SceneCtrl: Mixed Reality Enhancement via Efficient Scene Editing

Ya-Ting Yue[1], Yong-Liang Yang[2], Gang Ren[3], Wenping Wang[1]

[1]The University of Hong Kong, Hong Kong, China, {ytyue, wenping}@cs.hku.hk
[2]University of Bath, Bath, UK , y.yang@cs.bath.ac.uk
[3]Xiamen University of Technology, Xiamen, China , rengang@xmut.edu.cn

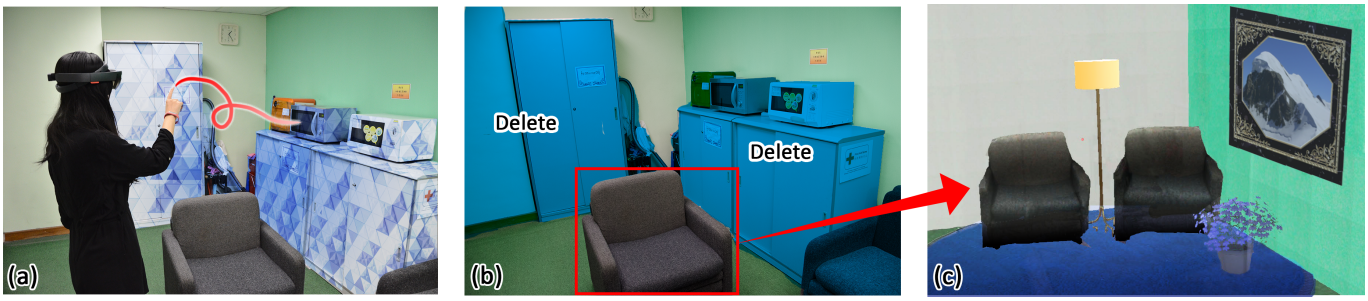


Figure 1. ***SceneCtrl*** **is a proof-of-concept system that enables efficient scene editing (delete, move, and copy real objects in the scene) for enhanced user experience in various mixed reality applications, such as room redecoration and environment design. In these applications, the reality often needs to be adapted to suit the virtuality. For instance,** ***SceneCtrl*** **allows the user to interactively select real objects in the scene by hand (a), then apply "Delete", "Move", or "Copy" to edit the scene (b), resulting in enhanced mixed reality experience due to resolving the conflicts between virtuality and reality (c).***

ABSTRACT

Due to the development of 3D sensing and modeling techniques, the state-of-the-art mixed reality devices such as Microsoft Hololens have the ability of digitalizing the physical world. This unique feature bridges the gap between virtuality and reality and largely elevates the user experience. Unfortunately, the current solution only performs well if the virtual contents complement the real scene. It can easily cause visual artifacts when the reality needs to be modified due to the virtuality (e.g., remove real objects to offer more space for virtual models), a common requirement in mixed reality applications such as room redecoration and environment design. We present a novel system, called *SceneCtrl*, that allows the user to interactively edit the real scene sensed by Hololens, such that the reality can be adapted to suit virtuality. Our proof-of-concept prototype employs scene reconstruction and understanding to enable efficient editing such as deleting, moving, and copying real objects in the scene. We also demonstrate *SceneCtrl* on a number of example scenarios in mixed reality, verifying that enhanced experience resolves conflicts between virtuality and reality.

ACM Classification Keywords

H.5.1 Information Interfaces and Presentation (e.g. HCI): Multimedia Information Systems - Artificial, augmented, and virtual realities

Author Keywords

Mixed reality, enhanced experience, scene editing.

INTRODUCTION

Recent advances in visual computing and interaction techniques have resulted in a new generation of Mixed Reality (MR) platforms, as Microsoft Hololens [11], that are able to sense the 3D space and model the shape of the reality. This significantly improves the immersive experience of MR applications, because the virtual contents can directly interact with the physical environment.

The current mechanism works well for cases where virtuality and reality are complementary (e.g., inserting virtual models on a real desk). However, in practice we often encounter the circumstances for which some incompatibilities occur and the involvement of virtuality requires a change of the reality. For instance, to redecorate an office, we may want to move the bookshelf to a new location or even replace it with a new bookshelf. Based on the current MR solutions, we will never achieve the desired visualization unless we manually modify the real scene by relocating the existing bookshelf, which can be very laborious. Otherwise the bookshelf will always be there and interfere with the new arrangement in the room, as see in Figure 2.

To address the above issue with conflicting virtuality and reality, we present a novel proof-of-concept system built upon Hololens, called *SceneCtrl*, to allow efficient scene editing for enhanced MR experience. The user can easily delete, move

*Please note that the Hololens images in this document are not captured through the device optics but from the device's video camera with digital objects rendered over that video feed. Actual display optics differs from these images as discussed in the "Optical See-through Experience" section.

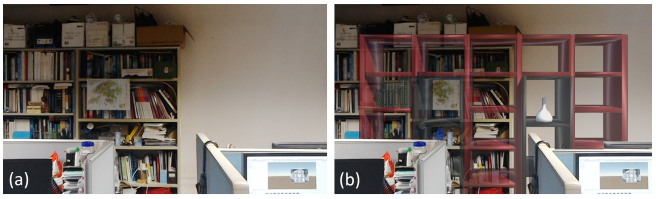


Figure 2. An example of incompatibility between virtuality and reality. (a) The real office scene. (b) The real bookshelf to be replaced interferes with the virtual bookshelf using the current MR solution in Hololens (captured using the RGB camera feed from Hololens).

or even copy the real objects in the scene, and resolve the incompatibility with the virtual contents. This facilitates a wide range of MR applications in which the reality needs to be adapted to suit the virtuality, such as room redecoration, environment design, etc. Compared with traditional MR solutions that facilitate virtual content manipulation (e.g., addition, alignment, deletion, etc.) in a real scene, the core of *SceneCtrl* is a set of intuitive scene editing tools that allow the user to edit *real* objects in the scene. To realize these tools solely on Hololens where only limited computational power and poorly sensed 3D data are available, we employ a set of straightforward yet effective scene reconstruction and understanding algorithms. Based on the 3D sensing ability provided by Hololens, we present a dynamic texture mapping and blending method to fuse texture information onto the raw geometric data. This enables automatic hole filling due to object removal, and free object rotation and relocation in the scene, making the edited scene visually plausible. We also analyze scene geometry and extract semantic planar parts to understand the global scene structure (e.g., floor, ceiling, wall), which facilitates the part-based object selection instead of tediously picking low-level geometric elements as triangles. By tracking the user's hand position, our system also allows intimate scene editing based on a gesture-driven object selection interface.

To demonstrate the applicability and scalability, we evaluate our system in four different scenarios with different editing contexts, ranging from indoor room redecoration to outdoor environment design. We also compare our results with the current MR solution available in Hololens. Both the results and comparisons exhibit the effectiveness of our system in resolving incompatible virtuality and reality.

In summary, we make the following contributions:

- A proof-of-concept system that allows efficient scene editing for enhanced MR experience;
- A set of novel scene editing tools based on scene reconstruction and understanding for efficiently editing *real* objects in the scene;
- A number of evaluation scenarios with conflicting virtuality and reality for demonstrating the effectiveness of the system.

RELATED WORK

Enhanced mixed reality and environment design

In recent years, a number of works have been presented to improve user experience and display quality in mixed reality from different perspectives. The appearance of real objects can be adjusted by projecting new light patterns based on radiometric compensation methods [21, 6, 36, 7]. IllumiRoom [16] and RoomAlive [15] further extend the physical room to a very large virtual display, providing novel user experience by mixing virtual contents and real environment. KinectFusion [14] reconstructs the scene geometry to allow intimate interaction between virtuality and reality. Spatial augmented reality could also be used to create a space enabling mixed reality for multiple viewers [2, 37, 18, 31]. Head mounted displays such as Hololens are becoming popular to enhance the user experiences in mixed reality applications, such as telecommunication [23], tourism [26] and scene editing [22]. Detailed illumination models are also used in indoor spaces to create realistic refurnishing visualization [38].

One major application of mixed reality is environment design. The design requirements have been identified [19] and various design platforms have been presented based on marker systems [25, 13]. Users could use various types of interaction methods like hand-held devices [22, 5] or gestures [1]. Furthermore, to enhance tangible user interaction, real object can be used together with virtual displays to provide the correlation between virtuality and reality [10]. More details of AR applications in building environments can be found in [35].

While improved user experience is achieved in different contexts, the scenario in which virtuality and reality are incompatible has not been considered so far. Our work presents a novel approach to resolve the conflicts by allowing the user to directly edit the real objects in the scene, enabling a number of mixed reality applications with enhanced user experience.

Scene modeling and understanding

With the fast development of 3D sensing techniques, how to effectively reconstruct the environment we are living in has gained attention in the visual computing field. Research has been conducted in different reconstruction contexts, ranging from indoor room scenes to outdoor urban scenes. A comprehensive review of scene modeling is beyond the scope of this paper. We refer interested readers to the recent survey papers such as [4] and [20] for indoor and outdoor reconstruction respectively. Our system utilizes the spatial mapping techniques embedded in Hololens for coarse scene geometry reconstruction.

Color image mapping optimization for 3D reconstruction has been well studied. State-of-the-art approaches based on consumer depth camera such as [39] can reconstruct the texture information of the entire scene by jointly optimizing camera poses and image mapping. However this technique is expensive in terms of time and space. We implement an efficient texture mapping and blending algorithm that only requires the user to take several photos of the region of interest, allowing texture synthesis in real time.

With the scene geometry being reconstructed, how to understand the scene semantics becomes very important for applications such as robotics, virtual reality, etc. Different approaches have been proposed and can be roughly classified into unsupervised methods as [30][40] and supervised methods as [9][8][32]. For our proof-of-concept system with coarse scene geometry, we apply a primitive-based unsupervised scene segmentation based on the built-in functions in HoloToolkit [12], resulting in a set of global structural elements of the scene, such as floor, ceiling, walls, etc.

SCENECTRL SYSTEM

SceneCtrl is implemented on the state-of-the-art MR platform Hololens [11]. Hololens is able to sense the real space, and model the 3D geometry of the environment based on advanced spatial mapping techniques. The reconstructed geometry enables immersive user experience of mixing the real scene with virtual contents, such as directly placing a virtual vase onto a real table, without using any artificial hints such as bar codes or other fudicial markers. *SceneCtrl* goes a step further by allowing efficient scene editing. This enhances the user experience when virtual contents and real scene are in conflict such as importing a new piece of furniture to replace the old one, rearranging objects in the scene, etc.

The current system has three major functions - "Delete", "Move", and "Copy", which correspond to eliminating, relocating and duplicating *real* objects in the scene respectively. To realize the above functions, we need to address the following challenges. First, deleting and moving objects can easily cause missing data that cannot be captured when digitalizing the scene in the first place, leading to visual artifacts in the mixed reality. Second, copying an object requires coupled geometry and texture information, since the copied object may be transformed and then viewed from a different perspective compared with the original object. Therefore, how to efficiently process the visual information in the scene (including 2D texture and 3D geometry) is the key of our system that comprises three components: scene reconstruction, scene understanding, and scene editing.

Scene Reconstruction

Our system is built upon the spatial mapping techniques of Hololens [33] to reconstruct the coarse scene geometry. The output from Hololens is a set of raw triangular mesh patches, the combination of which represents the entire real scene (see Figure 3). Note that these raw mesh patches contain no texture information and are only used for spatial sensing. To enable novel scene editing functions, we also need to create plausible textures for the mesh geometry within the region of interest (ROI), which contains the object to be moved or copied, or the objects/background behind the object to be deleted (we use textured mesh geometry behind to occlude the deleted objects to enable "Delete" function).

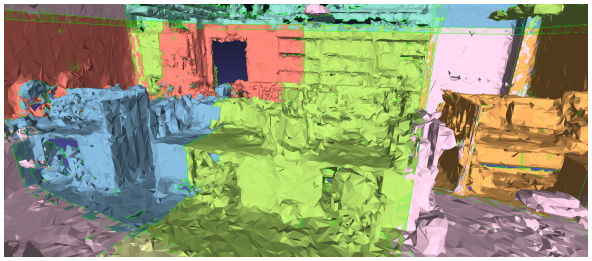
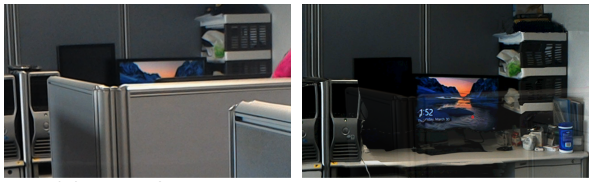


Figure 3. The reconstructed scene from Hololens comprises a set of coarse triangular mesh patches illustrated in different colors. Patch boundaries are highlighted as green lines. Neighboring patches have a strip-like overlapping area in-between.



(a) Original Scene (b) Scene Editing Result

Figure 4. Deleting the baffle boards in the scene reveals the objects behind, which requires taking photos of the blocked objects and synthesizing textures on them in order to generate plausible visualization. (a) The original scene. (b) The scene after edition (captured using the RGB camera feed from Hololens).

Thanks to the high-resolution camera of Hololens, we allow the user to take a number of photos for the ROI. And our system automatically synthesizes textures within the region for efficient editing. Figure 4 shows an example for deleting real objects in the scene. To observe the objects behind the baffle boards after deletion, we need to synthesize plausible texture information on them.

Dynamic Texture Synthesis

We employ a *dynamic* texture synthesis method (similar to [3]) to adaptively generate texture onto 3D geometry according to the user's perspective. The main reason is that the meshes generated from Hololens do not contain enough information, especially for darker objects or objects with complicated geometry. As a result, if we apply fixed texture captured from one perspective, it can easily lead to visual artifacts from a very different perspective due to highly distorted texture (as shown in Figure 5). Therefore, we take advantage of multiple photos taken from various perspectives, and dynamically synthesize texture from close-by perspectives. We find that plausible results (see Figure 6) can be achieved by directly adopting texture information from close-by perspective (within 30 degrees), or blending texture information from close-by perspectives.

Texture Mapping and Blending

For texture mapping from a single perspective, we simply project the captured photo onto the 3D geometry using the

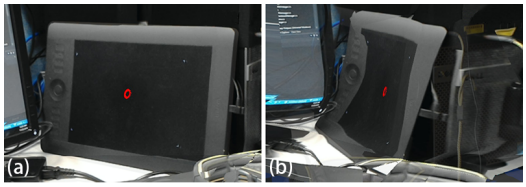


Figure 5. Directly apply color information captured from one perspective (a) to a very different perspective (b) results in visual artifacts with highly distorted texture.

Figure 6. Capturing color information in perspective (a) and (b), and dynamically synthesizing texture according to the current perspective generates much better visualization result.

camera projection information provided by Hololens. Now we will explain how to blend the texture information in the overlapping region of two photos from different perspectives. Note that multiple photos can also be interpolated by incrementally blending two photos. In practice, we find that blending two photos already provides good result, if the corresponding perspectives are close to the user's perspective. This strategy not only resolves the texture mapping distortions due to low quality mesh, but also reduces the texture color inconsistency caused by varying imaging conditions such as unstable exposure and white balancing (see Figure 7).

The key to blending textures from two photos is how to resolve color ambiguities and interpolate colors in the overlapping region. To identify the overlapping region, we un-project the two photos onto the 3D scene. We first estimate the corners of the two photos in the world coordinate system by 3D ray casting, then compute the intersection points of the un-projected photos. These feature points (illustrated as black points in Figure 8) are used to identify the overlapping region of the un-projected photos. Next, we project these feature points back onto the two photos, resulting in the corner points of the 2D overlapping region. Based on these corners points, we can identify the boundary and inner pixels of the overlapping region that will be used for blending textures.

The color of a boundary pixel is directly assigned from the photo in which it resides. The color of an inner pixel is interpolated based on its distance to the boundary of the overlapping region. Figure 9 illustrates the approach. For simplicity, the two photos are depicted in red and blue respectively. If the inner pixel is closer to the part of boundary (highlighted in blue), which lies inside the red photo while originating from the blue photo, its color should be closer to the red photo, and vice versa.

To achieve this, we sample pixels on the boundary of the overlap region, and calculate the blending weight W_B based on

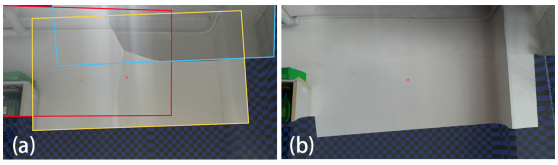


Figure 7. Texture synthesis comparison using three photos framed in red/yellow/blue. (a) The result generated by directly assigning pixel color from the photo with closest perspective. Visual artifacts appear in the middle due to color inconsistency among photos. (b) The result generated from texture blending eliminates the artifacts. The region with checker board texture is not covered by the three photos.

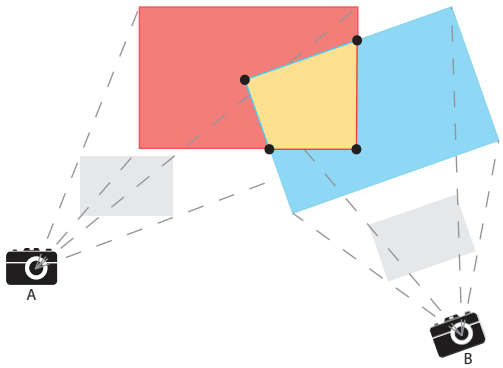


Figure 8. The overlapping region is identified by un-projecting the two photos into the 3D scene.

the distances from the inner pixel i to these pixels, as shown in Figure 9(a). The weight of inner pixel i regarding to the boundary is:

$$W_B = \frac{\sum_{j \in B} 1/d_{i,j}^2}{|B|}, \tag{1}$$

where $d_{i,j}^2$ is the square distance from inner pixel i to the sampled boundary pixel j, B is the set of sampled boundary pixels. The color $C(i)$ of inner pixel i is interpolated based on two sets of boundary pixels B_r and B_b, which originate from the boundary of red photo and blue photo respectively:

$$C(i) = \frac{W_{B_r}}{(W_{B_r} + W_{B_b})} C_r(i) + \frac{W_{B_b}}{(W_{B_r} + W_{B_b})} C_b(i), \tag{2}$$

where $C_r(i)$ and $C_b(i)$ represent the color of the corresponding inner pixel in red and blue photo respectively, and can be computed by image warping using the perspectives of the two photos. Figure 9(b) shows the texture blending result based on the boundary-driven principle illustrated in Figure 9(a).

In practice, our system incrementally blends textures in real time while the user takes photos. Only the photos in similar perspectives (within 30 degrees) are blended. Note that we have tried advanced texture mapping and blending algorithms, such as Poisson blending [24]. But it is computationally too expensive for Hololens (dozens of seconds cost) due to the limited computational power of Hololens (Intel Atom x5-Z8100 1.04 GHz) and the large photo size (2048×1152). On the

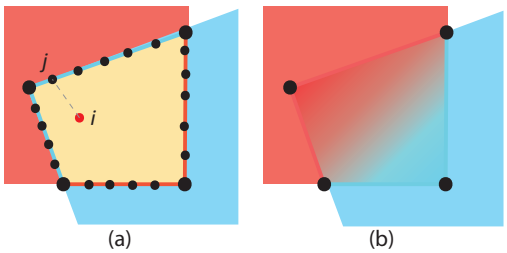


Figure 9. Blending colors in the overlapping region. (a) Compute distance from inner pixel i to a sampled boundary pixel j. (b) The color blending result.

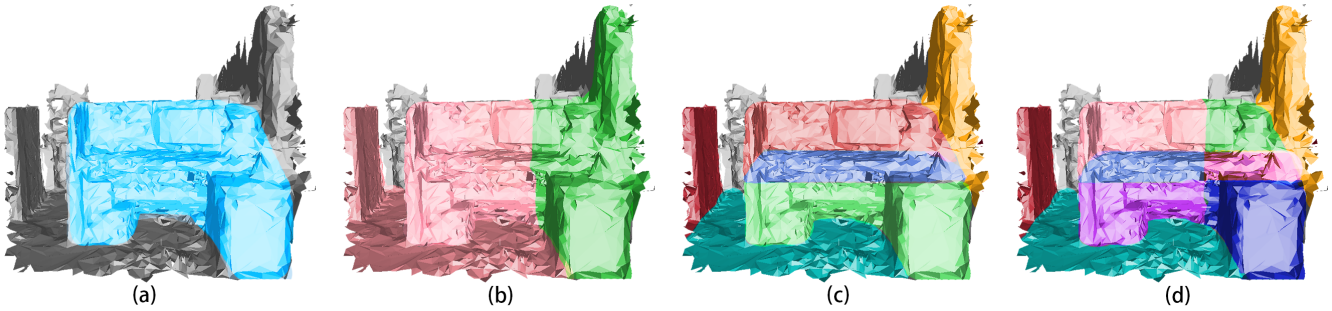


Figure 10. The blue part in (a) is a desk. But the raw reconstruction data from Hololens divides the scene into two patches (highlighted in red and green) in (b). And each patch contains more than one semantic object, e.g., wall, floor and (partial) desk. The ideal segmentation of the scene is shown in (c), but it is expensive to merge all the patches and process the whole scene. Instead, we segment and analyze semantic parts within each patch as in (d).

other hand, our method allows efficient blending of textures, providing immediate feedback to the user while taking photos.

Scene Understanding

Scene reconstruction results in a set of coarse mesh patches with texture information in the ROI. Although direct editing on the coarse geometry is possible, we can only rely on low-level geometric elements, i.e., the triangles. This is not convenient since the user usually wants to edit the scene at a higher level, such as edit the desk in Figure 10(a). On the other hand, high-level scene understanding can parse semantic objects from the scene. But they often require reasonable quality of the scene data, or computational power for sophisticated optimizations that are currently not available in Hololens. For example, SLAM++ [28] relies on prior knowledge from a database of specific objects, which is hard to generalize to arbitrary scenes. And the detection requires fine-grained GPU acceleration for real-time performance. SemanticPaint [34] requires high quality geometry and user interaction for segmentation, which will be challenging given the low quality mesh provided by Hololens. Our major goal is to facilitate the user to interactively select object in the scene, other than recognizing every semantic object in the scene. Therefore, we employ an unsupervised scene understanding method based on plane detection and the subsequent semantic classification (for 'floor', 'ceiling', 'wall', etc.), which is a good trade-off between robustness and efficiency.

Similar as in [27], *SceneCtrl* is able to identify planar parts based on planar extraction method provided by Hololens. Then the shape semantics are analyzed from the extracted planar parts based on prior knowledge such as plane position and orientation in the scene. For instance, the top horizontal planar surface above everything else is the 'ceiling'. In contrast, the lowest planar surface is the 'floor'. The remaining horizontal planes are labeled as 'desktop'. Similarly, the outer vertical planar surfaces are labeled as 'wall'. To filter small planar surfaces, we set a minimal area when clustering triangles. After the above semantic parts are recognized, the leftover scene geometry will be decomposed based on spatial connectivity.

Figure 10(b) shows an example of coarse mesh patches from scene reconstruction. *SceneCtrl* can label 'desktop' (blue), 'wall' (yellow) and 'floor' (dark green) as shown in Figure 10(c). Then the parts above and below the desktop (red and green respectively) can be clustered. Finally, based on connectivity, other parts can also be clustered, e.g., part of another desk (red) in Figure 10(c).

It is easy to see that the desk is divided into two parts in Figure 10(b) as Hololens stores the scene geometry as a set of mesh patches. It is expensive in terms of time and space to merge neighboring patches and process the whole scene. Therefore, we analyze each patch individually, leading to the result shown in Figure 10(d).

The current plane-based clustering can easily result in over-segmentation. For instance, the desk in Figure 10 is divided into several parts. To facilitate scene editing, we develop an intuitive gesture-driven sketching tool to *select* an object in the scene.

Scene Editing

With the help of scene reconstruction and scene understanding, *SceneCtrl* allows intuitive and efficient scene editing operations for enhanced mixed reality applications, in which virtual content and the real scene conflict each other.

Select

To edit the scene, the user needs to first specify the ROI. This is made simple using a sketch-based selection tool driven by the movement of the user's hand. Since Hololens is able to track hand position, the user can simply draw a space curve by hand, and *SceneCtrl* automatically projects the curve onto the scene geometry according to the current perspective (see Figure 11(a) and supplemental video). The user can use this interface to select multiple parts in the scene (see Figure 11(b)), circumventing the inconvenience caused by over-segmentation.

Delete, Move and Copy

SceneCtrl offers three major editing functions - "Delete", "Move" and "Copy". "Delete"/"Move" objects in the scene can cause holes on background scene geometry such as the floor, wall, etc. Based on semantic segmentation, we fill such holes by plane fitting. To improve efficiency, we do not seamlessly fill in triangles for each hole, but import a plane to fit the hole. The texture of the plane will be copied from nearby planar background surface. The above approach helps to filter the

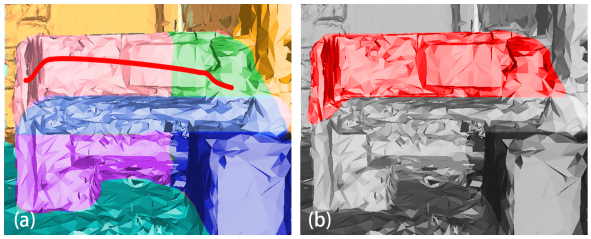


Figure 11. (a) The user can select object by sketching in the scene. The curved sketch in red is generated by tracking the user's hand and project its trace onto the scene. (b) The incident parts are selected for further editing (delete, move, or copy).

noise of planar surface and save time for filling every small, irregular holes on the coarse mesh patch. Also, "Delete"/"Move" objects may reveal the objects behind. Dynamic textures are generated for the occluded objects based on photos taken by the user (Figure 4). In addition, "Move" an object requires texture update according to the current perspective. "Copy" an object is similar to "Move" but no hole will be generated because the selected object is duplicated, not removed. All of the three functions could be triggered by voice instructions or gestures (see the supplemental video).

EXAMPLE SCENARIOS

SceneCtrl can be used in a wide range of MR applications where the reality has to be adapted to suit the virtuality. We have evaluated *SceneCtrl* on four example MR scenarios, including both indoor and outdoor scenes, in which the user experience is largely improved due to the availability of scene editing.

Note that Hololens is an optical see-through device that projects holographic images on the lenses to render virtual contents. As a result, the deleted real-world object cannot be fully occluded by projecting the images of reconstructed ROI that originally lies behind. In practice, the virtual models we see through Hololens look a little brighter than those shown in the figures, which are captured using the RGB camera feed from Hololens. For the detailed discussion of optical see-through experience, please refer to the "User Evaluation" Section.

Kitchen2MeetingRoom

In this example scenario, the user changed the function of the room from kitchen to meeting room (see Figure 12). Based on *SceneCtrl*, the user first applied "Delete" to the cabinet, microwave, etc., and only kept the sofa in the scene. Then the user applied "Copy" to the sofa and then applied "Move" to relocate the sofa. With additional decorations, such as the floor lamp and the flower, the new room can be realistically visualized without any interference from the underlying real scene. Note that the shape and texture of the sofa are successfully reconstructed. And the hole in the wall due to object deletion is automatically filled. (see also the supplemental video).

Kitchen2Bedroom

The user changed the function of the room from kitchen to bedroom in this example scenario (see Figure 13). Unlike the previous scenario, the user kept part of the scene unchanged (a row of sofa on the right), and only applied "Delete" to the remaining objects in the scene. New pieces of furniture were added to the scene afterwards, including double bed, wall lamp, etc. The virtual contents and the real objects fit together well in the final scene visualization.

PublicSpace2Gallery

In this example scenario, the user redesigned a public space to a gallery (see Figure 14). The user applied "Delete" to every object in the public space except the white table. A number of artworks were placed in the updated scene, including a vase on the table.

OutdoorSpace2Exhibition

The user redesigned an outdoor space to an exhibition place in this example scenario (see Figure 15). To create the space for the exhibition booth, the obstacles in the middle of the scene (highlighted in blue) were removed by the "Delete" function. An exhibition booth was placed there instead. Note that outdoor scene is large and we do not need to reconstruct the geometry of the entire scene. In the case, only the floor and the front part of brushwood (framed in red boxes) were reconstructed.

Comparison

To highlight the effectiveness of *SceneCtrl*, we show a comparison with the state-of-the-art MR solution (i.e., Hololens) based on the Kitchen2Bedroom example. Figure 16(a) shows the MR experience in Hololens. The virtual bed can easily collide with the real sofa and cabinet in the scene due to the lack of empty space. A naive way to eliminate collision is to apply traditional AR solution, i.e., impose virtual models on top of the real scene, as shown in Figure 16(b). However, this leads to the well-known spatial ambiguity. Specifically, the spatial relation between the bed and the cabinet is confusing. They are both aligned to the wall but the bed appears to be in front of the cabinet. Figure 16(c) shows the enhanced mixed reality experience enabled by *SceneCtrl*, where the bed naturally fits into the scene.

USER EVALUATION

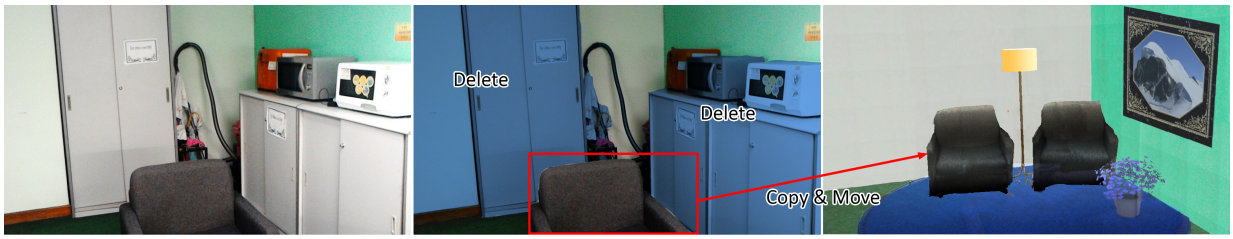
To evaluate *SceneCtrl*, we recruited 10 volunteers on campus (ages 20 – 30, 4 female). Only one participant had experience of using Hololens before (played a simple game with Hololens). Participants used our system in two different scenarios as shown in Figure 12 (#1) and Figure 14 (#2).

Procedure

We demonstrated the type of task to the participants via our supplemental video, then let them try our system with step by step guidance. Most novice users could learn the editing functions in 15 minutes.

After they became familiar with the system, they were asked to redecorate the corresponding room by referring to Figure 12, 13 or 14 respectively. The users were encouraged to freely edit the scene, such as copy and move different real objects or add virtual models they favored.

Immediately following the testing, we asked the participants to fill out a Simulator Sickness Questionnaire (SSQ) [17],

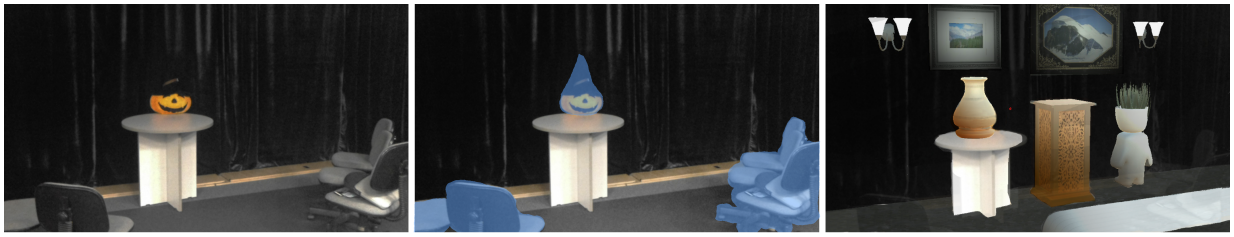


(a) Original Scene (b) Edit Parts (c) Result

Figure 12. Example Scenario 1 - Kitchen2MeetingRoom. The user rearranges a kitchen into a meeting room (the result image is captured using the RGB camera feed from Hololens).

(a) Original Scene (b) Edit Parts (c) Result

Figure 13. Example Scenario 2 - Kitchen2Bedroom. The user changed the room function from kitchen to bedroom (the result image is captured using the RGB camera feed from Hololens).



(a) Original Scene (b) Delete objects highlighted in blue (c) Result

Figure 14. Example Scenario 3 - PublicSpace2Gallery. The user redesigned a public space to a gallery (the result image is captured using the RGB camera feed from Hololens).

(a) Original Scene (b) Reconstruct (red) and Delete (Blue) (c) Result

Figure 15. Example Scenario 4 - OutdoorSpace2Exhibition. The user rearranged an outdoor space to an exhibition place (the result image is captured using the RGB camera feed from Hololens).

Figure 16. Comparison between different MR experiences. (a) Virtual models (the bed) can collide with the real objects (the sofa) in the current mixed reality solution in Hololens. (b) Overlay virtual models on top of real scene results in ambiguous spatial relation. The bed is actually aligned to the wall as the cabinet, but it looks like in front of the cabinet. (c) Enhanced mixed reality experience enabled by *SceneCtrl*. Cabinet and sofa are deleted from the scene. (Images are captured using the RGB camera feed from Hololens.)

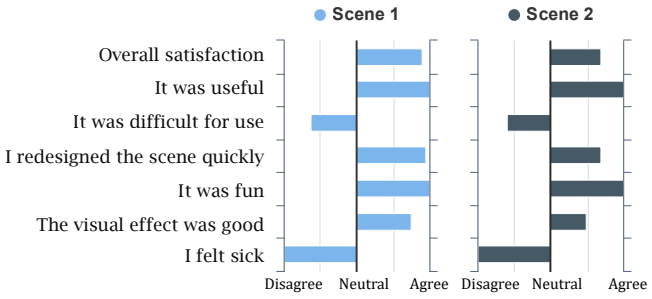


Figure 17. Mean rating of two scenes in the user study. Participants rated eight questions on a five point Likert scale from "strongly disagree" to "strongly agree".

evaluating their overall comfort level. Then the participants were asked to rate the "overall satisfaction" and other questions regarding system usefulness, usability, enjoyment and visual experience for their scenarios.

Results & Feedback

Participants responded positively to the system. As expected, no participant felt sick during the editing (see Figure 17) and the mean value of total SSQ score is 5.98 as shown in Figure 18, which is much lower than the normal ranges for playing stereoscopic video games [29]. This is because participants do not have big movements in our case. In addition, they could see the real world through Hololens, which helps to decrease nausea and disorientation. The most uncomfortable parts are related with Oculomotor and female is more sensitive.

In general, all the participants think *SceneCtrl* is useful and the editing process is fun (see Figure 17). Some participants said "It looks like I really moved the sofa/table.", "It is cool that I can delete real objects."

DISCUSSIONS

Optical See-through Experience

Due to the optical see-through principle of Hololens, when "deleting" a real object in the scene, the reconstructed ROI behind cannot fully occlude the deleted object in front. However, brighter ROI provides better occlusions. In practice, participants tend to be attracted by the reconstructed ROI and the *ghost* of deleted object is ignored. All the participants expressed that the edited scene was visually plausible. However as expected, the visual experience of the second scene is not so good as the first one (see Figure 17). This is because the

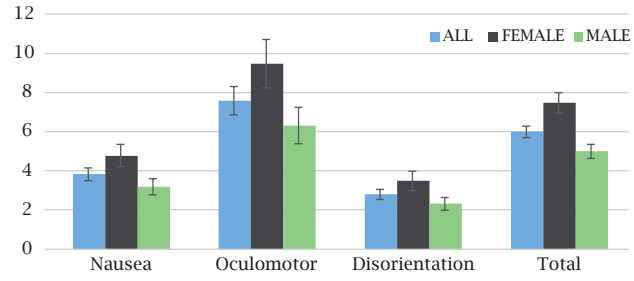


Figure 18. Mean values of SSQ (error bars show 95% CI).

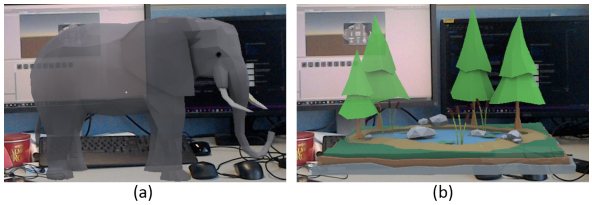


Figure 19. The internal mechanism of Hololens can cause unwanted visual artifacts when mixing virtuality and reality. (a) Brighter objects in the real scene such as the computer display can still be perceived behind darker virtual models, but will be occluded by virtual models in bright color as in (b). However, darker objects in the real scene can be easily occluded by virtual content, such as the powered off monitor on the right of (a), compared with the computer display on the left. (Images are captured using the RGB camera feed from Hololens.)

ROI behind the deleted object is dark, which results in weaker occlusion. Finally, the field of view of Hololens is limited. But most participants did not complain about this. Only one participant who wore glasses mentioned this problem, since glasses enlarged the distance between eyes and lenses.

Limitations

While *SceneCtrl* creates an enhanced MR experience when virtuality and reality are in conflict, the current implementation still has some limitations.

First, due to the low quality of the reconstructed scene geometry from Hololens, we allow the user to take photos for the ROI to generate plausible textures for the editing purpose. If the user observes the new scene in a significantly different perspective from those of the photos, there will be noticeable visual artifacts caused by distorted textures. Capturing more photos can alleviate this effect but requires extra user effort.

Also, as discussed before, the user's visual experience can be affected by the optical see-through characteristics of Hololens. The main challenge we found is that the deleted real objects, especially those bright ones, can still appear *ghosted* in the resulting visualization, even if they are occluded by some virtual models (as highlighted in Figure 19).

Finally, the lighting condition in the scene is taken into account by allowing the user to manually add and adjust light source. This relies on the user's expertise and may lead to unrealistic illumination effects. Advanced inverse rendering technique could be further applied to faithfully estimate the lighting condition, but is beyond the scope of this paper.

CONCLUSION AND FUTURE WORK

In this work, we present a novel system, called *SceneCtrl*, to allow efficient scene editing for enhanced mixed reality. The ability of interactively deleting, moving and copying real objects in the scene eliminates the collision and inconsistency between virtuality and reality and significantly improves the user experience. The editing functions are made possible by effective scene reconstruction and understanding, and an intuitive gesture-based object selection tool. We demonstrate *SceneCtrl* on various example scenarios to verify its usefulness for resolving conflicts between virtuality and reality.

In the future, we would like to employ state-of-the-art scene understanding techniques to analyze the semantics of the objects (e.g., chair, table) in the scene. This would further improve the efficiency of scene editing, such as object selection and removal.

ACKNOWLEDGMENTS

We thank all the volunteers for participation, reviewers for helpful comments. This work was supported in part by EPSRC Grant (EP/M023281/1), Fujian Provincial Social Science Project (FJ2016C095), Xiamen Overseas Scholar Project (XRS2016 314-10), and Fujian Provincial Natural Science Project (2017J01784).

REFERENCES

1. Santiago Arroyave-Tobón, Gilberto Osorio-Gómez, and Juan F Cardona-McCormick. 2015. Air-modelling: a tool for gesture-based solid modelling in context during early design stages in AR environments. *Computers in Industry* 66 (2015), 73–81.
2. Hrvoje Benko, Andrew D Wilson, and Federico Zannier. 2014. Dyadic projected spatial augmented reality. In *Proceedings of the 27th annual ACM symposium on User interface software and technology*. ACM, 645–655.
3. Chris Buehler, Michael Bosse, Leonard McMillan, Steven Gortler, and Michael Cohen. 2001. Unstructured Lumigraph Rendering. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '01)*. ACM, New York, NY, USA, 425–432.
4. Kang Chen, Yu-Kun Lai, and Shi-Min Hu. 2015. 3D indoor scene modeling from RGB-D data: a survey. *Computational Visual Media* 1, 4 (2015), 267–278.
5. Kai-Yin Cheng, Yu-Hsiang Lin, Yu-Hsin Lin, Bing-Yu Chen, and Takeo Igarashi. 2011. Grab-carry-release: manipulating physical objects in a real scene through a smart phone. In *SIGGRAPH Asia 2011 Emerging Technologies*. ACM, 13.
6. Michael D Grossberg, Harish Peri, Shree K Nayar, and Peter N Belhumeur. 2004. Making one object look like another: Controlling appearance using a projector-camera system. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, Vol. 1. IEEE, I–I.
7. Anselm Grundhofer and Oliver Bimber. 2008. Real-time adaptive radiometric compensation. *IEEE transactions on visualization and computer graphics* 14, 1 (2008), 97–108.
8. Saurabh Gupta, Pablo Arbeláez, Ross Girshick, and Jitendra Malik. 2015. Indoor Scene Understanding with RGB-D Images: Bottom-up Segmentation, Object Detection and Semantic Segmentation. *Int. J. Comput. Vision* 112, 2 (2015), 133–149.
9. S. Gupta, P. ArbelÂÍÂćez, and J. Malik. 2013. Perceptual Organization and Recognition of Indoor Scenes from RGB-D Images. In *2013 IEEE Conference on Computer Vision and Pattern Recognition*. 564–571.
10. 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)*. 1957–1967.
11. Hololens. 2017. http://www.hololens.com. (2017).
12. HoloToolkit. 2017. https://github.com/Microsoft/HoloToolkit-Unity. (2017).
13. Fu-Jen Hsiao, Chih-Jen Teng, Chung-Wei Lin, An-Chun Luo, and Jinn-Cherng Yang. 2010. Dream Home: a multiview stereoscopic interior design system. In *IS&T/SPIE Electronic Imaging*. International Society for Optics and Photonics, 75250J–75250J.
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 24th Annual ACM Symposium on User Interface Software and Technology (UIST '11)*. 559–568.
15. Brett Jones, Rajinder Sodhi, Michael Murdock, Ravish Mehra, Hrvoje Benko, Andrew 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 27th Annual ACM Symposium on User Interface Software and Technology (UIST '14)*. 637–644.
16. 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)*. 869–878.
17. Robert S Kennedy, Norman E Lane, Kevin S Berbaum, and Michael G Lilienthal. 1993. Simulator sickness questionnaire: An enhanced method for quantifying simulator sickness. *The international journal of aviation psychology* 3, 3 (1993), 203–220.
18. Jarrod Knibbe, Hrvoje Benko, and Andrew D Wilson. 2015. Juggling the effects of latency: Motion prediction approaches to reducing latency in dynamic projector-camera systems. (2015).
19. Tiina Kymäläinen and Sanni Siltanen. 2013. Co-Designing Novel Interior Design Services that Utilise Augmented Reality: A Case Study. *Advanced Research and Trends in New Technologies, Software, Human-Computer Interaction, and Communicability* (2013), 269.
20. P. Musialski, P. Wonka, D. G. Aliaga, M. Wimmer, L. Gool, and W. Purgathofer. 2013. A Survey of Urban Reconstruction. *Comput. Graph. Forum* 32, 6 (2013), 146–177.

21. Shree K. Nayar, Harish Peri, Michael D. Grossberg, and Peter N. Belhumeur. 2003. A projection system with radiometric compensation for screen imperfections.
22. Benjamin Nuernberger, Eyal Ofek, Hrvoje Benko, and Andrew D. Wilson. 2016. SnapToReality: Aligning Augmented Reality to the Real World. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. 1233–1244.
23. Sergio Orts-Escolano, Christoph Rhemann, Sean Fanello, Wayne Chang, Adarsh Kowdle, Yury Degtyarev, David Kim, Philip L Davidson, Sameh Khamis, Mingsong Dou, and others. 2016. Holoportation: Virtual 3D Teleportation in Real-time. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*. ACM, 741–754.
24. Patrick Pérez, Michel Gangnet, and Andrew Blake. 2003. Poisson Image Editing. *ACM Trans. Graph.* 22, 3 (July 2003), 313–318.
25. Viet Toan Phan and Seung Yeon Choo. 2010. Interior design in augmented reality environment. *International Journal of Computer Applications* 5, 5 (2010).
26. Christina Pollalis, Whitney Fahnbulleh, Jordan Tynes, and Orit Shaer. 2017. HoloMuse: Enhancing Engagement with Archaeological Artifacts through Gesture-Based Interaction with Holograms. In *Proceedings of the Tenth International Conference on Tangible, Embedded, and Embodied Interaction*. ACM, 565–570.
27. Renato F Salas-Moreno, Ben Glocken, Paul HJ Kelly, and Andrew J Davison. 2014. Dense planar SLAM. In *Mixed and Augmented Reality (ISMAR), 2014 IEEE International Symposium on*. IEEE, 157–164.
28. Renato F Salas-Moreno, Richard A Newcombe, Hauke Strasdat, Paul HJ Kelly, and Andrew J Davison. 2013. Slam++: Simultaneous localisation and mapping at the level of objects. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1352–1359.
29. Jonas Schild, Joseph LaViola, and Maic Masuch. 2012. Understanding User Experience in Stereoscopic 3D Games. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '12)*. ACM, New York, NY, USA, 89–98.
30. R. Schnabel, R. Wahl, and R. Klein. 2007. Efficient RANSAC for Point-Cloud Shape Detection. *Computer Graphics Forum* 26, 2 (2007), 214–226.
31. Carsten Schwede and Thomas Hermann. 2015. HoloR: Interactive mixed-reality rooms. In *Cognitive Infocommunications (CogInfoCom), 2015 6th IEEE International Conference on*. IEEE, 517–522.
32. S. Song and J. Xiao. 2016. Deep Sliding Shapes for Amodal 3D Object Detection in RGB-D Images. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 808–816.
33. SpatialMapping. 2017. https://developer.microsoft.com/en-us/windows/mixed-reality/spatial_mapping. (2017).
34. 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 Transactions on Graphics (TOG)* 34, 5 (2015), 154.
35. Xiangyu Wang, Mi Jeong Kim, Peter ED Love, and Shih-Chung Kang. 2013. Augmented Reality in built environment: Classification and implications for future research. *Automation in Construction* 32 (2013), 1–13.
36. Gordon Wetzstein and Oliver Bimber. 2007. Radiometric compensation through inverse light transport. (2007).
37. Andrew Wilson, Hrvoje Benko, Shahram Izadi, and Otmar Hilliges. 2012. Steerable augmented reality with the beamatron. In *Proceedings of the 25th annual ACM symposium on User interface software and technology*. ACM, 413–422.
38. Edward Zhang, Michael F. Cohen, and Brian Curless. 2016. Emptying, Refurnishing, and Relighting Indoor Spaces. *ACM Trans. Graph.* 35, 6 (2016), 174:1–174:14.
39. Qian-Yi Zhou and Vladlen Koltun. 2014. Color Map Optimization for 3D Reconstruction with Consumer Depth Cameras. *ACM Trans. Graph.* 33, 4 (2014), 155:1–155:10.
40. Qian-Yi Zhou and Ulrich Neumann. 2013. Complete Residential Urban Area Reconstruction from Dense Aerial LiDAR Point Clouds. *Graph. Models* 75, 3 (2013), 118–125.