Remixed Reality: Manipulating Space and Time in Augmented Reality

David Lindlbauer[1,2], **Andrew D. Wilson**[1]

[1]Microsoft Research
Redmond, WA, USA

[2] TU Berlin
Berlin, Germany

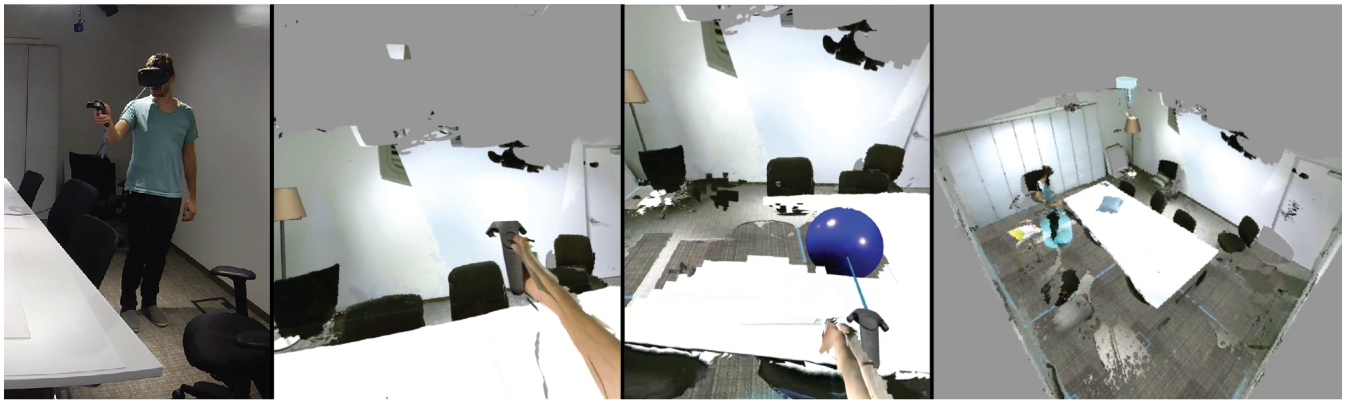

Figure 1. We present a novel take on mixed reality, called *Remixed Reality*. The user (*left*) wears a VR headset and sees a live reconstruction of the environment (*second left)* captured through multiple external depth cameras. This allows leveraging the benefits of virtual environments *(*e.g., easy modification of the environment by removing geometry*, second right*) and dynamic viewpoints while allowing users to see the actual physical world (e.g., teleportation, *right). All images in the paper were captured live with our working implementation of Remixed Reality.*

ABSTRACT

We present *Remixed Reality*, a novel form of mixed reality. In contrast to classical mixed reality approaches where users see a direct view or video feed of their environment, with Remixed Reality they see a *live 3D reconstruction*, gathered from multiple *external* depth cameras. This approach enables changing the environment as easily as geometry can be changed in virtual reality, while allowing users to view and interact with the actual physical world as they would in augmented reality. We characterize a taxonomy of manipulations that are possible with Remixed Reality: *spatial changes* such as erasing objects; *appearance changes* such as changing textures; *temporal changes* such as pausing time; and *viewpoint changes* that allow users to see the world from different points without changing their physical location. We contribute a method that uses an underlying voxel grid holding information like visibility and transformations, which is applied to live geometry in real time.

CHI 2018, April 21–26, 2018, Montreal, QC, Canada

ACM 978-1-4503-5620-6/18/04…$15.00
https://doi.org/10.1145/3173574.3173703

Author Keywords

Remixed Reality; Augmented reality; Virtual reality;

ACM Classification Keywords

H.5.m. Information interfaces and presentation (e.g., HCI).

INTRODUCTION

Mixed reality and augmented reality aim to alter users' perceived physical environment, for example by adding virtual objects, altering real-world objects, or replacing them with virtual content. Many of those changes, however, are challenging to achieve. To remove a physical object, for example to simulate remodeling of a room, any technology must be able to overlay the existing physical object with virtual content so that the object is no longer visible. This means it must rely on precise alignment of virtual and physical objects, the ability to extract background information from *behind* the physical object, and a display technology that is capable of overlaying the physical object with the extracted background information in a way that the original object is no longer visible. Since typical mixed reality technologies operate on front-facing cameras and a head-mounted see-through display or video see-through displays, these changes can be challenging to achieve, for example because the front-facing camera does not see the environment behind the physical object that should be removed.

In virtual reality (VR), on the other hand, the user only sees rendered geometry. Because this can be arbitrarily changed

in real time, every object or even the user representation itself can be subject to change. In contrast to mixed reality, however, the environment tthe user sees does not necessarily mirror the real world. Even a VR user viewing a virtual representation of the room they physically occupy would typically find that any changes in the real room would not be reflected in the virtual room since the geometry they see is not live.

In this work, we introduce *Remixed Reality*, an approach that aims to combine the benefits of augmented reality and virtual reality. We created a mixed reality environment in which users see a live view of their environment. In contrast to other mixed reality approaches such as the work by Miyaki and Rekimoto [28], users do not see the image from a head-mounted front-facing camera. Instead, we equip the environment with multiple RGB-D cameras (Microsoft Kinect v2). These depth cameras are positioned to obtain maximum visual coverage of the room. Users wear an immersive head-mounted display (HTC Vive) and see a *reconstructed live view* of the environment (see Figure 1). We transform the user's viewpoint in this reconstructed live representation of the room to match their physical position in the room. This means that physical objects and their virtual representations are aligned. Since the data is live, any changes in the physical environment (e.g., furniture moving, or people entering the room) are visible in real time to users.

Remixed Reality can be seen as a different type of see-through augmented reality that allows users to experience their physical environment (the *reality* in the taxonomy of Milgram and Kishino [27]), while it also enables manipulations that are usually only available in purely virtual environments (the *virtuality* [27])). For example, users can easily "erase" geometry in the room (Figure 1, *second right*), enabling applications such as simulated room remodeling. Since the cameras cover the environment from multiple angles, users can erase parts of the room and see the environment behind these parts. This approach also allows recoloring, moving or copying objects. Since all changes are applied continuously on the live geometry, any moved or copied object continues to update dynamically in real time after the operation is invoked. Furthermore, since the environment is a 3D model, changing the user's view is as simple as modifying the graphics camera eyepoint, enabling virtual motion (in VR referred to as *teleportation*). For example, a user can move to another part of the room virtually to inspect the environment from a different perspective without changing their physical location (Figure 1, *right*). Remixed Reality also allows making temporal changes. Users can halt time (i.e., suspend receiving live data) and *move freely* in their static reconstructed environment. They also can record events (e.g., meetings) and play them back at any desired speed.

Since the data from the environment changes at every frame, we separate the operations we wish to apply to dynamically changing geometry from the geometry itself. Instead of performing modifications on a polygonal reconstruction or point cloud, we use an underlying voxel grid that records interactions on the geometry, for example applying transformations like translation and scaling, and changing visibility. These interactions are applied continuously on live geometry, a crucial feature in our system.

Remixed Reality extends the space of possibilities in mixed reality and provides a way to seamlessly bridge the virtual and physical world. It gives users full control over how they perceive their physical environment. This allows them to make nearly arbitrary modifications, including those that would not be plausible in the real world.

Contributions

- The concept of Remixed Reality, blending the virtual and physical world in a novel way.
- A taxonomy of manipulations and interactions of our approach, outlining its benefits and limitations.
- A method to modify live depth data from multiple Kinects in real time using an underlying voxel grid.
- A detailed description of our platform that enables knowledgeable researchers to re-implement our system.

RELATED WORK

In this section, we discuss relevant related work from mixed reality and virtual reality that aims at blending the physical and virtual world. Furthermore, we outline work on handling live mesh data, which is important for our approach.

Blending physical and virtual

Research in human–computer interaction and computer graphics seeks to bridge the gap between the physical and the virtual world. There are various approaches to accomplish this, from optical and video see-through augmented reality to spatial augmented reality via projection mapping. Milgram and Kishino [27] categorized this work on a continuum from purely virtual to purely physical, outlining various technologies to achieve this. There exists a wide range of categorizations and terminologies for enriching the physical world with virtual content, for example Mediated Reality [23, 24, 25], Dual Reality [18], Hybrid Reality [44]. Most of them share the common goal of augmenting the view of the real world with virtual information, as demonstrated in early systems like NaviCam [38] or on commodity mobile phones [29]. We refer readers to work by Billinghurst et al. [4] on various techniques. These techniques have evolved over time and are now commercially available in recent products: video see-through augmented reality on recent smartphones, optical see-through head-mounted displays on Microsoft HoloLens, or video see-through on virtual reality headsets that are equipped with front-facing cameras such as the HTC Vive headset. In our work, we focus on a tight integration of physical and virtual content, blending the boundaries between the two. In contrast to prior work on augmented reality and available products, we do not show users a simple camera image of the physical world but a 3D reconstruction of their environment. This enables changing the physical world as easily as the virtual world, which is challenging with typical augmented reality techniques.

An alternative approach is *diminished reality* or *illusionary transparency* [23, 24, 25] which usually focuses on removing objects from a video feed. Typically, erasing requires inferring the environment behind the object through inpainting [11], approximation [52], or using multiple cameras as in our implementation. With SceneCtrl, Yue et al. [52] perform spatial changes in the environment (erase, copy, move) based on 3D reconstruction and display them with an optical see-through display (Microsoft HoloLens). Their approach relies on a static mesh generated during reconstruction, whereas changes performed with Remixed Reality apply to live geometry (e.g., a copied object and its source will both move if the source moves).

Lidarman [28] enables live out-of-body experiences by presenting users with a reconstructed view of a head-mounted light and range detection sensor. Remixed Reality uses external cameras to increase the range of possible viewpoints and enlarge the space of possible manipulations. Photoportals [17] allow users to record remote and local content, and review it in an immersive environment. Remixed Reality uses a similar mechanism, but on the whole environment.

Spatial augmented reality, in contrast to see-through augmented reality, directly augments the physical world to change its appearance (c.f., [5]), for example Shader Lamps [35], or work by Underkoffler et al. on i/o bulb [47] or Urp [48]. Lindlbauer et al. [21] demonstrates manipulating the appearance of physical objects by changing their surrounding space. The systems described in these works can change properties such as color, texture, or size. However, more dramatic changes such as completely removing an object are challenging. Remixed Reality additionally introduces manipulation of the user's viewpoint across space and time, which is very difficult with spatial augmented reality.

Several works take advantage of the possibility to use immersive virtual reality headsets to display physical reality with a camera, for example to increase the usability of virtual reality [26]. Roo et al. [39] propose a hybrid space, embedded in their concept of One Reality [40], in which users interact with objects in the physical world that are augmented with projection mapping. Users could switch to virtual reality to see virtual representations of the same physical objects. This allowed them to take on different, sometimes physically impossible, viewpoints (e.g., shrinking users).

Room-scale experiences

We aimed for a system where users are free to move within their environment while changing it. Spatial augmented reality, for example work by Jones et al. (IllumiRoom [15] and RoomAlive [16]), similarly allow the viewer to move throughout their environment. Raskar et al. [34] also altered the environment in the "Office of the Future" project. These previous works demonstrated changing the appearance of physical objects through direct augmentation. Remixed Reality allows changing the physical environment by presenting users with a live representation of the environment, which can be fully manipulated. Sra et al. [45] generated a virtual environment that roughly reflects a physical environment while allowing users to freely walk in it. As in typical virtual reality approaches, users see a purely virtual environment. A key feature of Remixed Reality is that that users see the physical environment and can leverage its benefits such as haptic feedback or being aware of other people in a room.

Interacting with live depth data

On a technical level, our approach relies on the ability to interact with the virtual environment and to manipulate it. Most approaches that focus on depth data either rely on static geometry or continuously refine geometry that is assumed to be static. KinectFusion [14], as one example, tracks the environment and refines a model by matching consecutive frames of depth data. Valentin et al. [49] extended this approach with segmentation. Innmann et al. [13] fit a model to the received depth data to accurately reconstruct volumetric deformation. With Fusion4D, Dou et al. [8] performed reconstruction on live data, however their approach does not work for large motion such as a user moving through a room. These approaches focus on mapping and reconstruction of environment rather than user interaction. Scheiblauer and Wimmer [41] used volumetric representations to perform selection in large scale point clouds. Owada et al. [32] introduced the "volume catcher," which allows users to select regions on a 3D mesh through 2D contours. Singh et al. [43] used skeleton data from meshes to infer possible selections. Bürger et al. [7] introduced manipulation techniques for volumetric data based on scalar fields. While we also use a volumetric representation, we focus on live data, i.e., the mesh changes every frame. Aforementioned work focused on interaction on static data. Therefore, manipulation performed in one frame would be lost in the next frame. In our implementation, users perform selections and manipulations on an underlying voxel grid, which enables consistency across frames.

REMIXED REALITY

We explore the concept of Remixed Reality by identifying a taxonomy of possible modifications, illustrated in Figure 2. The dimensions of the taxonomy are defined by four main types of modifications: spatial, appearance, temporal and viewpoint modifications. The taxonomy is inspired by work on shape-changing interfaces [36], augmented and virtual reality [27, 23, 21], and other dynamic interfaces [30, 20]. We refer readers to the accompanying video for a demonstration of our system which implements all modifications.

Requirements

The main requirements for Remixed Reality are a means to capture a live 3D model of the environment, and a display that allows users to see the (modified) physical world and virtual objects at the same time. For some modifications, it is important that the capture extends beyond what the user may physically see from their physical viewpoint and moment in time, so that, for example they should be able to overcome occlusion and see the physical world behind users. To achieve this, we currently use an immersive head-mounted display (HMD) and multiple depth cameras in a room. The number of cameras required depends on the environment to

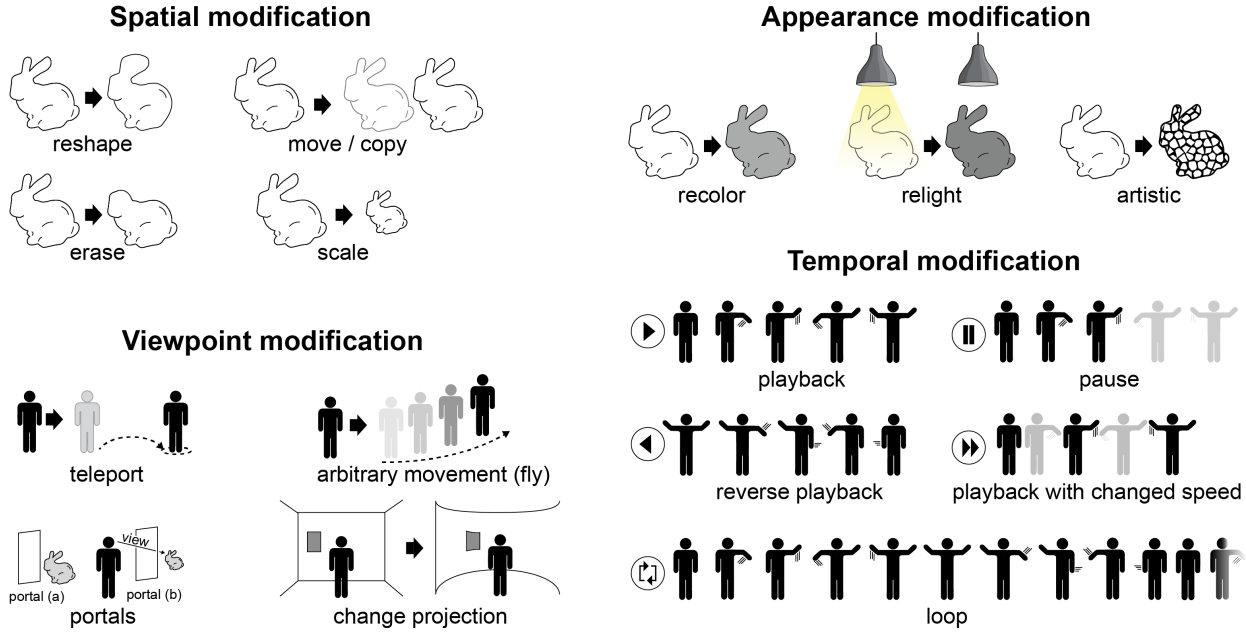

Figure 2. Taxonomy of possible manipulations for Remixed Reality with four dimensions for modifications: spatial, appearance, viewpoint and temporal. For demonstration of all effects, please refer to the accompanying video.

be covered and the location of potential points of interest in the room. We currently use eight Kinects for a room of approximately 4 *x* 5 meters. While the taxonomy presented in this paper was created with our current implementation in mind, we believe it also applies to other technological contexts. Future implementations might replace the immersive HMD with see-through displays that are capable of completely overlaying the physical world with content, and data could be acquired with future hardware such as a few 360 degree RGB-(D) cameras.

Spatial modification

Remixed Reality is appropriate for real-time modification of objects and the environment since *everything the users see is reconstructed geometry*. The ability to modify every aspect of the geometry, such as shape, size, or position, with available techniques from computer graphics, is key to making typically challenging modifications (e.g., completely removing an object from user's view) simple. All changes are illustrated in Figure 2 (*top left)* and can be performed manually by users or automatically by the system.

Reshape

Geometry including parts of physical objects and the environment can be reshaped by moving parts of the geometry or performing geometric operations such as erosion, expansion or morphing, for example. This enables applications such as live-sculpting of physical objects, e.g., for industrial design.

Erase

Objects in the environment can be partly or completely removed from the user's view. This can be valuable for privacy considerations (e.g., only specific users can see an object in a room) or room re-modeling. This type of diminished reality usually requires inferring the environment behind the object. Our approach resolves challenges of occlusion since the representation of the physical world and the virtual geometry shared the same coordinate system.

Move & Copy

Objects can be moved by users or the system, for example to unblock the view of the environment behind it, without physically moving the object. Note that motion is not constrained by physics, meaning that is possible to move a chair so that it appears to be levitating. Copying an object essentially creates a virtual clone of the physical geometry. Since the representation of the physical object and its copy consist of the exact same geometry and texture, it becomes difficult for users to distinguish the copy from the original. Since *both operations are performed on live geometry, any changes to the source object are propagated to the moved or copied object.*

Scale

Objects can be scaled arbitrarily, enabling users to remodel their environment at will. Scaling can be used to create world-in-miniature representations [46], or visualizations employing magic lens effects [9, 50], for example.

Appearance modification

Appearance modifications are distinct from spatial modifications in that they only alter the look (e.g., color, texture) of the environment, instead of its geometry (see Figure 3).

Recolor

Recoloring objects can be performed either by replacing an object's color or blending, for example for applications such as ambient displays or interior design. As for most other changes, recoloring can be performed on the whole object, parts of it, or the static environment. In our current implementation, the target is specified by the user's selection.

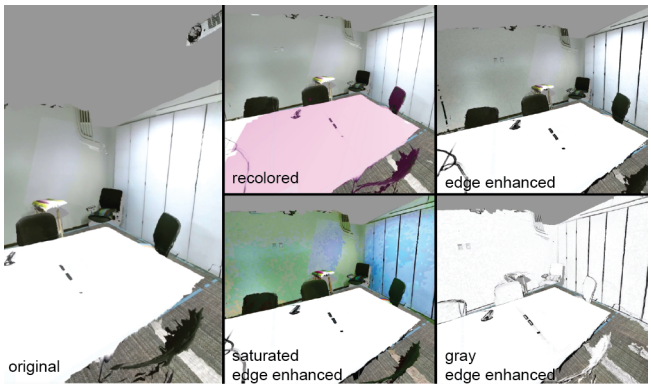

Figure 3. Appearance modifications applied to the live view.

Relight

Changing the light changes the color of the whole environment or parts of it indirectly. This enables for example dimming the room light to focus on a task, or gaming scenarios. Note that, as with any other changes, the light only changes in the virtual rendering of the room, and not in the real room.

Other artistic changes

Remixed Reality enables changing the appearance in various ways, e.g., for artistic purposes or gaming. This can be achieved by applying image filters such as edge enhancement, desaturation or blurring. Figure 3 shows examples of such changes, with edges being enhanced using a Sobel filter and colors being saturated and turned black and white, respectively.

Temporal modification

Remixed Reality also enables changing the temporal properties of an environment. Any data captured through the depth cameras can be paused, recorded and played back at any desired speed.

Pause time

Time is "paused" by no longer updating the geometry that is presented to users. This does not mean, however, that they are presented with a still image of the scene, but rather a static 3D reconstruction of the environment. Users can move freely in the scene and observe the room at the instant in time when the pause command was triggered. Figure 4 shows an example where a user paused time during jumping. Even though users see a static scene, time in the physical world resumes normally. Once the resume-command is triggered, users effectively make a "time jump". Events that occurred during a pause can be recorded and played back later.

Playback and loop

The representation of the physical world that is presented to users can be recorded and played back at various speeds. This allows for example slowly replaying short events (e.g., the jump in Figure 4), or fast playback of longer events such as meetings. Remixed Reality also allows users to repeatedly play back events (i.e., looping).

Figure 4. User "paused" time during jumping, then walked around the table to inspect the scene.

Altering viewpoint

An advantage of Remixed Reality over conventional mixed reality approaches is the ability to dynamically alter the user's viewpoint. Since Remixed Reality features large visual coverage of the environment, users are *free to choose their viewing location.*

Teleportation and arbitrary movement

Remixed Reality is naturally suited to camera and viewpoint control techniques used in games and virtual reality. We refer to discrete viewpoint changes (i.e., specifying a target position in the room and going there without physical motion) as *teleportation*, adopted from current virtual reality games. Furthermore, users are not constrained to walk on the ground but can also fly or walk on walls, for example. This is achieved by simply moving the virtual camera away from user's actual tracked position in the room.

Portals

Users are not only able to see the environment from their current physical or virtual perspective, but to equip the environment with "portals." Conceptually and in our current implementation, a single portal consists of a display area and a virtual camera. If no other portal is present, this single portal acts like a mirror. By linking multiple portals (e.g., portal A shows the view of portal B and vice versa), users can view the room from multiple perspectives. Figure 5 shows two portals, one placed beside the user, the other behind them. By looking at the portal to their left side the user can see themselves from behind.

Portals can be viewpoint-dependent or viewpoint-independent and have been used for navigation in virtual reality (e.g., [6, 10]). Viewpoint-dependent portals essentially behave like a fishtank VR display [42], meaning that the perspective (i.e., the virtual camera's rotation) changes according to a user's perspective. Viewpoint-independent portals are similar to conventional displays with a static virtual camera. Portals can also be used to constantly monitor regions-of-interest as in the work by Lin et al. [19].

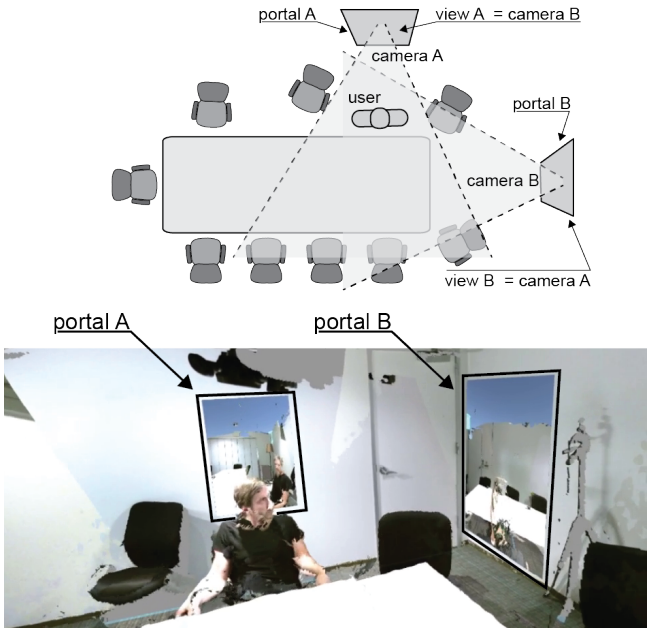

Figure 5. Portal A shows the view of portal B (side of user), while portal B shows the view of portal A (back of user).

In our current implementation, those portals can not only be used for viewing but also as ways to transport objects and users, inspired by the game Portal by Valve Corporation. By moving an object into a portal (i.e., triggering a collision), the object is moved "through" the portal to the location of the linked portal. Users can perform the same action themselves (i.e., walk through a portal) as an alternative to teleportation.

Changing perspectives

For certain applications, keeping a constant overview of the environment can be valuable. By default, the user's view employs a conventional perspective graphics camera. We can, however, use other camera models that feature, for example, fisheye lenses or arbitrary distortions to give users the ability to easily enlarge their field of view, as shown in Figure 6. This furthermore enables modifications of the viewport such as real-world zooming, or x-ray (i.e., viewing *through* objects), or taking on the perspective of another person (or even object) in the environment.

Virtual content in a reconstructed reality

Mixed reality approaches allow the environment to be augmented and extended with virtual objects. Typically, virtual content overlays real objects to convey information. Remixed Reality additionally allows erasing a physical object and *optionally* replacing it with a virtual one. One example is demonstrated throughout many of the example images in this paper: to fill holes in the reconstructed floor caused by an incomplete model of the room, we replaced the floor with a virtual ground plane with a texture similar to the physical floor. Since it is visually similar to the actual floor, it blends into the room nicely.

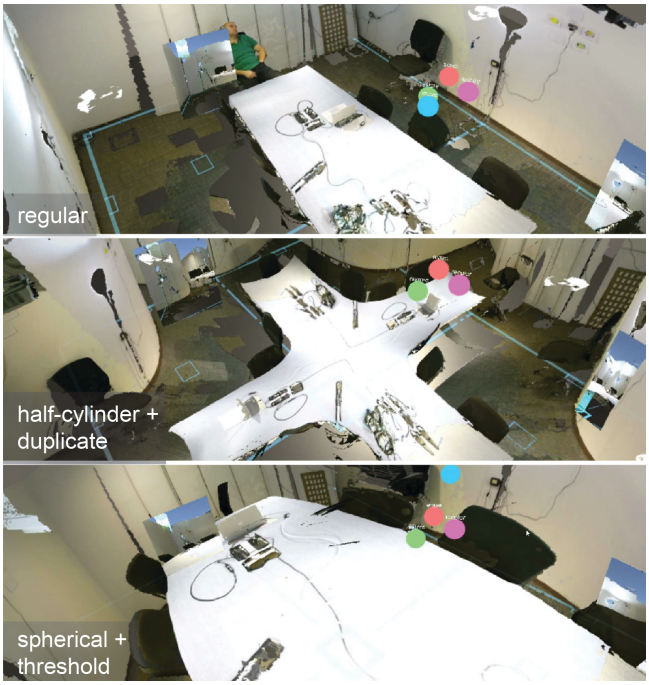

Figure 6. Users can be presented with different perspectives on the current environment. *Top* shows a perspective with a conventional camera model. C*enter* shows a projection to a half-cylinder and all contents duplicated. *Bottom* shows a projection onto a sphere for contents within a certain distance.

Conventional mixed reality applications aim to render physical and virtual objects with the same high visual quality, and therefore must properly handle lighting, occlusion and anchoring of objects in the environment. With our current depth camera-based implementation, *object occlusion and anchoring are resolved automatically* since the room's physical environment is effectively physical and virtual at the same time. Note that occlusion (i.e., areas without visual coverage) still occurs for regions that are not covered by any of the external cameras. This could be resolved by using more cameras or cameras with a wider field of view. Physical and virtual objects share the same coordinate space, making anchoring easy. Correct inter- and intra-object occlusion is achieved by presenting users with the 3D reconstruction of the physical world. Considering visual quality, our approach also requires equalizing visual quality of physical and virtual geometry. Due to the rather low resolution of the depth cameras and calibration noise, the overall visual quality of the environment suffers. Consequently, the visual quality of the virtual objects should probably be decreased as well so that they do not attract undue attention. This could be achieved by analyzing geometric noise and colors retrieved from the depth cameras and forming a model from those parameters. By applying this model as a filter to virtual objects, the gap in visual quality would be decreased. We plan to investigate this in the future.

Safety considerations

Erasing or displacing physical objects, pausing time, and teleporting lead to potential safety challenges. Users lose the ability to judge whether physical objects are in their way, which can lead to bumping into those objects. This could be resolved by more advanced "chaperoning" techniques, such as rendering erased geometry as opaque or semi-transparent when users are in their proximity. Furthermore, applying techniques such as redirected walking [37] or inverse haptic retargeting [1] (e.g., warping the environment in a way that users *do not* touch anything) could resolve this challenge. We plan to investigate this interesting part of research in the future, especially since this not only applies to our approach but many virtual reality scenarios.

IMPLEMENTATION

Our current implementation relies on two main components, the *data acquisition and reconstruction* of the environment, and *displaying and interacting* with the reconstructed environment. Note that the user is *not* equipped with a head-mounted *camera* (only a head-mounted display), since their position might be independent of their viewpoint. The external cameras in our approach allow easy handling of occlusion and geometry alignment. An overview of the system is illustrated in Figure 7.

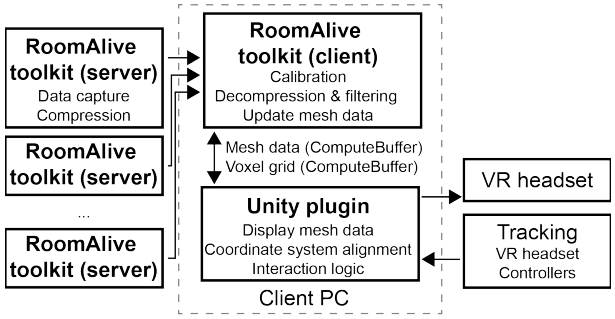

Figure 7. Overview of the Remixed Reality system.

Overview

Data acquisition and reconstruction is performed using the RoomAlive Toolkit[1] [16]. In the example shown in this paper, the room is equipped with eight Kinect v2 cameras. Each of the cameras is connected to a compact server PC (Intel NUC) which receives data from the depth cameras and sends the compressed depth and color data to the RoomAlive client, running on a commodity gaming PC. All PCs run Windows 10. The client performs decompression and reconstruction. Display and interaction is implemented in Unity 5.6 and SteamVR, which also runs on the client computer. A virtual reality headset (HTC Vive) and its trackers are connected to the client and used for displaying the geometry. Data exchange between the RoomAlive client application and Unity is implemented as a custom native Unity plugin. All shaders are implemented in DirectX 11.

Calibration & data acquisition

The depth cameras are calibrated using the RoomAlive calibration procedure, which was built for dynamic projection mapping. Six projectors were installed in the environment to display Gray codes onto the physical surfaces in the room. Using projectors could be omitted by performing the calibration directly on the received data for example using the iterative closest point (ICP) algorithm [3] for alignment.

The depth and color data is compressed (lossless depth compression based on [51], JPEG color compression) and sent over Ethernet to a computer running the RoomAlive client application. The eight Kinects produce approximately 400 Mbit/s of data, which is well within the limit of a regular network switch (typically 1 Gbit/s). This also means this approach would scale to more than eight cameras if the required network bandwidth is available. Data can be processed and rendered at the client in real-time at a maximum of 100 frames per second, which is higher than the framerate of the Kinect cameras (typically 30 frames per second).

The data is decompressed on the RoomAlive client and reconstructed using the camera parameters retrieved in the calibration procedure. The client also performs basic filtering and noise rejection and shares the data with Unity through a custom plugin.

Displaying live depth data

A custom Unity plugin handles data exchange between the RoomAlive client and Unity. In our current implementation, we create a Unity scene with multiple meshes for exchanging the data. The data of eight Kinects results in approximately 2 million vertices per frame. The meshes are represented as multiple compute buffers, shared between Unity and the RoomAlive client. This means that, while the data is updated constantly, Unity does not require its own copy of the data.

For display, we use a HTC Vive VR headset with the base stations for tracking mounted in the room. The coordinate systems of the mesh and the VR headset are manually aligned once (adjusting an affine transformation in Unity) and the resulting matrix is stored in Unity. This allows the physical world and its geometric representation to be tightly aligned. Note that there is no drift over time since the coordinate systems of the Kinects and the VR headset do not change over time. Calibration accuracy was in the range of few centimeters, which from our experience was sufficiently accurate for users to grab items (e.g., a remote control on the table). Accuracy could be further improved by using a semi-automatic multiple-point calibration procedure, for example.

The user is represented as camera in Unity. Since the tracked space of the headset and the mesh data are aligned, user motion in the physical space is forwarded as camera motion, allowing the user to move freely in the tracked space. For teleportation, we move the camera that represents the user.

[1] https://github.com/Kinect/RoomAliveToolkit

Modifying live depth data for interaction

Modifying a mesh, for example erasing or moving parts, typically involves modifying its underlying vertices. In our case, however, the mesh that represents the environment changes at a rate of 30 frames per second. Since there is no correspondence between the meshes over time, any change performed on the mesh at one frame would not carry over to the next frame. Retaining changes between frames would require aligning the raw vertices (e.g., using ICP or optical flow) and determining which vertices have been changed due to interaction. In our case of approximately 2 million vertices, this operation would be very computationally expensive, potentially prohibiting real-time interaction. Therefore, we rely on an *underlying volumetric representation of interaction data* that is maintained simultaneously. The polygonal reconstruction is used solely for display purposes. We implemented a 3D voxel grid comprised of 1.35 million voxels (150 width *x* 150 length *x* 60 height), resulting in a voxel size of approximately 3 cm in our room (4 m *x* 5 m x 2.5 m), shown in Figure 8. The voxel grid is initialized as a compute buffer shared between Unity and the native plugin and passed to all shaders. The voxel grid and the geometry from the depth cameras share the same coordinate system, meaning that each vertex has a corresponding voxel (however a voxel typically contains multiple vertices).

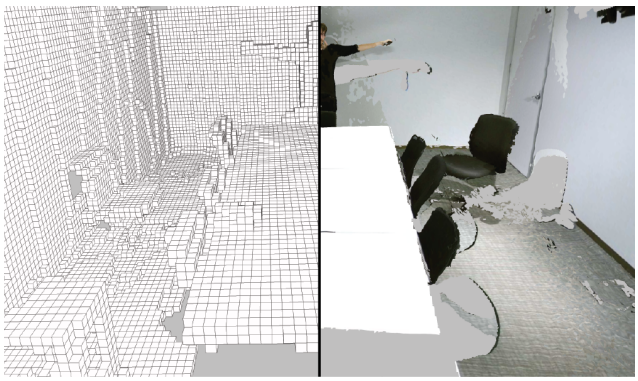

Figure 8. We store interaction data (e.g., changes in visibility, transform, color) in an underlying voxel grid (*left)* that is aligned with the geometry retrieved from the depth cameras *(right)*. Users do not see the voxel grid.

This means that although vertices between frames do not have any correspondence, they do fall in the same voxel if there was no motion between frames. This is key to enable modifying the geometry and preserving changes over time. When the user selects geometry, this selection is performed on the voxel grid rather than the geometry. Each voxel holds flags for interaction data, i.e., if it has been selected, colored, hidden, moved or copied. In the geometry shader that generates the mesh, each vertex queries its corresponding voxel. Based on the flags that are specified in the voxel, the shader performs the corresponding interaction, for example translating a vertex if the *move*-flag is active. This process is illustrated in a 2D example in Figure 9.

Note that in this approach all actions are performed on a per-vertex basis, requiring the examination of only a small fraction of the voxel grid. This means that the resolution of the voxel grid can be increased up to the limit of the graphics card's memory, since every interaction is basically a look-up in the voxel grid per vertex. Thus, the computational complexity only scales proportional to the number of vertices, not the number of voxels. While we currently use 1.35 million voxels for 2 million vertices, we tested this implementation with up to 50 million voxels on the same number of vertices. This only increased the time to upload the voxel grid onto the GPU (only performed once at the start of the software). Keeping voxel size constant, this means that this approach can cover a room significantly larger than the one used in our experiments.

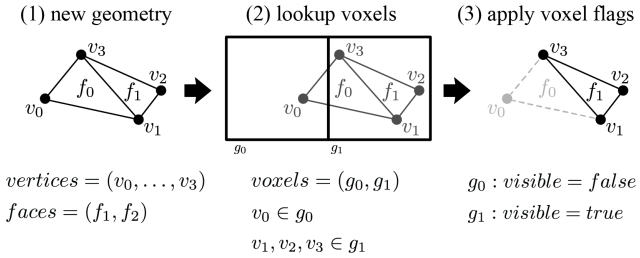

Figure 9. Workflow for modifying live mesh data. For all new geometry (*left)*, we perform a lookup operation to find the voxels at the same position (*center)*. Each voxel contains specific flags regarding properties such as visibility or translation. Those flags are then applied to the geometry (*right)*. In this case, vertices that fall within the left voxel are erased, i.e., discarded by the geometry shader.

User interaction with volumetric handles

Users interact with the environment using spherical selection handles. The interaction is related to the "go-go" technique [33], and allows performing selections on physical surfaces without bumping into them. The handles manipulate the voxel grid that is applied to the environment's geometry (for example the "erase" handle sets the corresponding voxels to "hidden"). This manipulation is performed on the GPU by passing the voxel grid to the shaders that render the handles.

APPLICATIONS

We utilize the proposed capabilities in a variety of applications taking advantage of spatial, temporal, viewpoint and appearance modification.

Interior design

We created an application that allows users to remodel their room. First, users erase any objects in the room they want to replace. They can then trigger a second purely virtual room that is equipped with furniture, and teleport themselves into this room. By moving furniture into the physical space, they can see what the remodeled room would look like, for example as in Figure 10. By placing virtual seating furniture where physical chairs are, they can also sit "on virtual objects", taking advantage of the physical objects as haptic proxies for virtual objects (as in [12]).

Relive a meeting

Because the complete physical environment in a room can be recorded, users can relive any past event, for example meetings. When "watching" the meeting while in the room, users can choose to either sit at an empty chair or take the perspective of another attendee. The perspective can be changed any time, for example through teleportation. Since the data is recorded, users can also choose to "attend" the meeting at increased speed or jump back in time during the meeting to important moments.

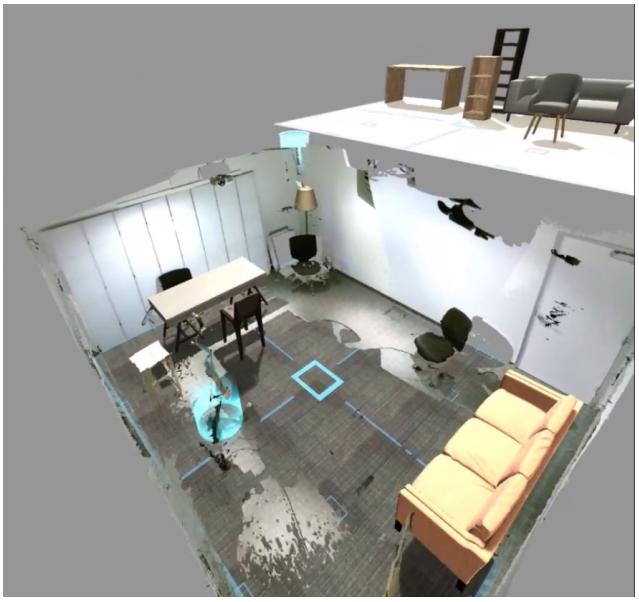

Figure 10. The user has erased the table and chairs in the room and replaced them with virtual furniture (couch, table, chair). Note the virtual second floor with furniture on the top right. Users can teleport there to retrieve virtual objects.

Gaming

Remixed Reality can be used for gaming purposes. To demonstrate this, we created a simple shooting game. Players can dim the light in the room (virtually), and use a blaster to erase geometry in the room. The geometry spawns back after a few seconds. Combining this with other changes in appearance, for example enhanced contours, and playback of prerecorded scenes, leads to an immersive gaming experience in the physical world without players being bound to physical constraints, e.g., they can fly rather walk on the ground.

Investigate posture

Users can halt time at any point. This allows for example applications such as investigating the posture of fast movements. As an example, by pausing time at the moment a baseball player hits a ball, common errors such as collapsing the back side or dipping the shoulder can be viewed first hand and corrected. Remixed Reality allows not only capturing and viewing this moment in 2D but as a full 3D representation that users can walk around and even virtually correct a posture through sculpting.

Remove distraction

By incorporating Kinect body tracking, users can choose to remove the unwanted distraction caused by other people in the room. In this mode, the system removes all users by erasing the geometry sufficiently near to the position of their tracked heads. By incorporating object tracking based on geometry rather than head position, this removal could be more fine-grained.

Fast viewpoint switch

When performing actions on larger objects, for example fastening a knot around a large box, or reaching for something that is occluded by a large object, users may need to walk around the object to see what they are doing. Remixed Reality allows quickly changing the user's perspective (currently by pushing a button on the controller), which enables them to look behind or around objects without changing their physical location.

EXTENSION: REMIXING REMOTE REALITY

While the core concept of Remixed Reality focusses on co-located interaction, users can also incorporate remote spaces into their environment *or* interact with a remote environment. Incorporating spatially distant locations into a single environment is performed by aligning and blending the two. Furthermore, this enables interactions such as adding a smaller representation of one environment into another, similar to concepts such as world-in-a-miniature [46] or Holoportation [31], as shown in Figure 11. This seamless blending (remixing) of multiple environments enables applications in telepresence or remote collaboration. Multiple blended environments are largely indistinguishable in terms of visual quality and immersion since they use the same capture and rendering processing pipeline.

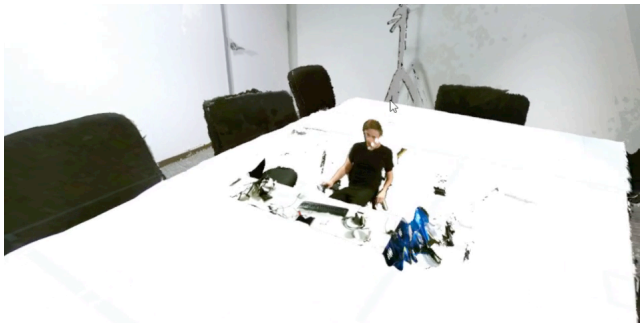

Figure 11. The live view of a remote user is placed as a miniature on the table.

LIMITATIONS

While Remixed Reality offers advantages in terms of modifications that go beyond conventional mixed reality approaches, several challenges arise.

Visual quality

The visual quality of the live 3D reconstruction presented to users depends on the resolution and positioning of the external depth cameras used by the system. Higher resolution depth cameras would partly alleviate the problem of low vis-

ual quality. Furthermore, by incorporating more advanced algorithms for live filtering and reconstruction or adding 3D scans of static parts of the environments (e.g., KinectFusion [14]), the visual quality could be improved. This, however, is challenging due to the large size of the overall reconstructed mesh. Any algorithm for visual enhancement would need to run in real time, a demanding requirement for many existing techniques. We plan to enhance the overall visual quality of our system as one of the next steps. This would further allow us to investigate which degree of visual fidelity is necessary for users to be even more immersed in the system, and potentially "forget" that they are wearing a headset.

Infrastructure and visual coverage

Our current implementation requires the installation of multiple depth cameras in a room to obtain large visual coverage of the environment. This, however, might not be feasible for many users. We envision that our approach will also work with different (future) camera technologies, for example through the use of a small number of 360-degree depth cameras. Future research also needs to tackle challenges when capturing surfaces that are not well captured by current cameras, such as transparent surfaces. We further plan to investigate how Remixed Reality can be implemented with fewer cameras, for example in part by using a single head-mounted camera that retains a copy of obtained geometry and updates this presentation constantly. This, however, would potentially prohibit large changes in viewpoint and lead to uncaptured areas due to occlusion.

Haptic feedback

Remixed Reality offers users a full, correct passive haptic experience before nearby geometry has been manipulated. This effect can be degraded as users modify their environment by erasing geometry, changing their viewport, or teleporting themselves, for example. Resolving this challenge would require techniques such as haptic retargeting [1] or other haptic proxies (e.g. [12]) to restore tactile sensation even when the physical and the virtual world no longer match.

DISCUSSION AND FURTHER WORK

Remixed Reality aims at combining the benefits of virtual and augmented reality, effectively blending physical and virtual spaces. The ability to easily modify many aspects of an environment is enabled by presenting users with a *reconstruction* of the environment. Benefits of mixed reality such as viewing the actual physical reality with real haptic sensations are preserved by using a *live view* of the environment. Remixed Reality enables a broad range of applications and interaction that have previously only been achievable in purely virtual environments, such as teleportation or temporal changes. We believe our work opens interesting extensions and directions for future work.

Extended scene understanding

In our current implementation, the representation shown to users is pure geometry (i.e., vertices and faces), without segmentation or knowledge of objects. Incorporating even basic scene understanding (e.g., via computing connected components) might be useful in some user interactions. This could for example be achieved by including segmentation algorithms like the ones used in Semantic Paintbrush [49].

With extended scene understanding in place, we envision that our approach also allows creating applications that allow recording, replaying and altering events with specific physical behavior (cf. Aftermath [22]), for example throwing a ball once and replaying this action with various physical parameters such as velocity or direction.

From Mediated Reality to Artificial Reality and back

In his seminal paper on Mediated Reality, Mann [23] introduced the idea of a wearable device that can capture and control incoming light rays as well as produce light rays, essentially giving users full control over what they see and perceive. We see Remixed Reality as one instantiation of this idea, since it affords *full control over what users see by converting their environment into live reconstructed geometry.* Remixed Reality allows users to seamlessly transition between different levels of virtuality, from seeing the pure physical world without augmentation, via partly altered reality to fully artificial reality. While Mann focused mostly on visual perception, we envision future versions of Remixed Reality capable of altering users' perception of physics (e.g., everything appears to be happening in a zero-gravity environment) and perception of time (e.g., everything moves slowly). Once users can no longer distinguish what is virtual and what is real, we can seek the most comfortable or appropriate level of mediation. We envision such a system to be suitable for a wide range of psychophysical experiments (e.g., investigate out-of-body perception) and to allow us to create and extend the experiences Mann envisioned, from showing users what a monochromatic environment looks like, what the world looks like from the perspective of another person, what living in an upside-down world feels like, or extending users capabilities with ability to zoom and see through walls. While Mann experimented with some of those effects, we see Remixed Reality as a first step of gaining full control of what we see and being able to choose the level of augmentation we desire.

CONCLUSION

In this work we introduced Remixed Reality, a novel form of mixed reality. It allows users to see the actual physical world but perform changes as easily as in virtual reality. This is enabled by presenting and modifying a live 3D reconstruction of the environment, captured through multiple external depth cameras. We believe this approach allows for applications that require altering users' surrounding space such as diminished reality, as well applications where users want to make temporal changes like pausing time, or recording and playing back specific events.

ACKNOWLEDGEMENTS

We would like to thank Eyal Ofek, Hrvoje Benko and Christian Holz for their advice and support, and Eric Whitmire, Evan Strasnik and Gierad Laput for extended discussions.

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, New York, NY, USA, 1968-1979. https://doi.org/10.1145/2858036.2858226
2. Hrvoje Benko, Andrew D. Wilson, and Federico Zannier. 2014. Dyadic Projected Spatial Augmented Reality. *In Proceedings of the ACM Symposium on User Interface Software and Technology* (UIST '14). ACM, New York, NY, USA, 645–655. http://doi.acm.org/10.1145/2642918.2647402
3. Paul J. Besl and Neil D. McKay. 1992. A Method for Registration of 3-D Shapes. IEEE Trans. Pattern Anal. Mach. Intell. 14, 2 (February 1992), 239-256. http://dx.doi.org/10.1109/34.121791
4. Mark Billinghurst, Adrian Clark, and Gun Lee. 2015. A Survey of Augmented Reality. *Found. Trends Hum.-Comput. Interact. 8, 2-3,* 73-272. http://dx.doi.org/10.1561/1100000049
5. Oliver Bimber and Ramesh Raskar. 2005. Spatial Augmented Reality: Merging Real and Virtual Worlds. A. K. Peters, Ltd., Natick, MA, USA.
6. Gerd Bruder, Frank Steinicke, Klaus H. Hinrichs. 2009. Arch-Explore: A natural user interface for immersive architectural walkthroughs. In *Proceedings of the IEEE Symposium on 3D User Interfaces* (3DUI'09). 75-82. https://doi.org/10.1109/3DUI.2009.4811208
7. Kai Bürger, Jens Krüger and Rüdiger Westermann. Direct Volume Editing. 2008. In *IEEE Transactions on Visualization and Computer Graphics*, 14 / 6, 1388-1395. https://doi.org/10.1109/TVCG.2008.120
8. Mingsong Dou, Sameh Khamis, Yury Degtyarev, Philip Davidson, Sean Ryan Fanello, Adarsh Kowdle, Sergio Orts Escolano, Christoph Rhemann, David Kim, Jonathan Taylor, Pushmeet Kohli, Vladimir Tankovich, and Shahram Izadi. 2016. Fusion4D: real-time performance capture of challenging scenes. In *ACM Trans. Graph.* 35/4. https://doi.org/10.1145/2897824.2925969
9. Steven Feiner, Blair MacIntyre, Marcus Haupt, and Eliot Solomon. 1993. Windows on the world: 2D windows for 3D augmented reality. *In Proceedings of the ACM Symposium on User Interface Software and Technology* (UIST '93). ACM, New York, NY, USA, 145-155. https://doi.org/10.1145/168642.168657
10. Sebastian Freitag, Dominik Rausch, Torsten Kuhlen. 2014 Reorientation in virtual environments using interactive portals. In *Proceedings of the IEEE Symposium on 3D User Interfaces* (3DUI'14). 119-122. https://doi.org/10.1109/3DUI.2014.6798852
11. Jan Herling and Wolfgang Broll. 2012. PixMix: A real-time approach to high-quality Diminished Reality. In *Proceedings of the IEEE International Symposium on Mixed and Augmented Reality* (ISMAR '12). IEEE Computer Society, Washington, DC, USA, 141-150. http://dx.doi.org/10.1109/ISMAR.2012.6402551
12. Anuruddha Hettiarachchi and Daniel Wigdor. 2016. Annexing Reality: Enabling Opportunistic Use of Everyday Objects As Tangible Proxies in Augmented Reality. In *Proceedings of the CHI Conference on Human Factors in Computing Systems* (CHI '16). ACM, New York, NY, USA, 1957–1967. http://doi.acm.org/10.1145/2858036.2858134
13. Matthias Innmann, Michael Zollhöfer, Matthias Nießner, Christian Theobalt, and Marc Stamminger. (2016) VolumeDeform: Real-Time Volumetric Non-rigid Reconstruction. In *Proceedings of the European Conference on Computer Vision* (ECCV '16). 362-379. https://doi.org/10.1007/978-3-319-46484-8_22
14. Shahram Izadi, David Kim, Otmar Hilliges, David Molyneaux, Richard Newcombe, Pushmeet Kohli, Jamie Shotton, Steve Hodges, Dustin Freeman, Andrew Davison, and Andrew Fitzgibbon. 2011. KinectFusion: real-time 3D reconstruction and interaction using a moving depth camera. In *Proceedings of the ACM Symposium on User Interface Software and Technology* (UIST '11). ACM, New York, NY, USA, 559-568. https://doi.org/10.1145/2047196.204727
15. Brett R. Jones, Hrvoje Benko, Eyal Ofek, and Andrew D. Wilson. 2013. IllumiRoom: Peripheral Projected Illusions for Interactive Experiences. *In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (CHI '13). ACM, New York, NY, USA, 869–878. http://doi.acm.org/10.1145/2470654.2466112
16. Brett R. Jones, Rajinder Sodhi, Michael Murdock, Ravish Mehra, Hrvoje Benko, Andrew D. Wilson, Eyal Ofek, Blair MacIntyre, Nikunj Raghuvanshi, and Lior Shapira. 2014. RoomAlive: magical experiences enabled by scalable, adaptive projector-camera units. In *Proceedings of the ACM Symposium on User Interface Software and Technology* (UIST '14). ACM, New York, NY, USA, 637-644. https://doi.org/10.1145/2642918.2647383
17. André Kunert, Alexander Kulik, Stephan Beck, and Bernd Froehlich. 2014. Photoportals: shared references in space and time. In *Proceedings of the ACM conference on Computer supported cooperative work & social computing* (CSCW '14). ACM, New York, NY, USA, 1388-1399. http://dx.doi.org/10.1145/2531602.2531727
18. Joshua Lifton, and Joseph A. Paradiso. 2010. Dual Reality: Merging the Real and Virtual. In *Facets of Virtual Environments* (FaVE '09). Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-11743-5_2

19. Yung-Ta Lin, Yi-Chi Liao, Shan-Yuan Teng, Yi-Ju Chung, Liwei Chan, and Bing-Yu Chen. 2017. Outside-In: Visualizing Out-of-Sight Regions-of-Interest in a 360 Video Using Spatial Picture-in-Picture Previews. In *Proceedings of the ACM Symposium on User Interface Software and Technology* (UIST '17). ACM, New York, NY, USA. https://doi.org/10.1145/3126594.3126656
20. David Lindlbauer, Jörg Müller, and Marc Alexa. 2016. Changing the Appearance of Physical Interfaces Through Controlled Transparency. In *Proceedings of the ACM Symposium on User Interface Software and Technology* (UIST '16). ACM, New York, NY, USA. http://doi.acm.org/10.1145/2984511.2984556
21. David Lindlbauer, Jörg Müller, and Marc Alexa. 2017. Changing the Appearance of Real-World Objects By Modifying Their Surroundings. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems* (CHI '17). ACM, New York, NY, USA, 3954-3965. https://doi.org/10.1145/3025453.3025795
22. Sang-won Leigh and Pattie Maes. 2015. AfterMath: Visualizing Consequences of Actions through Augmented Reality. In *Proceedings of the Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems* (CHI EA '15). ACM, New York, NY, USA, 941-946. https://doi.org/10.1145/2702613.2732695.
23. Steve Mann. Mediated reality. *Technical Report MIT-ML Percom* TR-260, University of Toronto, 1994.
24. Steve Mann and James Fung. 2001. Videoorbits on eye tap devices for deliberately diminished reality or altering the visual perception of rigid planar patches of a real world scene. In *Proceedings of the International Symposium on Mixed Reality* (ISMR'01). 48-55.
25. Steve Mann and James Fung. 2002. EyeTap devices for augmented, deliberately diminished, or otherwise altered visual perception of rigid planar patches of real-world scenes. In *Presence: Teleoperators and Virtual Environments* 11, 2 158-175. https://doi.org/10.1162/1054746021470603
26. 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 Annual ACM Conference on Human Factors in Computing Systems* (CHI '15). ACM, New York, NY, USA, 2143-2152. https://doi.org/10.1145/2702123.27023
27. Paul Milgram and Fumio Kishino. A taxonomy of mixed reality visual displays. In *IIEICE Transactions on Information Systems*, Vol E77-D, No.12 December 1994.
28. Takashi Miyaki and Jun Rekimoto. 2016. LiDARMAN: reprogramming reality with egocentric laser depth scanning. In *ACM SIGGRAPH 2016 Emerging Technologies* (SIGGRAPH '16). ACM, New York, NY, USA, Article 15. https://doi.org/10.1145/2929464.2929481
29. Mathias Möhring, Christian Lessig, and Oliver Bimber, Video see-through AR on consumer cell-phones. 2004 In *Proceedings of the IEEE and ACM International Symposium on Mixed and Augmented Reality*, (ISMAR '04). 252-253. https://doi.org/10.1109/ISMAR.2004.63
30. Simon Olberding, Michael Wessely, and Jürgen Steimle. 2014. PrintScreen: Fabricating Highly Customizable Thin-film Touch-displays. In *Proceedings of the ACM Symposium on User Interface Software and Technology* (UIST '14). ACM, New York, NY, USA, 281–290. http://doi.acm.org/10.1145/2642918.2647413
31. Sergio Orts-Escolano, Christoph Rhemann, Sean Fanello, Wayne Chang, Adarsh Kowdle, Yury Degtyarev, David Kim, Philip L. Davidson, Sameh Khamis, Mingsong Dou, Vladimir Tankovich, Charles Loop, Qin Cai, Philip A. Chou, Sarah Mennicken, Julien Valentin, Vivek Pradeep, Shenlong Wang, Sing Bing Kang, Pushmeet Kohli, Yuliya Lutchyn, Cem Keskin, and Shahram Izadi. 2016. Holoportation: Virtual 3D Teleportation in Real-time. In *Proceedings of the ACM Symposium on User Interface Software and Technology* (UIST '16). ACM, New York, NY, USA, 741-754. https://doi.org/10.1145/2984511.2984517
32. Shigeru Owada, Frank Nielsen, and Takeo Igarashi. 2005. Volume catcher. In *Proceedings of the Symposium on interactive 3D graphics and* games (I3D '05). ACM, New York, NY, USA, 111-116. http://dx.doi.org/10.1145/1053427.1053445
33. Ivan Poupyrev, Mark Billinghurst, Suzanne Weghorst, and Tadao Ichikawa. 1996. The go-go interaction technique: non-linear mapping for direct manipulation in VR. In *Proceedings of the ACM Symposium on User Interface Software and Technology* (UIST '96). ACM, New York, NY, USA, 79-80. http://dx.doi.org/10.1145/237091.237102
34. Ramesh Raskar, Greg Welch, Matt Cutts, Adam Lake, Lev Stesin, and Henry Fuchs. 1998. The Office of the Future: A Unified Approach to Image-based Modeling and Spatially Immersive Displays. *In Proceedings of the Annual Conference on Computer Graphics and Interactive Techniques* (SIGGRAPH '98). ACM, New York, NY, USA, 179–188. http://doi.acm.org/10.1145/280814.280861
35. Ramesh Raskar, Greg Welch, Kok-Lim Low, and Deepak Bandyopadhyay. 2001. Shader lamps: animating real objects with image-based illumination. In *Proceedings of the Eurographics conference on Rendering* (EGWR'01). Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 89-101. https://doi.org/10.2312/EGWR/EGWR01/089-101

36. Majken K. Rasmussen, Esben W. Pedersen, Marianne G. Petersen, and Kasper Hornbæk. 2012. Shape-changing Interfaces: A Review of the Design Space and Open Research Questions. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (CHI '12). ACM, New York, NY, USA, 735–744. http://dx.doi.org/10.1145/2207676.2207781

37. Sharif Razzaque, Zachariah Kohn, and Mary C. Whitton. 2001. Redirected Walking. *In Proceedings of Eurographics* (EG '01), 289–294. http://dx.doi.org/10.2312/egs.20011036

38. Jun Rekimoto and Katashi Nagao. 1995. The world through the computer: computer augmented interaction with real world environments. In *Proceedings of the ACM Symposium on User Interface Software and Technology* (UIST '95). ACM, New York, NY, USA, 29-36. http://dx.doi.org/10.1145/215585.215639

39. Joan Sol Roo, Martin Hachet. Towards a Hybrid Space Combining Spatial Augmented Reality and Virtual Reality. In *Proceedings of the 2017 IEEE Symposium on 3D User Interfaces* (3DUI '17), Los Angeles, CA, USA, 195-198. https://doi.org/10.1109/3DUI.2017.7893339

40. Joan Sol Roo, Martin Hachet. One Reality: Augmenting How the Physical World is Experienced by combining Multiple Mixed Reality Modalities. In *Proceedings of the ACM Symposium on User Interface Software and Technology* (UIST '17). ACM, New York, NY, USA. https://doi.org/10.1145/3126594.3126638

41. Claus Scheiblauer, Michael Wimmer, Out-of-core selection and editing of huge point clouds. 2011. *Computers & Graphics, Volume 35, Issue 2*, Pages 342-351, ISSN 0097-8493, http://dx.doi.org/10.1016/j.cag.2011.01.004.

42. Ian Stavness, Billy Lam, and Sidney Fels. 2010. pCubee: A Perspective-corrected Handheld Cubic Display. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (CHI '10). ACM, New York, NY, USA, 1381–1390. http://doi.acm.org/10.1145/1753326.1753535

43. Vikas Singh, Deborah Silver and Nicu D. Cornea. 2003. Real-time volume manipulation. In *Proceedings of the Eurographics/IEEE TVCG Workshop on Volume graphics* (VG '03). ACM, New York, NY, USA, 45-51. http://dx.doi.org/10.1145/827051.827057

44. Adriana De Souza E Silva. 2009. Hybrid Reality and Location-Based Gaming: Redefining Mobility and Game Spaces in Urban Environments. *Simul. Gaming 40, 3* (June 2009), 404-424. http://dx.doi.org/10.1177/1046878108314643

45. Misha Sra, Sergio Garrido-Jurado, Chris Schmandt, and Pattie Maes. 2016. Procedurally generated virtual reality from 3D reconstructed physical space. In *Proceedings of the ACM Conference on Virtual Reality Software and Technology* (VRST '16). ACM, New York, NY, USA, 191-200. https://doi.org/10.1145/2993369.2993372

46. Richard Stoakley, Matthew J. Conway, and Randy Pausch. 1995. Virtual reality on a WIM: interactive worlds in miniature. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (CHI '95). ACM, New York, NY, USA, 265-272. http://dx.doi.org/10.1145/223904.223938

47. John Underkoffler and Hiroshi Ishii. 1998. Illuminating Light: An Optical Design Tool with a Luminous-tangible Interface. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (CHI '98). ACM, New York, NY, USA, 542–549. http://dx.doi.org/10.1145/274644.274717

48. John Underkoffler and Hiroshi Ishii. 1999. Urp: A Luminous-tangible Workbench for Urban Planning and Design. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (CHI '99). ACM, New York, NY, USA, 386–393. http://doi.acm.org/10.1145/302979.303114

49. Julien Valentin, Vibhav Vineet, Ming-Ming Cheng, David Kim, Jamie Shotton, Pushmeet Kohli, Matthias Nießner, Antonio Criminisi, Shahram Izadi, and Philip Torr. 2015. SemanticPaint: Interactive 3D Labeling and Learning at Your Fingertips. *ACM Trans. Graph.* 34, 5, Article 154. http://dx.doi.org/10.1145/2751556

50. John Viega, Matthew J. Conway, George Williams, and Randy Pausch. 1996. 3D magic lenses. In *Proceedings of the ACM Symposium on User Interface Software and Technology* (UIST '96). ACM, New York, NY, USA, 51-58. http://dx.doi.org/10.1145/237091.237098

51. Andrew D. Wilson. 2017. Fast lossless depth image compression. In *Proceedings of ACM Interactive Surfaces and Spaces* (ISS '17). ACM, New York, NY, USA. https://doi.org/10.1145/3132272.3134144

52. Ya-Ting Yue, Yong-Liang Yang, Gang Ren, and Wenping Wang. 2017. SceneCtrl: Mixed Reality Enhancement via Efficient Scene Editing. In *Proceedings of the ACM Symposium on User Interface Software and Technology* (UIST '17). ACM, New York, NY, USA. https://doi.org/10.1145/3126594.3126601