

IT Project Management

(MCA303)

**Sign Language Alphabet Gesture Recognition**

**SUBMITTED BY: SUBMITTED TO:**

Bipan Tiwari (202201011) Dr. Harkiran Kaur

Devaksha Bhatt (202201060)

Harmanjot Singh (202201013)

Kabir Juneja (202201017)

**WBS Dictionary Entry**

**Date 01-09-2023**

**Project Title:** Sign Language Alphabet Detection

**WBS Item Number:** 1.0

**WBS Item Name:** Planning

**Description:** In this project, we are going toplan and estimate the time, cost, and effort for the project. Through this information we aim to create a robust, real-time system.

**WBS Dictionary Entry**

**Date 01-09-2023**

**Project Title:** Sign language Alphabet Detector

**WBS Item Number:** 2.0

**WBS Item Name:** Requirement gathering

**Description:** In Requirement Gathering, we are going to Analysis the requirements and we will priorities some of the requirements and make SRS(software requirement specification ) Document. This phase helps in clear and comprehensive understanding of functional and non-functional requirement.

**WBS Dictionary Entry**

**Date 01-09-2023**

**Project Title:** Sign Language Alphabet Detection

**WBS Item Number:** 3.0

**WBS Item Name:** Design

**Description:** In the Design Phase, DFDs, UML diagrams are designed which will help in the code implementation and also helps in understanding the logical structure of the project.

**WBS Dictionary Entry**

**Date 01-09-2023**

**Project Title:** Sign Language Alphabet Detection

**WBS Item Number:** 4.0

**WBS Item Name:** Coding

**Description:** In the Implementation phase, we build the project step by step. Each part is tested individually (unit Testing) to make sure it works correctly. This step can help in catching and fixing any error.

**WBS Dictionary Entry**

**Date 01-09-2023**

**Project Title:** Sign Language Alphabet detection

**WBS Item Number:** 5.0

**WBS Item Name:** Testing

**Description:** After the coding of the project testing will be conduct, Unit testing and Integration testing will be conduct then test cases will be prepared and then alpha testing and beta testing will be done.

**CPM**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Activity | Name | Days | Early-Start | Early-Finish | Late-Start | Late-Finish | Slack |
| 1 | Planning | 8 | 0 | 8 | 0 | 8 | 0 |
| 2 | Requirements Gathering | 9 | 8 | 17 | 8 | 17 | 0 |
| 3 | Architectural Design | 2 | 17 | 19 | 18 | 20 | 2 |
| 4 | Interface Design | 3 | 17 | 20 | 17 | 20 | 0 |
| 5 | DFDs | 2 | 17 | 19 | 18 | 20 | 2 |
| 6 | Coding | 8 | 20 | 28 | 20 | 28 | 0 |
| 7 | Unit Testing | 2 | 28 | 30 | 30 | 32 | 2 |
| 8 | Coding Reviews | 4 | 28 | 32 | 28 | 32 | 0 |
| 9 | Integration Testing | 3 | 30 | 33 | 32 | 35 | 3 |
| 10 | System Testing | 3 | 32 | 35 | 32 | 35 | 0 |
| 11 | Deployment Plan | 2 | 35 | 37 | 35 | 37 | 0 |
| 12 | Deploy Application | 2 | 37 | 39 | 37 | 39 | 0 |
| 13 | Configure System | 1 | 39 | 40 | 39 | 40 | 0 |

Critical Path:

1->2->4->6->8->10->11->12->1

1

2

5

3

4

6

8

7

9

10

11

12

13

8

9

2 3 2

8

2 4

3

3

2

2

1

PERT

Topt = 40

Tlikely = 50

Tpess = 60

**E**  = Topt +( 4 x Tlikely )+ Tpess / 6

40 + (4 x 50) + 60 / 6

**E**  = 50

**Variance** = (Tpess – Topt)2 / 62

= 400 / 36

= 11.11

**Functional Point**

|  |  |
| --- | --- |
| External Input | 3 |
| External Output | 2 |
| External Queries | 2 |
| Internal Logical Files | 2 |
| External Interfaces | 4 |

**UFP = 3x4+2x5+2x4+2x10+4x7**

**UFP = 12+10+8+20+28**

**UFP = 78**

**CAF = 0.65+0.01+(14x3)**

**CAF = 1.07**

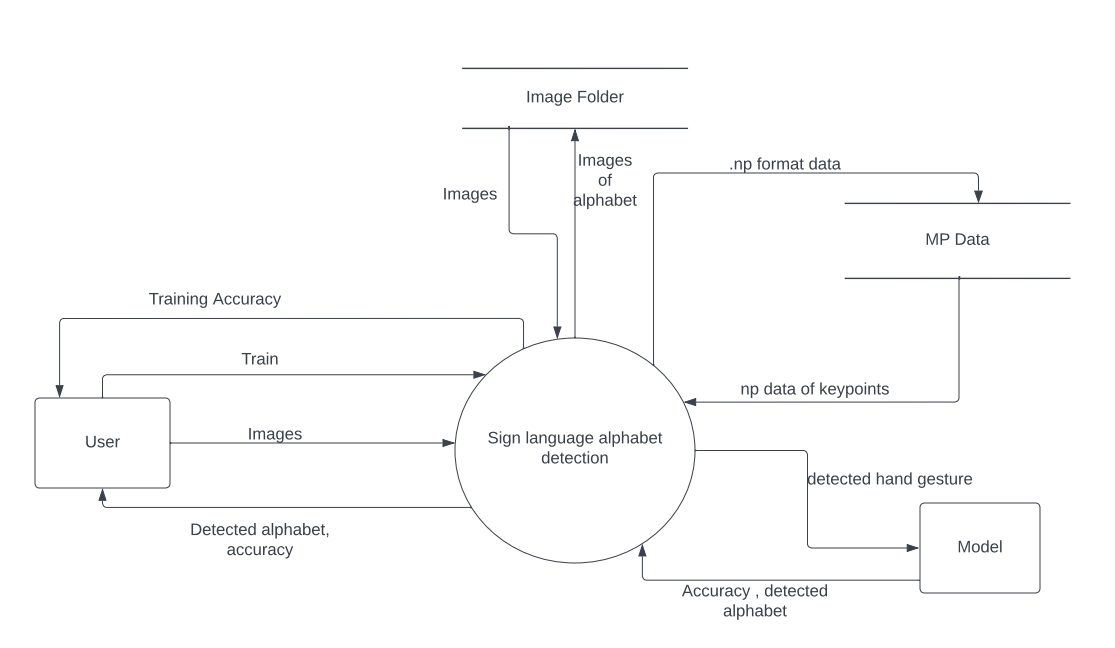
**FP = CAFxUFP**

**FP = 83.46**

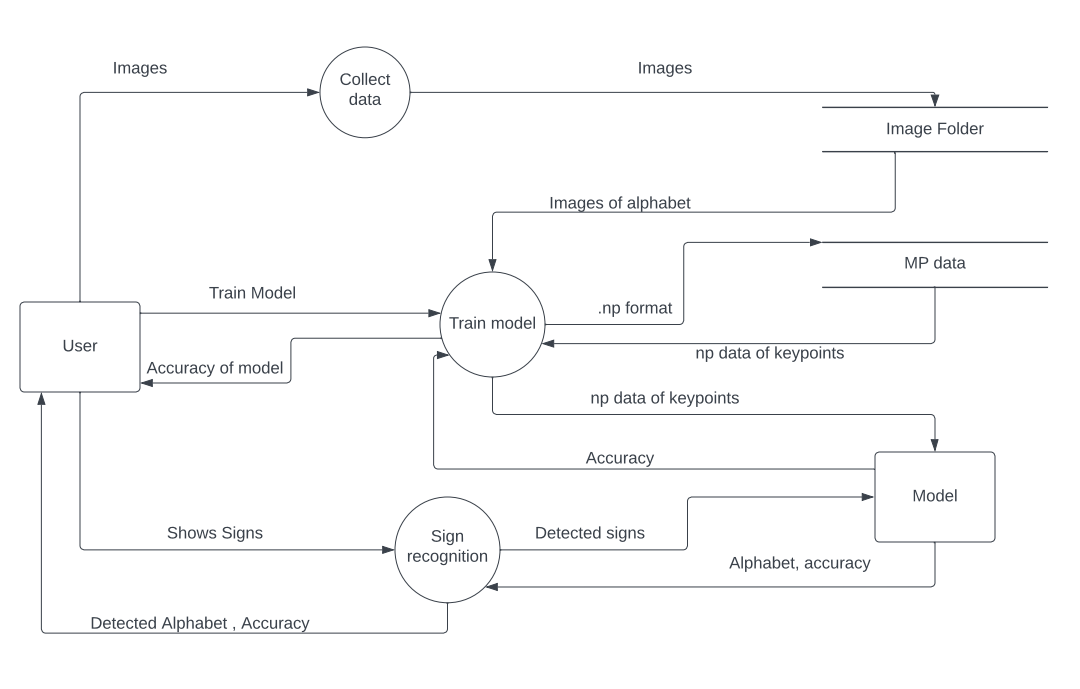
**(\***Assuming the weighting factor to be average.)

**Data Flow Diagrams**

Level 0 – Context Free Diagram



Level 1 –



***Sign Language Alphabet Gesture Recognition***

**Software Requirements Specification Document**

**Version 1.0**

**TABLE OF CONTENTS**

**Chapter No. Topic Page No.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1. |  |  | Introduction | 3 |
|  | 1.1 |  | Purpose of this Document | 3 |
|  | 1.2 |  | Scope of the Development Project | 3 |
|  | 1.3 |  | Definitions, abbreviations and acronyms | 5 |
|  | 1.4 |  | References | 7 |
|  | 1.5 |  | Overview | 7 |
| 2. |  |  | Overall Description | 8 |
|  | 2.1 |  | Product Perspective | 8 |
|  | 2.2 |  | Product functions | 11 |
|  | 2.3 |  | User Characteristics | 12 |
|  | 2.4 |  | General Constraints, Assumptions and Dependencies | 13 |
|  | 2.5 |  | Apportioning of the requirements | 14 |
| 3. |  |  | Specific Requirements | 15 |
|  | 3.1 |  | External Interface Requirements | 15 |
|  | 3.2 |  | Detailed Description of Functional Requirements | 15 |
|  |  | 3.2.1 | Functional Requirements for Student Welcome Screen | 16 |
|  |  | 3.2.2 | Functional Requirements for Staff Welcome Screen | 16 |
|  | 3.3 | 3.2.3 | Functional Requirements for Student cum Staff Welcome Screen  Performance requirements | 17  17 |
|  | 3.4 |  | Logical database requirements | 18 |
|  | 3.5 |  | Quality attributes | 19 |
|  | 3.6 |  | Other requirements | 19 |

1. Change History 19
2. Document Approvers 20
3. **Introduction**
   1. **Purpose of this Document**

The purpose of this SRS document is to provide a detailed overview of our software product, its parameters and goals. This document describes the project's target audience and its user interface, hardware and software requirements. It defines how our client, team and audience see the product and its functionality.

* 1. **Scope of the Development Project**

The goal is to design software for that enables real time recognition and presentation of ASL alphabet signs in order to eliminate communication gaps between the deaf and hearing communities. This project is designed to recognize and classify ASL alphabet letters based on hand gestures captured through a webcam in real-time. It integrates seamlessly with a user's web camera, allowing them to perform ASL alphabet signs for recognition. The software employs advanced computer vision algorithms to analyze these hand gestures, delivering a real-time output that displays the recognized ASL alphabet letter on the screen. Additionally, the software provides a user-friendly interface for capturing webcam frames and offering user feedback or instructions, ensuring an accessible and inclusive communication tool for both signers and non-signers.

The software must be able to perform the following operations:

1. **Alphabet Mapping:** The system's underlying model is to be tasked with the intricate challenge of recognizing and categorizing characters from the sign language alphabet based on hand gestures, using advanced machine learning and pattern recognition techniques.
2. **Web Camera Integration:** The software must establish a seamless connection with the user's webcam, enabling users to perform ASL alphabet signs in front of the camera for real-time recognition. This requires webcam access, video stream processing, and real-time frame capture.
3. **Real-time Gesture Analysis:** The software is to incorporate cutting-edge computer vision algorithms, which process hand gestures captured by the webcam in real-time. This involves frame-by-frame analysis, feature extraction, and pattern matching to provide swift recognition.
4. **Visual Output:** The software is responsible for visually rendering the recognized ASL alphabet character on the screen, ensuring it's visible to both the sign language communicator and the observer. This involves graphical rendering and user interface interaction for seamless character presentation.
   1. **Definitions, abbreviations and acronyms**

**Definitions**

Table 1 gives explanation of the most commonly used terms in this SRS document.

**Table 1: Definitions for most commonly used terms**

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Term** | **Definition** |
| 1 | ASL | American Sign Language (ASL) is a natural language that serves as the predominant sign language of Deaf communities in the United States of America and most of Anglophone Canada.. [1] |
| 2 | LSTM | LSTM (Long Short-Term Memory) is a recurrent neural network (RNN) architecture widely used in Deep Learning. It excels at capturing long-term dependencies, making it ideal for sequence prediction tasks. |
| 3 | MP | MediaPipe is a Framework for building machine learning pipelines for processing time-series data like video, audio, etc. This cross-platform Framework works on Desktop/Server, Android, iOS, and embedded devices like Raspberry Pi and Jetson Nano. |
| 4 | Tensorflow | TensorFlow is a free and open-source software library for machine learning and artificial intelligence. It can be used across a range of tasks but has a particular focus on training and inference of deep neural networks. |
| 5 | OpenCV | OpenCV is a great tool for image processing and performing computer vision tasks. It is an open-source library that can be used to perform tasks like face detection, objection tracking, landmark detection, and much more. |

**Abbreviations**

Table 2 gives the full form of most commonly used mnemonics in this SRS document.

**Table 2: Full form for most commonly used mnemonics**

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Mnemonic** | **Full Form** |
| 1 | ASL | American Sign Language |
| 2 | LSTM | Long-Short Term Memory |
| 3 | MP | Media Pipe |

* 1. **References**

|  |  |
| --- | --- |
| [1] | "Wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/American\_Sign\_Language. |
| [2] | J. Brownlee, "Machine Learning Mastery," 27 7 2022. [Online]. Available: https://machinelearningmastery.com/introduction-python-deep-learning-library-tensorflow/. |
| [3] | A. Gupta, "Analytics Vidhya," 22 08 2022. [Online]. Available: https://www.analyticsvidhya.com/blog/2021/09/some-amazing-applications-of-opencv-library/. |
| [4] | S. Saxena, "Analytics Vidhya," 25 7 2023. [Online]. Available: https://www.analyticsvidhya.com/blog/2021/03/introduction-to-long-short-term-memory-lstm/#:~:text=LSTM%20. |
| [5] | Kulki, "LearnOpenCV," 1 3 2022. [Online]. Available: https://learnopencv.com/introduction-to-mediapipe/. |

* 1. **Overview**

The remaining sections of this document provide a general description, including characteristics of the users of this project, the product's hardware, and the functional and data requirements of the product. General description of the project is discussed in section 2 of this document. Section 2 gives the functional requirements, data requirements and constraints and assumptions made while designing the multi-utility system. It also gives the user viewpoint of product use. Section 3 gives the specific requirements of the product. Section 3.0 also discusses the external interface requirements and gives detailed description of functional requirements.

1. **Overall Description**
   1. **Product Functions**

The product should be able to perform the following operations:

1. The software should accurately recognize and classify the letters of the ASL alphabet based on hand gestures input.
2. The embedded software should perform gesture recognition with minimal latency, ensuring timely feedback and display of results.
3. The software interfaces with the user's web camera, allowing them to perform ASL alphabet signs in front of the camera for recognition.
4. The recognized ASL alphabet letter should be displayed on the screen for both the signer and observer to see.
   1. **User Characteristics**

The goal is to design software for a Sign Language Alphabet Gesture Recognition for different users. These user types are listed below as follows:

* + 1. Deaf Individuals
    2. Non-Signers
    3. Staff Members
    4. Students
    5. Educators

As one can see from the list, each user will have different educational background and expertise level in using the system. Our goal is to develop software that should be easy to use for all types of users, including the non-signers and signers. Thus while designing the software one can assume that each user type has the following characteristics:

* The user is computer-literate and has little or no difficulty in using this application.
* In order to use the software it is not required that a user be aware of its internal working but he/she is expected to know what happens when they make a hand gesture.
  1. **General Constraints, Assumptions and Dependencies**

The following list presents the constraints, assumptions, dependencies or guidelines that are imposed upon implementation of the Sign Language Alphabet Gesture Recognition:

* The software's effectiveness is dependent on the availability of web cameras and compatible hardware on the user's device. Limited hardware capabilities may constrain the software's performance.
* The system's recognition accuracy may vary under different lighting conditions, hand sizes, skin tones, and background environments, potentially limiting its effectiveness.
* Adapting the software to support sign language alphabets in different regions and languages may be constrained by resource availability and expertise.
* It is assumed that users have access to a functional web camera on their devices to capture and transmit hand gestures for recognition.
* Users are expected to operate in environments with adequate lighting to facilitate accurate gesture recognition.
* The software assumes that data captured by the webcam is processed locally on the user's device, minimizing concerns about data privacy.
* The successful implementation of real-time recognition depends on the integration of advanced computer vision algorithms that can accurately analyze and interpret hand gestures.
* To embed the software within other applications, developers rely on well-documented APIs and integration mechanisms to access the ASL alphabet recognition functionality.
  1. **Apportioning of requirements**

The Sign Language Alphabet Gesture Recognition is to be implemented in the following three phases:

1. **Phase 1: Proof of Concept and Basic Recognition:** Establishing the groundwork, this phase develops a rudimentary model for recognizing a limited set of ASL alphabet signs. Basic webcam integration captures hand gestures for real-time analysis, with a simple user interface displaying recognized letters. The primary goal is to demonstrate feasibility and lay the foundation for subsequent enhancements in later phases.
2. **Phase 2: Prototype Enhancement and User Interface Refinement:** Building upon the proof of concept, this phase expands the model to cover the entire ASL alphabet. Webcam integration is enhanced with additional features, and computer vision algorithms are improved for accurate and continuous hand gesture analysis. The user interface undergoes refinement for a more polished and engaging user experience, including visual feedback for effective communication. The goal is to create a robust prototype with increased accuracy and improved usability.
3. **Phase 3: Optimization, Scaling, and Deployment:** In this phase, the software undergoes optimization for improved accuracy and efficiency. Webcam integration is fine-tuned for compatibility and performance across various systems. Advanced optimization techniques are implemented for real-time processing, and the user interface is further optimized for different screen sizes. The software is deployed for widespread use, with mechanisms in place to handle larger user bases and scalable server infrastructure. Continuous accuracy monitoring is established, ensuring a robust and deployable Sign Language Alphabet Recognizer Software.
4. **Specific Requirements**
   1. **External Interface Requirements**

The following list presents the external interface requirements:

* The product requires very limited graphics usage with webcam for taking the user input and a screen for displaying output.
* The product does not require usage of sound or animation.
  1. **Detailed Description of Functional Requirements**
     1. **Functional Requirements for Training a Model**

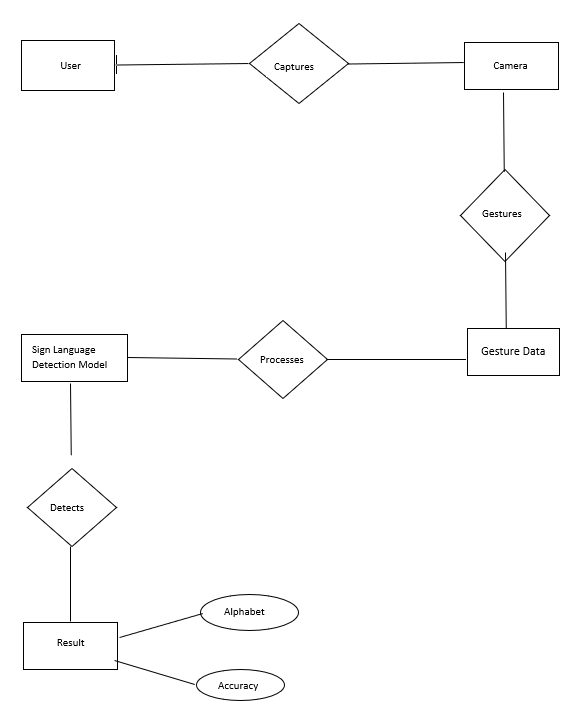
|  |  |
| --- | --- |
| **Purpose** | To train our model for the ASL Detection |
| **Inputs** | Hand gestures captured by user’s Web-camera |
| **Processing** | Accurate mapping of ASL alphabets through with the help of LSTM and MediaPipe to train the model. |
| **Outputs** | Trained model with accuracy. |

* + 1. **Functional Requirements for Alphabet Detection**

|  |  |
| --- | --- |
| **Purpose** | To recognize and display the ASL alphabet |
| **Inputs** | Hand gestures captured by user’s Web-camera |
| **Processing** | To use the trained model for predicting/recognizing the ASL alphabet. |
| **Outputs** | Detected alphabet letter is displayed on screen with the accuracy. |

* 1. **Performance Requirements**
* Implement robust security measures for data transmitted and processed, ensuring the privacy and integrity of user information.
* The software should provide real-time recognition of ASL alphabet signs, with a response time of 1 to 2 seconds for optimal communication.
* Optimize database queries and indexing for efficient data retrieval, contributing to overall system responsiveness.
  1. **Logical Database Requirements**

**Figure 3** shows the E-R diagram for the entire system.



**Figure 3: E-R Diagram for the Sign Language Alphabet Gesture Recognition**

* 1. **Quality Attributes**

1. Accuracy: The software must accurately recognize and classify ASL alphabet letters based on hand gestures. Achieving a high level of accuracy to ensure reliable communication between signers and non-signers.
2. Robustness: The system should be robust enough to handle variations in lighting conditions, hand sizes, skin tones, and backgrounds. Robustness ensures consistent performance across diverse scenarios, making the tool accessible in various environments.
3. Security and Privacy: User data captured by the webcam should be processed locally and not stored. Ensuring security and privacy safeguards user information, building trust in the system. Adherence to privacy regulations and ethical considerations is crucial.
4. Performance: The system's response time for recognizing a single sign gesture should be minimized. Optimizing performance enhances user experience, allowing for seamless and natural communication between signers and non-signers.
   1. **Other Requirements**

None at this time

1. **Change History**



Version 1.0 – Initial Release

200209

1. **Document Approvers**

SRS for Sign Language Alphabet Gesture Recognition approved by:

**Dr. Harkin Kaur**

**Associate Professor**

**Date: 13/10/2023**