Hand Gesture Recognition

Samuel Oncken, Steven Claypool

FUNCTIONAL SYSTEM REQUIREMENTS

REVISION - 2
3 October 2022

FUNCTIONAL SYSTEM REQUIREMENTS
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, 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 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.

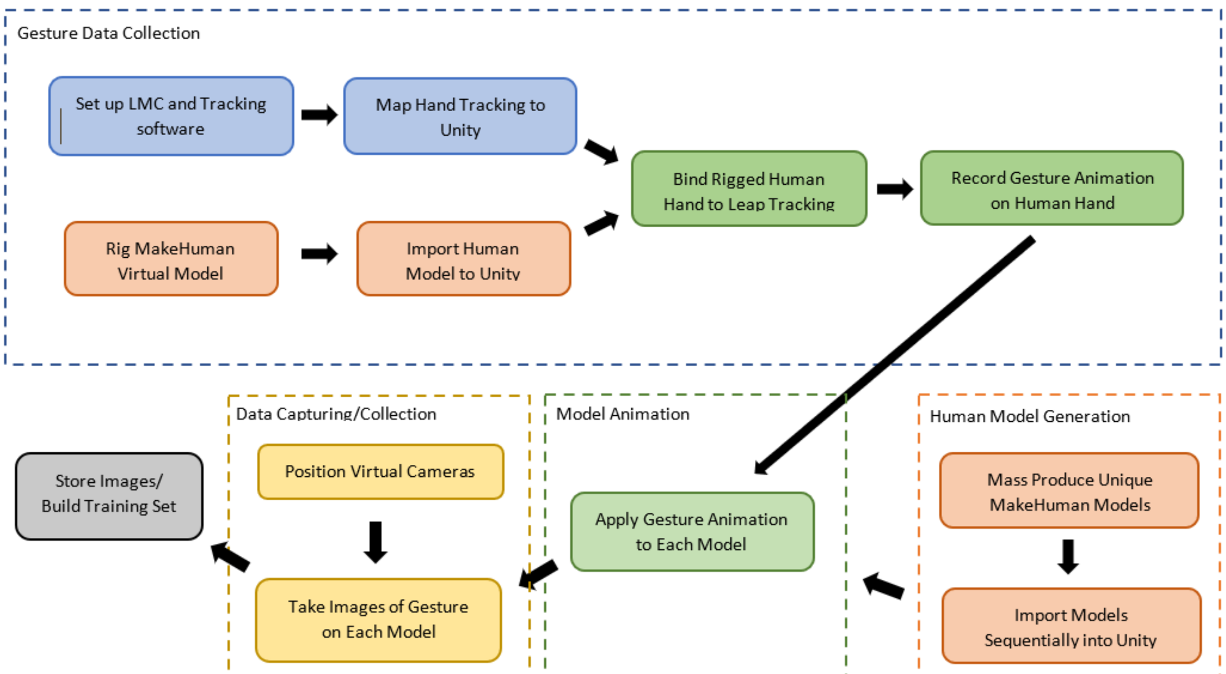

Figure 1. Project Conceptual Image

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
Human 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.

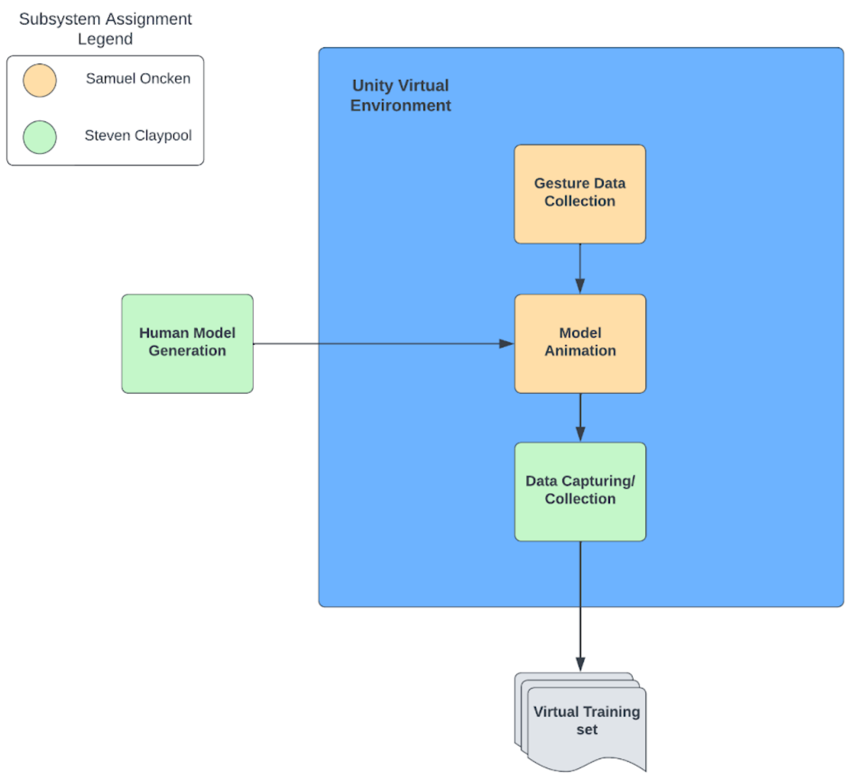

Figure 2. Block Diagram of System

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 2.8.0
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 randomly generated human models. We will be importing models with a Default*

rig, which includes all hand and finger bones 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.

3.2.4.7 Image Classifiers Python Package

The latest version (1.0.0b1) of PyPI's image-classifiers python package is needed for loading pre trained neural networks for use.

Rationale: The Keras API in TensorFlow is missing some useful image classification models that PyPI's package has, including models we used such as ResNet18.

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.

In the virtual environment, lighting shall be manipulated and randomized while performing and recording animations.

Rationale: By varying lighting conditions, we can achieve higher gesture recognition accuracy for more edge cases in testing where images might be darker or brighter than expected.

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 according to the dataset. This will simply be room temperature in our case as we will be using the LMC in a controlled environment.

3.2.5.2.2 Humidity

The LMC shall be operated at a humidity between 5% to 85%. Again, we will operating the LMC in normal controlled room conditions, which on average is around 35% humidity.

3.2.5.2.3 Altitude

The LMC shall be operated at altitudes between 0 to 10,000 feet. We will be operating the LMC in College Station, which is well within this range.

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.
Neural Network	Combinations of layers and algorithms that operate similar to a traditional biological definition of the human brain in order to recognize patterns and relationships between large amounts of data.
Virtual Environment	Within our research, a virtual environment is defined as the representation of real-world features such as humans and structures from a virtualized Unity project. All humans are computer built models which are animated to resemble real human movement.

Appendix C: Interface Control Documents

Interface Control Documentation is provided in a separate document.