2024

Group ViitTomakieli

Kandel Sushila, Lipponen Seppo, Viitala Jussi & Varpula Teemu

[kandel.sushila242@gmail.com](mailto:kandel.sushila242@gmail.com) https://github.com/sush01

[seppo.lipponen@outlook.com](mailto:seppo.lipponen@outlook.com) https://github.com/SeppoLipponen

[juzu.viita@protonmail.com](mailto:juzu.viita@protonmail.com) https://github.com/nullstare

[teemu.varpula@gmail.com](mailto:teemu.varpula@gmail.com) https://github.com/TeemuJVarpula

Table of Contents

[Document Change Log 2](#__RefHeading___Toc719_1892727294)

[Project Version and Deadline 2](#__RefHeading___Toc721_1892727294)

[Introduction 2](#__RefHeading___Toc731_1892727294)

[Project overview 2](#__RefHeading___Toc723_1892727294)

[Hardware Requirements: 2](#__RefHeading___Toc737_1892727294)

[Software Requirements: 2](#__RefHeading___Toc725_1892727294)

[Functional requirements 3](#__RefHeading___Toc739_1892727294)

[General description 3](#__RefHeading___Toc741_1892727294)

[Components and functions 3](#__RefHeading___Toc743_1892727294)

[Image Capturing 3](#__RefHeading___Toc755_1892727294)

[Image recognition 3](#__RefHeading___Toc757_1892727294)

[Show state of recognition of image 3](#__RefHeading___Toc759_1892727294)

[Show recognized characters to receiver 3](#__RefHeading___Toc761_1892727294)

[Non-functional requirements 3](#__RefHeading___Toc745_1892727294)

[Training of the model 3](#__RefHeading___Toc747_1892727294)

[Raspberry Pi cooling 3](#__RefHeading___Toc749_1892727294)

[Casing 3](#__RefHeading___Toc751_1892727294)

[Limitations 4](#__RefHeading___Toc727_1892727294)

[Implementation 4](#__RefHeading___Toc729_1892727294)

[Changes to the original requirements 4](#__RefHeading___Toc770_1892727294)

[Project setup 4](#__RefHeading___Toc776_1892727294)

[Collect images 5](#__RefHeading___Toc778_1892727294)

[Work Division 6](#__RefHeading___Toc780_1892727294)

# Document Change Log

|  |  |  |
| --- | --- | --- |
| **Version of the document** | **Version description / Changes made** | **Date** |
| 1.0 | First version of document | 14.2.2024 |
| 1.3 | Table of contents added | 23.2.2024 |
| 1.7 | Flowchart diagram of the program added | 29.2.2024 |
|  |  |  |

# Project Version and Deadline

|  |  |  |  |
| --- | --- | --- | --- |
| Project Version | Deadline for System Completion | Relevant Dates | Scope and Deliverables |
| 1.0 | 1.3.2024 | 26.2.2024 | Raspberry Pi setup |
| 1.1 | 6.3.2024 | 29.2.2024 | Raspberry Pi image recognition |
| 1.2 | 8.3.2024 | 8.3.2024 | Raspberry Pi working display |
| 1.3 | 8.3.2024 | 8.3.2024 | Three character identification |
| 1.4 | 15.3.2024 | 15.3.2024 | Whole alphabet identification and control |
| 1.5 | 27.3.2024 | 27.3.2024 | Document including implementation |

# Introduction

The aim of this project is to develop a system that can detect signs of Finnish sign language alphabet using OpenCV and MediaPipe running on Raspberry Pi. The system will use our own image collection to train the recognition model. Program will recognise the Finnish sign language alphabet excluding characters from Z-Ö that would require motion. In addition a few control signs are required to trigger events like write character, space, backspace and clear buffer.

# Project overview

Recognition is made from live video feed from the Raspberry Pi Camera Module or from regular webcam if run on PC or laptop. Recognition level will be indicated to the signer and if sign is recognized, it will be show in text screen for the receiver and signer.

## Hardware Requirements:

* Raspberry Pi
* Raspberry Pi Camera Module
* 1602 LCD display (Optionally can use second one for the signer)

## Software Requirements:

* Python
* OpenCV
* MediaPipe

# Functional requirements

## General description

System needs to recognize atleast one Finnish sign alphabet character and display it on the screen.

## Components and functions

### Image Capturing

User key press to capture image or video from camera attached to Raspberry. Live recognition is preferred if possible.

### Image recognition

* Project uses machine learning to train the recognition model and computer vision to identify signs from images. The main steps how the recognition works.
* Training will be done with MediaPipe
* Sign detection from live video

### Show state of recognition of image

Recognition level is shown to signer with leds and if display is available, they could also follow the recognition program running and indicating level of recognition.

### Show recognized characters to receiver

Recognized characters are shown to receiver from 1602 LCD display. New characters are appended to the end when recognized.

# Non-functional requirements

## Training of the model

Training the AI model can take quite a long time and should be done on more powerfull pc than Raspberry Pi.

## Raspberry Pi cooling

Considering image prosessing is quite a demanding task, processor might get hot without proper heatsink. Temperature should be monitored during use and specially during live video recognition.

## Casing

Possibility of 3D printed case is considered.

# Limitations

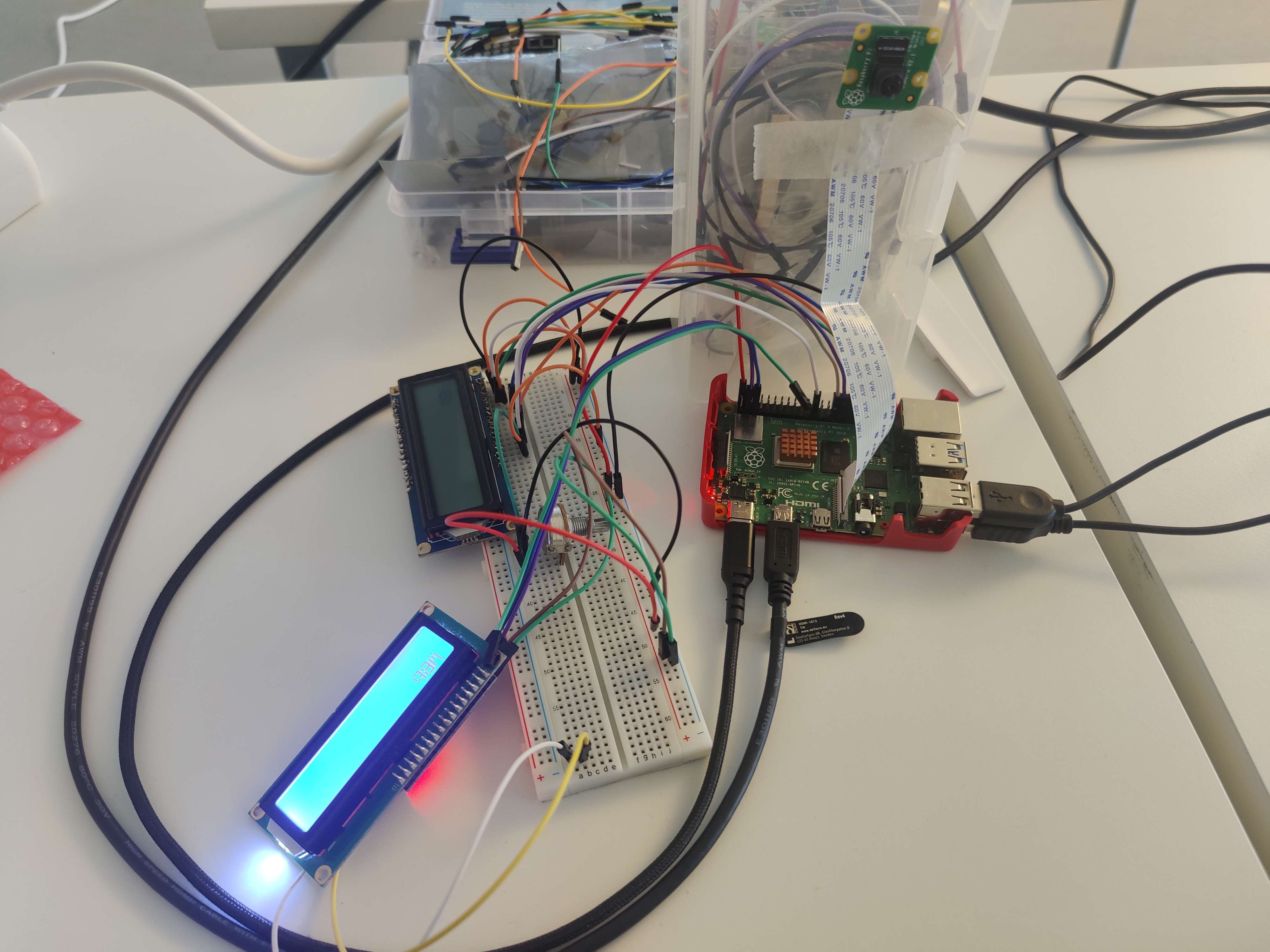
Only characters from Finnish sign language are recognized and only stationary signs are supported.

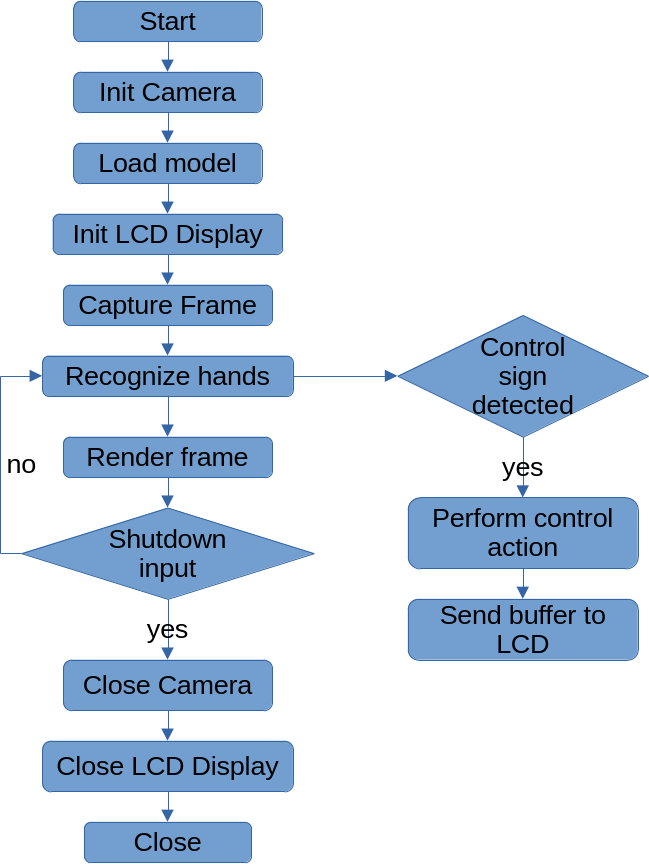
# Implementation

This chapter will discuss the final implementation and some changes made to the original plan.

## Changes to the original requirements

The whole still image idea was skipped since video capturing examples were readily available. Generating the data model is quite a long process, but it is still required to be made on Raspberry Pi to ensure generated binary compatibility. For cooling, the simple heatsink has been sufficient and the CPU temperature has not been risen much over 60°C when recognition program has been running. Simple cardboard casing is used instead of 3D printed one. Availability of two 1602 LCD displays led to the idea for providing them for both signer and receiver.

Figure 1: Project setup.

Figure 2: Inference flowchart

## Work Division

* Raspberry camera config: Jussi
* Camera Library: Sushila
* Hardware Setup: Sushila, Jussi
* 1602 LCD Display Library: Seppo
* Data Collection: Jussi, Sushila, Seppo, Teemu
* Creating Dataset: Jussi , Teemu
* DataAdder: Jussi
* Inference Classifier: Jussi, Teemu, Seppo, Sushila
* Train Classifier: Jussi, Teemu, Seppo, Sushila
* Presentation: Seppo, Teemu
* Documentation: Sushila, Jussi