Hand Gesture Recognition

Samuel Oncken, Steven Claypool

CONCEPT OF OPERATIONS

REVISION – 1
3 October 2022

CONCEPT OF OPERATIONS
FOR
Hand Gesture Recognition

TEAM <72>

APPROVED BY:

_____
Project Leader Date

_____
Prof. Kalafatis Date

_____
T/A Date

Change Record

Rev.	Date	Originator	Approvals	Description
-	9/15/2022	Samuel Oncken Steven Claypool		Draft Release
1	9/28/2022	Samuel Oncken Steven Claypool		1st Revision for FSR Release
2	11/25/2022	Samuel Oncken Steven Claypool		2nd Revision for Final Report Release

Table of Contents

List of Figures

1. Executive Summary

To create large training sets for neural networks, our solution is to create a virtual environment in Unity as opposed to using real humans and equipment. The virtual environment takes as input a real hand gesture dataset recorded by a user with the Leap Motion Controller, applies the gestures to hundreds of diverse human models from MakeHuman (imported into Unity), and records images and videos to build a virtual hand gesture training set. This virtual training set requires significantly less time and personnel and additionally will train the neural networks to similar if not improved gesture recognition accuracy when compared to existing real gesture training sets. Using our virtual environment to create synthetic gesture datasets, users can quickly and easily build reliable training sets tailored precisely to their application.

2. Introduction

Building neural network training sets for hand gesture recognition takes significant time and personnel to get diverse and comprehensive data. Thousands of images or videos must be taken manually to build a training set for a new set of hand gestures. To bypass this, we will create a virtual environment that builds training sets from gesture data recorded by one individual that is applied to hundreds of diverse human models and recorded. The neural networks would show similar accuracy in hand gesture recognition with the virtual training sets, potentially replacing the need for “real” training sets. This would notably expedite the creation of new training sets and simplify the entire process.

2.1. Background

Many scholarly articles covering the concept and feasibility of computer vision have been written in the last decade. However, it is no longer an argument that artificial intelligence and machine learning are here to stay. Through our research of the related work done in the field of gesture recognition, we have recorded a wide variety of methods to bring the idea of computer vision into light. In our project, we will not be focusing on the algorithms used in the creation of neural networks such as Convolutional Neural Networks (CNN) [1,2], but instead we will aim to uncover how the construction of a training set can alter the accuracy of a gesture recognition neural network.

Many of the datasets we have found online, summarized by [3], require a handful of real subjects to perform similar actions in front of a motion sensor (Kinect, Wii remote, etc.) which is recording from a set location. By using a number of different human hand models for the same gesture, some variance between gestures is collected which is required given that a gesture recognition system must not only recognize one size or shape of hand for it to function properly. Other datasets like the American Sign Language Lexicon Video Dataset and those derived from it [4] record human subjects using synchronized cameras all recording from different angles. Once again, this is beneficial to the machine learning process because in practice, a system will not always be looking at a human from a single angle.

Our research is aimed at bringing these important considerations together by generating a “virtual” dataset. By virtual, we mean that instead of having a set number of human subjects come into a studio to record movements, we will be recording the movements of one human (under various conditions such as lighting, distance from sensor, etc.) and applying those movements to randomly generated 3-D human models within the Unity game engine environment. After the virtual human model performs the gesture, we will record images and videos from several virtual camera angles to ensure that our training set aids in the recognition of a gesture from all directions. Similar research has been done using a virtual train set of hand gestures[5]. We are looking to expand upon this student’s research because creating a virtual training set is a much more cost-effective method of gathering data that still produces high accuracy results, which we are seeking to prove when testing our virtual datasets with state of the art hand gesture recognition neural networks.

A major obstacle in human gesture recognition is that reliable datasets are limited in the number of gestures they contain when compared to the vast number of gestures humans use every day. Our solution makes it fast and easy to create entirely new datasets consisting of application specific gestures, resulting in much more diverse gesture sets available for use and reducing the initial limitation.

2.2. Overview

We will design a virtual environment that can be used to create training sets for hand gesture recognition neural networks. This virtual environment will be implemented through Unity and will be able to map hand gestures recorded through the Leap Motion Controller onto randomly generated human models. We will be recording the gestures from a first-person point of view, mounted specifically on the head or chest of the individual performing the gestures.

The Unity environment will have a script that places virtual cameras at random locations in front of the human model performing the gesture (3rd person point of view) to collect varying angles of each gesture. The final script written for the Unity environment will generate and import a random human model, apply a randomly chosen “animation” from the real hand gesture database to the model, randomly place multiple virtual cameras in front of the model, then record and store the images taken as members of the final train set for the performed gesture. Finally, once the train set is produced, we will apply it to a benchmark, state of the art neural network for hand gesture recognition and analyze the accuracy achieved and compare it to the results using a standard, real gesture datasets. In comparing the recognition accuracy of a real gesture training set versus our synthetic (virtual) dataset, we will recognize whether using a virtual training set is a viable and effective method to train a hand gesture recognition neural network. We plan to create two separate training sets (comprised of different gestures) to train against two separate neural networks for increased confidence of results.

2.3. Referenced Documents and Standards

[1] J. Nagi *et al*., "Max-pooling convolutional neural networks for vision-based hand gesture recognition," *2011 IEEE International Conference on Signal and Image Processing Applications (ICSIPA)*, 2011, pp. 342-347, doi: 10.1109/ICSIPA.2011.6144164.

[2] J. Yu, M. Qin, and S. Zhou, “Dynamic gesture recognition based on 2D convolutional neural network and feature fusion,” *Sci Rep* 12, 4345, 2022. https://doi.org/10.1038/s41598-022-08133-z

[3] M. Asadi-Aghbolaghi *et al*., "A Survey on Deep Learning Based Approaches for Action and Gesture Recognition in Image Sequences," *2017 12th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2017)*, 2017, pp. 476-483, doi: 10.1109/FG.2017.150.

[4] C. Correia de Amorim, C. Zanchettin, “ASL-Skeleton3D and ASL-Phono: Two Novel Datasets for the American Sign Language,” *arXiv:2201.02065 [cs.CV]*, 2022 https://doi.org/10.48550/arXiv.2201.02065

[5] Z. Helton. *VR Hand Tracking for Robotics Applications.* May 2022, Texas A&M Electrical and Computer Engineering 404 Course Archives.

3. Operating Concept

3.1. Scope

The virtual environment can be used to generate various virtual hand gesture training sets for gesture recognition neural networks. These training sets can be composed of images or videos of the hand gestures. The scope of our project is limited exclusively to hand gesture training sets. With some adjustments to the environment, creating training sets for other body parts or full body gestures is possible as well.

3.2. Operational Description and Constraints

To create a virtual training set, the user must make or find a dataset of gesture recordings and import it to the environment. The environment will animate tens to hundreds of diverse human models according to the imported gesture recordings and take images on each to build the training set for a gesture recognition neural network.

The virtual environment to create gesture recognition training sets uses the Unity game engine and MakeHuman software, both of which are open-source. Users can find gesture datasets online but creating a new gesture dataset would require sensor equipment. In our research, we are using the Leap Motion sensor and tracking software which provides the tracking scripts that allow us to directly map bone transformations to virtual human model hands. We will be recording the transformation information including the position, rotation, and scale of each hand/finger component and exporting the files as animation clips, which are then applied to other (imported) MakeHuman virtual models to be used in data collection. Users of our research can use the same equipment or any other tracking technology with the ability to record rigged component transformations.

3.3. System Description

The system to create virtual training sets is made up of multiple subsystems: Gesture Data Collection, Human Model Generation, Model Animation, and Data Capturing/Collection.

The first subsystem is the Gesture Data Collection subsystem. Bone structure data of the hand of one individual is recorded using the Leap Motion Controller to create a dataset of related hand gestures, such as sign language. Once the gesture animations are recorded and stored properly, they are ready to be used in our complete dataset generation virtual environment.

The Human Model Generation subsystem is where virtual human models are created in MakeHuman and exported as .fbx files to be imported into Unity for future use. This system works in tandem with the Model Animation subsystem that uses the gesture recordings/animation clips to animate the models.

Within the completed dataset generation scene of our virtual environment is the Data Capturing/Collection subsystem that takes images of each model that gets generated, compiling the data into a training set. The subsystem uses a script to take the images or videos at randomized angles to collect comprehensive data of each gesture to ensure the neural networks are trained to acceptable accuracies in recognition.

Shown below is a flowchart depicting the relation of each subsystem.

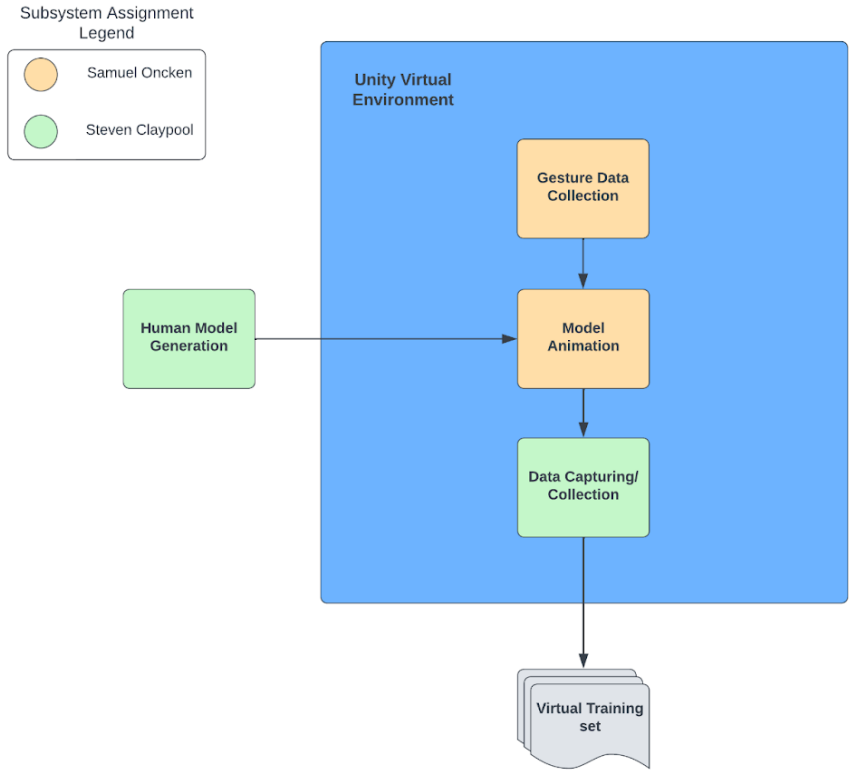

Figure 1: Flowchart of Virtual Hand Gesture Recognition Training Set Generation System

3.4. Modes of Operations

Since our project focuses on the validation of virtual datasets for hand gesture recognition neural networks rather than the creation of an operational tool, it does not necessarily have a mode of operation. However, the process of creating a train set using our virtual environment does contain stages: gesture recording and train set generation. During the gesture recording phase, a user must manually record his/her own hand gestures or find an online hand gesture dataset that they want to use. In the train set generation phase, a user will apply their hand gesture data to our virtual environment and the platform will automatically complete the train set using the imported gesture data.

3.5. Users

The users of our virtual hand gesture train set generation platform will be anyone who wants to build a dataset for their own applications. We will discuss a few scenarios in the coming section to exemplify this statement. Our environment as well as our tested virtual datasets will be available to download through GitHub. Once we validate through this research that using a virtual environment to create a train set results in as accurate (if not better) recognition than using real humans and equipment, users will be free to apply our platform to any human dataset that they wish to build or expand. They will be required to download Unity software

and have some knowledge working within Unity. Users will also be required to be able to create/find a gesture animation dataset to import into our platform as discussed previously, which might require additional software for their chosen sensor (i.e. Leap Motion Controller and Hand Tracking Software).

3.6. Support

Support will be in the form of a written manual which will detail the process of creating a training set using our virtual environment and the Leap Motion Controller. In addition, we will include our gesture animation clips (containing data of the hand gestures recorded with the Leap Motion Controller) and their respective virtual training sets, along with the gesture recognition neural networks we used. A document picturing types of gestures within the datasets will also be included. This can function as a benchmark for the users as they create their own training sets.

4. Scenario(s)

4.1. Software Developers Creating a Game

While creating a game that utilizes sensors such as a VR headset or Kinect, a developer wants to create a gesture dataset of movement controls for a computer vision model to recognize, but he/she does not have the time to find participants or equipment to record the gestures. Instead, the developer will use our virtual train set platform, where he/she will only need one camera to record each gesture. After recording the necessary hand gestures, the developer will apply them to our environment and be able to create an extensive train set with the reliability that a real, time-consuming train set would’ve produced.

4.2. Sign Language Translation

A user wants to create a dataset to train a model to recognize a distinct/less common form of sign language that does not have an existing dataset. He/she will use our virtual hand gesture train set generation platform to do this. First, the user must record the sign language gestures that he/she wants to implement, which requires only his/her movements (no need for large real human data collection). After importing the collected gesture data into our Unity virtual environment, the user will run our train set generation tool to output a large hand gesture dataset, composed of hundreds of unique human models and photographed from many distinct virtual camera angles. The user can then apply the virtual train set into their sign language recognition neural network.

5. Analysis

5.1. Summary of Proposed Improvements

When compared with the traditional method of collecting “real” gesture data, virtually creating a train set will have a variety of requirements/improvements including:

- Requires only one human to perform each gesture. We do not require paid participants, reducing cost and time for multiple parties.

- Utilizes virtual cameras within the Unity environment, allowing us to store hand gesture data from numerous angles while only recording from one stationary camera location in real life.
- Utilizes the bone/joint location detection within the Leap Motion Hand Tracking software, allowing us to map accurate gesture animations within Unity.
- Applies gesture animations to hundreds of randomly generated human models allowing for the neural network to train using comprehensive data of varying body structures, skin tones, etc.
- Human data recording is not always legal. At a minimum, human data collection requires some amount of paperwork for each participant. Using our research, we reduce legal risk.

5.2. Disadvantages and Limitations

Within our project, we can think of a few potential limitations that include:

- There is one person performing a gesture, then the recorded animation is applied to different human models. As a result, the recognition accuracy for a certain hand gesture might change depending on how a person may perform the gesture differently from others.
    - This can be mitigated by recording multiple versions of the same gesture to account for the randomness in the way a gesture can be performed. Another method is to use one or two more participants to perform each gesture. In both methods, a random version of the gesture recording would be applied to the virtual human model at run time.
- The Leap Motion Controller might not accurately depict the hand gesture being performed by a user into Unity. As a result, a user might have to record his/her gestures multiple times or from different angles to provide the best representation of the real gesture before proceeding to take images and videos for the train set.

5.3. Alternatives

Alternatives to our virtual hand gesture train set generation platform include:

- Traditional process as it stands now: creating a training set using “real” data. Different people perform a gesture while being recorded from either a mounted camera or a variety of cameras. Images and videos of the physical actions are then used to train a neural network. Time consuming and resource intensive.
- Different methods to create virtual datasets. We could use a different game engine software, human model generating software, and motion sensor/tracking software for recording gestures.

5.4. Impact

Our project has a significant influence on society in technological advancement and availability.

By validating the process of creating virtual training sets for neural networks in hand gesture recognition, the possibilities for future virtual training sets expands greatly. Developers could use our research to create diverse gesture recognition training sets much more easily than before. They could also use our research to create new virtual environments for different applications beyond hand gesture recognition.

Through our research, the availability of training set data would drastically increase. As more people begin to use our virtual environment to create training sets, more diverse training sets of certain gesture datasets will be readily available online without needing to find the money and participants to gather data of real individuals. This will also boost the technological advancement of machine learning and AI as more research with training neural networks can be done on much lower budgets.

In addition, our research has certain legal impacts as mentioned previously. Without the need for large amounts of human participants, the creation of a virtual human training set requires far less paperwork and legal permissions to use private data of the participants since only one human is required to perform each gesture that will eventually be applied to a large number of virtual humans.

Hand Gesture Recognition

Samuel Oncken, Steven Claypool

FUNCTIONAL SYSTEM REQUIREMENTS

REVISION – Draft
3 October 2022

FUNCTIONAL SYSTEM REQUIREMENTS
FOR
Hand Gesture Recognition

PREPARED BY:

Team 72 10/3/2022
Author Date

APPROVED BY:

_____
Project Leader Date

_____
John Lusher, P.E. Date

_____
T/A Date

Change Record

Rev.	Date	Originator	Approvals	Description
-	10/3/2022	Samuel Oncken Steven Claypool		Draft Release
1	11/29/2022	Samuel Oncken Steven Claypool		1st Revision for Final Report Release

Table of Contents

List of Tables

List of Figures

1. Introduction

1.1. Purpose and Scope

Collecting the large amount of gesture training set data required to teach a hand gesture recognition neural network is time consuming and resource intensive. Our aim is to establish a new method of training set generation using virtual data that takes significantly less time, human participants, and money than the traditional routine. Through our research, we shall develop a platform that takes as input any user recorded gestures (through an LMC) and outputs a full training set, consisting of hundreds of pictures of each gesture (taken from random camera angles) on unique virtual human models.

Additionally, we seek to understand whether a virtual training set can teach a hand gesture recognition neural network to similar if not improved performance when compared to real gesture training set. We shall do this by building at least two virtual training sets that include the exact gestures of two existing, real training sets, then comparing the performance metric of each when used in the same two hand gesture recognition neural network.

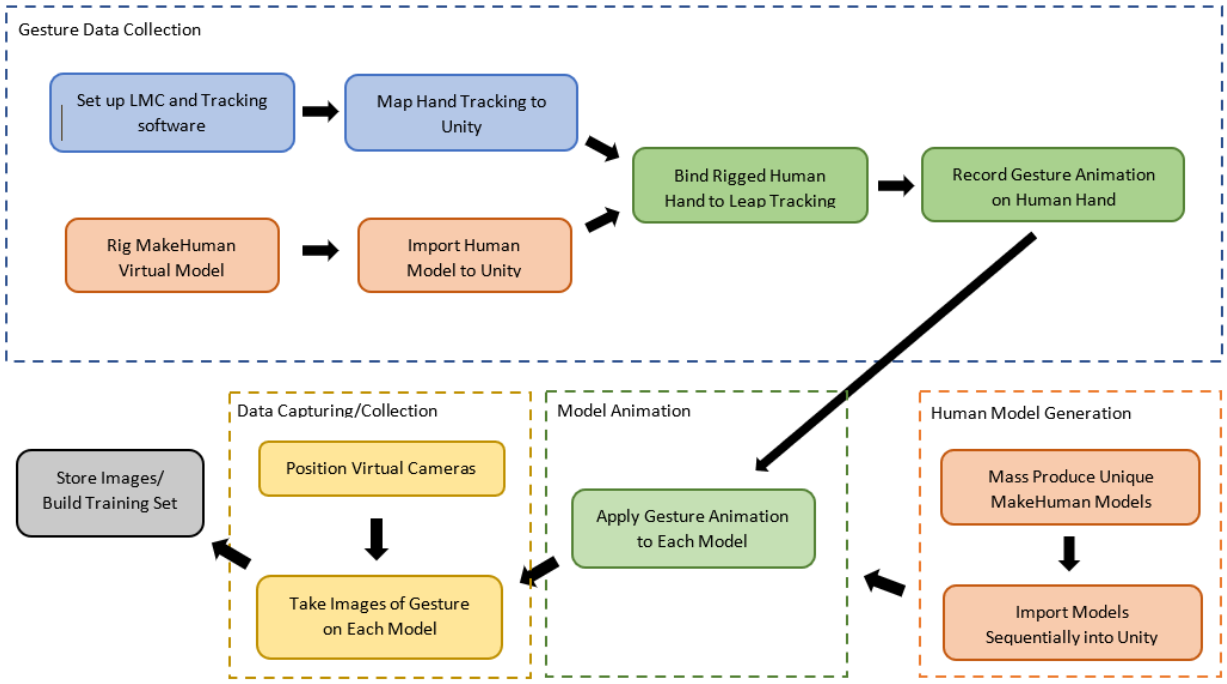

Figure 1. Project Conceptual Image

1.2. Responsibility and Change Authority

The team leader, Samuel Oncken, is responsible for validating that all requirements of this project have been met. If changes to the project are necessary, they will only be carried out after the approval of each team member, project sponsor Professor Stavros Kalafatis, and secondary project sponsor Pranav Dhulipala.

Table 1: Project Subsystem Responsibilities

Subsystem	Responsibility
Gesture Data Collection	Samuel Oncken
Hand Model Generation	Steven Claypool
Model Animation	Samuel Oncken
Data Capturing/Collection	Samuel Oncken

2. Applicable and Reference Documents

2.1. Applicable Documents

The following documents, of the exact issue and revision shown, form a part of this specification to the extent specified herein:

Table 2: Applicable Documents

Document Number	Revision/Release Date	Document Title
UH-003206-TC	Issue 6	Leap Motion Controller Data Sheet

2.2. Reference Documents

The following documents are reference documents utilized in the development of this specification. These documents do not form a part of this specification and are not controlled by their reference herein.

Table 3: Reference Documents

Document Number	Revision/Release Date	Document Title
N/A	Version 2021.3 09/23/2022	Unity User Manual 2021.3 (LTS)
N/A	2021	Ultraleap for Developers - Unity API User Manual

2.3. Order of Precedence

In the event of a conflict between the text of this specification and an applicable document cited herein, the text of this specification takes precedence without any exceptions. Any requirements by the sponsor takes precedence over all specifications and documents.

All specifications, standards, exhibits, drawings or other documents that are invoked as “applicable” in this specification are incorporated as cited. All documents that are referred to within an applicable report are considered to be for guidance and information only, except ICDs that have their relevant documents considered to be incorporated as cited.

3. Requirements

3.1. System Definition

The hand gesture recognition system allows users to easily create extensive and diverse hand gesture training sets using a virtual environment. This avoids the need for hundreds of participants for gesture images and videos while still training neural networks to equivalent recognition accuracy. The system has four subsystems: Gesture Data Collection, Human Model Generation, Model Animation, and Data Capturing/Collection.

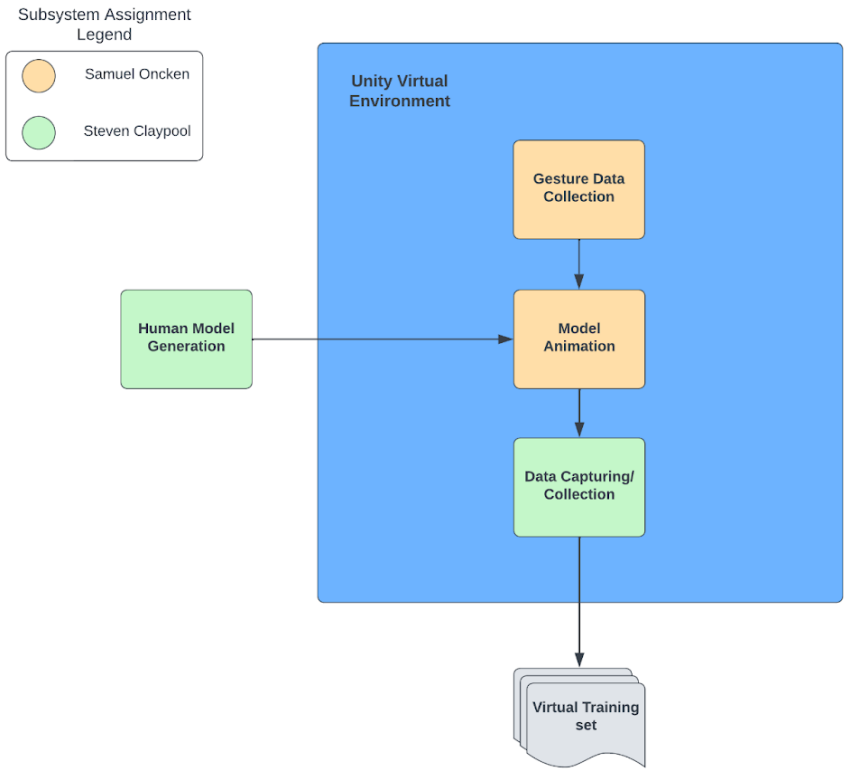

Figure 2. Block Diagram of System

Our Unity environment shall consist of 2 separate scenes. The first scene will house the Gesture Data Collection subsystem, where hand gestures will be recorded using the LMC (mapped to an example MakeHuman model) in Unity and saved as animation clips. Each of these gesture recordings will be accessible to the second scene components and stored to form an animation dataset.

The second scene will house the remaining subsystems. First, the Human Model Generation subsystem will randomly create tens/hundreds of diverse human models and import each sequentially to the Unity environment. This leads to the Model Animation subsystem, which takes a random hand gesture from the animation dataset and applies it to the spawned human model. As this model performs the gesture, the

Data Capturing/Collection subsystem takes images from a randomized virtual camera angles (centered around palm of hand) and compiles the recordings into complete virtual training set folders by name.

3.2. Characteristics

3.2.1. Functional / Performance Requirements

3.2.1.1 Gesture Recognition Accuracy

The metric of performance for the neural network trained with a virtual training set shall be equivalent to the performance metric when using the benchmark training set. We have decided that the gesture recognition accuracy when training using our synthetic data (or combinations of real and synthetic data) shall be no more than 5% below the metric when using the benchmark real training set.

> *Rationale: For a comparison to be valid, the performance metric of each test must be the same. The gesture recognition accuracy with the virtual training set must be roughly equivalent or better than the accuracy with the benchmark training set to validate the usage of virtual training sets over real training sets.*

3.2.1.2. Computer Requirements

The computer running the environment shall use Windows 7+ or Mac OS X 10.7+ with X86 or X64 architecture CPU, a dedicated GPU, 2 GB of RAM, and a USB 2.0 port.

> *Rationale: Specifications provided by LMC datasheet and Unity 2021.3 manual.*

3.2.2. Physical Characteristics

3.2.2.1. LMC Mounting Position

The LMC will operate in a head mounted position to record hand gesture motions.

> *Rationale: We tested various LMC mounting positions and found that it most accurately records the hand structure data of gestures when mounted on the head directed at the hands.*

3.2.2.2. Head Mounting

The head mounting apparatus shall be able to hold 32g of weight.

> *Rationale: The LMC weighs 32g, which is the only item being mounted.*

3.2.3. Electrical Characteristics

3.2.3.1. Input

3.2.3.1.1 LMC Power Input
The LMC requires a 5V DC input with a minimum of 0.5A via USB.

Rationale: Specification provided by LMC datasheet.

3.2.3.2. Output

3.2.3.2.1 Data Output
The system shall output a complete virtual hand gesture training set, composed of images in the file format of .jpg. Images can also be stored as .png files, but that must be changed in the code of the SceneController script. The input file format of the used gesture recognition neural network must be equivalent to the data output produced by our system.

3.2.3.3. Connectors
The Hand Gesture Recognition system shall use a USB-A to Micro-b USB 3.0 cable to connect the LMC to the computer.

3.2.3.4. Wiring
Not applicable to our research.

3.2.4. Software Requirements

3.2.4.1 Neural Networks with TensorFlow
The neural networks used shall work with TensorFlow

Rationale: TensorFlow is a free, open-source software library commonly used in the training of neural networks. We were instructed to use TensorFlow by co-sponsor due to its high accessibility, ease of use, and large support network.

3.2.4.2 Virtual Environment - Unity
The virtual environment used shall be any version from Unity 2021.3 and newer.

Rationale: Unity is a free game engine software that allows for easy use of scripting within the created environments. Additionally, Ultraleap provides Unity specific plug-ins and documentation/support for the use of the LMC hand tracking software.

3.2.4.3 MakeHuman
The MakeHuman software shall be the latest stable version (1.2.0)

Rationale: MakeHuman offers a mass produce function that allows us to import randomly generated human models with correct bone structure for the hand

gesture animation to run. We will be using the latest stable version to avoid any potential incompatibility.

3.2.4.4 Leap Motion Controller

The Leap Motion Controller shall use Gemini - Ultraleap's 5th Generation Hand Tracking software

Rationale: This is the latest version of hand tracking software offered by Ultraleap. It offers the highest quality tracking data and large amounts of usage/support documentation.

3.2.4.5 Unity Plugins

The Ultraleap, Animation Rigging and Unity Recorder Unity plugins are needed to create and run the Unity virtual environment.

Rationale: Ultraleap Unity Plugin includes required scripts for hand tracking/binding as well as useful prefabs of fully rigged hand models for testing. The Unity Recorder is used in the gesture recording process to track bone transform data and export the animation as a usable animation clip file for application on future models. The Animation Rigging package offers a Bone Rendering option which displays bone position on the character model, allowing for easy manipulation of transform data and access to specific bones.

3.2.4.6 MakeHuman Plugin

The MakeHuman "Mass Produce" plugin is needed for large-scale model generation.

Rationale: The plugin is necessary as a means to automatically mass produce diverse human models for animating in the Unity environment.

3.2.5. Environmental Requirements

3.2.5.1 Lighting

There shall be adequate lighting when recording hand gestures with the LMC.

Rationale: The LMC loses gesture recording accuracy in darker conditions, reducing virtual hand gesture quality and therefore decreasing gesture recognition neural network accuracy.

3.2.5.2. LMC Operating Conditions

The LMC Shall be operated in consideration of the following constraints.

Rationale: Operating conditions are provided by the LMC datasheet.

3.2.5.2.1 Temperature

The LMC shall be operated from 32° to 113° F.

3.2.5.2.2 Humidity

The LMC shall be operated at a humidity between 5% to 85%.

3.2.5.2.3 Altitude

The LMC shall be operated at altitudes between 0 to 10,000 feet.

3.2.6 Failure Propagation

Not applicable to our research

4. Support Requirements

Users of the virtual environment to build virtual hand gesture training sets will require a computer with a gigabyte of storage (more or less depending on the size of the training set being built) and enough computational power (CPU and dedicated GPU) and a display to run the system well. Users must provide power to the computer. The virtual environment is provided along with all the scripts necessary to run the subsystems. A sample neural network and training set will also be provided to act as a benchmark. Any technical issues should be resolved by referencing any specification datasheets, manuals, or by contacting the software companies directly.

Appendix A: Acronyms and Abbreviations

LMC	Leap Motion Controller
CPU	Central Processing Unit
GPU	Graphics Processing Unit

Appendix B: Definition of Terms

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.
Game Object	Building blocks of a Unity environment. These are blank slates which can be characters, objects, or environments that can be manipulated by adding components such as scripts for actual functionality to occur.
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.

Appendix C: Interface Control Documents

Interface Control Documentation is provided in a separate document.

Hand Gesture Recognition

Samuel Oncken, Steven Claypool

INTERFACE CONTROL DOCUMENT

REVISION – Draft
3 October 2022

INTERFACE CONTROL DOCUMENT
FOR
Hand Gesture Recognition

PREPARED BY:

Team 72 10/3/2022

_____

Author Date

APPROVED BY:

_____

Project Leader Date

_____

John Lusher II, P.E. Date

_____

T/A Date

Change Record

Rev.	Date	Originator	Approvals	Description
-	10/3/2022	Samuel Oncken Steven Claypool		Draft Release
1	11/29/2022	Samuel Oncken Steven Claypool		1st 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.

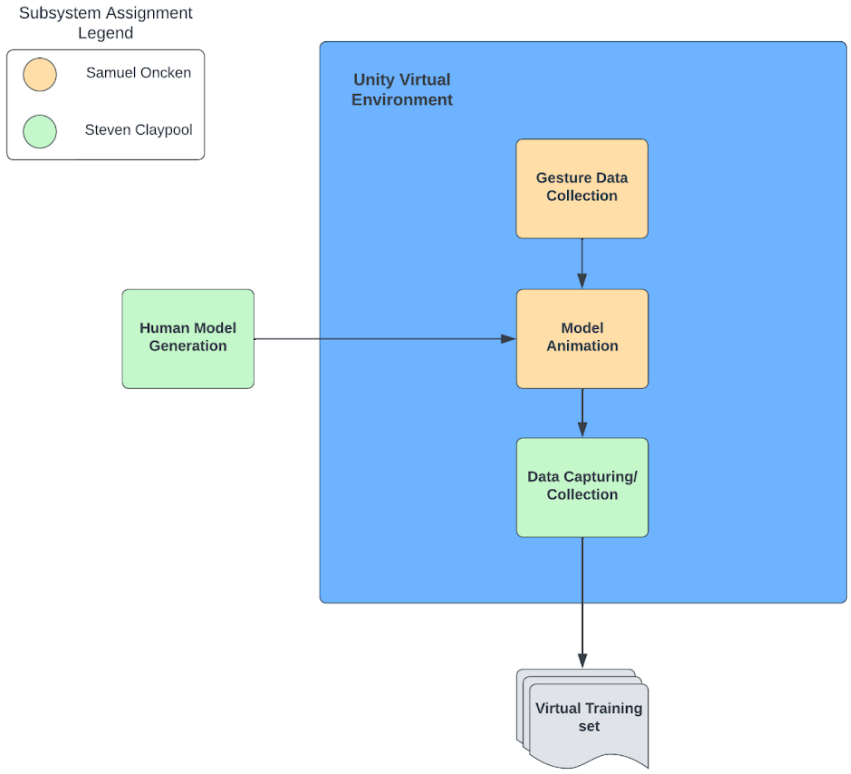

Figure 1: Block Diagram of subsystem interaction

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.

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.

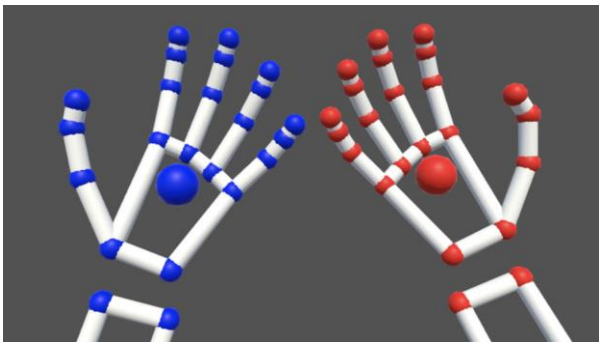

Figure 2: Ultraleap Rigged Hand Model in Unity

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 .fbx 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.*

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.

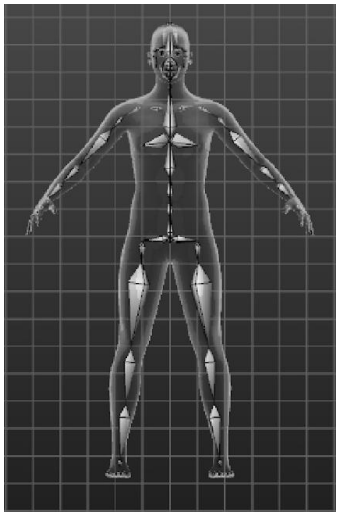

Figure 3: MakeHuman default rig for a human model

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 gesture in the file explorer of the created Unity project. The script will save the full synthetic training set to a designated folder location.

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.

Hand Gesture Recognition

Samuel Oncken, Steven Claypool

EXECUTION AND VALIDATION PLAN

REVISION – Draft
3 October 2022

Change Record

Rev.	Date	Originator	Approvals	Description
-	10/3/2022	Samuel Oncken Steven Claypool		Draft Release for FSR
1	11/25/2022	Samuel Oncken Steven Claypool		Validation Alterations for Final Presentation/Report

VALIDATION PLAN

Table 1: 403 Validation Plan

Test Name	Success Criteria	Methodology	Status	Responsible Engineer(s)
Benchmark Dataset Training	Gesture recognition neural network can run on our home computer and train using real dataset. Results quantified	Download the benchmark dataset and code for the CNN. Run the code and confirm similar accuracy to benchmark logs provided	TESTED – Pass	Steven Claypool
Virtual Dataset Training	Gesture recognition neural network can train using our built dataset and provide accuracy results	Take a final virtual dataset modeled after a real benchmark dataset and use it to train the same CNN as the benchmark. Obtain and compare recognition accuracy to real dataset training run.	TESTED - Pass	Steven Claypool
Gesture Recognition Accuracy	Accuracy is within 5% of benchmark accuracy using real dataset	Train gesture recognition neural network using real and synthetic sets and compare accuracy	TESTED - Pass	Steven Claypool
Synthetic Train on Real Test accuracy	Test real data with CNN trained on synthetic data and achieve accuracy similar or improved to benchmark train results	Train a CNN using different methods/compositions with synthetic and/or real data and test using real data	IN PROGRESS	Steven Claypool
Unity Hand Mapping	Real hand movement from LMC is mapped using Ultraleap prefab hand models in Unity	Set up Unity environment, install Ultraleap plug-ins, familiarize with Ultraleap scripts/prefabs, read Ultraleap Unity API documentation, and map hands.	TESTED - Pass	Samuel Oncken
Import a Rigged MakeHuman Model	A fully rigged MakeHuman model is imported into Unity with specific materials/textures extracted	Configure MakeHuman model with “Default” rig and desired appearance. Export model as .fbx file and import model into Unity. Make sure all bones are aligned properly and can be manipulated through space. Approve that model mesh (appearance) is as desired.	TESTED - Pass	Steven Claypool
Virtual Model Unity Hand Mapping	Map hand movement onto an imported rigged MakeHuman model with natural looking movements	Import a rigged MakeHuman model, access the wrist bone and add a hand binder component. Fix advanced issues such as bone orientation and mapping accuracy. Make sure user movement is accurate to movement of the model (without unnatural joint movements).	TESTED – Pass	Samuel Oncken
Mounting Stability	Head mounted LMC remains intact with mounting apparatus even if head motion is present	Apply LMC to the mounting apparatus and plug the device into the computer. Rotate head in all directions (looking up, down, left, right) and shake head left to right. Make sure LMC is still firmly in place.	TESTED - Pass	Samuel Oncken

Gesture Recording	From within the Unity environment, gesture animation clips can be produced and stored for application on future models	From within the scene used to validate the Unity hand mapping, download the Unity Recorder from the package manager in Unity. Choose output file name and destination, exporting as an animation clip, recorded from the lower arm bone of the virtual model	TESTED – Pass	Samuel Oncken
Apply Example Animation to Model	MakeHuman model is able to perform an imported full body gesture accurately.	Visit Mixamo.com, download a .fbx animation file from the choices listed. Apply the animation to the rigged human model using an Animator component to validate that the rigged structure is able to move as intended.	TESTED - Pass	Samuel Oncken
Apply Recorded Gesture Animation to Model	Freshly imported rigged MakeHuman model is able to perform the gesture animation as recorded in the Gesture Data Collection subsystem	After recording a gesture animation, apply it to a freshly imported MakeHuman model using the Animator component located in the model's lower arm bone	TESTED - Pass	Samuel Oncken
Create and Import Rigged MakeHuman Models to Unity	Minimum 30 MakeHuman models are generated uniquely and imported to Unity environment	Use MakeHuman “mass produce” function to generate unique character models, each fit with a “Default” rig. Produce around 20% of models with edge/corner case metrics (gloves, unnatural colors, etc.)	TESTED - Pass	Steven Claypool
Data Capture Output and File Type	Virtual camera outputs image data as a .png file or video data as mp4 (TBD from neural networks used).	After images or videos are taken of a model performing a gesture, validate that the images are stored, organized, and are of the desired file type.	TESTED – Pass	Samuel Oncken
Final System Validation	With the press of a button, a large and diverse virtual training set is produced	Run system and validate in output files that each gesture has at least 500 images of gesture performance on differing human models from numerous camera angles	TESTED – Pass	Samuel Oncken

Execution Plan

Table 2: 403 Execution Plan Status Indicator Legend

Complete	
In progress	
Planned	
Behind Schedule	

Table 3: 403 Execution Plan

Task	Deadline	Responsibility	Status
Develop research outcomes/goals	9/7/2022	All	
Get familiar with LMC, Unity, and MakeHuman	9/14/22	All	
Write ConOps	9/15/22	All	
Determine optimal mounting position of LMC with testing of tracking accuracy	9/21/22	Samuel Oncken	
Generate 4 test MakeHuman models, instantiate them into Unity	9/21/22	Steven Claypool	
Map movement of hands onto Ultraleap hand prefabs in Unity	9/28/22	Samuel Oncken	
Add rig settings and animator component to humans in Unity, test online example animation on models	9/28/22	Samuel Oncken	
Set virtual cameras in Unity to face human model upon spawn	9/28/22	Steven Claypool	
Write FSR, ICD, Execution and Validation plans	10/3/2022	All	
Map real hand movements onto virtual human model and record a gesture animation	10/5/2022	Samuel Oncken	
Research Image Synthesis for Gesture Recording process	10/5/2022	Steven Claypool	
Find at least 2 real gesture datasets, tested against SOTA gesture recognition neural networks for us to replicate	10/5/2022	Steven Claypool	
Midterm Presentation	10/12/2022	All	
Record all gesture animations for the replicated virtual gesture datasets	10/12/2022	Samuel Oncken	
Build Unity environment with multiple virtual cameras angled at imported model hand	10/12/2022	Samuel Oncken	

Apply recorded animation to a virtual model	10/19/2022	Samuel Oncekn	
Generate human models using mass produce and export each as a .fbx file to import into Unity	10/19/2022	Steven Claypool	
Find gesture recognition neural network for each dataset and test on machine	10/26/2022	Steven Claypool	
Status Update Presentation	10/27/2022	All	
Complete Unity environment; spawn unique model, apply gesture, record image and store into dataset specific folders	11/4/2022	Samuel Oncken	
Set up Unity environment on WEB 156 lab computer – generate virtual datasets for sign language for numbers and alphabet	11/7/2022	All	
Preprocess training sets and begin training the gesture recognition neural networks using virtual datasets	11/11/2022	Steven Claypool	
Test various compositions of training set data. Combinations of real/synthetic data for NN training, test on more real data	11/18/2022	Steven Claypool	
Train using various compositions of real/synthetic data and test on pictures of our own hands	11/25/2022	Steven Claypool	
Explore future datasets for replication next semester (full body animations/full body images) ex: HaGRID	11/25/2022	All	
Final Demo/Presentation	12/1/2022	All	
Complete Final Paper	12/4/2022	All	

Note: The training/testing process using our virtual data has been much more time consuming and difficult than expected by our sponsor, hence the training being behind schedule. Steven plans to continue training and testing into winter break to explore the most optimal use of our virtual training sets when it comes to gesture recognition accuracy.