Spatial Reconstruction using Microsoft HoloLens™

Department of Computer Science
Faculty of Engineering
University of Hong Kong

Interim Report

GUPTA, Aman	ZAFAR, Waleed	SCHNEIDERS, Dirk
3035206885	2013600952	Superviser

January 2018

Abstract

Advancements in hardware have led to the invention of the Microsoft HoloLens™, a mixed reality system that allows users to perform physical interactions with the digital world in real time. The device combines features of Virtual Reality (VR) with the physical world creating Augmented Reality (AR). Nowadays, 3D indoor models can be constructed using construction blueprints and plans; however sometimes, the plans are unavailable or might not be a true representation. In such cases, the areas must be surveyed and reconstructed through a time consuming and costly process. This project aims to showcase the ease with which The Microsoft Hololens™ can be used to perform spatial reconstruction of indoor environments and to visualize them as holograms. The reconstructed, processed holograms will be complemented with interactive features. This report presents the technology being used, the team's methodology, and the schedule for the project. The interim report also discusses the findings of the initial feasibility research as well as the current status of the project.

Contents

List of Figures

List of Tables

Acknowledgement

The team acknowledges and thanks Professor Dirk Schneiders for supervising and consulting them to proceed on the project.

The team also acknowledges the Department of Computer Science, the Faculty of Engineering, and the University of Hong Kong for providing the necessary hardware for this project.

Abbreviations

API Application Programming Interface

AR Augmented Reality

BSD Berkeley Software Distribution

CGAL Computational Geometry Algorithms Library

GGV Gaze-Gesture-Voice

HLA HoloLens™ Application

LiDAR Light Detection and Ranging

PCL Point Cloud Library

PRPU Post-Recording Processing Unit

RANSAC Random Sample Consensus

SDK Software Development Kit

UWP Universal Windows Platform

UX User Experience

VCG Visualization and Computer Graphics Library

VR Virtual Reality

1 Introduction

Computer modeling software have allowed designers and architects to visualize 3-dimensional models on 2-dimensional screens. With advancements in technology and the advent of VR and holographic projections, it is now possible to visualize and interact with environments in unforeseen ways. The Microsoft HoloLens™ is the leading tool for holographic manipulation. It features some of the most advanced AR technology that is commercially available. Interested developers have already created some groundbreaking applications to bring the activities of designing and modeling to the AR space.

The market for applications that project holograms is booming; however, the development of these holograms usually requires using complex modelling software to mirror a real-world indoor environment. Capturing environments and objects to rebuild them as holograms is an unexplored realm in this nascent industry. Existing systems trying to achieve the same objective usually exhibit accuracy and quality issues owing to hardware limitations, or cost prohibitions. The Microsoft HoloLens™ provides a decent trade off between cost and hardware, and has not been previously used, in a public and commercial capacity, for indoor 3D scanning purposes.

The team sees an opportunity in using the HoloLens™ device to create a comprehensive solution to serve the needs of capturing and visualizing indoor environments. The approach of scanning indoor environments instead of designing them on a software is analogous to what a photograph is to a painting - a user will be able to build more accurate 3D models at a much faster pace.

At the current state of the project, the team has developed a prototype application for the HoloLens™ which allows a user to record a 3D mesh of an indoor environment. The team is currently developing a robust spatial reconstruction algorithm as part of the PRPU. Completion of spatial reconstruction on the PRPU would allow the team to move on to the completion of the HoloLens™ Application (HLA) alongside development of a web portal which the team aims to deliver as part of the project.

This report begins with a review of relevant works pertaining to spatial reconstruction. It then highlights the primary objectives for the project, followed by an overview of the technology utilized by the team as well as the overall application design for the project. This is followed by a description of the team's

algorithmic design for performing spatial reconstruction, accompanied by a detailed analysis of some existing algorithms for spatial reconstruction. The report finally outlines the methodology employed by the team, the project's current status as well as a brief discussion of the challenges and limitations for the project.

2 Related Works

There are several hardware and software solutions in the public domain that have attempted to solve the problem: scan and reconstruct real world environments. However, they have been limited by hardware; sensors and processing power alike. Microsoft HoloLens™ is the first Augmented Reality device of its kind.

2.1 Microsoft Kinect™ Fusion [6][14]

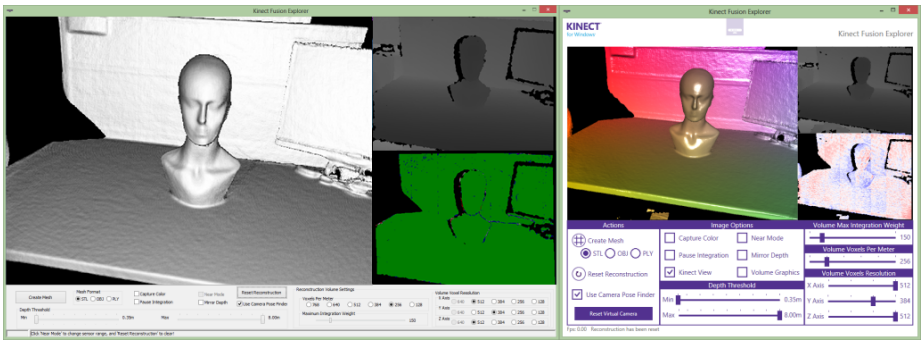

Figure 1: Kinect Fusion Scanning and Reconstruction [14]

Kinect Fusion 3D provides object scanning and model creation using a Microsoft Kinect™ sensor. Capabilities of the project include the ability to simultaneously see and interact with detailed 3D models scanned using the sensor. However, Kinect Fusion™ utilizes a single depth sensing camera for tracking, which results in poor performance while scanning flat surfaces. Furthermore, the Kinect™ sensor is not wireless, hindering mobility for the project in varying indoor environments.

2.2 Laser Scanning[12]

Laser Scanning is one of the most popular methods of indoor surveying to generate 3D models, and is used in a number of commercial applications. One of the most common technologies under laser scanning utilizes Light Detection and Ranging (LiDAR) to gather 3D point cloud data. However, extracting actual building models from large data-sets generated via LiDAR is a costly and time-consuming endeavor and requires experienced technicians at work. Some commercial solutions utilizing laser scanning technology include the Trimble LaserAce 1000 Rangefinder[16] and the Leica ScanStation P40[11]. These solutions do exhibit problems similar to those faced by the LiDAR, issues include shape distortion and angular inconsistencies while scanning indoor environments.

Figure 2: Leica ScanStation P40

Figure 3: Trimble LaserAce 1000 Rangefinder

Apart from the specific limitations described above, most solutions do not use mesh correction algorithms to fill in the holes and remove bias from generated 3D models. The absence of a strong, coherent mesh correction process results in inaccurate 3D models which might not be suitable for applications that demand high precision, such as construction and industrial automation.

3 Objectives

The project is aiming to construct a HLA, a PRPU as well as a complementary web platform. Therefore, the key goals are as follows:

1. **HoloLens™ Application (HLA):** The application will feature a clean and simplistic user interface that is controlled by voice commands and hand gestures. The friendly User Experience (UX) allows the user to record new environments or view existing holograms easily.

2. **Record New Environment:** The team plans to use the depth-sensing camera and existing APIs of the Microsoft HoloLens™ to create 3D mappings of surrounding environments. The HoloLens™ provides hardware and software that is already capable of stitching together surface meshes as a user moves around in an indoor environment. The user simply presses the record button then walks around a room scanning its features.

3. **Hologram Visualization:** The proposed application will give the users the ability to project holographic representations of 3D models scanned via the application, and span, zoom, and rotate the holograms on the surface it is projected.

4. **Streamlined Web Application (proposed extension):** A simplistic web application that allows the user to manage their recorded environments on the web.

4 Technology Used

4.1 Microsoft HoloLens™

Figure 4: HoloLens™ Developers Edition Kit [7]

It is an advanced head-mounted display with tinted visors that function as a see-through display, as shown in figure 4. The HoloLens™ has 4 environment sensing cameras and 1 depth sensing camera[3]. The cameras mounted on the device are already optimized for spatial mapping and come coupled with complex algorithms that are able to create 3D triangular meshes for the spaces it captures.

The device supports voice commands as well as hand gestures as input. It uses the orientation of the head to identify the gaze vector of the user and point. Selections in the AR environment can be either made by a tap gesture by the index finger (Figure 5) or using the HoloLens™ clicker pictured in the center of Figure 4 [4].

Figure 5: Tap Gesture within the HoleLens' field of view [4]

4.2 HoloLens™ SDK

The HoloLens™ follows the Universal Windows Platform (UWP) development framework and therefore has no separate special SDK or toolkit. However, Microsoft has released an open-source collection of scripts and components to accelerate the development of applications targeting Windows Mixed Reality [8]. Libraries such as HoloToolkit and Mixed Reality Design Lab are essential for development of the application. The development environment includes:

1. **Visual Studio 2017:** Integrated development environment for building and deploying UWP applications to the HoloLens.

2. **Unity 2017.1:** A Unity engine that comes with Mixed Reality Support [5] and is primarily used to develop applications in 3D space. Prefab com-

ponents such as Spatial Understanding, Input Mapping, and HoloLens™ Camera are inserted into a Unity scene along with 3D objects to design interactions and behaviours.

3. **Mixed Reality Portal:** The Mixed Reality Portal is the latest emulator developed by Microsoft to test Mixed Reality applications without a HoloLens™ [10]. Although the HoloLens™ device is essential for testing capturing techniques, development of the UX work-flow can be done on an emulator.

4.3 PCL [15]

The PCL is one of the most widely used standalone, open-source point cloud processing library. It is distributed using the Berkeley Software Distribution (BSD) license and has implementations for several spatial reconstruction algorithms. The project provides a C++ API with a wide array of third-party libraries and algorithm implementations. The team has identified the classes and modules that will be essential for the project. They include:

1. **pcl::OBJReader Class:** The Microsoft HoloLens™ captures raw data in Wavefront Advanced Visualizer file-format [17]. This format is not directly supported by the PCL library therefore, the OBJReader class will be used to convert the Object file into Point Cloud data.

2. **Sample Consensus Module:** This module contains methods such as RANSAC algorithms that can be used in conjunction with Filters Module to clean the raw data before performing surface reconstruction.

3. **Surface Module:** The main aim of this module is to provide developers with algorithms used for surface reconstruction from 3D scans. The library has algorithms that can deal with smoothing and re-sampling as well as hole-filling tasks. Examples of classes included in this module are **pcl::ConcaveHull**, **pcl::GreedyProjectionTriangulation** and **pcl::Poisson**.

4. **Filters Module:** This module contains mechanisms for outlier and noise removal for a point cloud data. This is an important step as the raw data from the HoloLens™ will contain several outliers and noise.

4.4 Visualization and Computer Graphics Library (VCG) & Computational Geometry Algorithms Library (CGAL)

VCG is a large open-source library to process and manipulate triangular and tetrahedral meshes. It is distributed using the GPL license and comprises of various useful algorithms for computation of curvatures, smoothing and, mesh reparing. Similarly, CGAL also provides an extensive API for triangulation, surface and volume mesh generation, convex hull algorithms as well as point-set processing.

By combining the various features available in each of these libraries, the team will have a vast amount of library functions and algorithms to employ for the spatial reconstruction phase.

5 Application Design

The final system will incorporate the web portal, the PRPU as well as the HLA. All users will be able to use the HLA to scan an indoor environment as well as view the processed holograms by accessing their private or public collections. Although the HoloLens™ is the only device that provides the hologram visualization and manipulation abilities, the users will also be able to port the holograms as immersive environments for VR headsets (proposed extension).

The design of the project is subject to change with the progress of study of the capabilities of devices, manuals and documentations, and the timeliness of the development. This section will describe the team's design ideas and subsequent sections will discuss the progress of implementation.

5.1 HLA

Once deployed on and installed through the Windows Store, the HLA can be accessed through a HoloLens™ device's *bloom* menu. On first initialization by the user, the HLA will display a unique username and password created for them to access their web portal. Subsequently, the application will display a holographic menu allowing a user to either view their collection or begin recording a new environment.

Figure 6: Triangular mesh over an unidentified column and ceiling

Figure 7: Blue lines over a partially identified wall

5.1.1 Recording Indoor Environments

This is the core feature of the project. The heavy weight of spatial mapping is already taken care of by the device which records a 3D mesh of triangles to identify planes in its vicinity. However, the existing software is limited by the buffer of mesh it can record and its inability to fill gaps left by unidentified planes.

On selection of the Record button, the solution is going to begin spatial mapping and offer the ability to Stop or Cancel the recording. Walls and planes that are being mapped by the HoloLens™ will show up as the triangular mesh (Figure 6) where as identified planes will be overlaid with blue lines as shown in Figure 7. It is in the user's best interest to ensure that most planes are overlaid with the blue lines before they stop recording and handover the recorded object to the processing service.

5.1.2 Visualization

The user will be able to revisit their recorded environments using the *My Library* button in the main scene. On selecting an environment, the user will have the ability to place the environment's hologram on a flat surface such as table and manipulate it. Additional features such as identifying, placing, or removing objects such as walls or furniture in the hologram will also be made available to the user. The team will also explore designing a multi-user platform (Figure 8) for visualization so that businesses can creatively use the application for their needs collectively as a team. API for the aforementioned feature is already available in the UWP SDK however, due to the unavailability of multiple HoloLens™ devices, the team faces an additional challenge while developing a

Figure 8: A team of designers looking at a holographic model. [7]

multi-user platform.

5.2 PRPU and Web Portal

The need for a processing unit arises because the spatial data collected from the HoloLens™ contains outliers, holes and noise. Once the user stops their recording, the recorded spatial data will be delivered to the PRPU server through the internet. The PRPU will be responsible for applying filters and spatial reconstruction algorithms to create a useful model which will then be saved on the server as well as the HoloLens™ device.

The team is exploring the use of Web2Py Framework for the development of the PRPU. As the PCL API is written in *C++*, *Python* packages such as *ctypes* will be used to communicate with it. *Firebase* will be used as the solution's primary database while front-end technologies such as *Angular* and *React* will be used to design the user interface.

The team also recognizes the need for data security during network transmissions hence, all user data would be encrypted using AES-256 encryption.

Figure 9: Application Flowchart

The flowchart in Figure 9 provides an overview of the steps the user takes through the application.

6 Spatial Reconstruction

This section outlines the overall system design for spatial reconstruction performed on the mesh obtained from the HoloLens™, alongside some of the algorithms and filters whose utility the team is currently investigating as part of the development process.

6.1 System Design

Spatial reconstruction for the spatial data provided by the HoloLens™ is performed in two separate stages. The first stage is executed within the HLA, and utilizes the Spatial Understanding API provided by Microsoft™ as part of the Mixed Reality Toolkit for Unity. The second portion of the spatial reconstruction process is handled by the PRPU, involving algorithms developed by the team specifically for the project.

6.1.1 Spatial Understanding API

The Spatial Understanding API is available for the Microsoft HoloLens™ as a *prefab* in the Mixed Reality Toolkit for Unity. The API provides three important interfaces: Topology Query, Object Detection and, Object Placement. For the purpose of Spatial Reconstruction, the Topology Query interface is the most relevant to the team.

Figure 10: Left: Mesh Scanning along with Playspace Stats Information. Right: Finalized Mesh by the HoloLens

The recording stage performs three basic steps:

1. **Indoor Space Scanning and Quality Check:** In this step, the application uses the *PlayspaceStats* structure to assess the quality of the mesh at the time of recording. Information such as *TotalSurfaceArea, HorizontalSurfaceArea, WallSurfaceArea, FloorSurfaceArea, CellCount_IsSeenQuality* are retrievable through the *GetStaticPlayspaceStats()* method. Using this information, the application is able to ensure that a certain threshold for the pre-processed mesh is maintained.

2. **Request Finish Scanning:** The *RequestFinish()* method is called by the HLA once the predefined criteria for the indoor environment are met. The *ScanState* of the Spatial Understanding prefab transitions from *Scanning* to *Finishing*. Once in the finishing state, the HoloLens™ makes an attempt to create a water-tight mesh for the indoor environment. Thereafter, the *ScanState* is set to *Done*.

3. **Mesh Retrieval:** This step calls on the *GetMeshFilters* method. Using the *MeshFilters* structure and the *MeshSaver* interface, the HLA is able to generate a mesh object that can be uploaded to the PRPU.

6.1.2 PRPU

The PRPU receives a mesh that has already undergone processing via the Spatial Understanding API. The mesh provided by the HoloLens™ provides three key pieces of information for spatial data - vertices, vertex normals and faces. There are three primary problems[9] with the spatial data generated by the HoloLens™ which the PRPU addresses. These include holes, where some real world surfaces are missing from the data, hallucinations, visible as surfaces or objects within the data that do not exist in the real world, and bias, whereby surfaces in the spatial mapping data are misaligned with their counterparts in the real world. The following list showcases the steps that a mesh undergoes within the PRPU. All images referenced within this list are part of figure 11.

1. **Partially processed mesh retrieved from the HoloLens™**
   The PRPU, running on the team's server, receives a partially processed mesh from the HoloLens™. This mesh is available as an OBJ file, and includes information regarding vertices, vertex normals and faces. The

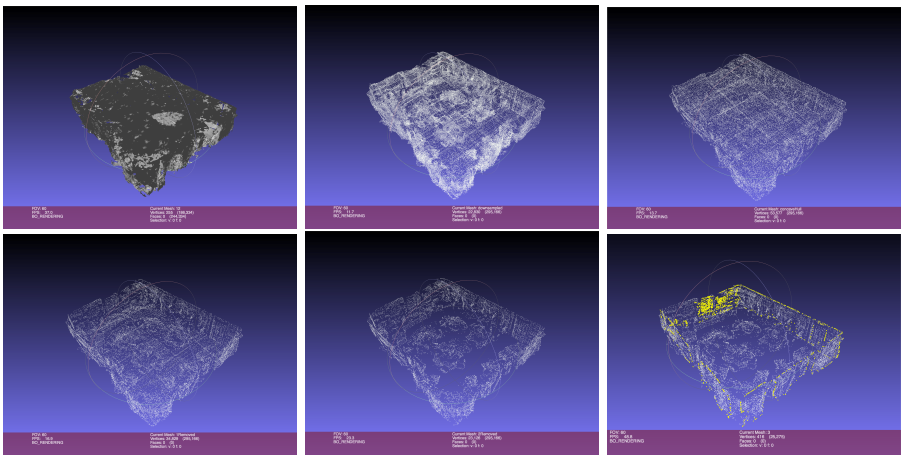

Figure 11: Clockwise from top left: a. Imported Mesh. b. Point Cloud Conversion. c. Concave Hull. d. Ceiling Segmentation. e. Floor Segmentation. f. Wall Segmentation.

team has found the information provided by the vertex normals to be comparatively unreliable, and has thus elected not to rely on that information for reconstruction purposes as yet. A sample mesh fetched from the HoloLens™ is visible in part (a) of figure 11.

2. **Conversion to point cloud**
   The team is currently using PCL, which solely relies on point clouds. This mandates conversion of the retrieved mesh to a point cloud, as visible in part (b) of figure 11.

3. **Concave hull generation**
   A concave hull is generated for the entire model based on the point cloud data, primarily to create a tight outline of the point cloud data as well as, to reduce any holes within the outer hull. This is a computationally intensive operation, and one of the key operations that mandate the use of an external server as opposed to performing all spatial reconstruction operations on the fly within the HLA. The point cloud following generation of the concave hull is visible in part (c) of figure 11.

4. **Detection and removal of horizontal surfaces**
   The team's approach to the problem of spatial reconstruction involves

modularization. They attempt to segment the available model into smaller models that can be reconstructed independently and merged at the end. This results in a final 3D model that requires comparatively less computational power and time to generate. Owing to the existence of bias in the data, horizontal planes such as the roof and the floor can be most reliably extracted from the spatial data at the first step. This is because these planes are expected to lie parallel to the XZ plane. The team utilizes the RANSAC algorithm to detect these planes. Following plane detection, the detected plane is stored as a separate point cloud, and removed from the original point cloud. This step results in 3 distinct point clouds, representing the roof, the floor and the remaining elements for the model, visible in parts (d) and (e) of figure 11.

5. **Detection and removal of outer walls**
   The model available at this step already has the roof and the floor removed from the model. The model is then passed through a voxel grid filter to reduce noise present in the model. The team then utilizes the *SACSegmentation* class from the PCL to generate segments of the outer hull i.e. the walls for the model. The team utilizes the plane coordinates for the floor segment, attempting to find segments perpendicular to the floor. Following detection, each segment is removed from the original model and stored separately, in the same manner as the floor and the roof. For visualization purposes, a concave hull for each segment is generated, visible in part (f) of figure 11, which shows the first four segments detected for the model under processing.

6. **Reconstruction of individual segments**
   Once all the horizontal planes (i.e. floor and roof) and vertical planes (walls) for the outer hull are detected, plane coordinates for each individual segment are calculated and stored. These plane coordinates can then be used to reconstruct a mesh corresponding to each point cloud segment.

7. **Construction of outer hull based on segment intersections**
   The reconstructed meshes for each individual segment will be extrapolated within their own planes until an intersection with another plane segment is detected. This would present the team with an edge at the two planes' intersection. This method would be repeated to utilize all the reconstructed

segments until a completely reconstructed and closed outer hull for the 3D model is obtained.

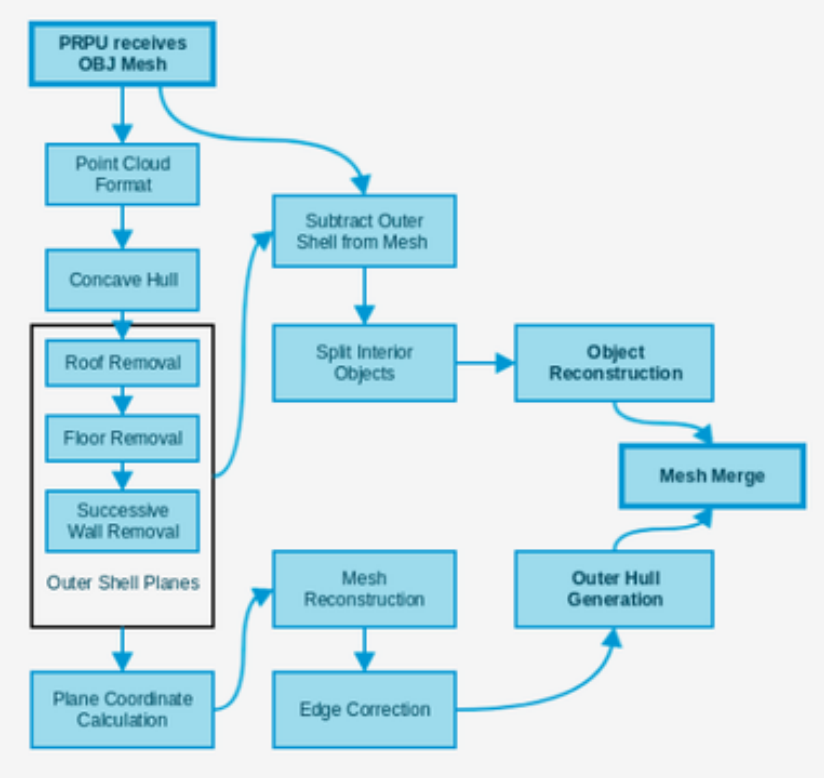

Figure 12: PRPU Process Flow Diagram

8. **Interior elements' construction**

   The team's approach handles mesh construction for interior elements of a model independently from the construction of an outer hull. All preceding steps serve only to construct an independent outer hull for the model. A major problem that the team is yet to address is the task of object identification, whereby the algorithm needs to determine whether an object present inside a given model actually needs to be included in the final model. This necessitates distinction between objects that need to be included in the model, such as pillars, against objects that need not be included, such as tables or chairs. The steps below outline a possible approach for the reconstruction of interior elements.

(a) **Subtraction of outer hull points from remaining point cloud**
The finalized mesh generated for the outer hull will be converted, as a copy, to a point cloud. Any intersection between the remaining point cloud and the outer hull's point cloud will be removed from the model under processing, resulting in a point cloud of elements not comprising the outer hull.

(b) **Segmentation of interior elements**
The model would then be further segmented, likely via some form of cluster analysis, to represent each individual element within model. The objective at this stage is to obtain individual point clouds for individual elements, such as a point cloud representing a single pillar or a single table.

(c) **Mesh reconstruction for individual elements**
Following the segmentation of interior elements, a mesh can be reconstructed for each element individually. Possible algorithms for mesh reconstruction are detailed in section 6.2. Once each individual element is reconstructed as a mesh, the team can explore possibilities for object recognition, so as to discard interior elements that need not be included in the final 3D model. The filtering process would be followed by merging the remaining individual models to form a single model for all interior elements that are to be included.

9. **Merging interior elements with exterior hull**
At this stage, there would be two separate 3D models, one containing the exterior hull and one containing all the interior elements that need to be included in the model. The final step would be to simply merge the two models, resulting in the final 3D model. The final model would then be pushed back to the HLA and be visible within the user's library.

6.2 Algorithms

6.2.1 Down-sampling using Voxel Grid Filter

The need for down-sampling arises due to the noisy nature of the point-cloud data obtained from the HoloLens™. Voxel Grid Filter in the PCL library down-samples the data by taking a spatial average of the points in the cloud. Essen-

tially, the algorithm divides the data into small cubes known as voxels and all the points present in a voxel will be approximated with their centroid [15].

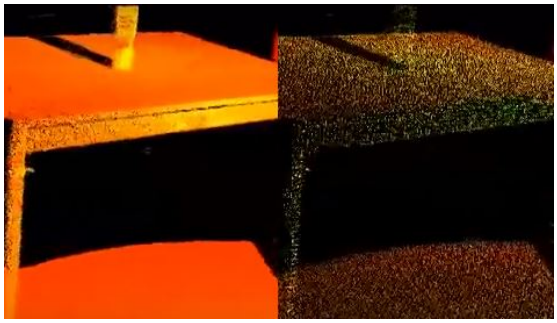

Figure 13: The original point-cloud data and result after application of Voxel Grid Filter [15]

Figure 13 shows how the Voxel Grid filter can be used to down-sample a noisy point-cloud data. The table in the left image is almost solid as it has 460400 data points whereas, after applying the filter, the resultant point-cloud only has 41049 data points. This is approximately a $10 factor$ reduction in the number of data points.

6.2.2 RANSAC Algorithm for Plane Detection and Fitting[18]

RANSAC Algorithm is one of the most widely used algorithms for plane detection. In principle, the algorithm searches for the best plane in a 3D point cloud. For time efficiency, the algorithm randomly selects three points and calculates the parameters of the corresponding plane. After this, the algorithm identifies all points in the original cloud belonging to the calculated plane with respect to a given threshold. The algorithm performs this procedure several times comparing the obtained result with the best result obtained in the previous iterations.

The primary objective of the RANSAC algorithm will be to identify planes and separate them so that hole-filling tasks can be performed on each plane separately. The pseudo code [18] of the RANSAC algorithm is as follows:

Algorithm 1 RANSAC Algorithm

1: $bestStd \leftarrow \infty$
2: $i \leftarrow 0$
3: $\epsilon \leftarrow$ *1 - forseeable-support/length(point-list)*
4: $N \leftarrow$ **round** $\log(1-\alpha)/\log(1-(1-\epsilon))$
5: **while** $i \leq N$ **do**
6: $j \leftarrow pick3pointsrandomlyamong(point-list)$
7: $pl \leftarrow pts2Plane(j)$
8: $dis = dist2plane(pl, point-list)$
9: $s = find(abs(dis) \leq t)$
10: $st = standardDev(s)$
11: **if** $length(s) < bestSupport$ **OR** $(length(s) = bestSupport$ **AND** $st < bestStd)$ **then**
12: $bestSupport \leftarrow length(s)$
13: $bestPlane \leftarrow pl$
14: $bestStd \leftarrow st$
15: $i \leftarrow i + 1$

6.2.3 Advancing Front Algorithm [2]

Cohen-Steiner D. and Da F. have worked extensively in the field of Advancing Front Alogirthms. Their proposed Greedy Delaunay Based Surface Reconstruction Algorithm stitches triangles incrementally on the basis of their plausibility to fill holes. Their selection criteria uses a confidence-based mechanism along with a control on the topology of the reconstruction to reduce ambiguities common in greedy approaches. Each triangle t that is considered by the algorithm must share an edge e with the boundary of the current reconstruction. This condition leads to four different situations where t can be added to the surface as shown in Figure 14

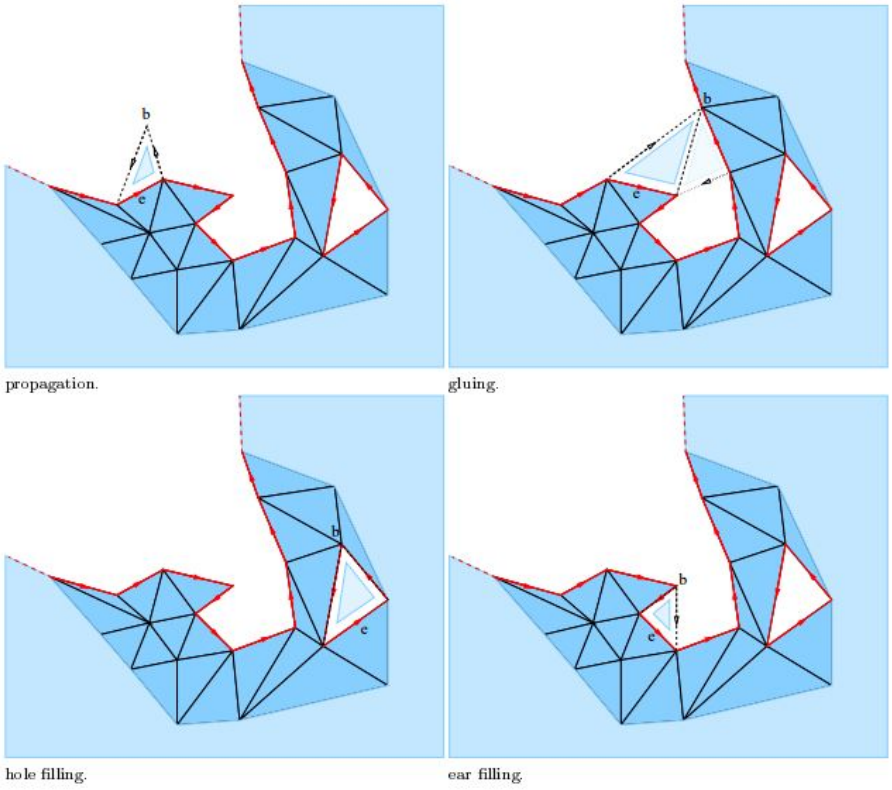

Figure 14: The four situations where triangle t can be added[2]

6.2.4 Poisson Surface Reconstruction[13]

Poisson surface reconstruction uses an implicit function framework by computing a 3D indicator function and then extracting an appropriate isosurface. The algorithm focuses on the relationship between oriented points on a surface. Since Poisson systems are resilient to the presence of outliers or imperfect data, the system works very well with the problem of surface reconstruction. Instead of using local fitting techniques, this particular algorithm attempts to find a global solution hence, considers the entire point cloud at once. Poisson Surface Reconstruction algorithm creates a smooth solution that robustly removes noisy data giving it a greater advantage when dealing with closed figures.

6.2.5 Convex and Concave Hull [15] [1]

In a convex hull algorithm, the smallest subset of the given points is generated such that the subset is able to encapsulate all the points. A concave hull algorithm, on the other hand, can be used to generate an envelope for a set of points such that the tightest boundary for the given points is generated. The Quick Hull algorithm relies on this concept together with the Beneath-beyond algorithm. The idea can be realized through Figure 15. The point-cloud of the object is encapsulated in a wireframe that consumes the least number of data points and visible facets.

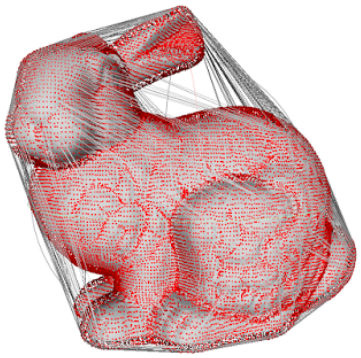

Figure 15: 3D point cloud data (red dots) with the convex hull (wire-frame) [1]

7 Methodology

This chapter presents the methodology that is being employed throughout the development of the project.

7.1 Hardware Setup and SDK Installations - Completed

Microsoft provides a detailed instruction set to install the SDK[5]. The team is using Unity Engine 2017.1 and the latest iteration of Visual Studio to benefit from the on-going developments in the field of Mixed Reality.

Furthermore, the team has also cloned repositories from GitHub with a

library of prefabricated (prefab) components and scripts designed for HoloLens™ development[8].

7.2 Feasibility Study - Completed

The main aim of this task was to understand the structure and components of the SDKs as well as the functions of the components. Therefore, the team carried out API research on the HoloLens™ and explored tutorials created by Microsoft Academy to understand the development process for a HLA.

The Feasibility Study also comprised of discovery and assessment of existing libraries for spatial reconstruction, plane identification, and mesh-processing. Software to visualize and manipulate meshes on a debugging machine would boost the development time-line and therefore needed to be extensively researched during the phase.

It was essential that the team understand the capabilities of the device in the early stages of the project. Knowing these limitations has helped the team decide and design the objectives and scale of the project.

7.3 Spatial Reconstruction - In Progress

After understanding the HoloLens™ device, the team focused on developing the spatial reconstruction part of the project. This part consists of the HLA and the PRPU.

Spatial reconstruction for the raw mesh generated by the HoloLens™ will be performed in two distinct phases. The first phase utilizes the Spatial Understanding API available via the HoloToolkit for Unity. The mesh generation feature performs the recording and partial reconstruction within the HLA as described in Section 5.1.1. The partially reconstructed mesh is then uploaded to the application's PRPU for the second phase of spatial reconstruction.

7.4 Visualization and User Collection

Upon successfully completing the tougher task of recording and processing the indoor environments, the team will devote their energy on the visualization of the spaces. First the application will support placing a scaled down model of the recorded objects onto a flat plane. Easy manipulations such as rotations,

pick and drop will be incorporated. Thereafter, support for additional features mentioned in Section 5.1.2 will be developed.

7.5 Environment Export and Web Portal

Once the application is able to perform the two aforementioned functions to an acceptable degree of satisfaction, the team will begin working on making the service web-friendly. A cloud service will be setup where the users will be able to manage their collection as well as view open-source spatial mappings of other users on the team's solution. The team will incorporate capabilities for exporting the models in different formats so they are compatible across various Virtual Reality devices.

7.6 Fine Tuning

After a prototype for the web portal is completed, the team will start to refine the solution. Details and user interfaces will be improved and thoroughly tested before presenting the final product.

8 Current Status

This chapter will discuss and conclude the various milestones achieved by the team across the various development phases.

8.1 Feasibility Study

Through extensive research of the HoloLens™ hardware and SDK, the team has come to a conclusion to move ahead with the project. The team is confident that a solution can be developed within the time range using HoloLens™ and Unity. The results of the study proved, beyond reasonable doubt, that the solution would have the ability of observing the spatial mapping to recognize the surface of ceiling, walls, table and floor. Moreover, the team realized that the SDK provides the developers with the ability to easily build Gaze-Gesture-Voice (GGV) input as controlling methods.

The feasibility study helped the team discover software such as MeshLab that can be used to visualize meshes or point-cloud data. MeshLab is essential for debugging the PRPU as it implements most of the algorithms available in

the Point-Cloud Library. The team also realized the utility of libraries such as VCG and CGAL through the course of the feasibility study. These libraries as mentioned in 4.4 are essential for the reconstruction of the spatial mesh.

8.2 Spatial Reconstruction

The team has produced successful results with spatial mapping through the HoloLens™ Debugging Console. As earlier stated, the spatial mapping information is stored as triangular meshes, it is categorized into major groups such as walls, floors and ceilings. The HoloLens™ uses the depth camera to perceive the geometry of the surroundings within about 3.1 meters. By employing a mesh caching mechanism, the HoloLens™ stores, updates and discards the information of spatial surfaces for a whole room. This feature will prove essential in the future development of the HLA .

8.2.1 Current Status

The PRPU utilizes an algorithmic design currently being developed by the team. There are three primary open source libraries which the team is utilizing: PCL, VCG and CGAL. The input to the team's algorithm is a mesh containing information including vertices, faces and vertex normals. The algorithm performs reconstruction in two separate streams, reconstructing the interior separately from the exterior hull.

When the team performed spatial mapping of their laboratory with an approximate area of $250m^2$ the device showed no significant decline in performance of mapping surroundings. These results suggests that the application should have no performance bottlenecks while mapping indoor environments. On the other hand, the results also showed that the device has limited capacity to process complex surfaces such as curves and creases. The reconstructed holograms will be limited to simple surfaces like tables, walls and floors.

8.3 Next Steps

The team has completed the feasibility study and determined the scope of the project. It is now focusing on designing the spatial mapping and object processing features of the solution as seen in Table 1. During the interim presentation,

Schedule	Task	Status
October - November	Feasibility Study Determining the Scope of the Project	Completed
December - Mid February	Spatial Mapping and Object Processing	In Progress
February - March	Hologram Visualization	Planned
March	Web Application Export to VR	Planned
April	Project Refinement and Optimization	Planned

Table 1: Project Schedule

the team showed a demo HLA which has the capabilities of scanning a spacious room, and displayed the processed result on a development machine.

Once the Spatial Reconstruction phase is complete (estimated Week 1, February), the focus will shift to developing the visualization features including the web application. Refinements to the processing algorithms as well as optimization, mainly on CPU and GPU efficiency, will follow.

Finally, the team will perform extensive pre-release testing to ensure final quality of the production.

9 Limitations and Difficulties

Despite the fact that the HoloLens™ is the most advanced AR device with futuristic depth sensing capabilities, the limited number of triangular meshes make it difficult to deal with complex surfaces. The team is investigating this limitation via the HLA, but understands the implications of a lower resolution of triangles in the mesh. This limits the types of surfaces that the project will be able to capture. The team has limited the current development of the project to handle only flat surfaces that are reasonably apart. The team has not yet considered algorithms that might be utilized to detect curved surfaces or multi-tiered surfaces such as a raised portion of the floor or a staircase. The limited number of triangles available within the mesh severely limit capability to detect elements interior to the model because such elements require considerably more detail to accurately map. However, the team is confident that future iterations of the device will come with further improvements to the technology enabling

the solution to capture complicated objects.

Another limitation of the project is that extensive movements while wearing the device can lead to disorientation and lossy surface stitching. There is scope of reducing these issues by warning users to move slowly especially during scanning activities.

Furthermore, the depth sensing capabilities of the device are limited to only 3.1 meters. This reduces the scope of spatial data scanning as it become difficult to scan indoor areas with high ceilings. However, the users will be able to scan objects within a radius of 3.1 meters to t which should be sufficient for general use cases.

10 Conclusion

The team is developing a new technique for mapping and reconstructing indoor spaces through a futuristic device, the Microsoft HoloLens™. The team's progress has been in synchronization with the schedule, such that the spatial reconstruction phase is near completion. The application has immense benefits for designers and engineers as it gives them a convenient new way to scan and visualize environments as holograms.

This report describes the implementations of the Spatial Reconstruction Solution, a HLA coupled with a PRPU and web application that will be used to create holographic models of existing indoor environments. Early results suggest that there is a great scope for this application and the choice of hardware is fitting for the team's use case. The HoloLens™ adequately identifies walls, floors, tables, and other surfaces in real time paving the way for the team to perform spatial reconstruction.

However, the quality of the spatial mapping is limited to flat surfaces. This limits certain applications of the solution especially when used to map complex surfaces such as curtains and furniture. These limitations would not deter the team as it focuses more on the processing of the spatial mapping using algorithms that perform the three tasks of hole filling, dynamic object filtering and plane identification. Algorithms to overcome these challenges such as Voxel Grid, Plane Segmentation through RANSAC, and Advancing Front Algorithm are readily available in libraries such as PCL.

After the completion of the current phase, the team would focus on devel-

oping the visualization features. Once integrated, the final application would be a holistic demo on its own. By incorporating a web portal, the team adds interesting features to the project that would improve the user experience and increase the use cases for the application. The team is confident that the project would be completed by April 2018.

References

[1] C Bradford Barber, David P Dobkin, and Hannu Huhdanpaa. *The quickhull algorithm for convex hull*. Tech. rep. Technical Report GCG53, The Geometry Center, MN, 1993.

[2] David Cohen-Steiner and Frank Da. *A Greedy Delaunay Based Surface Reconstruction Algorithm*. Tech. rep. RR-4564. INRIA, Sept. 2002. URL: https://hal.inria.fr/inria-00072024.

[3] Microsoft Corporation. *HoloLens hardware details*. URL: https://developer.microsoft.com/en-us/windows/mixed-reality/hololens_hardware_details.

[4] Microsoft Corporation. *HoloLens Use Gestures*. URL: https://support.microsoft.com/en-us/help/12644/hololens-use-gestures.

[5] Microsoft Corporation. *Installation checklist for HoloLens*. URL: https://developer.microsoft.com/en-us/windows/mixed-reality/install_the_tools.

[6] Microsoft Corporation. *KinectFusion Project Page*. URL: https://www.microsoft.com/en-us/research/project/kinectfusion-project-page/.

[7] Microsoft Corporation. *Microsoft HoloLens*. URL: https://www.microsoft.com/en-us/hololens.

[8] Microsoft Corporation. *MixedRealityToolkit - Unity*. URL: https://github.com/Microsoft/MixedRealityToolkit-Unity.

[9] Microsoft Corporation. *Spatial Mapping Design*. URL: https://developer.microsoft.com/en-us/windows/mixed-reality/spatial_mapping_design.

[10] Microsoft Corporation. *Using the Windows Mixed Reality simulator*. URL: https://developer.microsoft.com/en-us/windows/mixed-reality/using_the_windows_mixed_reality_simulator.

[11] Leica Geosystems. *Leica ScanStation P40 / P30 - High Definition 3D Laser Scanning Solution*. URL: http://leica-geosystems.com/products/laser-scanners/scanners/leica-scanstation-p40--p30.

[12] A. Jamali et al. “3D Indoor Building Environment Reconstruction using Least Square Adjustment, Polynomial Kernel, Interval Analysis and Homotopy Continuation”. In: *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* XLII-2/W1 (2016), pp. 103–113. DOI: 10.5194/isprs-archives-XLII-2-W1-103-2016. URL: https://www.int-arch-photogramm-remote-sens-spatial-inf-sci.net/XLII-2-W1/103/2016/.

[13] Michael Kazhdan and Hugues Hoppe. “Screened poisson surface reconstruction”. In: *ACM Transactions on Graphics (TOG)* 32.3 (2013), p. 29.

[14] Microsoft Corporation Developers Network. *Kinect Fusion*. URL: https://msdn.microsoft.com/en-us/library/dn188670.aspx.

[15] Radu Bogdan Rusu and Steve Cousins. “3D is here: Point Cloud Library (PCL)”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. Shanghai, China, May 2011.

[16] Trimble. *LaserAce 1000 Rangefinder*. URL: http://www.trimble.com/support_trl.aspx?Nav=Collection-77546&pt=LaserAce%201000%20Rangefinder.

[17] The University of Utah. *Object Files (.obj)*. URL: http://www.cs.utah.edu/~boulos/cs3505/obj_spec.pdf.

[18] Michael Ying Yang and Wolfgang Förstner. “Plane detection in point cloud data”. In: 2010.