All-Around Virtual Photo Walls

Selin Barash
ETH Zürich
bselin@ethz.ch

Boris Bernegger
ETH Zürich
bboris@ethz.ch

Ahmet Özüdoğru
ETH Zürich
oahmet@ethz.ch

Onat Vuran
ETH Zürich
ovuran@ethz.ch

Abstract

We present "All-Around Virtual Photo Walls", an application developed for Microsoft HoloLens 2 that augments a real-world room with virtual photo walls. We bring the idea of photo walls to the space of mixed reality by displaying the photos only virtually through the aid of a mixed reality device. This approach is less permanent, more flexible, and sustainable compared to traditional photo walls. The presented application allows a user to pre-scan a room. After detecting suitable vertical surfaces, the application generates photo wall layouts and populates the world with photos located in the camera roll folder of the user. We implement two algorithms for automatic layout generation. By employing asynchronous layout generation and pre-fetching of photos from disk we can ensure a smooth user experience. Furthermore, our application provides a suite of customization options to the user. These allow the user to adjust the auto-generated layouts to their personal liking. We perform a user study and evaluate the results to consolidate a list of further improvements. The study shows that the presented application provides an overall positive user experience and fulfills its purpose. The code of our application is available under https://github.com/MixedRealityETHZ/RealityMixers

1. Introduction

Motivation Photos hold a special place in our lives, serving as tangible reminders of important moments and loved ones. Traditional photo walls, in which physical prints are displayed on a wall or other surface, have long been a popular way to showcase these memories. However, creating and maintaining a traditional photo wall can be time-consuming and costly, as it requires the physical printing and framing of photographs.

In recent years, mixed reality (MR) technology has emerged as a promising platform for augmenting the physical world with digital content. In this paper, we present "All-Around Virtual Photo Walls", an application that brings the idea of photo walls into the MR space. Our application allows users to create virtual photo walls in their real-world surroundings by displaying photographs only virtually while still considering the topology of the real-world environment. This approach offers several benefits over traditional photo walls: it is less permanent, more flexible, and more sustainable.

Relevant Tools and Resources Our application was developed for the Microsoft HoloLens 2 [1], an MR headset that allows users to see and interact with holographic content in the real world. The HoloLens 2 features a wide field of view, a high-resolution display, and advanced spatial mapping capabilities, which make it well-suited for creating and interacting with virtual photo walls.

To develop the application, we used a range of tools and resources, including the Mixed Reality Toolkit 2 (MRTK 2) [8], the Unity [11] game engine, and the C# programming language.

Related work There are numerous websites (e.g., [2]) and smartphone applications (e.g., [3]) that allow users to create virtual wall galleries. A recent publication /cite presents an app for designing photo galleries using augmented reality. This app suggests fitting photos and layouts around a chosen focal object. While there are some similarities between this app and our own, our application does not require any focal objects and lets the user choose a layout algorithm among the implemented ones.

Main Contributions We have developed an application that successfully scans a physical room, selects suitable vertical surfaces for photo walls, and places images on them according to an algorithm chosen by the user. To evaluate the performance and user experience of our application, we conduct a user study and analyze the results. Based on the findings, we suggest a list of further improvements that could enhance the overall user experience of our application.

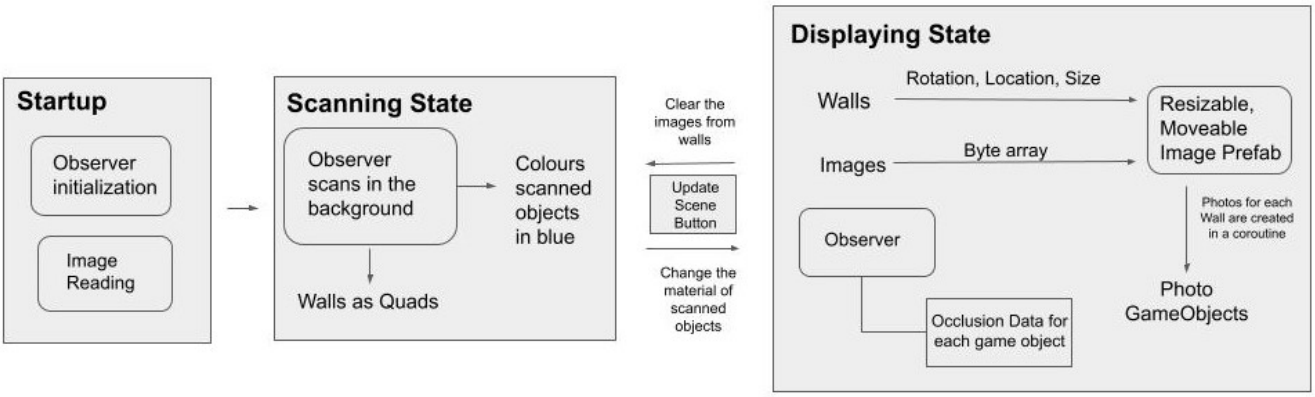


Figure 1. App architecture

Figure 2. Main menu

2. App Design

After the startup the application alternates between two states (see in Figure 1): A scanning state and a display state. It starts in the scanning state, during which the user walks around the room and scans the walls they want to decorate with photos by simply moving closer and inspecting them visually. Successfully scanned surfaces are highlighted with a blue mesh. The user can then open the main menu by holding their hand in front of them and facing their palm towards the HoloLens. The user can see at what state the app is by checking the text box on top of the main menu. There is also a dropdown menu to choose a layout algorithm and a button to change the displaying state. We have implemented two algorithms for automatic layout generation: Random placement and circular placement.

Random Placement places photos at random locations on the wall while being careful not to intersect them. Circular Placement on the other hand starts by putting a picture in the middle of the wall and then continues placing the photos in the circular format centered around the initial one. Furthermore, the user can customize the layouts to their personal liking by moving and resizing the individual photos. To ensure a smooth user experience, we pre-fetch the photos from the disk and implemented batch loading, meaning we asynchronously generate the layout for each wall.

3. Implementation

Our application backend consists of a controller and two backend components, an observer and a photo wall generator. The observer is an instance of WindowsSceneUnderstandingObserver Class [7] provided by the MRTK 2. It scans the environment using the HoloLens Lidar sensor and RGB camera and detects all physical objects in the scene. PhotoWallGenerator class is a custom component responsible for creating the images on the provided walls. It supports two different photo placement algorithms which are described in detail in section 3.5.

3.1. Controller

The controller is responsible for reading the user image data from the Windows Photo Library and the communication between the backend components and the UI. Our app has two states as described in section 2. The controller also handles these state transitions according to the user input and updates the text box in the main menu about the changes.

At the start of the application, the controller reads the images in the user's Photo Library, initializes the observer, and switches the app to the scanning state.

In Scanning State, the controller sets the observer update interval to 5 seconds. The observer default material is changed to MRTK Wireframe Blue Material to inform the user how much of the actual world is already scanned by the HoloLens.

In the transition to the display state, our controller takes a snapshot of all the objects detected until the transition time and reduces the observer update interval to 8 seconds to ensure better performance while interacting with photos.

Among the detected objects, the controller filters the walls and saves their location information for the PhotoWallGenerator. The controller further converts all detected physical object materials and the default material of the observer to the MRTK Occlusion Material. This change ensures that physical objects can provide occlusion to virtually placed photos. The PhotoWallGenerator coroutine is started.

In the display state, while the PhotoWallGenerator coroutine is running, photo walls are generated wall by wall. During the generation and after the generation is over, the user can look and interact with the placed photos. WindowsSceneUnderstandingObserver continues to scan the room in the background. Newly observed objects are only used for occlusion purposes, no new photos are placed even if there is a new wall in the scene.

In the transition to the scanning state, all previously placed photos and the virtual walls are deleted. All observed object materials and the default material of the observer are converted back to the MRTK Wireframe Blue material for display.

File reading uses different default locations for images in Unity and on the device. On the device, we use the Windows.Storage.KnownFolders.PicturesLibrary and on the editor we use a demo Images folder for debugging. To use the Picture Library, one needs to request Picture Library access from the user. Therefore, that capability is checked from Unity Publishing Settings.

3.2. The Observer

The observer component is an instance of the WindowsSceneUnderstandingObserver class. It scans the environment using the Lidar sensor and RGB Camera, creates groups of meshes or quads, and assigns a label to each of them. Labels can be Floor, Ceiling, Wall, Platform, Background, World, or Unknown. Along with these labels it creates a GameObject for each group of meshes. When a new object is detected it assigns its *Default Material* to the newly observed object and displays it. It runs in the background and updates the scene every *Update Interval* seconds. We also set the mesh searching radius to three meters because the increased radius creates too many objects for HoloLens 2 to handle.

In our app, we use a medium level of detail for meshes. Our controller sets the *Default Material* and *Update Interval* parameters depending on the state and sets *Mesh Grouping Type* to "Use Quads" at the start.

3.3. Photo Wall Generator

Photo Wall Generator is responsible for creating the virtual photo wall for a given set of images and wall locations. For each wall, it takes a wall processing followed by an image placement step **(Batch Loading)**. It looks to the DropDown menu in the UI for algorithm selection.

3.3.1 Wall Processing

After semantic understanding of the scene as described in section 3.1 our application knows the parameters for a set of walls in the physical world. For each of the detected walls, an instance of the custom *Wall* class is created. The *Wall* class stores the parameters of the physical wall and creates an invisible Unity GameObject and maps it to the corresponding location in the virtual world. It also determines which side of the wall is facing the user.

For each wall, a photo wall layout must be generated and displayed. This is done asynchronously to display some viewable content to the user as soon as possible. To generate the layouts, each wall is passed to an instance of the *PhotoWallGenerator* class which applies one of the layout generation algorithms described in section 3.5. The algorithm returns an instance of the *Layout* class. This instance describes where a set of selected images should be displayed relative to a two-dimensional view of the Wall.

3.3.2 Image Placement

The images that are displayed in the virtual world are represented as instances of a custom *Photograph* class. This class fulfills several purposes. First, it stores the metadata, such as aspect ratio, of the source image as well as the source image itself in the form of a bytes array. It also provides a *Draw* member function, which is called once the image should be displayed in the virtual world. The *Draw* method instantiates a Unity GameObject from a prefab to increase performance. The wall on which the image should be displayed is then set as the parent of the freshly generated instance. This allows us to place the photos relative to the wall, without the need to consider the placement of the wall in three dimensions. They then appear at the proper location in the real world once the wall is placed there. The HoloLens 2 styling is applied to the handles and bounding boxes. The source image is applied as a texture to the front of the reshaped GameObject.

3.4. User Interface

Our User Interface consists of a hand-tracking main menu and interactable photos on the generated virtual photo wall.

3.4.1 Interactable Photos

Photo objects are instantiated using a Photograph Prefab. This prefab uses an opaque material with a standard unity shader to allow occlusion by other objects. For visual aesthetics, we add handles to the instantiated GameObject in creation.

For movement, we are using Object Manipulator Script [9] provided by Microsoft. This script, along with the Near-

InteractionGrabable script, allows the object to be moved in any direction. It also makes the photos easily grabbable. We added a constraint to block movement perpendicular to the walls.

The scaling capability of the photos is implemented using the BoundsControl script described in [6]. The photos are flattened in the depth dimension. The rotation and scaling along the wall's normal direction are constrained using a constraint script.

3.4.2 Hand Tracking Menu

Our Hand Tracking Menu uses SolverHandler and HandConstraintPalmUp scripts to be able to track the hand and to get activated when the palm is directed upwards. For the parameters of the tracking, we are using Microsoft-recommended values. The article and the default practices for hand menus can be accessed from [5] and [4].

Our menu consists of a ButtonGroup with one button, which is of type PressableButton HoloLens 2 presented in [10] and a dropdown menu. Dropdown Menu is an instance of Unity Dropdown presented in [13]. Unity components do not work natively on HoloLens 2, that's why it needs to be placed inside an MRTK Canvas. To make it interactable by touch in addition to the pointer, we added a NearInteractionTouchableUnityUI script which receives and processes pointer events.

3.5. Layout Generation Algorithm

We develop and implement two photo wall layout generation algorithms. A photo wall consists of a (multi-)set of virtual images displayed on a real wall. A photo wall *layout* specifies the location, scale, and orientation of the virtual images in the virtual world. For a sensible placement, it must be aware of the real-world topology of a room. Our algorithm takes as input the dimensions of a single virtual wall, for which the mapping to the real world is known, and a set of images and generates a photo wall layout.

3.5.1 Random Placement Algorithm

Our first algorithm places the photos randomly on the wall such that no two photos on that wall overlap and that they are contained within the boundaries of the wall. The algorithm also takes a parameter that limits the number of photos that can be displayed per wall. We set this limit to a maximum of 25 photos per wall. This was introduced for performance reasons.

The algorithm places photos by iterating through the set of provided images. For each image, it tries to place the photo randomly on the wall, such that it is contained within the wall. It then checks if the current photo overlaps any of the previously placed photos. If this is the case, the placement is rejected and the photo is not displayed on this wall. Otherwise, the photo is added to the virtual photo wall and displayed to the user.

3.5.2 Circular Layout Algorithm

Our second algorithm aims to generate photo wall layouts that place photos more densely around the center, usually a focal point, of the wall. The idea is to place the images along a set of concentric circles, slightly perturbing the position of images if they overlap other previously placed images. The algorithm initially places a random photo in the center of the wall. It then iteratively applies the following two steps:

The first operation simply computes the smallest circle with center point at the center of the wall that strictly contains all previously placed photos. It then serves as a guideline to determine the display locations for the next batch of photos.

The second step places photos on the circle computed in the first step such that they don't overlap with any of the previously placed photos. To that end, the circle is split into n_i equidistant segments, where n_i is computed with the formula shown in equation 1. The subscript i indicates that this number corresponds to the i-th concentric circle. In equation 1 ceil is the ceiling function, r_i the radius of the i-th concentric circle, w the average width of the photographs, and h the average height.

$$n_i = \text{ceil}(\frac{2r_i}{w} + \frac{2r_i}{h}) \tag{1}$$

At each endpoint of these segments, the algorithm attempts to place a randomly chosen image in an iterative manner. If the image does not overlap any of the previously placed images at that position, it is placed there, and the algorithm proceeds. Otherwise, the algorithm greedily tries to determine the minimum amount the image has to be moved along a single axis to avoid overlap. If it succeeds, the image is placed at the adjusted position. Otherwise, it is rejected and no image is displayed at the corresponding point on the circle.

The iterative application of the two steps stops as soon as one of the two exit conditions is fulfilled. The first exit condition is fulfilled once the smallest enclosing circle of the placed photos intersects the boundary of the wall. In this case, the wall is considered to be full of photos. The second exit condition depends on an algorithm parameter. The layout generation is stopped once a maximal number of concentric circles is reached. This threshold was introduced to guarantee good performance in rooms with very large walls or rooms with a very high number of walls. We set the limit of concentric circles to 3.

3.6. Implementation Details

The entirety of our custom source code is implemented in C#. For details regarding the implementation please refer to the source code published on our GitHub repository: https://github.com/MixedRealityETHZ. The code can be found under Assets/Scripts. In the same folder, a README.md is provided that indicates the purposes of the various files.

4. Evaluation

In this section, we will talk about the user study that we conducted to evaluate our application. First, we will give an overview of the setup of the study and describe the questionnaire we handed to the users. Later, we analyze the results.

User Study Setup We evaluated our application with eleven ETH Zurich students. At the beginning of the study, the moderator explains the aim of the application and asks whether the user had previous experience with HoloLens2. If not, the moderator gives a quick introduction session showing the basic functionalities such as how to use the main menu, open and close an application, and click UI elements. After the introduction session, the moderator hands the HoloLens2 over to the user and explains the tasks the user needs to perform. The six tasks posed to the user are, in order, the following:

- Open the Virtual Photo Wall application from the main menu
- Scan the room by walking around
- Select the random layout generation algorithm
- Update the scene to display the photos
- Change the location of one of the pictures
- Change the size of one of the pictures

After fulfilling the tasks the user has the option to explore the application on their own. When the user finishes the session, the moderator hands out the user experiment questionnaire which we will describe in more detail in the next subsection.

Questionaire The questionnaire consists of fourteen questions of two different types. There are seven open-ended questions and seven rating scale questions. The seven rating scale questions are the subset of the short version of UEQ (User Experience Questionnaire) [12]. In the following, we will list all the questions and their results.

4.1. Rating-Scale Questions

The rating scale questions in UEQ aim to cover a comprehensive impression of the user experience. It measures classical usability aspects such as efficiency, perspicuity, dependability, and user experience aspects e.g. originality and stimulation. We used the subset of the short version of this questionnaire. Below, we list the pairs of contrasting attributes that the users came across. Instead of seven, we used five scales.

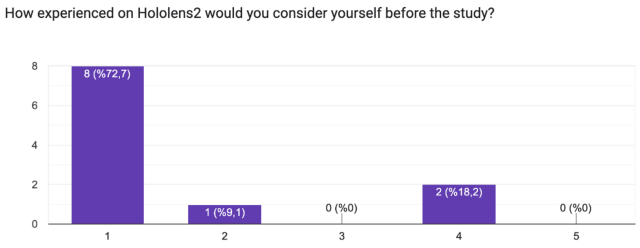
- complicated vs. easy
- inefficient vs. efficient
- confusing vs. clear
- boring vs. exciting
- not interesting vs. interesting
- conventional vs. inventive
- usual vs. leading edge

The first question in the questionnaire is about the experience level of the users using HoloLens2. The question has five scales. One means that the user has no experience and five means that the user has a lot of experience with HoloLens2 respectively. As displayed in Figure 3a, eight out of eleven participants were totally new to interacting with the HoloLens 2. One of the other participants had little experience and two of the participants had some experience rating four out of five.

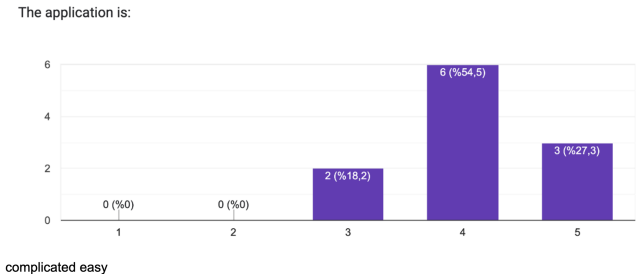
Next, we would like to discuss the usability of the application. Figure 3b shows the users' ratings about the complexity of the application. One represents that the application was complicated and five easy, respectively. Nearly all of the users, specifically nine of eleven users, rated at least 4 out of five. A similar result can be observed for the rating of the clarity of the application. As displayed in Figure 4a, ten out of eleven users thought the application was clear to them. All in all, for most of the users it was clear and easy to use the application.

Then, we asked about the efficiency of the application. Figure 3c exhibits the ratings of the users about the efficiency. Rating one means that the application was inefficient and five means it was very efficient. Observing Figure 3c, one can see that nine out of eleven users rated four and above meaning that the users did not have significant efficiency issues performing the tasks.

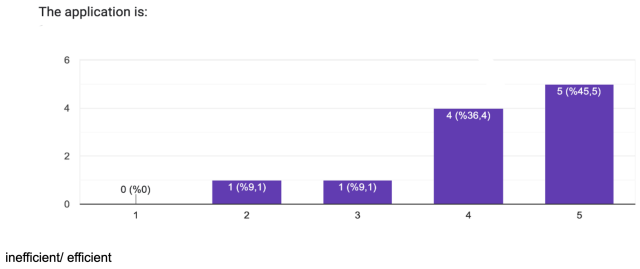
Lastly, we wanted to know how appealing the application was for the users. For that, we asked whether they found it interesting, exciting, and innovative. Observing the graphs 4c, 4d and 4e, one can conclude that the application was stimulating for most of the users overall.



(a) experience



(b) complicated vs. easy



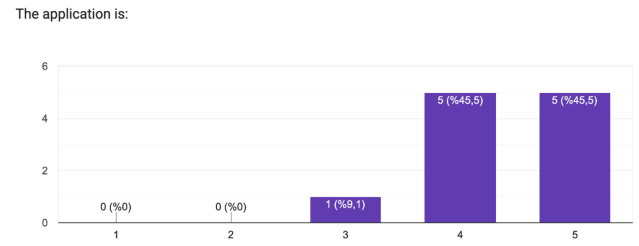
(c) inefficient vs. efficient

Figure 3. Results part one

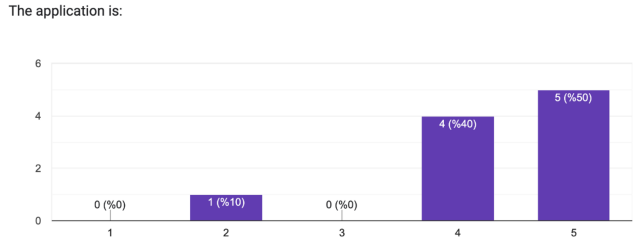
4.2. Open-Ended Questions

The first two open-ended questions were about whether the users could complete the tasks described by the moderator and the steps that were hard to complete. All of the users were able to complete all the tasks with minor help. Regarding the second question, most of the people expressed some frustration about the UI of the application. According to the answers, some of the users found it hard to click on the update button to scan the room. There were also some comments about the dropdown menu. For some users, when clicking on the dropdown menu to choose the algorithm type, the menu closed completely and it was hard to choose the algorithm.

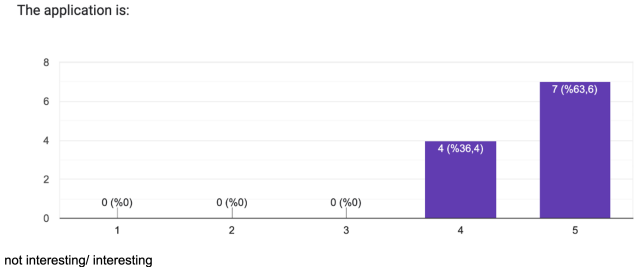
The next two open-ended questions were about the features they liked and disliked while using the application. Common frustration points were clicking the buttons and interacting with the photo objects. Users commented that they had a hard time resizing the photos from far away and they observed that there appears to be a small gap between the photos and the walls. The users liked the responsiveness of moving and resizing the photos, having different algorithms to display the photos, and the loading speed of the photos.



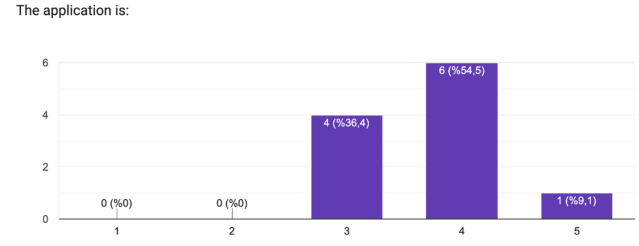
(a) confusing vs. clear



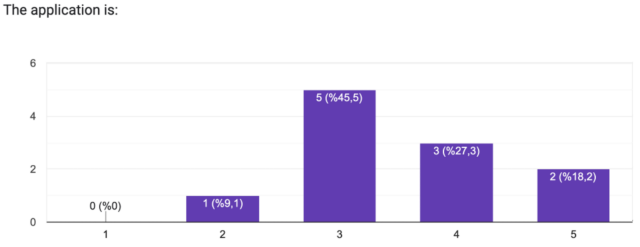
(b) boring vs. exciting



(c) not interesting vs. interesting



(d) conventional vs. inventive



(e) usual vs. leading edge

Figure 4. Results part two

For the last two questions, we asked the users whether an

essential feature was missing for them and what they would add if they had the chance. For the first question, some users would want to rotate the photos and some of them would like to change the displayed pictures. Furthermore, there were users who would like to add a button to shuffle the displayed photos. They also would like to display videos and web browsers next to the static photos. Another idea was to provide different shapes for the photo frames.

5. Discussion

5.1. Challenges

Throughout the development process, we faced a fair number of challenges that required various design decisions to address. We briefly discuss the major challenges here and the approaches we took to solve them.

Performance The main challenge we encountered was performance related. The HoloLens 2 has limited computational capabilities and several parts of our application are computationally heavy: The world mesh is scanned constantly to compute occlusion information. Images need to be loaded from the disk and the photo wall layouts need to be computed without freezing the UI for an optimal user experience. We addressed these issues with several design decisions. First, we moved the file reading operations to the start of the app as described in Section 3. Secondly, we decreased the observer radius to 3 meters so that the HoloLens will not run out of RAM in very big or very crowded rooms. This is needed because our occlusion feature is a costly operation. Lastly, we optimized the number of images on each wall because placing each image was taking some time, and as the number of images grow the placement time gets longer too. To ensure a reasonably fast user experience, we limited the number of images on each wall.

User Interface Another challenge we faced was the design of an intuitive and clean user interface. Due to the nature of our application, it is anticipated that the user roams freely around rooms in the real world. A UI with a static location in the world was therefore deemed unsuitable and we investigated UI concepts that move with the user. In the first iteration, we implemented a UI box that automatically followed the user around the room with a very slight delay. Parametrizing the behavior of the UI, however, turned out to be hard and we were not able to arrive at a satisfactory result. The UI was perceived as frustrating to interact with by the development team, as it moved away from the user when leaning forward to touch buttons. It also invaded the personal space of the user by following the user too closely in the real world. As a response to these observations, we switched to the final UI concept, which displays a UI box when the user looks at one of their palms and disappears again when the hand face is turned away from the gaze. This UI behavior has several advantages. First, the user is in control of the position of the UI. This makes interaction easier and more predictable. Furthermore, the UI is hidden from view when not needed, enhancing the immersion when utilizing the application.

5.2. Evaluation of the User Study

The user study indicates that our app provides an overall good user experience. Even users with very little to no prior experience with HoloLens 2 were able to navigate the UI of the app successfully. The smooth performance when interacting with the images was well-received.

The user study revealed two flaws. Some users had trouble interacting with the UI elements, such as the button and the dropdown menu. One part of the issue is the disappearance of the UI menu when clicking the dropdown menu. This is a bug in the application which requires further investigation as the cause could not be determined so far. The second flaw is a lack of advanced and extensive customization options. The users valued the given customization options but indicated the desire to have additional tools to interact with the photos.

5.3. Limitations

Limitations of the project do arise from various aspects. Performance limitations and their solutions were already described in 5.1. Moreover, the wall detection quality was also limited. Sometimes some wall-like surfaces such as doors, whiteboards, or closets could falsely be detected as walls. The detection capability increases with the scanning time but the perfect detection requires an infeasibly long scanning time or may even be impossible. This trade-off would even exist for a custom developed model. However, we believe that Microsoft Team has better data and insight for detection using HoloLens Sensors, so today what we use might be the cutting-edge technology. Finally, we can also deduce that using a HoloLens UI usually does not feel natural and is perceived as hard by looking at our user study. Three-dimensional buttons and menus do not feel natural and the user mostly can not get satisfactory feedback about the success of the user's clicking operation. Moreover, UI development is very hard for developers wishing to develop custom components. Improving the UI capabilities of the HoloLens is still an ongoing research topic for Microsoft but certainly, there is a long way to go before all UI limitations are solved.

5.4. Future Work

We see two main directions for future work.

Improved User Experience As indicated by participants of the user study, extending the customization options is a

widely requested feature. Adding additional tools and capabilities such as the ability to rotate photographs, replace photographs, add videos to the photo wall or the option to change the shape of the displayed photos could greatly enhance user experience.

Improved Scene Understanding As mentioned in the discussion of limitations, the quality of the wall detection is not optimal in many cases. Until more efficient computer vision algorithms are available, a post-processing pipeline might help to increase the quality of the detected walls. One option considered throughout this project was the internal LIMAP library, provided by Rémi Pautart and Shaohui Liu, which could be utilized to refine the boundaries of the walls. However, this approach was not further investigated as HoloLens 2 does not support native execution of the library at the moment.

6. Conclusion

In this paper, we present "All-Around Virtual Photo Walls", an application developed for the Microsoft HoloLens 2 that allows users to create virtual photo walls in their real-world surroundings. We conduct a user study to evaluate the performance and user experience of the application, and the results show that the application provides an overall positive user experience. Based on the findings of the study, we provide a list of further improvements that could enhance the overall user experience of the application, such as the ability to switch photos or display videos. We also want to note that our application also benefits greatly from any improvements in the scene understanding and the UI capabilities of HoloLens 2.

Overall, "All-Around Virtual Photo Walls" represents a promising new way for users to create and display photos in a mixed reality environment using the Microsoft HoloLens 2. The ability to create virtual photo walls that are flexible, sustainable, and customizable, as well as the positive user experience demonstrated in our study, make it a valuable tool for users who want to showcase their memories in a unique and innovative way.

References

[1] Microsoft hololens 2. https://www.microsoft.com/en-us/hololens/. Accessed: 2010-01-06.

[2] Microsoft hololens 2. https://www.printique.com/app/walleditor/room/ab7b2f6d-8028-4aea-a6b1-ab4cf68aa6e6/choose-arrangement/. Accessed: 2010-01-06.

[3] Microsoft hololens 2. https://www.contrado.com/blog/free-wall-art-visualizer-apps/. Accessed: 2010-01-06.

[4] Microsoft hololens documentation, hand menu examples. https://www.microsoft.com/de-ch/p/mrtk-examples-hub/9mv8c39l2sj4?rtc=1&activetab=pivot:overviewtab. Accessed: 2023-01-06.

[5] Microsoft hololens documentation, hand menu practices. https://learn.microsoft.com/en-us/windows/mixed-reality/design/hand-menu. Accessed: 2023-01-06.

[6] Microsoft Inc. Bounds control — mrtk2. https://learn.microsoft.com/en-us/windows/mixed-reality/mrtk-unity/mrtk2/features/ux-building-blocks/bounds-control?view=mrtkunity-2022-05. Accessed: 2023-01-06.

[7] Microsoft Inc. Mixed reality toolkit 2020 - 2.8.0. https://learn.microsoft.com/en-us/dotnet/api/microsoft.mixedreality.toolkit.windowssceneunderstanding.experimental.windowssceneunderstandingobserver?view=mixed-reality-toolkit-unity-2020-dotnet-2.8.0. Accessed: 2023-01-06.

[8] Microsoft Inc. Mrtk2. https://learn.microsoft.com/en-us/windows/mixed-reality/mrtk-unity/mrtk2/?view=mrtkunity-2022-05. Accessed: 2023-01-06.

[9] Microsoft Inc. Object manipulator — mrtk2. https://learn.microsoft.com/en-us/windows/mixed-reality/mrtk-unity/mrtk2/features/ux-building-blocks/object-manipulator?view=mrtkunity-2022-05. Accessed: 2023-01-06.

[10] Microsoft. Buttons. https://www.microsoft.com/de-ch/p/mrtk-examples-hub/9mv8c39l2sj4?rtc=1&activetab=pivot:overviewtab. Accessed: 2023-01-06.

[11] Unity Technologies. Unity webpage. https://unity.com/pages/unity-pro-buy-now?gclsrc=ds&gclsrc=ds. Accessed: 2023-01-06.

[12] Team UEQ. User experience questionnaire. https://www.ueq-online.org/. Accessed: 2023-01-06.

[13] Unity. Dropdown. https://docs.unity3d.com/Packages/com.unity.ugui@1.0/manual/script-Dropdown.html. Accessed: 2023-01-06.