Human-Machine Collaboration with the HEAP Walking Excavator using AR

Cafer Mertcan Akcay, Irem Kaftan, Adrian Taubner, Christopher Tibaldo
ETH Zurich
Ramistrasse 101, 8092 Zurich, Switzerland
{cakcay, ikaftan, ataubner, tibaldoc}@student.ethz.ch

Abstract

With the increasing demand and potential for automation in the construction industry over the last years, the Robotics System Lab (RSL) has developed an autonomous walking excavator (HEAP) as one way to fulfill this potential. The excavator needs to get instructions from a planner in order to operate in a seamless manner. Accordingly, this project aims to develop an augmented reality (AR) application which overlays the geolocated dataset of Irchel Park with its real world location and visualizes terrain changes using Microsoft HoloLens. In our project, we display the terrain by using the point cloud data obtained by a drone and pass it through Unity shaders to efficiently render the data. We then align the terrain data with the physical world by using QR codes and World Locking Tools. Furthermore, we implemented an editing option to modify the terrain at uniform angles and a hand menu for the user to choose between different editing modes, as well as to switch between the original and the modified terrain. In the end, we were able to satisfy the requirements of the project by integrating these features successfully. We see this project as a possible starting point for future work, such as by integrating it with the autonomous walking excavator and by trying it in different construction scenarios. The demo video can be found at Point Cloud Terrain Editor with HoloLens and the code is publicly available at Point Cloud Terrain Editor with HoloLens.

1. Motivation

While the Robotics System Lab (RSL) has been making progress in the development of the hardware and algorithms powering the autonomous walking excavator (HEAP), on the planners' side, progress needs to be made in the preparation and visualization of instructions for the machine.

For successful human-machine collaboration, planners and operators of the autonomous walking excavator need passive on-site control of the machine without having to sit in the cockpit. On the one hand, it should be possible to check planned terrain changes on site, and on the other hand, it should be possible to visualize the possible paths and any no-go zones. In many cases, this data, such as path finding, is calculated by the machine itself and, thus, it has no clear external graphical interface with which the planner can check the calculation for accuracy. Hence, there is a need for planning and visualizing changes on the terrain data before sending the instructions to the machine.

For this purpose, we implemented an AR application that can align a large georeferenced 3D dataset of a construction site with its corresponding location. The application allows the planner to move through the point cloud terrain and edit the terrain data to visualize changes on site. Particularly, we selected Irchel Park as our terrain and used its colored point cloud data obtained by a drone. We aligned the terrain data with the physical world using QR codes and World Locking Tools. We then implemented an option to modify the terrain and visualize the changes.

This AR application serves as an initial attempt to plan and visualize terrain changes on the construction site while being there physically. It can be used for different construction sites by the addition of new datasets. It might, in the future, also be integrated with the autonomous walking excavator in order to prepare instructions for it.

2. Prior Work

Human-machine collaboration (HMC) has become an essential part of manufacturing industry over the last years. Autonomous robots have been increasingly used in factories since they increase productivity and boost efficiency by automating monotonous tasks and supporting human operators in risky tasks [1]. In addition, various research groups have started to develop AR technologies to provide collaboration between humans and robots. For example, Hernandez *et al*. developed an AR interface in order to communicate high-level requests to the robot and to preview, modify, or approve the actions of the robot [5]. In their approach, a human operator manipulates virtual objects through the AR interface and determines what he/she wants from the robot [5]. As the virtual objects match with their real-life

counterparts, the robot understands the task and leverages its decision-making capabilities in order to achieve the desired goal. Another example comes from PTC's Reality Lab where they developed an AR interface to control robotic movements [3]. Particularly, a human operator uses kinetic AR to send a mobile robot from one station to another [3]. Their mobile robot uses kinetic AR and spatial mapping to plan its motion between these stations [3].

There are also many applications of AR technologies for assembly guidance which aim at enhancing the collaboration between operators and robots at industrial workplaces and decreasing the completion time of assembly tasks [9]. In these applications, the operator sees virtual objects that are superimposed on the real world scenes in the form of computer graphics and the robot helps the operator by loading the virtual objects [2], [8]. The task plan of the robot is provided to the operator in the form of graphical instructions using AR during the assembly task so that he/she has additional information for deciding his/her next assembly step [2], [8]. It is found that the completion times decrease and the assembling accuracies increase significantly using AR-assisted systems [9], [2], [8], [4].

Although AR technologies are commonly used in different industries, there are not many applications in the construction industry. The reasons include the compartmentalization in the construction industry and the lack of interoperability of software for different tasks [6]. Kjyanek *et al*. proposed an interactive timber prefabrication workflow in which the production and assembly tasks alternate between a construction worker and a robotic system [6]. The worker communicates with the robotic system through an AR interface which allows them to influence production sequencing and plan robotic trajectories [6]. Another example comes from Mitterberger *et al*. who developed a craft-specific AR system for in situ construction in order to show that an augmented manual process can match the complexity and precision seen in robotic fabrication [7]. This system is based on a real-time object-based visual-inertial tracking framework to achieve increased context-awareness and dynamic optical guidance for bricklayers [7]. It also allows bricklayers to have enhanced spatial freedom by capitalizing on their craft of mortar handling [7].

The AR applications in the construction industry mainly focus on the fabrication of materials with a few examples in automated bricklaying. Therefore, our project serves as a novel approach to plan and visualize terrain changes on a construction site.

3. Methodology

There were three main tasks in our project: The first consisted of loading the point cloud data into Unity and displaying it using shaders. The second required us to align the point cloud data with the physical world using QR codes and World Locking Tools. Lastly, we implemented terrain editing options along with a hand menu for user interaction. The system diagram of our project is shown in Fig. 1.

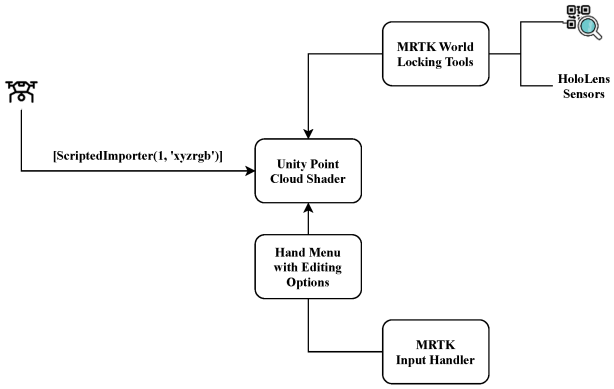


Figure 1. The system diagram of our project.

3.1. Visualizing the Terrain

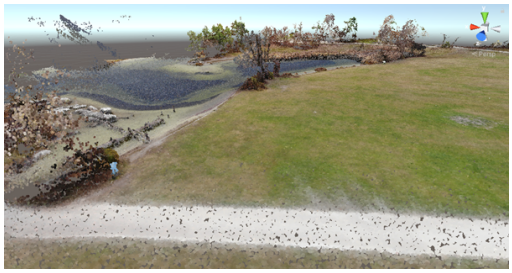
We had two options to display large terrains on the HoloLens: either displaying the raw point cloud data obtained by scanning the environment or using a large mesh which can be obtained by processing the captured data. We decided to use the former approach since it results in a better viewing experience on the limited viewport of the HoloLens. As the autonomous walking excavator can process both point cloud and mesh data, working with point cloud data does not cause a problem in our case.

We first worked with a coarse point cloud dataset which was acquired by the canton of Zurich with an airplane and a mounted LIDAR system on it. We then started working with a dense point cloud dataset which was acquired with a DJI Mavic Air and then processed in Agisoft Metashape using photogrammetry. The latter dataset of Irchel Park contained 2.3 million points with color. The data is visualized in Unity and shown in Fig. 2.

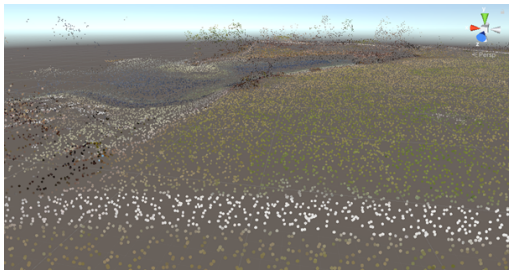
Figure 2. The dense colored point cloud data of Irchel Park.

As the data was hard to handle due to its size, we decided to subsample it and only use 23.000 points. Subsampling the data did not cause any problems in aligning the terrain

with the corresponding physical world location. The original and subsampled point cloud data are shown in Fig. 3.



(a) The original point cloud.



(b) The subsampled point cloud.

Figure 3. The original and subsampled point cloud of Irchel Park.

3.1.1 Loading Point Cloud Data in Unity

We decided to use Unity's `[ScriptedImporter(1, "xyzrgb")]` feature to streamline the process of loading point cloud data in Unity. In this way, any file with a `.xyzrgb` ending is recognized and constructed into a Unity Prefab object which can then easily be added to the scene. When the asset gets imported, our custom `OnImportAsset()` function defines the components to be added to the new GameObject. In our case, it is a C# script which loads the desired shader.

3.1.2 Using Shaders to Display Point Clouds

We decided to use Unity shaders to visualize the large amount of points in the scene. This allows us to send an array containing the coordinates and position of each point to the GPU and, thus, to offload work from the limited power of the CPU of HoloLens. The data transfer can be performed in Unity with the use of a `ComputeBuffer`. After passing the buffer to the GPU using Unity's `Graphics.DrawProcedural()` call, the shader takes over. We made use of a Vertex shader to place the point at the desired position and give it a radius while the Fragment shader took care of the color.

3.2. Aligning the Terrain with the Physical World

We decided to use World Locking Tools (WLT) provided by MRTK in order to align the point cloud data of Irchel Park with the corresponding real world location. When we start the application, we scan QR codes placed in the physical world whose precise locations in the scene are known. These serve as anchor points which WLT uses to perform terrain alignment. The reason for using multiple QR codes is to make the alignment more robust to rotation as well as to correct errors which accumulate due to inherent inaccuracies of the HoloLens tracking system. When the QR code is scanned successfully, a green box is displayed on it as shown in Fig. 4. The QR code placement in the real word is matched with the QR code placement in the scene. Therefore, the entire holograph space of our AR application is locked to the physical world.

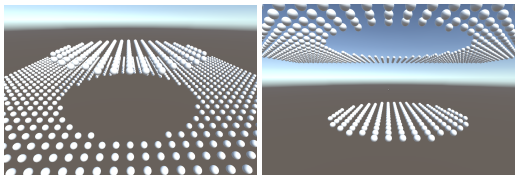
Figure 4. Successful QR code scanning.

3.3. Editing the Terrain

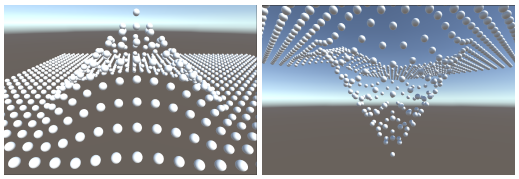
The terrain editing feature allows common operations that excavators can perform: digging the ground or creating a hill. These operations should be intuitive and not cumbersome for the user of the AR application. Therefore, we designed the interaction part such that when the user pinches the point cloud with their hand, the terrain within a specified radius is selected. If the user then moves their hand upwards, the terrain is lifted. Similarly, if the user moves their hand downwards, the terrain is dug.

The geometry of editing can be determined by the user. We implemented three options for the shape of the edit: creating a flat hill and a hole, a Gaussian-shaped hill and a hole, or a V-shaped hill and a hole, which are shown in Fig. 5. The user is able to select the radius of edit by a slider on the hand menu so that only the points within that radius are modified. There is also an option to select the σ of Gaussian by a slider on the hand menu to determine the width of the hill or the hole.

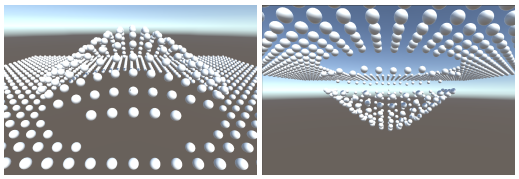
Based on the feedback of our supervisor, we modified the angle-editing option, such that the angle of repose of the hill or the hole can be determined with a slider. The user can select the angle and the radius of edit by two sliders on the hand menu. The neighbouring points of the pinched point are repositioned according to the angle and the radius of edit as shown in Fig. 6.



(a) Flat hill. (b) Flat hole.

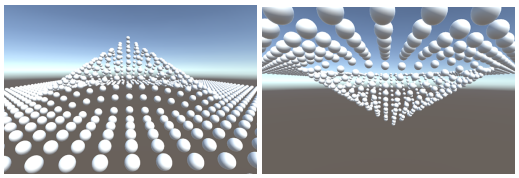


(c) V-shaped hill. (d) V-shaped hole.

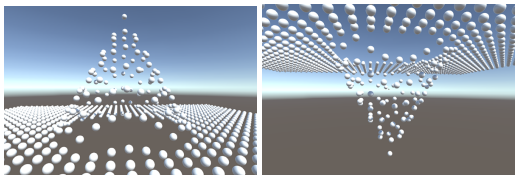


(e) Gaussian hill. (f) Gaussian hole.

Figure 5. The different terrain editing options displayed in Unity.



(a) Hill at 33°. (b) Hole at 33°.



(c) Hill at 60°. (d) Hole at 60°.

Figure 6. The edited terrain at different angles displayed in Unity.

To make a user-friendly application and enable user interaction, the settings can be selected by a hand menu which is shown in Fig. 7. The angle of repose is determined by using the angle slider on the menu. It can be between 0° and 90° depending on the particular application. The radius of edit is also determined by using the radius slider on the menu. Furthermore, the user can cancel the modifications or confirm them after visualization. It is also possible to switch between the original and modified terrain in order to compare the changes.

We used `InputHandler` of MRTK in order to handle input events resulting from user interaction. When the user fires an input event, such as pinching the point cloud, the input system recognizes it and determines which points in the point cloud are in the selected area. It then instantiates placeHolder GameObjects which aim to show the area which is currently being edited. These newly created GameObjects have white color during the editing process. When the user confirms the changes by pressing the confirm button on the hand menu, their positions get saved and they get the color of the terrain from where they are originated.

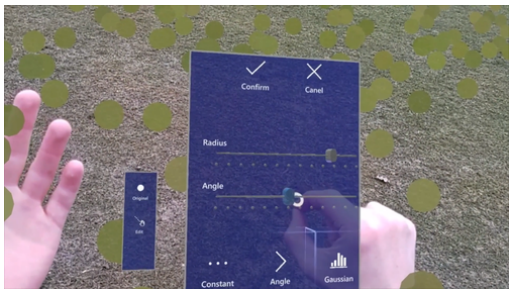
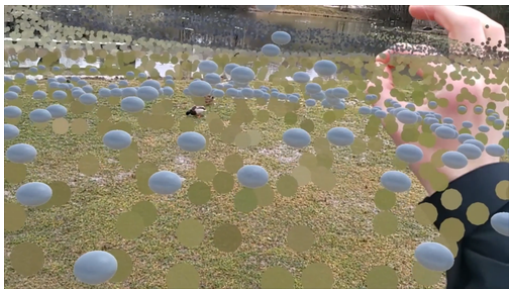


Figure 7. The hand menu.

4. Evaluation

The process of creating a hill is shown in Fig. 8 in which the hill is created in the editing mode (the point clouds are white) and then the changes are confirmed (the point clouds take the color of the grass).



(a) The created hill in the editing mode.

(b) The created hill after the changes are confirmed.

Figure 8. The creation of a hill using HoloLens.

Overall, we were able to satisfy the requirements of our project which were aligning the terrain data with its physical world location and editing the terrain by using the angle of repose. The process of aligning the terrain with its physical

world location depends on the placement of the QR codes so the alignment may be faulty if the QR codes are not placed correctly.

We also realized that the application is very sensitive to illumination changes so its performance differs based on the amount of sunlight. If there is a lot of sunlight, it becomes harder to see the points. On the other hand, if there is not enough sunlight, it becomes harder for the user to scan the QR codes and for HoloLens to identify gestures. Moreover, the initially used sliders were not perceiving hand inputs all the time, so we changed them to a newer version for the final submission. Although this change did not solve the problem completely, it improved the user interaction quality significantly.

5. Discussion

The presented work here is intended as a starting point for future projects. For instance, the AR application can be integrated with the HEAP autonomous walking excavator in order to direct and be aware of the actions of the machine more accurately by defining new excavation sites on operation and being warned of its intended trajectories. It can also be used to plan and visualize terrain changes on site so it can help planners and operators save time and effort.

There are possible improvements to be made on the displayed terrain. On shader level, one can include functions, such as frustum culling, or level of detail functions which decimate points that are far away. One can also use mixed point cloud-mesh data depending on the distance to the user in order to better visualize the terrain. Particularly, the regions that are close to the user can be displayed using mesh data and the regions that are far away from the user can be displayed using point cloud data. Moreover, one can display half-transparent mesh in addition to control points and use mesh decimation based on distance in order to show closer points with higher resolution.

There are also possible next steps to be implemented. In addition to cancelling or confirming changes, one can add an undo or redo option so that the user can switch between the changes quickly. One can also have an export file in two ways: either a `.txt` file with the point cloud data which is less elegant but more robust to process or a list of transformations with the anchor points and slope angles at those points which is lighter and easier to process. Moreover, one can implement a hand shader to mask the points behind the hand in order to get a better depth perception.

References

[1] Augmented reality could transform human-robot collaboration. [Online] Available at https://innovate.ieee.org/innovation-spotlight/human-robot-collaboration/, 2020. 1

[2] Siam Charoenseang and Tarinee Tonggoed. Human-robot collaboration with augmented reality. In Constantine Stephanidis, editor, *HCI International 2011*, pages 93–97, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg. 2

[3] Anna Fuste, Benjamin Reynolds, James Hobin, Andrea Braga, and Valentin Heun. Kinetic ar: Robotic motion planning and programming using augmented reality interfaces. New York, NY, USA, 2020. Association for Computing Machinery. 2

[4] Christos Gkournelos, Panagiotis Karagiannis, Niki Kousi, George Michalos, Spyridon Koukas, and Sotiris Makris. Application of wearable devices for supporting operators in human-robot cooperative assembly tasks. *Procedia CIRP*, 76:177–182, 2018. 7th CIRP Conference on Assembly Technologies and Systems (CATS 2018). 2

[5] Juan David Hernández, Shlok Sobti, Anthony Sciola, Mark Moll, and Lydia E. Kavraki. Increasing robot autonomy via motion planning and an augmented reality interface. *IEEE Robotics and Automation Letters*, 5(2):1017–1023, 2020. 1

[6] Ondrej Kyjanek, Bahar Al Bahar, Lauren Vasey, Benedikt Wannemacher, and Achim Menges. Implementation of an augmented reality ar workflow for human robot collaboration in timber prefabrication. 06 2019. 2

[7] Daniela Mitterberger, Kathrin Dörfler, Timothy Sandy, Foteini Salveridou, Marco Hutter, Fabio Gramazio, and Matthias Kohler. Augmented bricklaying: Human–machine interaction for in situ assembly of complex brickwork using object-aware augmented reality. *Construction Robotics*, 4, 12 2020. 2

[8] N. Pathomaree and S. Charoenseang. Augmented reality for skill transfer in assembly task. In *IEEE International Workshop on Robot and Human Interactive Communication*, pages 500–504, 2005. 2

[9] Miaolong Yuan, S K Ong, and Andrew Nee. Augmented reality for assembly guidance using a virtual interactive tool. *International Journal of Production Research*, 46:1745–1767, 04 2008. 2