POLITECNICO DI TORINO

Master's Degree in Mechatronics

Master's Degree Thesis

Hand gesture recognition based on Time-of-Flight sensors

Supervisors

Prof. Fabrizio LAMBERTI

Dr. Curti MASSIMILIANO

Candidate

Huda ELNIEMA ABDRAHMAN ABDALLA

July 2021

Summary

Since the emergence of computer systems,researchers have developed various solutions in gesture recognition in the context of human-machine interaction (HMI). Nowadays, the avant-garde in this field is touchless gesture recognition, as the name indicates a sort of communication that doesn't imply touching any sort of hardware.

Touchless technology has been incorporated in applications ranging from the industrial environment (e.g. human-robot interaction) to entertainment applications, to healthcare, and the automotive industry.

Until recently, much of the touchless gesture recognition has been focused on using computer vision techniques. However, these techniques require high computational power to understand images successfully and proved to be not helpful in the dark or other low visibility conditions. Based on the Gesture Recognition and Touchless Sensing Market Global Analysis, "sensor-based technology is anticipated to endure the largest share of the market during the next period" [1].

Two types of sensors are majorly utilized in touchless sensing devices: infrared sensors and capacitive sensors. However, each of these sensors has proven to have significant drawbacks. In this thesis, a design of a touchless sensor prototype has been proposed using Time-of-Flight (TOF) technology. The devised strategy aims to obtain the best configuration with a low cost and high reliability.

The proposed system is composed of a horizontal array of three TOF sensors. The sensors are employed for data collection regarding seven hand gestures, namely up, down, left, right, clockwise (CW), counterclockwise (CCW), and unknown, performed by the user at a predefined distance from the prototype. The hand gesture movements are classified using Deep Neural Networks (DNNs). Finally, the proposed approach is validated with a Graphical User Interface (GUI) representing a virtual cluster.

Moreover, this thesis offers an insight into the different technologies used in touchless sensing by providing a comparative performance evaluation of the well-known types of gesture recognition sensors available in the market.

Acknowledgements

First of all, I would like to express my sincere gratitude to Teoresi group for offering me the opportunity to work on this project. I would like to single out my supervisor, Dr.Massimiliano Curti; I want to thank him for your patience, support and for all of the opportunities I was given to further my work.

Furthermore, I would like to thank my supervisor, Prof. Lamberti, for the thoughtful comments and recommendations on this thesis.

To conclude, I cannot forget to thank my family and friends for all the unconditional support and their collaborative effort during data collection.

Table of Contents

List of Tables

List of Figures

Acronyms

HMI Human Machine Interface

TOF Time-of-Flight

DNN Deep Neural Network

GUI Graphical User Interface

CW Clockwise

CCW CounterClockwise

ML Machine Learning

IR InfraRed

FOV Field of View

PIR Passive InfraRed

USART Universal Synchronous Asynchronous Receiver/Transmitter

MLP Multilayer Perceptron

SVM Support Vector Machines

SGD Stochastic Gradient Descent

Adam Adaptive Moment Estimation

CNN Convolutional Neural Network

RNN Recurrent Neural Network

WAF Wrap Around Filter

ECE Early Convergence Estimate

Dmax Max Detection Range

NVM Non volatile memory

Chapter 1

Introduction

This chapter is structured to provide brief, intuitive contextualization to the thesis. It presents the thesis motivation, research objectives, research contributions, and thesis structure.

1.1 Motivation

As technology advances, human-machine interaction is expected to be more realistic and natural; therefore, there is significant interest in gesture recognition research. Furthermore, companies face a dilemma in choosing the correct gesture recognition sensor to embed in their devices.

Gesture recognition has a variety of applications such as in surgery operating rooms [2], collision avoidance in human-robot environments [3], automotive infotainment systems [4], and many other applications.

One of the technologies used in gesture recognition is vision based sensors; these sensors are the most advanced ones for natural interaction, even though they require high computational power, illuminations, colors, shapes, and a considerable amount of other noises could affect their outputs.

As an alternative, sensor-based technology is gaining attention since it satisfies the market new requirements for hand gesture recognition systems, including low power consumption, low cost and simple hardware setup.

This thesis aims to research the use of TOF technology in a gesture recognition system capable of recognizing a varied set of gestures such as swipe gestures and circle gestures. The classification model is developed by examining Multilayer Perceptron (MLP) to obtain good results and verify the advantage of deep learning technologies in training and execution time.

Additionally, the thesis provides a comprehensive evaluation of the TOF sensor prototype and compare it with the IR and capacitive sensors performance.

1.2 Research objectives

This research aims to explore the usage of TOF technology in the recognition of simple gestures. The research has the following objectives.

- Study the TOF technology, and select the sensors based on that technology.
- Design a prototype using the selected sensors.
- Implement multi-sensor ranging code to get the data from each sensor.
- Select the gestures set and collect training datasets of these gestures.
- Implement a gesture recognition model using DNNs.
- Validate the model in terms of accuracy and use the system in a virtual cluster GUI.

1.3 Contributions

This thesis is primarily concerned with gesture recognition using TOF sensors, evaluating their performance, and subsequently, comparing them with two other sensors available in the market.

1.4 Thesis structure

This thesis is organized into six chapters:

Chapter 2 gives an introduction to the technical knowledge required to understand the thesis. It also offers an insight into the previous research work done in the field of gesture recognition.

Chapter 3 discusses the criteria for selecting the hardware components in this thesis and provides a detailed description of the methodology and tools used for implementation and testing.

Chapter 4 delves into the design and implementation of the prototype. In addition, it illustrates the steps for implementing the DNN.

Chapter 5 discuss the results of tests performed on the different sensors.

Chapter 6 is where conclusions are drawn, and future work is proposed.

Chapter 2

State of the art

This chapter is structured to present relevant information to the area of work developed in the thesis. It will start by outlining essential concepts related to gestures, gesture recognition technologies, and their applications. Finally, it provides an introduction to DNNs.

2.1 Gestures

2.1.1 Definition of gestures

Gestures have always played an essential role in human communication and are among the most powerful ways for humans to communicate non-verbally [5]. A gesture is a form of communication in which visible body actions communicate particular messages.

Gestures can be complex, and the information communicated by the gestures can be complicated as well. Some gestures are universal and have identical meaning irrespective of the country or culture they are used in [6].

2.1.2 Types of gestures

There is a tremendous variety of literature for gesture classification, from machine learning and computer vision to psychology and linguistics. A possible classification of gestures is based on gesture change over time, where two types of gestures are distinguished: static gestures and dynamic gestures.

A static gesture is when the body part remains unchanged during a period of time [7]. If the body part is the hand, a typical static gesture could be the “stop” gesture and the “OK” gesture.

Figure 2.1: Examples of static gestures [8]

A dynamic gesture typically has a specific trajectory and transmits a message using this trajectory. These gestures are used in touch-based interfaces or touchless interfaces.

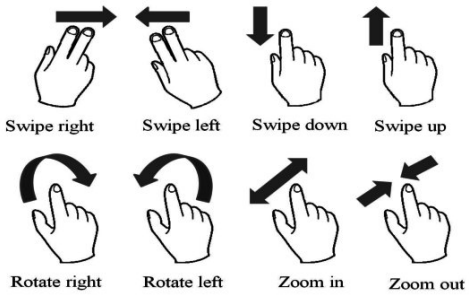


Figure 2.2: Examples of dynamic gestures [9]

2.1.3 Applications of gesture recognition

Gesture recognition is currently an ongoing research field that is applied in a variety of applications, ranging from applications that are related to humans safety such as :

- using hand gestures to control the infotainment system in cars to reduce driver distraction during driving as in [4].
- Gesture recognition in application related to healthcare in order to decrease the risk of contamination during surgical procedures [10] and recently contactless interfaces to reduce the spread of Covid19 [11].

Another set of applications is for human conveniences, such as gaming [12], electronic devices [13], and smart homes.

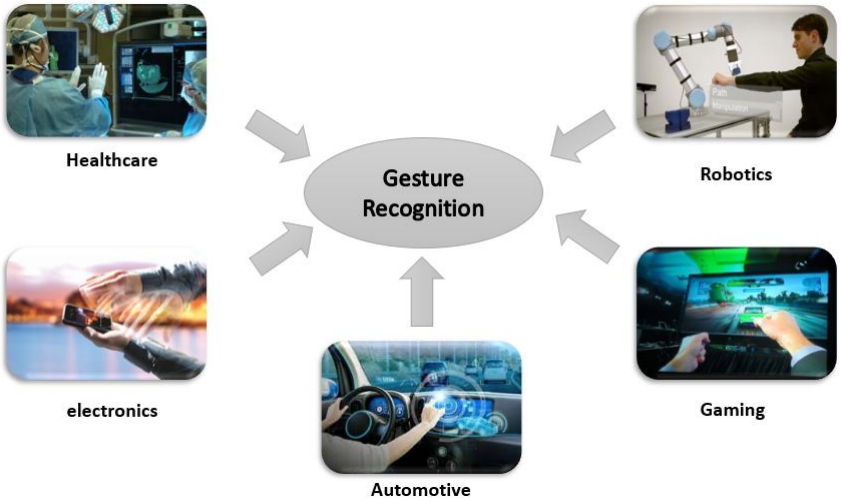


Figure 2.3: Example of gesture recognition applications

2.2 Gesture recognition technologies

This section will briefly present the technologies used in gesture recognition.

2.2.1 Capacitive sensors

Any living organism produces a small electric field generated by cell activity and ionic currents of the nervous system [14]. Consequently, it is possible to measure the impact of the human body in two methods.

- setting within an electrical field and measuring the influence of human body movement.
- coupling the human body to a transmitter and measuring the resulting electric field.

Capacitive sensing is a technology based on the capacitive coupling that can detect and measure anything that is conductive or has a dielectric different from air[15]. In [16], Joshua Smith introduced different measuring modes that can be distinguished in capacitive sensing.

- Transmit mode where the distance to the human body can be determined by coupling the body to a changing electric potential and receiving the transmitted signal using the grounded plate. In this mode, the sensor is defined as a receiver measuring the incoming displacement current [17].

- Shunt mode, employed in the presented hardware platform, is based on the principle that a grounded body part affects the electric field between a transmitter and a receiver electrode. In shunt mode, the definition of a sensor is not intuitively apparent. As each transmitter-receiver combination may deliver a unique measurement result, a sensor is defined as a combination between a transmitter and a receiver.
- Loading mode, one can measure the displacement current from a transmitter electrode to a grounded body part, in order to determine the distance [16]. Similar to transmitter mode, the displacement current is measured by the transmitter, which is considered to be a sensor.

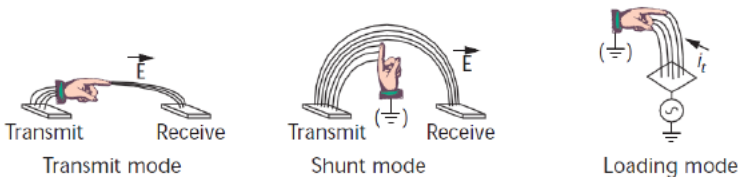


Figure 2.4: Measurement modes for capacitive proximity sensing [16]

An example of shunt mode based structure is the GestIC technology by Microchip Technology [18].It consists of transmit electrode which is at the bottom layer, and four smaller receiver electrodes are placed on the edges of the top layer, as shown in Figure 2.5.

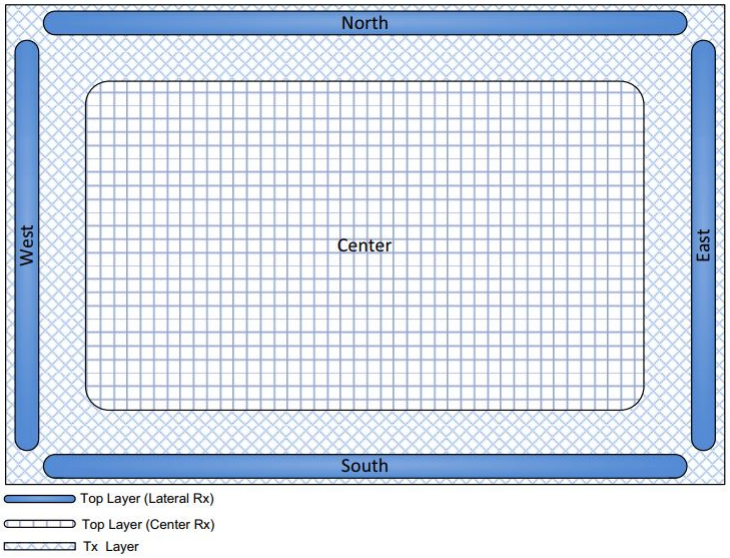


Figure 2.5: MGC3130 electrodes [18]

When it is expected that other objects might disturb the measurement of capacitive sensors, shielding is necessary. In case of no shielding, the electric field will span in all directions, as shown in Figure 2.6. There are several benefits of shielding, which are reported below.

- It leads and focuses the sensing zone to a particular area.
- It minimizes and annihilates parasitic capacitances and interference.
- It excludes temperature variation effects on the ground plane.

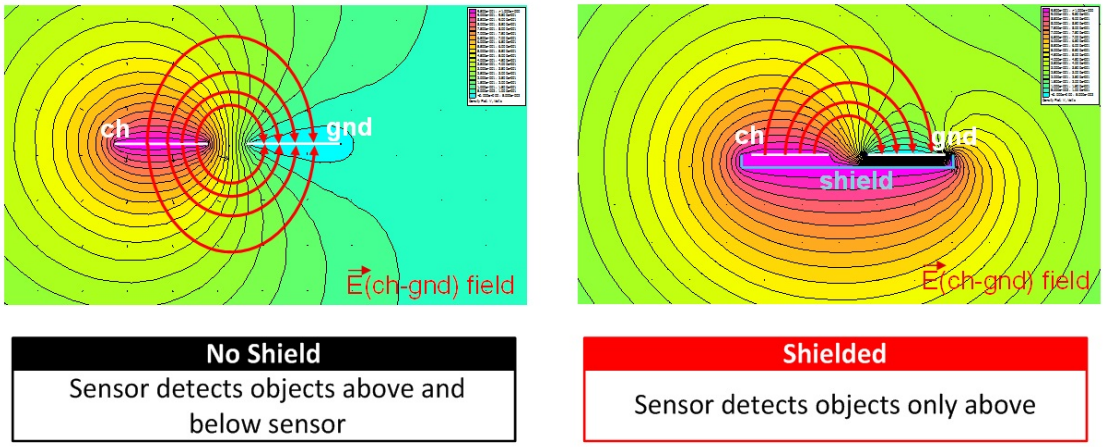


Figure 2.6: Shielding in capacitive sensors [19]

This following section presents a brief summary of researches using capacitive sensing for gestural interaction.

- Thracker

  Thracker is a prototype that embeds a regular monitor to detect hand gestures using capacitive proximity sensors, allowing two modes for interaction. As presented in Figure 2.7 [20], it has four sensors arranged around the screen. It has two modes which are 3D interactions and "Pick and Drop" interactions. The first mode allows the interaction with objects on the screen by performing a picking gesture, whereas the second movement will be interpreted as clicks (3 cm from the screen) and pointer for more faraway distances. Researchers concluded that by studying 10 participants, the gesture interface is comfortable and intuitive, yet they struggled to achieve good results when considering gestures such as zooming.

Figure 2.7: Thracker prototype [20]

- Swiss-Cheese Extended

  It is a prototype for gesture recognition using capacitive proximity sensors to detect the 3D position of multiple hands. The gestures supported are swipe right with one hand, and motion from bottom to top with two hands, plus supports zoom and rotation, grasp, and release.

  The prototype device operates on shunt mode measurements. It consists of eight transmitters that are located at the device's edges and receivers in the center. They achieved a resolution of approximately 3.5mm at object distances around 50mm and at object distances of 200mm.

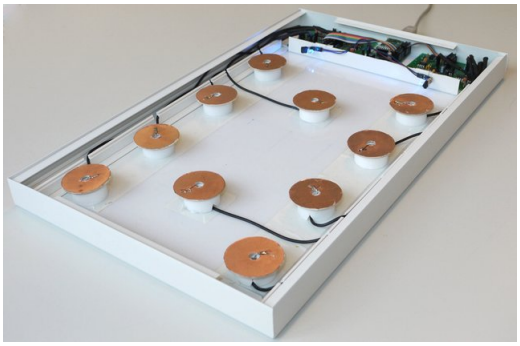


Figure 2.8: Swiss-cheese prototype [21]

In [22],the authors designed a prototype for reliable recognition of finger gestures by tracking the time history of capacitance variations of an electrode array.

It explained that given enough resolution along the timeline, the read-out electronics could be avoided. Concerning optimal electrical processing of the capacitance variations, a shielding strategy is adopted, which helped to suppress the body to ground capacitances and its variations due to changing environmental conditions.

2.2.2 Ultrasonic sensors

Ultrasonic sensors operates on emitting sound waves at frequency high (>20KHz) for humans to hear. They anticipate the sound to be reflected, calculating the distance based on the time required, as shown in Figure 2.9.

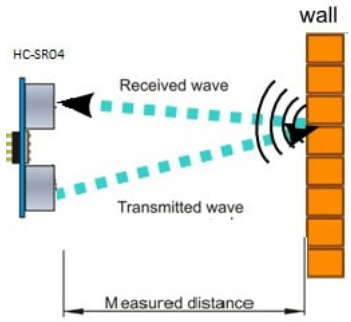


Figure 2.9: Ultrasonic working principle

This technology has various drawbacks. It is highly affected by the changes in temperature, pressure, and humidity. Secondly, it has more difficulties in reading reflections from soft, thin, and small objects.

2.2.3 Radar sensors

Radar is a technology alike ultrasound in terms of the concept, because both rely on the propagation and reflection of wave signals. Radar, though, works not with sound waves but with electromagnetic waves (Frequency back MHz to GHz). The target should not be non-conductive materials or any materials with low dielectric constants.

The utilization of this technology in the field of gesture recognition is still in progress, but there are some prototypes under development. Currently, the disadvantages of this technology are radar size and cost.

2.2.4 Infrared sensors

An IR sensor is an electronic device that measures and detects infrared radiation in its surrounding environment. Infrared radiation was accidentally discovered by an astronomer named William Herchel in 1800. While measuring the temperature of each color of light (separated by a prism), he noticed that the temperature just beyond the red light was highest. IR is invisible to the human eye, as its wavelength is longer than that of visible light (though it is still on the same electromagnetic spectrum). Anything that emits heat (everything that has a temperature above around five degrees Kelvin) gives off infrared radiation [23].

Active IR sensor

Active IR sensors operate using particular light-sensing elements which are sensitive to electromagnetic waves. In detail, an infrared light source emits light within a specific FOV when the light hits an object and returns to the sensing device. Based on the time or intensity, it is possible to calculate the distance of the target. A TOF sensor is an example of active IR.

Passive infrared sensor

Passive infrared sensors (PIRs) are sensitive to thermal radiation emitted by the human body in the range of 8 - 14µm [24]. Tiny deviations from the thermal equilibrium of the surrounding environment can be detected [24].

PIR sensing is commonly used in commercial applications to detect the presence of humans or trigger alarms. PIR sensors have also been explored for much more complex applications such as human localization[25], motion direction detection [26],etc.

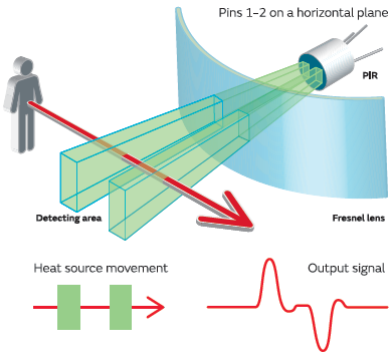


Figure 2.10: PIR working principle [27]

2.2.5 Camera systems

The camera,vision-based sensor is a common, suitable and applicable technique because it provides contactless communication between humans and computers [28].In terms of camera technologies in gesture recognition, two main types are considered: 2D and 3D camera systems.

- 2D camera systems

  2D camera systems have limited use in gesture recognition. The information that can be collected is limited (since there is no depth), so the gestures detected are usually simple like finger gestures, and motion tracking [29].

- 3D camera systems

  3D camera systems enable high-quality video processing, facial recognition, 3D imaging, noise cancellation, etc. TOF and structured light are only two of the several existing 3D imaging techniques.

  – TOF cameras

    A TOF camera uses laser or IR light to calculate distance between camera and subject by measuring the time taken by the light signal to make the round trip. The camera illuminates the scene with a varying light source and registers the reflected light, then translates that into a distance measurement. The simplest version of a TOF camera uses light pulses or a single light pulse. The illumination is switched on for a very short time, the resulting pulse lights the scene and is reflected by objects in its FOV [30].

  – Structured-light cameras

    Structured-light illumination (SLI) [31] is a non-contact, optical, active, triangulation-based 3D reconstruction technique known for its simple implementation, low cost, and high accuracy. However, the high rate of processing demanded in real-time has, until now, proved unattainable.

    Structured light works on the following principle, where an object's coordinate in 3D space is derived by triangulating between pixels of two cameras. SLI avoids the computational complexities of matching pixels across camera views by replacing one of the two-component cameras with a projector that generates a series of striped patterns. By analyzing the change in the pattern at a particular point on the target object surface (a process known as demodulating the captured images), unique correspondences can be derived between the camera and projector pixels [32].

2.2.6 Time of flight technology

Basic concepts

TOF is an accurate and easy to understand technology used for distance measurement.It measure distances using the time that it takes for photons to travel between two points, from the sensor emitter to a target and then back to the sensor receiver. The following formula computes the distance:

$$d = \frac{1}{2} c\tau \tag{2.1}$$

Where d is the measured distance, c is the speed of light, and τ is the photons travel time.

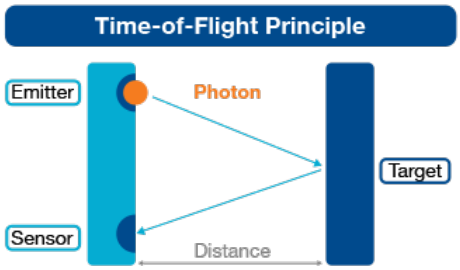


Figure 2.11: TOF working principle [33]

There are two ways to measure TOF.

- In the direct method, the sensor transmits pulses that last few nanoseconds and, consequently, measure the time it needs for the emitted light to return.
- In indirect method, a continuous modulated sinusoidal light wave is emitted, and the phase difference between outgoing and incoming signals is used to compute how fat the target is.

Both types are shown in Figure 2.12.

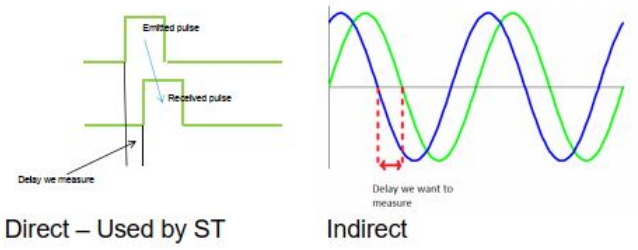


Figure 2.12: Direct and indirect TOF [33]

The foremost advantages of using a TOF sensor are listed below.

- The reflectance of the target does not influence the measurement.
- It can operate in low light conditions.
- The system is compact.
- It presents accurate results.

There are also some constraints when using this technology.

- It is hard to use outdoors because high-intensity light can saturate the sensor.
- If the light is reflected multiple times (on corners), it can alter the measurement.
- The cover glass distorts the reflected signal

FlightSense™

This technology was patented by the semiconductor company STMelectronics and solved some of the issues that have the TOF sensors. First of all, there is the compensation algorithm for correcting the distortion produced by the cover glass. As the cover glass is always located in the same place, it has fixed optical characteristics that are used to correct the measurement. Another system improvement is about the performance outdoors. The technology keeps ambient photons out with optical filtering and also rejects the remaining ones (lower wavelengths) thanks to a time-domain rejection [34].

These sensors have been used successfully in number of applications such as human robot interaction[35] and collision avoidance [36][3].

2.3 Artificial intelligence

Artificial Intelligence (AI), is one of the most promising interdisciplinary science; it acts as the main driver for emerging technologies like: robotics, IoT, and big data. Continuously, its applications are becoming highly involved in daily life, from predicting cancers, online shopping predictions, and auto correct.

The father of AI is the British logician and computer pioneer Alan Turing. He proposed a 'Turing test' in 1950 designed to provide an operational definition of intelligence. A necessary tenant of the Turing test is that the computer does not have to think as we think. Instead, the computer must simulate intelligence to be indistinguishable from our intelligence. If a machine passes Turing test, it is said to be intelligent. But no machines have completely passed this test as of yet [37].

2.4 Machine learning

Machine learning is a branch of artificial intelligence, simply put by Former Chair of the Machine Learning Department at Carnegie Mellon University, Tom M. Mitchell as follows : "Machine learning is the study of computer algorithms that improve automatically through experience."[38]

Machine learning uses historical data as input and predicts the expected output. There are three types of machine learning: supervised learning, unsupervised learning, and reinforcement learning.

Supervised learning is the most basic type of machine learning; here, the algorithm is trained on labeled data, it finds relationships between the parameters given (training datasets and labels). In unsupervised learning, the algorithm is trained on unlabeled data, thus creates relationships between any two data points. Lastly, reinforcement learning is where the algorithm improves upon itself learning from new situations using the trial-and-error method. Desirable outputs are "rewarded", and non-desirable outputs are "punished".

Classification problems belong to the supervised learning category, and they are defined as the process of predicting the class (output) for a specific series of features(inputs). This thesis implements supervised learning as the dataset includes labeled gesture data which is a supervised classification task.

2.4.1 Supervised learning

The input variables are denoted as input features, whereas the output variable is denoted as a target. A pair of input features and target variable is called the

training set. A validation dataset is a separate section of the dataset (usually 20-30%) used to estimate model skill while tuning its hyperparameters.

2.4.2 Classification

In machine learning, classification refers to a predictive modeling problem where a class label is predicted for a given example of input data [39].

2.4.3 Deep learning

Deep learning is a subset of machine learning, which on the other hand, is a subset of AI. Deep learning is inspired by the biological neurons of the human brain. Therefore, deep learning algorithms try to draw the same conclusions as the human brain by analyzing the data. Achieving the goal of deep learning requires a multi-layered structure called a neural network.

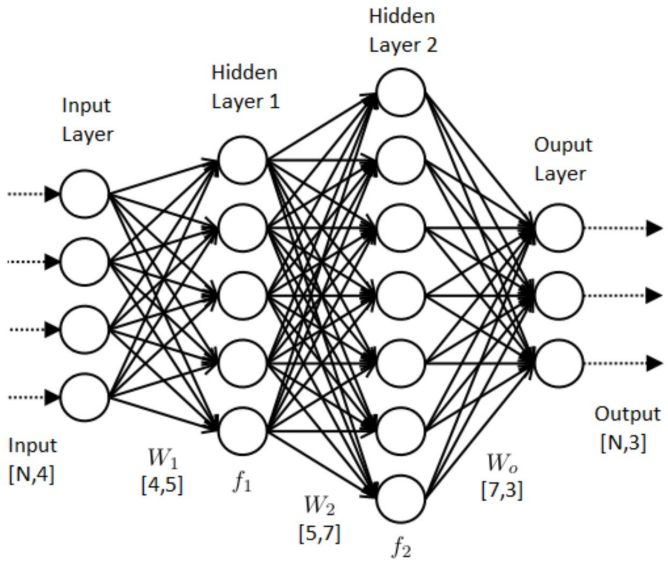


Figure 2.13: Deep Neural Network

Why deep learning is so popular

The first advantage of deep learning over machine learning is that it removes the need for feature extraction, which is usually quite complex. All machine learning algorithms are called flat algorithms, which means they can't be applied to raw data (.csv, images, etc.). On the other hand, deep learning models are capable of learning directly from the raw data.

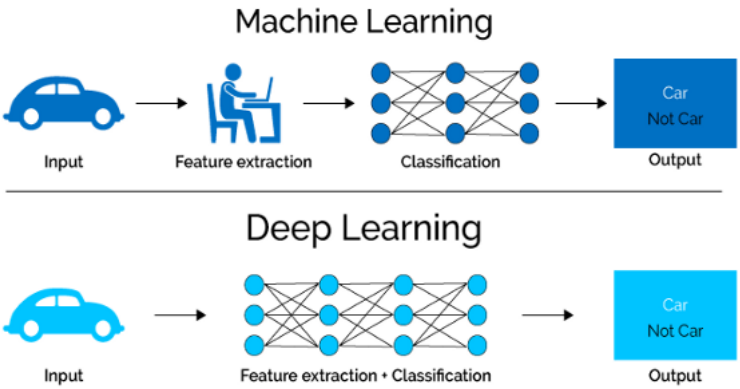


Figure 2.14: Deep learning vs. machine learning [40]

The second advantage of the deep learning models is that they manage to increase their accuracy with the increasing amount of training data, whereas traditional machine learning models such as Support Vector Machines (SVM) and Naive Bayes classifiers stop improving after a saturation point.

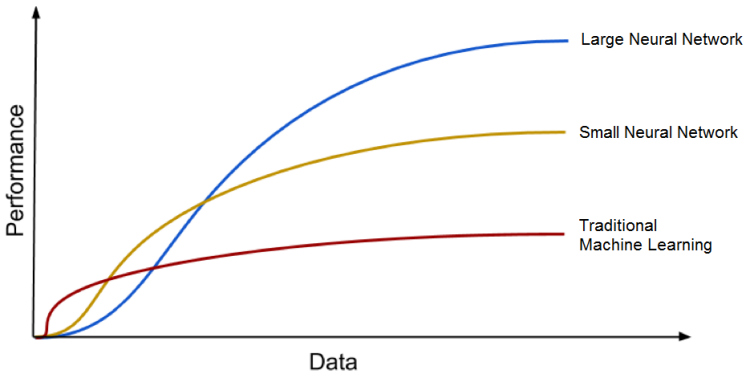


Figure 2.15: Deep learning algorithms [40]

Perceptron

The perceptron was first introduced by American psychologist Frank Rosenblatt in 1957 at Cornell Aeronautical Laboratory [41]. He worked on the model introduced by Warren McCulloch and Walter Pitts [42] in 1943, where they disputed that neurons with a binary threshold activation function were comparable to first order logic sentences [43].

He was also inspired by the work of Donald Hebb, which later became referred to as Hebb's rule. Hebb's rule states that "When an axon of cell A is near enough to excite a cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased" [44]. Rosenblatt's model of a perceptron was learning in the "Hebbean" sense through the weighting of inputs. Figure 2.16 illustrates the mathematical model for the perceptron and its biological inspiration.

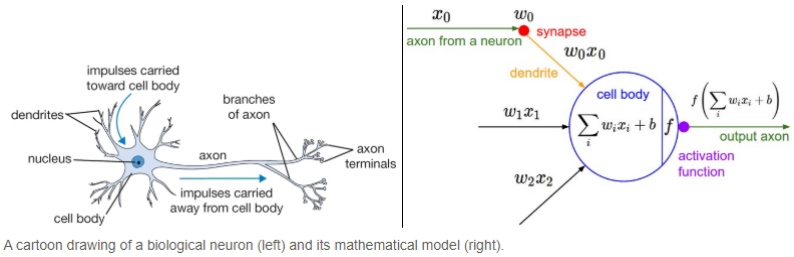


Figure 2.16: Biological inspiration for the perceptron [45]

Feed-forward networks

In feed-forward networks, information streams from the left to the right of the model, as in Figure 2.15. The input features x are used to compute the responses of the first layer through an activation function. These computed values are then fed into the next hidden layer as inputs and this process is performed by all neurons in all layers until the output layer, whose neuron output, is the result of the network:

$$z = \sum_{i}^{N} x_i w_i + b \tag{2.2}$$

$$a_{out} = f(z) \tag{2.3}$$

where w is weight and x is the input and f is the activation function

Activation function

The activation function defines the output of a node given an input or set of inputs. The activation functions that are widely used are the Sigmoid, Piecewise Linear, ReLu, and Softmax. ReLu and Softmax are the most popular.

ReLu is famous as the function is one of the most widely used, and it has proven to be faster and more efficient for large neural networks due to its linear nature [46]. Lastly, produce a good approximation of a target function often a highly nonlinear function is needed; usually, the sigmoid function is used for the output layer.

Figure 2.17 illustrates a summary of some available activation functions in Keras. Keras is a powerful and easy-to-use free, open-source Python library for developing and evaluating deep learning models [47].

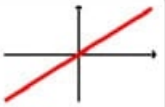
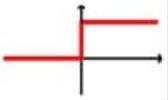
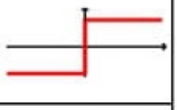
Activation Function	Equation	Example	1D Graph
Linear	$\phi(z) = z$	Adaline, linear regression	
Unit Step (Heaviside Function)	$\phi(z) = \begin{cases} 0 & z < 0 \\ 0.5 & z = 0 \\ 1 & z > 0 \end{cases}$	Perceptron variant	
Sign (signum)	$\phi(z) = \begin{cases} -1 & z < 0 \\ 0 & z = 0 \\ 1 & z > 0 \end{cases}$	Perceptron variant	
Piece-wise Linear	$\phi(z) = \begin{cases} 0 & z \leq -\frac{1}{2} \\ z + \frac{1}{2} & -\frac{1}{2} \leq z \leq \frac{1}{2} \\ 1 & z \geq \frac{1}{2} \end{cases}$	Support vector machine	
Logistic (sigmoid)	$\phi(z) = \frac{1}{1 + e^{-z}}$	Logistic regression, Multilayer NN	
Hyperbolic Tangent (tanh)	$\phi(z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}}$	Multilayer NN, RNNs	
ReLU	$\phi(z) = \begin{cases} 0 & z < 0 \\ z & z > 0 \end{cases}$	Multilayer NN, CNNs	

Figure 2.17: List of activation functions available in Keras [40]

Loss function

When the forward propagation is performed to form the initial prediction, an error function (E) called the Loss function determines how far the result is from the actual value. There are two commonly used Loss functions in Keras:

$$E(y, \hat{y}) = -\sum_{i=1}^{k} y_i.log(\hat{y}_i) \tag{2.4}$$

$$E(y, \hat{y}) = \frac{1}{m}\sum_{i=1}^{m}(y_i - \hat{y}_i)^2 \tag{2.5}$$

where m is the total number of training examples and k is the output size, and y is the target label, and $\hat{y}$ is the predicted value. Each equation works for a particular type of problem. Hence,Equation 2.4 represents categorical CrossEntropy ,is used in classification problems,while Equation 2.5 a Mean Squared Error,is used for regression.

The goal then becomes to find a set of weights that reduces the value of E over the whole training set. The mean of the Loss function utilized to all samples of the training test is known as the Cost function. Individual weight impact on the loss function is determined via backpropagation.

Backpropagation

In order to obtain the optimal set of weights, backpropagation is used. Backpropagation is an efficient method of computing the first derivative of the error function with respect to the neural network parameters. It works by computing the gradients at the output layer and using those gradients to compute the gradients at the previous layer, and so on.

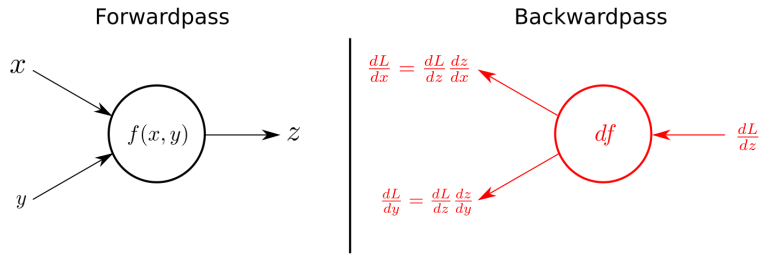


Figure 2.18: Forwardpass and backwardpass in backpropagation [45]

Learning rate

The learning rate is a hyperparameter that controls how much we adjust the network weights concerning the loss gradient. In other words, the learning rate is the size of steps taken to reach the minima [40]. In the case of using a low learning rate, this means the gradient descent takes many steps to converge and can be stuck in a plateau region. On the other hand, when the learning rate is too high, the gradient descent fails to reach the minimum.Both cases are shown in Figure 2.20.

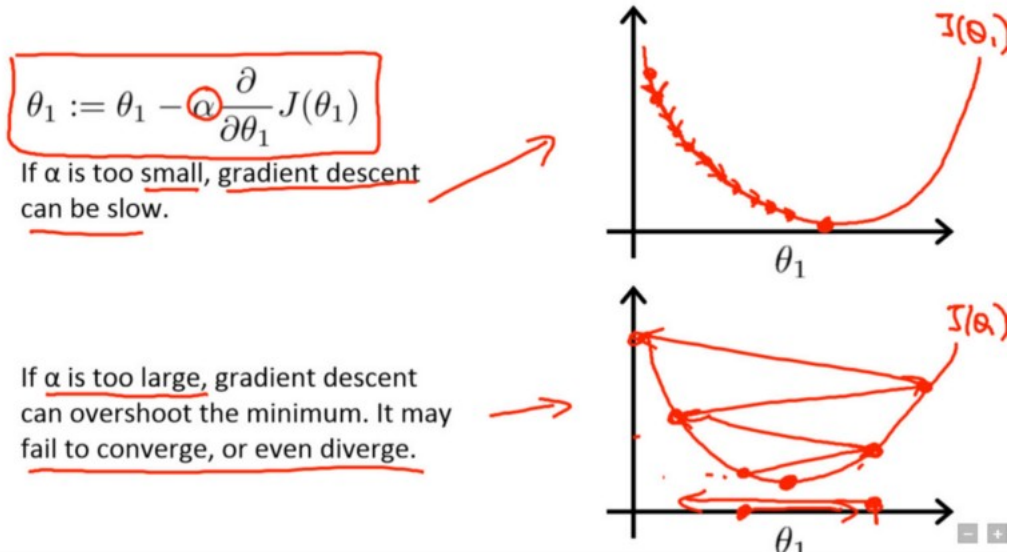


Figure 2.19: Gradient descent with small (top) and large (bottom) learning rates [45]

Leslie N. Smith [48] demonstrated that it is possible to estimate a reasonable learning rate by training the model initially with a very low learning rate and increasing it (either linearly or exponentially) at each iteration.

If the learning is recorded at each iteration and plotted as in the Figure 2.20, there will be a point where the loss stops decreasing and starts to increase.

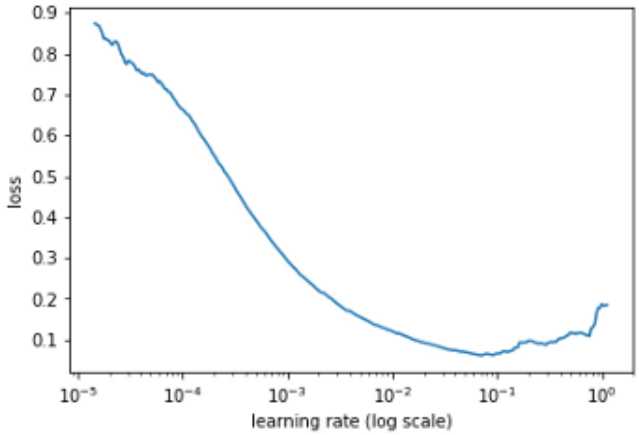


Figure 2.20: Learning rate [48]

Epochs

The number of epochs is a hyperparameter that defines the number of times that the learning algorithm will operate on the entire training dataset. One epoch means that each sample in the training dataset has had an opportunity to update the internal model parameters [39].

Batch Size

The batch size defines the number of samples propagated through the network after which parameter update happens [39]. Good default for batch size might be 32,64,128,etc.

Optimizers

Optimizers are algorithms used to change the weights and the learning rate in order to reduce the loss. There are different types of optimizers. Some are classic optimizers such as Gradient descent and Stochastic gradient descent (SGD), and there are adaptive gradient descent algorithms such as Adagrad, Adadelta, RMSprop, Adam.Adam optimizer is used in this thesis.

Adaptive moment estimation (Adam) was presented by Diederik Kingma from OpenAI and Jimmy Ba from the University of Toronto in 2015 [49].

They described the benefits of using Adam on non-convex optimization problems, as follows.

- It is straightforward to implement and computationally efficient.

- It is well suited for problems that are large in terms of data and parameters.
- It is appropriate for problems with very noisy/or sparse gradients.

"Adam computes individual adaptive learning rates for different parameters from estimates of first and second moments of the gradients" [49].

In [49], Adam was demonstrated empirically to show that convergence meets the expectations of the theoretical analysis. Adam was applied to the logistic regression algorithm on the MNIST digit recognition and IMDB sentiment analysis datasets, a MLP algorithm on the MNIST dataset, and Convolutional Neural Networks on the CIFAR-10 image recognition dataset. The authors conclude: "Using large models and datasets, we demonstrate Adam can efficiently solve practical deep learning problems."[49]. The performance of different optimizers is shown in Figure 2.21 .

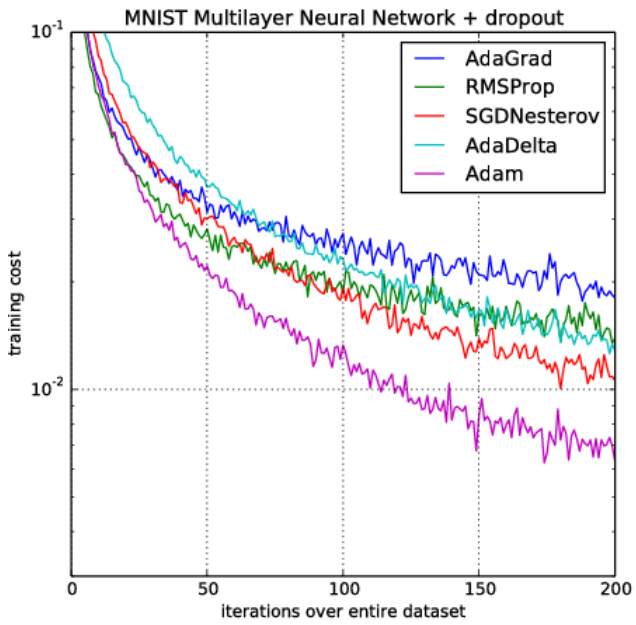


Figure 2.21: Comparison of Adam to other optimization algorithms training a MLP [49]

2.4.4 Categorical data

Categorical data are variables that contain label values rather than numeric values such that each value represents a different category [39]. Many machine learning

algorithms require all inputs and output variables to be numeric. This leads to why one needs to convert the categorical data to numerical form. There are two methods to convert the categorical data.

- Ordinal coding: here, each different category is assigned to an integral value, for example, place: “first” is 1, “Second” is 2, and “third” is 3. The drawback of this method is allowing the model to assume a natural order between categories and leads to poor performance in the case where no ordinal relationship exists.

- One hot encoding: it is preferable when there is no ordinal relationship between categories. This is the method considered during this thesis, taking our gesture labels as an example: A “1” value is located in the binary variable for the gesture and “0” values for the other gestures.

Gestures	Left	Right	Up	Down	CW	CCW	Unknown
	1	0	0	0	0	0	0
	0	1	0	0	0	0	0
	0	0	1	0	0	0	0
	0	0	0	1	0	0	0
	0	0	0	0	1	0	0
	0	0	0	0	0	1	0
	0	0	0	0	0	0	1

Table 2.1: Example of gesture labels using one hot encoding

2.4.5 Multi-layer perceptron

A MLP is a feedforward artificial neural network model that maps input data sets onto a set of appropriate outputs. An MLP consists of multiple layers of simple nodes that interact using weighted connections [50].

The MLP can work as a universal approximator [51], and it is fast to implement and require lower CPU utilization compared to other approaches such as Convolutional neural network (CNN), recurrent neural network(RNN), etc.

2.4.6 Overfitting and underfitting

An essential consideration in machine learning is how the model generalizes to new data. By generalizing how well the model works on unseen data.

There are terms used in machine learning when speaking about how well a machine learning model learns and generalizes new data, namely overfitting and underfitting. Overfitting and underfitting are the two most significant causes of poor algorithm performance.

Underfitting happens when the neural network cannot accurately predict the training set, not to mention the validation set.Underfitting can be avoided by adding more training samples .

On the other hand, overfitting is when the neural network is good at learning the training set but is unable to generalize to unseen examples. Overfitting can be avoided by early stop (stop the training when the model starts to over fit)and adding dropouts (a hyperparameter that randomly prevents from learning a certain percentage of neurons in every training iteration).

Chapter 3

Technologies

3.1 Technology specification

This chapter presents the research methodology adopted in this dissertation, followed by an overview of the gesture recognition system design and software and hardware requirements specification.

3.1.1 Technologies selection criteria

For this dissertation, it's crucial to select a technology that can be used in gesture recognition. There are many technologies already on the market, and the selection of the technologies will be based on:

- Availability in the market
- Detection range
- Size
- Cost
- Efficiency

3.1.2 Selection of technology

Based on the criteria defined above and taken into account market evaluation,TOF technology was chosen.

Other technologies were taken into account such as IR and capacitive technologies that will used for comparison purposes.

3.1.2.1 TOF vs other technologies

Based on the three options considered ,a general comparison is reported in the table 3.1.

	Capacitive	IR	TOF
size/Weight	Small/light	Small/light	Small/light
Mechanical integration	Complex	Easy	Easy
Signal amplitude	No	Yes	Yes
Real distance output	No,unprecise	No (Computed)	Real distance in mm
Minimum distance	0cm	0cm	0cm
Maximum distance	Few cms	20cm	Up to 4 meters
Reliable (vs objects color and reflectance)	No, may detect target in all directions	No, impacted	Yes
Reliable (vs material finish and roughness)	No,sensitive to body or object change	No, angular dependency	Yes, angular dependency

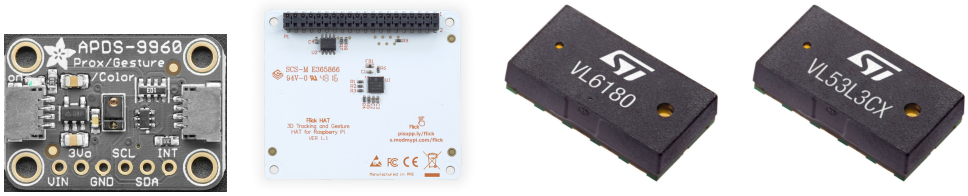
Table 3.1: Comparison of capacitive ,IR and TOF technologies

Referring to the Table 3.1 ,it is possible to highlight the advantages of TOF in terms of distance ,range as well as reliability.

3.1.3 Hardware selection and specifications

3.1.3.1 Selection of sensing devices

After market surveying and taking into consideration the detection range and cost.Four sensors have been selected for analysis and development ,as illustrated in the Figure 3.1.



(a) APDS9960 **(b)** Flick hat **(c)** VL6180 **(d)** VL53L3CX

Figure 3.1: Sensing devices

The first sensor is APDS9960 Melopero in Figure 3.1a.This sensor is an IR sensor which has the ability to detect up, down, right and left gestures or more complex gestures.

The second sensor is MGC3130 in Figure 3.1b.It is the first electrical field (E-field) based sensor; it features a detection range of 10 cm. Moreover, it allows the acquisition of positional data.

The Flick hat board contains MGC3130, which allows the following gestures: left, right, up, down swipes as well as rotational gestures clockwise(CW) and counterclockwise(CCW).

The third sensor is a TOF sensor named ST VL6180 in Figure 3.1c. This sensor can provide absolute distance independent of target reflectance and ranges up to 2 meters or 6 meters with reduced resolution.

The sensor in the Figure 3.1d is VL53L3CX which works using the same technology as the third sensor. It has an accurate range (up to 5 meters) and can detect multiple targets.

3.1.4 Criteria for the selection of development board

The selection of the development board depend on number of factors ,which are listed below :

- Availability in the market
- Cost
- Performance
- Communication protocols (I2C,SPI and UART)

3.1.5 Selection of development board

Flick hat is compatible the raspberry pi,therefore the only development board can be used is Raspberry pi.The model that is used is Raspberry pi4 model B Figure 3.2,owing to Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz

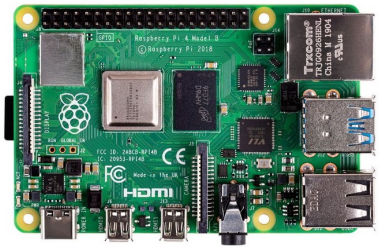


Figure 3.2: Raspberry Pi4-Model B

The development board Wemos D1 R2 Figure 3.3 has been utilized with APDS-9960 because of its cost and compatibility with Arduino.

Figure 3.3: Wemos D1 R2

The development board STM32F401RE in Figure 3.4 from ST,which will be used with time of flight sensors VL6180 and VL53L3CX. It was picked due to its architecture ARM Cortex M4 and its compatibility with TOF sensor's expansion board.

Figure 3.4: STM32F401RE

3.2 Software specification

In this section,the development tools and programming languages utilized in the thesis are illustrated..

3.2.1 Programming languages

Based on the selection of sensing devices and the development board, it was concluded that C programming and Python language are preferable.Both languages are supported by most development platforms.

3.2.2 Software platform

STM32Cube IDE

"STM32CubeIDE is an advanced C/C++ development platform with peripheral configuration, code generation, code compilation, and debug features for STM32 microcontrollers and microprocessors. It is based on the Eclipse®/CDT framework and GCC toolchain for the development, and GDB for the debugging" [52].

It is described as a set of free tools and embedded software bricks to enable fast and easy development on the STM32, including a Hardware Abstraction Layer and middleware bricks.

Arduino IDE

"The Arduino Integrated Development Environment (IDE) is a cross-platform application (for Windows, macOS, Linux) that is written in functions from C and C++.It is used to write and upload programs to Arduino compatible boards, but also, with the help of third-party cores, other vendor development boards" [53].

3.2.3 System architecture

The overall system design that has been used for the prototyping is shown in Figure 3.5 below:

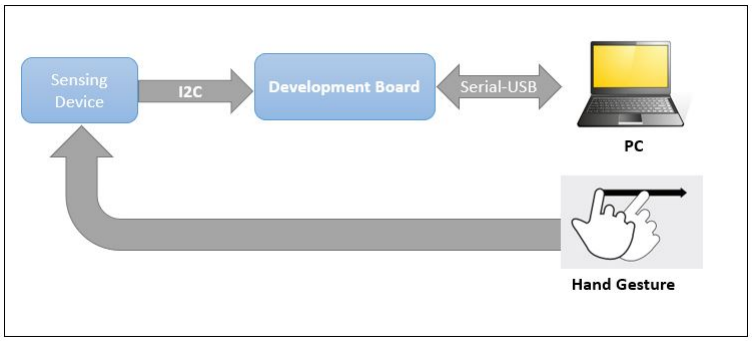


Figure 3.5: System Architecture

The architecture comprises the sensing devices, development boards,the PC that is used to collect measurements, and the DNN that classifies the gestures. Since the sensors have different technologies and interfaces, the following sections will be dedicated to discuss each sensor.

- **Melopero APDS-9960**

  The APDS-9960 is a multi purpose sensor that can be used in gesture detection,proximity sensing, ambient light sensing and color sensing

  The sensor uses I2C communication protocol, therefore it is easy to use with microcontrollers. It operates on a voltage range of 2.4V-3.6V (typically 3.3V) and consumes a small current of 0.2mA, so it is considered a low power consumption device.It has a detection range up to 20 cm. Focusing on the architecture of the gesture engine,it features automatic activation (based on proximity engine results), ambient light subtraction, crosstalk cancellation, dual 8-bit data converters, power-saving inter-conversion delay, 32-dataset FIFO, and interrupt-driven I2C-bus communication.

  The APDS-9960 sensor has four photodiodes to collect data from hand movements. The reflected IR energy, sourced by integrated LEDs, is converted from motion to digital data. Whenever a gesture is performed, the IR signal transmitted by the LED is reflected by the obstacle and then detected by the photodiodes. Then the motion information is converted to digital data .

A data set is defined as 4-byte directional block corresponding to U-D-L-R photodiodes. An interrupt is generated based on data available in the FIFO [54].Even though the I2C is compatible up to 40Hz (fast mode),this sensor doesn't detect the motion of gestures easily.

The directional sensors are designed such that the diode opposite to the directional motion endures a more significant portion of the reflected IR signal upon entry, then a smaller amount upon exit [54]. The Figure 3.6 illustrates a downward or rightward motion of a target.

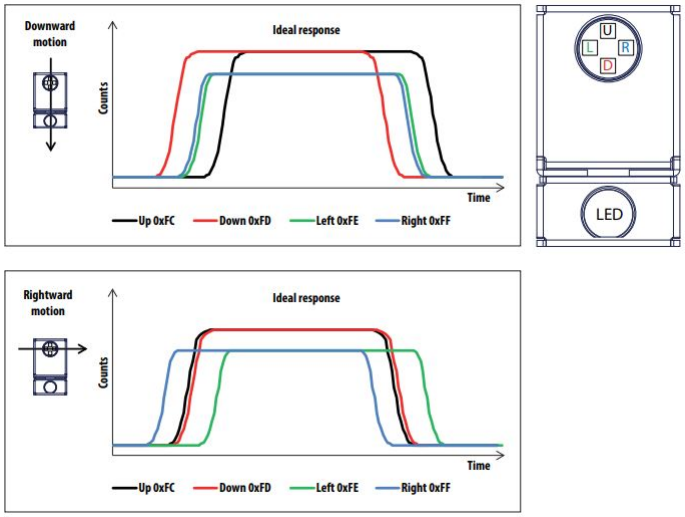


Figure 3.6: Directional swipes [54]

- **Flick Hat-MGC3130**

  Microchip Technology's MGC3130 is a three-dimensional (3D) gesture recognition, motion tracking, and approach detection controller based on Microchip's patented *GestIC*© technology for embedded usage. It enables user command input with natural hand and finger movements [18].

  The sensor has the following key features

  – Recognition of 3D hand positional data x,y,z
  – Proximity capabilities
  – 3D signal processing unit
  – The sensor ranges from 0 to 15 cm

- Position rate : this sensor is able to detect 200 positions/sec
- In order to detect the motion the sensor has only four of five receive (RX) channels and one transmit channel
- Contains on-chip auto calibration

The MGC3130 Single-Zone 3D Tracking and Gesture Controller Data Sheet [55] states the following applications for the MGC3130: Audio products, Notebooks/Keyboards/PC Peripherals, Home Automation, White Goods, Switches/Industrial Switches, Medical Products, Game Controllers, Audio Control.

This sensor utilizes an electric field (E-field) that is generated by electrical charges and spreads in the three dimensions around the surface, carrying an electric charge.MGC3130 can generate a Tx signal of about 100 kHz, which corresponds to a wavelength of 3 km with electrode geometries less than 14x14 cm .

The sensor works by detecting variations in a self-generated magnetic field by the introduction of conductive objects such as fingers.In the Figure 3.7 ,the field lines are lured by the hand due to the conductivity of the human body itself and shunted ti the ground.GestIC technology uses a minimum of four RX electrodes to detect the origin of the electric field variations; the collected information is used to calculate the position and track movements.

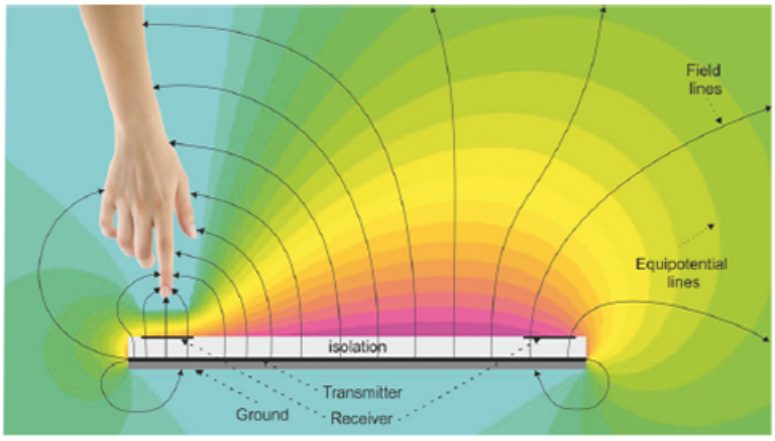


Figure 3.7: Equipoterntial lines of a distorted E-Field [18]

The maximum range of detection is 15 cm in the perpendicular axis to the sensor. However, from empirical testing, the range for detection appears to be consistently less than 3 cm in any direction from the center of the device [56].

"Major drawback of this sensor ,is its sensitivity to changes in ambient lighting, necessity of readjustment for various species and even degrees of ripeness, difficulties to recognize objects of interest in heavy foliage and affection by atmospheric conditions (for, rain,dust etc.)" [57].

STMicroelectronics TOF sensors

STMelectronics is a world leader in TOF solutions,offering 20+ years of innovation in imaging and optical sensing solutions. In the last decade, ST has introduced a new generation of high-performance proximity and ranging sensors based on FlightSense™ ToF technology [58].

ST has pioneered and transitioned TOF technology from its research labs to a fully industrialized family of market-leading products. Up till today, ST has 4 generations of products and working with more than 50 OEMs as well as 42,000 development kits, circulating among customers and in the market [58].

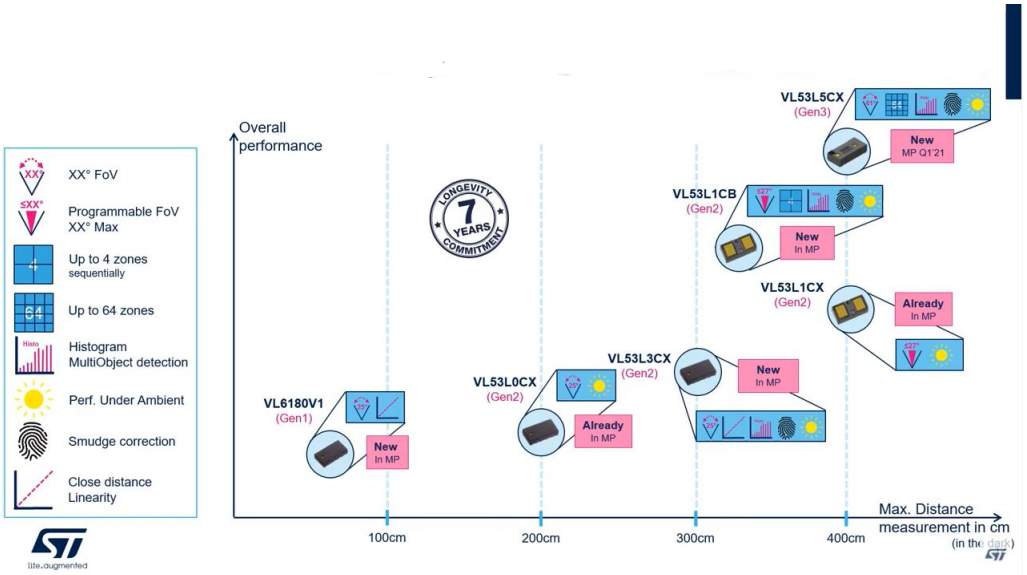


Figure 3.8: FlightSense™ roadmap

For the purpose of this thesis, two TOF sensors have been elected.

- **VL6180V1**

  VL6180 is based on ST's patented technology. It precisely measures the time the light takes to reach the nearest object and reflect back to the sensor. The VL6180 is straightforward to integrate and designed for low power operation.

  This sensor features a two-in-one optical module (proximity sensor and VCSEL IR light source 850 µm). It can range up to 62 cm(depending on the conditions), independent of object reflectance and ambient light rejection.

  Main features definition

  This part defines the main features of the VL6180:

  Ranging

  It is the measurement of the distance between VL6180 and the target.The sensor provides three options to extend the range with less resolution to make the sensor adaptable to multiple applications:

  – Upscale factor = 1, VL6180 measures distances up to 20 cm with a granularity of 1 mm.
  – Upscale factor = 2, VL6180 measures distances up to 40 cm with a granularity of 2 mm.
  – Upscale factor = 3, VL6180 measures distances up to 60 cm with a granularity of 3 mm [59].

  The range output of VL6180 with each target should be linear with range.Figure 3.9 shows the typical output from VL6180 for different targets(Munsell gray target) at different distances; the test has been performed in the dark with no cover glass. It is possible to have an offset error and can be corrected by manual offset calibration, which will be discussed next.

  Offset error

  Offset error is the difference between the actual range results and the actual target as shown in Figure 3.10 .

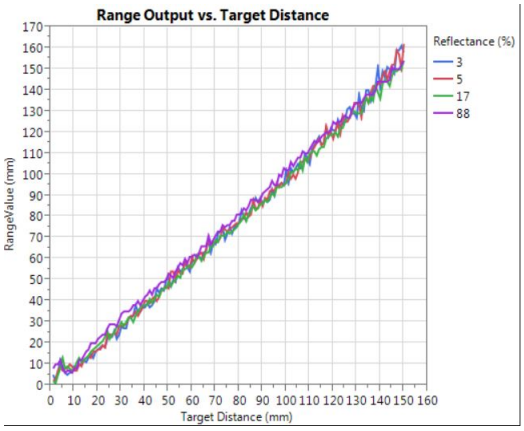


Figure 3.9: Range output vs. target distance [59]

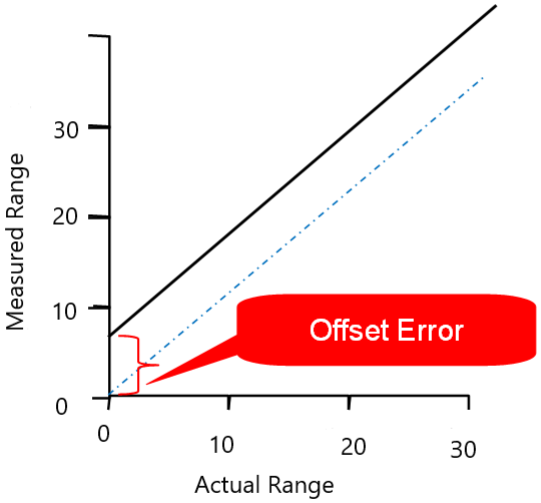


Figure 3.10: Range Offset [59]

Crosstalk error

Crosstalk can basically be defined as a signal bounced back to the sensor that was not reflected from the target. Optical crosstalk can come from a direct path, bouncing within a glass cover or window or from reflections in the cavity surrounding the optical components.

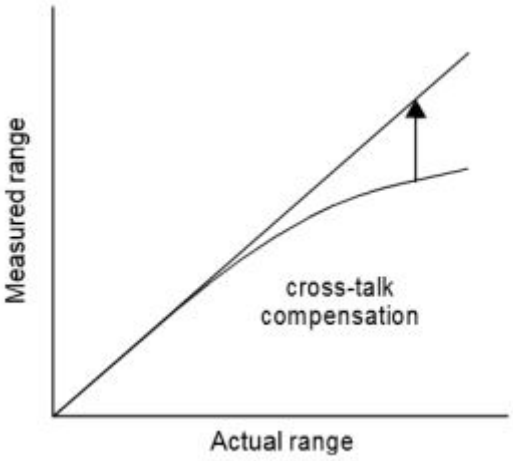


Figure 3.11: Crosstalk compensation error [59]

Offset and crosstalk calibration will be discussed in detail in the next chapter.

Max convergence time

The maximum convergence time represents the time to run measurements. If the max convergence time is reached, the ranging is aborted. In VL6180, the convergence time is 1-63ms.By default the max convergence time is set to 49ms during device power-up.

Inter-measurement period

The inter-measurement time is the time from the start of one ranging operation to the next one.

Wrap around filter (WAF)

In specific situations, when the target is a mirror or very reflective metal, the VL6180 gets a wrong distance(wrap around effect). The goal of the Wrap around filter is to recognize this wrap around effect and filter it by returning a non-valid distance.

Early convergence estimate (ECE)

Early convergence estimate (ECE) is a programmable feature designed to minimize power consumption when there is no target in the field of view (FOV). ECE works by calculating the rate of convergence 0.5 ms after the measurement has been started; if the return signal reported by the device is below the set ECE threshold, the measurement is aborted [59].

Figure 3.12 shows an example of the current consumption of the VL6180 with and without ECE feature enabled and for a range of inter-measurement periods. In this example, the max convergence time is 50 ms. The ECE ratio is set to 95% [59].

Inter measurement period (ms)	ECE = ON	ECE = OFF		
	No Target	No Target	Target @ 50mm	Target @ 100mm
2000	0.03	0.53	0.08	0.15
1500	0.04	0.72	0.11	0.2
1000	0.05	1.09	0.18	0.3
750	0.07	1.42	0.22	0.4
500	0.09	2.19	0.33	0.59
250	0.18	4.2	0.62	1.42
100	0.42	9.85	1.45	3.32
50	0.77	18.08	2.59	5.63

Figure 3.12: VL6180 current consumption versus ECE feature and inter-measurement period (in mA) [59]

Max detection range (Dmax)

Maximum detection range (Dmax) function is able to define the maximum distance up to which a 17% reflective target is detected with the current ambient light condition.

When the ambient light level increases,Dmax decreases, so a target may not be detected by the VL6180 because it is too far for a given ambient light condition. When no target is detected, no valid distance is reported.

- **VL53L3CX**

The VL53L3CX is another ToF from ST and embeds the company's third-generation FlightSense technology.It combines the benefits of a high-performance proximity sensor with ranging capability up to 3 m, whatever the target color and reflectance. In addition, the VL53L3CX has superior linearity that increases short-distance measurement accuracy[60].

The miniature reflowable package integrates a single photon avalanche diode (SPAD) array and physical infrared filters to achieve the best ranging performance in various ambient lighting conditions, with a wide range of cover glass windows[60].

The main features of the sensor are:

- Emitter: 940 nm invisible laser (VCSEL) and its analog driver
- Low-power
- Size: 4.4 x 2.4 x 1 mm
- Histogram based technology
- Up to 300 cm+ detection with full field of view (FoV=25°)
- Immune to cover glass crosstalk and fingerprint smudge at long distance
- Dynamic fingerprint smudge compensation
- Multi target detection and distance measurement

Detection cones and optical field of view

Extended ranging distance enables a (very) large target detection area to fit each application. Figure 3.13 presents the detection radius available for each of the ST's TOF sensors.

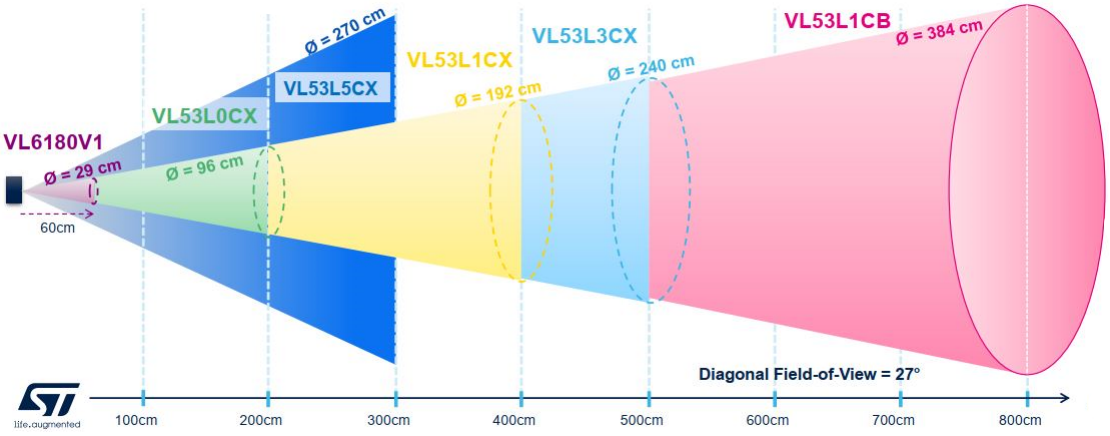


Figure 3.13: TOF sensors FOV

Timing budget

The timing budget is defined as the programmed time required by the sensor to perform and return ranging measurement data. Figure 3.14 illustrates the ranging sequence including the time budget and inter-measurement time.

Figure 3.14: VL53L3CX ranging sequence [61]

Distance modes

The user can determine the optimum distance depending on the application. The datasheet [60] provides benefits of each range.

Distance mode	Benefit
Short	Better ambient immunity
Medium(Default)	Lower power Consumption
Long	Maximum distance

Table 3.2: VL53L3CX distance modes

VL53L3CX calibration

The VL53L3CX requires the same calibration as VL6180 which will be discussed in section 4.1 .

Histogram

The histogram is based on 24 bins, where the bin is a “time window” outlining the number of photons received by the sensor during a specific time.

The histogram enables multi-object detection. However, to detect two objects, the separation between them must be at least 80 cm(each detected object is expressed using three bins).

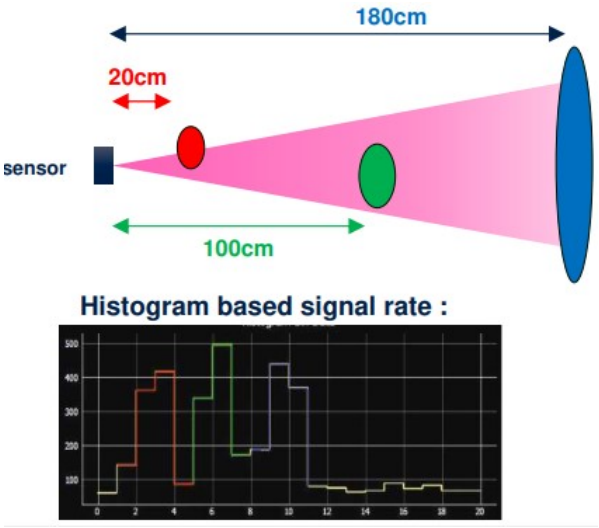
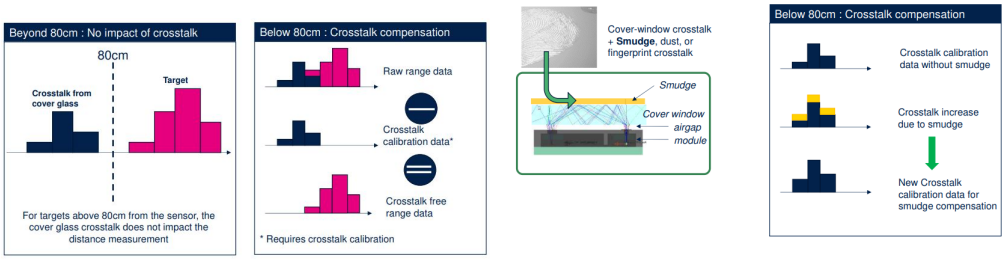


Figure 3.15: How multiple objects are represented in histogram using VL53L3CX [60]

Histogram enables accurate distance output whatever the smudge or crosstalk.Beyond 80cm,the crosstalk and smudge has no impact on the distance measurements. Therefore,crosstalk compensation is needed below 80cm .



(a) Crosstalk Management **(b)** Smudge detection

Figure 3.16: Smudge detection and crosstalk immunity

Chapter 4

Design and realization

This chapter describes the design and realization phases of this work.

The implementation process of this thesis can be divided into hardware and software. In terms of hardware, the main steps are related to the device calibration before measurements and and the creation of the prototype. Regarding the software, the main focus is on presenting the ranging implementation steps, mentioned in the previous chapter. Furthermore,the chapter illustrates dataset collection and algorithm used in gesture recognition.

4.1 TOF sensor calibration procedure

To guarantee the best performance, different calibration parameters are required. Mainly, offset calibration is necessary for all cases,but the calibration is desired only in case of cover glass. The crosstalk causes internal reflection, and the sensor will detect this as unwanted signals.

4.1.1 Calibrating the offset

The TOF sensor requires a unique part-to-part range offset correction. The factory-calibrated Non volatile memory (NVM) offset is used by default. Manual calibration is only required if the offset is incorrect, resulting in incorrect range measurements [59]. In order to achieve the offset calibration, the following procedure was performed.

- A white target was placed at a 50mm distance from the sensor.
- Ten range measurements were collected, and the mean of the range results was calculated.

- The following equation is applied to calculate the offset:

$$\text{Offset} = \text{Target Distance (50mm)} - \text{Average Range}$$

- Store the offset in the host memory to be used at each device boot.

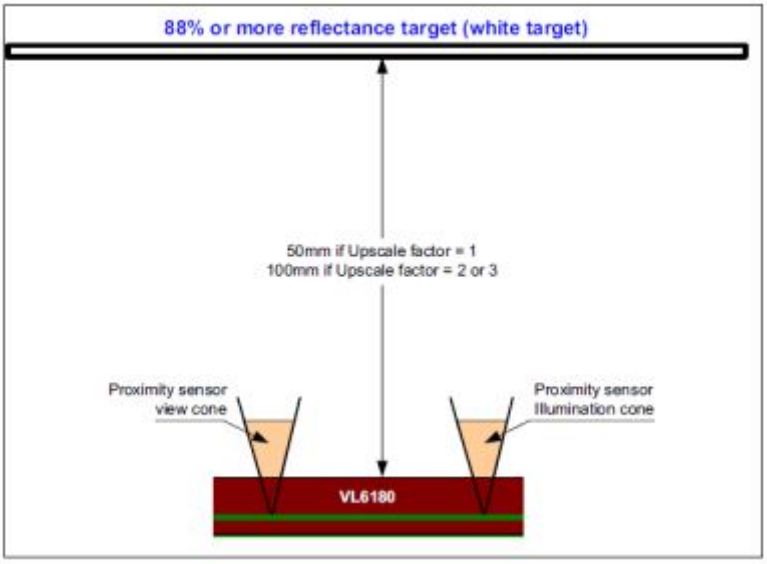


Figure 4.1: Offset calibration environment [59]

4.1.2 Calibrating the crosstalk compensation factor

The cover glass in front of the TOF sensor introduces crosstalk. The resulting stray light disrupts the range measurement. However, it can be corrected by applying crosstalk compensation. An experiment is required to determine a unique crosstalk compensation factor.

In the following, the procedure for calibrating crosstalk is described:

- The offset calibration had to be performed before the crosstalk calibration since incorrect offset calibration leads to inaccurate crosstalk calibration.
- A black target was placed at a 100mm away from the sensor.
- For reliable results measurement,ten measurement has been collected and mean of the range results and return signal rate were found.

- The following equation is applied to calculate the crosstalk:

$$\text{Crosstalk[Mcps]} = \text{Rtn Rate[Mcps]} \times \text{(1-Average range)}$$

- Store the value in the host memory to be used at each device boot.

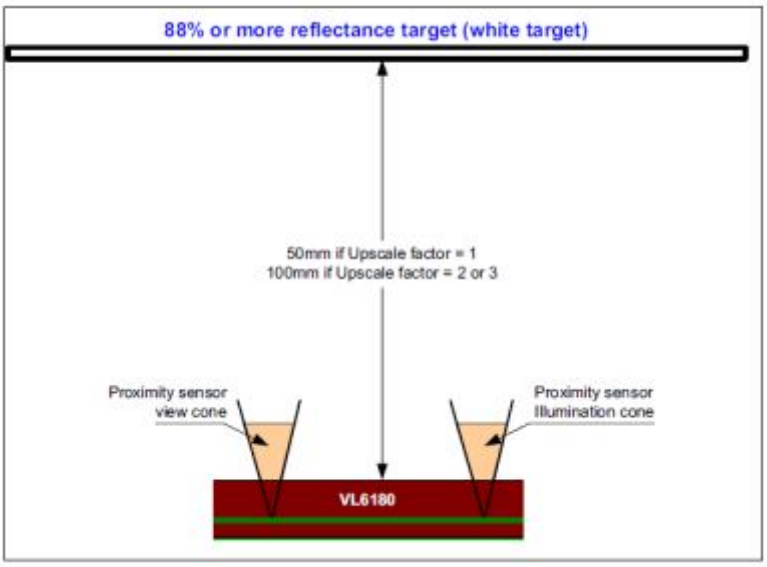


Figure 4.2: Crosstalk compensation environment [59]

4.2 VL6180

This section explains how multiple VL6180 sensors can be used on a board design while only using a single I2C interface to communicate with all the devices.
Each VL6180 device has both a reset pin GPIO0 and an interrupt pin GPIO1 , which can be used to enable this setup.
Process to initialize VL6180 devices is explained in Figure 4.3, where X represents sensor number [0,1,2].

- Take sensor number 0 out of rest by bringing reset pin high.
- Change the address of device 0,the default address value 0x52,with the following equation:

$$\text{Final I2C Address} = 0x52 + 2\ (X+1)$$

- To ensure it is working correctly,read device model ID with its new I2C address.
- If device is communicating on its new I2C address then repeat the steps 1-3 for each sensor.

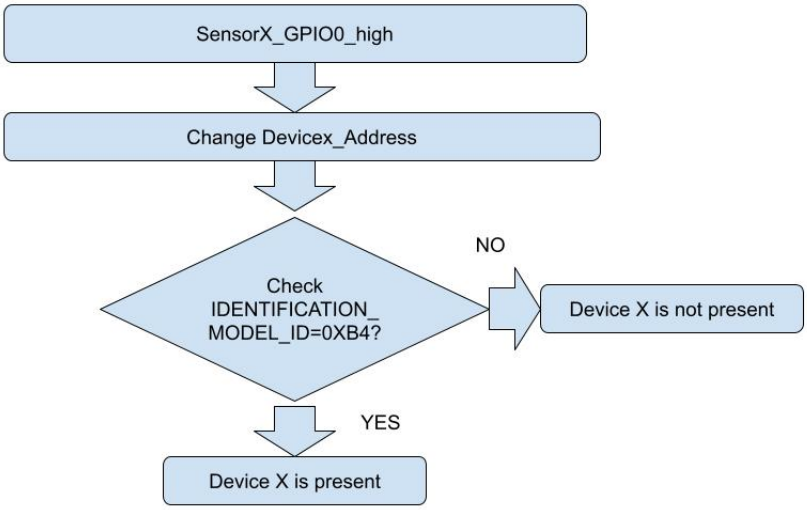


Figure 4.3: Initialization of VL6180 devices

The VL6180 runs in interrupt mode, which means that the CPU is idle unless the interrupt pin is high (data ready).

The multi-sensor range code workflow is as follows.

- The initialization is demonstrated in Figure 4.3.
- All pending interrupts are cleared, and the calibration parameters are set.
- The sensors start measuring the target distance.
- When the measurements are ready, the CPU is notified by setting the interrupt pin to low.
- The data is transmitted via UART as packets, each data packet contains 2 start bytes,a payload of 3 bytes, and 2 stop bytes.
- The interrupt is cleared and the CPU is idle until a new interrupt arrives.

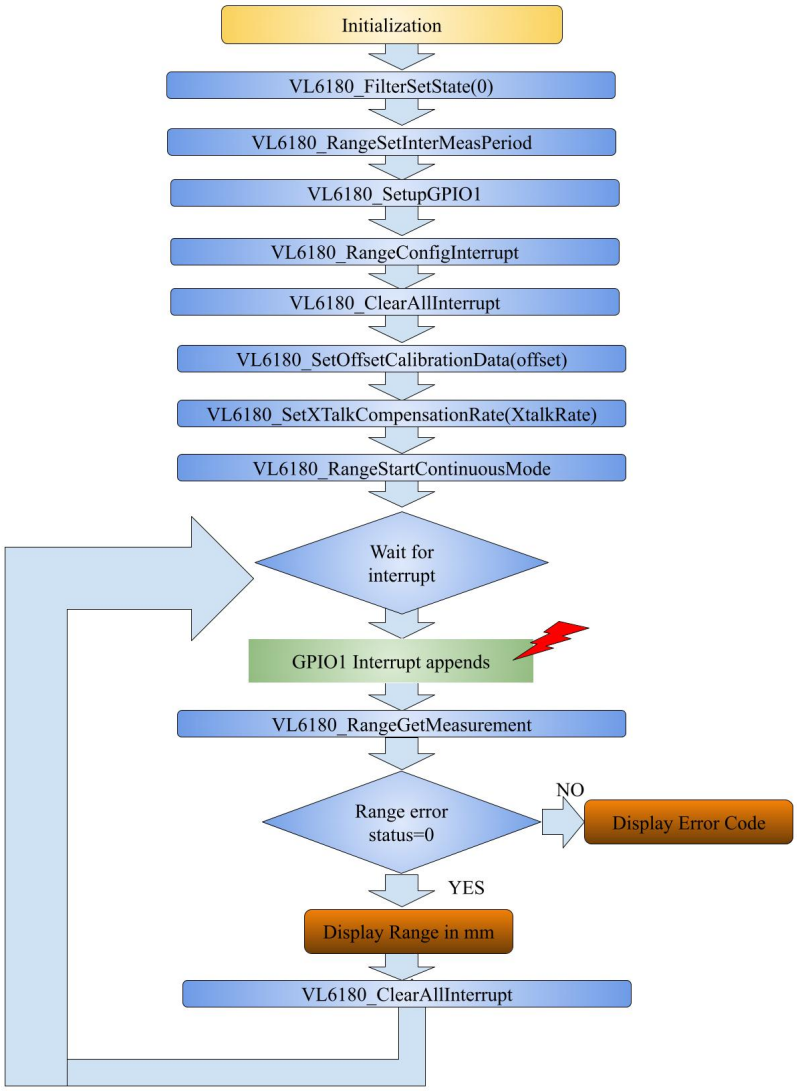


Figure 4.4: Prototype ranging measurement flow

4.3 VL53L3CX

The idea of selecting two types of TOF sensors is to test the performance and decide which sensor is more suitable for the application of interest.

4.3.1 VL53L3CX ranging flow

VL53L3CX system is comprised of the VL53L3CX module and a bare driver that contains functions accessible to the host.

The VL53L3CX can operate using either polling or interrupt but since the interrupt allows faster data transfer it will be used. Moreover, the mechanism utilized in the measurement is illustrated in Figure 4.5 is referred to as the handshake mechanism (when data is ready, the sensor raises an interrupt, the host acquires the data and enables the next interrupt by clearing the interrupt and enabling the next ranging).

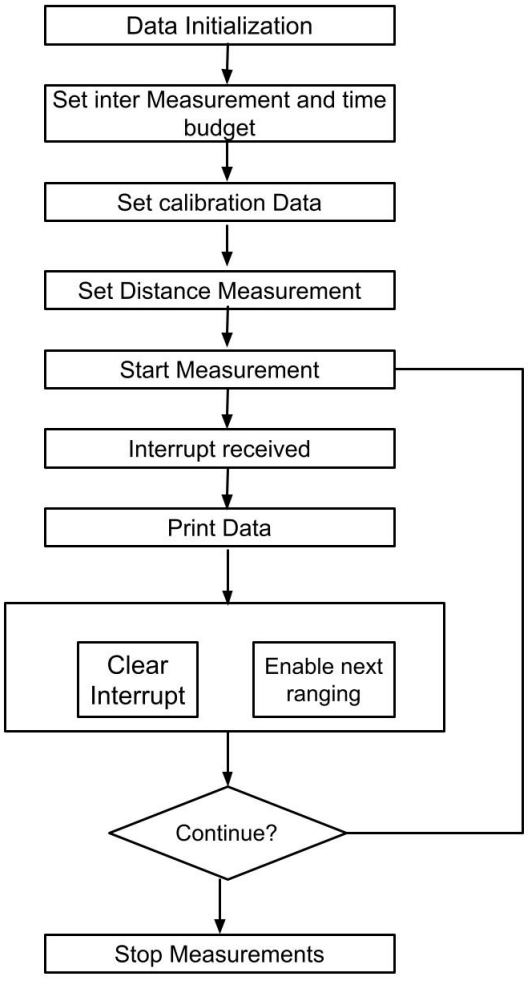


Figure 4.5: VL53L3CX ranging flow

4.3.2 VL53L3CX multi sensor ranging

The Expansion Board allows the user to program three of the VL53L3CX sensors, and the microcontroller controls the sensors through the I2C bus. Since the application is using multiple sensors using interrupt mode , the datasheet [60] provides solder drop configuration as follows: "The VL53L3CX interrupt of the left and right breakout boards, $GPIO1L$ and $GPIO1R$, can be activated by fitting U10 and U15 respectively."

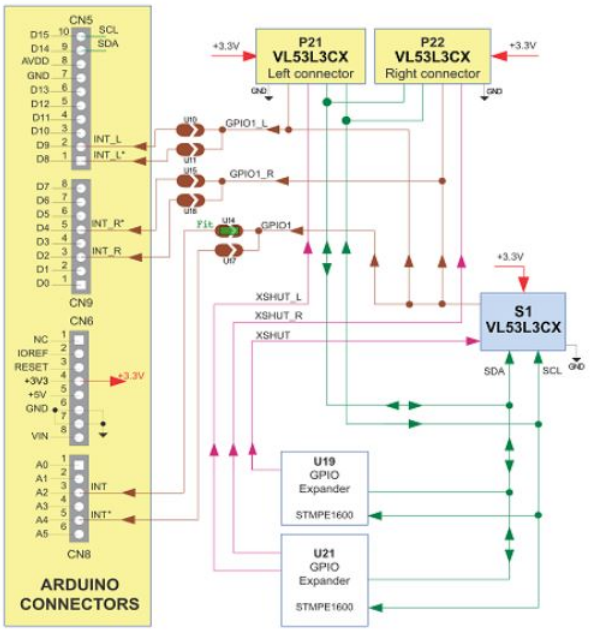


Figure 4.6: X-NUCLEO-53L3A2 Expansion Board connector layout [60]

4.4 Prototype design

This section aims to describe the sensor prototype design, which can be utilized in various applications. While designing the prototype, minimizing the number of sensors was taken into account. An equally important consideration was to maintain an excellent recognition accuracy of multiple gestures.

The VL6180 is chosen as the prototype TOF sensor to carry out the distance measurement. The choice is mainly due to VL53L3CX having a longer default range than its counterpart. Therefore, most of its measurement range would be

not be utilized. While on the other hand, the VL6180 range matches the desired operating range.

In addition to the steps of the selection criteria explained above, two essential factors had to be taken into consideration:

- the FOVs of the sensors shall not overlap to avoid crosstalk.
- there should not be any dead zone between the sensors where motion is not detectable.

The horizontal array of three VL6180 sensors stand out as a suitable prototype Given that FOV = 25°as shown in Figure 4.7 ,it has been found that best distance between the sensors is 4.4cm.

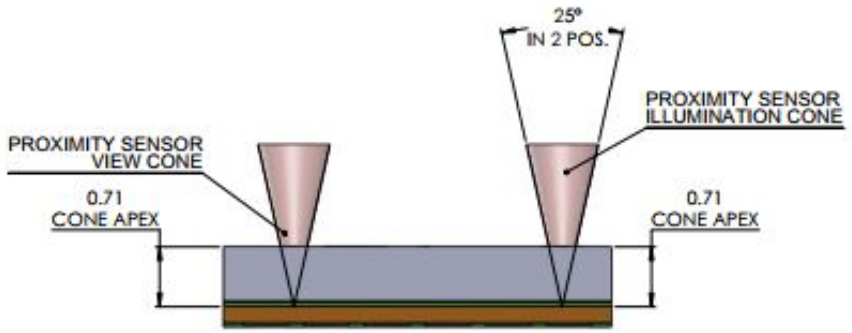


Figure 4.7: VL6180 outline drawing [62]

The proposed prototype is shown Figure 4.8,where the black boxs represent the sensors,red cones represent the FOV of the sensors. The brown box contains the development board STM32F401RE along with the VL6180A1(Expansion Board) that is connected to the PC using USB mini(shown in Figure 4.9)

4.5 Data collection

Each gesture execution time varies according to its complexity and speed of execution. In other words, each gesture takes a specific time or equivalent number of measurement samples. To find a sufficient number of samples that capture all gestures, including their variation, the measurement time of each sensor is investigated. Additionally, the chosen gestures are introduced.

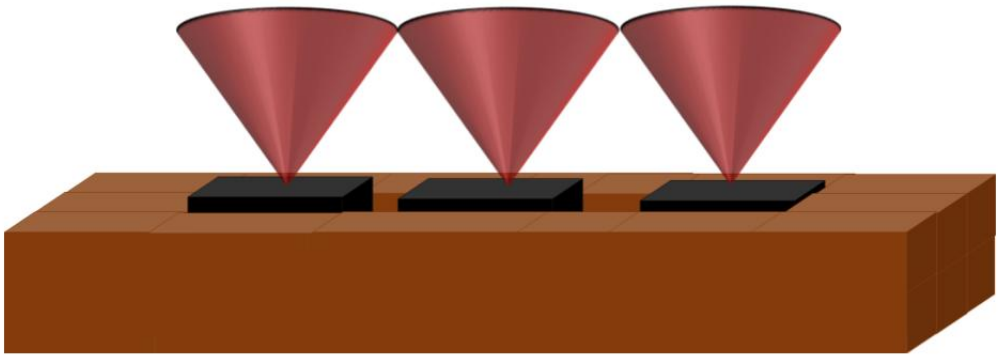


Figure 4.8: Proposed TOF prototype

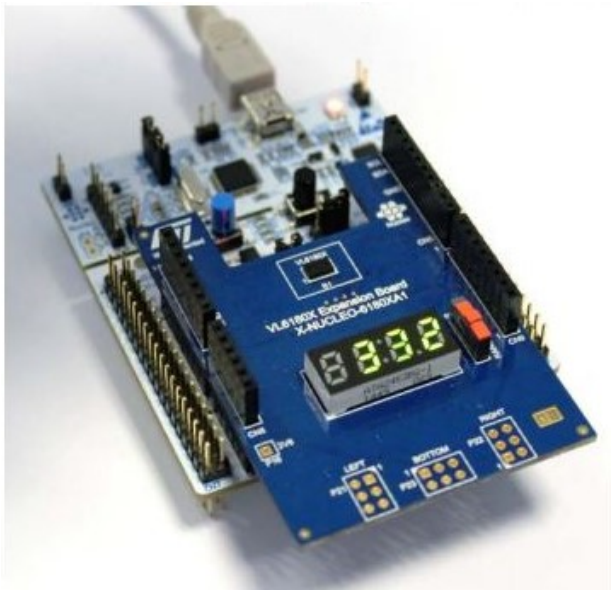


Figure 4.9: VL6180A1 Expansion Board plugged on a STM32 Nucleo board

4.5.1 Measurement Timing

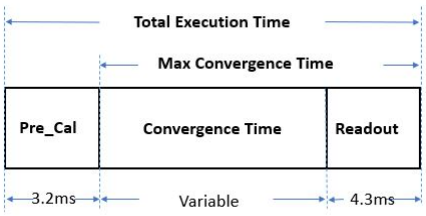
Choosing longer measurement ranges may lead to capturing undesired objects. On the contrary, selecting shorter spans may restrict the user working space. Therefore, the range used in VL618 is 20cm.

Figure 4.10a, gives a breakdown of total execution time for a single range

measurement:

- The pre-calibration phase is fixed (3.2 ms).
- The range convergence time is variable and depends on target distance and reflectance as shown in Figure 4.10b .
- The recommended readout averaging period is 4.3 ms. Readout averaging helps to reduce measurement noise.

  Effective max convergence time depends on the actual convergence time plus readout averaging sample period setting.



(a) Total range execution time [62]

Range (mm)	Target reflectance			
	3%	5%	17%	88%
10	0.43	0.33	0.18	0.18
20	0.94	0.73	0.28	0.18
30	1.89	1.40	0.51	0.18
40	3.07	2.25	0.81	0.18
50	4.35	3.24	1.18	0.24
60	5.70	4.22	1.60	0.32
70	7.07	5.35	2.07	0.49
80	8.41	6.45	2.58	0.50
90	9.58	7.56	3.14	0.61
100	10.73	8.65	3.69	0.73

(b) Range convergence time (ms) [62]

Figure 4.10: Execution and convergence time

The paper [63] indicates that the typical human hand reflectance is higher than 50%. Moreover, Table 4.10b does not specify the convergence time for human hands nor distances above 100mm. However, it shows the convergence time for objects with 17% and 88% reflectance. Given all the previous information, it is possible to interpolate the data using a second-order polynomial to estimate the convergence time for a human hand up to 200mm as demonstrated in Figure 4.11. The result of the interpolation is shows in Table 4.1.

Examining the worst-case scenario where the hand is 200 mm from the sensor, in this case, calculations are done as follows:

$$\text{Max convergence time} = 7.92 + 4.3 = 11.2\text{ms}$$

$$\text{Total execution time} = 3.2 + \text{Max convergence time} = 15.42\text{ms}$$

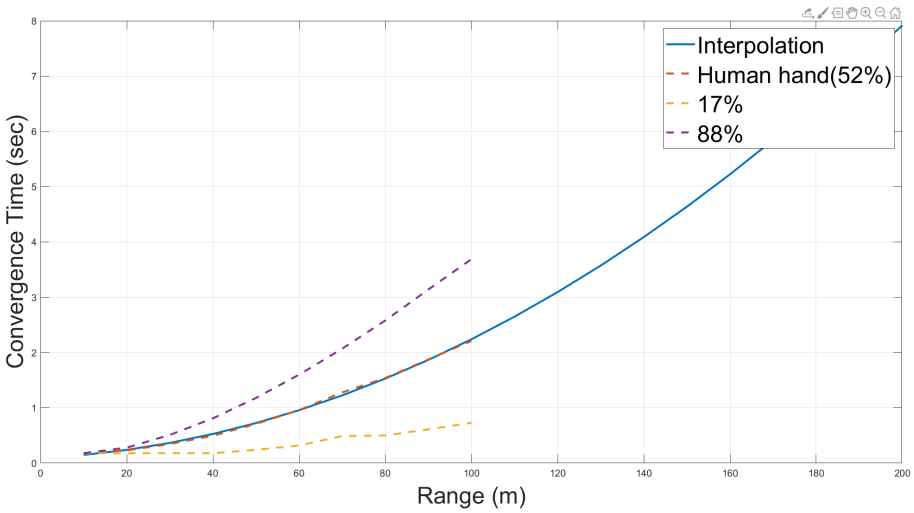


Figure 4.11: Interpolation for range convergence time

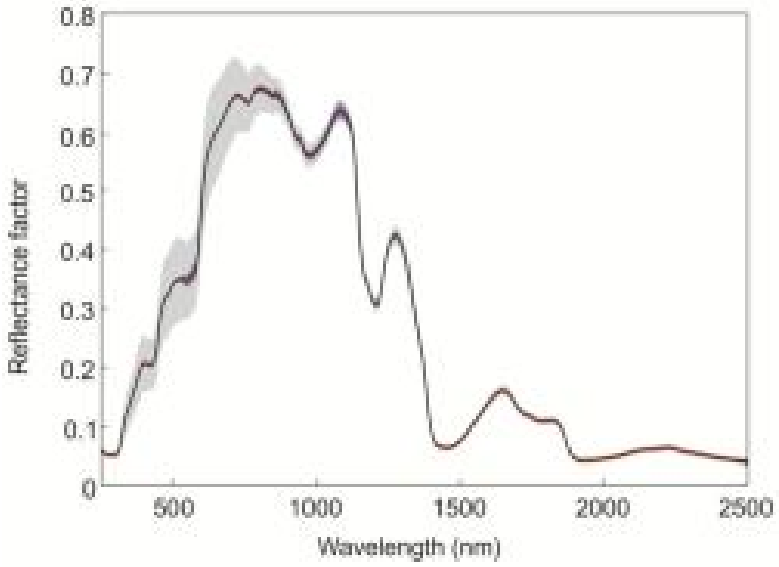


Figure 4.12: Reflectance of human-body [63]

4.5.2 Gestures

The chosen gestures included 4 directional swipes and 2 rotational movements as in Table 4.2.

These gestures are collected by 5 users, who will be collecting the training dataset.

Range(mm)	Target reflectance 50%
10	0.15
20	0.24
30	0.37
40	0.53
50	0.73
60	0.96
70	1.23
80	1.53
90	1.87
100	2.24
110	2.65
120	3.09
130	3.57
140	4.09
150	4.64
160	5.22
170	5.85
180	6.50
190	7.19
200	7.92

Table 4.1: Range convergence for 50% reflectance

Moreover, to understand how fast each user can make each gesture, an analysis is first carried out on experimentally recorded gestures to determine the maximum number of data samples needed to capture the longest gesture.

A Python script was executed to register the time instance and the number of samples when the hand enters and leaves the sensor FOV.

The gestures that took the longest time are the rotational(CW and CCW). The number of samples needed to capture all of the variety of rotational gestures was 100 samples.

STM32CubeMonitor is a tool that allows real-time sampling and visualization of user variables while the application is running. We define the following flow as in Figure 4.13 to monitor and visualize live monitoring of the three TOF sensors using STM32CubeMonitor. Figures 4.14 shows a plot of the three TOF sensors distance measurement Vs time for each gesture.

Gesture	Description
Swipe left	A swipe from the right of the board, over the sensors and then to the left and out of range of the sensors.
Swipe right	A swipe from the left of the board, over the sensor and then to the right and out of range of the sensor.
swipe up	A swipe from the bottom of the board, over the sensors and then to the top and out of range of the sensors
Swipe down	A swipe from the top of the board, over the sensor and then to the bottom and out of range of the sensor
Clockwise	The hand or just a finger is moved over the right sensor in the clockwise direction
Counterclockwise	The hand or just a finger is moved over the right sensor in the counterclockwise direction
Unknown	The hand moves rapidly,or without specific direction

Table 4.2: Description of proposed gestures

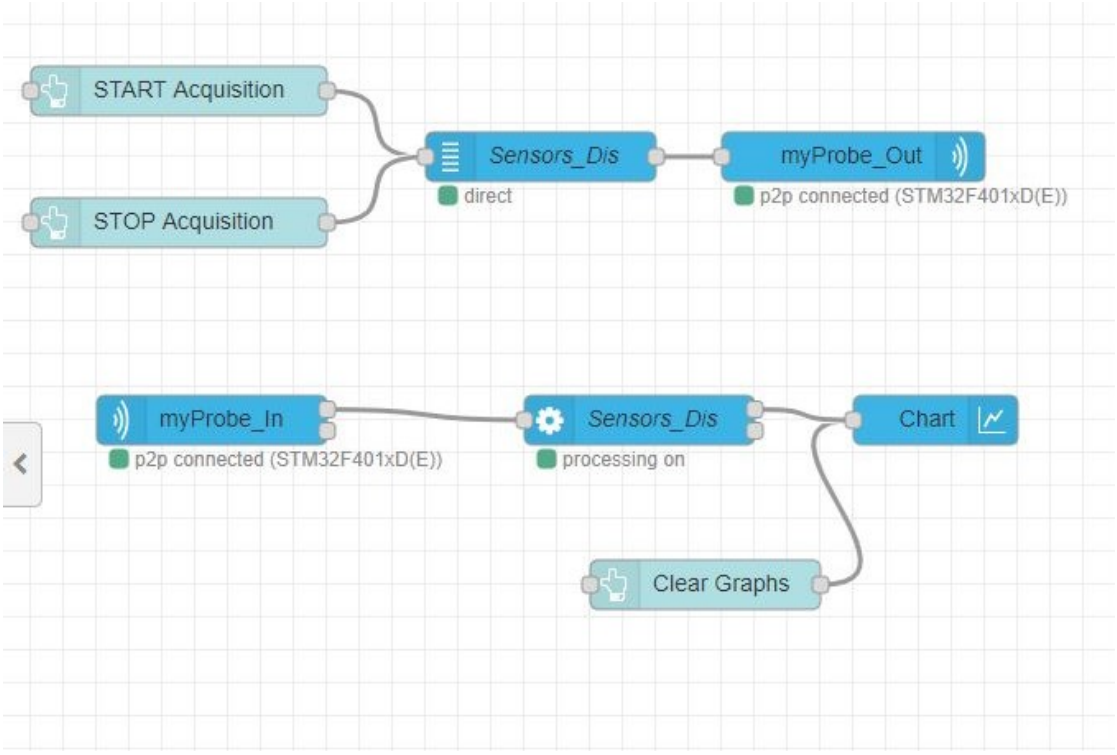
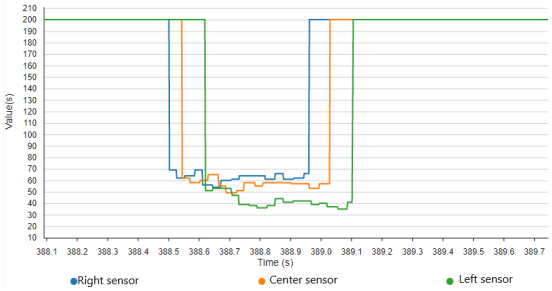
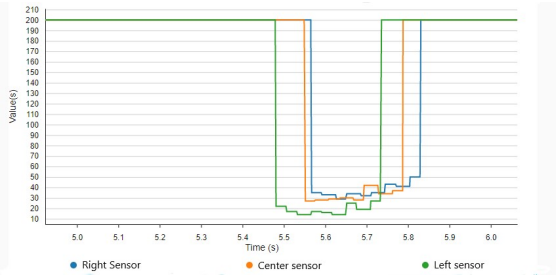


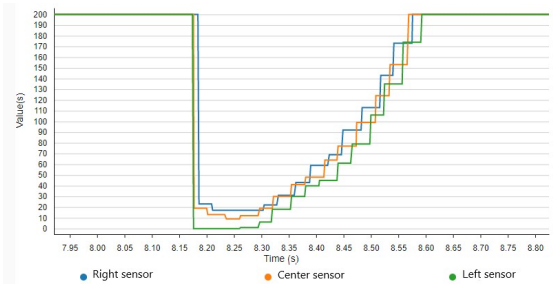
Figure 4.13: STM32 monitor



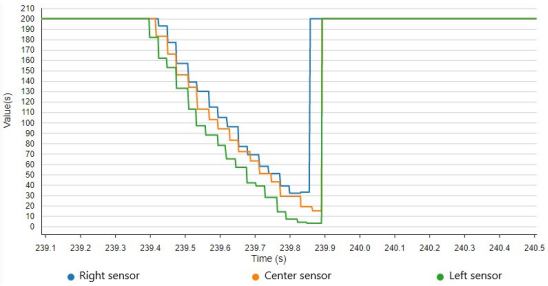
(a) Swipe left



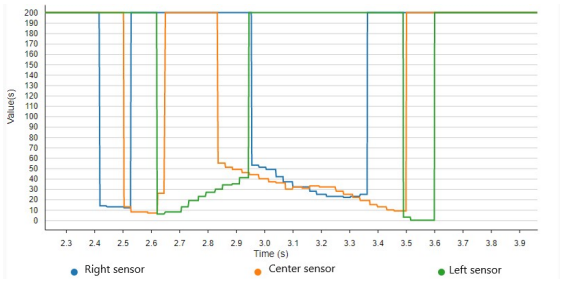
(b) Swipe right



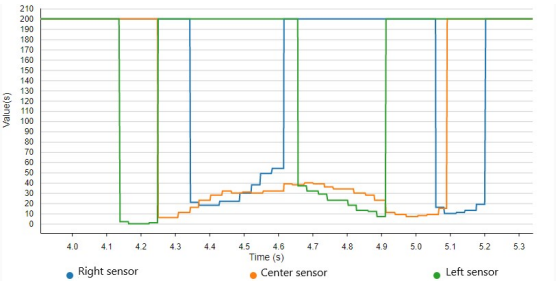
(c) Swipe up



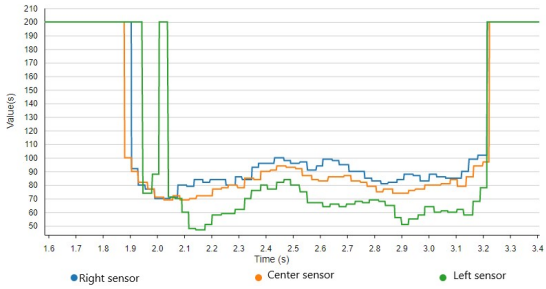
(d) Swipe down



(e) Clockwise gesture



(f) Counter clockwise gesture



(g) None

Figure 4.14: Gestures characteristics in time

4.5.3 Handling missing data

Before the user performs any gesture, no data are collected. However, when the hand passes over one of the TOF sensors, 100 consecutive measurement samples are recorded. The recorded set of data can either be used for training, validation or real-time recognition. If the motion ends by exiting the FOV of all sensors before completing 100 samples, then the remaining data are filled with the maximum value of the sensor. The maximum value is selected in this case because the default measurement of the TOF when no object is present in its range is the complete range. Figure 4.15 demonstrate using a flow chart the mechanism for handling missing data.

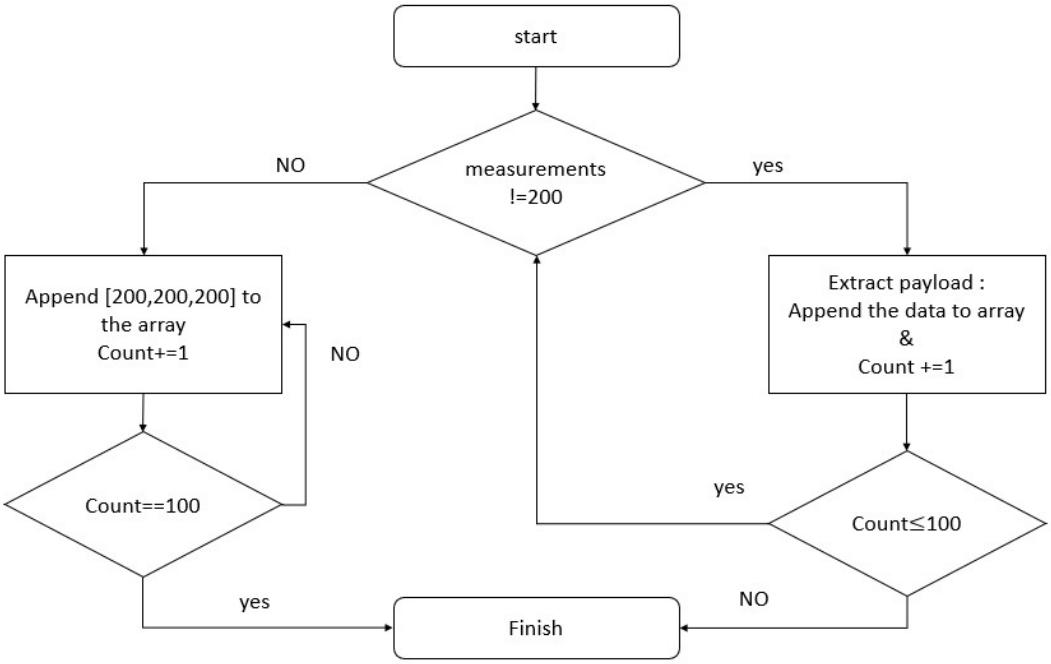


Figure 4.15: Flow chart data collection

Finally, the training dataset consists of 17,500 balanced samples collected by 5 users in different light conditions. Each TOF training set contains 100 samples. Additionally, each sample is composed of 3 TOF measurements from the left, right, and center sensors. The 300 TOF measurements are used as input features to the DNN model. Furthermore, each training, validation, or testing set has a corresponding label indicating which gesture the input set belongs.

4.6 Flick Hat 3D tracking and gesture hat

The core hardware that is explored in this section is the MGC3130 Flick Hat.As presented in Figure 4.16, the Flick Hat gesture recognition is initiated by a hand or finger movement in the operational sensor range. Subsequently, the MGC3130 translates the change of electric field due to hand movement to 3-dimensional position data. Afterward, the set of position data is forwarded to the DNN model to recognize the executed gesture.

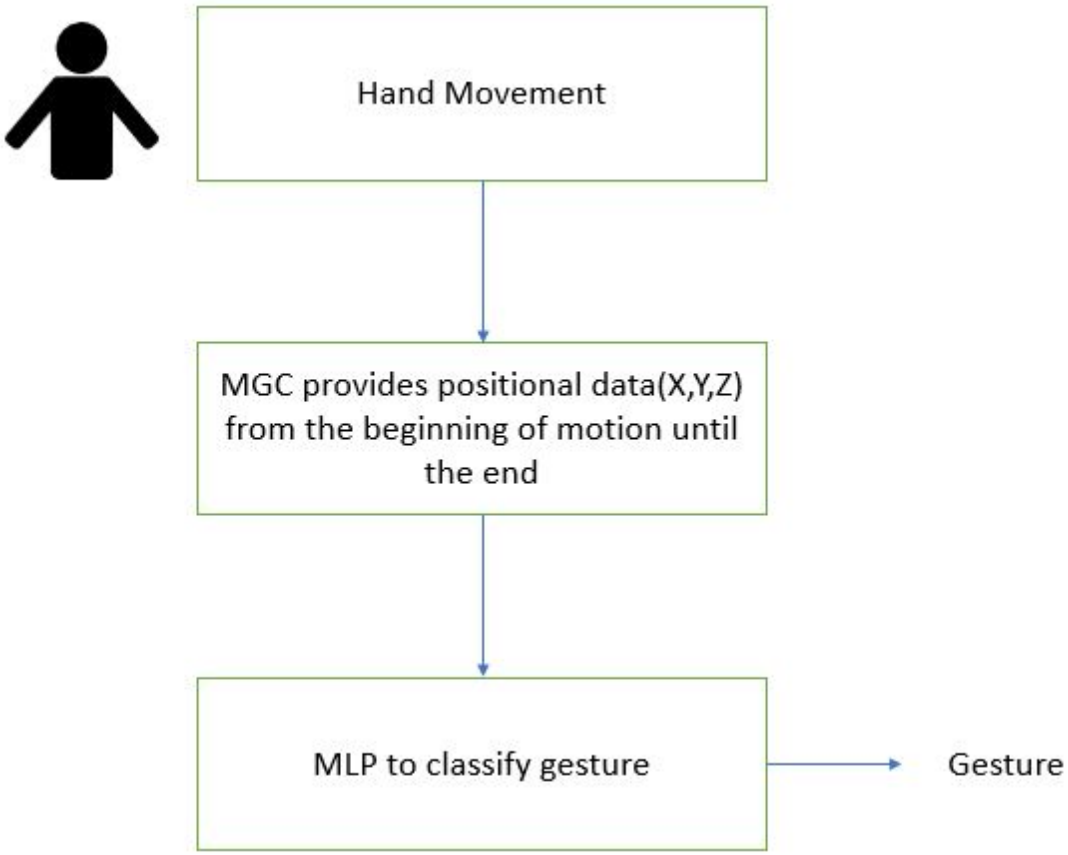


Figure 4.16: Flick Hat gesture recognition flow

The MGC3130 microchip does not provide the measurement of the raw electrodes to the user directly. However, these measurements are converted via the library to meaningful information. Therefore the details of electric field computation are hidden, and the chip is regarded as a black box.

The first step in the project is to access the 3-dimensional data from the MGC3130. Furthermore, the Colibri suite in the chip acts as a preprocessing module. It has three main functionalities, listed below.

- Approach detection

  It is a power-saving feature of Colibri suite; it sends MGC3130 to sleep mode and scans the sensing area periodically to detect the presence of a hand.

- Position tracking

  It provides the three-dimensional hand position over time. The absolute position is provided regarding the origin of the Cartesian coordinate (x,y,z). The origin of the coordinate system is defined as the lower-left corner(south-west) at the surface. Positional tracking data is continuously obtained up to 200 positions /sec.

- Gesture recognition

  The suite detects and classifies the hand movements using a stochastic classification based on Hidden Markov Model(HMM). The set of gestures includes swipes in the four directions, namely left, right, up, down, in addition to round-shaped hand movement.

The last functionality of the Colibri suite will not be used since the project objective is to develop an MLP for gesture recognition.

Aurea GUI

Microchip also provides Aurea GUI, which supports visualizing the hand's position in space and its history in a 3-dimensional plot as in Figure 4.17. The GUI has no option for customized scripts. The GUI user guide [64] provides detailed information about its functionalities.

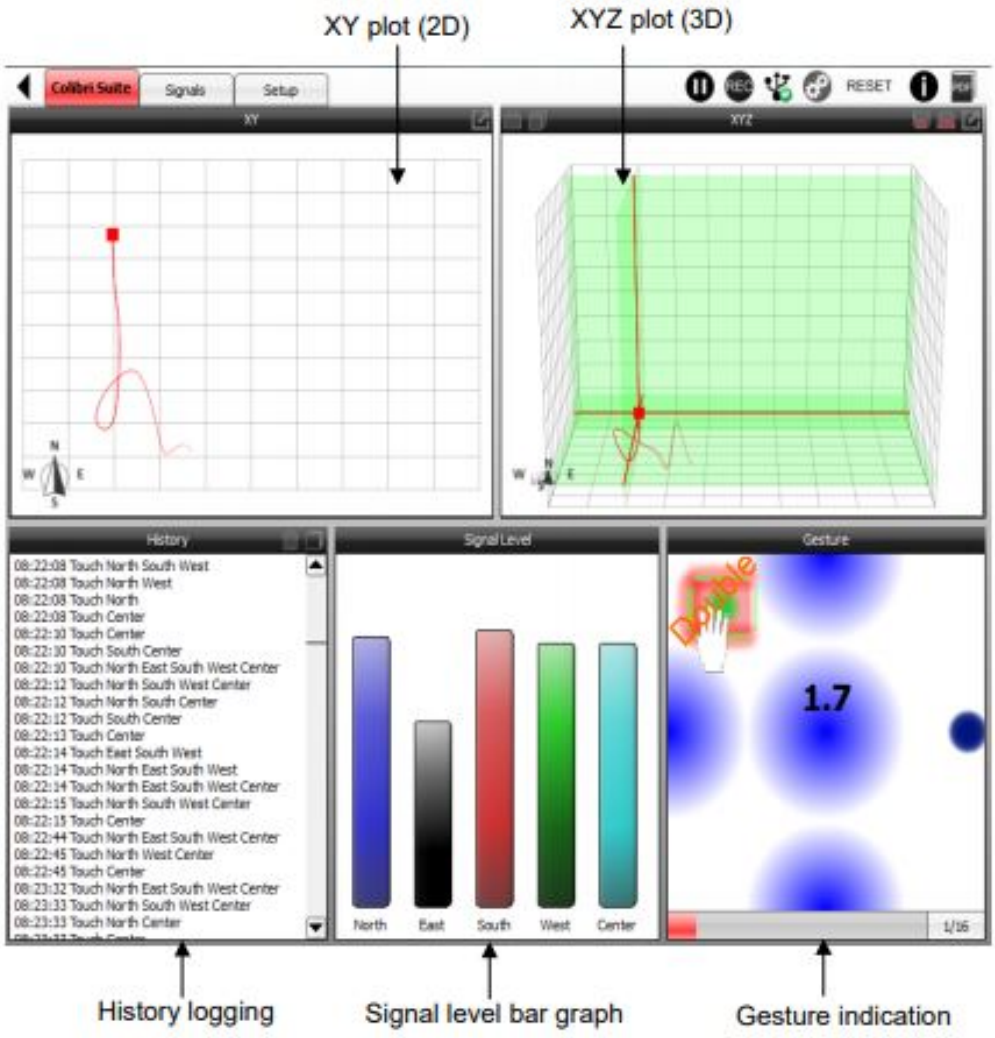


Figure 4.17: Aurea GUI [64]

Data preparation and collection

To collect the coordinates x,y,z of the hand motion, the flick hat library flicklib.py is modified such that each gesture example has a window of 200 positional data to record, as in Equation 4.1:

$$\begin{bmatrix} X_{1,1}, Y_{1,1}, Z_{1,1} \dots X_{1,200}, Y_{1,200}, Z_{1,200} \\ \vdots \quad \vdots \quad \vdots \quad \dots \quad \vdots \quad \vdots \quad \vdots \\ \vdots \quad \vdots \quad \vdots \quad \dots \quad \vdots \quad \vdots \quad \vdots \\ X_{N,1}, Y_{N,1}, Z_{N,1} \dots X_{N,200}, Y_{N,200}, Z_{N,200} \end{bmatrix} \tag{4.1}$$

In the case of hand, motion is fast, and the 200 data are not completed; the remaining data is filled with the last Cartesian coordinates provided by the sensor.

The datasets for the sensor training includes 7,000 examples collected by five users and balanced between the seven classes. One-hot encoding is used for the labels as in TOF sensors prototype.

4.7 APDS 9960

As explained in Chapter 3, APDS9960 utilizes four directional photodiodes to measure the reflected IR energy. The proximity sensor ADC converts the reflected IR intensity to a proximity value as illustrated in Figure 4.18 .This value ranges from 255 to 0 for nearest and furthest targets, respectively.

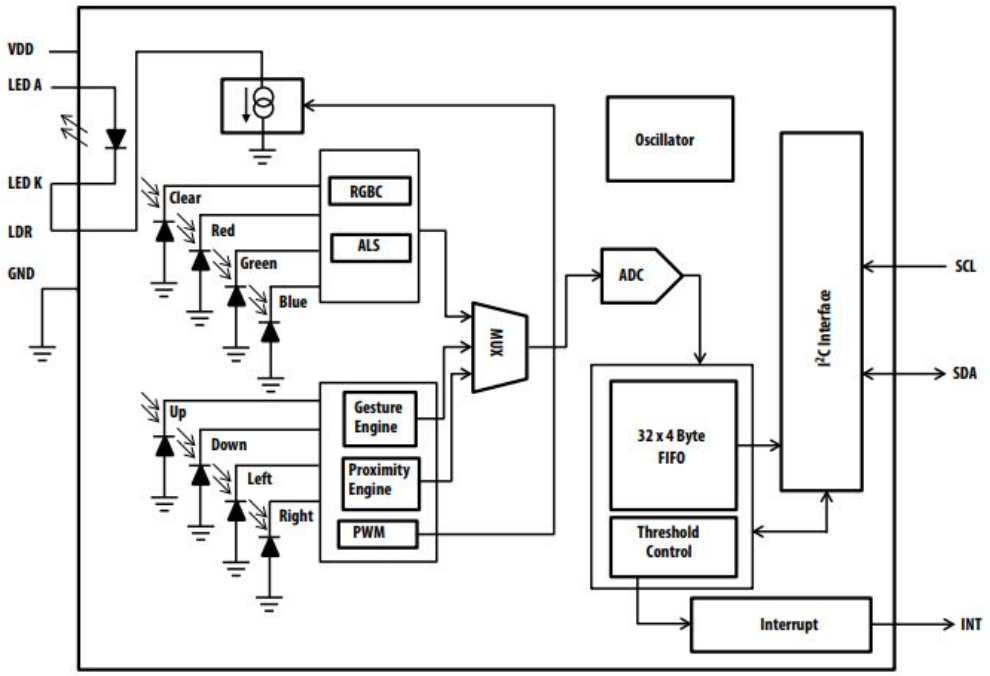


Figure 4.18: APDS9960 functional diagram

From datasheet [54],the required time to get one measurement from the four photodiodes is typically 5ms. Moreover, using experimental data, the longest gesture takes no longer than one second to be performed on the APDS9960. Therefore, the number of required measurement samples is 200 to capture all possible gestures.

The training dataset consists of 7,000 balanced samples collected by five users. The total number of features are 800 (200 measurement × 4 photodiodes data) as input features to the DNN model. Furthermore, each training, validation, or testing set has a corresponding label indicating which gesture the input set belongs.

4.8 Deep neural netowrk

The DNN that has been implemented within the frame of this thesis used Keras, and the training is done in Google Colab.

Keras is a powerful and easy-to-use free, open-source Python library for developing and evaluating deep learning models [47].

Google Colab, is a free online cloud-based Jupyter notebook environment that allows us to train our machine learning and deep learning models on CPUs, GPUs, and TPUs [65].

4.8.1 Data pre-processing

First step in any machine learning project is to load the datasets .The datasets are saved in CSV files with delimiter comma (“,”) character. The datasets are loaded using numpy.loadtxt() function.Furthermore examples are normalized such that each value falls between 0 and 1. The normalization of each sensor is done with the prior knowledge of the specified maximum range of each sensor type as in Equation 4.2. The pre-processing code is shown in Listing 4.1.

$$D_{Normalized} = \frac{D_{Raw}}{R_{max}} \tag{4.2}$$

Where $D_{Normalized}$ is the normalized data ,D_{Raw} is the raw data and R_{max} is the maximum range.

Listing 4.1: Data loading and pre-processing

```
1  train_samples = loadtxt('datasets.csv', delimiter=',', usecols=range(300))
2  scaled_train_samples=train_samples/200
3  train_labels = loadtxt('data_Labels.csv', delimiter=',', usecols=range(7))
4  test_datasets = loadtxt('test_datasets.csv', delimiter=',', usecols=range(300))
5  test_datasets=test_datasets/200
6  test_Labels = loadtxt('test_Labels.csv', delimiter=',', usecols=range(7))
```

4.8.2 Model architecture

In Keras, it is possible to create a model in two different ways: Sequential pattern and Functional API. Usage of Sequential pattern is pretty straightforward, as the model is constructed by adding layers sequentially. It is enough to declare the input shape in the first layer and the parameters for each layer. On the other hand, Functional API is used to create a more elaborated model that can have an inception module or multiple inputs/outputs.

The devised model structure utilizes a sequential pattern with layer setting Dense, which is the most common type of layer used on MLP models. Furthermore, dropout is applied to the model, placing a fraction of inputs to zero to reduce overfitting.

The number of layers in the model and the model size (number of nodes in the model) represent the parameters. The most reliable way to configure these parameters for each specific forecasting problem is via systematic experimentation [39].

Listing 4.2 shows a DNN keras model with 3 hidden layers:

Listing 4.2: keras DNN model with 3 hidden layers

```
model = Sequential()
model.add(Dense(35, input_shape = (scaled_train_samples).shape[1:],activation='relu'))
model.add(Dropout(0.2))
model.add(Dense(18 ,activation='relu'))
model.add(Dropout(0.2))
model.add(Dense(7, activation='softmax'))
model.summary()
```

4.8.3 Hyperparameter tuning

The problem of interest is a multi-classification problem. One-hot encoding is used for the labels while the loss function is chosen as $CategoricalCrosstropy$.

Furthermore, the activation functions used in hidden layers is Relu because it overcomes the vanishing gradient problem, allowing models to learn faster and perform better. The output layer is softmax, which is designed for multi-class classification tasks.

Conceptually, hyperparameter tuning is an optimization task, much like model training. However, these two tasks are considerably different in practice. When

training a model, the quality of a proposed set of parameters can be written as a mathematical formula (usually called the loss function). However, when tuning hyperparameters, it is impossible to write the hyperparameters quality in a closed-form formula because it depends on the outcome of a black box (the model training process) [66].

Therefore hyperparameter tuning is considered problematic. Until a few years ago, the only feasible methods were grid search and random search. In the last few years, there's been a growing interest in auto-tuning. Several research groups have worked on the problem, published papers, and released new tools.

In the scope of our model, Keras Tuner has been utilized to limit the scope of the investigation of optimal tuning parameters of the MLP.

Google research has developed the Keras Tuner Toolkit, a user-friendly platform for the automated search for optimal hyperparameter combinations. Keras Tuner offers the main hyperparameter tuning methods: random search, Hyperband, and Bayesian optimization [67].

To start tuning the target model, a hyperparameter space containing the minimum and maximum value of each parameter must e defined as shown in Listing 4.3. Using the defined hyperparameter space Keras Tuner then aims to find the hyperparameters that maximize an objective. In Listing 4.3 the objective the maximize the accuracy. Moreover, The tuned hyperparameters are number of nodes in each layer, number of layers and the learning rate.

Listing 4.3: Keras Tuner code

```
def build_model(hp):
  model = keras.models.Sequential()

  model.add(layers.Dense(hp.Int('input_units',
                                min_value=32,
                                max_value=256,
                                step=32), input_shape = (scaled_train_samples).shape[1:]) )

  model.add(Activation("relu"))
  for i in range(hp.Int('n_layers', 1, 4)):  # adding variation of layers.

              model.add(Dense(hp.Int('units_'+str(i),
                                min_value=32,
                                max_value=512,
                                step=32) ) )
              model.add(Activation("relu"))

```

```
    model.add(Dense(7))
    model.add(Activation("softmax"))
    model.compile(
        optimizer=keras.optimizers.Adam(
            hp.Choice('learning_rate',[1e-2,1e-3,1e-4])),
            loss="categorical_crossentropy",
            metrics=["accuracy"])

    return model


tuner = RandomSearch(
    build_model,
    objective='val_accuracy',
    max_trials=10,  # how many model variations to test?
    executions_per_trial=4,  # how many trials per variation? (same model could perform differently)
    directory='project')
    # project_name='Tuning Parameters Results')
tuner.search(x=x_train,
                y=y_train,
                verbose=0,
                epochs=300,
                batch_size=32,
                validation_split=0.25
                )

#summary
tuner.results_summary()
```

4.8.4 Experimentation and evaluation

In this section ,we will describe the experimental settings and evaluate the performance of our proposed framework.

In order to test the performance of the 3 different DNN models of the capacitive, IR, and TOF sensors, additional datasets were created. These datasets were kept separate from that used for training.

Performance evaluation

The following evaluation metrics are taken into account.

Recall

A metric that measures the probability that a test will indicate a true positive case among the positive cases of a system. This metric is also called the true positive rate or probability of detection, since measures the proportion of positives that are correctly identified.

$$Recall = \frac{TruePositive}{TruePositive + FalseNegative} = \frac{TP}{TP + FN}$$

Specificity

It is a metric that measures the probability that a test will indicate a true negative case among the negative cases of a system. For this reason, the specificity is also called the true negative rate.

$$Specificity = \frac{TrueNegative}{TrueNegative + FalsePositive} = \frac{TN}{TN + FP}$$

Precision

Is the number of correct results over the number of all returned results.

$$Precision = \frac{TP}{(TP + FP)}$$

F1 Score

it is calculated from precision and recall given by the equation:

$$F1\ Score = \frac{2 * Precision * Recall}{(Precision + Recall)}$$

Accuracy

It is a metric to measure the probability that a tested set indicate as a true negative and a true positive among the cases in the study.

$$Accuracy = \frac{TP + TN}{(TP + FP + TN_F P)}$$

Confusion matrix

In the area of machine learning and particularly considering the problem of classification, it is common to use tools like the confusion matrix, also known as error matrix. It is table that is often used to illustrate the performance of a classification model on test data for which the true labels are known.

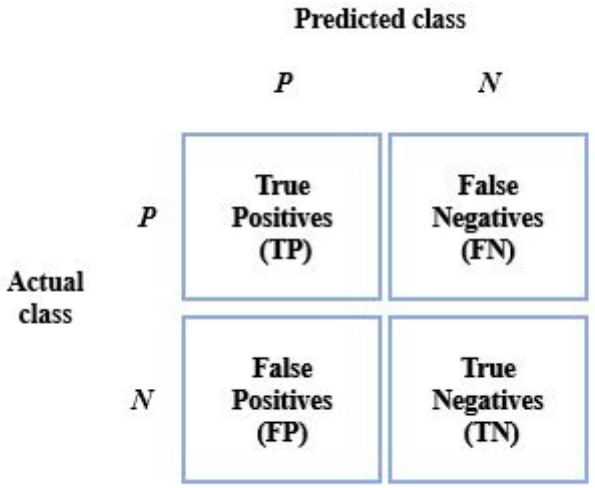


Figure 4.19: Confusion matrix example

The diagonal elements represent the number of points for which the predicted label is equal to the true label, while off-diagonal elements are those that are mislabeled by the classifier. The higher the diagonal values of the confusion matrix, the better, indicating many correct predictions [68].

4.9 Graphical user interface

The GUI application used in the thesis is a virtual instrument cluster built and developed using Altia Design and Altia DeepScreen [69].

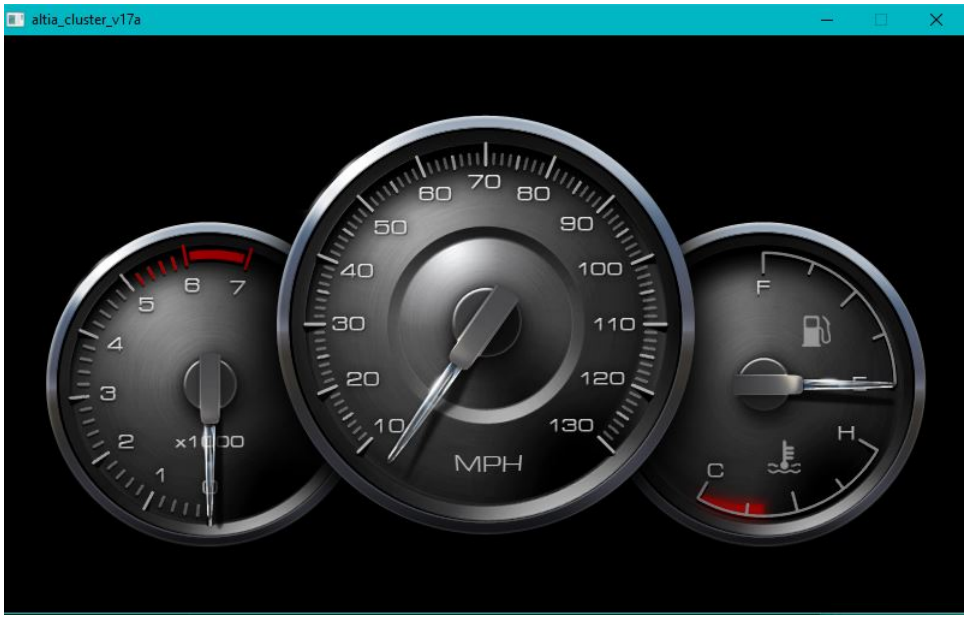


Figure 4.20: Altia virtual instrument cluster

The GUI is managed by two Python programs (because Altia Python library was built using Python 3.8)

- The first Python program is developed to predict the gesture from the trained TOF model. The code writes the predicted gesture in a file.
- Altia provides Python module that wraps some of the Altia API functions,which allow to control Altia interface from Python prediction program. Each predicted gestures are used to trigger specific event. Table 4.3 and Figure 4.21 illustrate the effect of each gesture on the GUI.

Gesture	Event
Swipe left	Display the menu
Swipe right	Close the menu
Swipe up	Move up in the menu
Swipe down	Move down in the menu
CW	Rotate the needles of the gauges in CW direction
CCW	Rotate the needles of the gauges in CCW direction
Unknown	Has no effect on GUI

Table 4.3: Gestures and events on GUI [69]

(a) Use swipe left gesture to open menu

(b) use swipe down gesture to move down in menu

(c) use swipe up gesture to move up in menu

(d) use swipe right gesture to close menu

(e) use CW gesture to increase gauges value

(f) use CCW gesture to decrease gauges value

Figure 4.21: GUI controlled by TOF Model predictions

Chapter 5

Experiments and results

In this chapter, all the experimental results obtained after the implementation and integration phase of the system are described.

Initially, the tests will be based on the analysis of calibration and the testing of the different sensors of the system. In this step, it is necessary to test if the individual elements are working and the system functioning and behavior are acceptable. Secondly, the architectures of each DNN models are demonstrated, and the accuracy of each model is evaluated considering accuracy. Thirdly a comparison between the performance of each sensor is made using the metrics described in chapter 4 . Finally, the results of the implementation of the gesture recognition algorithms and the interaction with the GUI system are analyzed.

5.1 TOF sensors calibration and ranging results

This section demonstrates the results of the calibration tests described in section 4.1 for both the VL6180 and VL53L3CX sensors. Moreover, it displays the measurements after the calibration.

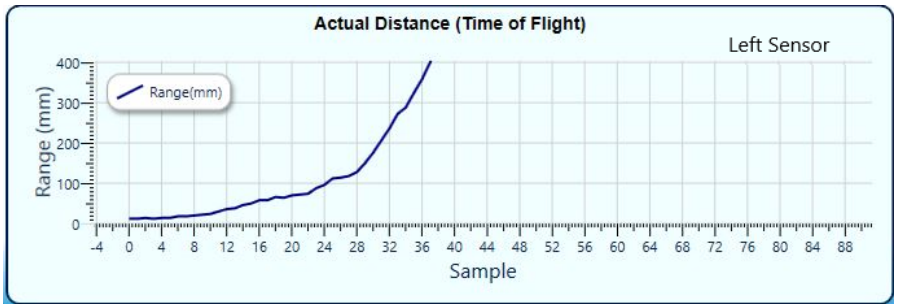
5.1.1 VL6180 calibration and ranging results

The VL6180 allows to customize the calibration as it has been described in chapter 4,Table 5.1 shows the results of offset and crosstalk calibration .

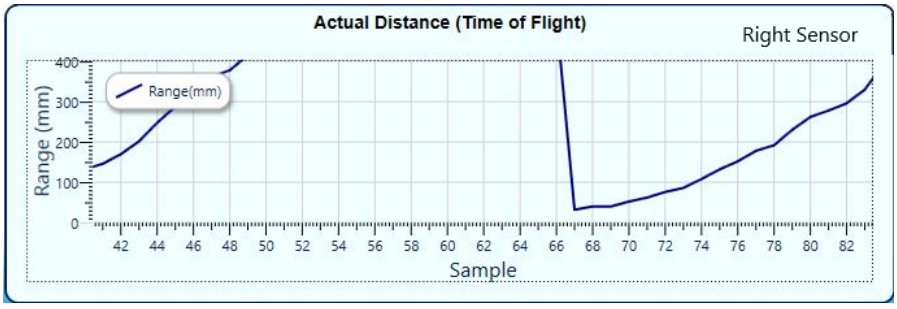
The plots in Figure 5.1 illustrate the actual ranges reported by the sensors after calibration; the maximum is considered 400 mm (upscale=2).

VL6180 sensor	Offset (mm)	crosstalk (Mcp)
Right	10	0.68
Center	1	0.92
Left	13	1.85

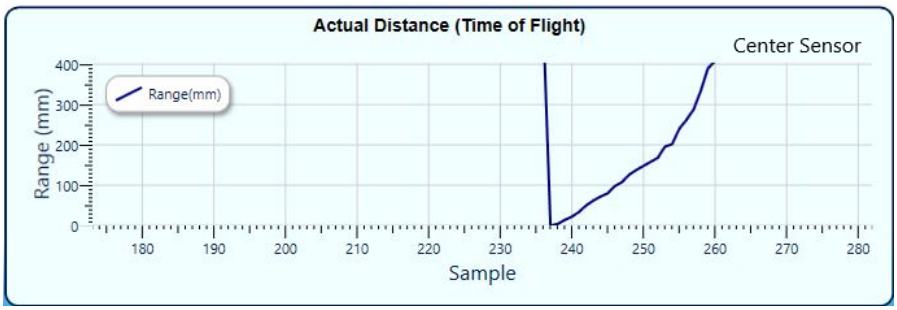
Table 5.1: VL6180 calibration results



(a) Left sensor



(b) Right sensor



(c) Center sensor

Figure 5.1: VL6180 ranging after calibration

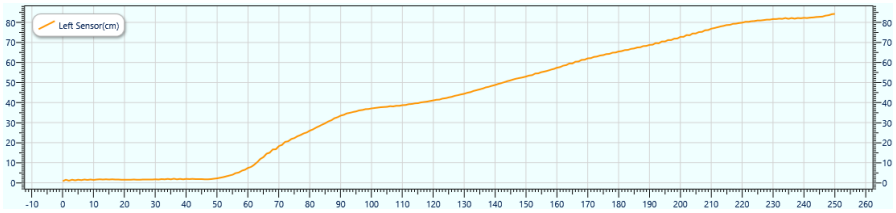
From the Figures, it can be concluded that the sensors show very reliable results up to 20 cm and not dependant on the factors such as reflectance.

5.1.2 VL53L3CX calibration and ranging results

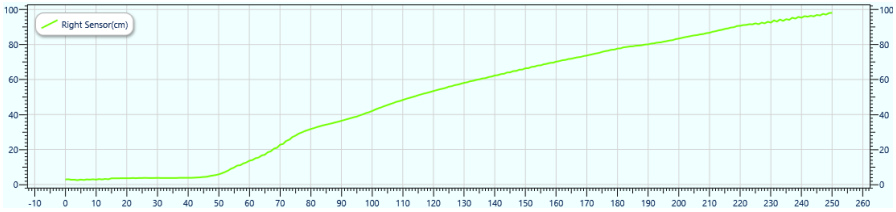
The Table 5.2 shows the offset and crosstalk calibration results of VL53L3CX sensors, whereas Figure 5.2 illustrates the ranging results (short mode =1.3 meter) after the calibration is performed.

VL53L3CX sensor	Offset (mm)	crosstalk (Mcp)
Right	-6	0.04
Center	-2	3.03
Left	-8	0.5

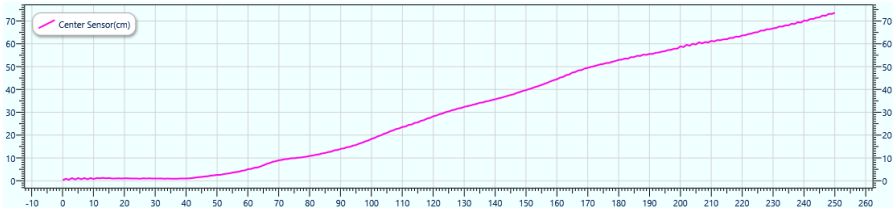
Table 5.2: VL53L3CX calibration results



(a) Left sensor



(b) Right sensor



(c) Center sensor

Figure 5.2: VL53L3CX ranging after calibration

Both TOF sensors show good distance measurement, but the VL6180 was chosen for comparison with the other sensors because it has a suitable range for gesture recognition. In contrast, the VL53L3CX has an extendable range of 1.3-5 meters.

5.2 Monitoring sensor data

Before starting the data collection for MLP, it is necessary to analyze the behavior of sensors; an identical ranging test is conducted for each sensor.

VL6180

To test the API developed for this sensor, a initial test consists in configuring and collecting data from the sensing device. This data is sent using UART in packets (the data is transmitted in hex format) as in Figure 5.4.

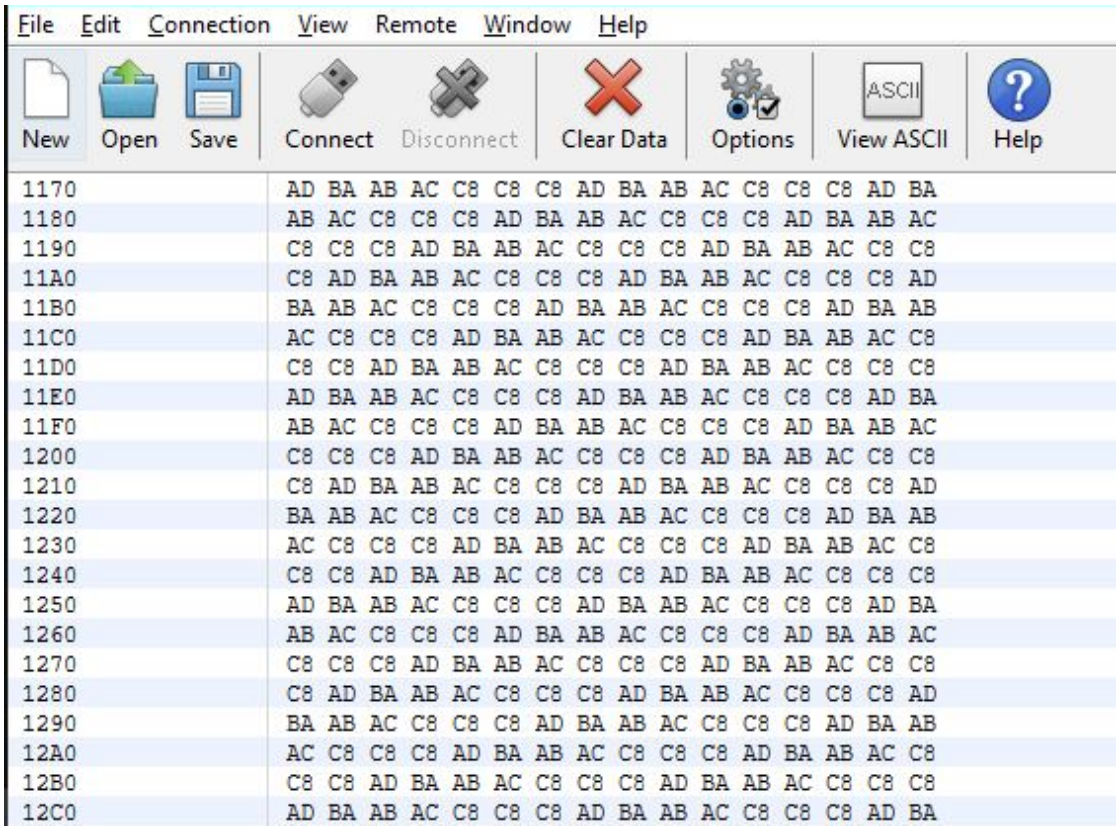


Figure 5.3: VL6180 serial data

VL53L3CX

To test the API developed for this sensor, an initial test consists in configuring and collecting data from the sensing device.The Figure 5.4 illustrates how the VL53L3CX sensors can detect a target at 2200mm.

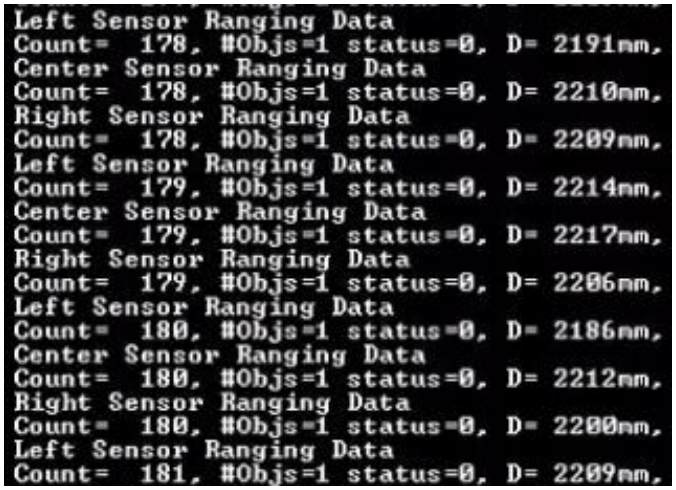


Figure 5.4: VL53L3CX serial data

The test was successful, because the sensing devices were well configured

MGC3130

To test the API developed for this sensor, a initial test is used to collect the data from the sensing device.The device is used to obtain the hand trajectory expressed through the hand positional data. Figure 5.5 shows the sequence of points of a right swipe.

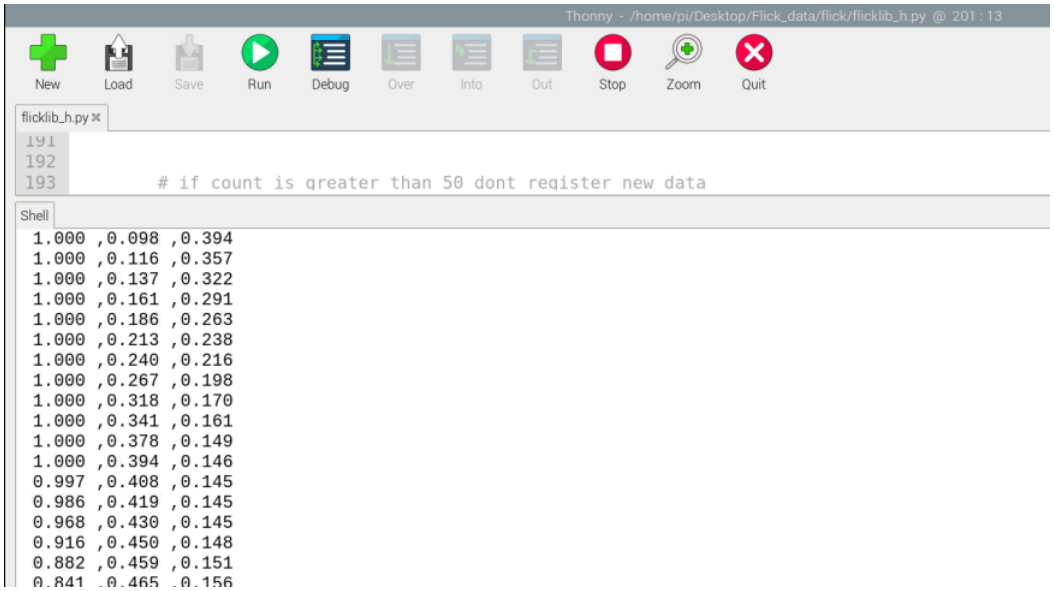


Figure 5.5: MGC3130 serial data

APDS9960

In order to detect the direction of the motion, the four diode values are recorded from the beginning to the end of the gesture. Figure 5.6 shows the results for right swipe.

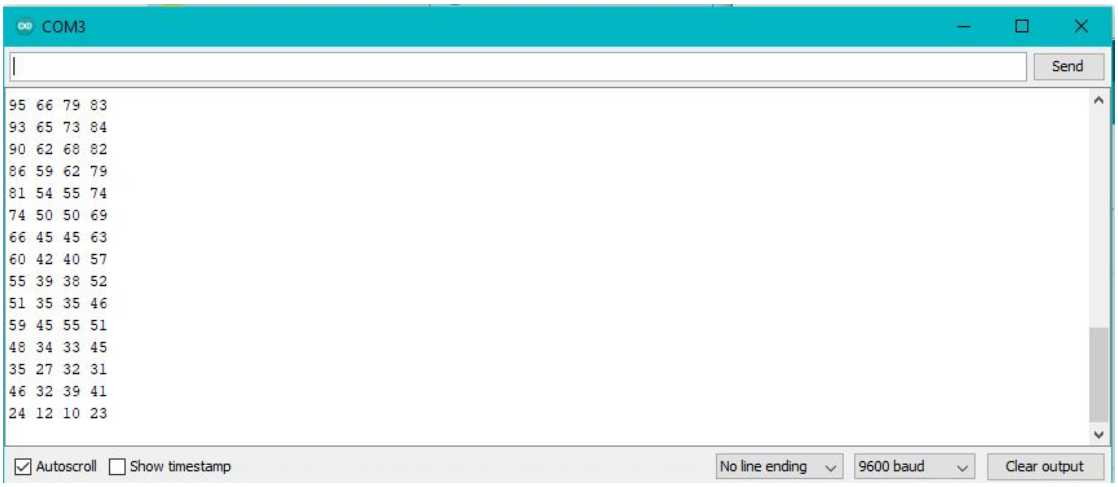


Figure 5.6: APDS9960 serial data

5.3 VL6180 prototype DNN

The dataset includes 17,500 balanced examples collected by five volunteers (each volunteer repeated each gesture 500 times). The dataset is split such that the training dataset contains 12,250 examples, while the validation dataset contains 5,250 examples. Three of the volunteers collected the test dataset, and each volunteer was asked to perform each of the gestures 50 times ($3 \times 50 \times 7$=1050 examples).

The Table 5.3 summaries the final architecture of the MLP model obtained from Keras Tuner to get the best results.

Layer	Nodes	Activation function
1	350	ReLU
2	250	ReLU
3	100	ReLU
Output	7	Softmax

Table 5.3: MLP model architecture for TOF

The model learning curves have been plotted in Figures 5.7 and 5.8 The model accuracy reached a value of 95%.

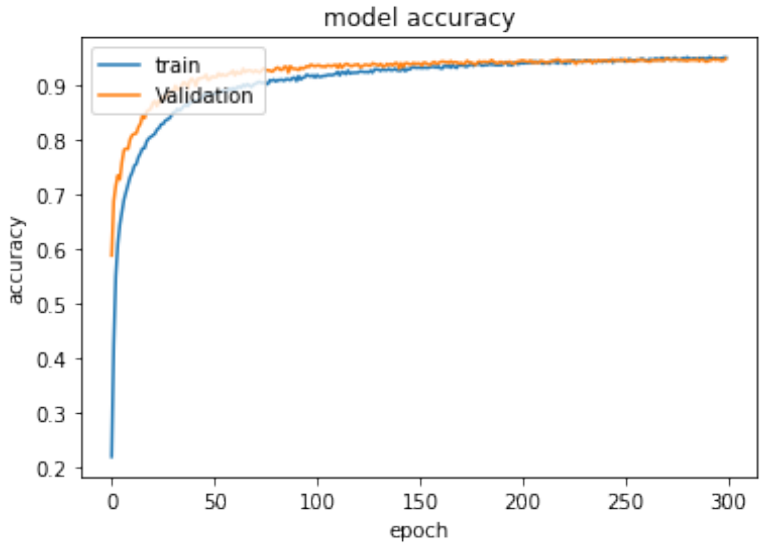


Figure 5.7: TOF model accuracy over epochs

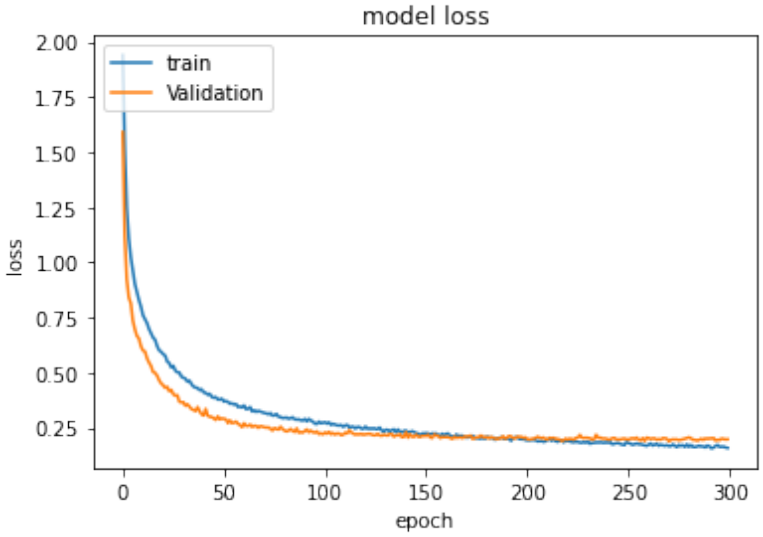


Figure 5.8: TOF model loss over epochs

From Figure 5.8, we can conclude that the training loss decreases to a point of

stability . Additionally, the validation loss drops and has a small gap with the training loss.

5.4 MGC3130 DNN

Like in the previous case, five users collected the datasets from MGC3130. The first dataset includes 7,000 balanced instances divided into a training dataset (4,900 examples) and a validation dataset (2,100 examples). The second dataset is a testing dataset collected from the same volunteers and includes 350 examples (5 × 10 × 7). The following Table reports the architecture after using Keras Tuner:

Layer	Nodes	Activation function
1	85	ReLU
2	45	ReLU
3	32	ReLU
4	32	ReLU
Output	7	Softmax

Table 5.4: MLP model architecture for MGC3130

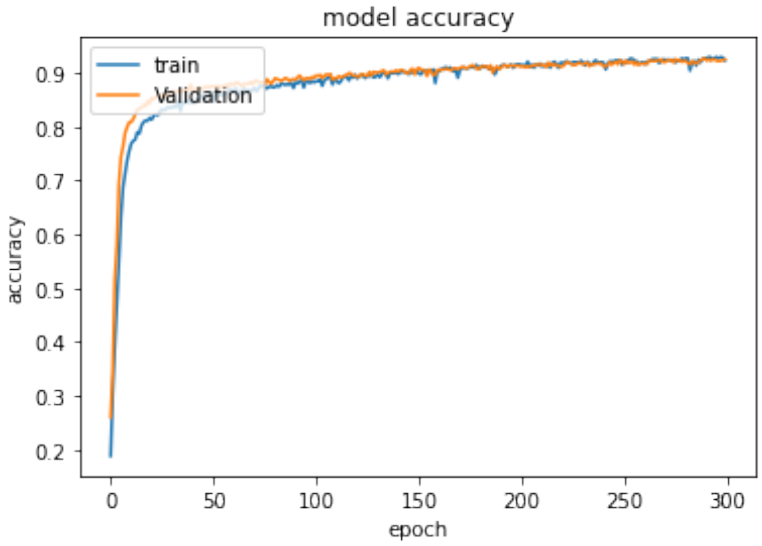


Figure 5.9: MGC3130 model accuracy over epochs

From the Figure 5.9, the model accuracy is 94% on both the training and the validation set.

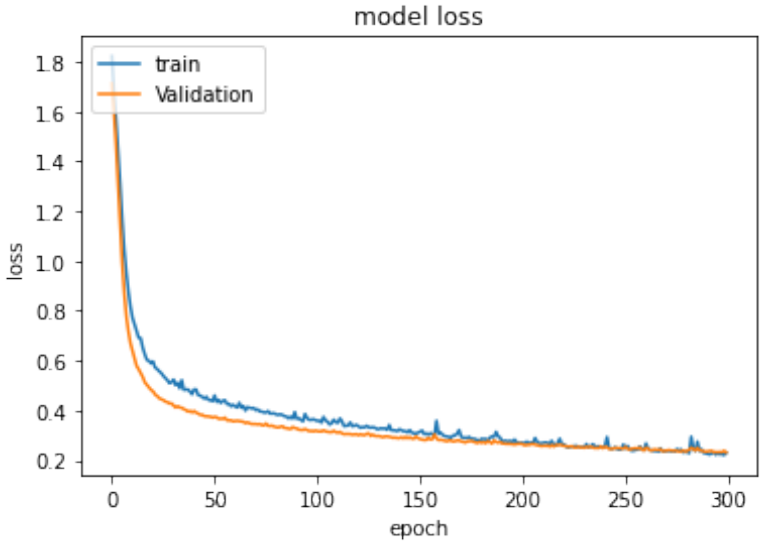


Figure 5.10: MGC3130 model loss over epochs

5.5 APDS9960 DNN

Like in the previous case, five users collected the datasets from the APDS9960. The total dataset includes 7,000 balanced instances divided into a training dataset (4,900 examples) and a validation dataset (2,100 examples). The same volunteers collected a test dataset contains 175 examples ($5 \times 7 \times 5$). The Table 5.5 reports the architecture after using Keras Tuner :

Layer	Nodes	Activation function
1	60	ReLU
2	55	ReLU
3	40	ReLU
4	20	ReLU
Output	7	Softmax

Table 5.5: MLP model architecture for APDS9960

From Figure 5.12, the model accuracy is 93% on both the training and the validation set.Even though, the accuracy is good ,there is a large gap between test and train results.

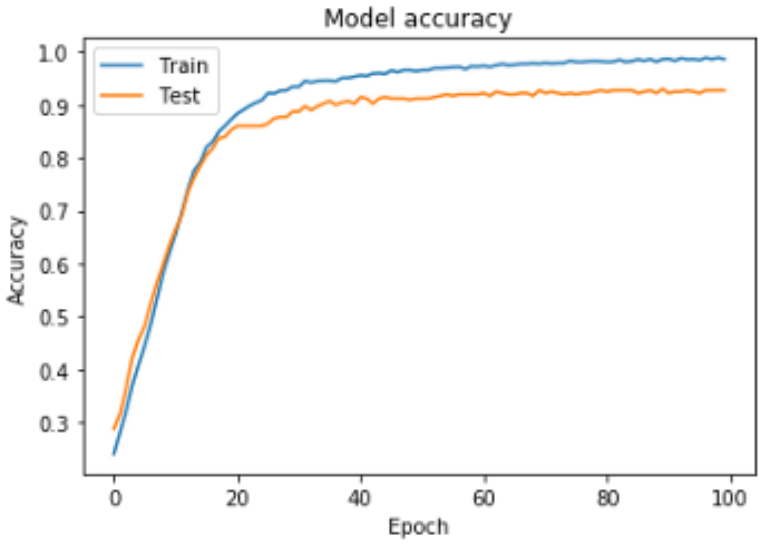


Figure 5.11: APDS9960 model accuracy over epochs

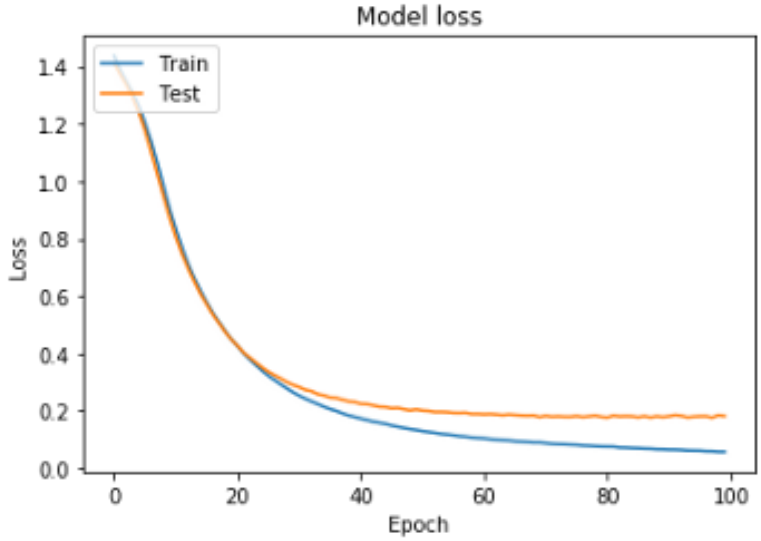


Figure 5.12: APDS9960 model loss over epochs

5.6 Evaluation and analysis

The F1-Measure analysis and confusion matrix, described in Chapter 4, was used to analyze the gesture recognition results for each sensor.

VL6180

The prediction (P), recall (R),accuracy (A), and F1 values were calculated for each gesture recognized by the ST VL6180 sensor: left,right,up,down,CW,CCW and unknown. The values of the four scores were calculated, and overall accuracy for the sensor was also calculated at the end. Table 5.6 shows the data obtained by the test cases. The confusion matrix, illustrated in Figure 5.9, shows the results of gesture recognition and the wrongly detected gestures. The results of the F1 measure analysis are shown in Table 5.7.

	Left	Right	Up	Down	CW	CCW	Unknown
True Positive(TP)	0.97	0.95	1.00	1.00	0.98	0.99	0.73
False Negative(TN)	0.98	0.98	1.00	0.99	1.00	0.99	1.00
False Positive(FP)	0.02	0.02	0.00	0.01	0.	0.01	0.00
True Negative(FN)	0.03	0.05	0.00	0.00	0.02	0.01	0.27

Table 5.6: Collected data for ST VL6180

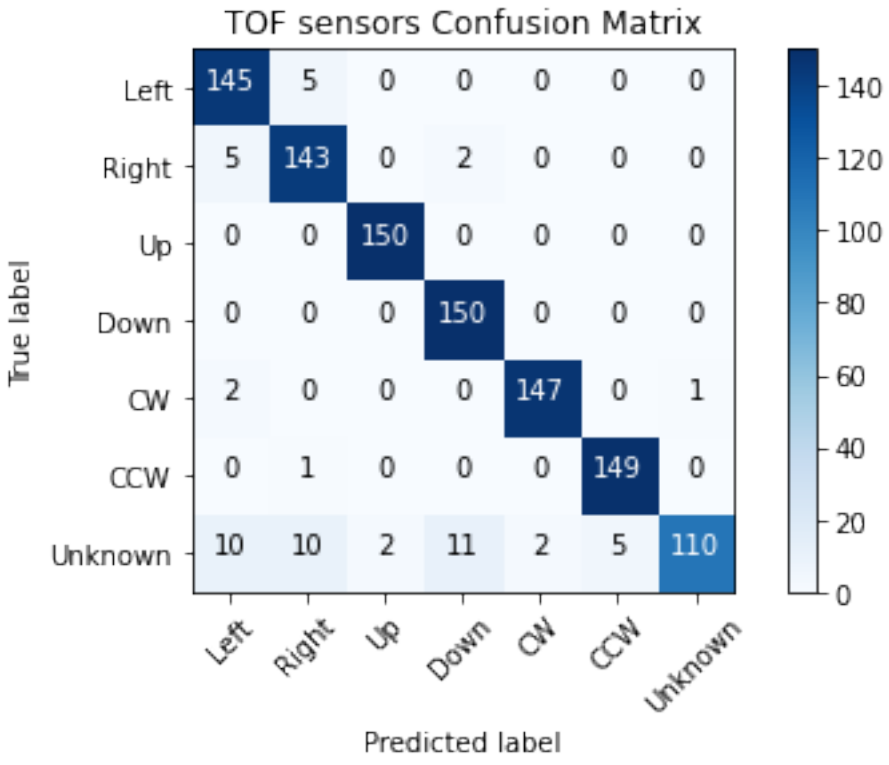


Figure 5.13: Confusion matrix for ST Vl6180 data

	Left	Right	Up	Down	CW	CCW	Unknown
P	0.90	0.90	0.99	0.92	0.99	0.97	0.99
R	0.97	0.95	1.00	1.00	0.98	0.99	0.73
A	0.98	0.98	1.00	0.99	1.00	0.99	0.96
F1	0.93	0.93	0.99	0.96	0.98	0.98	0.84

Table 5.7: F1-Measure analysis for ST VL6180

Flick Hat

The prediction (P), recall (R),accuracy (A), and F1 values were calculated for each gesture recognized by the MGC3130 sensor: left,right,up,down,CW,CCW and unknown. The values of the four scores were calculated, and overall accuracy for the sensor was also calculated at the end. Table 5.8 shows the data obtained by the test cases. The confusion matrix, illustrated in Figure 5.14, shows the results of gesture recognition and the wrongly detected gestures. The results of the F1-Measure analysis are shown in Table 5.9.

	Left	Right	Up	Down	CW	CCW	Unknown
True Positive(TP)	0.98	0.96	0.98	1.00	0.88	0.98	0.8
False Negative(TN)	1.00	0.99	1.00	0.99	0.99	0.98	0.99
False Positive(FP)	0.	0.01	0.	0.01	0.01	0.02	0.01
True Negative(FN)	0.02	0.04	0.02	0.	0.12	0.02	0.2

Table 5.8: Collected data for MGC3130

	Left	Right	Up	Down	CW	CCW	Unknown
P	0.98	0.96	1.00	0.94	0.92	0.88	0.91
R	0.98	0.96	0.98	1.00	0.88	0.98	0.80
A	0.99	0.99	1.00	0.99	0.97	0.98	0.96
F1	0.98	0.96	0.99	0.97	0.90	0.92	0.85

Table 5.9: F1-Measure analysis for MGC3130

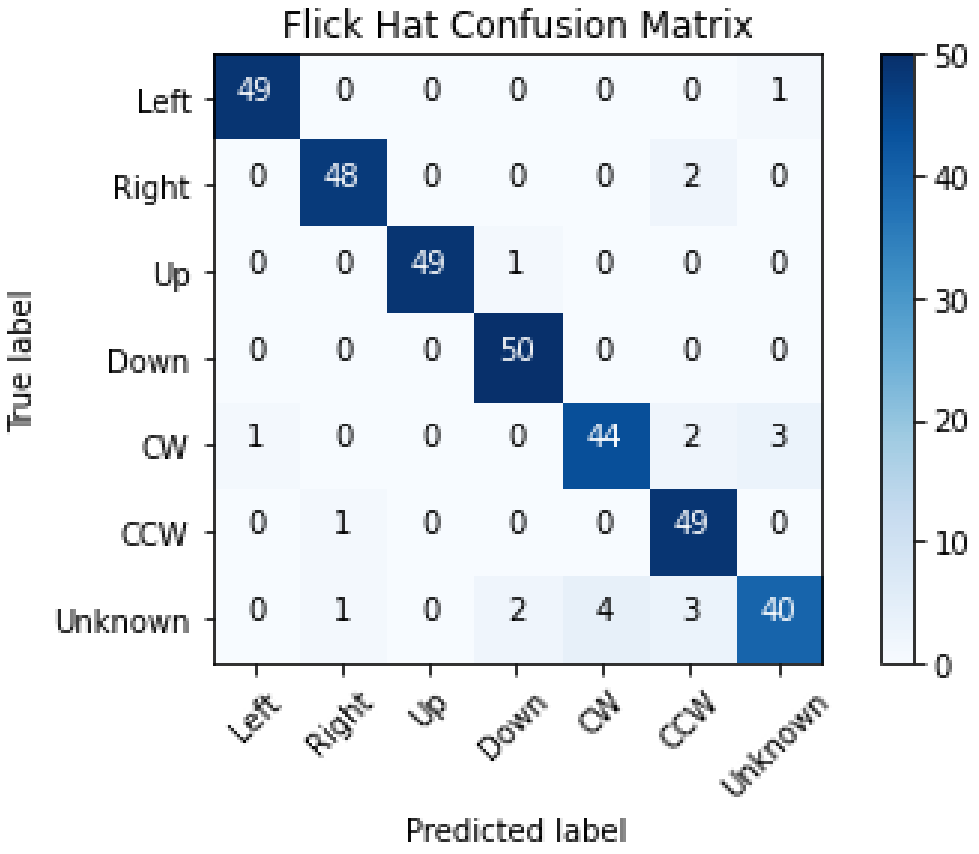


Figure 5.14: Confusion matrix for MGC3130 data

APDS 9960

The prediction (P), recall (R),accuracy (A), and F1 values were calculated for each gesture recognized by the APDS sensor: left,right,up,down,CW,CCW and Unknown. The values of the four scores were calculated, and overall accuracy for the sensor was also calculated at the end. Table 5.10 shows the data obtained by the test cases. The confusion matrix, illustrated in Figure 5.15, shows the results of gesture recognition and the wrongly detected gestures. The results of the F1-Measure analysis are shown in Table 5.11.

	Left	Right	Up	Down	CW	CCW	Unknown
True Positive(TP)	0.96	0.96	0.96	0.92	0.88	0.8	0.88
False Negative(TN)	0.98	0.97	0.97	0.99	0.99	1.00	0.99
False Positive(FP)	0.02	0.03	0.03	0.01	0.01	0.	0.01
True Negative(FN)	0.04	0.04	0.04	0.08	0.12	0.2	0.12

Table 5.10: Collected data for APDS9960

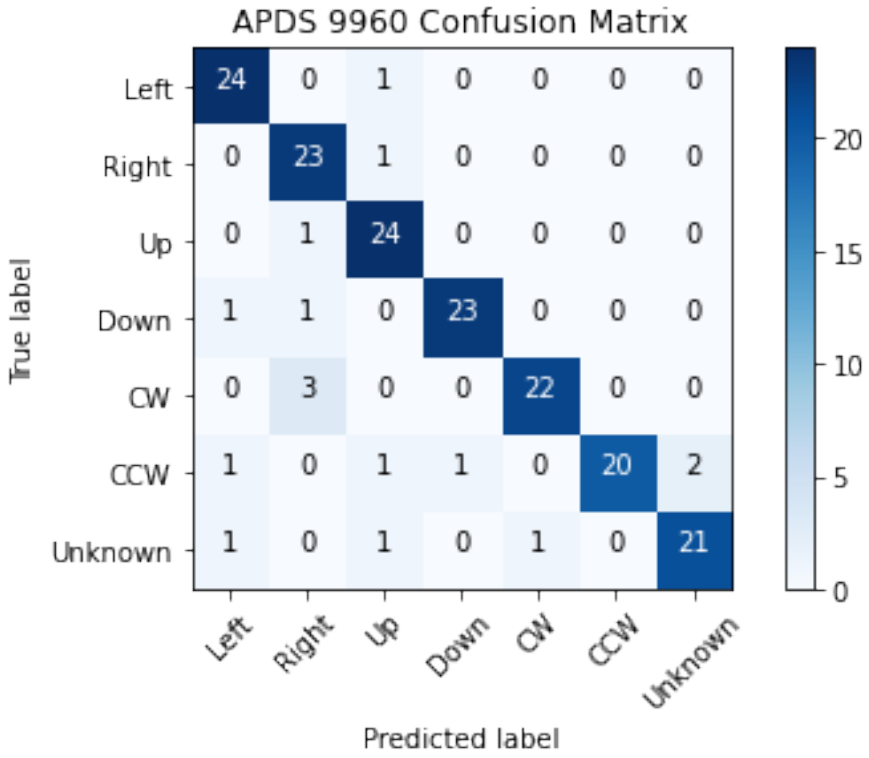


Figure 5.15: Confusion matrix for APDS9960 data

	Left	Right	Up	Down	CW	CCW	Unknown
P	0.90	0.81	0.93	0.96	0.94	1.00	1.00
R	0.93	0.97	0.87	0.80	0.97	1.00	0.97
A	0.98	0.97	0.97	0.98	0.98	0.97	0.97
F1	0.92	0.88	0.90	0.87	0.95	1.00	0.98

Table 5.11: F1-Measure analysis for APDS9960

5.7 Results discussion

The results of the test cases provide interesting information. The results show that the VL6180 prototype has the best performance among the three sensors.

The accuracy of the VL6180 model is 95%, while the accuracy of the MGC3130 model and the APDS model were 94% and 93%, respectively. Based on the precision and recall, both MGC3130 and VL6180 have similar results for directional swipe gestures. In addition, VL6180 also provides excellent results for rotational gestures.

In summary, the VL6180 prototype model has the best overall performance overall, followed by MGC3130. APDS 9960 wrongly detected many gestures, mainly CW, CCW, and unknown, which reduces the overall precision.

APDS9960 doesn't detect gestures over 2 cm, which contradicts the datasheet of range 10 cm to 20 cm,probably because this sensor doesn't allow custom calibration to adjust measurements for varying environments.

MGC3130 maximum range reported by datasheet is up to 15 cm; however, from experimental tests, the range for detection appears to be consistently less than 5 cm in any direction from the center of the device. Additionally, the sensor also struggles with continued distortions of the surrounding electromagnetic radiations. This is illustrated when the user holds a hand in a fixed position near the sensor. After around a second, the measurements vary uncontrollably without any motion from the user, leading to false data.

Chapter 6

Conclusion and future work

This chapter presents a brief reflection on the thesis development as well as a proposal for future work.

6.1 Conclusion

This paper gives an overview of various technologies used in gesture recognition with a deep focus on three of them, namely IR, capacitive, and TOF technologies. It began with the objective of evaluating the gesture recognition performance of four sensors using DNNs.The following sensors were considered: APDS9960, MGC3130, VL6180, and VL53L3CX, where VL53L3CX was excluded due to its long range.

The work also analyzes the working principle and the type of data provided by each sensor to measure the target distance.MGC3130 provides positional data, while APDS9960 represents the target distance using four proximity values.

The VL6180 provides only one distance value, which is insufficient to identify the gestures. Therefore, a prototype was designed consisting of a horizontal array of three VL6180 sensors.

Unknown, directional swipes and rotational gestures were considered to be classified using the DNNs model. Experimental data were collected from volunteers to determine the number of measurements required to identify a gesture. Consequently, the same volunteers collected a separate data set for each sensor. Each sensor dataset was split into training and validation datasets such that the training dataset was used to fit the model. On the other hand, the validation dataset was used to tune the model parameters.

Keras Tuner was used to tune the hyperparameters, and finally, the performance of each MLP model was evaluated using performance metrics, namely accuracy, precision, recall, and F1 score. The accuracy of the VL6180 model was higher than the accuracy of the other two sensors. Moreover, based on precision and recall, MGC3130 and VL6180 have similar results for directional swipe gestures. In addition, VL6180 also provides excellent results for rotational gestures.

In summary, the VL6180 prototype model has the best overall performance, followed by MGC3130. In addition, the VL6180 prototype was able to control a GUI in real-time operation. Several observations were made during testing and data collection: the APDS9960 has a smaller operating range than the range specified in the datasheet [54]. Similarly, the capacitive sensor measurements are distorted by nearby conductive materials.

6.2 Future work

A suggestion for future work is to integrate the VL6180 prototype into a real-time hardware application and evaluate its performance.

In terms of gesture recognition using machine learning, a suitable improvement can be either improving the current algorithms or introducing new algorithms. These changes in algorithms may lead to better recognition results.

Bibliography

[1] *Gesture Recognition and Touchless Sensing Market by Technology (Touch-Based and Touchless)- Global Forecast to 2025*. `https://www.researchandmarkets.com/reports/5011620`. [Online]. 2020 (cit. on p. ii).

[2] Jia-Qing Liu, Tomoko Tateyama, Yutaro Iwamoto, and Yen-Wei Chen. «Kinect-based real-time gesture recognition using deep convolutional neural networks for touchless visualization of hepatic anatomical models in surgery». In: *International Conference on Intelligent Interactive Multimedia Systems and Services*. Springer. 2018, pp. 223–229 (cit. on p. 1).

[3] Nikola Laković, Miodrag Brkić, Branislav Batinić, Jovan Bajić, Vladimir Rajs, and Nenad Kulundžić. «Application of low-cost VL53L0X ToF sensor for robot environment detection». In: *2019 18th International Symposium INFOTEH-JAHORINA (INFOTEH)*. IEEE. 2019, pp. 1–4 (cit. on pp. 1, 13).

[4] Nico Zengeler, Thomas Kopinski, and Uwe Handmann. «Hand gesture recognition in automotive human–machine interaction using depth cameras». In: *Sensors* 19.1 (2019), p. 59 (cit. on pp. 1, 4).

[5] Jana M Iverson and Susan Goldin-Meadow. «Why people gesture when they speak». In: *Nature* 396.6708 (1998), pp. 228–228 (cit. on p. 3).

[6] David McNeill. *Language and gesture*. Vol. 2. Cambridge University Press Cambridge, 2000 (cit. on p. 3).

[7] Chunyu Zou, Yue Liu, Jiabin Wang, and Huaqi Si. «Deformable part model based hand detection against complex backgrounds». In: *Chinese Conference on Image and Graphics Technologies*. Springer. 2016, pp. 149–159 (cit. on p. 3).

[8] Karol Miądlicki, Mateusz Saków, et al. «The use of machine vision to control the basic functions of a CNC machine tool using gestures». In: *Czasopismo Techniczne* 2017.Volume 12 (2017), pp. 213–229 (cit. on p. 4).

[9] T NagaKarthik, Eun Hye Ahn, Yun Sik Bae, and Jun Rim Choi. «TCAM based pattern matching technique for hand gesture recognition». In: *2013 International SoC Design Conference (ISOCC)*. IEEE. 2013, pp. 368–369 (cit. on p. 4).

[10] Ebrahim Nasr-Esfahani, Nader Karimi, SM Soroushmehr, M Hossein Jafari, M Amin Khorsandi, Shadrokh Samavi, and Kayvan Najarian. «Hand gesture recognition for contactless device control in operating rooms». In: *arXiv preprint arXiv:1611.04138* (2016) (cit. on p. 4).

[11] Siba Kumar Udgata and Nagender Kumar Suryadevara. «Advances in Sensor Technology and IoT Framework to Mitigate COVID-19 Challenges». In: *Internet of Things and Sensor Network for COVID-19*. Springer, 2021, pp. 55–82 (cit. on p. 4).

[12] Doe-Hyung Lee and Kwang-Seok Hong. «Game interface using hand gesture recognition». In: *5th International Conference on Computer Sciences and Convergence Information Technology*. IEEE. 2010, pp. 1092–1097 (cit. on p. 4).

[13] Jérôme Van Zaen, Jody Hausmann, Kevin Salvi, and Michel Deriaz. «Gesture recognition for interest detection in mobile applications». In: *2014 International Conference on Multimedia Computing and Systems (ICMCS)*. IEEE. 2014, pp. 345–350 (cit. on p. 4).

[14] David Cohen. «Magnetic fields around the torso: production by electrical activity of the human heart». In: *Science* 156.3775 (1967), pp. 652–654 (cit. on p. 5).

[15] Wikipedia contributors. *Capacitive$_s$ensing*. Online; accessed 22-May-2021. 2004. URL: `https://en.wikipedia.org/wiki/Capacitive_sensing` (cit. on p. 5).

[16] Joshua Reynolds Smith. «Electric field imaging». PhD thesis. Massachusetts Institute of Technology, 1999 (cit. on pp. 5, 6).

[17] Tobias Grosse-Puppendahl and Andreas Braun. «Honeyfish-a high resolution gesture recognition system based on capacitive proximity sensing». In: *Embedded World Conference*. Vol. 12. 2012 (cit. on p. 5).

[18] Microchip. *MGC3030/3130 3D Tracking and Gesture Controller Data*. DS40001667E¦ ev. Microchip: Chandler, AZ, USA, 2017. Oct. 2018 (cit. on pp. 6, 31, 32).

[19] . *shielding*. URL: `https://e2e.ti.com/blogs_/b/analogwire/posts/what-are-you-sensing-active-shielding-for-capacitive-sensing_2c00_-part-1` (cit. on p. 7).

[20] Raphael Wimmer, Paul Holleis, Matthias Kranz, and Albrecht Schmidt. «Thracker-using capacitive sensing for gesture recognition». In: *26th IEEE International Conference on Distributed Computing Systems Workshops (ICDCSW'06)*. IEEE. 2006, pp. 64–64 (cit. on pp. 7, 8).

[21] Tobias Grosse-Puppendahl, Andreas Braun, Felix Kamieth, and Arjan Kuijper. «Swiss-cheese extended: an object recognition method for ubiquitous interfaces based on capacitive proximity sensing». In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 2013, pp. 1401–1410 (cit. on p. 8).

[22] Leonhard Haslinger, Simon Wasserthal, and BG Zagar. «P3. 1-A capacitive measurement system for gesture recognition». In: *Proceedings Sensor 2017* (2017), pp. 616–620 (cit. on p. 8).

[23] . *Infrared sensors*. URL: `https://coolcosmos.ipac.caltech.edu/page/herschel_experiment` (cit. on p. 10).

[24] Chieko Asakawa, Hironobu Takagi, Shuichi Ino, and Tohru Ifukube. «Auditory and tactile interfaces for representing the visual effects on the web». In: *Proceedings of the fifth international ACM conference on Assistive technologies*. 2002, pp. 65–72 (cit. on p. 10).

[25] Shiri Azenkot and Emily Fortuna. «Improving public transit usability for blind and deaf-blind people by connecting a braille display to a smartphone». In: *Proceedings of the 12th international ACM SIGACCESS conference on Computers and accessibility*. 2010, pp. 317–318 (cit. on p. 10).

[26] Ali Abdolrahmani, Ravi Kuber, and Amy Hurst. «An empirical investigation of the situationally-induced impairments experienced by blind mobile device users». In: *Proceedings of the 13th International Web for All Conference*. 2016, pp. 1–8 (cit. on p. 10).

[27] Saroja Kanta Panda and Sushanta Kumar Sahu. «Design of IoT-Based Real-Time Video Surveillance System Using Raspberry Pi and Sensor Network». In: *Intelligent Systems: Proceedings of ICMIB 2020* (2020), p. 115 (cit. on p. 10).

[28] Harpreet Kaur and Jyoti Rani. «A review: Study of various techniques of Hand gesture recognition». In: *2016 IEEE 1st International Conference on Power Electronics, Intelligent Control and Energy Systems (ICPEICES)*. IEEE. 2016, pp. 1–5 (cit. on p. 11).

[29] Rajeshri R Itkarkar and Anilkumar V Nandi. «A survey of 2D and 3D imaging used in hand gesture recognition for human-computer interaction (HCI)». In: *2016 IEEE International WIE Conference on Electrical and Computer Engineering (WIECON-ECE)*. IEEE. 2016, pp. 188–193 (cit. on p. 11).

[30] *3d-time-of-flight-camera*. https://www.terabee.com/3d-time-of-flight-cameras-an-introduction/. [Online]. 2020 (cit. on p. 11).

[31] Joaquim Salvi, Jordi Pages, and Joan Batlle. «Pattern codification strategies in structured light systems». In: *Pattern recognition* 37.4 (2004), pp. 827–849 (cit. on p. 11).

[32] Kai Liu, Yongchang Wang, Daniel Lau, Qi Hao, and Laurence Hassebrook. «Towards gesture-controlled computers with real-time structured light». In: () (cit. on p. 11).

[33] . *STMelectronics*. [Online; accessed 2020]. 2017. URL: https://www.st.com/content/dam/technology-tour-2017/session-1_track-4_time-of-flight-technology.pdf (cit. on pp. 12, 13).

[34] Raquel Nicolau Vidal. «Omnidirectional scanner using a time of flight sensor». B.S. thesis. Universitat Politècnica de Catalunya, 2018 (cit. on p. 13).

[35] Odysseus Alexander Adamides, Anmol Saiprasad Modur, Shitij Kumar, and Ferat Sahin. «A time-of-flight on-robot proximity sensing system to achieve human detection for collaborative robots». In: *2019 IEEE 15th International Conference on Automation Science and Engineering (CASE)*. IEEE. 2019, pp. 1230–1236 (cit. on p. 13).

[36] Gorkem Anil Al, Pedro Estrela, and Uriel Martinez-Hernandez. «Towards an intuitive human-robot interaction based on hand gesture recognition and proximity sensors». In: *2020 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*. IEEE. 2020, pp. 330–335 (cit. on p. 13).

[37] Stuart Russell and Peter Norvig. *Artificial intelligence: a modern approach*. 2002 (cit. on p. 14).

[38] Tom Michael Mitchell. *The discipline of machine learning*. Vol. 9. Carnegie Mellon University, School of Computer Science, Machine Learning . . ., 2006 (cit. on p. 14).

[39] . *Deep Neural Network*. [Online; accessed 1-Jan-2020]. URL: https://machinelearningmastery.com/ (cit. on pp. 15, 21, 22, 61).

[40] . *Machine Learning*. [Online; accessed 25-April-2019]. URL: https://towardsdatascience.com/understanding-learning-rates-and-how-it-improves-performance-in-deep-learning-d0d4059c1c10 (cit. on pp. 16, 18, 20).

[41] Frank Rosenblatt. «The perceptron: a probabilistic model for information storage and organization in the brain.» In: *Psychological review* 65.6 (1958), p. 386 (cit. on p. 17).

[42] Warren S McCulloch and Walter Pitts. «A logical calculus of the ideas immanent in nervous activity». In: *The bulletin of mathematical biophysics* 5.4 (1943), pp. 115–133 (cit. on p. 17).

[43] C. State. *History of the perceptron*. 2017 (cit. on p. 17).

[44] Donald Olding Hebb. *The organization of behavior: A neuropsychological theory*. Psychology Press, 2005 (cit. on p. 17).

[45] . *Machine Learning*. [Online; accessed 12-Feb-2012]. URL: `https://www.coursera.org/learn/machine-learning` (cit. on pp. 17, 19, 20).

[46] Karen Simonyan and Andrew Zisserman. «Very deep convolutional networks for large-scale image recognition». In: *arXiv preprint arXiv:1409.1556* (2014) (cit. on p. 18).

[47] . *Keras*. URL: `https://keras.io/` (cit. on pp. 18, 60).

[48] Leslie N Smith. «Cyclical learning rates for training neural networks». In: *2017 IEEE winter conference on applications of computer vision (WACV)*. IEEE. 2017, pp. 464–472 (cit. on pp. 20, 21).

[49] Diederik P Kingma and Jimmy Ba. «Adam: A method for stochastic optimization». In: *arXiv preprint arXiv:1412.6980* (2014) (cit. on pp. 21, 22).

[50] Sankar K Pal and Sushmita Mitra. «Multilayer perceptron, fuzzy sets, classifiaction». In: (1992) (cit. on p. 23).

[51] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. «Multilayer feedforward networks are universal approximators». In: *Neural networks* 2.5 (1989), pp. 359–366 (cit. on p. 23).

[52] STMicroelectronics. *Integrated Development Environment for STM32*. `https://www.st.com/en/development-tools/stm32cubeide.html`. [Online] (cit. on p. 29).

[53] . `https://en.wikipedia.org/wiki/Arduino_IDE`. [Online] (cit. on p. 29).

[54] *Digital Proximity, Ambient Light, RGB and Gesture Sensor*. AV02-4191EN. Rev. 3. Avago Technologies. Nov. 2015 (cit. on pp. 31, 59, 84).

[55] *MGC3030/3130 single zone 3D Tracking and Gesture Controller Product brief;* DS40001667E. Microchip: Chandler, AZ, USA, 2017. 2015. URL: `https://ww1.microchip.com/downloads/en/DeviceDoc/40001662B.pdf` (cit. on p. 32).

[56] Josue Ferri, Raúl Llinares Llopis, Jorge Moreno, Javier Ibanez Civera, and Eduardo Garcia-Breijo. «A wearable textile 3D gesture recognition sensor based on screen-printing technology». In: *Sensors* 19.23 (2019), p. 5068 (cit. on p. 33).

[57] Oded Cohen, Raphael Linker, and Amos Naor. «Estimation of the number of apples in color images recorded in orchards». In: *International Conference on Computer and Computing Technologies in Agriculture*. Springer. 2010, pp. 630–642 (cit. on p. 33).

[58] . *STMelectronics*. [Online; accessed 16-Dec-2020]. URL: https://electronicsmaker.com/the-vl53l5-is-a-state-of-the-art-tof-laser-ranging-sensor-enhancing-the-st-flightsense-product-family (cit. on p. 33).

[59] *API User Manual for the VL6180 proximity sensor*. UM2760 - Rev 2. STMelectronics. 2020. URL: https://www.st.com/resource/en/user_manual/dm00727462-api-user-manual-for-the-vl6180-proximity-sensor-stmicroelectronics.pdf (cit. on pp. 34–37, 41–43).

[60] *Datasheet for the VL53L3CX sensor*. DS13204 - Rev 3. STMelectronics. March 2021. URL: https://www.st.com/resource/en/datasheet/vl53l3cx.pdf (cit. on pp. 37–40, 47).

[61] ST. URL: https://www.st.com/resource/en/user_manual/dm00474730-vl53l1x-api-user-manual-stmicroelectronics.pdf (cit. on p. 39).

[62] . *STMelectronics*. [Online; accessed 3-Jan-2021]. URL: https://cdn.sparkfun.com/datasheets/Sensors/Proximity/VL6180_ApplicationNote.pdf (cit. on pp. 48, 50).

[63] Catherine C Cooksey and David W Allen. «Reflectance measurements of human skin from the ultraviolet to the shortwave infrared (250 nm to 2500 nm)». In: *Active and Passive Signatures IV*. Vol. 8734. International Society for Optics and Photonics. 2013, 87340N (cit. on pp. 50, 51).

[64] . URL: M.%20T.%20Incorporated,%20Aurea%20Graphical%20User%20Interface%20User%20E2%80%99%20s%20Guide.%202015. (cit. on pp. 57, 58).

[65] . *Colab*. URL: https://colab.research.google.com/notebooks/intro.ipynb?utm_source=scs-index#recent=true (cit. on p. 60).

[66] . *Hyperparameters tuning*. URL: https://www.oreilly.com/library/view/evaluating-machine-learning/9781492048756/ch04.html (cit. on p. 62).

[67] AF Rogachev and EV Melikhova. «Automation of the process of selecting hyperparameters for artificial neural networks for processing retrospective text information». In: *IOP Conference Series: Earth and Environmental Science*. Vol. 577. 1. IOP Publishing. 2020, p. 012012 (cit. on p. 62).

[68] . *plot confusion matrix*. URL: https://scikit-learn.org/stable/auto_examples/model_selection/plot_confusion_matrix.html (cit. on p. 65).

[69] URL: https://www.altia.com/ (cit. on pp. 65, 66).