Instructions

Endong Sun
under supervision of Prof Krystian Mikolajczyk

A Thesis submitted in fulfilment of requirements for the degree of
Master of Science
MSc Applied Machine Learning
of Imperial College London

Department of Electrical and Electronic Engineering
Imperial College London
September 4, 2022

Contents

List of Figures

List of Tables

Appendix A

Hololens2 software: AOAMRTKApp

A.1 Hololens AOAMRTK Application Development

A.1.1 Prerequisite

Programming Language: C#

Mixed Reality applications development in Hololens 2 device is based on C# which is one of the most popular development languages in the fields of games and Microsoft software development community, e.g. Unity development and Software development in Windows system etc.

Software Requirements

To achieve the development of the 6DoF pose estimation application in Hololens2, Unity and other software should be installed so that the development environment can be prepared first. These are software requirements as the following shows:

1. Unity 2019.4;

2. Visual Studio 2019 with the Universal Windows Platform development (UWP) workload and the Windows 10 SDK (10.0.18362.0 or newer) component

3. .NET SDK

Hardware Requirements

All the Hololens2 applications are developed in Windows 10 system as Microsoft requests officially. And Hololens2 is equipped with Windows Holographic OS which is a 'flavour' of the Windows 10 system. The Hololens2 device should be up to date and has developer mode enabled.

Useful Github link

Azure Object Anchor (AOA) software development kit (SDK) is the example project published on Github by Microsoft for developers to experience the 6DoF pose estimation of Hololens2. It is composed of model conversion and example applications, which can play a significant role in AOA APP development.

Azure Account registration

Create an Object Anchors account from the Azure Portal. Then save three important pieces of information for the configuration of AOAMRTKApp development and 3D model conversion: a. Account Domain; b. Account ID; c. Primary key.

A.1.2 Development Procedures

1. Meet requirements in the local laptop as Prerequisite shows

2. Clone the AOA GitHub so that the example app can be installed in the laptop

3. Install Mixed Reality Feature Tool to install the "Microsoft Azure Object Anchors" feature package and "Mixed reality toolkit" (MRTK) packages into the Unity project folder of MRTK as FigureA.1 shows.

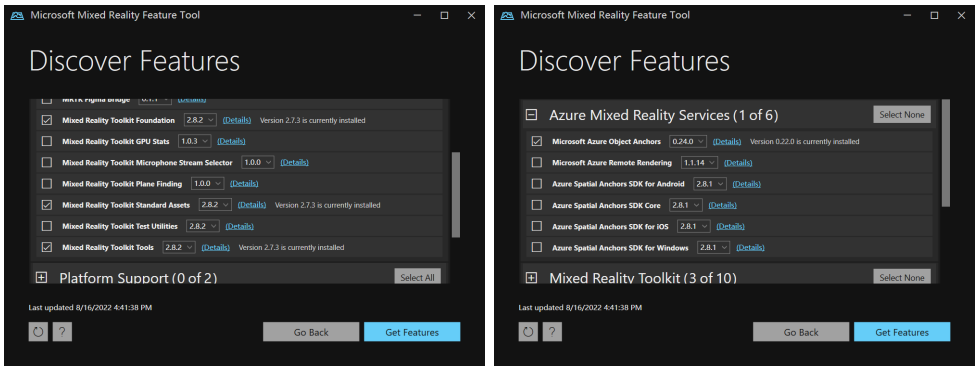

Figure A.1: MRTK packages installation

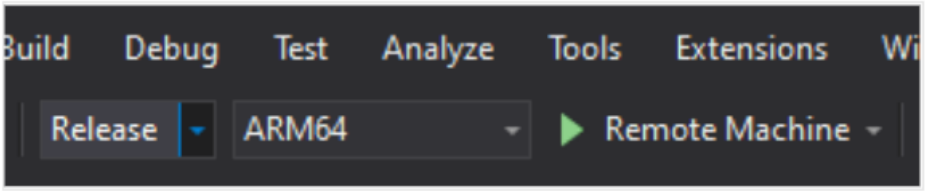

Figure A.2: Configuration of Visual studio

4. Open .sln using Unity 2019.4 (Assets >Open C# project >open `AOAMRTKApp.sln`)

5. After opening, change it as A.2 in visual studio: `Release and ARM64 and Remote Machine`

6. Then right click `AOAMRTKApp(Universal Windows)` >Properties >Configuration Properties >Debugging

7. Change the Machine Name value to the Hololens2 device's IP address and click Apply as Figure A.3 shows.

8. Close the property page. Click **Remote Machine**. The app should start to build and deploy to the remote device. Make sure the Hololens2 device is active.

9. After the Unity splash screen, a message indicates that the Object Observer has been initialized.

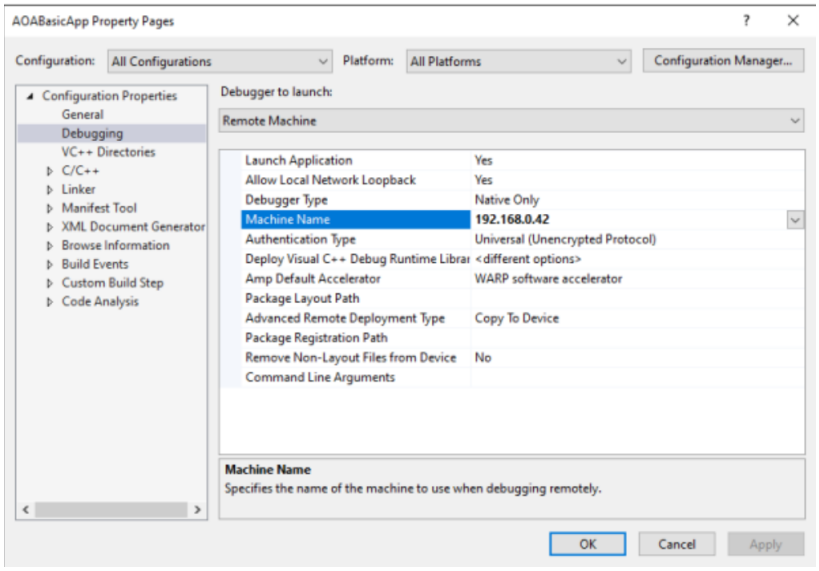

Figure A.3: Configuration of Visual studio

A.2 AOAMRTKApp To Estimate 6DoF Pose of Objects

A.2.1 AOAMRTKApp Modification

After the development of Hololens2 `AOAMRTKApp`, 6DoF poses can be calculated by modifying some technical details in the Unity scripts, which is called `TrackedObject.cs` under the scripts of `MixedReality.AzureObjectAnchors`. The reason to modify is to output 6DoF parameters in the `.dat` file and visualize them in the mixed reality view of Hololens2. Moreover, it is easier for future research if the file can be downloaded to the local machine.

Methods of implementing and extracting `.dat` is to use Unity built-in function which is called `PlayerPrefs` and the scripts. So that the 6DoF Poses will be stored after running the `AOAMRTKApp` by pressing stop search.

A.2.2 AOAMRTKApp User Interface

Figure A.5 illustrates the visualisation of UI interface of `AOAMRTKApp`.

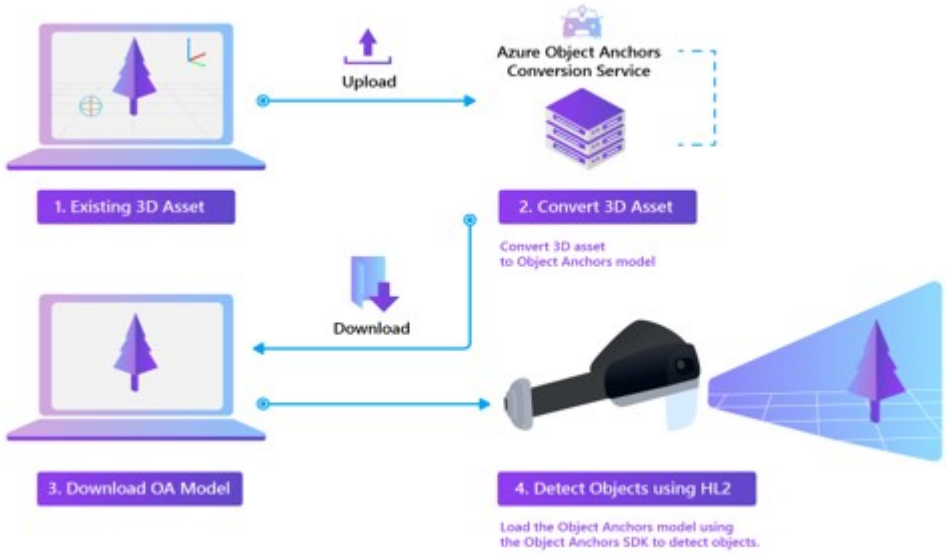

Figure A.4: 6DoF estimation flow chart of Hololens2 AOA APP

Start/Stop Search

Users can press this button to Start/Stop the `AOAMRTKApp` for tracking and estimating the physical object by using a 3D mesh model.

Spatial Mapping

Spacial mapping can describe the surface of the physical world environment as a triangle mesh attached to a world-locked spatial coordinate system.

Tracker Setting

1. Active Observation Mode: Visualize the spacial mapping during tracking.
2. AnchorPlacement Mode: There are 2 modes, Single and Multi, which can output one or more Anchors during tracking the physical object.
3. HighAccuracy Mode: To get higher accuracy will change the mode to high latency.

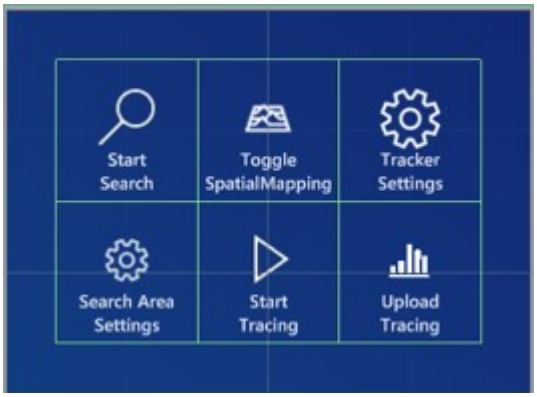

Figure A.5: 6DoF estimation flow chart of Hololens2 AOA APP

Search Area Setting

1. Lock area: Lock the bounding box for object pose estimation.
2. Adjust area: Track and align the object with the bounding box automatically.

A.2.3 Bounding Box

Bounding Box is the search area which can be defined by users. Users can change the position and the size of the box. It is built to reduce the spacial mapping area in order to locate the position of the physical object roughly by users themselves and also reduce the computing power.

A.3 6DoF Pose Estimation Process

6DoF Pose Estimation Process can be displayed in Figure A.4. The following sections are the user guide for estimating poses.

A.3.1 3D Asset Creating

There are twenty-one 3D models chosen from the YCB-V dataset. However, the original 3D models cannot be converted because of the unit of measurement. 360 Fusion is an effective tool to rescale the unit of measurement to 'm' from the original 'mm' unit.

A.3.2 Convert 3D Asset

Azure Object Anchors SDK has 2 functions which are firstly useful for `AOAMRTKApp` development. The other function is implemented based on the visual studio configuration and Object Anchors cloud service, which can convert the 3D model into `.ou` format.

Upload the 3D object with the format of `.ply` and `.obj` to the Azure cloud service by filling in the Account Domain, Account ID and Primary key inside the script of `Configuration.cs`. Then set the field of InputAssetPath to the path that stores 3D models, change the unit of measurement to 'm' and Gravity to '0, -1, 0' according to the visualisation of the 3D model in `AOAMRTKApp`. In this step, each model conversion will cost around 3 minutes. Finally, the 3D asset will be converted as `.ou` from `.obj`, so that `AOAMRTKApp` can use these 3D models.

A.3.3 Insert `.ou` Models

The `.ou` 3D models are downloaded automatically after 3D model conversion, which can now be dragged into the Hololens2 3D model document after connecting Hololens2 and the local computer through USB.

A.3.4 Presetting

Before starting the APP, remove all holograms which could ensure objects can be properly detected in their current positions in case they were recently moved.

Pre-scan the object by wearing Hololens and walking around the object from multiple angles and distances to get better results for dark and highly reflective objects.

A.3.5 APP Startup

The bounding box will appear in the view of Hololens2 after running the `AOAMRTKApp` in Hololens2. The bounding box can be moved and scaled by hand. For instance, pinching the thumb and forefinger can move the bounding box. Ensure the physical object is inside the box after moving, and the bounding box should be locked. Be mentioned, users cannot

move their body location and object's location during the startup of the App. Because the camera will localize itself in world coordinates at the start of the App running.

A.3.6 User Interface Setting

Then set modes by pressing the buttons on the UI. The high accuracy mode can guarantee more precise results although it gives a significant delay to the App. Single Anchor placement can only estimate one object's 6DoF poses. Active tracking mode can scan the 3D model by its built-in function which could detect physical objects more accurately.

A.4 Result of Pose Estimation Using Hololens2

A.4.1 Results Visualisation

Finally, the 6D poses composed of rotation and translation will be visualised in the view of Hololens2 as shown in Figure A.6. The yellow triangle mesh model is the `.ou` model which has been aligned with the physical Object after `AOAMRTKApp`. The other result in Figure A.7 can be downloaded from the windows portal of Hololens2. The position and rotation were displayed in the document, where rotation is in the form of a quaternion. The path to download is under `AppData/Local/Packages/ProductPackageId/LocalState/playerprefs.dat`.

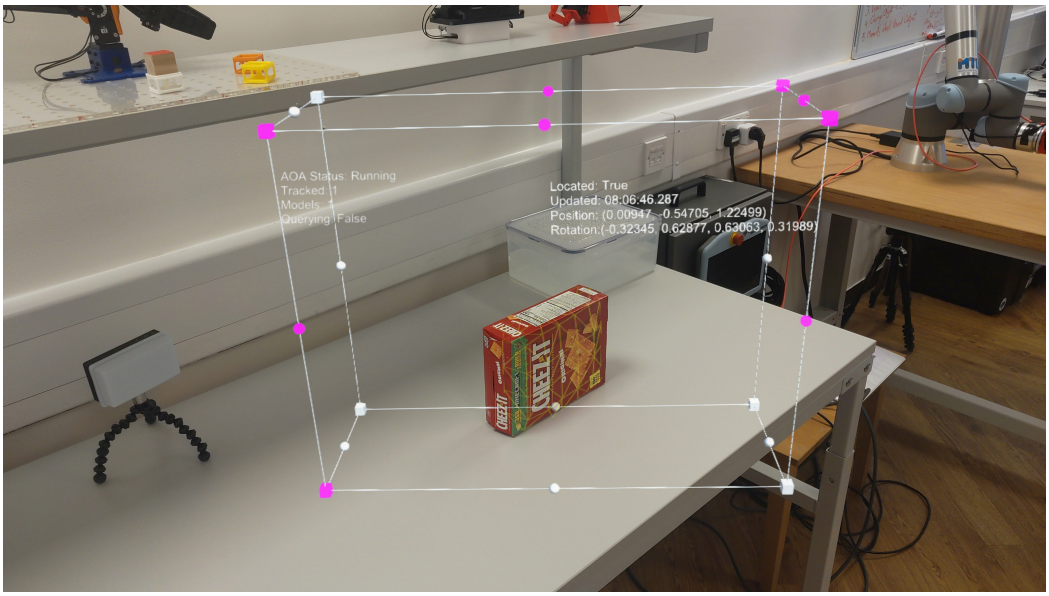

Figure A.6: 6DoF estimation visualisation of Crackerbox using Hololens2

playerprefs.dat

```
1  Position(0.00743, -0.54640, 1.22548)
2  Rotation(-0.31848, 0.63060, 0.63511, 0.31232)
```

Figure A.7: 6DoF estimation parameters of Crackerbox using Hololens2

Appendix B

Hololens2 software: StreamRecorderApp

B.1 Data Collection Tool Development

Hololens2ForCV is the most useful GitHub link for Hololens2 development. It was launched at European Conference on Computer Vision (ECCV) 2020 online tutorial to train how to access all the raw streams on the device [**?**]. There are four example applications which are Calibration Visualization, Camera With CV And Calibration, Sensor Visualization and Stream Recorder, where the Stream Recorder is the APP to be developed in this chapter.

The steps to develop the Stream Recorder (`StreamRecorderApp`) are similar to `AOAMRTKApp` in Appendix A.

B.1.1 Prerequisite

Software Requirement

Visual Studio 2019.4 with installed C++, UWP, and Windows SDK components are required for the laptop.

Figure B.1: Research Mode Setting

Hardware Requirement

Windows 10 system with the desktop version is necessary. For Hololens2, the system needs to be updated greater than 1904.1364.

B.1.2 Procedures of App development

Research Mode On

`StreamRecorderApp` is in the research mode, which means developers must turn it on. The first step is to turn on the developer mode and device portal on the Hololens2 device. Then the research mode can be turned on (Fig B.1) through the device portal which is displayed by the local WLAN link in the Microsoft Edge browser.

Research Mode Setup

`perceptionSensorsExperimental` is the capability that should be added to the `Package.appxmanifest` like the Fig B.1 shows.

StreamRecorderApp Development

1. Meet requirements in the local laptop as Prerequisite shows. Setup Research Mode on Hololens2.

2. Clone the Hololens2ForCV so that the example app can be installed in the laptop

3. Open .sln using Unity 2019.4 (StreamRecorderApp in GitHub folder>open `StreamRecorderApp.sln`)

4. After opening, change it as A.2 in visual studio: `Release and ARM64 and Remote Machine`

5. Then right click `StreamRecorderApp` >Properties >Configuration Properties >Debugging

6. Change the Machine Name value to the Hololens2 device's IP address and click Apply as Figure A.3 shows.

7. Close the property page. Click **Remote Machine**. The app should start to build and deploy to the remote device. Make sure the Hololens2 device is active.

8. After the Unity splash screen, a message indicates that the Object Observer has been initialized.

B.2 Results of Dataset

Running the `StreamRecorderApp` can extract camera streams and save them in the local machine to post-process these data. Hololens2 camera types conclude Grey scale Visible Light Cameras (VLC), Depth Camera and RGB camera: photo video camera (PV).

B.2.1 Post Processing

Post-processing python scripts are used to unzip the RGB frames compressed file to RGB images. Moreover, the point cloud of the current of view can be established through the streams of depth camera of Hololens2.

Dataset Visualisation

Fig B.2 shows each file containing steams and parameters, where `PV` has each frame of the RGB camera, `pv.txt` extracts the intrinsic parameters and camera pose which is the

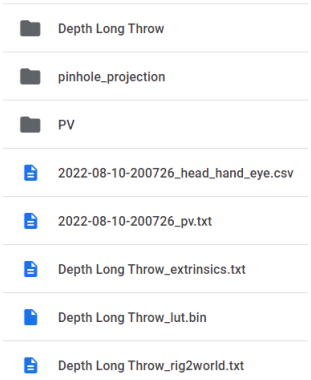

Figure B.2: Data files visualisation

most important file for evaluation experiment in chapter **??**. `pinhole_projection` stores RGB-D images with their camera parameters, which can be converted into point clouds.

B.2.2 Results Descriptions

Results of RGB images and their camera parameters are most significant for the evaluation of the Hololens2 estimation method. These are descriptions of them after running the `StreamRecorderApp` and post-processing.

Photo Video (PV) camera file

RGB images are captured during running the APP for 3-4 seconds, which means roughly 100 frames can be extracted from the PV file. Each image has an image size of 760×428 which is corresponding to width and height, respectively. The result of capturing the image during estimating the 6DoF pose of a cracker box is shown in FigB.3.

`pv.txt` parameters

`pv.txt` file provide all the essential camera parameters such as principle points, focal length, image size and camera pose. Fig B.4 illustrates each camera parameter, where the principle points(ox, oy) inside of the red border and image size within the yellow border are fixed for each frame. The focal length with the green border and camera pose underlined with the blue colour will change along each frame during the stream record.

Figure B.3: RGB image of capturing the cracker box

```
373.25,201.301,760,428
1330322740996543 58,587.207,586.645,0.999603,0.0140571,0.0245431,0.0150394,-
0.0246454,0.858637,0.511994,0.0327601,-0.0138754,-0.512393,0.85864,-0.174719,0,0,0,1
1330322740999875 56,587.732,587.151,0.999592,0.0145853,0.0246643,0.0152123,-
0.025167,0.858405,0.512358,0.0326744,-0.0136978,-0.512768,0.85842,-0.174945,0,0,0,1
1330322741003207 53,588.083,587.489,0.999587,0.0158342,0.0241112,0.015413,-
0.0259629,0.85809,0.512846,0.0325891,-0.0125679,-0.513258,0.858144,-0.175131,0,0,0,1
1330322741006539 51,588.608,587.995,0.999599,0.0170523,0.0227365,0.015533,-0.02631,0.857733,0.513424,0.0326366,-
0.0107456,-0.513815,0.857836,-0.175379,0,0,0,1
```

Figure B.4: PV camera parameters

The camera pose is composed of a 4×4 matrix, which can be transferred as extrinsic parameters of the camera.