Reducing Latency in a Collaborative Augmented Reality Service

Wennan He
u5833304@anu.edu.au
Australian National University
Canberra, ACT

Ben Swift
ben.swift@anu.edu.au
Australian National University
Canberra, ACT

Henry Gardner
Henry.Gardner@anu.edu.au
Australian National University
Canberra, ACT

Mingze Xi
Mingze.Xi@csiro.au
Commonwealth Scientific and
Industrial Research Organisation
Canberra, ACT

Matt Adcock
Matt.Adcock@csiro.au
Commonwealth Scientific and
Industrial Research Organisation
Canberra, ACT

ABSTRACT

We show how inter-device location tracking latency can be reduced in an Augmented Reality (AR) service that uses Microsoft's HoloLens (HL) devices for multi-user collaboration. Specifically, we have built a collaborative AR system for a research greenhouse that allows multiple users to be able to work collaboratively to process and record information about individual plants in the greenhouse. In this system, we have combined the HL "world tracking" functionality together with marker-based tracking to develop a one-for-all-shared-experience (OFALL-SE) dynamic object localization service. We compare this OFALL-SE service with the traditional Local Anchor Transfer (LAT) method for managing shared experiences and show that latency of data transmission throughout the server and users can be dramatically reduced. Our results indicate that OFALL-SE can support near-real-time collaboration when sharing the physical locations of the plants among users in a greenhouse.

CCS CONCEPTS

• **Human-centered computing** → **Mixed / augmented reality**; **User interface toolkits**; **Interface design prototyping**.

KEYWORDS

Augmented Reality, collaborative, latency, interface

ACM Reference Format:
Wennan He, Ben Swift, Henry Gardner, Mingze Xi, and Matt Adcock. 2019. Reducing Latency in a Collaborative Augmented Reality Service. In *The 17th International Conference on Virtual-Reality Continuum and its Applications in Industry (VRCAI '19), November 14–16, 2019, Brisbane, QLD, Australia.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3359997.3365699

1 INTRODUCTION

Augmented Reality (AR) tools integrate digital holograms into a user's physical environment [Azuma 1997] and, when combined with real-time rendering, can allow users to interact meaningfully with both the physical environment and the computed holograms. One key component of an AR system is the tracking mechanism, which allows it to place virtual objects in the real word. Fiducial marker-based tracking is a popular way to anchor the holograms, and is supported by systems such as the ARToolKit [Kato et al. 1999], ARTag [Fiala 2005] and Vuforia [Simonetti Ibañez and Paredes Figueras 2013]. AR systems using marker-based tracking have been used in medicine [Kamphuis et al. 2014], training [Lee 2012; Webel et al. 2013] and communication [Zhou et al. 2015]. Consumer AR applications (e.g. those using Apple's ARKit [Permozer and Orehovački 2019] or Google's ARCore [Alkandari et al. 2019]) have also become increasingly popular in recent years.

In general, marker-based tracking supports single-person use. The inability of such systems to understanding the physical environment greatly limits their support for multi-user applications. By contrast, the Microsoft HoloLens (HL) headset [Evans et al. 2017] does support multi-user shared AR experiences using "world-tracking" technology in conjunction with the spatial mapping generated by each individual HL device [Turner et al. 2019]. This spatial mapping technology creates 3D reconstructions of the physical space that enables placement, occlusion, physics and navigation within AR environments. With the data generated from the spatial mapping, the shared coordinates can be used to support collaboration [Bray et al. 2018] and several users can share information in the same physical environment when using multiple HL devices. However, this can introduce significant temporal latency, causing AR overlays to become out of sync between users while the spatial maps are synchronized between devices.

In this paper, fiducial marker-based tracking is combined with HL world tracking to provide a low-latency collaborative AR environment. These two technologies are combined together to create a new method of distributed dynamic object localization. The application context we consider is that of multiple technicians working in a research greenhouse, where the technicians must interact with (feed, water, and record information for) many different plants, grouped into plant trays which are themselves mobile. We will show how the latency of sharing data (e.g object localization with moving objects) can be reduced using our new method. This collaborative greenhouse scenario is depicted in Figure 1.

This paper is structured as follows. Section 2 presents related work on shared experiences in mixed reality services. Section 3 describes the collaborative scenario for which our system has been

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
VRCAI '19, November 14–16, 2019, Brisbane, QLD, Australia
© 2019 Association for Computing Machinery.
ACM ISBN 978-1-4503-7002-8/19/11…$15.00
https://doi.org/10.1145/3359997.3365699

RIGHTSLINK

developed and discusses the marker-based and world tracking technologies used by the HL. Section 4 describes the tools used and their roles in this collaborative system. Section 5 describes the architecture of our system and the functionality that it provides. Section 6 describes how our system uses shared World Anchors to achieve the shared experience of a collaborative AR while reducing latency. Sections 7 and 8 describe and discuss an experiment that has been undertaken to evaluate system latency and to compare it with a the traditional HL collaborative architecture. Finally, Section 10 concludes the whole paper and discusses future directions for this work.

2 RELATED WORK

In this section, we describe existing documented methods for developing multi-user collaborative HoloLens applications using Microsoft's Mixed Reality Toolkit (MRTK) and Unity 3D. We also describe recent work on shared collaborative AR experiences.

2.1 Developer Documentation for Hololens

Microsoft provides a user manual for the MRTK. This user manual gives examples of how to build a system for shared mixed-reality experiences with multiple HL users [Zeller et al. 2019a].

When using the MRTK to build "Shared experiences in Unity" [Zeller et al. 2019b], the manual suggests two different alternatives for developers. One is to use *Azure Spatial Anchors* (ASA) [Arguelles et al. 2019] and the other is the *Local Anchor Transfer* method (LAT) [Turner et al. 2018]. ASAs offer developers three main features: multi-user experiences, way-finding and persisting virtual content in the real-world [Arguelles et al. 2019]. These features provide developers the tools to design and build a multi-user AR system, however developers are required to use an external Azure cloud service to provide the ASAs. This may not be possible (if the environment is air-gapped from the wider internet) or desirable (due to regulations around data storage and provenance). The second alternative is the LAT technique, where each HL can export a World Anchor which can then be imported by another HL device, without the need to communicate with the external Azure service [Turner et al. 2018]. The benefits of the LAT method are that it is less complicated than using ASAs, and that it can be built as a standalone C# application in Unity. However, the drawbacks are that LAT does not support iOS and Android devices [Zeller et al. 2019b] and that it provides "less robust" anchor recall than ASA [Turner et al. 2018].

We also need to emphasize that our "one-for-all-shared-experience" (OFALL-SE), method described below in Section 6, is modified from Microsoft's OFALL method [Zeller et al. 2019c]. The Microsoft documentation does not describe whether and how OFALL can be used for multiple-user scenarios (shared experiences) apart from stating that it could be used with the latest version of ASA. Thus, it is worth investigating how OFALL can be adapted for shared experiences and whether it can be also integrated with LAT. Although in their document, they suggested using ASA rather than LAT and also emphasized that LAT transfers the anchors by using device-to-device (without internet) process [Turner et al. 2018], we also developed a customized cloud server to achieve the anchor sharing process from internet by using LAT and this could also tackle the problem that the developers may concern about the privacy issues while sending the spatial anchors to MS' server by using ASA. We will return to discuss the technical details of OFALL-SE and of World Anchors in more detail in Section 6.

2.2 Collaboration for Socialization

Zhang et al. present a prototype for collaboration in augmented reality that they call "Collaborative Augmented Reality for Socialization (CARS) [Zhang et al. 2018]". CARS was developed to improve the user-perceived Quality of Experience (QoE) for AR while sharing the overlapped virtual holograms with object recognition technology through multiple users' smart phones. They also emphasized the need to reduce the end-to-end latency on network transmission and recognition time in their system in order to improve user experience. In their paper, Zhang et al. observed that "CARS does not require localization" and note that "localization alone is not enough for AR. Even if we know which painting a user is looking at via localization, it cannot tell us the position of that painting within a camera view for augmentation." [Zhang et al. 2018]. However, even if localization is not sufficient for a quality AR experience, the quality of localization itself is greatly improved using AR technologies such as the HL. Indeed, the results of the present paper will show that HL technology can be generally used to determine an accurate localization and that this localization can be shared with multiple users with low latency.

3 COLLABORATIVE SCENARIO

Greenhouses are used extensively in scientific research and agricultural trials. In modern greenhouses, trays of plant seedlings are subjected to varying conditions (plant type, nutrition, watering, temperature, etc.) and plant characteristics are measured over time. A picture of a typical seedling tray is shown in Figure 2.

The collaborative scenario we consider in this paper is one with a group of technicians working in a greenhouse which contains hundreds of seedling trays. Each tray needs to be labelled with the characteristics of each seedling in that tray, and each tray will need to be regularly moved from its growing site to other sites in the greenhouse in order to take measurements, for watering, etc. In such a workflow, the plant trays will be passed from one technician to another, and it may or may not be important that a plant tray is eventually returned to its original growing location.

Clearly, labelling the plant trays using tags such as a QR code makes them amenable for interactive tracking and data analysis using computer systems. Conventionally, this has been achieved using mobile devices. Using some extant systems [Wagner et al. 2005] a technician is able to scan a tray's label and read or enter data for the plants within the tray. The technician is then able to update this data once the plant tray has been moved to a new location. Such a workflow involves frequent stopping and use of mobile devices, which usually involve both hands to operate. An objective of our system was to explore whether an augmented reality headset could allow the continuous update of plant tray information and location while a technician is using both of their hands to lift, transport and otherwise process the plant tray. In our system, AR holograms show data menus associated with each slot in a plant tray as in Figure 3.

RIGHTSLINK

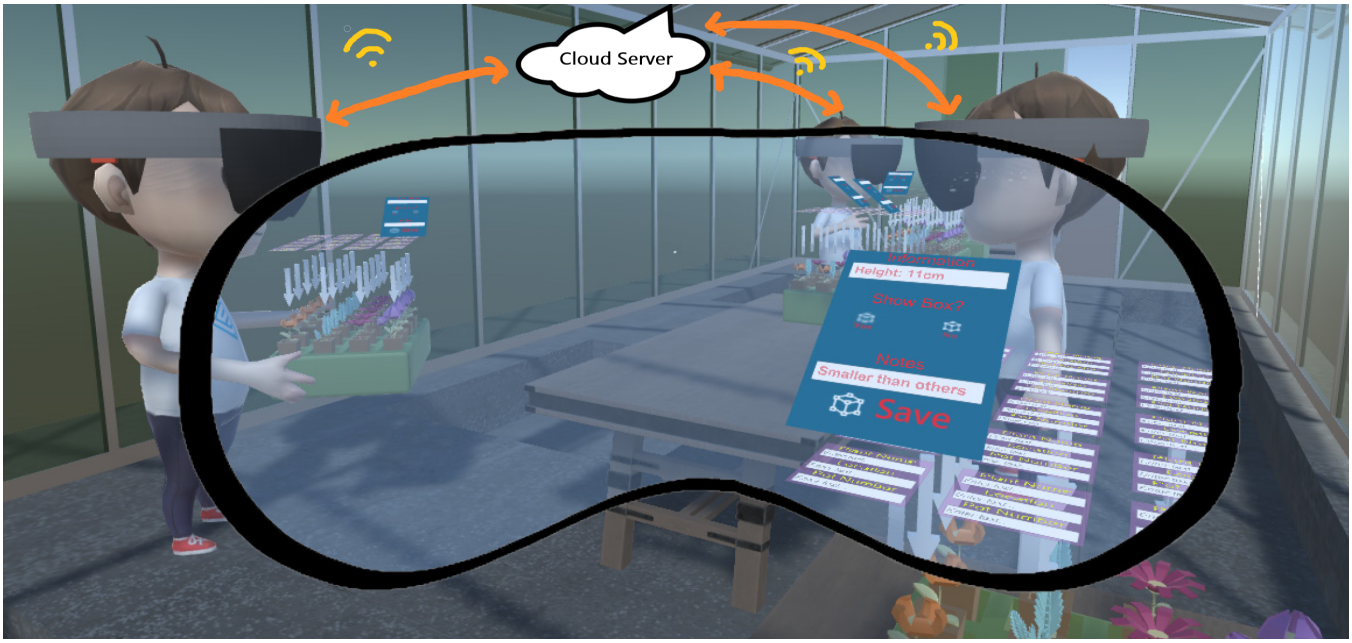

Figure 1: A collaborative scenario showing technicians wearing HoloLens devices interacting with plants in an example greenhouse. Plant data (location and other information) is always shared and updated among all users in real-time via a cloud server (which also provides persistent storage).

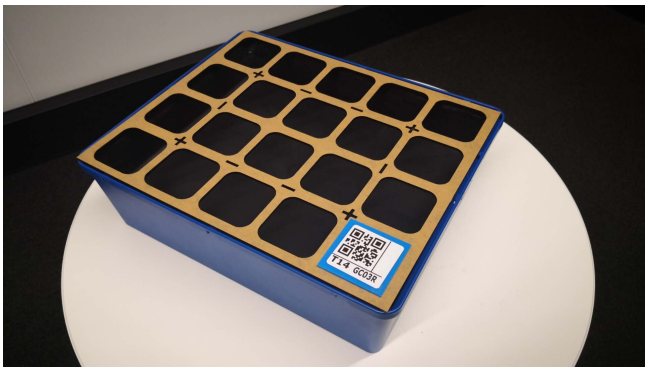

Figure 2: A plant tray containing 19 slots for growing plant seedlings and one slot with a QR code for identifying this particular tray.

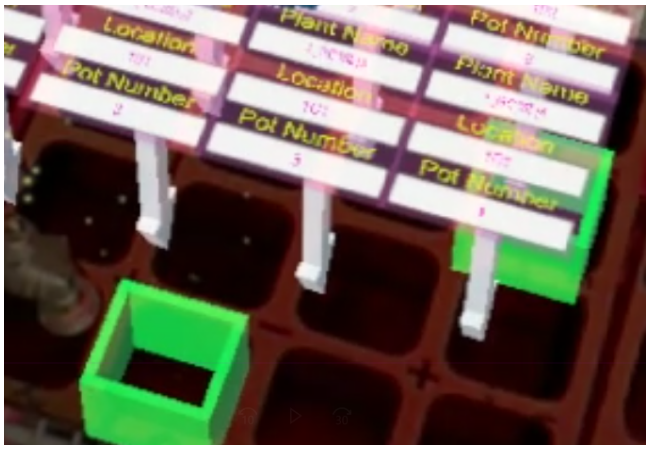

Figure 3: View of a plant seedling tray through an AR headset showing data menus for the individual slots in that tray. In the hologram, arrows indicate which information belongs to which slot in the plant tray.

4 SYSTEM TECHNOLOGIES

Our system is based on the following technologies:

- Head-mounted display (HMD): the Microsoft HoloLens (HL) (1st gen) [Evans et al. 2017] is an AR headset that provides stand-alone computation to enable users to interact with holograms overlaid on their view of the world.
- Software rendering: Unity 3D is a game engine developed by Unity Technologies that is widely used in AR development. In this project we used Unity version `2018.3.5`.
- Marker-based tracking: Vuforia [Simonetti Ibañez and Paredes Figueras 2013] is an AR platform that plugs into Unity to provide computer vision functionality such as object and QR code recognition.
- User interaction: The Mixed Reality Toolkit (Version 2 [Microsoft 2019]) is provided by Microsoft for developers to build Hololens applications within Unity.
- Data transmission and storage: a CentOS 7 cloud server (1vCPU, 1024MB RAM, 25GB SSD) hosted the infrastructure for data storage and sharing data between HL devices.

RIGHTSLINK

The server application uses Python (application code) and MySQL (relational database). The data stored by the server included the physical location of each plant tray in the greenhouse, as well as the plant information data (PID)—typically plant height and seed data—for each plant in every tray. Communication and data exchange between the server and HL devices was done over TCP using a JSON data format.

5 SOFTWARE ARCHITECTURE

Based on the needs of the greenhouse users in their collaborative workflow, we designed an interface to allow technicians to record the plant information data such as seeds, height etc. This interface contains two parts:

(1) a Plant Information Panel (PIP) showing detailed information about a specific plant and providing an interface for updating this plant data (see Figure 4)
(2) a Plant Name Panel (PNP) for showing an overview of all plants in the tray

The PNP allows users to identify each plant in a given plant tray; the display of all PNPs for a given tray can be seen in Figure 3—we call this a PNP set. The *Location* label indicates the coordinate location of each plant inside the plant tray, while the *Pot Number* label indicates the index (from 1–19) of the pot inside the plant tray.

At startup, the system will display the PNP set (as shown in Figure 3) for each plant tray, while each PIP is deactivated (invisible) until a technician clicks the corresponding PNP. In other words, the overview information is shown by default, while the detail information for a given plant is hidden until requested.

In a multi-user scenario, plant trays may be moved outside the field-of-view of a given user. As the technicians move around the greenhouse, if the HL successfully tracks a new QR code (with the Vuforia Engine), the technician may click a virtual button above this QR code and the system will automatically generate a PNP set above this new plant tray. If this tray already has a PNP set displaying somewhere else (e.g. from the plant tray's previous location, as moved by another technician) the system will move that PNP set to new location of this QR code. If the technician moves the plant tray to a new location in the greenhouse, the system will similarly update the location of the PNP set display. In addition, when a technician clicks the virtual button above a plant tray, the system will automatically upload the latest location of this plant tray to the database, and immediately update this information in the HL display of any other technicians in the greenhouse.

When a technician launches the system by turning on the HL and loading up the application, the application will first synchronize the current information of all plants in the database, and update their location and information accordingly. After this initial data synchronization step, the system will present a Plant Assistant Panel (PAP) which has three buttons:

- *Synchronize*: synchronize the plant data (in case the technician has network connectivity issues and failed the auto-synchronization step, or requires a manual re-sync for other reasons)
- *Main Tools*: open up a context-sensitive tools menu (used in our case for running latency experiments as described in Section 8)

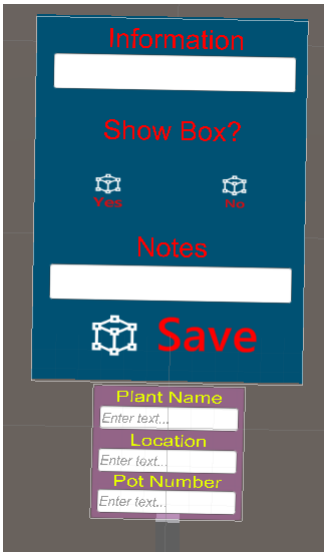

Figure 4: The plant interface designed in Unity for users to record the data.

- *View Map*: display the status map of all the PNP sets in the greenhouse (this status map is shown in Figure 5)

The most important part of the status map is the navigation function. This feature allows the HL display to guide the technician (by displaying a holographic direction arrow) to a particular plant or plant tray. This arrow is dynamic, and even as the technician moves will always point in the direction of the plant (as seen in Figure 5). This is especially useful in large greenhouses with hundreds of plant trays.

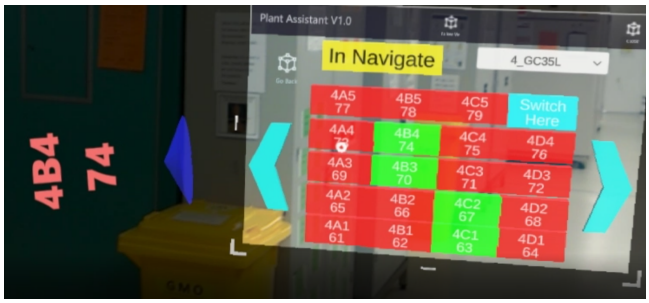

Figure 5: The navigation view—note the blue arrow to the left of the display, which indicates the direction of the particular plant being searched for.

6 WORLD ANCHORS AND SYSTEM LATENCY

The core goal in designing in this system is to achieve multi-user collaboration with low latency synchronization for greenhouse workers to share plant data. There are two types of data that are needed to be shared among the greenhouse workers:

- The physical locations of the plant trays in greenhouse.

RIGHTSLINK

- The plant information data (PID) that is input and edited by users. The exact size and format of the PID will vary depending on the plant, then experiment being run and the needs of the greenhouse technicians. It will typically include plant height, seed number, and relevant planting and harvest dates.

The PID is plain text, designed primarily for humans to read and write, and is relatively straightforward to store and share between HL devices. The more challenging part of the data sharing problem is the task of sharing the physical locations of the plant trays.

In order to share real physical locations among different HL devices, the MRTK provides a method to acquire physical locations using a "spatial map" to track the world (also known as a "World Tracking" method). The basic principle of this method is to overlap views of objects in the real world with a collection of virtual geometry surfaces. HL devices are then able to recognize the real world through this World Tracking method and transmit (share) the physical locations to each other.[Turner et al. 2019]

In Unity 3D, the location of a game object in a virtual scene is represented as a 3D vector relative to the scene's co-ordinate system. However, each HL device has a different co-ordinate system (initialised at start-up). As a result, physical location shared between HL devices cannot be represented by a simple 3D vector, but as a data structure known as a *World Anchor* (WA) [Vassallo et al. 2017]. WAs are necessary for two reasons in building a collaborative HL system:

- They can be used to persist the physical locations of holograms in HL devices. This indicates that the same WA components could be shared among users' HL devices and ensure that all users could see the same holograms in the same physical locations while they are working together in the same working space.
- They can stabilize the holograms in HL devices. The locations of holograms in HL device would be changed if the HL device loses tracking in the world (for example if the brightness of the space is too low the HL may temporarily lose tracking). Also, if the user gets too far away from the holograms, they will also become unstable (and start "shaking") in the HL display.

In order to use Unity World Anchors in a collaborative HL application, WA data needs to be serialized and transmitted to a server and then downloaded and de-serialized at a later time. This can give rise to considerable latency in a collaborative application where a group of HLs are attempting to synchronize their data. The size of the WA is dependent on the complexity of the physical environment. MS have not published an official specification for the size of the WA, but our testing in the greenhouse environment shows that each WA object is approximately 10MB (uncompressed) in memory.

We now consider two alternative designs that demonstrate this latency problem.

6.1 One-for-one Locating

This one-for-one location-sharing technique is so-called because it requires one World Anchor (WA) object per tracked object (i.e. per plant tray). This one-for-one method is suggested by Unity Official Document [UnityTechnologies 2019]. In our greenhouse scenario, this one-for-one localization approach is shown in Figure 6, which shows 5 plant trays, each with a corresponding WA attached. The physical locations of these 5 plant trays would be determined by these 5 corresponding WAs, and so each WA must be sent to the server and communicated to any other HL devices. The synchronization algorithm pseudo code is shown in Algorithm 1. Note that any of the five plant trays may be moved at any time by the greenhouse technicians, triggering a full re-synchronization.

ALGORITHM 1: One-for-one Locating with Local Anchor Transfer

```
Users: Synchronize All WAs and PIDs;
while true do
    forall Users u in UserList do
        if u Moves(PlantTray) then
            UploadToServer(WA);
        end
        if u Modify(PIDs) then
            UploadToServer(PIDs);
        end
        ServerNotify(u_new in UserList.delete(u));
    end
end
```

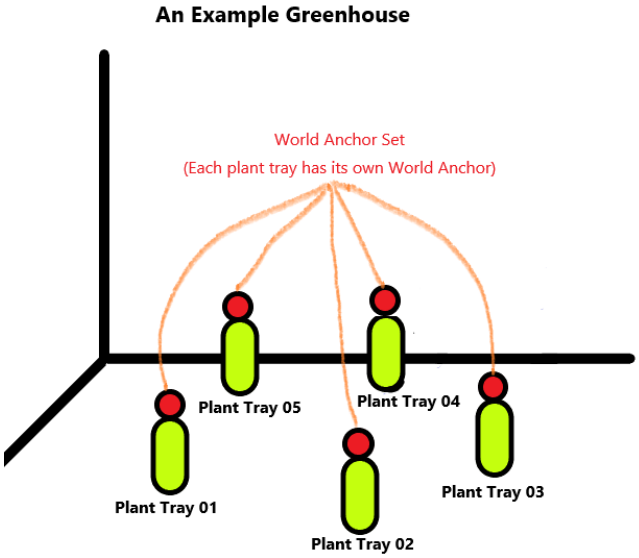

Figure 6: The relationship between the plant trays and the attached WAs (red circles) in a sample greenhouse with the one-for-one method. Note that there are 5 plant trays in this example.

In the greenhouse workflow, two types of data events may be triggered by users. One is plant data generation and access, and the other is the synchronizing of location data. When a user first launches the system, the HL device is unaware of any plant trays, and a synchronization request is needed to synchronize all the stored WAs (physical locations of the plant trays) and the PIDs.

RIGHTSLINK

Once the system is running, any movement of any of the plant trays requires generating and storing this updated location to the database, and synchronizing this new location (via the new WA) among all other HL devices.

Because of the size of the WA data object, the data transmission latency using this one-for-one approach is significant (as we will show in Section 7). The plant information data also needs to be transmitted among users, however this data is much smaller (2̃00 bytes). Thus the major issue which will affect the latency on this collaborative system is caused by the WA operations which include:

- Serializing the WA (with `WATB.ExportAsync()` in Unity) de-serializing (with `WATB.ImportAsync()` in Unity).
- Transmitting (over wifi) the serialized WA from user to server and then on to all other users.

6.2 One-for-all (OFALL) Locating

ALGORITHM 2: OFALL Locating

```
Administrator: Setup one WA on server;
Users: Synchronize One WAs and All PIDs;
while true do
    forall Users u in UserList do
        if u Moves(PlantTray) then
            UploadToServer(PlantLocationData);
        end
        if u Modify(PIDs) then
            UploadToServer(PIDs);
        end
        ServerNotify(u_new in UserList.delete(u));
    end
end
```

In contrast, the modified method that we use in our system is called "one-for-all-shared-experiences" (OFALL-SE) locating, in which one "reference" WA is used per device, and the location of each plant tray is calculated relative to the reference WA. Instead of using a WA to represent the physical locations of each plant tray, the OFALL-SE method uses only a lightweight (x, y, z) 3-vector in each device's scene coordinate system, combined with an offset to the reference WA.

Storing one WA is still necessary because HL uses its built-in spatial map to locate the holograms in the physical world. If the system stores *only* the device's scene co-ordinate, all the holograms would be offset when HL loses its world tracking. Storing an offset to the WA means that even if a given HL device loses its world tracking, it will recover all the holograms once it is able to track the world and recognize the physical space again. Therefore, this new method combines Unity's orientation system with WA together to allow the system to send only the bytes of the 3D vectors to other users whenever a plant is moved, so that the other devices can update the locations of their holograms of plant trays. This new method is represented graphically in Figure 7.

The reference WA object shown in Figure 7 is the core component in this new method: it is used to determine the locations of all the other holograms in other HL devices. In this new method, the WA

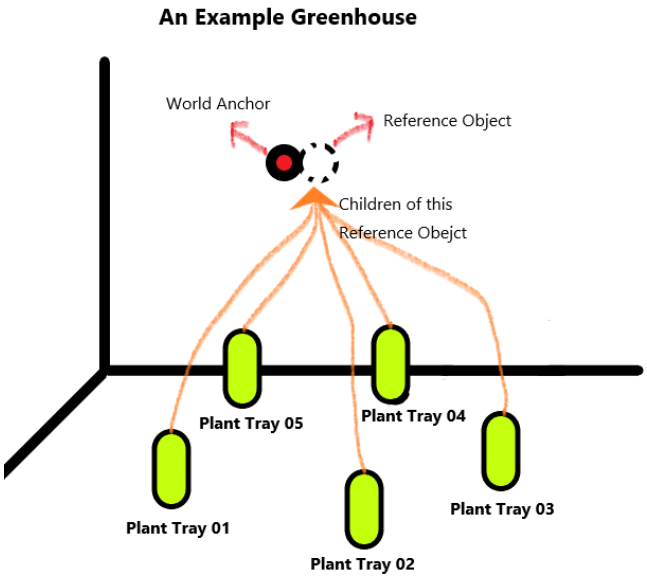

Figure 7: The diagram of the relationship between the plant trays and the reference WA object (red circle) in a sample greenhouse with the OFALL method. Note that there are 5 plant trays in this example.

only needs to set once and could be managed by an administrator; users will not need to interact with the WA to update the plant location data. This new procedure is shown in Algorithm 2.

Each WA would be sufficient for all users working in the same greenhouse. If multiple greenhouses are required, then the administrator could generate multiple WAs for these greenhouses and users could synchronize the corresponding WA in different greenhouse as required.

One key difference between Algorithm 1 and Algorithm 2 is that the expensive WA serialization and transfer operations are no longer inside the loop. Instead, the system will serialize only the 3D vector offsets of the plant trays and to be sent to the server and other HL devices. The major benefit of this new method is that the size of the location data of plant trays under this method is very small—around 200 bytes, much smaller than the 1̃0MB WA.

To summarize the two different collaborative location-sharing approaches:

- One-for-one: plant tray hologram requires a separate WA to locate into the physical world. Transmitting the WAs among users has higher latency due to the size of WA.
- One-for-all-shared-experiences: one reference WA is needed to determine all the physical locations of the holograms of plant trays. Representing the location of holograms of plant trays is achieved using the offset position in the HL device's scene space.

7 EXPERIMENT

In this section, we report the results of an experiment to empirically investigate the synchronization latency for different numbers of plant trays between the one-for-one and one-for-all methods.

RIGHTSLINK

This experiment was done in a research greenhouse at the Australian National University (ANU) Plant Phenomics Facility.

The experimental system was as described in Section 4. For different numbers of plant trays, the time-to-completion of synchronizing the PNP set to greenhouse (the synchronization latency) was be measured by the HL software and stored in the database.

For this test, we prepared 8 WAs (corresponding to 8 plant trays in the greenhouse) which the HL would import in order to locate 8 PNP sets. In this experiment, the HL did not need to download the initial WAs from server (this data was cached locally) so as to only measure the time to synchronize the locations between HL devices.

In this experiment, we investigated the latency cost during the location of the 8 PNP sets corresponding to 8 plant trays in greenhouse. The experiment involved completing a specific scenario—locating all the PNP sets in the HL device—with both one-for-one and one-for-all methods and between 1 and 8 plant sets. In each scenario, the full procedure was repeated 10 times, with the total time measured in each case. All latency timings were measured automatically and uploaded to the database once the locating process had been completed.

A sample procedures of measuring the latency with the one-for-one method is:

(1) Set the number of PNP sets which are required to be located in the greenhouse throughout HL device (From 1 to 8).
(2) Press the button "Sync with Old system".
(3) Wait for the process until all the PNP sets have been successfully located and the colour of the indicator cube would be turned into green.
(4) The latency time would be shown in the content "Time Used Status", and also stored into the database.
(5) Clear all the existing PNP sets which had been located by clicking the button "Reset".
(6) Repeat Step 1 to 6 by 10 times with increasing numbers of PNP sets

An image of the interface upon completion of a successful experiment is shown in Figure 8.

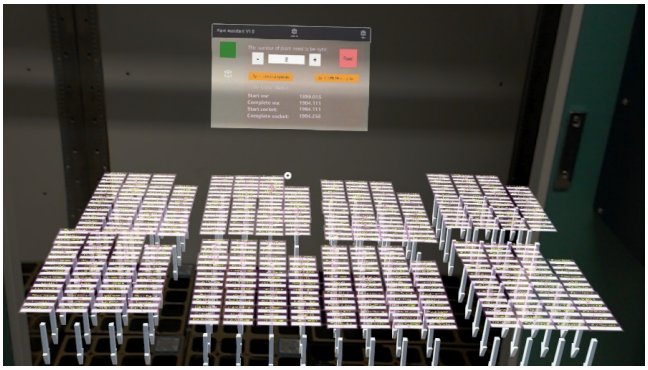

Figure 8: A view of the interface after associating all PNP sets with their corresponding plant trays with one-for-all method.

7.1 Results

Two sets of latency times were measured. One from the one-for-one method, which only includes the time taken to synchronize the corresponding number of WAs to locate the PNP sets. The other is the measured latency from the one-for-all method which not only includes the time used on synchronizing one WA for the reference object, but also on synchronizing the stored plant location data from the database.

We used Box and Whisker method to plot three graphs of the latency between the one-for-one and one-for-all methods. The synchronization time for the one-for-one method can be seen in Figure 9. The other two figures (Figures 10 and 11) show the synchronization time with the one-for-all method, however, one of them demonstrates the total latency cost with synchronizing one WA and the plant location data, the other only demonstrates the latency cost to synchronize the plant location data. This is split into two graphs because the user will only need to synchronize the WA once at device start-up; any further changes in the location of the PNP requires only the sharing of the scene vector data. Figure 11 therefore is useful for comparing the "time-to-resynchronization" in the case of a change in the location in one of one or more of the plant trays.

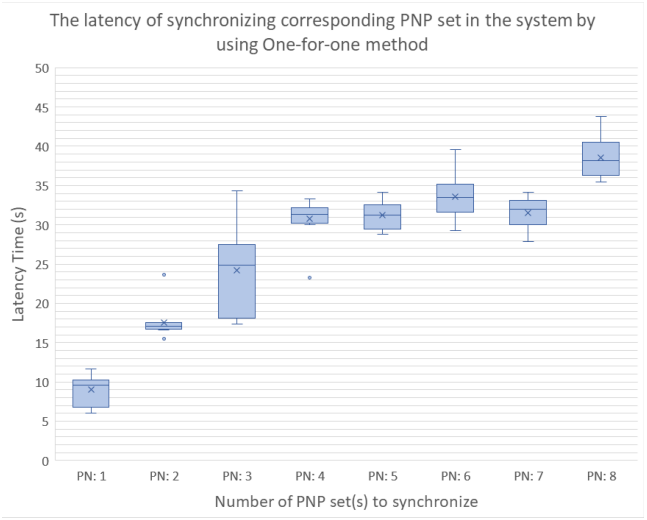

Figure 9: The latency cost on locating the corresponding PNP sets with one-for-one method.

8 DISCUSSION

Figure 9 shows the latency times that were generated on synchronizing 1 to 8 PNP sets to the greenhouse under the one-for-one method. The total time increased linearly with increasing the number of PNP sets, as expected. There are some outliers, however this could be due to issues with the HL device itself. We found that since the synchronize process took a long time, the device got very hot, and we also observed some frame-rate stuttering in the HL display when rendering the holograms of the PNP sets. This was observed when synchronizing more than PNP sets ($PN > 3$).

RIGHTSLINK

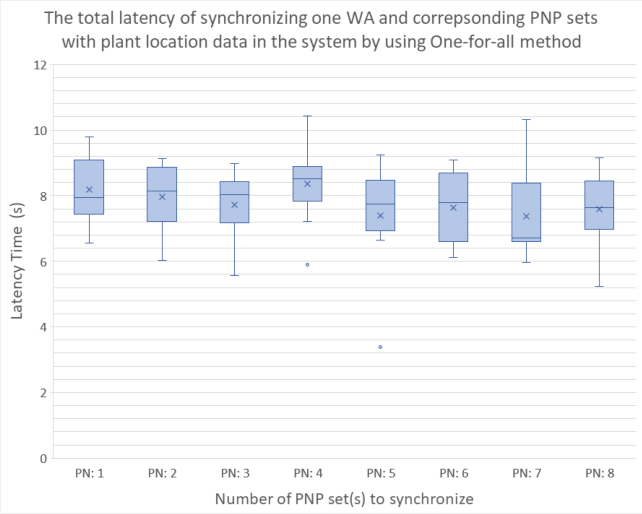

Figure 10: The total latency cost on locating the PNP sets with synchronizing both WA and plant location data.

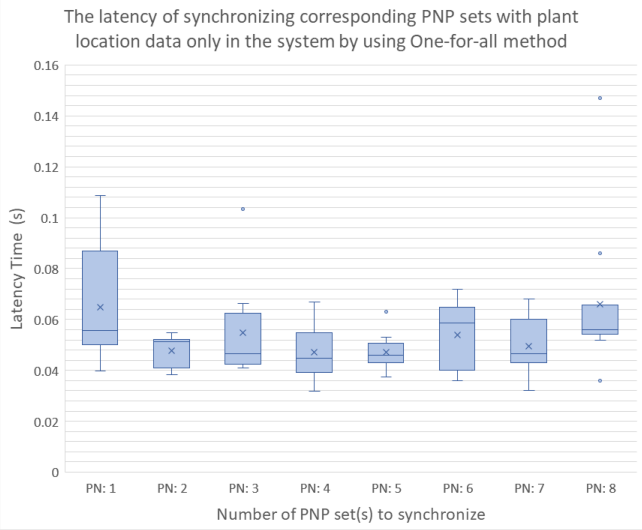

Figure 11: The latency cost on locating the PNP sets with only synchronizing the plant location data.

This could also explain why MS writes in the documentation that the (on-device) LAT method will sometimes fail and provide less robust anchor recall than the (cloud-based) ASA method, as mentioned in Section 2. The highest latency time ($PN = 8$) reached 40s, and this latency is too high to support natural real-time collaboration. Even in the $PN = 1$ case, the latency time was approximately 9s. This indicates that the LAT method provided by MS and Unity has obvious drawbacks for creating multi-user shared experiences with HL. The size of the WA used in LAT is too large to be transmitted among users with an acceptably low latency.

In contrast, Figure 10 shows a clear improvement over the one-for-one method. The latency time is approximately constant through synchronizing the different number of PNP sets (from 1 to 8). In each case, the total latency falls under 12s. There are also two outliers when $PN = 4$ and 5. This could be because the import process retried fewer times in these two cases. When testing the one-for-all method, the system did not get stuck or exhibit the stuttering as seen with the one-for-one method. Since synchronizing the WA only happens on start-up, in further use the WAs would not be in the processed again.

Figure 11 shows the time for synchronizing only the the plant locations (using the lightweight vector approach). This is a useful direct comparison with the Figure 9 plot representing the one-for-one method (where the full WA synchronization cost is incurred each time a plant tray is moved). Note that the latency shown in Figure 11 are included within Figure 10. The overall latency for all trials is similarly constant around 0.05s, although there is more variation in the results. This is to be expected because the latency time is so small. Again, the reason why this latency is so small is because of the size of the plant location data is much smaller in this case—around 200 bytes.

This indicates that after users have done the initial synchronization on system start-up, they could work collaboratively in real-time under this system.

The OFALL-SE method does have some disadvantages. In particular, the "reference" WA has a limitation on the size of the offset distance with which it can be used accurately. MS states this limitation in their official document as:

"Spatial anchors stabilize their coordinate system near the anchor's origin. If you render holograms more than 3 meters from that origin, those holograms might experience noticeable positional errors in proportion to their distance from that origin due to lever-arm effects."[Zeller et al. 2019c]

Therefore if a user is more than 3 meters from the reference WA, the corresponding PNP sets might have the positional errors throughout HL devices. Thus, the one-for-all method, may be limited to small greenhouses. This was not a constraint in our test greenhouse environment, but for other scenarios such as a larger greenhouse this should be considered and addressed by developers.

9 CONCLUSION

In this paper we describe a new AR system for low-latency asset tracking and collaboration and examined its performance in the context of a research greenhouse environment. We combined the methods of world tracking from the HL device and the marker-based tracking to track the physical location of the plant tray.

We have also modified the OFALL technique to speed up the transmission of the plant spatial locations among the users via a customized cloud server without using ASA. We have also completed an experiment which showed that the latency on synchronizing the spatial locations of the plant with the new OFALL-SE method was consistently better than the one-for-one method recommended in the Unity official user manual, and that using this modified method the sharing of spatial locations in a multi-user scenario could be operated with sufficiently low latency for real-time collaboration. This also enable the users who would not like to use ASA to share the anchors by MS' LAT method with their own customized cloud server.

RIGHTSLINK

REFERENCES

Abdulrahman Alkandari, Nayfah Mohsen Almutairi, Wasmya Alhayyan, and Abeer Essa Alomairi. 2019. Google Project Tango and ARCore Under the View of Augmented Reality. *Journal of Computational and Theoretical Nanoscience* 16, 3 (2019), 1127–1133.

Ramon Arguelles, Matt Wojciakowski, Cory Fowler, and Ross McAllister. 2019. Azure Spatial Anchors overview. https://docs.microsoft.com/en-us/azure/spatial-anchors/overview Accessed: 2019-07-17.

Ronald T Azuma. 1997. A survey of augmented reality. *Presence: Teleoperators & Virtual Environments* 6, 4 (1997), 355–385.

Brandon Bray, Matt Zeller, Vanessa Arnauld, Neeraj Wadhwa, Jesse McCulloch, Eliot Cowley, and Kyle Burns. 2018. MR Sharing 240: Multiple HoloLens devices. https://docs.microsoft.com/en-us/windows/mixed-reality/holograms-240 Accessed: 2019-07-17.

Gabriel Evans, Jack Miller, Mariangely Iglesias Pena, Anastacia MacAllister, and Eliot Winer. 2017. Evaluating the Microsoft HoloLens through an augmented reality assembly application. In *Degraded Environments: Sensing, Processing, and Display 2017*, Vol. 10197. International Society for Optics and Photonics, 101970V.

Mark Fiala. 2005. ARTag, a fiducial marker system using digital techniques. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, Vol. 2. IEEE, 590–596.

Carolien Kamphuis, Esther Barsom, Marlies Schijven, and Noor Christoph. 2014. Augmented reality in medical education? *Perspectives on medical education* 3, 4 (2014), 300–311.

Hiroshima Kato, M Billinghurst, R Blanding, and R May. 1999. ARToolKit manual. *PC version* 2 (1999).

Kangdon Lee. 2012. Augmented reality in education and training. *TechTrends* 56, 2 (2012), 13–21.

Microsoft. 2019. UI and Interaction Building blocks. https://github.com/Microsoft/MixedRealityToolkit-Unity#ui-and-interaction-building-blocks Accessed: 2019-07-17.

Ivan Permozer and Tihomir Orehovački. 2019. Utilizing Apple ARKit 2.0 for Augmented Reality Application Development. In *42nd International Convention on Information and Communication Technology, Electronics and Microelectronics*.

Alexandro Simonetti Ibañez and Josep Paredes Figueras. 2013. *Vuforia v1. 5 SDK: Analysis and evaluation of capabilities*. Master's thesis. Universitat Politècnica de Catalunya.

Alex Turner, Jesse McCulloch, Brandon Bray, Matt Zeller, Eric Fiscus, Sean Ong, Eliot Cowley, and Graham and Bury. 2019. Coordinate systems. https://docs.microsoft.com/en-us/windows/mixed-reality/coordinate-systems Accessed: 2019-07-17.

Alex Turner, Jesse McCulloch, and Graham Bury. 2018. Local anchor transfers in Unity. https://docs.microsoft.com/en-us/windows/mixed-reality/local-anchor-transfers-in-unity Accessed: 2019-07-17.

UnityTechnologies. 2019. HoloLens Anchor Sharing. https://docs.unity3d.com/Manual/windowsholographic-anchorsharing.html Accessed: 2019-07-17.

Reid Vassallo, Adam Rankin, Elvis CS Chen, and Terry M Peters. 2017. Hologram stability evaluation for Microsoft HoloLens. In *Medical Imaging 2017: Image Perception, Observer Performance, and Technology Assessment*, Vol. 10136. International Society for Optics and Photonics, 1013614.

Daniel Wagner, Thomas Pintaric, Florian Ledermann, and Dieter Schmalstieg. 2005. Towards massively multi-user augmented reality on handheld devices. In *International Conference on Pervasive Computing*. Springer, 208–219.

Sabine Webel, Uli Bockholt, Timo Engelke, Nirit Gavish, Manuel Olbrich, and Carsten Preusche. 2013. An augmented reality training platform for assembly and maintenance skills. *Robotics and Autonomous Systems* 61, 4 (2013), 398–403.

Matt Zeller, Jesse McCulloch, Eliot Cowley, Brandon Bray, and Alex Turner. 2019a. Shared experiences in mixed reality. https://docs.microsoft.com/en-us/windows/mixed-reality/shared-experiences-in-mixed-reality Accessed: 2019-07-17.

Matt Zeller, Alex Turner, Jesse McCulloch, Jackson Wonderly, and Brandon Bray. 2019b. Shared experiences in Unity. https://docs.microsoft.com/en-us/windows/mixed-reality/shared-experiences-in-unity Accessed: 2019-07-17.

Matt Zeller, Alex Turner, Jesse McCulloch, Jackson Wonderly, and Brandon Bray. 2019c. Spatial anchors. https://docs.microsoft.com/en-us/windows/mixed-reality/spatial-anchors Accessed: 2019-07-17.

Wenxiao Zhang, Bo Han, Pan Hui, Vijay Gopalakrishnan, Eric Zavesky, and Feng Qian. 2018. CARS: Collaborative Augmented Reality for Socialization. In *Proceedings of the 19th International Workshop on Mobile Computing Systems & Applications (HotMobile '18)*. ACM, New York, NY, USA, 25–30. https://doi.org/10.1145/3177102.3177107

Dylan TX Zhou, Tiger TG Zhou, and Andrew HB Zhou. 2015. Wearable augmented reality eyeglass communication device including mobile phone and mobile computing via virtual touch screen gesture control and neuron command. US Patent 9,153,074.

RIGHTSLINK