Hand Gesture Recognition

Samuel Oncken, Steven Claypool

INTERFACE CONTROL DOCUMENT

REVISION - 2
3 October 2022

INTERFACE CONTROL DOCUMENT FOR Hand Gesture Recognition

PREPARED BY:

Team 72	10/3/2022
Author	Date

APPROVED BY:

Samuel Oncken	10/3/2022
Project Leader	Date
John Lusher II, P.E.	Date
T/A	Date

Change Record

Rev.	Date	Originator	Approvals	Description
1	10/3/2022	Samuel Oncken Steven Claypool		Draft Release
2	11/28/2022	Samuel Oncken Steven Claypool		Revision for Final Report Release

Table of Contents

List of Figures

1. Overview

This document will describe how each subsystem will interface with one another to produce the results discussed in the Concept of Operation and Functional System Requirements. We will walk through the inputs and outputs of each subsystem to fully understand how each build upon the next. Additionally, we will discuss a few physical characteristics of our system.

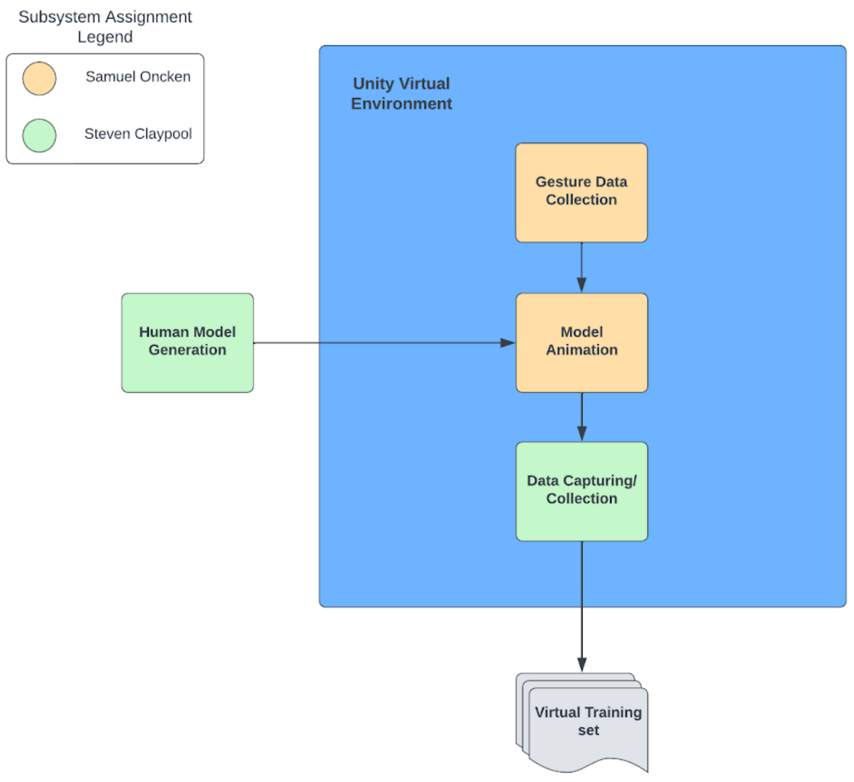

Figure 1: Block Diagram of subsystem interaction

2. References and Definitions

2.1. References

Unity User Manual 2021.3 (LTS)
Version 2021.3
09/23/2022

Ultraleap for Developers - Unity API User Manual
Release date 2021

UH-003206-TC
Leap Motion Controller Data Sheet
Issue 6

2.2. Definitions

LMC	Leap Motion Controller
SOTA	State of the Art
VR	Virtual Reality
FBX	Filmbox File Format
XR	Extended Reality
Spawning	Loads an existing object or model into a scene
Transform data	Data describing the position, rotation, and scale in the x, y, and z directions of a game object within Unity. Rigged models contain parent/child components, which means the child transform is relative to the transform of the parent. For example, the tip of the index finger is a child of the middle bone within the index finger.
Prefab	A fully configured game object that includes specific components/settings that can be stored and reused in scenes or projects. MakeHuman models and Ultraleap-provided rigged hands are examples of prefabs.
Rigged Model	Any model that contains an internal structure that defines its motion. The MakeHuman virtual models are imported with a default rig, defining their skeletal structures with over one hundred unique bones.
Plugin	Software components that add functionality to an existing system. In Unity, we are using multiple plugins that allow us to map rig structures more easily, use prefabs and pre-written Ultra Leap hand tracking scripts, record gesture motion, and export animations in the correct file type.

3. Physical Interface

3.1. Weight

The LMC weighs 32 grams. The cables to connect the LMC to a computer are negligible in weight.

3.2. Dimensions

3.2.1. Dimension of LMC within Gesture Data Collection Subsystem

The LMC is 80mm long, 30mm tall, and 11.30mm deep.

3.3. Mounting Locations

Mounting locations of the LMC for recording hand gesture data include head, chest, screen and desk mounted. From testing, the most accurate recording results from head mounting the LMC, which gives a clear, unobstructed view of the hands and fingers for data collection.

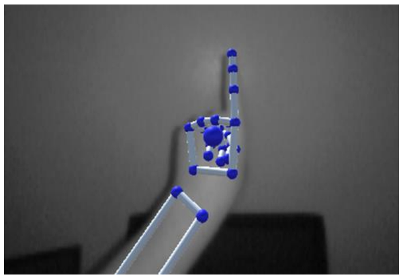

Figure 2: Hand Mapping Through LMC Control Panel

4. Thermal Interface

Not applicable to our research. As mentioned in the FSR, the Leap Motion Controller will function in temperatures from 32° to 113°F, which we will not be exceeding. Therefore, we do not require a thermal interface.

5. Electrical Interface

5.1. *Primary Input Power*

The LMC needs to receive 5V at a minimum of 0.5A from the USB port.

5.2. *Polarity Reversal*

Not applicable to our research

5.3. *Signal Interfaces*

Not applicable to our research

5.4. *Video Interfaces*

Not applicable to our research

5.5. *User Control Interface*

Not applicable to our research. User facing interfaces are all within our Unity environment, which is described in the Software Interface section of this report.

6. Communications / Device Interface Protocols

6.1. Wireless Communications (WiFi)

Not applicable to our research. WiFi is required for downloading the necessary software, but once our environment is set up, there is no need for a WiFi connection.

6.2. Host Device

The host device needs a USB 2.0 port minimum for the LMC.

6.3. Video Interface

Not applicable to our research.

6.4. Device Peripheral Interface

The LMC is a peripheral device of our system. Future plans may include VR headsets such as Vive Pro or Oculus headsets.

7. Software Interface

7.1. LMC Interface

7.1.1 LMC Tracking in Unity

The Leap Motion Controller is able to interface with Unity using the "Ultraleap Plugin for Unity" downloaded from the Ultraleap website. This plugin includes a Service Provider (XR) prefab which enables hand tracking and displays transform data relative to the head mounted LMC as shown in Figure 2 below. Additionally, this plugin includes a number of rigged hand prefabs as well as scripts that enable the tracking of each individual bone.

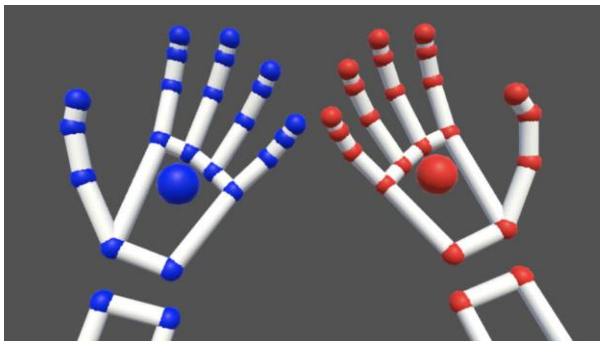

Figure 3: Ultraleap Rigged Hand Model in Unity

7.1.2 LMC Tracking on MakeHuman Model

The LMC must interface with an imported MakeHuman model within Unity for the Gesture Data Collection process. This can be carried out inside of Unity using the Service Provider (XR) prefab as well as the "Hand Binder" script offered in the Ultraleap Unity Plugin. Using this script, we are able to directly map the hand and finger bones of the rigged MakeHuman model to the tracking software, which allows us to display our hand motion on the virtual human.

Rationale: Through our research thus far, we have attempted to record animation clips using the Ultraleap provided rigged hand models then apply the animation to the human model. This worked to some extent, however, the x,y, and z orientations for some bones were different which resulted in incorrect direction of motion on the virtual human. As a result, we found that directly mapping the LMC to the MakeHuman model was more consistent, thus requiring an interface between the two.

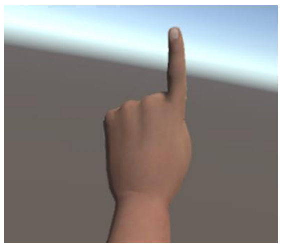

Figure 4: LMC Tracking directly to MakeHuman Model in Unity

7.2. Human Model Interface

7.2.1 MakeHuman Input Interface

For inputting the human models created in the Human Model Generation subsystem into the Unity environment, all models must be set to the default rig and exported as .fbx files. Using the mass produce plugin for MakeHuman, generate and export at least 20 unique virtual models to a “Models” folder within the “Assets” folder of the virtual environment. This will allow the Model Animation Subsystem to access the human model files for spawning and animation within the Unity environment.

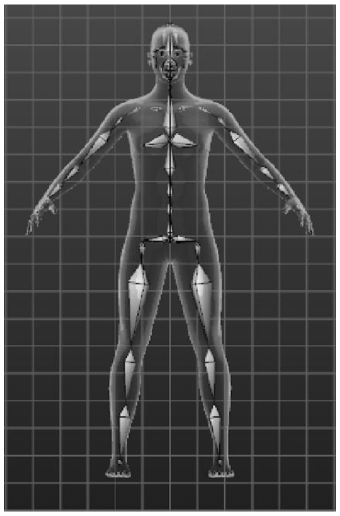

Figure 5: MakeHuman default rig for a human model

7.2.2 Human Model Animation Interface

After gesture recording is completed and a finalized animation dataset is created, it is necessary for the animations to be applied to randomly generated MakeHuman models. This shall be done by adding an “Animator” component to the lower arm bone of the MakeHuman rigged model. The behavior of the animator is determined by an animation controller, where each gesture animation clip is placed. When run, the model will perform the gesture recorded in the animation clip. This will be randomized to apply any given gesture on any given MakeHuman model.

7.3. Unity Camera Interface

The Data Capturing/Collection subsystem works concurrently with the Model Animation subsystem. As human models are animated, multiple cameras in front of the human model take images or videos at various angles. All images are taken as .jpg files.

7.4. Data Storage Interface

The images or videos taken by the cameras in Unity will be organized by dataset and by labeled gesture in the file explorer of the created Unity project. The script will save the full synthetic training set to a designated folder location. Each image will also be labeled according to the naming convention of the real dataset being replicated. For instance, in the Sign Language for Numbers dataset, gesture images of one are labeled as “one” followed by the image number, which is how our system will implement naming as well.

7.5. Full Body Animation Interface

In future integration, our system will be combined with a full body animation environment for a body gesture recognition system. The subsystems of the parent full body gesture system will take precedence over our hand gesture model subsystems except for the Gesture Data Collection subsystem. Our Gesture Data Collection subsystem will record hand gestures that will be spliced onto full body animations randomly. The combined animations will continue through the Model Animation and Data Capture/Collection subsystems of the body gesture recognition system, and the models used will come from the body gesture system's Model Generation subsystem.