Photocopy and Use Authorization

In presenting this thesis in partial fulfillment of the requirements for an advanced degree at Idaho State University, I agree that the library shall make it freely available for inspection. I further state that permission for extensive copying of my thesis for scholarly purposes may be granted by the Dean of the Graduate School, Dean of my academic division, or by the University Librarian. It is understood that any copying or publication of this thesis for financial gain shall not be allowed without my written permission.

_____

Michael Crump

Author

Sizing Exoskeleton and Prosthetic devices using Augmented Reality: Implementation Perspectives

by

Michael Crump

A thesis

submitted in partial fulfillment

of the requirements for the degree of

Master of Science in the Department of Mechanical Engineering

Idaho State University

August 2023

Committee Approval

To the Graduate Faculty:

The members of the committee appointed to examine the thesis of Michael Crump find it satisfactory and recommend that it be accepted.

Marco P. Schoen, Ph.D.

Major Advisor

Anish Sebastian, Ph.D.

Committee Member

Nancy Devine, Ph.D.

Graduate Faculty Representative

Acknowledgements

I would firstly like to thank my primary advisor, Dr. Marco Schoen, for the opportunities that were afforded to me by him. He has supported and guided me throughout the duration of my research for this project. His guidance about how to keep my focus on tasks that I could accomplish timely was extremely useful in the completion of this project. Our shared excitement of this topic has helped to keep me inspired. I would also like to thank the committee member Dr. Anish Sebastian and the Graduate Faculty Representative Dr. Nancy Devine.

I am very thankful to Jared Cantrell for the use of the HoloLens 2 device for this project. His help with getting me in contact with individuals with experience with this device was of utmost importance in the beginning stages. I am also grateful for the assistance of Jaden Palmer throughout the project. Without his assistance, this would have been even more difficult.

Finally, I am extremely grateful for my mother Janet Wilson, my children, and my uncles for the support and love they have shown me through the duration of this project.

Table of Contents

Table of Figures

Sizing Exoskeleton and Prosthetic devices using Augmented Reality: Implementation Perspectives

Thesis Abstract—Idaho State University (2023)

This thesis details the theory and implementation of 3D models on the augmented reality platform Microsoft HoloLens 2. In recent years, augmented and virtual reality (AR/VR) systems have become more prevalent as aids for individuals with limited mobility or loss of limb. With the use of these platforms, it is seen that improvements have been made to the quality of life for individuals. There is limited research in the use of such platforms to help alleviate some of the time and cost of creating prosthetics or exoskeletons. By utilizing some of the sensors on HoloLens 2, it may be possible to create and perform individual sizing of parts simply by donning the headset and interacting with created 3D models. This project explores the technology for achieving such goals. It reports on software limitation and implementation issues directly related to HoloLens 2 and the programming of virtual and augmented reality environments to accommodate such functionality.

Key Words: Augmented Reality, Exoskeleton, Rehabilitation

1 Introduction

Objective

The objective of this work is to describe the creation of 3D models for use in HoloLens 2. Further, an explanation is provided of the limitations and potential of the utilized software that may be engaged in for the creation of an exoskeleton as applicable to limited mobility patients. The topics discussed show the potential for hardware and software capabilities used to make the rehabilitation process more economical and less time consuming for the patient.

Rehabilitation with AR/VR

In recent years, there have been great strides in using AR/VR as rehabilitation aids. Most of the aid has come in the form of games that help stroke victims regain some of the mobility that is lost. (Phan et al.) As games have been developed to aid in the mobility of patients, the use of VR has become more widespread. It seems that AR has been used more for surgical procedures than for rehabilitation. A review of projects that have used Microsoft HoloLens was conducted in 2021 to help show the capabilities of this hardware. It was found that HoloLens was used for a wide variety of applications including surgical imaging, rehabilitation gaming, industrial training, academic teaching, industrial robotics, and engineering. (Park et al.) It has been seen that the Microsoft HoloLens 1 and 2 can achieve great measurement accuracy that makes visualization in VR lifelike. (Müller et al.) Frantz et al. used Vuforia and HoloLens 2 to help with neurosurgery. A 3D model of the patient was created from imaging data using 3D Slicer. This model was deployed to HoloLens using Unity. This made it possible for the surgeon to view the patient and the 3D model simultaneously to allow for accurate placement of pressure

relieving holes in the patients head. (Frantz et al.) The research that has been done for rehabilitation has had great affects for patients and their mobility, but more work needs to be done to alleviate some of the price and time that it takes to get proper fit for prosthetics and exoskeletons. One of the deciding reasons for choosing AR in this project is the greater benefit of the added ability to see the individual's own arm along with the 3D model of the prosthetic or exoskeleton during sizing.

AR use in prosthetics and exoskeletons

Projects related to prosthetics or exoskeletons in the past have focused on using existing technology and procedures for fitting and sizing the hardware. It seems that little if any research has been done to make this process easier and cheaper for the patients. It has been stated that some of the issues with exoskeleton use for patients seem to arise from a lack of motivation. This may stem from lack of home-based rehabilitation. (Mubin et al.) By making the hardware easier to attain by lowering the cost and time for fitting, more applications can be made readily available to patients. Current exoskeleton designs seem to be bulky in nature and difficult to power. Along with these limitations, they exhibit challenges to proper fit and customization for individual users. (Tran et al.) By creating a design that is easily adaptable to each user this issue may be made less significant. This project uses a simple exoskeleton design solely for visualization purposes in the HoloLens 2 application.

Proposal of work

The proposed work of this thesis is to explore the software and hardware currently available to determine if a solution to the issues of expense and time required for exoskeleton and prosthetic fitting can be reduced. By looking at the Unity software it is believed that a script

can be created to read the information from the HoloLens 2 sensors and relay the information in a way that can be used by SolidWorks® to size a prosthetic device or exoskeleton to any individual. It seems that the HoloLens device has very good capabilities of accurately measuring real-world items, further, the capabilities of SolidWorks® to use Excel documents to change measurement values of a drawing. Together, these should reduce the need for multiple visits by patients for fitting purposes. With the advancements in additive manufacturing the cost of 3D printing prosthetic and exoskeleton part, the cost of these devices would be further lowered.

Outline

This thesis describes the exploration of hardware and software used for creating a holographic AR experience that has the potential to make fitting and pricing of exoskeleton and prosthetic devices more easily attainable to patients.

Section 2 explains the basic startup for HoloLens and how to make it ready for applications. This section also discusses the reason for using this device instead of the others that were available.

Section 3 describes what software is used and what it is for. Further, this section explains the basic setup of the software Unity, Blender, and Visual Studio, and what SolidWorks® is capable of.

Section 4 explains the difference between 2D image recognition and 3D object recognition.

Section 5 details how the different software packages were used for the project specifically.

Section 6 explains what was learned about how 2D and 3D applications differ, and that 3D recognition is difficult and requires in-depth knowledge of Unity and the components within.

Section 7 concludes the thesis and gives suggestions on future work for the project.

2 Experimental Setup

Hardware

The hardware that is available for this project includes HoloLens 1, HoloLens 2, and Oculus Rift. With the Oculus Rift being a VR platform, direct measurements through this platform are not possible, and additional hardware integration would be necessary. Specifications of each of the AR devices allow for a determination of which system would be more suitable to accommodate the sensing and the simulation tasks. The HoloLens 1 device, compared to the HoloLens 2 device, has a smaller storage capacity. For the project, it is estimated that the system should have more than 2GB of RAM available. HoloLens 2 has a better Holographic processing unit as well as the newer Bluetooth LE 5.0 connection. Since HoloLens 2 is available for the project, the remainder of the description pertains to HoloLens 2. To efficiently use the HoloLens devices, proper setup is of great importance in order to make it usable as a platform for new applications. This setup includes putting the device in Developer Mode. This is a setting that must be done during the initial unpacking of the device to allow future developers to add projects to the device. Developer mode is turned on by going into the settings menu on the device. This is done by selecting the All-apps button on the side of the display window upon startup. This will show the apps that are currently loaded onto HoloLens 2. The settings icon is selected to open the Settings window. From this window, the icon “Update

and Security” is selected which opens the next window. From this window, on the left side, is an icon “For Developers.” A visual representation of these instructions can be seen in Figure 1 and Figure 2 respectively.

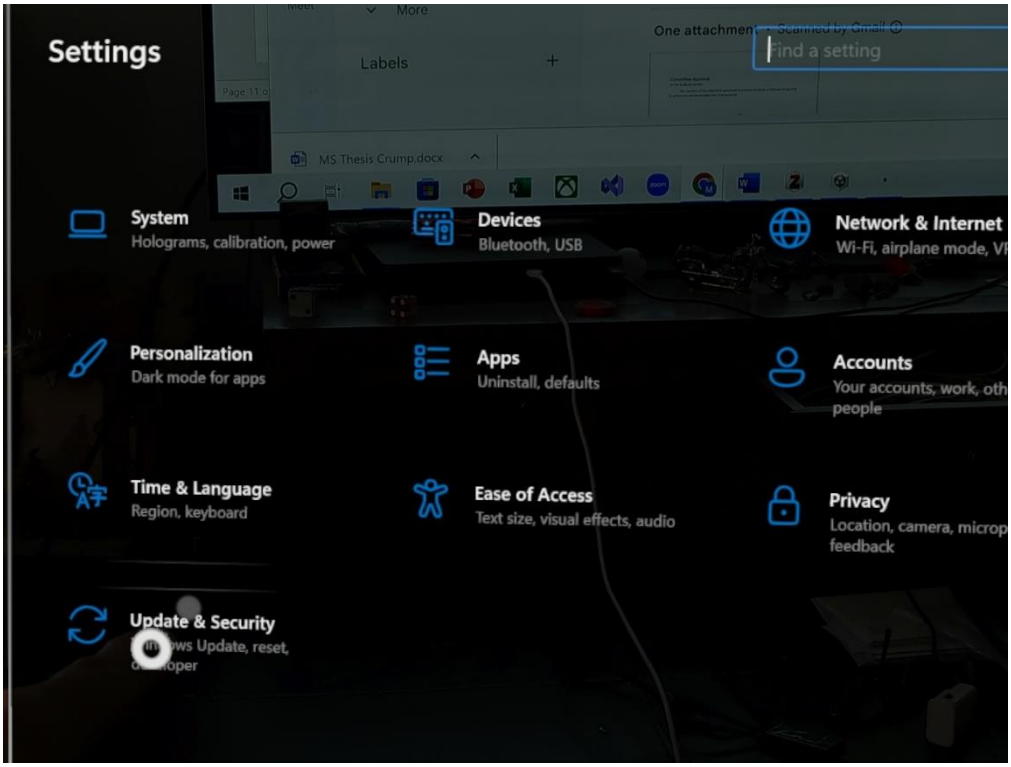

Figure 1. HoloLens 2 Settings Menu

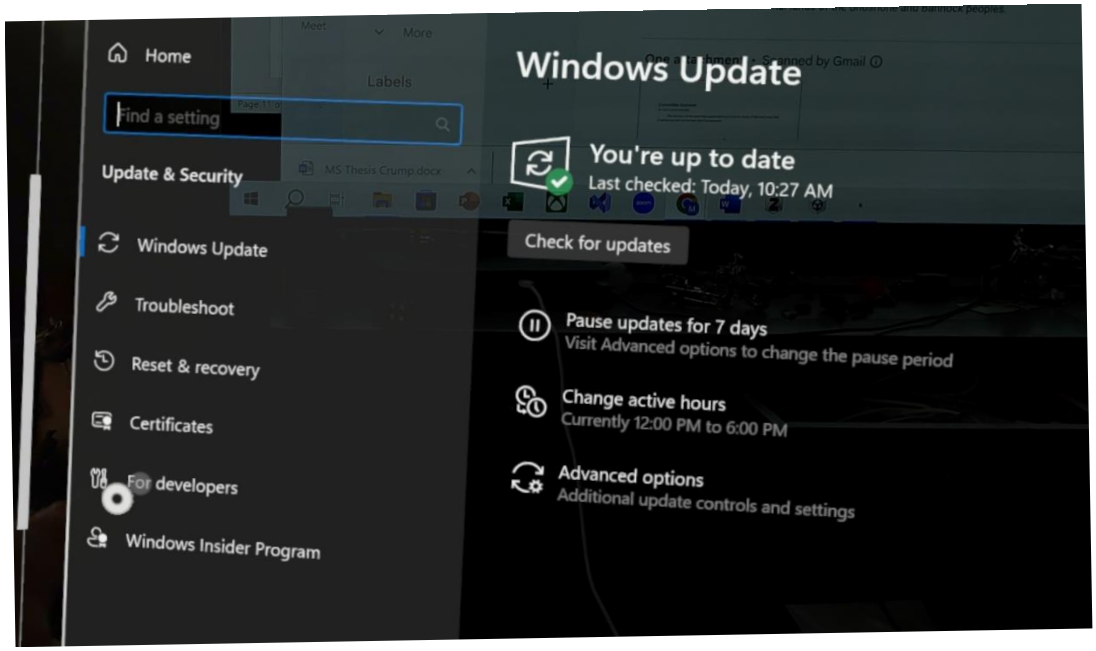

Figure 2. HoloLens 2 For Develpoers Menu

After this is selected, the toggle for “User Developer Features” is turned to the on position. It should be noted that the “Device Discovery” option needs to be turned on to allow for connection to the computer to be detected as shown in Figure 3.

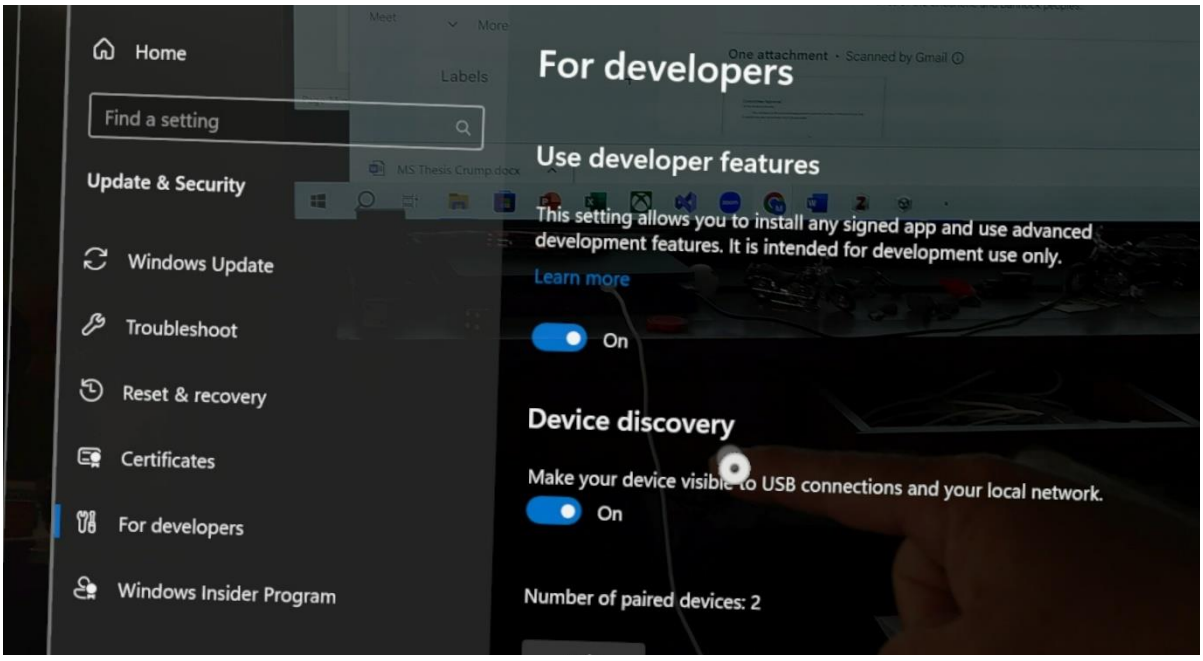

Figure 3. HoloLens 2 Device Discovery Option.

In the case that the HoloLens is turned off, the “Enable USB Data” button needs to be pressed each time the device is turned back on. This is done from the settings window under the Devices window. By selecting the “USB” button on the left of the window, the button is visible at the bottom of the USB window. Each of these settings are shown in Figure 4, Figure 5, and Figure 6 respectively.

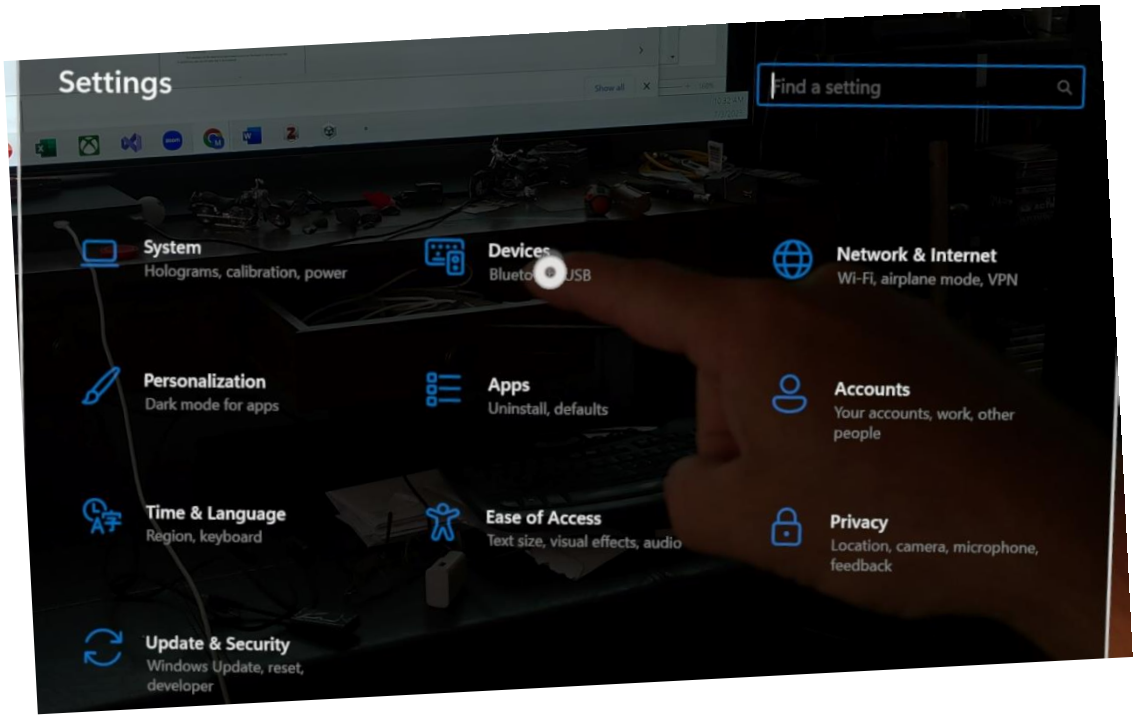

Figure 4. HoloLens 2 Devices Settings.

Figure 5. HoloLens 2 USB Device Option.

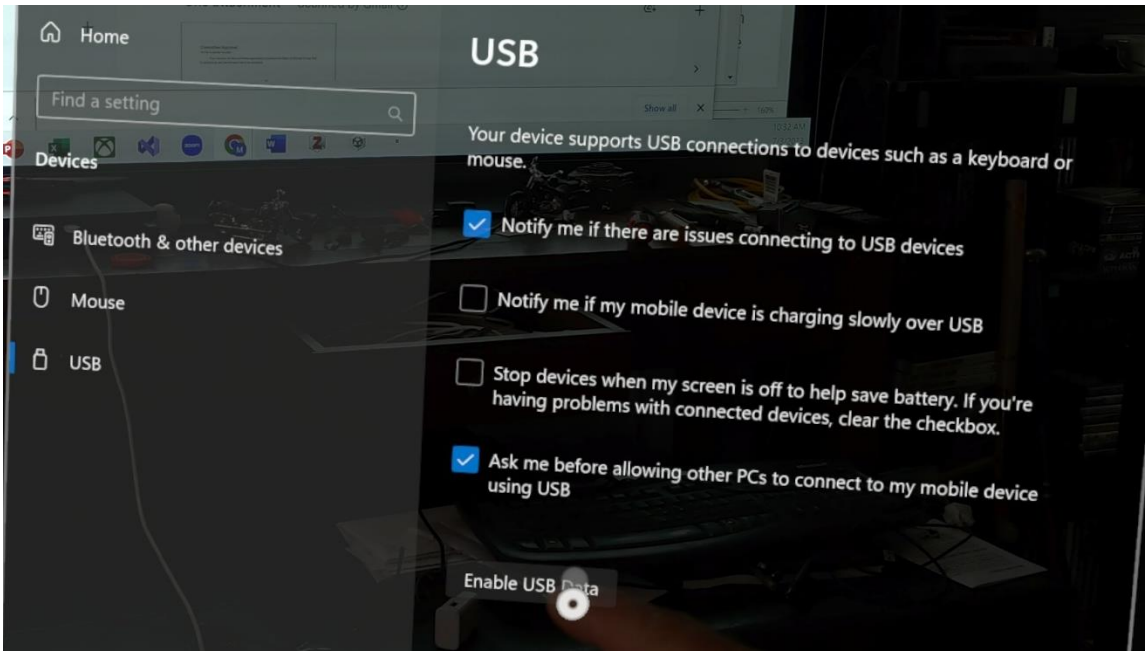

Figure 6. HoloLens 2 USB Data Enable Button.

After these settings are enabled, the addresses of the device are needed for future access. All the available addresses of the device are found by selecting the “Network and Internet” button from the settings window as shown in Figure 7.

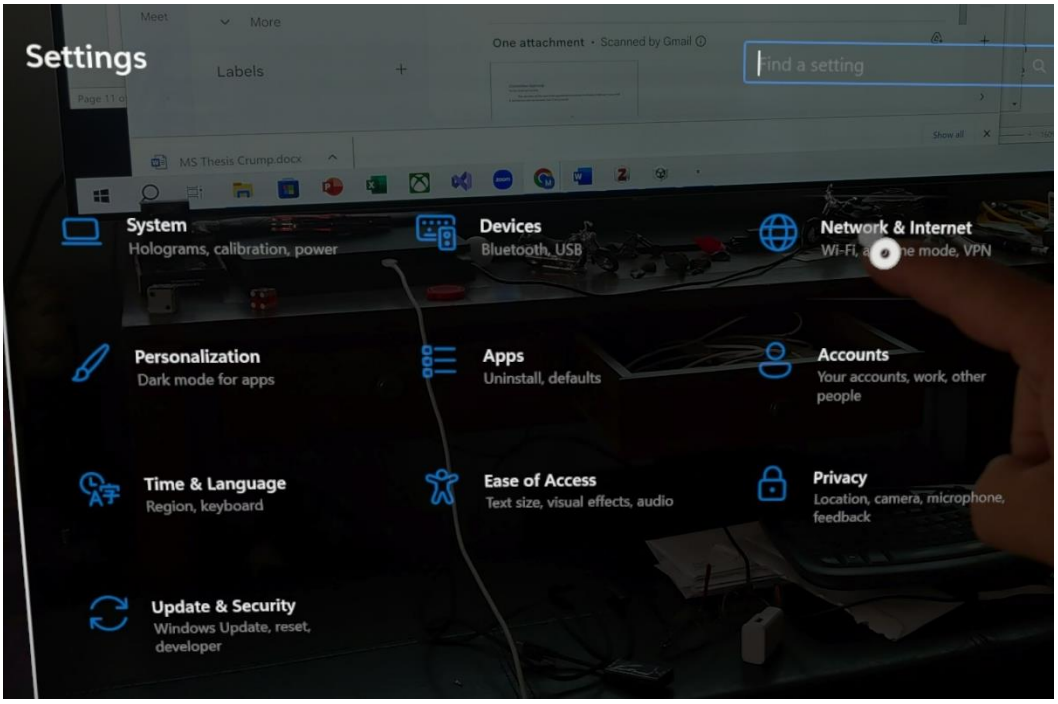

Figure 7. Network and Internet Settings.

To see a list of all the addresses, the blue “View hardware and connections properties” button is selected. The “View hardware and connections properties” button is shown in Figure 8, but it is not seen as blue due to it being highlighted.

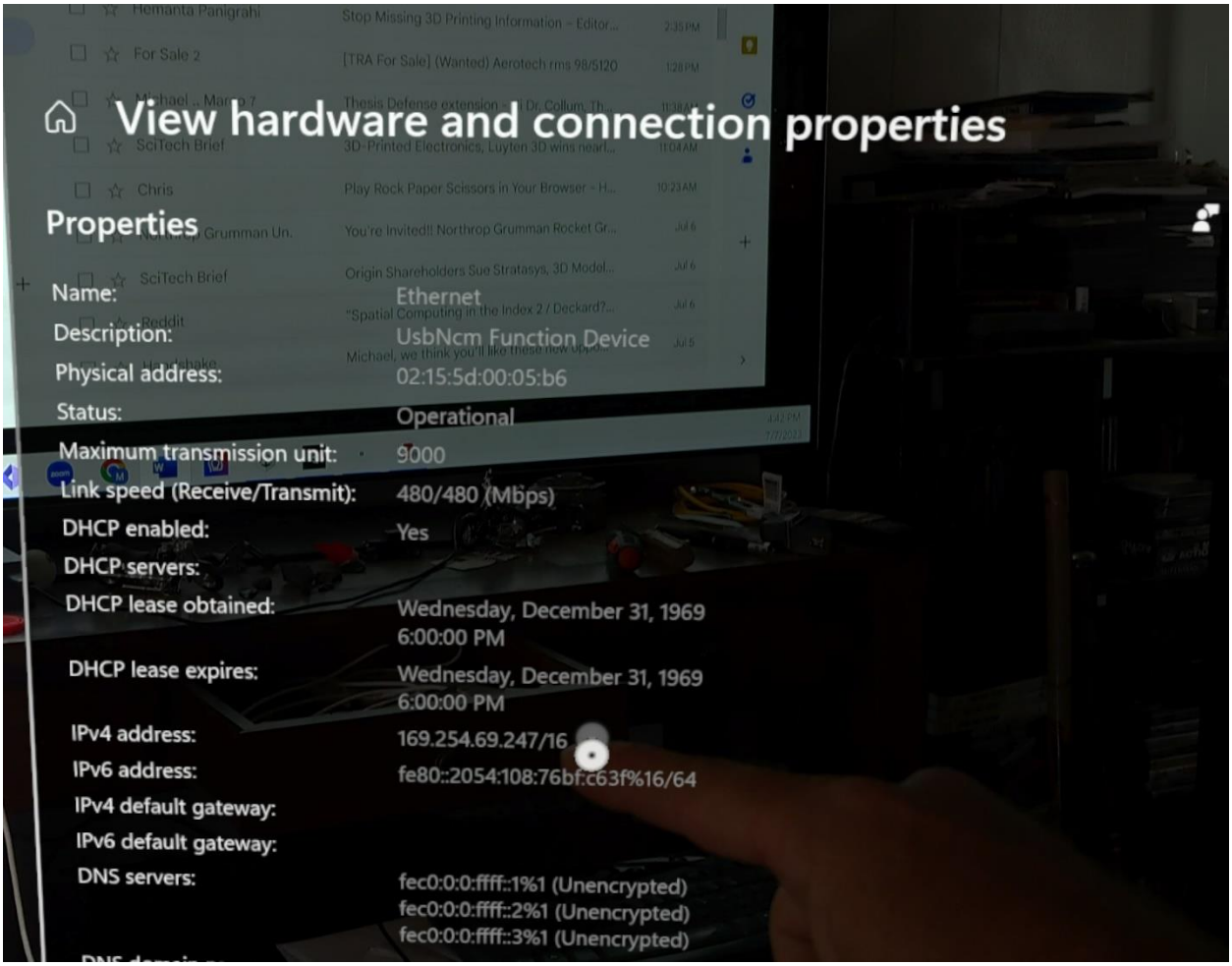

Figure 8. HoloLens 2 IP Address.

It must be noted that if the HoloLens device requires an update, the correct address is not available until after the update has been completed. For all connections for this project the Ethernet UsbNcm Function Device address was used. This address is IPv4 169.254.69.247. This address is used for uploading the Build from Unity from the computer to HoloLens 2 as described in Section 5.

The specifications of the project require that the knowledge of what types of files can be utilized by HoloLens 2. HoloLens requires a certain file type to be able to display an image or 3D object as a hologram. By following the guidelines of the Microsoft learn webpages, supported

file types for 3D objects include; .fbx, .gbl, .gitf, .stl, and .ply. Supported file types for images include; .png, .jpg, .jpeg, .bmp, and .tif. Supported file types that include video along with audio include; .mp4, .mov, .wmv, .asf, .avi, .mkv, and .wav. (*What File Typre Does Hololens Use for 3d Objects - Google Search*) Image and video file types are typically easy to create using a variety of available devices that many people have on hand. Therefore, images compatible with HoloLens 2 are easy to acquire. 3D object file types are somewhat more difficult to acquire. There are websites that have many 3D objects available for free use or for a small fee. A simple Google search for 3D objects will give results. For this project, a basic exoskeleton model was created using SolidWorks® which will be explained in Section 5. After all the necessary software and hardware is acquired, the creation of the application in Unity is examined in the next section.

3 Software setup

Unity

Unity is a platform that is used to create 2D and 3D digital content that can be used in a virtual environment. It has an interface that allows for the creation of interactions of objects and utilizes realistic physics that makes the object perform as if it were a real-world experience. To create the application that would be used in HoloLens, Unity must have certain packages installed. To figure out which packages are needed, tutorials and training are referenced in this text to aid in becoming familiar with these tasks. One of the more helpful resources for accomplishing this is using the Microsoft website. (qianw211, *Unity Development for VR - Mixed Reality*) Other useful resources with guided tutorials can be found at:

- (qianw211, *Choosing a Unity Version and XR Plugin - Mixed Reality*)
- (*About XR SDK for Windows MR | Windows XR Plugin | 3.4.0*)

- (*Augmented Reality (AR) Tutorial for Beginners Using Unity 2022*)
- (*How To Setup the Unity OpenXR Plugin With MRTK for HoloLens 2*)

Care must be taken when selecting the version of Unity because modifications to the software have added to, and created limitations to, the use of some of the components of AR/VR applications. This is explained in more detail in Section 5 under the Scenes heading. From these tutorials, it is found that Vuforia provides some useful capabilities in the realization of 3D augmentation tasks, including helping HoloLens with image and object tracking. This package would take away the burden of computing from the HoloLens to save time for hologram imaging. However, Vuforia may represent an older package that may no longer be needed with the advancements made in Unity. Along with Vuforia being expensive to use, the 3D objects needed for this project would be difficult or impossible to track using Vuforia due to the complexity of movements of the objects. Vuforia requires very limited movement of objects in order for them to be tracked. Also, items to be tracked need to have very limited moving parts. (*Object Recognition | VuforiaLibrary*) The packages that are needed for the given project include the Mixed Reality Tool Kit (MRTK), Windows 10 SDK, and OpenXR. These packages will be explained in more detail in Section 5. Another tool that is needed to make sure the MRTK has all the necessary packages is Mixed Reality Feature Tool that can be downloaded from Microsoft. To make sure all components are added correctly, a YouTube video titled “How to setup the Unity OpenXR Plugin with MRTK for HoloLens 2.” (*How To Setup the Unity OpenXR Plugin With MRTK for HoloLens 2*) provides all the necessary details. Items that are also essential downloads are Universal Windows Platform support and Windows Build support which includes IL2CPP. Unity and HoloLens 2 must work together in this project, so file types are an

essential collaboration point. The file types that both software packages are compatible with are .obj and .fbx. For this project SolidWorks® is used for 3D model creation.

SolidWorks®

SolidWorks® is a solid modeling 3D Computer Aided Drafting (CAD) software program that has many uses. There are multiple add-ins available for purchase that can help with design and engineering applications. There are also multiple analysis tools that will help to understand the strength and durability of a part that has been created. SolidWorks® also allows for directly communicating with catalogs such as Microsoft Excel. Each part created in SolidWorks® can be modified using features that enable Microsoft Excel spreadsheets to change the dimensions of the part depending on how it has been drawn. Future use of this feature may help with fitting prosthetics and exoskeletons to each patient. Each part that is created using SolidWorks® can be converted to multiple different file types that will store different information. It is necessary to find out what information each file type stores to ensure the most information is transferred between different software applications. Through information comparison of the following websites,

- https://en.wikipedia.org/wiki/STL_(file_format) (“STL (File Format)”)
- https://help.solidworks.com/2021/english/SolidWorks/sldworks/c_dxf-dwg_files_dxf_dwg-files.htm (*DXF/DWG Files (*.Dxf, *.Dwg Files) - 2021 - SOLIDWORKS Help*)
- https://help.solidworks.com/2021/English/SolidWorks/sldworks/c_vrml_files.htm
- https://blog.medit.com/medit/the-battle-of-file-formats-stl-vs-obj-vs-ply (Corbett),

it is determined that a .ply file may hold the most relevant information. In particular, it saves color properties, transparencies, surface normals, and textures. (“PLY (File Format)”) However, it is found that when creating a .obj file with SolidWorks®, all the details of .ply files along with

material properties are also saved. For the present work, the 2022 student license version of SolidWorks® is used to create a 3D model of a simple exoskeleton that could be used to understand what components needed to be included in the file that would allow the object to be viewed correctly in a VR environment. An explanation of the creation of the .obj file is explained further in Section 5.

Blender

Blender is a software package that many game creators use to create objects that have textures, colors, and movements. The object can be 2D or 3D and be simple or very complex. Blender is a very powerful tool that is being used to create realistic objects and scenes for AR and VR experiences. In this project, the exoskeleton and 3D arm model are both modified with this software to create the textures and materials needed. When creating an object for a simple VR environment, it must include at least textures and materials to be visible. For more advanced environments, the movement of the object must also be defined. Blender is used to create the movement boundaries of the object. The scope of this project does not include movement, so no explanation is provided. Because of the capabilities of SolidWorks® the exoskeleton object was created by this software alone. Blender is used in this project to help make one of the 3D arm models more realistic. A further explanation of this is given in Section 5. Future work for this project will require the use of the tools that Blender provides such as defining the movement of the exoskeleton.

Visual Studio

Visual Studios is the software that is used as a development tool for AR projects that are deployed to HoloLens 2. It uses C++ language that is compatible with Unity and HoloLens 2.

Visual Studios can be used with HoloLens 2 using different versions. Most of the current tutorials use version 2019, but this project is using version 2022. The options that need to be set in Visual Studios to allow the project to be deployed to the HoloLens 2 are:

- ARM64 needs to be selected from the dropdown menu.
- Release needs to be selected from the dropdown menu.
- Remote Machine needs to be selected from the dropdown menu.
- Machine name must be changed in the Project Settings.

The dropdown menus are located near the top center of the screen when Visual Studio is open. More details about the specific use of this software are provided in Section 5.

4 Recognition

For gaining familiarity with creating scenes using Unity, it is recommended to study tutorials. (*Hololens 2 Development Tutorial - Getting Started (MRTK V2)*) The first tutorials involve creating a 2D scene and adding components that will allow a real-world item to be detected. This is known as recognition. It is important to understand that 2D recognition and 3D recognition have different components that must be understood when creating scenes in Unity. 2D recognition includes a mapping of an image to compare it to the real-world item. To detect an object, Unity identifies the bounding box around the item to be recognized. 3D object recognition needs to include the spatial awareness component for the software to understand where the object is at in the real world. For Unity to detect an object, a 3D model of the object is included in the scene with spatial component added to it. This component is somewhat convoluted and difficult to understand. More information is provided in Section 6.

5 How to use the software

SolidWorks®

SolidWorks® can be used to create a model that includes the texture and material that are required for the model to be viewed in the AR environment. As mentioned earlier, the file types that are needed are either .obj or .gltf. The model created is saved as a part file which is .sldprt. In the past, SolidWorks® had an add-in that was able to export the part file directly to a .obj file. (*Free Solidworks OBJ Exporter v2.0 | SOLIDWORKS Forums*) This add-in was discontinued due to limited user experience around 2018. Due to this discontinuation, different techniques are used to create the object needed for the project. ("Polygon Models") To create the .obj file, the original part file is saved as .wrl file type. The ScanTo3D add-in needs to be enabled for the next steps. Once the Add-in is enabled, the part is opened, from File-Open dropdown, as .wrl and from the file format selector dropdown, "Scan To 3D Mesh Files (…)" is selected. When the file is open in this format, the polygon image of the part can be changed as needed. The final step is to export the file to the desired format, .obj in this case. This is done by selecting "Scan To 3D(*.obj)" under the file format selector from SaveAs. Once the file has been created in the correct format, this object can be opened using the software, Blender. Blender is used to fine tune the object to get it ready for the VR environment, as discussed earlier. The created objects can now be used in Unity to create the scene that is included in the environment.

Unity

For Unity to work correctly as a VR creation platform, certain components and packages need to be loaded and turned on. This project requires Mixed Reality Toolkit (MRTK) and XR

packages and the use of Universal Windows Platform (UWP) to create the application for the environment. Using Microsoft learn pages that are available, will make understanding this process easier. (*Hololens 2 Development Tutorial - Getting Started (MRTK V2)*)

Mixed Reality Toolkit

The MRTK in Unity is an open-source platform that is available from Microsoft that makes creating mixed reality applications easier for developers. There are a variety of scripts, components, and prefabs available that help create immersive and interactive scenarios for different VR platforms. Key features of MRTK include:

- a robust input management system that handles many input devices such as, controllers, hand tracking, gestures, and voice commands,
- spatial mapping and understanding tools that enable the creation of physical objects in the virtual world,
- includes scripts and objects that make virtual object able to be interacted with including grabbing and sizing,
- teleportation and locomotion systems that make navigation in VR more comfortable for the user,
- cross-platform support which makes the created application able to be deployed to different platforms,
- customizable extensions that allow developers to create their own scripts and components or use pre-created scripts from the Unity Asset Store or GitHub.

The MRTK package must be installed from the Unity Asset Store or from a GitHub repository. Import MRTK into a Unity scene by opening the Asset menu and select “Import Package” from the dropdown menu. The MRTK is loaded by opening the Microsoft Mixed Reality Feature tool and pointing the directory to where the project is saved. Select “Discover Features” to start the feature selection tool. Select “Mixed Reality Toolkit” and “Platform Support” from this list. Specifically, the items that must be selected under MRTK are “Mixed Reality Toolkit Foundation” and “Mixed Reality Toolkit Standard Assets” as seen in Figure 9.

Other items that are optional that may be of future use are “Mixed Reality Toolkit Examples” and “Mixed Reality Toolkit Tools.”

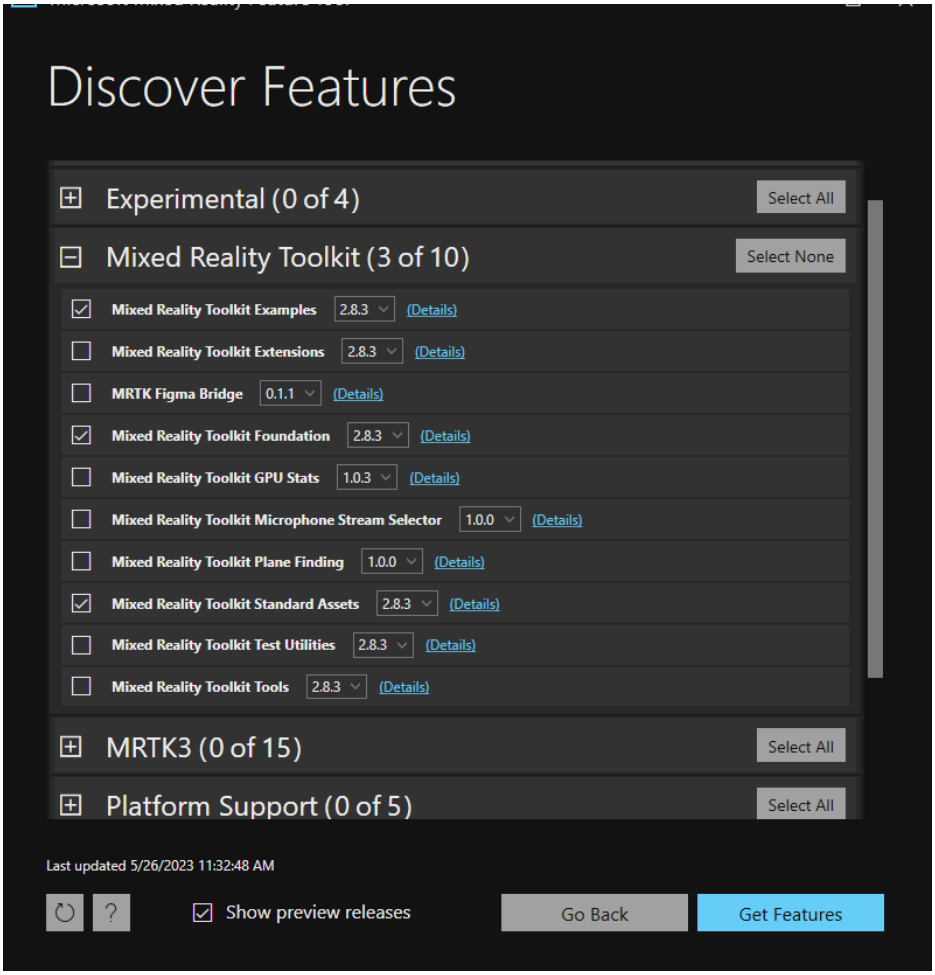

Figure 9. MRTK feature selection.

The item that must be selected under Platform Support is “Mixed Reality OpenXR Plugin.” Optional items are “Mixed Reality Moving Platform SDK” and “Mixed Reality Scene Understanding.” By clicking the “Get Features” button, a list of packages that will be imported is shown. Validation is an option that may be selected on the next window but is not required. However, it is recommended to select this option to make sure that all the items in the Unity environment are compatible with what is selected. Click the “Import” button to show the

complete list of items and select “Approve.” The final window explains that the project items have been loaded. By Exiting the application and clicking on Unity, the update and download is begun. Because Unity has updated certain mixed reality components, there is a warning that prompts for restarting the editor to enable the older items. Selecting “Yes” will load the backend items so MRTK and XR can be used for the project. Following the prompts on the following windows will setup MRTK for use in the application. These updates have created new issues with using Unity for recognition services. Some of the packages used in early scene creations have become incompatible with some of the components of MRTK, so care must be taken when selecting certain components. Additional research is needed to understand new steps that may be taken to allow this project to proceed. During the setup process, open the Build Settings from the File menu and select UWP and change the platform. This will create the MRTK profile that can now be setup from the Project Settings window. Select “XR Plugin Management” and check the box for “OpenXR” and “Microsoft HoloLens feature group.” When these features are selected, yellow triangle icons are created showing that there are some items that are not incorporated yet. To fix this, click on any of the triangles to open the Validation window. At the top of the window is an option to “Fix All” issues. This button is clicked, and all problems are repaired automatically. Closing this window and selecting “skip this step” will open another window asking if the project setting should be updated. The “Apply” button is pressed to confirm the project settings are applied. “Next” is pressed to allow for the Text Mesh Pro (TMP) Essential package to be imported by recommendation. It is unclear what this package consists of as it is not used in this project. By selecting the next obvious buttons, the setup is complete for the MRTK.

Universal Windows Platform

UWP is the essential development platform that is used for developing applications for HoloLens and other Windows devices. By using this platform, deploying to HoloLens 2 is made possible. By using this platform, optimization Intermediate Language to C++ (IL2CPP) is used to enhance the performance by reducing the memory needed in the application. This provides the necessary Application Programming Interfaces (APIs) for good AR experience. An API is a set of rules and tools that allow the interaction of different software applications. They allow developers to interact with software systems without having to understand the full details of that software.

Scenes

Scene creation in Unity is how an application is made. When starting a project with Unity Hub, it is essential to first select the version of Unity that is to be used. For this project Unity version 2020.3.xx and 2019.3.xx were used. The reason for not using the most current version of Unity is that some of the capabilities for HoloLens 2 have been discontinued or changed in the newer versions. These changes have created incompatibilities in some of the components in MRTK and OpenXR. The creation of a scene is done after a new project is created or an existing project is opened. A view of the Unity window can be seen in Figure 10. For understanding of how to create a scene that will recognize a real-world item from an image and then replace that image with a video in an AR environment, the YouTube video labeled “Augmented Reality (AR) tutorial for beginners using Unity 2022” is a good reference. (*Augmented Reality (AR) Tutorial for Beginners Using Unity 2022*)

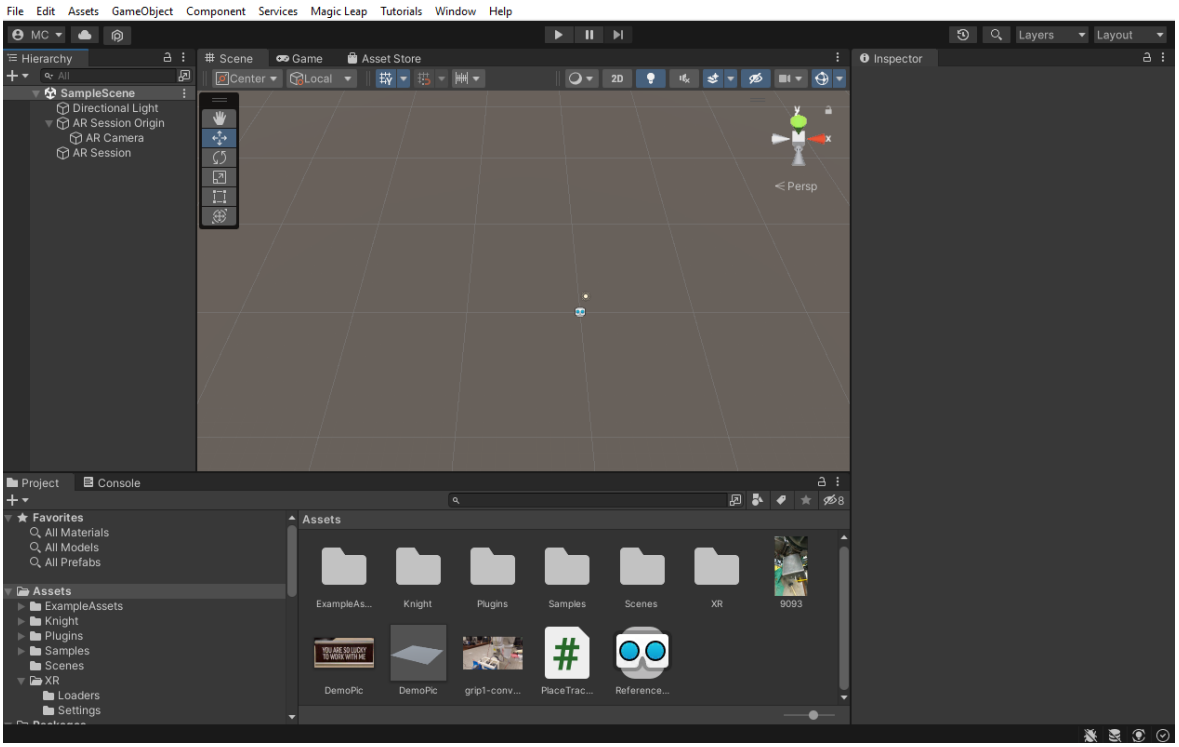

Figure 10. Unity Image Recognition Scene.

An explanation of 2D image recognition is as follows. For image recognition, a picture of the item must be entered into the Unity platform as a target image. This is done by adding an AR script to the AR Session Origin component of the scene. By selecting the AR Session from the hierarchy and looking at the Inspector pane, the AR Tracked Image Manager component is added by searching for it in the search bar as seen in Figure 11 and Figure 12 respectively.

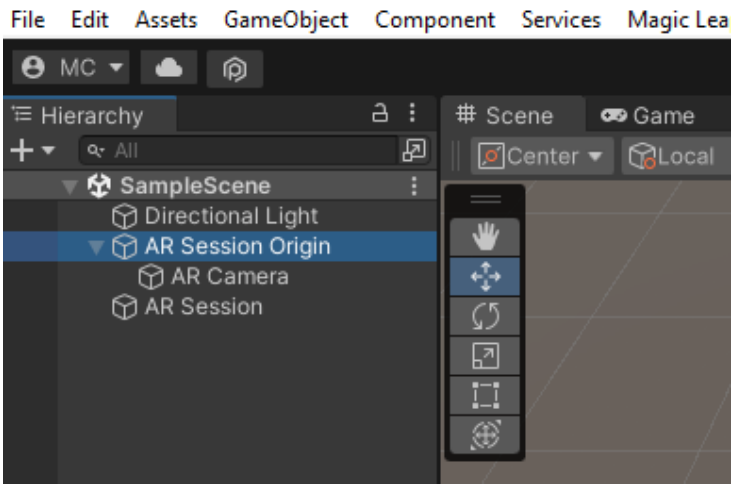

Figure 11. Selecting AR Session Origin.

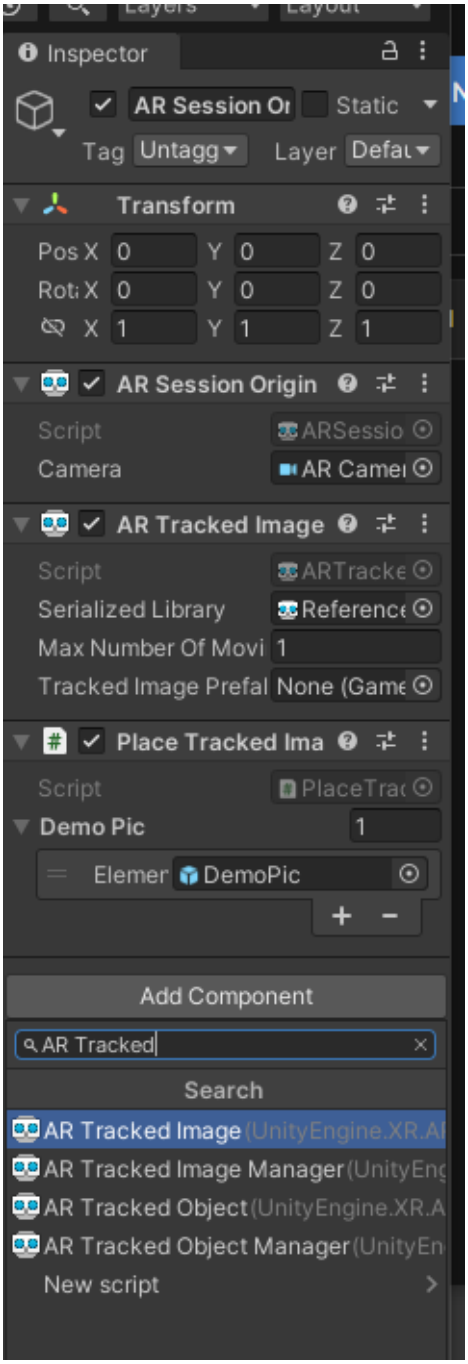

Figure 12. AR Tracked Image component search.

Before the image is placed into the scene, a reference library must be created by selecting "Reference Image Library" from the Create and XR dropdown menu as seen in Figure 13 and Figure 14.

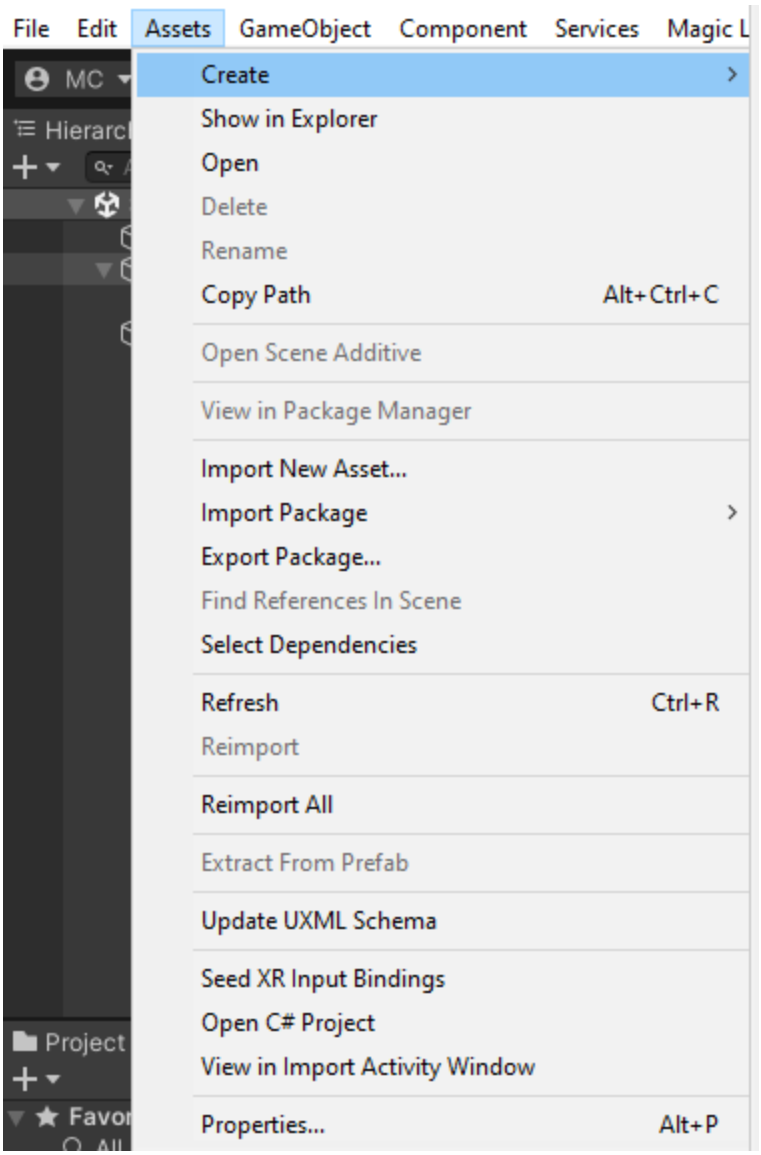

Figure 13. The Create dropdown menu.

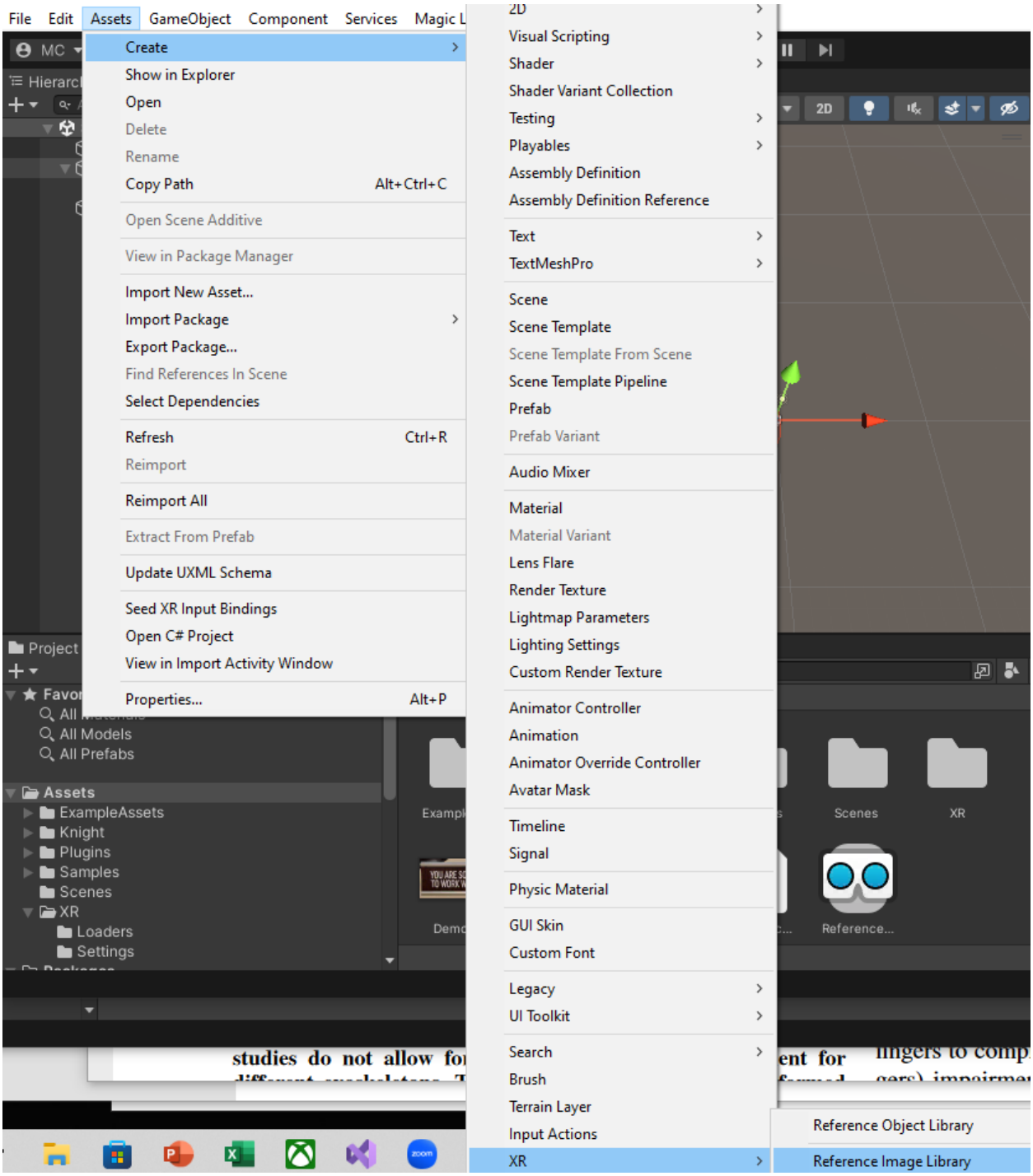

Figure 14. Reference Image Library option.

This creates the Icon for the Image Library as seen in Figure 15.

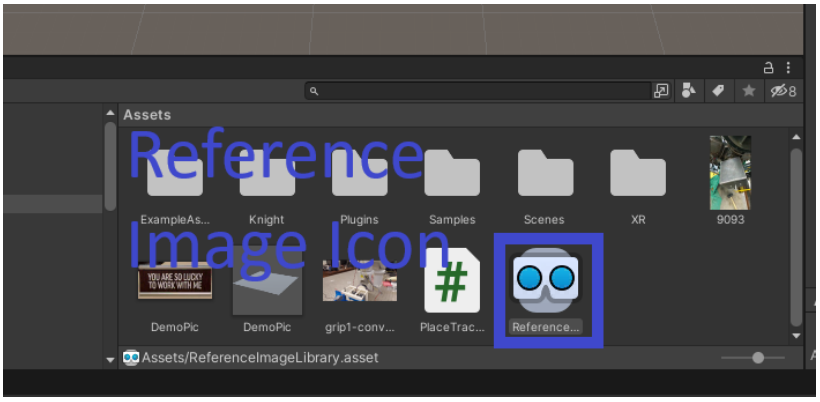

Figure 15. Reference Image Library Icon.

With this icon in the Assets folder of the Unity application, the image can be imported and the settings for it can be modified. It can be seen in Figure 15 that DemoPic and 9093 items have been added. DemoPic is the images that is tracked and 9093 is a video that is over-layed when the image is recognized, respectively. The PlaceTrackedImage script is also seen in this figure. These items will be explained later in this section. By selecting the ReferenceImageLibrary Icon, the DemoPic image is dragged into the small box in the upper left corner of the Inspector window as seen in Figure 16.

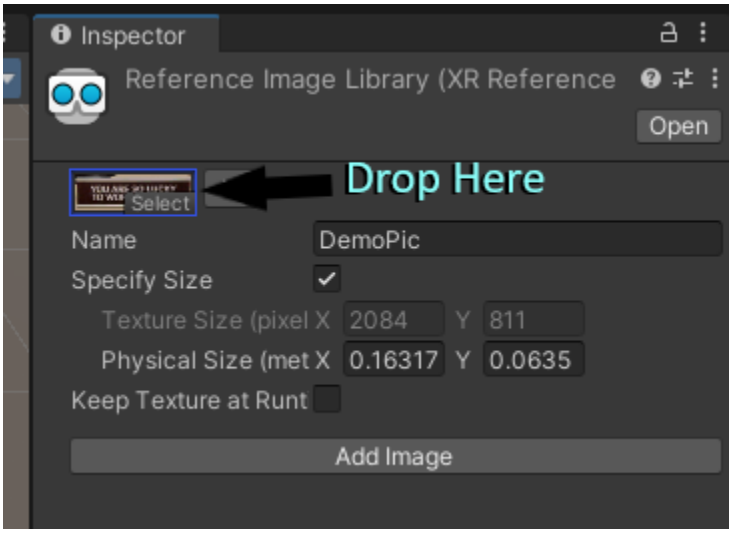

Figure 16. Reference Image placement area.

Also contained in this window is the space to size the image to match the physical size of the real-world item the image represents. The units for Unity are in meters, therefore the size of the object must be inserted as such. This sizing is shown in Figure 17.

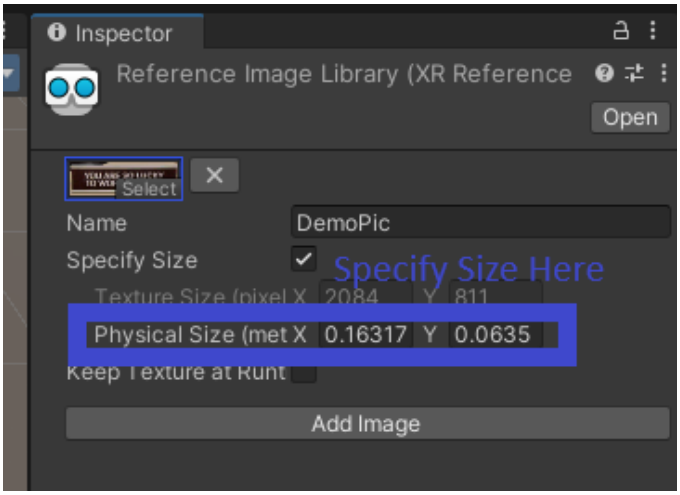

Figure 17. Real world item size.

A script that tells Unity what to do with the image must be created and added to the Image Manager. The actual script is provided in the comments of the video by the author of the YouTube video referenced earlier. From the hierarchy window, select AR Session Origin and look at the Inspector window. Figure 18 shows that the Reference Library icon is dragged from the Assets folder into the AR Tracked Image Manager (script) where the box for Serialized Library is.

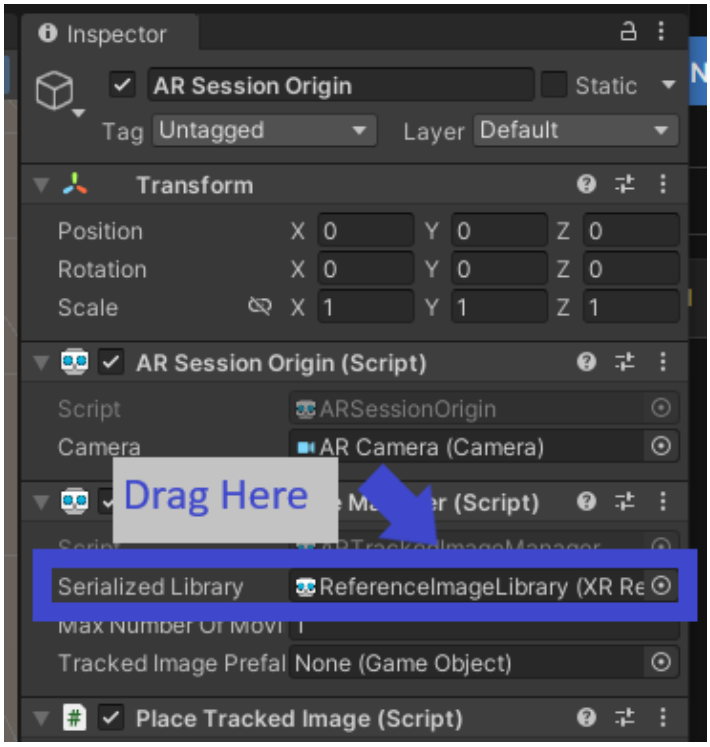

Figure 18. Reference Image Library placement area.

From the same window, the “Add Component” button is clicked, and “New Script” is selected from the dropdown menu to create the script that will tell Unity what to do with the image that is in the Reference Image Library. By double clicking the existing script, Visual Studio is opened showing the generic script. The correct script is copied from the comments section of the YouTube video referenced. The script is pasted into Visual Studio and then saved as “PlaceTrackedImages” and then closed. For this script to work it is noted that the initial code must be carefully inspected to ensure that spelling and capital letters are input correctly. Details of the supplied script are explained in the YouTube video between time stamp 00:15:17-00:24:05. Once the scene is complete, it is ready to be built.

An explanation of 3D object recognition is as follows. This setup is done by following the AR Foundation Object Tracking tutorial from the Unity website. (*AR Tracked Object Manager | AR*

Foundation | 3.0.1) Some of the steps for 3D detection are the same as 2D recognition. However, the components that are added are different. The MRTK package must be loaded into the scene as explained above. Select the MixedRealityPlayspace icon under Hierarchy to add components needed to the scene. Instead of adding the Image Recognition component, the AR Tracked Object component is added to the scene from the search bar as seen in Figure 19.

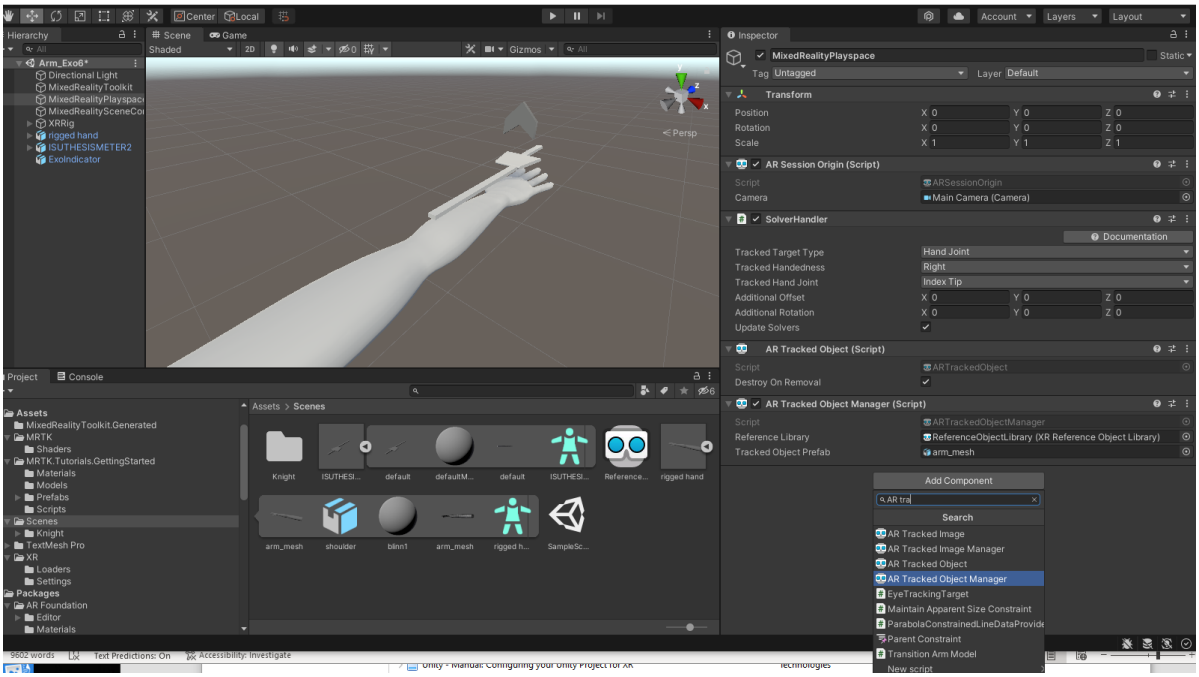

Figure 19. Added components to create object tracking.

Along with the “AR Tracked Object”, other items are also added using the search bar. These components are “AR Tracked Object Manager” and “SolverHandler” as shown in Figure 19. It is shown in figure 19, the “rigged hand” object and “ISUTHESISMETER2” objects have already been imported to the scene. The “rigged hand” is the 3D arm that is downloaded from Sketchfab website that is discussed at the end of this section under the Blender heading, and “ISUTHESISMETER2” is the simple exoskeleton created in SolidWorks®. Importing the items is done by dragging the items from the computer location onto the Scene folder in the project.

The “SolverHander” component is used to tell Unity that object tracking is needed in the scene. For object tracking to know what it is trying to track, the item needs to be added to the AR Tracked Object Manager. Creating the ReferenceObjectLibrary for 3D objects is similar to creating a ReferenceImageLibrary for 2D objects. Select “Reference Object Library” from the Assets>Create>XR menu as seen in Figure 14. Select the Reference from the asset folder and then select the “Add Reference Object” button. Only the name of the object needs to be entered in this field. Figure 20 shows that in this project, the name of the object is “rigged hand.”

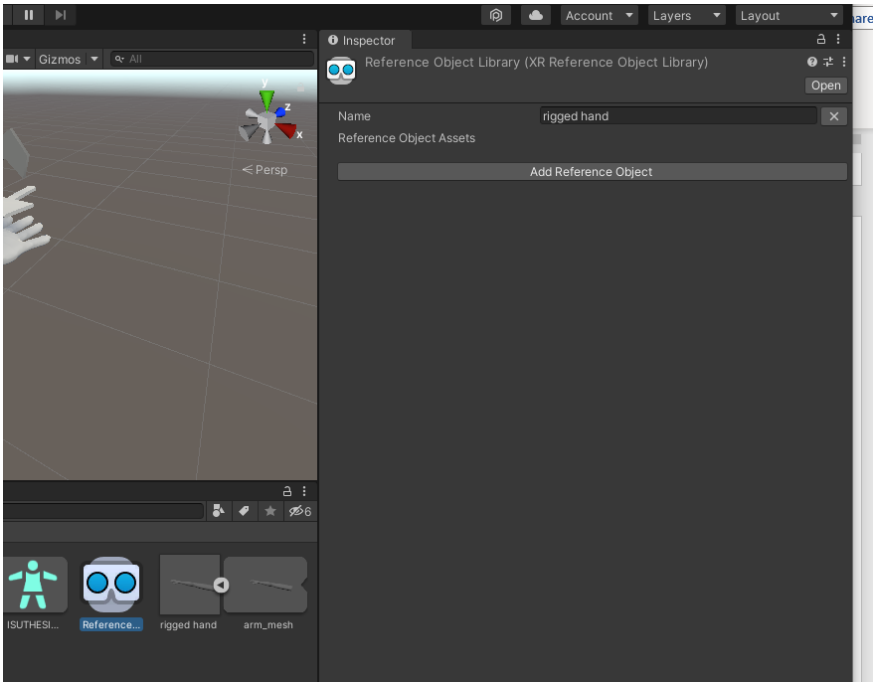

Figure 20. Refence Object Library object name.

Once the library is created, it is dragged into the box for the “Reference Library” as shown in Figure 21.

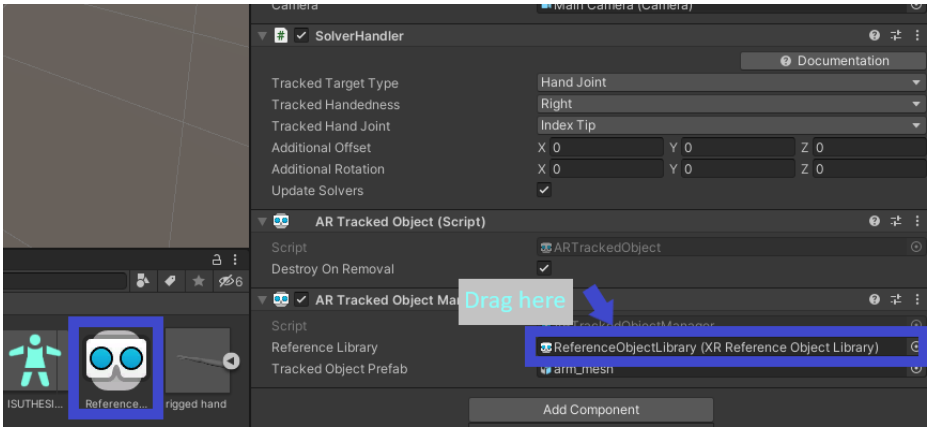

Figure 21. ReferenceObjectLibrary placement.

To include the “Tracked Object Prefab” into object manager, the “arm_mesh” is dragged into the box from the “rigged hand” asset as shown in Figure 22. The arm_mesh is the polygon representation of the 3D arm object that Unity uses as a place holder in the scene.

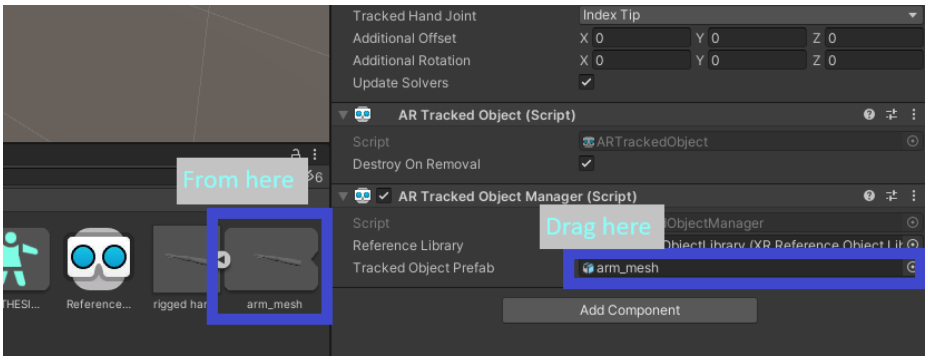

Figure 22. Arm_mesh drag location.

Selecting the “rigged hand” object from the Hierarchy allows the components that need to be added to the scene possible. The components that are added by searching from the “Add Component” button are “SolverHandler” and “SpacialAwarenessSystemManager” as shown in Figure 23. It is noted that the “Spatial Awareness” layer must be activated under the Inspector tab to tell Unity this object follows the real-world object, also shown in Figure 23.

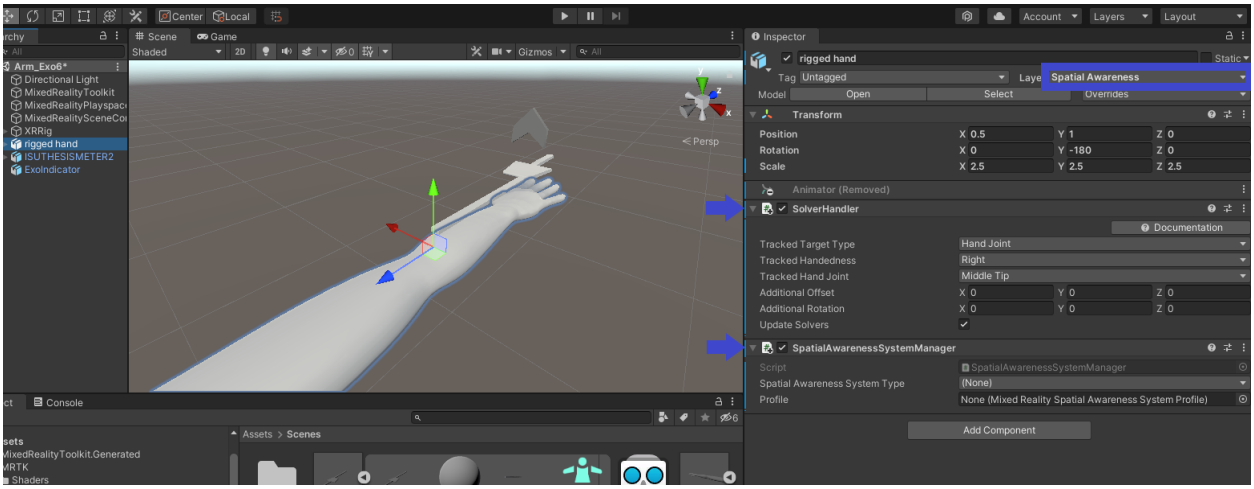

Figure 23. Rigged hand component additions.

To be tracked correctly, the SolverHandler component must be told which specific areas to watch. This is done by manipulating the menus associated with “Tracked Target Type,” “Tracked Handedness,” and “Tracked Hand Joint” as shown in Figure 24.

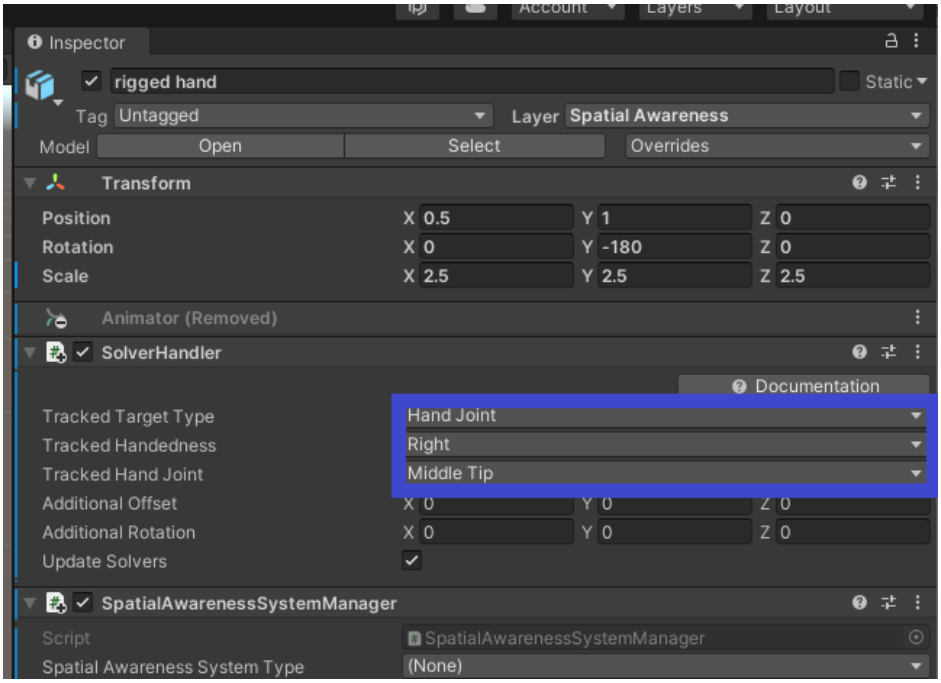

Figure 24. SolverHandler tracking options.

As seen in Figure 24, the scale, rotation, and position of the 3D arm under Transform have been manipulated for the object to be viewed correctly in HoloLens. These values are found by

deploying the scene to HoloLens and changing the scale, rotation, and position values until it is visually correct. Selecting “ISUTHESISMETER2” object under Hierarchy allows for the addition of the components necessary for the exoskeleton to be tracked in the AR environment. The components are added by selecting the “Add Component” button and searching for “Box Collider,” “NearInteractionTouchable,” “Object Manipulator (Script),” and “NearInteractionGrabbable.” The “Manipulation Constraint” component is automatically added when the “NearInteractionGrabbable” component is added. These added components can be seen in Figure 25 through Figure 27.

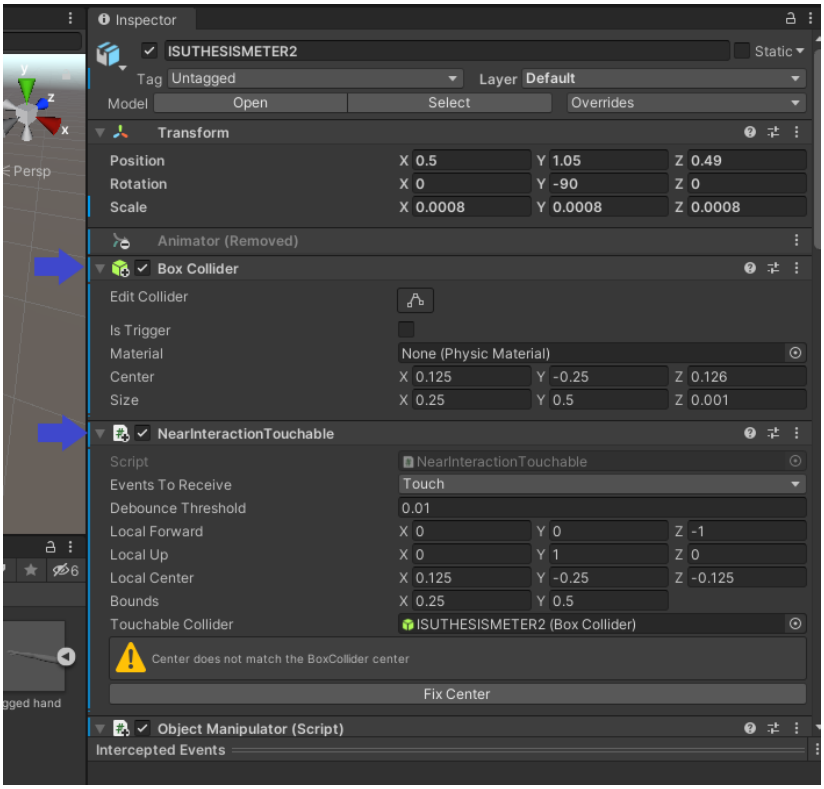

Figure 25. Exoskeleton components 1.

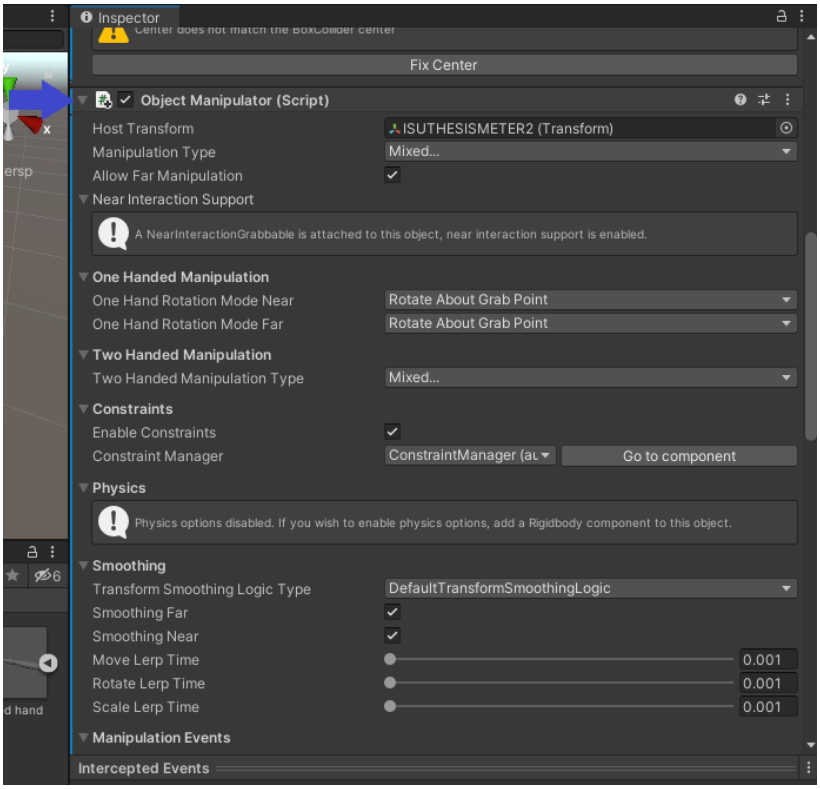

Figure 26. Exoskeleton components 2.

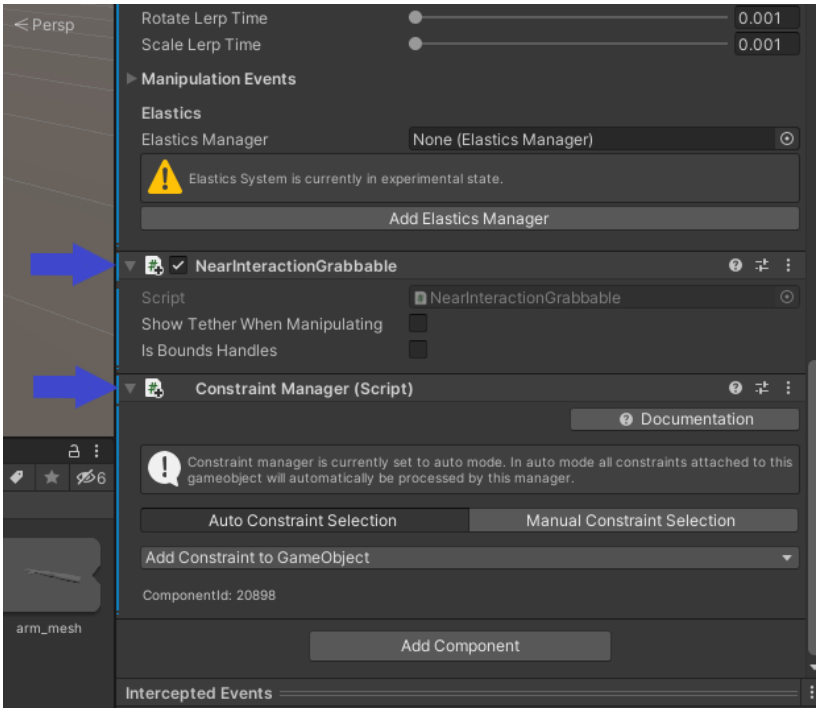

Figure 27. Exoskeleton components 3.

The only modification needed for the exoskeleton components is the scale, rotation, and position. It is shown in Figure 25 that the scale and rotation are changed so that the exoskeleton is positioned over the 3D arm object. This is done so both objects are in the same position when the application is started in HoloLens. The last item that is added to the scene is a marker that is used to direct the user to face the exoskeleton when the application is running. When the view is taken off the exoskeleton, the chevron shape appears in the environment and points to the direction of the exoskeleton. The chevron is moved from the MRTK.Tutorials.GettingStarted folder and dragged into the Hierarchy under the exoskeleton as shown in Figure 28.

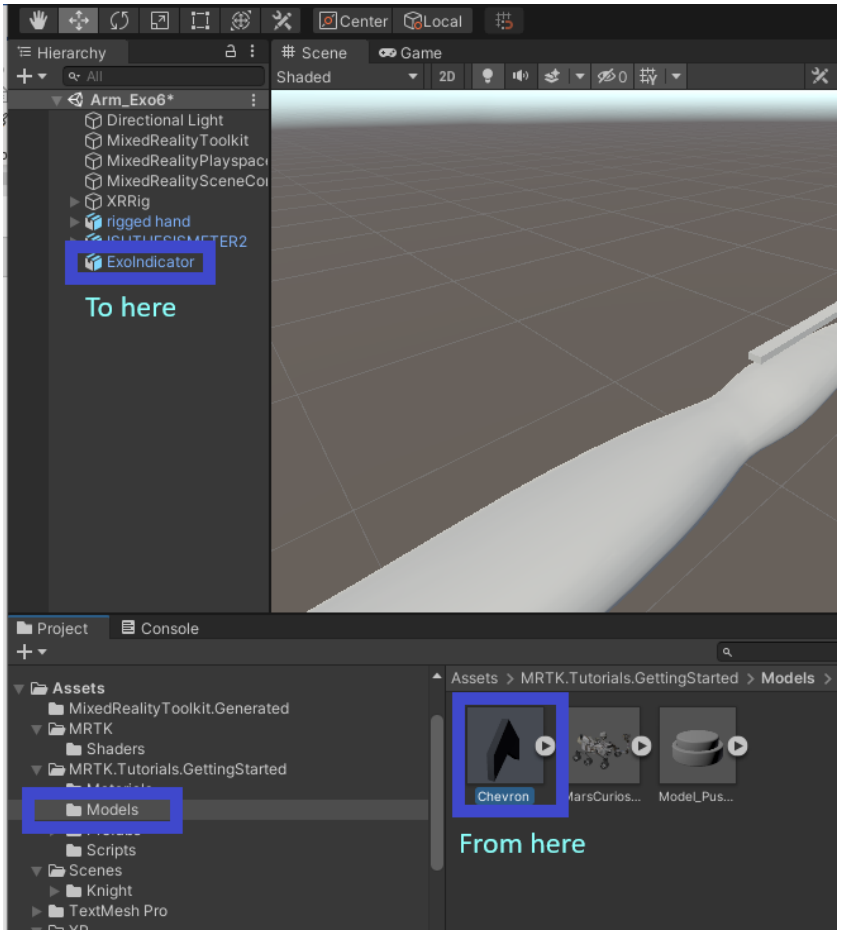

Figure 28. Chevron location.

From there the scale, position, and rotation are changed to match the exoskeleton as shown in Figure 28. All the components of the chevron are already loaded so nothing else is changed.

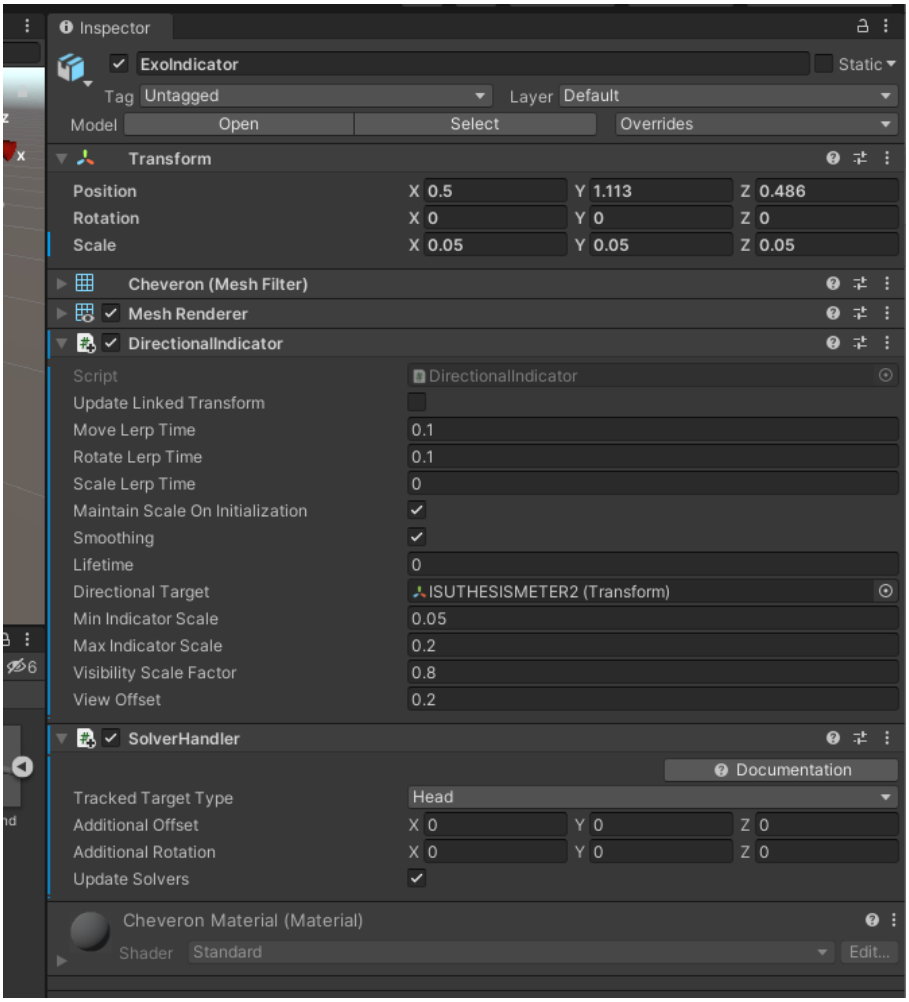

Figure 29. Exoskeleton direction indicator.

Now that all the objects and components are added to the scene, it is ready to build and is sent to Visual Studio.

A build is the final product of the application created in Unity. Creating a build means that the application is processed in a way that Visual Studio can run the solution and ultimately deploy it to HoloLens 2.

Blender

Blender is used to revise the simple exoskeleton .obj file that is imported to the Unity scene and to make the scanned 3D image of a human arm more realistic. After the 3D object is created in SolidWorks®, it is imported to Blender. When Blender is opened for a new object creation, there is a cube that is displayed in the center of the screen. When importing a new file, the cube that is displayed must be deleted for the new object to be the only thing in the file. To create the texture needed for the object to be displayed in Unity, a tutorial is followed to get the details correct. ("Blender") To delete an item in Blender, the item is selected from the Scene Collection pane on the right side of the window. By right clicking on the item and selecting Delete, this action is performed as seen in Figure 30.

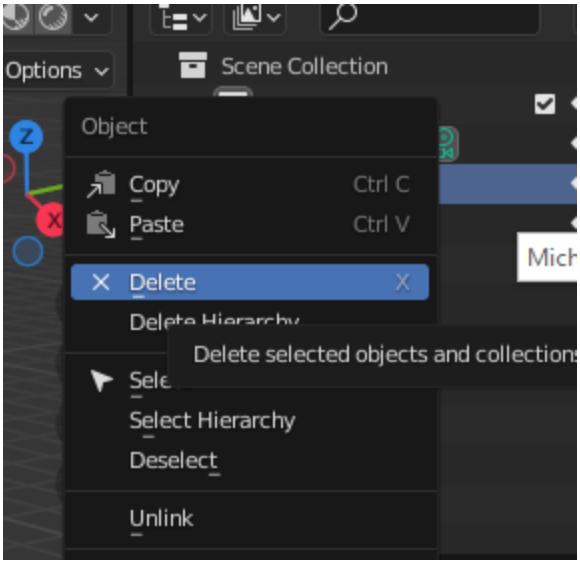

Figure 30. Initial cube deletion.

Importing a new file is done by selecting Import from the File menu and selecting Wavefront(.obj) from the list. When selecting the file, it is important to decide how it will be displayed in the final software and the orientation must be adjusted in the first step before the

Import button is selected. For this project, the Forward Axis is -Z and the Up Axis is Y as shown in Figure 31.

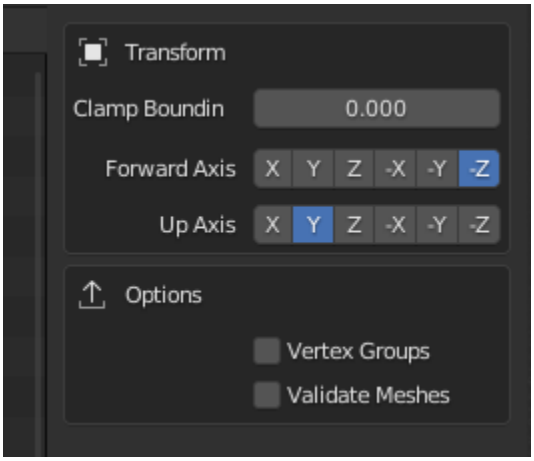

Figure 31. Object orientation setting.

If the orientation is set incorrectly, it can be changed later in the revision or after it is uploaded into Unity. The scale of the object is changed in Blender by using the scale feature in the Object Properties tab as seen in Figure 32.

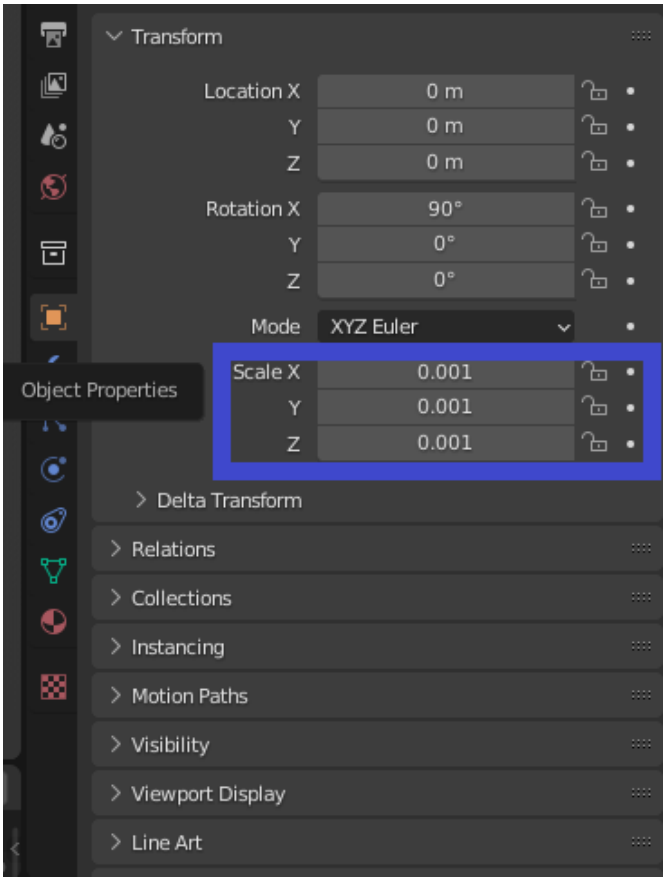

Figure 32. Object scale in Blender.

The scale for the exoskeleton is seen in Figure 32 as being .001 meters. It is not understood why the scale must be made so small as the part was created in SolidWorks® in metric units. It is noted that when it is uploaded into Unity, the scale must also be adjusted for the object to be viewed correctly. To add a texture to the object, the software must be put into "Edit Mode." This is done by selecting "Edit Mode" from the dropdown menu in the upper left corner of the screen or by pressing the Tab key on the keyboard. To ensure all the surfaces of the model have a texture added, it must be unwrapped. In Blender this is known as UV unwrapping. It is explained that unwrapping is like cutting the object along the edges and placing them on a piece of paper. The axes of the paper are considered the dimensions of the vertical and horizontal dimensions of the object, U being vertical and V being horizontal. To unwrap the exoskeleton

model, select the model from the Scene Collection pane and press the “a” on the keyboard. This selects all the parts of the object, and the shading becomes orange as shown in Figure 33.

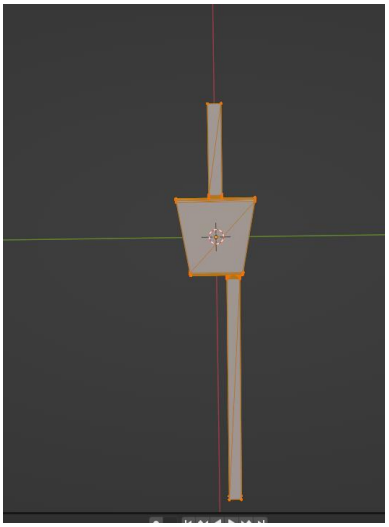

Figure 33. Object selected with "a".

From the menu bar select Smart UV Project from the UV menu as shown in Figure 34.

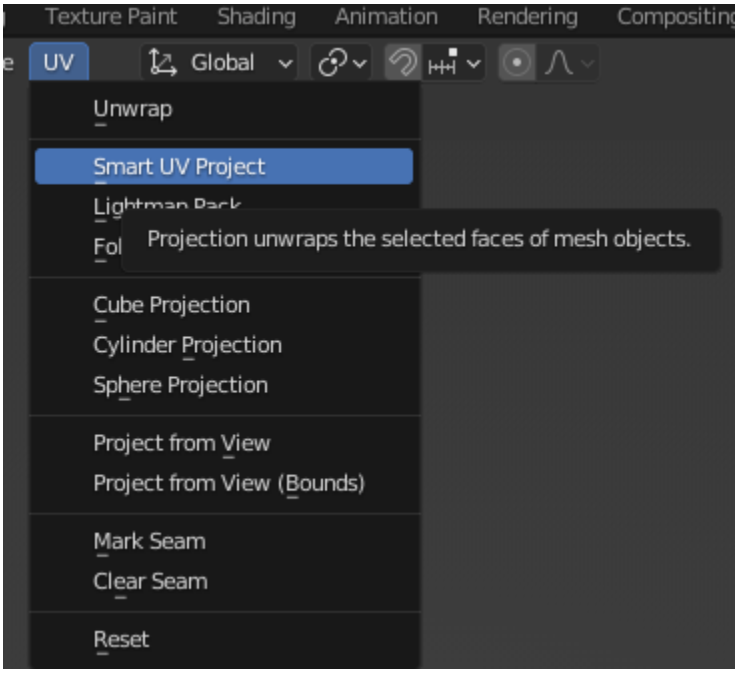

Figure 34. Smart UV Project menu selection.

Figure 35 shows the window that is opened to allow for properties of the UV unwrapped item to be changed, however, none of the properties are changed so OK is pressed.

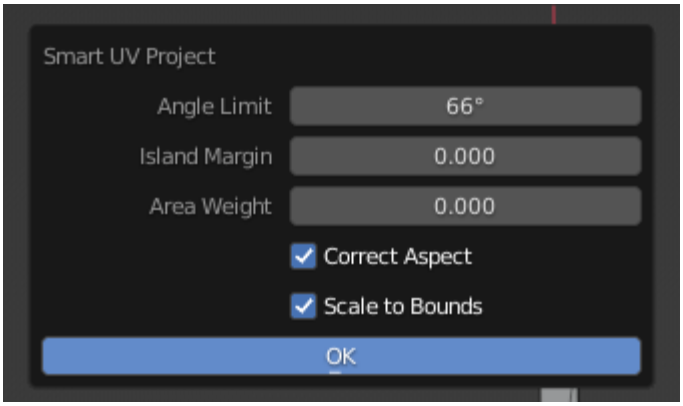

Figure 35. Unwrap properties.

To make the object have the correct appearance, the texture must be created using the UV Editor. This is accomplished by selecting the "UV Editor" from the Viewer Type dropdown menu in the upper left corner as seen in Figure 36.

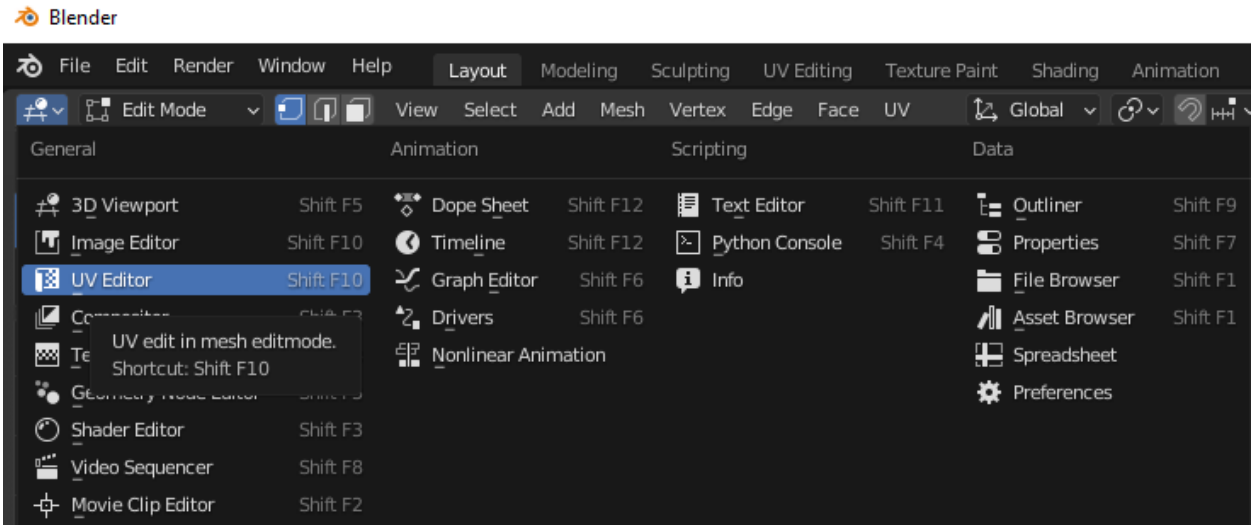

Figure 36. UV Editor option.

After the object is presented in the UV editor, the pieces of the object must be organized that represents the model in an efficient way. This is done by selecting "Pack Islands" from the UV dropdown menu as shown in Figure 37.

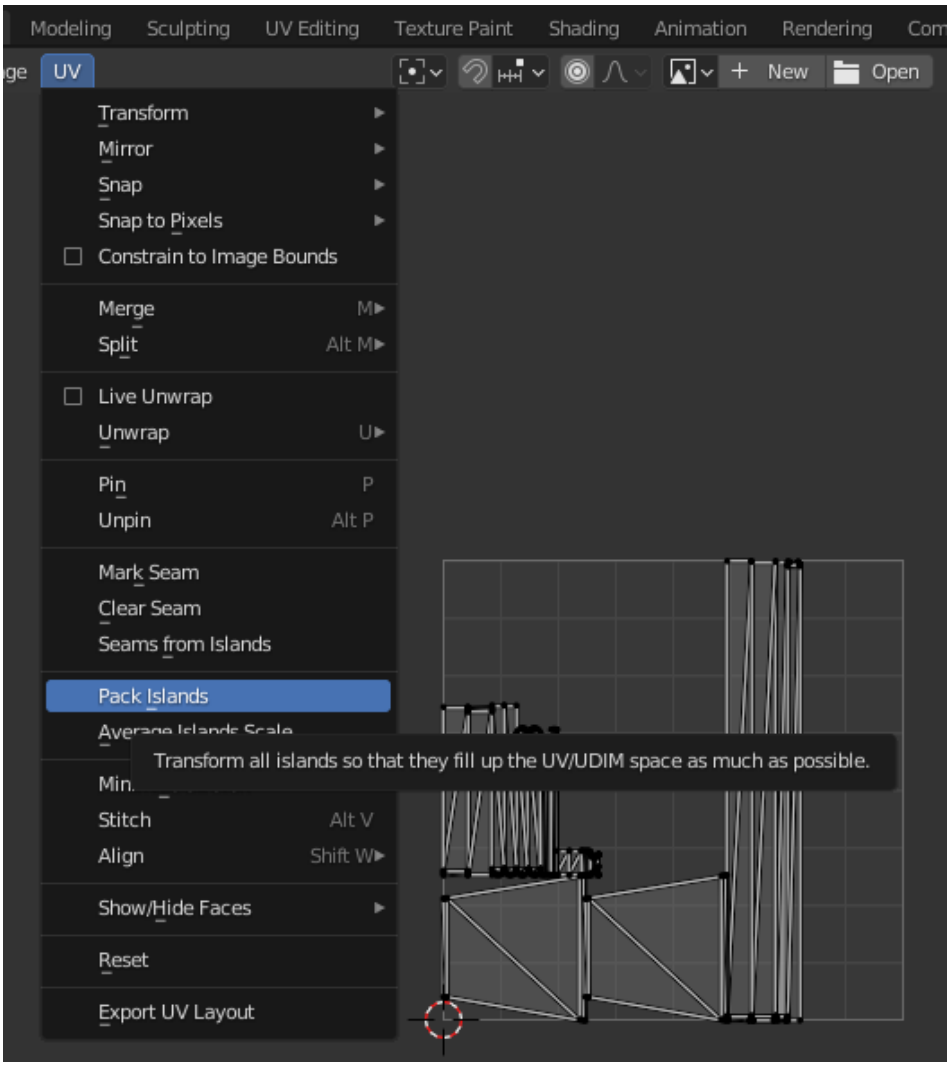

Figure 37. Pack Islands option in UV menu.

The object pieces are displayed with orange highlights to signify they have been packed. Now that the object is separated into sections, a duplicate of the mesh of the object is created to bake the texture onto. Return to the 3D viewport by pressing Shift+F5 or selecting it from the dropdown menu and enter Object Mode. Select the model and right click to open its menu. By selecting Duplicate Objects and pressing the spacebar, a copy of the object is created and placed directly over the original object. If the mouse is moved before pressing the spacebar, the object is moved from the original position. It is unclear if this is problematic for placement in Unity regarding positioning. There are now two of the same objects in the Scene Collection

pane. The second object will have number placeholders at the end of the file name as seen in Figure 38.

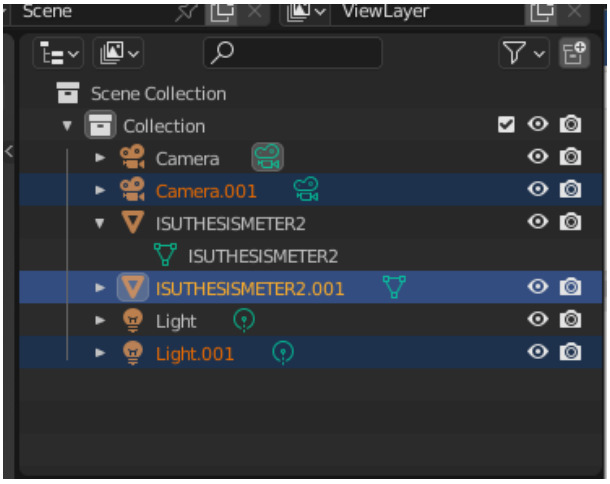

Figure 38. Duplicate objects.

The newly created model is the only one that needs to be seen. For simplicity, hide the original model, select it from the “Scene Collection” pane and click on the eye icon. Now only the duplicate can be seen. Creating the materials that will be baked onto the new model requires adding a new or created image of the texture. This is done by opening the “Materials” tab after the new model is selected. The “Materials” tab is on the right side of the Blender window under the “Scene Collection” pane. A view of the “Materials” icon is shown in Figure 39. Figure 39 also shows where the buttons are to add and delete materials.

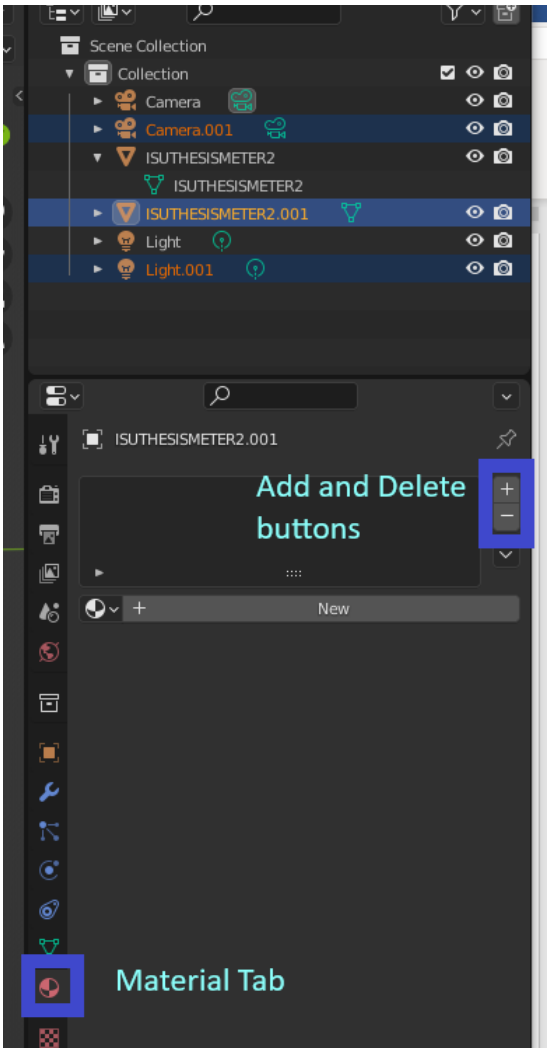

Figure 39. Blender Materials tab.

It is noted that this object has no materials listed in the window. If the object is collected from a website, there may be materials in this list. Each of these materials needs to be removed by pressing the "-" symbol next to the list. Enter the Edit Mode by selecting it from the upper left corner or press Tab on the keyboard. Ensure the copied model is selected and press a on the keyboard to select all the parts. To add materials to this list, the "+" symbol is pressed and then select the "New" button on the bottom of the window. Now the "Assign" button is pressed to create the menu for the new texture. Visual aids of the "New" button, "Assign" button, and a view of the new menu can be seen in Figure 40, Figure 41, and Figure 42 respectively.

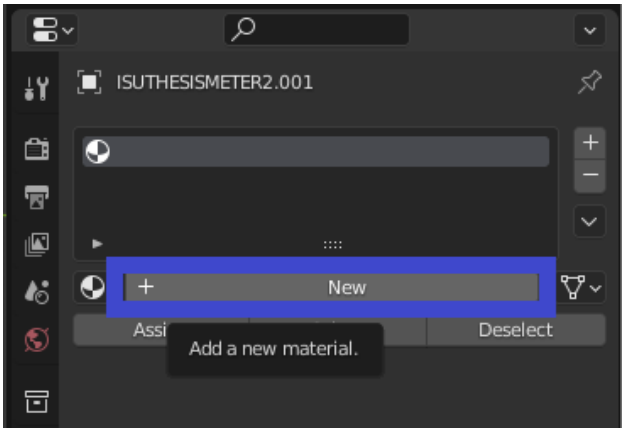

Figure 40. New material button.

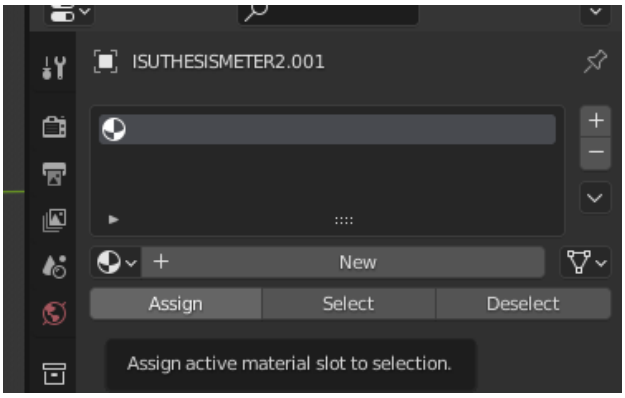

Figure 41. Assign material button.

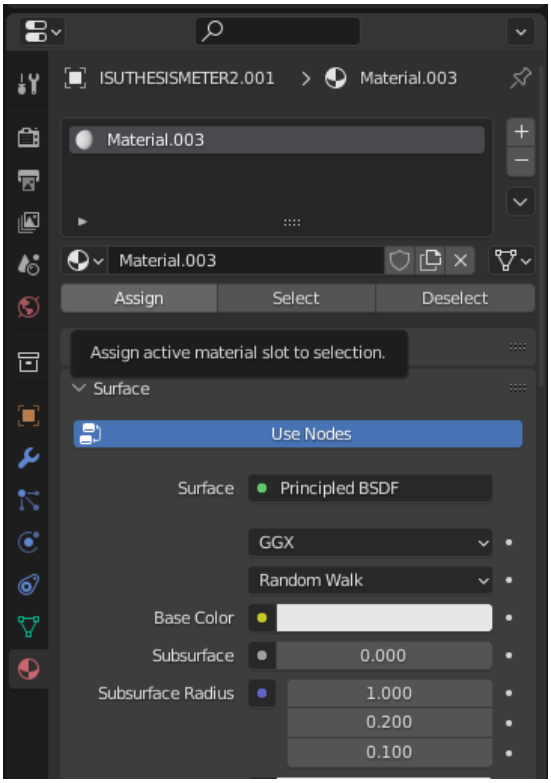

Figure 42. New material menu.

The new object now has a menu that can receive the texture that is created in the next steps. From the Editor Type dropdown menu, select Image Editor to start the editing process. This can also be done by pressing Shift+F10. Create the new image by selecting New from the View dropdown menu as shown in Figure 43. This is saved as ExoTexture.

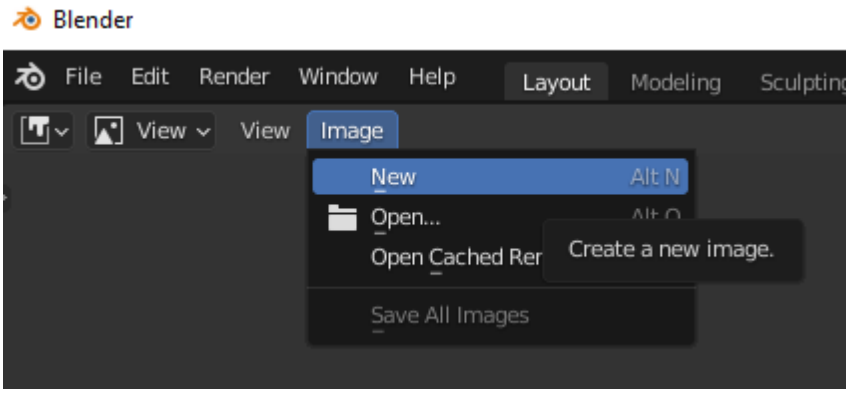

Figure 43. Creating new texture Image.

After the image is named, all other values must be left as default and OK is pressed. Now that the texture has been created, it must be added to the duplicate model. Enter the Shader Editor from the Editor Type menu from the upper left corner of Blender as seen in Figure 44.

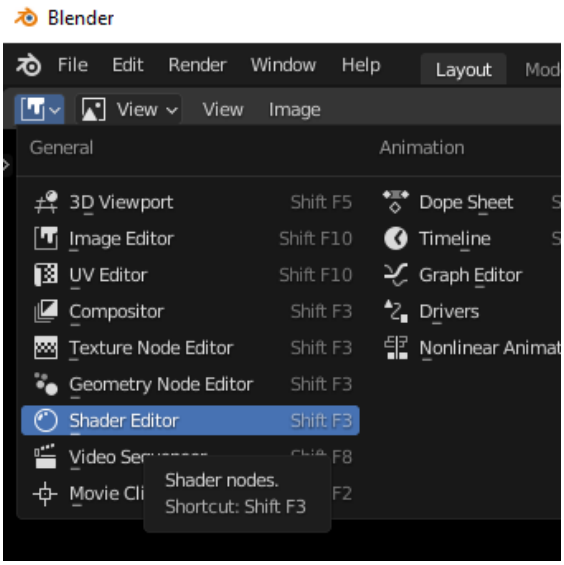

Figure 44. Shader Editor option.

In the Texture Editor, adding an Image Texture creates a node for the texture of the 3D object. This is done by selecting Texture and Image Texture from the Add dropdown menu as shown in Figure 45.

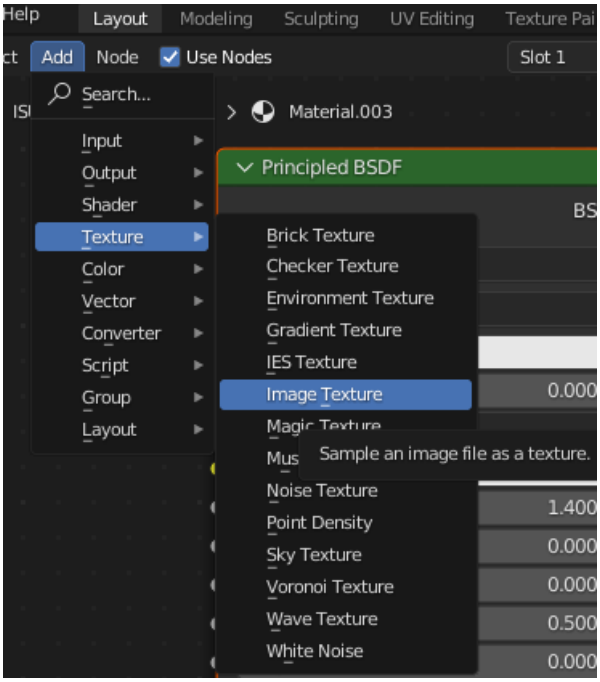

Figure 45. Adding Image Texture node.

In the viewing area of the editor, different boxes can be seen that have different color headers. The nodes of two of the boxes must be connected and the texture image that was created must be selected as the defining attribute. The “Color” node of the orange Image Texture box is dragged to the “Base Color” node of the green Principled BSCF box as shown in Figure 46 and the name of the created texture is selected from the Image icon as shown in Figure 47.

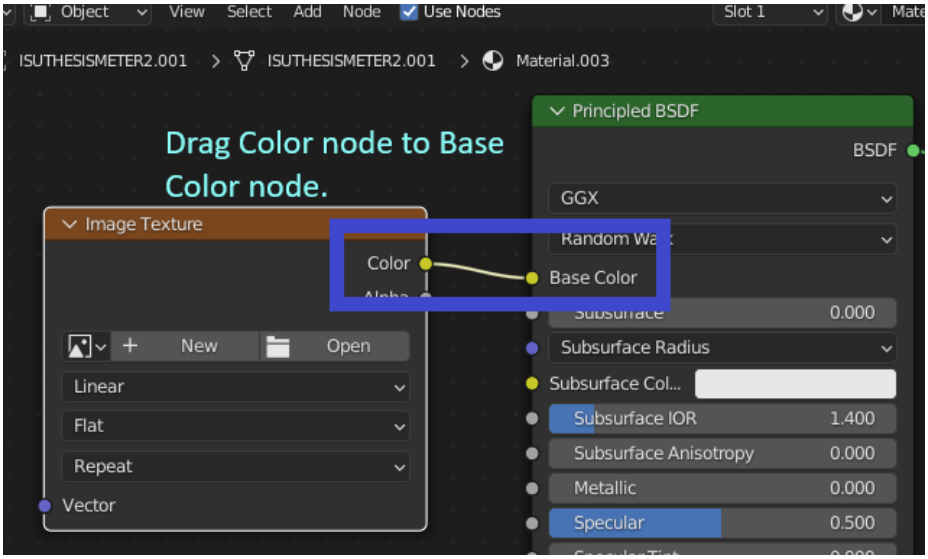

Figure 46. Color node connections.

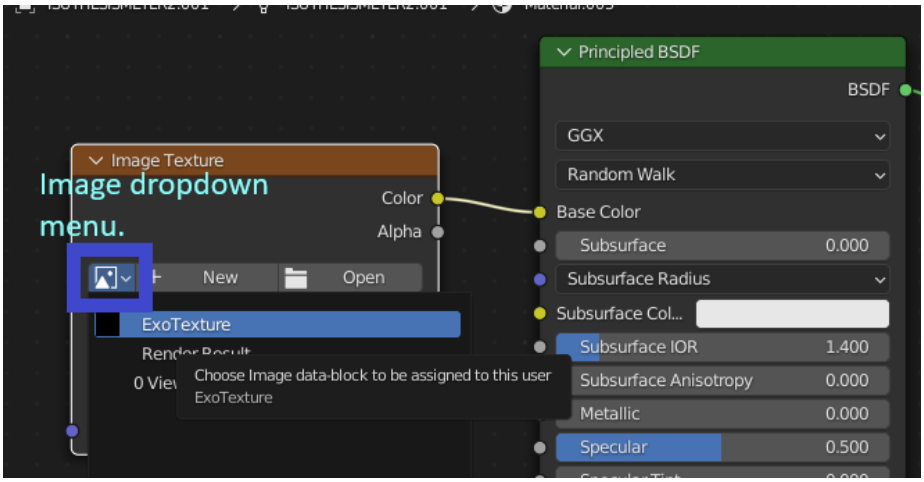

Figure 47. Image menu and texture name.

To finish the process of creating the texture, the image is baked onto to duplicate object. Baking saves the texture image of the original object onto the duplicate object by wrapping it onto the pieces created earlier. Return to the 3D Viewport by selecting it from the Editor Type dropdown menu. Select the Render tab under the Scene Collection window and select “Cycles” as the Render Engine as seen in Figure 48.

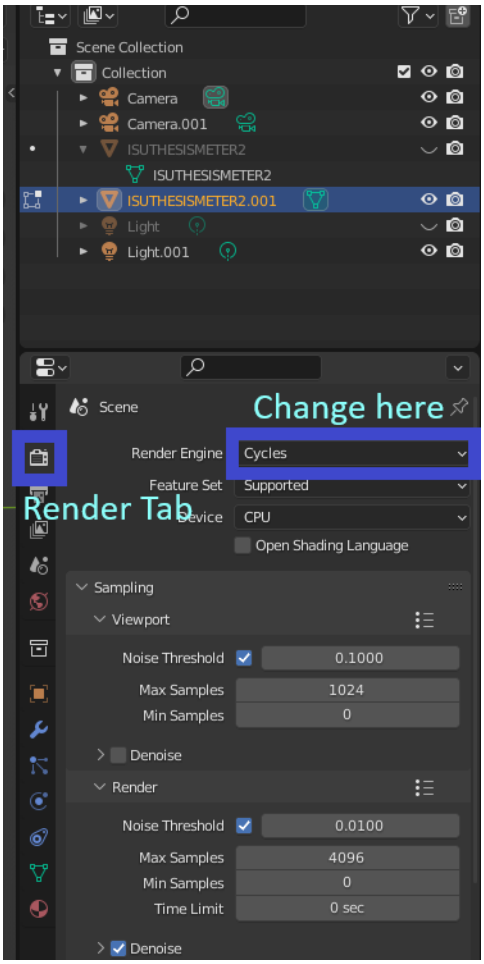

Figure 48. Render tab and Render Engine type.

In the Bake menu of the same tab, change the Bake Type to “Diffuse” and uncheck the boxes Direct and Indirect. The next step requires selecting the original object under the “Scene Collection” pane and while holding Shift, select the duplicate model. With both models selected, in the Render tab under the Bake menu, select the Selected to Active check box. Click on the arrow to extend the menu at Selected to Active and change the value in Ray Distance from 0 to 0.01. These steps are shown in Figure 49. The final step for baking is to press the Bake button.

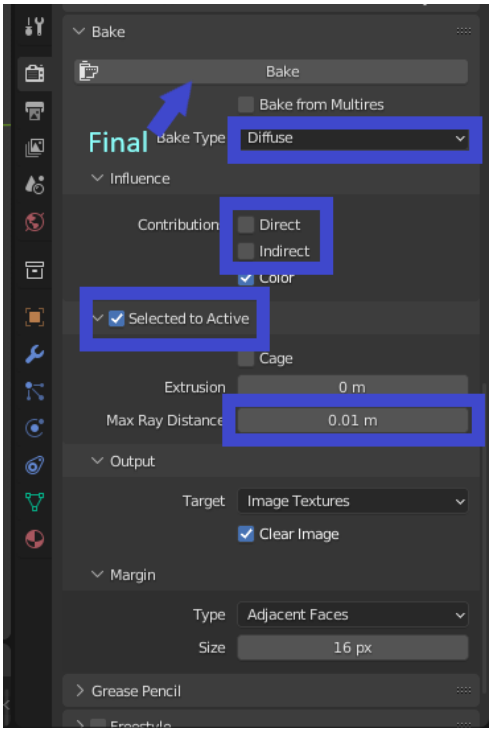

Figure 49. Render tab Bake option changes.

To use the object in Unity it is exported as a .obj or a .fbx file. For this project the file was saved as .obj. This process iss also used for the 3D arm scan that was created using an Einscan Pro 2X which can be seen in Figure 50.

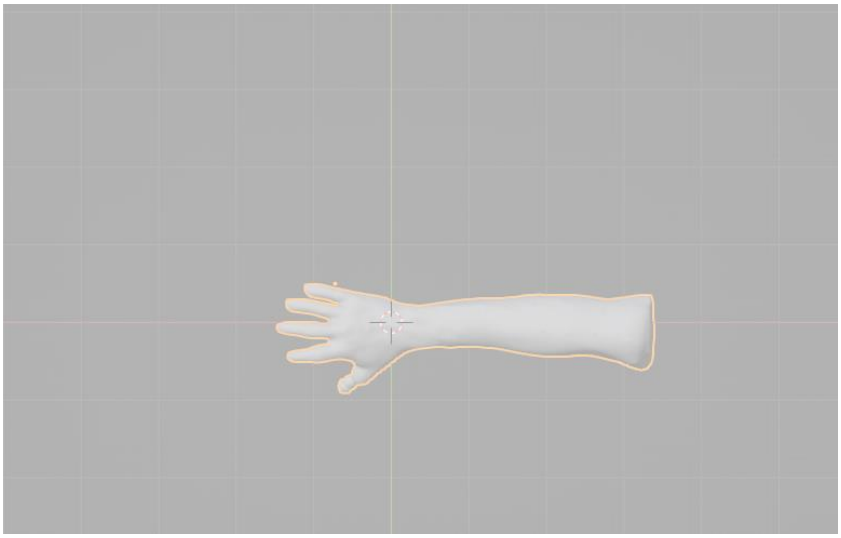

Figure 50. Einscan Pro2X arm scan shown in Blender.

The Einscan Pro2X is a portable 3D scanner that uses lasers and a depth camera to create 3D objects. The file is imported to Blender and when these steps are tried with the arm scan, the software causes the computer to stop responding. This is due to the number of the object's polygons being too large. Another computer is also tried with the same results. This problem makes the object not compatible with Unit and HoloLens. To remedy this issue, a 3D arm is downloaded from the Sketchfab website. (*Human Arm 3d Model - Google Search*) Since the arm is downloaded from a website, it already has textures added to it so none of the previous steps are performed. All Unity builds are created using this arm model and the Visual Studio solution is deployed to the HoloLens device.

Visual Studio

The file that has been created from the Unity build is in .sln format which is the Visual Studio solution file. This file is opened to prepare it for deployment to HoloLens 2 device. As discussed in Section 3 under the Visual Studio heading, the dropdown menus must be changed to match the system that the application is being deployed to. A visual representation of these dropdown menus along with the other options available in them can be seen in Figure 51 through Figure 53.

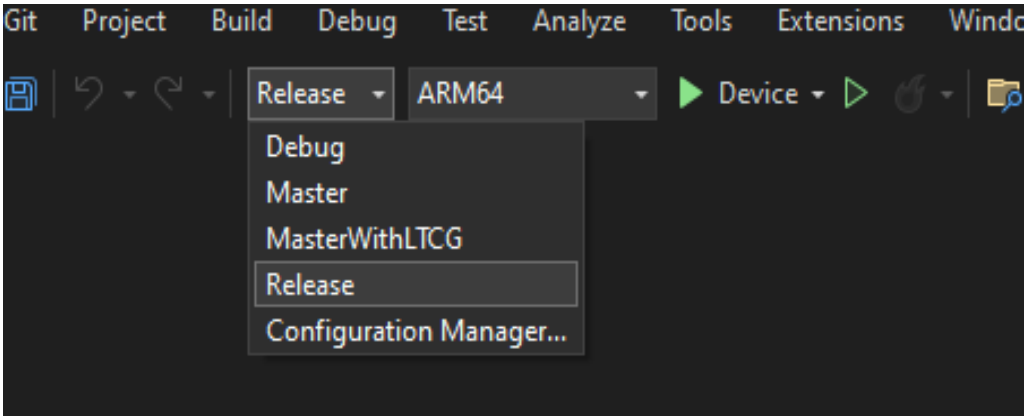

Figure 51. Release Dropdown Menu.

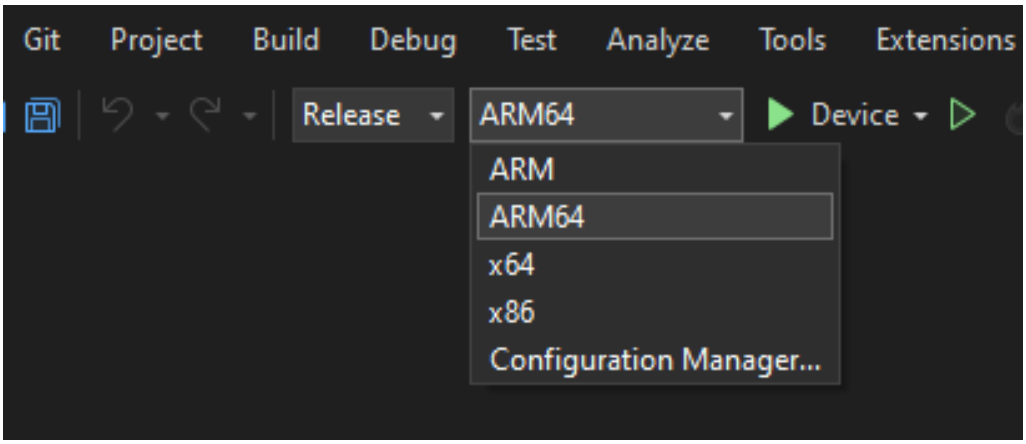

Figure 52. ARM64 Dropdown Menu.

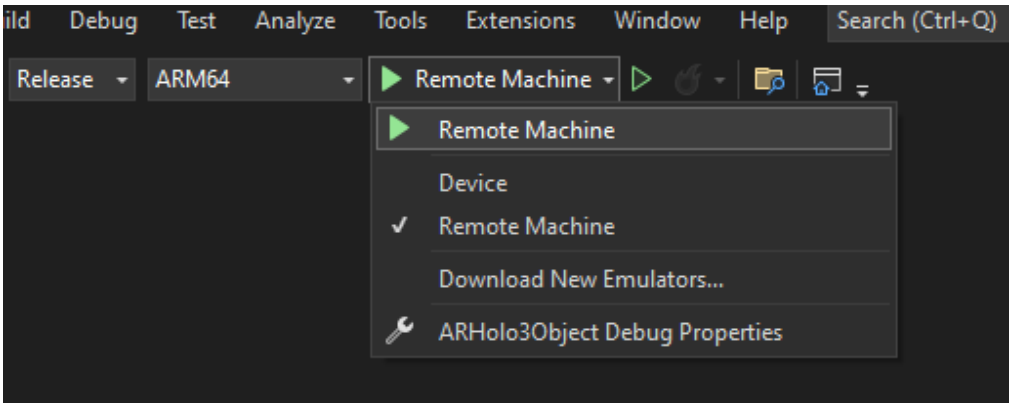

Figure 53. Remote Machine Dropdown Menu.

To change the settings of the project, select Settings from the Project tab in the top menu bar as shown in Figure 54. The Machine Name is the IPv4 address that was discussed in Section 3 under the Visual Studios heading and can be seen correctly input in Figure 55.

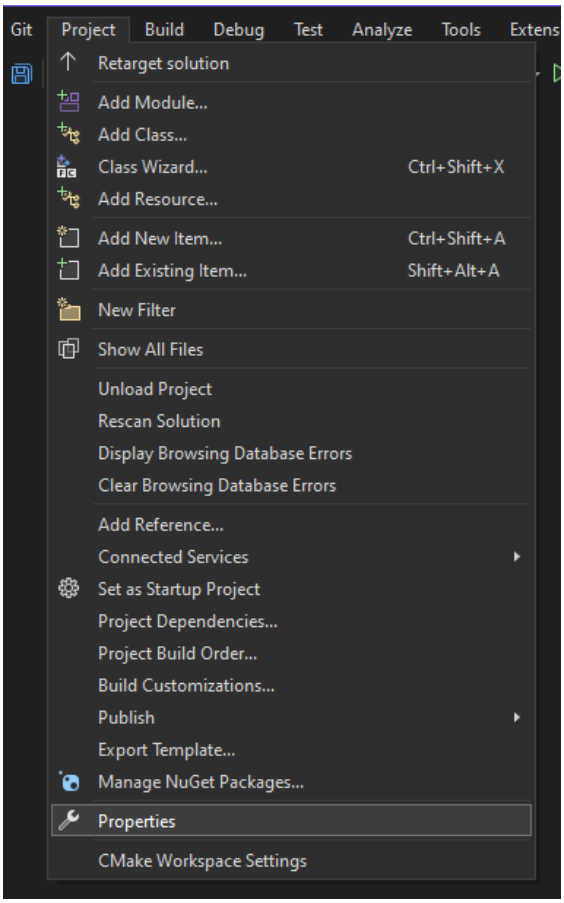

Figure 54. Project Properties Tab.

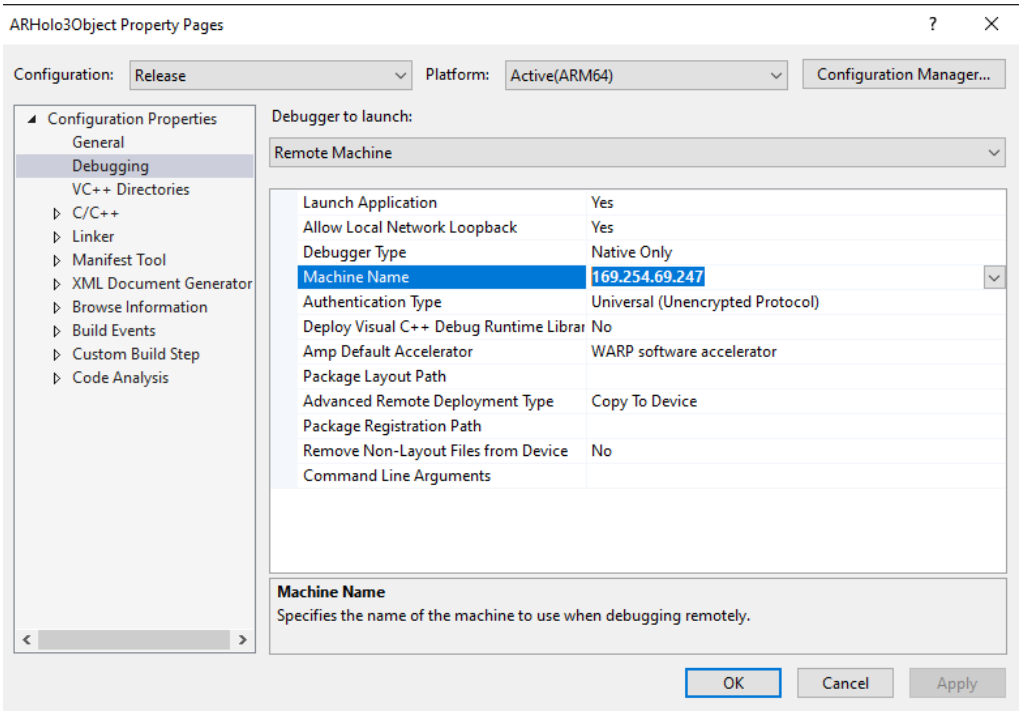

Figure 55. Machine Name Window.

Once these settings are set in the software, it is now ready to be used to transfer the project onto the HoloLens 2 device. To start deployment, simply press the dropdown menu labeled Remote Machine with the green play icon. At this time the HoloLens 2 must be connected to the computer using a USB cable and turned on. The application is automatically deployed and started on the HoloLens device after all the files have been transferred. At this point, the connection between the computer and the HoloLens 2 is stopped by pressing the red stop sign button. This stops the playback of the application in HoloLens and the computer. The application is restarted by selecting it from the “All apps” menu and pressing the name of the application.

6 Lessons Learned

Creating the 2D image detection scene can be achieved by following the before mentioned YouTube videos and tutorials. 3D object detection on the other hand is much more complicated and requires more insight into the different dependencies and functioning’s of the

Unity and HoloLens platforms. In the tutorials from Wikitude, the detection scene is created and built. However, when deployed to HoloLens, the object doesn't seem to be detected. (*Object Tracking Wikitude SDK Hololens 9.3.0 Documentation*) It is unclear as to why the exoskeleton does not follow the arm in the AR environment. It may be that a machine learning API must be implemented in the Unity application. Wikitude and Vuforia examples are the only platforms that were explored for this project. Utilizing ChatGPT for an example of how to create a scene and implement it to HoloLens 2 results in a possible new approach for continued research. Full text of the ChatGPT response can be seen in Appendix A.

After exploring Blender tutorials and working with a scanned 3D object, it has become evident that these objects can generate an excessive number of faces, posing challenges for moderately powerful computers to handle. Research into minimizing faces of 3D objects is a crucial step in future creations.

SolidWorks® has become a very powerful platform that not only has CAD capabilities but can be used for generating objects of different file types. These file types can be shared by many software platforms for multiple uses. A hopeful future prospect for SolidWorks®, with the advancement of AR/VR platforms, will be a reintroduction of the OBJ Exporter Add-in. The creation of the .obj file type directly in SolidWorks® made importing to Unity much easier. This is made easier due to the elimination of the lengthy process of baking the material onto the object.

Unity and HoloLens' requirement for animations of holographic images to enable realistic movement, although difficult, add a very important visual aspect to the AR environment.

Blender and Unity are both exciting platforms to discover. Each has many capabilities that need to be discovered to be able to create the envisioned application explored in this project. In Unity, the MRTK package needs further investigation to fully understand how to create the object recognition that will allow objects to be recognized and measured. The spatial awareness components of MRTK are vital to the scene creation that will enable the exoskeleton and prosthetic devices to be properly aligned and sized using AR environments.

7 Conclusion

While great strides have been made in the development of AR/VR rehabilitation aids, there are still improvements to be made. If an AR environment can be successfully created that would immediately size the components of an exoskeleton or a prosthetic device, not only the time required to build the device could significantly be decreased but also fitting costs can be reduced. With the advancement in additive manufacturing the cost of the device could be further reduced. The improvements that can be made to the current project include creating a script that can read the distance information from the depth camera on the HoloLens 2 device and create a Microsoft Excel document. This document could then be used by SolidWorks® to use the distance information to automatically resize the components of a complete exoskeleton. Another future improvement is the addition of a human skin texture to the 3D arm. This may allow the real-world arm to be detected more easily as the object would appear more life-like. More insight into object scanning may need to be explored to allow for a cleaner scan that would require less modifications.

References

About XR SDK for Windows MR | Windows XR Plugin | 3.4.0. https://docs.unity3d.com/Packages/com.unity.xr.windowsmr@3.4/manual/index.html. Accessed 1 June 2023.

AR Tracked Object Manager | AR Foundation | 3.0.1. https://docs.unity3d.com/Packages/com.unity.xr.arfoundation@3.0/manual/tracked-object-manager.html. Accessed 18 May 2023.

Augmented Reality (AR) Tutorial for Beginners Using Unity 2022. Directed by Playful Technology, 2022. *YouTube*, https://www.youtube.com/watch?v=gpaq5bAjya8.

"Blender: UV Mapping – Simply Explained." *All3DP*, 9 Mar. 2021, https://all3dp.com/2/blender-uv-mapping-simply-explained/.

Corbett, Jeremy. *The Battle of File Formats: STL vs OBJ vs PLY*. https://blog.medit.com/medit/the-battle-of-file-formats-stl-vs-obj-vs-ply. Accessed 12 May 2023.

DXF/DWG Files (.Dxf, *.Dwg Files) - 2021 - SOLIDWORKS Help*. https://help.solidworks.com/2021/english/SolidWorks/sldworks/c_dxf-dwg_files_dxf_dwg-files.htm. Accessed 12 May 2023.

Frantz, Taylor, et al. "Augmenting Microsoft's HoloLens with Vuforia Tracking for Neuronavigation." *Healthcare Technology Letters*, vol. 5, no. 5, Oct. 2018, pp. 221–25. *DOI.org (Crossref)*, https://doi.org/10.1049/htl.2018.5079.

Free Solidworks OBJ Exporter v2.0 | SOLIDWORKS Forums. https://forum.solidworks.com/thread/54270?start=0&tstart=0. Accessed 3 Mar. 2023.

Hololens 2 Development Tutorial - Getting Started (MRTK V2). Directed by Dinesh Punni, 2020. *YouTube*, https://www.youtube.com/watch?v=mSSVcT2PpKk.

How To Setup the Unity OpenXR Plugin With MRTK for HoloLens 2. Directed by Dilmer Valecillos, 2021. *YouTube*, https://www.youtube.com/watch?v=4PpqOlCnbGM.

Human Arm 3d Model - Google Search. https://www.google.com/search?q=human+arm+3d+model&tbm=isch&sa=X&ved=2ahUKEwjgnKzA_MX-AhV9AjQIHSFYDrEQ0pQJegQIDBAB&biw=1212&bih=664&dpr=1.25#imgrc=jz-ymP73MyjxeM. Accessed 1 June 2023.

Mubin, Omar, et al. "Exoskeletons With Virtual Reality, Augmented Reality, and Gamification for Stroke Patients' Rehabilitation: Systematic Review." *JMIR Rehabilitation and Assistive Technologies*, vol. 6, no. 2, Sept. 2019, p. e12010. *DOI.org (Crossref)*, https://doi.org/10.2196/12010.

Müller, Fabio, et al. "Augmented Reality Navigation for Spinal Pedicle Screw Instrumentation Using Intraoperative 3D Imaging." *The Spine Journal*, vol. 20, no. 4, Apr. 2020, pp. 621–28. *DOI.org (Crossref)*, https://doi.org/10.1016/j.spinee.2019.10.012.

Object Recognition | VuforiaLibrary. https://library-archive.vuforia.com/features/objects/object-reco.html. Accessed 27 May 2023.

Object Tracking Wikitude SDK Hololens 9.3.0 Documentation. https://www.wikitude.com/external/doc/documentation/9.9/hololens/objecttrackingnative.html. Accessed 17 May 2023.

Park, Sebeom, et al. “Review of Microsoft HoloLens Applications over the Past Five Years.” *Applied Sciences*, vol. 11, no. 16, Aug. 2021, p. 7259. *DOI.org (Crossref)*, https://doi.org/10.3390/app11167259.

Phan, Huu Lam, et al. “Effectiveness of Augmented Reality in Stroke Rehabilitation: A Meta-Analysis.” *Applied Sciences*, vol. 12, no. 4, Feb. 2022, p. 1848. *DOI.org (Crossref)*, https://doi.org/10.3390/app12041848.

“PLY (File Format).” *Wikipedia*, 24 Oct. 2022. *Wikipedia*, https://en.wikipedia.org/w/index.php?title=PLY_(file_format)&oldid=1117917512.

“Polygon Models: You Have These Possibilities in SOLIDWORKS.” *PLM Group Support Center*, 2 Oct. 2020, https://support.plmgroup.eu/hc/en-us/articles/360014001718-Polygon-models-You-have-these-possibilities-in-SOLIDWORKS.

qianw211. *Choosing a Unity Version and XR Plugin - Mixed Reality*. 1 Feb. 2023, https://learn.microsoft.com/en-us/windows/mixed-reality/develop/unity/choosing-unity-version.

---. *Unity Development for VR - Mixed Reality*. 4 April 2022, https://learn.microsoft.com/en-us/windows/mixed-reality/develop/unity/unity-development-wmr-overview.

“STL (File Format).” *Wikipedia*, 3 Feb. 2023. *Wikipedia*, https://en.wikipedia.org/w/index.php?title=STL_(file_format)&oldid=1137197385.

Tran, Phillip, et al. “Review: Hand Exoskeleton Systems, Clinical Rehabilitation Practices, and Future Prospects.” *IEEE Transactions on Medical Robotics and Bionics*, vol. 3, no. 3, Aug. 2021, pp. 606–22. *DOI.org (Crossref)*, https://doi.org/10.1109/TMRB.2021.3100625.

What File Typre Does Hololens Use for 3d Objects - Google Search.

https://www.google.com/search?q=What+file+typre+does+hololens+use+for+3d+objects&oq=What+file+typre+does+hololens+use+for+3d+objects&aqs=chrome..69i57j33i10i299.13905j0j7&sourceid=chrome&ie=UTF-8. Accessed 1 May 2023.

Appendix A

Chat GPT Response

Here's a step-by-step breakdown of implementing 3D object recognition in Unity for HoloLens 2:

1. Set up the HoloLens 2 Development Environment:
- Install Unity: Download and install the appropriate version of Unity from the Unity website.
- Install Windows Mixed Reality (WMR) Package: In Unity, navigate to the Package Manager and install the Windows Mixed Reality package, which provides the necessary tools for HoloLens development.
- Set up SDKs and Developer Mode: Follow the official Microsoft documentation to install the Windows 10 SDK and enable Developer Mode on your HoloLens 2 device.
2. Acquire 3D Object Recognition Dataset:
- Gather 3D Models: Collect or create a dataset of 3D models representing the objects you want to recognize. Ensure the dataset covers different variations and viewpoints of the objects.
- Annotate 3D Models: Manually annotate the 3D models with additional information, such as bounding boxes, keypoints, or semantic labels, to assist in the recognition process.
3. Train a Machine Learning Model:
- Choose a Framework: Select a machine learning framework compatible with Unity, such as TensorFlow or PyTorch.
- Set up Training Environment: Configure your development environment to work with the chosen framework and install the necessary dependencies.
- Preprocess the Dataset: Prepare the dataset by preprocessing the 3D models, extracting features, and splitting it into training and validation sets.
- Design Model Architecture: Define the architecture of your machine learning model suitable for 3D object recognition tasks, such as PointNet, PointNet++, or VoxNet.
- Train the Model: Use the labeled 3D dataset to train the model using the chosen framework. Adjust hyperparameters, such as learning rate or batch size, for optimal performance. Iterate on the training process as needed.
4. Integrate the Trained Model into Unity:
- Import the Model: In Unity, use the ML-Agents package or other machine learning frameworks to import the trained model into your Unity project.
- Define Input and Output: Specify the input format the model expects (e.g., point cloud data, voxel grid representation) and the output format (e.g., object class labels, pose estimation).
- Test Model Integration: Validate that the model is properly integrated by running test inputs and inspecting the model's output within Unity.
5. Access Sensor Data and Perform 3D Object Recognition:
- Utilize HoloLens 2 Capabilities: Use Unity's APIs and the Mixed Reality Toolkit (MRTK) to access sensor data, including depth information, from the HoloLens 2 device.

- Capture Depth Data: Retrieve depth frames from the HoloLens 2 depth sensor and process them to obtain accurate 3D point cloud data representing the scene.
- Preprocess Input Data: Prepare the captured depth frames or point cloud data by applying necessary transformations or normalizations required by the model.
- Run Inference: Feed the preprocessed input data into the trained model and run inference to obtain predictions for 3D object recognition, such as object class labels or estimated object poses.
- Post-process Results: Process the model's output to interpret the predictions and extract relevant information, such as object labels, confidence scores, or 3D object poses.

6. Visualize Recognition Results in Augmented Reality:

- Overlay Augmented Reality Visuals: Utilize Unity's augmented reality capabilities to overlay the recognition results onto the real-world view seen through the HoloLens 2 device.
- Display Labels, Bounding Boxes, or 3D Models: Use Unity's UI system or 3D objects to display labels, bounding boxes, or 3D models representing recognized objects within the user's field of view.
- Update Visualizations in Real-Time: Continuously update the augmented reality visualizations as the sensor data refreshes and the model provides new predictions.

By following these step-by-step instructions, you can successfully implement 3D object recognition in Unity for HoloLens 2. This allows you to detect and recognize objects in a three-dimensional space, overlay augmented reality visualizations, and provide a more immersive and interactive experience for the user.