| **Sign Language Interpretation System** | | |
|---|---|---|
| Team Information | 1. Braden Bagby<br>2. David Gray<br>3. Robert Stoner<br>4. William Hughes<br>5. Zach Langford | |
| Document Type: Software Requirement Specification v1.0 | | *Last updated on*: 09-14-2020 |

**Document Approval**

This Software Requirements Specification has been accepted and approved by the following stakeholders:

| **Printed Name** | **Title** | **Signature and Date** |
|---|---|---|
| | | |
| | | |
| R. IQBAL | Customer/Course Instructor | |

# Contents

# 1. Introduction

## 1.1 Purpose

This document was written to provide an overview of the sign language interpretation system and to outline its requirements and its goals. It is written as a communication device for developers and testers as a means of defining how the system is expected to perform and the standards expected from the stakeholders to fulfill their needs.

## 1.2 Scope

The sign language interpretation system will take a video stream input from an integrated camera and interpret hand gestures into pre-defined commands. It will recognize the gestures with high accuracy in real time. The system will be tested with a smart home interface connected to the program output, which would be a valid command that is interpreted from the sign language input.

The system will act as a means of processing hand sign communication for the potential use by individuals with hearing impairments or other disabilities that require visual communication devices for interacting with technology. The actualization of this product will provide a way for more people to intuitively access technology that was previously diffcult, or impossible, to use. The primary goal of the sign language interpretation system is to bridge those gaps.

## 1.3 Definitions, Acronyms, and Abbreviations

**System**: References the overall core product.

**Gesture**: A specific hand movement that the gesture interpretation module will interpret into an actual letter or number. Ex: The ASL hand positioning for the letter A.

**Noise**: Every frame of the video stream that does contain a recognizable hand or hand gesture.

**Sequence**: A collection of individual gestures that can represent a command or phrase.

**Commands**: Specific functions that can be called by a sequence (eg. Turning on lights).

**Local Environment**: The hardware that is used to receive initial data input from the user, and the software for recieving the video feed and providing object detection and hand tracking (Camera, PC, local softwware, etc.).

**Cloud Environment:** The remote environment that will preform all of the complex computational processing (Applying machine learning (ML) algorithms).

**Real Time:** When an action produces a result between 50 milliseconds and 500 milliseconds after.

**Hand Tracking Data**: A collection of data on the positions of twenty-one different points on the wrist, palm, and fingers that are within the camera view in the video stream.

**Hand Tracking Module**: The module in the local environment that detects if a hand is on screen, and if that hand is moving significantly enough to send the hand tracking data back to the gesture interpretation module.

**Gesture Interpretation Module**: The module on the cloud environment that receives hand tracking data from the hand tracking module, and attempts to interpret it as a valid and recognizable sign language gesture.

**Formal Grammar Module:** The module in the cloud environment that makes sense, via natural language processing, of the letters, numbers, and words output from the gesture interpretation module.

**Command Output Module**: The module in the local environment that receives the output commands from the cloud environment and gives them to a smart home device connected to the local environment.

## 1.4 References

1. Bhuyan, M. K., Ghosh, D., & Bora, P. K. (2006). Hand motion tracking and trajectory matching for dynamic hand gesture recognition. Journal of Experimental & Theoretical Artificial Intelligence, 18(4), 435–447. doi:10.1080/09528130600975931
2. Abid, M. R., Petriu, E. M., & Amjadian, E. (2015). Dynamic Sign Language Recognition for Smart Home Interactive Application Using Stochastic Linear Formal Grammar. IEEE Transactions on Instrumentation and Measurement, 64(3), 596–605. doi:10.1109/tim.2014.2351331

## 1.5 Overview

This SRS document contains three major sections beyond this introductory section, which outlines the purpose, scope, and defintions of the project. First, the General Description section goes over very basic and non-concrete aspects of the project. It includes a perspective on the project in regards to itself and any external systems, a guide to basic project functionalities as gathered from stakeholder meetings, characteristics of intended users, and general constraints, external dependencies, and assumptions about features of the project. The Specific Requirements section outlines all the actual requirements for this project, including functional requirements, non-functional requirements, and constraints. Some requirements are broken into further sections, such as interfacing and database requirements. The final section, Change Management Process, outlines the procedures for changing any requirements for this project after the project has started.

# 2. General Description

## 2.1 Product Perspective

This sign language interpretation system is not intended to be a self-contained system. The output of the entire system, command phrases interpreted from gesture sequences, will be able to be used as an input to smart home devices, such as Amazon's Alexa or Google's Google Home. There could be many other larger systems besides smart homes that our system could be integrated with if one of the desired effects was to output gestures as words, phrases, or commands. For example, one could potentially integrate our system into a larger computer application or a website .

This system takes a live video stream as input from a camera connected to a computer. The computer is used to detect when a hand is in frame and tracks the hand, identifying twenty-one points on the wrist, palm, and fingers. The hand tracking data is sent from the local environment to the cloud environment where the data will be processed by the gesture interpretation module into sequences of letters and numbers. These sequences are then passed to the formal grammar module for natural language processing resulting in grammatically correct words and phrases to be sent back to the local environment and translated into commands by the command output module. The equipment we will use will be the webcam within the local environment to record a video stream, and a cloud computing server (or possibly a Raspberry Pi or other PC) to host the gesture interpretation module and formal grammar module.

## 2.2 Product Functions

This sign language interpretation system has a few basic functionalities for each module that follow a linear fashion. First, the hand tracking module should have these basic functionalities:

- Ability to detect whether or not a hand is within the camera's view
- Ability to track hand and digit position and movement
- Ability to send hand tracking data to the gesture interpretation module

Second, the gesture interpretation module should have these functions:

- Ability to accurately distinguish alphabetic and numeric sign language gesture sequences from hand tracking data
- Ability to output alphabetic and numeric characters from gesture sequences
- Ability to receive hand tracking data from the hand tracking module
- Ability to perform the previous functions in real-time

Third, the formal grammar module, should be able to perform the following functions:

- Ability to receive sequences of characters from interpreted gestures
- Ability to determine whether these sequences are actual commands, even if some of the information from the command is missing (For example: The module receives an input of "LIG?T ON", where "?" is a gesture that could not be interpreted, and the formal grammar module will be able to predict that the correct command was "LIGHT ON")
- Ability to output interpreted commands (words and phrases) to a console. This functionality will aid in the testing and debugging process.

- Ability to send interpreted commands to the command output module

Fourth, the command output module should have the following functions:

- Ability to receive commands from the formal grammar module
- Ability to tailor commands into the correct API for the local device to understand
- Ability to perform commands

## 2.3 User Characteristics

This system was initially conceived as a tool that could be used by the sick, bed-ridden, or disabled who are not able to speak, and thus are not able to use a traditional smart home device with their voice. This is the driving force behind the idea of a gesture detection system that can easily be integrated into a few smart home systems. The system must be adaptable for various smart home systems, and customizable to a variety of commands. We also must consider any future developers working on this project as users, who may be adding to this project or performing maintenance. As such, we will be integrating a video stream debugging output for the hand tracking module and a console output for the formal grammar module.

## 2.4 General Constraints

There are a few constraints on the sign language interpretation system. The largest constraints on the system are that the gesture interpretation must be performed with 90% accuracy and in real-time as the user performs the gestures. We must consider constraints on the local environment the system will be obtaining a video feed from. We must ensure that we still have the required accuracy, even in environments with less optimal backgrounds (more noise in the image) or dim lighting. We must also consider the resolution of our camera that we intend to use. Currently, we are going to use a camera with a minimum resolution of 480p. The internet network connection for users must also be of a minimum speed and bandwidth in order to use our system. Similarly, our cloud service machine that hosts our system must have enough processing power to perform all of its functionalities in real-time.

## 2.5 Assumptions and Dependencies.

**Setup**: It is assumed that the person setting up the sign language interpretation system would be able to deploy the project onto a computer and attach a camera to the system, as well as attach the output of the system to a smart home device.

**Sign Language**: It is assumed that the user of the system would know basic alphabetic and numeric gestures in American Sign Language (ASL).

**Cloud Services**: Our system will utilize a reliable cloud service.

**Hand Gesture Library**: We will be using hand gesture libraries from Google MediaPipe to train Machine Learning models.

# 3. Specific Requirements

## 3.1 Functional Requirements

**FR.1**
The local environment must be able to detect a hand in the camera view.
Source: Dr. R. Iqbal
Priority:
Introduction: The local environment needs to be able to identify a hand from the other elements (noise) within the video feed input.
Inputs: Video camera feed.
Processing: Achieved using external libraries such as Google MediaPipe.
Outputs: Location of hand in the camera view.
Error Handling: This requirement is necessary for basic functionality of the system. If any implementation we use does not work correctly, we will need to adjust or redesign it until it works.

**FR.2**
The local environment must be able to track the hand and its movements based off of twenty-one specific points in the wrist, palm, and fingers.
Source: Dr. R. Iqbal
Priority:
Introduction: We must be able to track not only the hand, but its position, significnt features, rotations, and movements.
Inputs: Live video feed.
Processing: Achieved using external libraries such as Google MediaPipe.
Outputs: Hand tracking data.
Error Handling: This requirement is necessary for basic functionality of the system. If any implementation we use does not work correctly, we will need to adjust or redesign it until it works.

**FR.3**
The local environment must be able to identify meaningful hand movements apart from noise.
Source: Dr. R. Iqbal
Priority:
Introduction: To recognize hand gestures the local environment will distinguish meaningful hand movements from all other nosie in order to only send the necessary data to the cloud for further computation.
Inputs: Hand tracking data.
Processing: An algorithm that uses the output from frame-to-frame hand tracking data, and is able to calculate any significant movement from smaller, insignificant, or unintended movement.
Outputs: Meaningful hand tracking data.
Error Handling: This requirement is necessary for basic functionality of the system. If any implementation we use does not work correctly, we will need to adjust or redesign it until it works.

**FR.4**

The local environment must be able to send hand tracking data to the cloud environment.
Source: Development Team
Priority:
Introduction: The local environment will be able to send hand tracking data to the cloud environment for further processing.
Inputs: Hand tracking data.
Processing: Setting up the cloud environment and testing sending packages of data.
Outputs: Hand tracking data to the cloud environment.
Error Handling: This requirement is necessary for basic functionality of the system. If any implementation we use does not work correctly, we will need to adjust or redesign it until it works.

**FR.5**
The cloud envirnoment must be able to translate hand tracking data of gestures into letters and numbers.
Source: Dr. R. Iqbal
Priority:
Introduction: Our cloud environment will be using an existing dictionary of sign language letters and numbers to translate the captured sign language gestures.
Inputs: Hand tracking data of sign language gestures.
Processing: Dictionary of sign language to letters and numbers.
Outputs: Associated letter or number.
Error Handling: If the captured gestures cannot be translated into any alphabetic or numeric characters, then we will have to send the gesture output as a default error symbol of some kind so that the following systems know that the gesture was not properly interpreted.

**FR.6**
The cloud envirnoment must be able to translate hand tracking data of gestures into words.
Source: Development Team
Priority:
Introduction: Our cloud environment will be using an existing dictionary of sign language words to translate the caputed sign language gestures.
Inputs: Hand tracking data of sign language gestures.
Processing: Dictionary of sign language to words.
Outputs: Associated word.
Error Handling: If the captured gestures cannot be translated into any words, then we will send the gesture output as a default error symbol of some kind so that the following systems know that the gesture was not properly interpreted.

**FR.7**
The system will be able to understand a variety of different sign languages.
Source: Development Team
Priority:
Introduction: The gesture interpretation module will be trained to interpret other popular sign languages such as British Sign Language, Indo-Pakistani Sign Language, etc.
Inputs: Hand tracking data of sign language gestures.

Processing: Dictionary of sign language to words.
Outputs: Associated word.
Error Handling: If the captured gestures cannot be translated into any words, then we will send the gesture output as a default error symbol of some kind so that the following systems know that the gesture was not properly interpreted.

**FR.8**
The cloud envirnoment must translate letters and numbers into gramatically accurate phrases.
Source: Development Team
Priority:
Introduction: The cloud enviroment will be translating captured gestures using a formal grammar recognition module to help accurately form correct phrases.
Inputs: Letters and numbers.
Processing: Formal grammar recognition machine learning model.
Outputs: Sequences of accurate phrases, or words.
Error Handling: If the machine learning module is not able to accurately detect gramatically correct phrases, then we will either retrain it or come up with another method of detecting phrases.

**FR.9**
The cloud envirnoment must be able to identify abbreviatations.
Source: Development Team
Priority:
Introduction: The cloud environment will be able to translate certain abbreviations of commands instead of having to perform many gestures for a command.
Inputs: Letters and numbers.
Processing: Formal grammar recognition machine learning model.
Outputs: Sequences of accurate phrases, or words.
Error Handling: If the system is not able to detect a valid abbreviation, then the output will produce an error notification stating that there was no valid corresponding abbreviation.

**FR.10**
The cloud envirnoment must translate phrases of letters and numbers into commands.
Source: Development Team
Priority:
Introduction: The cloud environment will translate letters, numbers, and phrases into applicable commands for use in the local environment.
Inputs: Sequences of letters and numbers.
Processing: The gramatically correct phrase will be checked against a library of key command phrases.
Outputs: Commands for controlling various technologies.
Error Handling: If the system cannot detect phrases as valid commands, then the output will reflect as such with an error notification saying that there was to valid corresponding command.

**FR.11**

The local environment must perform commands.
Source: Dr. R. Iqbal
Priority:
Introduction: The local environment will perform whichever command it received and perform it on the corresponding technologies. Dr. Iqbal already has some technology in place that will allow us to do this if all other phases of the project are successful.
Inputs: Command from command interpreting on cloud server.
Processing: A smart home device or similar pre-existing system that can perform the specified command. An example command could be “Light 1 On”.
Outputs: Action performed on smart home device.
Error Handling: If a smart-home device is unable to use our commands, we may need to use a different one.

## 3.2 Non-Functional Requirements

**NFR.1**
The system must perform the detection and translation of gestures into commands in real-time.
Source: Dr. R. Iqbal
Priority:
Introduction: The system will be able to detect and translate the gestures to sign language and commands in real-time.
Inputs: Hand gestures from the local environment.
Processing: Image processing module and formal grammar module will be what we utilize to achieve operation in real time.
Outputs: A command.
Error Handling: If we cannot detect and translate gestures close to real-time, then we will have to re-evaluate our detection methods, or settle for having these actions done quickly but not in real-time.

**NFR.2**
The gesture interpretation module must recognize gestures with an accuracy of 90% or higher.
Source: Dr. R. Iqbal
Priority:
Introduction: The image processesing module will be able to perform the recognition of gestures at an accuracy rate of 90% or higher under conditions of adequate lighting of the hand used to perform the gestures.
Inputs: Hand gestures from the local environment.
Processing: Image processing module using machine learning tools from MediaPipe.
Outputs: The character correcsponding to the ASL gesture.
Error Handling: If we cannot detect and translate gestures above 90%, then we will have to re-evaluate our detection methods, or settle for having our gestures translated close to but not exactly above 90% of the time.

**NFR.3**
The formal grammar module must enhance the systems ability to accurately predict words and phrases.
Source: Dr. R. Iqbal

Priority:
Introduction: The formal grammar module will be able to effectively predict words and phrases from a sequence of letters and numbers.
Inputs: Sequence of letters and numbers.
Processing: An autocomplete/search functionality utilizing a library of predetermined, common smart home commands to base predictions off of.
Outputs: A word or phrase.
Error Handling: If this module hinders the ability of our system to associate hand gestures to phrases it will be further developed and constricted to only predict phrases associated with a dictionary of available commands.

**NFR.4**
The hand tracking module will continuously track hands that are in the view space of the camera.
Source: Development Team
Priority:
Introduction: The hand tracking module must continuously track a hand in the view space while the system is deployed and active.
Inputs: A video stream.
Processing: A machine learning module that looks for hands within video streams.
Outputs: A data object containing information on the position and orientation of various points on the hand.
Error Handling: If the video stream is interrupted, our system will throw an error to the rest of the system notifying it that the video stream has been interrupted, and stopping other operations until the stream resumes or the system is reset.

**NFR.5**
The command output module will be able to tailor command outputs to compatible devices.
Source: Development Team
Priority:
Introduction: The command output module will assist our products portability in being used on a variety of devices by matching command outputs to the device API.
Inputs: Command outputs.
Processing: The command output module.
Outputs: Device API outputs.
Error Handling: If this module does not work we will work around this by removing this feature and only make our system compatable with only one basic device like a common smart home.

**NFR.6**
The hand tracking module will only be able to track one hand in the video stream at once.
Source: Development Team
Priority:
Introduction: We will only support the ability to track a single hand in the view space of the video stream at once.
Inputs: A video stream.
Processing: A machine learning module that looks for hands within video streams.

Outputs: A data object containing information on the position and orientation of various points on a single hand.
Error Handling: If other hands appear on the screen, our hand tracking module will track the first one it is able to connect.

## 3.3 Design Constraints

**DC.1**
The local environment must have a functioning video camera.
Source: Dr. R. Iqbal
Priority:
Introduction: The local environment must have a functioning camera, either built in or external in order to function properly.
Inputs: A functioning video camera.
Processing: N/A
Outputs: A video feed.
Error Handling: N/A

**DC.2**
The camera must have at least a 480p resolution quality.
Source: Development Team
Priority:
Introduction: For the system to function properly, the camera must be of 480p resolution in order to properly capture gestures.
Inputs: A functioning video camera.
Processing: N/A
Outputs: A 480p video feed.
Error Handling: If 480p is too low of a minimum standard for resolution for our purposes, then we can upgrade to a slightly higher resolution video feed if necessary.

**DC.3**
The local environment must have a stable broadband internet connection.
Source: Development Team
Priority:
Introduction: For the system to send and receive captures and commands, it will need a reliable connection to the internet.
Inputs: N/A
Processing: N/A
Outputs: N/A
Error Handling: If we are unable to have a reliable internet connection for some reason, we may have to run all of our components on a local machine.

**DC.4**
The system will interpret sign language using ASL.
Source: Dr. R. Iqbal
Priority:
Introduction: The image processing module will be trained to recognize ASL.

Inputs: ASL gestures.
Processing: N/A
Outputs: English character associated with ASL gesture.
Error Handling: N/A

## 3.4 External Interface Requirements

### 3.4.1 User Interfaces

**UIR.1**
The hand tracking module must have an output of the hand detection video stream for debugging.
Source: Development Team
Priority:
Introduction: It would be useful to have an output window of the video stream with an overlay based on our hand tracking data for debugging purposes as a developer in order to ensure that the detection is working as needed.
Inputs: Video data.
Processing: A communication between the video stream, hand and finger position detection, and a video output system. Google MediaPipe Visualizer can help us perform this process.
Outputs: Video data with hand and finger position detection data overlayed.
Error Handling: If the video feed freezes or cuts out, then the debugging system will have to stop showing the feed accordingly. If the hand tracking module also

**UIR.2**
The gesture interpretation module must have an output of its processes and detected gestures in a console for debugging.
Source: Development Team
Priority:
Introduction: It would be useful to have a console output of the processing actions and successful or unsuccessful gesture detections that this module is performing as it is performing them so that we as developers can ensure it is working as intended.
Inputs: Hand tracking data.
Processing: Determining whether the hand tracking data contains a recognizable gesture and recording those processes as they occur.
Outputs: A console window of text describing the processing that occurs and the recognized gesture output if the system is able to successfully detect one, including any errors.
Error Handling: If the console debugging does not describe a certain process we are wanting to see as it is happening, we will have to add more code that details those processes.

**UIR.3**
The formal grammar module must have an output of its processes and valid commands that have been found in a console for debugging.
Source: Development Team
Priority:

Introduction: It would be useful to have a console output of the processing actions and command phrase interpretations that this module is performing as it is performing them so that we as developers can ensure it is working as intended.
Inputs: A sequence of alphabetic and numeric characters.
Processing: Determining whether the sequence of characters contains a valid command phrase and recording those processes as they occur.
Outputs: A console window of text describing the processing that occurs and the recognized command phrases output if the system is able to successfully detect one, including any errors.
Error Handling: If the console debugging does not describe a certain process we are wanting to see as it is happening, we will have to add more code that details those processes.

### 3.4.2 Hardware Interfaces

**HIR.1**
The system must have a camera that can record video data.
Source: Dr. R. Iqbal
Priority:
Introduction: We need to have a camera connected to the local environment that can record video data for the hand tracking module to process. This can be connected to the PC on the local environment via Universal Serial Bus (USB).
Inputs: An environment to record.
Processing: Connecting to the camera from the local environment code in order to transmit the video data to the hand tracking module.
Outputs: Video data of the environment.
Error Handling: If the camera malfunctions and occasionally skips frames, or does not record video data for a short amount of time, we should make sure that the rest of the modules in the system are aware that the local environment has stopped receiving data.

**HIR.2**
The system must have a PC connected to the internet.
Source: Development Team
Priority:
Introduction: The local environment will run on a PC that is connected to a camera, and which also has an internet connection to be able to connect to cloud environment. This PC will also be able to connect to a smart home device. The PC will host the hand tracking module and the command output module.
Inputs: Video stream from the camera.
Processing: Extracting hand tracking data from the video stream and sending it to the cloud environment, and receiving back valid command phrases.
Outputs: Valid commands to a smart home device.
Error Handling: If the local PC stops working for any reason suddenly, then we must ensure that our system is able to handle a system crash. If the internet connection for the local PC is interrupted or does not work properly, then we will have to make sure the system can properly communicate that it is having connection issues.

**HIR.3**

The system must have a cloud environment to host the gesture interpretation module and the formal grammar module.
Source: Development Team
Priority:
Introduction: We want the larger and more computationally complex parts of the system to be hosted on an external cloud environment that will be connected to by the local environment via the internet. The cloud environment will receive hand tracking data from the local environment and send back valid command phrases through the gesture interpretation module and the formal grammar module if possible.
Inputs: Hand tracking data from the local environment.
Processing: The hand tracking data will be put through the gesture interpretation module and the formal grammar module.
Outputs: Valid command phrases.
Error Handling: If the cloud environment is unable to communicate with the local environment to receive hand tracking data for a long period of time, we will have to include a timeout function in the cloud environment modules in order to ensure the cloud environment knows if it has lost connection.

### 3.4.3 Software Interfaces

**SIR.1**
The hand tracking module must have an interface that sends hand tracking data to the gesture interpretation system.
Source: Development Team
Priority:
Introduction: The hand tracking module must be able to communicate hand tracking data to the gesture interpretation module on the cloud environment.
Inputs: Hand tracking data as it is processed by the hand tracking module.
Processing: Preparing the hand tracking data and combining it with any other necessary data to send to the cloud environment modules.
Outputs: A data object containing hand tracking data to be received by an interface on the gesture interpretation module.
Error Handling: If the hand tracking module cannot send the data object to the cloud environment to the gesture interpretation module, the system must be able to safely throw an error so that the entire program can be notified of this failure to commincate.

**SIR.2**
The gesture interpretation module must have an interface that receives hand tracking data from the hand tracking module.
Source: Development Team
Priority:
Introduction: The gesture interpretation will interface with the hand tracking module via a cloud internet communication to receive hand tracking data, which it will then use to interpret as individual characters.
Inputs: A data object containing hand tracking data.

Processing: Receiving and unpacking the hand tracking data, and any other communication data sent that is important to the gesture interpretation process.
Outputs: Hand tracking data to be used by the gesture interpretation module.
Error Handling: If the data object cannot be successfully, fully received by the gesture interpretation module, then the gesture interpretation module must be able to safely throw an error so that the entire program can be notified of this failure to commincate.

**SIR.3**
The formal grammar module must have an interface that receives a sequence of characters from the gesture interpretation module.
Source: Development Team
Priority:
Introduction: The formal grammar module will interface with the gesture interpretation module to receive sequences of characters which will then be interpreted as command phrases if possible.
Inputs: A sequence of characters.
Processing: N/A
Outputs: N/A
Error Handling: If the gesture interpretation system has an error when processing the hand tracking data, it must notify the formal grammar module in order for that module to stop processing the data.

**SIR.4**
The formal grammar module must have an interface that sends a command phrase to the command output module.
Source: Development Team
Priority:
Introduction: The formal grammar module will interface with command output module in order to send command phrases that in interprets to it.
Inputs: An interpreted command phrase.
Processing: N/A
Outputs: N/A
Error Handling: If the gesture interpretation system has an error when processing the hand tracking data, it must notify the formal grammar module in order for that module to stop processing the data.

**SIR.5**
The command output module must be able to receive command phrases from the formal grammar module.
Source: Development Team
Priority:
Introduction: The command output module will receive command phrases over the internet from the formal grammar module, which it will then send to a smart home device via the API that the chosen smart home uses.
Inputs: A data object containing a command phrase to be used.

Processing: Retrieving and unpacking the command phrase from the data object.
Outputs: A command phrase to be used in a smart home system.
Error Handling: If the command output system is unable to successfully receive the data object from the formal grammar module, then it must be able to safely throw an error so that the entire program can be notified of this failure to commincate.

### 3.4.4 Communications Interfaces

**CIR.1**
The local environment must be able to communicate via the internet to the cloud environment.
Source: Development Team
Priority:
Introduction: The local environment must be able to send the hand tracking data to the cloud environment and receive interpreted commands from the cloud environment over the internet.
Inputs: Data from the local environment.
Processing: An API or similar that transmits over a network communication system.
Outputs: Data from the cloud environment.
Error Handling: If we are unable to use a cloud environment or properly send data from a cloud environment that hosts our processing module, then we may have to run the entire system locally on a single machine, or multiple.

**CIR.2**
The cloud environment must be able to communicate via the internet to the local environment.
Source: Development Team
Priority:
Introduction: The cloud environment must be able to send the interpreted commands to the local environment after it receives hand tracking data from the local environment over the internet.
Inputs: Data from the cloud environment.
Processing: An API or similar that transmits over a network communication system.
Outputs: Data from the local environment.
Error Handling: If we are unable to use a cloud environment or properly send data from a cloud environment that hosts our processing module, then we may have to run the entire system locally on a single machine, or multiple.

## 3.5 Logical Database Requirements

*Not Applicable*

## 3.6 Other Requirements

*Not Applicable*

# 4. Change Management Process

Using the project management framework Agile Scrum, change is common. We must prepare for changes by allowing them to be implemented as part of the process.

Stakeholders and team members can submit a change request using our Change Request Form (see below). The product owner will see this submission and decide if it is urgent and its

impact on the project, or whether it can wait until the next sprint planning meeting. If this change is simply another feature and has minimal impact on the project, the product owner can decide to accept the change immediately and add a user story to the product backlog for visibility of the scrum team at the next sprint planning meeting. If the change will have significant impact on the project, he will give the request to the scrum master.

The scrum master will receive the request and look at how it will affect the current sprint. In the case that the change would result in the current sprint to become usesless, he will hold a scrum meeting to immediately discuss the change and possibly terminate or modify the current sprint. If it will not affect the current sprint, he will wait until the next sprint planning meeting to discuss the change.

During discusson of the change, the scrum team will go over how this change will affect the project and its priority. The team must consider previously completed story points that will now become obsolete with this change. The team also must discuss the time this change adds or removes from the project, as with Agile Scrum we attempt to keep within our time box. They must look at how this change will affect other requirements as well. With all this information ready to be presented, the team goes to discuss with the product owner. The team will tell the product owner if they believe the change should be accepted or not and present their reasons, but the product owner has the final say on whether to accept the change.

If the change is accepted, it is the scrum team or the product owner's responsibility to add the change as a user story to the product backlog, and update any user stories that are affected by this change. At every sprint planning meeting, Trello will be updated based on the product backlog. Once Trello is aligned with the product backlog, the sprint planning can continue as usual. Once a change is final, the team will update the related technical documents for the entirety of the project, including the SRS document.

However, if a change is rejected it is the teams responsibility, including the product owner, to provide an explanation for the rejection to the stakeholders. The team must explain why the change is either too far out of scope of the product or that the impact of implementing this change is too great of a risk. The rejection of the change will be documented by the team to help explain why other changes may also be rejected.

With the above process, changes can be discussed and implemented into the Agile Scrum process every 2 weeks at the sprint planning meeting. If a change requires immediate action, the current sprint may have to be terminated because once a sprint starts its requirements cannot change. A setup like this expects changes and allows them to become part of the process.

# Change Request

Date:

Submitted By:

Requirements:

Priority:

Description:

Benefits:

Impact on Project (How many currently completed story points will become worthless? Will this terminate the current sprint?):

# Appendices

Google MediaPipe:
Proves concept and gives direction on real-time gesture detection. Open source.
https://ai.googleblog.com/2019/08/on-device-real-time-hand-tracking-with.html
Uses a detection model called BlazePalm. Detects palm, marks points, then performs gesture recognition. This improves performance and allows quick detection in real-time.