphics* 35, 4 (jul 2016), 1–12. https://doi.org/10.1145/2897824.2925883

[40] Sungjoon Choi, Qian-Yi Zhou, and Vladlen Koltun. 2015. Robust reconstruction of indoor scenes. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 5556–5565. https://doi.org/10.1109/CVPR.2015.7299195

[41] Martin Usoh, Ernest Catena, Sima Arman, and Mel Slater. 2000. Using Presence Questionnaires in Reality. *Presence: Teleoperators and Virtual Environments* 9, 5 (oct 2000), 497–503. https://doi.org/10.1162/105474600566989

[42] Andrew Wilson. 2017. Fast Lossless Depth Image Compression. In *Proceedings of the Interactive Surfaces and Spaces on ZZZ - ISS '17*. ACM Press, New York, New York, USA, 100–105. https://doi.org/10.1145/3132272.3134144

[43] Andrew D Wilson and Hrvoje Benko. 2016. Projected Augmented Reality with the RoomAlive Toolkit. In *Proceedings of the 2016 ACM on Interactive Surfaces and Spaces*. ACM, 517–520.

[44] Jacob O Wobbrock, Leah Findlater, Darren Gergle, and James J Higgins. 2011. The aligned rank transform for nonparametric factorial analyses using only anova procedures. In *Proceedings of the 2011 annual conference on Human factors in computing systems - CHI '11*. ACM Press, New York, New York, USA, 143. https://doi.org/10.1145/1978942.1978963

[45] Yiwei Zhao and Sean Follmer. 2018. A Functional Optimization Based Approach for Continuous 3D Retargeted Touch of Arbitrary, Complex Boundaries in Haptic Virtual Reality. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems - CHI '18*. ACM Press, New York, New York, USA, 1–12. https://doi.org/10.1145/3173574.3174118