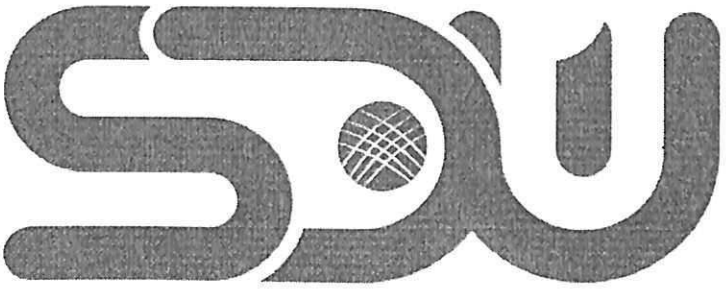
Ministry of Education and Science of the Republic of Kazakhstan
Suleyman Demirel University



Orazaiym Sarybay

Recognition of basic hand gestures on a horizontal surface using a single camera.

THESIS

Presented in Partial Fulfillment for the
Degree of Master of Science in Computer Science
(degree code: 7M06102)
Department of Computer Sciences
Faculty of Engineering and Natural Sciences

Supervisor: **Associate Professor, PhD Azamat Zhamanov**

Kaskelen, 2022

Suleyman Demirel University

Faculty of Engineering and Natural Sciences

Department of Computer Science

Dean of Faculty

Associate Professor, PhD Zhamanov A.

«20» 06 2022

Topic of the thesis:

Recognition of basic head gestures on a horizontal surface using a single camera

Thesis submitted as part of the requirements for the award of the MSc in "7M06102 - Computer Science", SDU, 2020-2022

Head of Department Assoc. Prof. PhD Cemil Turan

Academic Supervisor Assos. Prof. PhD Aqamat Zhamanov

Master's student Sarybay Orazayim

Kaskelen, 2022

Abstract

The horizontal hand gesture recognition is an innovative, cheaper way for human-computer interaction. Currently, most researchers work with sensors, devices for hand gesture recognition, which require more resources. Instead, the presented horizontal method for hand gesture signal recognition by frames. A key element of this work is the research of a recognition algorithm using only single camera. In the presented framework, the hand detection works as a converting BGR image to RGB before processing. Then, the palm and fingers are segmented so as to detect and recognize the fingers. There are handedness and hand landmarks on the image as a result of a hand detection. Each point of landmark has coordination x, y, z values. There is comparing algorithm of points to recognize hand gesture by fingers. The model has been implemented getting landmark values on a data set of hand images, which collected from video frames. In the presented framework, the hand detection works with computer vision (CV) algorithms, in general MediaPipe as a converting blue, green, red (BGR) image to red, green, blue (RGB) before processing. There are handedness and hand landmarks on the image as a result of a hand detection. Each point of the landmark has coordination x, y, z values. The performance of the method highly depends on the result of hand detection on horizontal surface and collected dataset.

Аңдатпа

Көлденең қол қимылын тану адам мен компьютердің әрекеттесуінің инновациялық, арзан әдісі болып табылады. Қазіргі уақытта зерттеушілердің көпшілігі қол қимылын тану құрылғыларымен, сенсорлармен жұмыс істейді. Ол құрылғылар көбірек ресурстарды қажет етеді. Оның орнына бұл жұмыста кадрлар арқылы қол қимылын көлденең жазықтықта танудың әдісі ұсынылған. Бұл жұмыстың негізгі элементі бір ғана камераны пайдаланып тану алгоритмін зерттеу болып табылады. Ұсынылған құрылымда қолды анықтау өңдеуден бұрын BGR кескінін RGB түріне түрлендіру ретінде жұмыс істейді. Содан кейін саусақтарды анықтау және тану үшін алақан мен саусақтар сегменттерге бөлінеді. Қолды анықтау нәтижесінде қолдың суреті және қол белгілері, қолдың нүктелері бар. Әрбір бағдар нүктесінде x, y, z координациялық мәндері болады. Қолдың қимылын саусақпен тану үшін саусақ нүктелерін салыстыру алгоритмы жазылды. Кескіндерден алынғанбейне кадрлардан жиналған қол суреттерінен деректер жинап, бағдарлық мәндерінен модельді оқыту жүзеге асырылды. Ұсынылған құрылымда қолды анықтау компьютерлік көру (CV) алгоритмдерімен жұмыс істейді. MediaPipe кітапханасы өңдеуден бұрын көк, жасыл, қызыл (BGR) кескінді қызыл, жасыл, көк (RGB) түрлендіру ретінде алады. Қолды анықтау нәтижесінде бейнеленген қолдың суреті және қол белгілері, нүктелері бейнеленген. Белгінің әрбір нүктесінде x, y, z координациялық мәндері болады. Әдістің өнімділігі көлденең беттегі қолды анықтау нәтижесіне және жиналған деректер жиынтығына байланысты.

Аннотация

Горизонтальное распознавание жестов рук — это инновационный и дешевый способ взаимодействия человека с компьютером. В настоящее время большинство исследователей используют датчики, устройства для распознавания жестов рук, которые требуют больше ресурсов. Вместо этого представлен горизонтальный метод распознавания жестов рук по кадрам. Ключевым элементом этой работы является исследование алгоритма распознавания с использованием только цветной камеры. В представленной структуре обнаружение руки, используется преобразование изображения BGR в RGB перед обработкой. Затем применяется сегментация ладони и пальцев, чтобы обнаружить их точки. В результате обнаружения руки на изображении присутствуют рукоятки и ориентиры рук. Каждая точка ориентира имеет координационные значения x, y, z. Также применяется алгоритм для сравнения точек и распознавания жестов руки по пальцам. Модель реализована для получения значений ориентиров на наборе данных изображений руки, собранных из видеокадров и ручного сбора по кадрам. В представленной структуре, используются алгоритмы компьютерного зрения(CV) для обнаружение руки, библиотека MediaPipe, и перед обработкой идет преобразование синего, зеленого, красного (BGR) изображения в красное, зеленое, синее (RGB). В результате обнаружения руки на изображении показываются ориентиры рук с точками. Каждая точка ориентира имеет координационные значения в пространстве x, y, z. Производительность метода сильно зависит от результата обнаружения руки на горизонтальной поверхности и собранного набора данных. вития.

Acknowledgements

I want to say thanks to Ministry of Science of Republic of Kazakhstan for grant they gave me to be a student again of the great Suleyman Demirel University. Thanks to all teachers, university members, who works for the good of the university. Big thanks to my supervisor, he always tried to help me with dissertation, solve the problems of the dissertation.

Thanks to my husband, to his a great support and help. Thanks to my daughter, who was born during this wonderful period of the study. She is a wonderful girl. Thanks to my parents, who supported me and helped me with a study. Thanks to everybody, who helped with documents, to hand them over.

Contents

Nomenclature

AI Artificial Intelligence

BGR Blue Green Red

CV Computer Vision

HCI Human Computer Interaction

KNN k-Nearest Neighbor Algorithm

ML Machine Learning

NN Neural Networks

OpenCV Open Source Computer Vision Library

RGB Red Green Blue

SVM Support Vector Machine

1. Introduction

1.1 Motivation

During the course of the last several decades, a significant number of researchers have focused their efforts on developing hand gesture detection technologies. Hand gesture recognition has a lot of uses, some of which include recognizing sign language [11, 10, 21], augmented reality (virtual reality) [3, 9, 4, 5], sign language interpreters for the disabled [6], and robot control [15, 7].

The authors of [1, 2] first recognize American Sign Language by identifying the hand area in the input photos, and then they follow and analyze the movement route of the hand. Shimada et al. [10] offer a control interface for a television that makes use of hand gesture recognition. In order to categorize different hand gestures, Keskin et al. [7] split the hand into 21 distinct areas and then train an SVM classifier to simulate the combined distribution of these regions for each of the different hand motions.

The identification of hand gestures, as demonstrated by Zeng et al. [6] results in an improvement to the quality of medical care. There are five different hand gestures and three different compound states that make up the HCI recognition system of the intelligent wheelchair. Their method operates dependably both inside and outdoors, as well as in conditions where the illumination might shift.

Working with only two fingers, discovering the gesture by using these finger instructions rather than only one command, which can be done using only one frame, is the primary distinction between this project and other hand recognition projects. Both Luiz Henrique da Silva Santos and Matheus Vyctor Aranda Espindola are working on a project called "Gesture Hand Controller," which detects hand commands made with any or all of the fingers. They demonstrate one command with their hand, the program recognizes it, and then it creates the

appropriate command [12]. For instance, the hand motion "like" the big finger up implies "click," "zoom out," and "zoom in." These commands are detected by the first and second fingers of the hand. The zoom out command will be executed when the distance between the fingers is large, and the zoom in command will be executed when the space between the fingers is little. There will be no animation for any of these commands [14]. This project's dataset was also compiled using data collected by the MediaPipe library. It functions properly with the dataset, which is a single file containing x, y, and z values. Because of this, it is able to identify the frame in real time [13].

1.2 Aims and Objectives

Using CV and ML algorithms to make life simpler is one of the goals of this research. This is especially true for the lives of handicapped persons. The cheapest method for human-computer interaction (HCI) is the identification of hand gestures by a single camera, because every laptop contains a camera. There are algorithms based on both CV and ML that may be used to achieve the outcome of a hand gesture on a horizontal surface. Hand recognition requires the software to have hand detection, hand landmarks detection algorithms, and trained models based on datasets. These are the components that make up the program.

1.3 Thesis Outline

One of the most essential aspects of human-computer interaction is the use of vision-based technologies, such as hand gesture recognition. Sensor displays and touch screens, keyboards and mice are still used for basic human-computer interaction in certain projects today; in other situations the rapid development of technology and software has necessitated the creation of more advanced technologies. In order to aid those with impairments, this project focuses on the issue of using only one camera on a computer and recognition algorithms.

Gesture represents physical conduct and emotional expressiveness. It consists of both body and hand gestures. It may be divided into two categories: static gesture and dynamic gesture [11]. The posture of the body or the movement of the hand signifies a signal for the individual. The use of gesture as a means of com-

munication between computer and human is possible [10]. It differs significantly from conventional hardware-based technologies and can achieve human-computer interaction via gesture recognition. The activity of the user is determined by the recognition of the gesture or movement of the bodily parts. Currently, hand gesture recognition projects utilize sensors, Ovasion, Raspberry PI, and other technologies. These approaches promise more equipment and gadgets, but their use in everyday life is awkward. First, the hand area is extracted from the original photos captured by the camera input device. Then, certain types of characteristics are retrieved to characterize hand motions, and the recognition of hand gestures is achieved by comparing the feature data. The input devices that provide the original picture data include a standard camera. The hand region is detected and segmented based on the skin color's sensitivity to lighting conditions and feature points. Finding finger indices and angles was one of the initial stages in hand recognition. The MediaPipe algorithm aids in detecting hands and locating landmarks, hence illustrating handedness. The best solution is the algorithm with the most training.

2. Preliminaries

There are some mathematical formulas used in algorithms. Every ML, AI, NN algorithms work with mathematical definition, theorem or formulas. They include mathematics. For example, SVM algorithm works wit Euclidean norm of complex numbers and finding destination.

Definition 2.1. Similarly, the distance of a hyperplane equation:

$$(w^T)(x) + b = 0$$

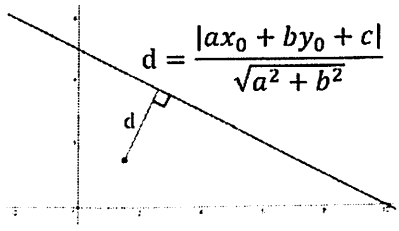
from a given point vector (x0) can be easily written as :

$$d_H(\phi(x_0)) = \frac{|w^T(\phi(x_0)) + b|}{||w||_2}$$

Euclidean norm of complex numbers:

$$||w|| = \sqrt{w^2 + w^2 + w^2 + w^2 + n}$$

So, the plane destination or distance measure is:



Definition 2.2. Lagrange multiplier. Below x is the original primal variable and to minimize the function f under a set of constraints given by g, and rewriting for a new set of variables called Lagrangian multipliers.

Basic Primal Form :	*Rewritten as Lagrange Multiplier*
$min_x\, f(x)\ ;\ g_i(x) \leq 0$	$L(x\ ,\ \{\lambda_i\}) = f(x) + \sum_{i=1} \lambda_i\, g_i(x)\ ;\ \lambda_i \geq 0$

Solving the primal variables by differentiating the unconstrained Lagrange.

$$\frac{\partial L}{\partial x} = 0 \ \twoheadrightarrow\ x = h(\{\lambda_i\})$$

And lastly substituting back in the Lagrange equation and rewriting the constraints

$$L(\{\lambda_i\}) = f(h(\lambda_i)) + \sum_{i=1}^{n} \lambda_i g_i(h(\{\lambda_i\})$$

$$max_{\{\lambda_i\} \geq 0} min_x\, L\,(x\ \{\lambda_i\})$$

As the KNN is a supervised learning, there is the largest probability formula which it uses.

Definition 2.3. Chain rule is a formula that expresses the derivative of the composition of two differentiable functions f and g in terms of the derivatives of f and g. More precisely, if

$$h = f \circ gh = f \circ g$$

is the function such that

$$h(x) = f(g(x))h(x) = f(g(x))$$

for every x, then the chain rule is, in Lagrange's notation,

$$h'(x) = f'(g(x))g'(x).$$

or, equivalently,

$$h' = (f \circ g)' = (f' \circ g) \cdot g'.h' = (f \circ g)' = (f' \circ g) \cdot g'.$$

The chain rule may also be expressed in Leibniz's notation. If a variable z depends on the variable y, which itself depends on the variable x (that is, y and z are dependent variables), then z depends on x as well, via the intermediate variable y. In this case, the chain rule is expressed as

$$\frac{dz}{dx} = \frac{dz}{dy} \cdot \frac{dy}{dx}, \frac{dz}{dx} = \frac{dz}{dy} \cdot \frac{dy}{dx},$$

and

$$\left.\frac{dz}{dx}\right|_x = \left.\frac{dz}{dy}\right|_{y(x)} \cdot \left.\frac{dy}{dx}\right|_x, \left.\frac{dz}{dx}\right|_x = \left.\frac{dz}{dy}\right|_{y(x)} \cdot \left.\frac{dy}{dx}\right|_x,$$

for indicating at which points the derivatives have to be evaluated.

Definition 2.4. Mean squared error is chosen as the loss function for regression problems and cross entropy for classification problems. Let's take a regression problem and its loss function be mean squared error, which squares the difference between actual (y) and predicted value (y).

$$MSE_i = (y_i - \hat{y}_i)^2$$

Definition 2.5. Goal is to discover a function h:X→Y so that having an unknown observation x, h(x) can positively predict the identical output y

In the classification problem, the K-nearest neighbor algorithm essentially said that for a given value of K algorithm will find the K nearest neighbor of unseen data point and then it will assign the class to unseen data point by having the class which has the highest number of data points out of all classes of K neighbors.

For distance metrics, it will use the Euclidean metric.

$$d(x, x') = \sqrt{(x_1 - x_1')^2 + \ldots + (x_n - x_n')^2}$$

Finally, the input x gets assigned to the class with the largest probability.

$$P(y = j|X = x) = \frac{1}{K} \sum_{i \in A} I(y^{(i)} = j)$$

Definition 2.6. Precision (also called positive predictive value) is the fraction of relevant instances among the retrieved instances, while recall (also known as sensitivity) is the fraction of relevant instances that were retrieved. Both precision and recall are therefore based on relevance.

$$\text{Precision} = \frac{True\ Positive}{True\ Positive + False\ Positive}$$

$$\text{Recall} = \frac{True\ Positive}{True\ Positive + False\ Negative}$$

Definition 2.7. The balance value between precision and a recall called a F1 score.

$$F1 = 2 \times \frac{Precision \cdot Recall}{Precision + Recall}$$

3. Programming Environment. Design Architecture

Hand gesture recognition and command execution are common features in a wide range of electronic devices. For example, my camera, niose device, and a smart TV (TV) with wifi, interference device reflect the provided signal. For example, the smart light bulb contains a noise adaptor and can detect whether or not a signal is on by reflecting noise back. In order to use them, you need specific identification adapters and devices, which aren't cheap. Cameras are the only unique equipment needed for this project, which can be found on most laptops. This project is focused on human-computer interaction since the machine will detect and carry out a manual instruction thanks to a software, a trained model. Interaction between humans and computers (users) is the subject of Human-Computer Interaction (HCI), which is a study of computer technology design and usage. For human-computer interaction (HCI), researchers look at how people interact with computers and build technologies that allow for new ways of interacting with computers.

Human–computer interaction (HCI) is a nexus of multiple disciplines, including computer science, psychology, design, and media studies. "Human-computer interaction" is a concept coined by Stuart K. Card, Allen Newell, and Thomas P. Moran in 1983. Carlisle was the first to employ it in 1975. According to the phrase, multi-use computing means that unlike other tools with particular and restricted purposes, open interaction between the user and the computer is commonplace in multi-use computing. As a theoretical analogue for human-to-human interaction, the notion of conversation treats human-machine interaction with human-to-human contact [18].

With the OpenCV library in PyCharm's environment, this project is created

in Python and provides assistance to Python function writers. Integrated development environment for Python programming language, PyCharm, is available. Debugging and unit testing frameworks are supported as well as code analysis tools. Based on the IntelliJ IDEA platform, JetBrains has created PyCharm.

Python is a general-purpose, high-level programming language that may be interpreted. The use of extensive indentation is a key part of its design philosophy, which prioritizes code readability. [20].

OpenCV is an open-source computer software library for computer vision. OpenCV is aimed to offer a standard infrastructure for computer vision applications and to facilitate the usage of machine sensing in commercial goods. As a BSD-licensed project, OpenCV enables businesses to use and alter the code cites18 with relative ease. OpenCV is able to work with images, photographs, and cameras, among other things. The base architecture of the project are represented by(Figure 3.1):

- Camera part
- Detection part
- Recognition part
- Interface part

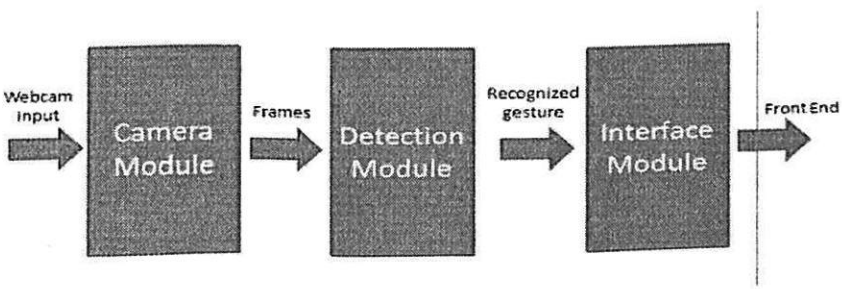


Figure 3.1: Basic Project Architecture

3.0.1 Camera part

This module is responsible for connecting and capturing input from the various types of image detectors before sending it to the detection module in the form of frames for processing. Common input capture technologies include data gloves, hand belts, and cameras. In our system, both static and dynamic motions are recognized using the built-in camera, which is cost-effective. The system has the capability to accept input from a USB camera, however the user would have to

incur additional costs. The acquired picture frames are in the form of a video. There is a code to play the video, but its value is 0, indicating real-time video and camera play.

3.0.2 Detection part

This module performs image processing operations. The output of the camera module is subjected to a number of image processing techniques, including color conversion, noise reduction, and thresholding, before contour extraction. If the picture has flaws, the gesture is identified beneath the convex faults. If there are no faults, the pictures are categorised for gesture detection using the Haar cascade. In the event of dynamic gestures, the detecting module performs the aforementioned actions: If Microsoft PowerPoint is loaded with the slideshow feature enabled and the webcam detects continuous palm movement for five frames, dynamic gesture scanning will be recognized.

3.0.3 Recognition part

This module is the project's primary component. There are several ML and NN training and testing algorithms. Training a model entails just learning appropriate values for all weights and biases from annotated examples. There are both supervised and unsupervised ways of learning. In supervised learning, a machine learning algorithm constructs a model by analyzing a large number of instances and searching for a model that minimizes loss; this method is known as empiric risk reduction. Unsupervised learning is a technique for machine learning in which the user is not required to monitor the model. Instead, it enables the model to discover previously unnoticed patterns and information on its own. It mostly deals with unlabeled data. This is supervised learning since it includes attributes and markings in addition to the gesture type.

3.0.4 Interface part

Hand gestures are mapped to their corresponding actions in this module. In addition, these activities are sent on to the appropriate application There are three windows on the front of the house. The video input from the camera is displayed

in the first window, along with the name of the gesture that was identified. In the second window, you can see the picture contours as they were originally drawn up in the first. The third window displays a smoothed-out interpretation of the image. Because the user is made aware of background irregularities that may impede system access, he or she can adjust the camera web on your laptop or desktop to avoid these issues. Better results will follow as a result of this.

CV devised the fundamental concept for this project. For instance, it aids in detecting hands, playing frames, video frames, dividing video into frames, and locating hand signals. CV is a subfield of AI that enables computers and systems to extract meaningful data from digital photos, videos, and other visual input - and then take action or make suggestions based on that data. If AI enables computers to think, then computer vision enables them to see, observe, and comprehend [16].

CV provides trained library models to developers. For example, CV has produced facial recognition library collections. For instance, MediaPipe was first created for facial identification by identifying facial landmarks. The concept of hand discovery emerged following the detection of facial landmarks, face meshes, etc.

In addition, they utilize ML techniques to train and evaluate models. Why is a trained model necessary for the project? Because there is a dataset previously collected by researchers, but there is no ready-made system that can distinguish two-finger orders. For this reason, a trained ML algorithm is required to trigger this program. Numerous algorithms and their findings are utilized. AI is intelligence exhibited by machines, as opposed to the innate intelligence possessed by humans and other animals. AI exploration is described as the study of intelligent agents, which refers to any system that is wary of its surroundings and modifies its behavior to increase its chances of attaining its goals.

Machine learning is the study of computer programs that can improve themselves automatically via the use of experience and data. It is a component of artificial intelligence. In order to generate predictions or choices without being explicitly trained to do so, machine learning algorithms create a model using training data and sample data. In many operations, such as medicine, mail filtering, speech recognition, and computer vision, when it is difficult or impossible to design conventional algorithms to execute these effects on important tasks, machine learning algorithms are utilized. [19].

Are there more details regarding the dataset and its nature? A dataset, often known as a data set, is a collection of data. In the case of tabular data, a data set corresponds to one or more database tables, where each column represents a particular variable and each row corresponds to a single record from the data set in question. Data is a crucial component of every AI model and, in essence, the primary basis for the current explosion in popularity of machine learning. Due to the availability of data, scalable machine learning (ML) algorithms are now feasible as genuine products that may bring value to a business, rather than as byproducts of its fundamental activities. Client purchases, product appeal, and seasonality of customer flows have always played a significant part in the development of a corporation. Nonetheless, with the development of machine learning, it is vital to include this data into databases. You can examine trends and hidden patterns and make judgments based on your own data set if you have sufficient data volume. However, despite its seeming simplicity, working with data is more difficult since, first and foremost, it involves careful treatment of the data you have, from the aim of utilizing the dataset to the preparation of the raw data. [17].

4. Data Collection

Hand detection is a very difficult process. As a first step, a palm detector is trained instead of a hand detector, as estimating the bounding boxes of rigid objects like hands with articulated fingers is substantially easier than identifying palms and fists. The non-maximum suppression approach also works well for two-hand self-occlusion scenarios, such as handshakes, because palms are smaller objects [1]. As a result, the number of anchors needed to simulate palms may be reduced by a factor of 3-5 by adopting square bounding boxes (anchors in ML language). A second method involves the employment of an extractor of encoder-decoder features in order to make even little objects in a scene aware of their context (similar to the RetinaNet approach). [1].

There are gesture types what the researches need:

- down
- up
- right
- left
- zoom in
- zoom out

There are many lists of frames collected manually to classify the gesture signs with the following signs: zoom in, zoom out, right, left. Each type of hand gesture is found using only 2 points from the hand fingers (see Figure 4.1). There are also some faster data collection methods like to cut video by 4 frames, and collect key points and keep them in the classified folders. There are 4 frames are accessible for reading and setting landmarks by MediaPipe.

There is a requirement that the hand recognition model should be read on a horizontal surface and the distance between hand and camera should be 1m. The hand tracking model first finds a palm and then draws landmarks of fingers where

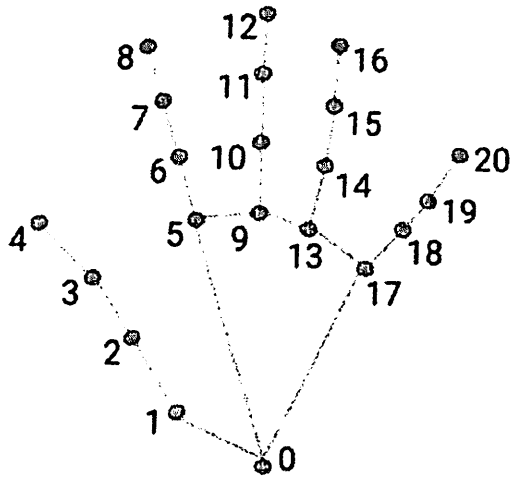


Figure 4.1: Hand Landmarks

MediaPipe recognition algorithms best fits for. A palm detector that operates on a whole picture input and generates palm bounding boxes for localisation. There are many algorithms utilized to recognize the hand and identify its landmarks: initially, a video frame is extracted. Then combine photos, generate hand presence, and identify handedness dots and lines. (Figure 4.3). The main algorithm firstly finds the palm, then by palm finds finger landmarks. There are five points on the palm, four of which correspond to finger positions and one of which is located at the palm's base. Every finger has 3 landmarks: top side, middle, bottom side (Figure 4.1). (Figure ??fig:p16) illustrates how the hand landmark model acts on bounding boxes to produce key-point localization of 21 3D hand coordinates via regression techniques that pass to coordinate prediction. The model develops a consistent hand position representation that is accurate even when partial hand visibility is available [2].

```
def extractImages(pathIn, pathOut):
    count = 0
    vidcap = cv2.VideoCapture(pathIn)
    success,image = vidcap.read()
    success = True
    while success:
        vidcap.set(cv2.CAP_PROP_POS_MSEC,(count*1000))
        success,image = vidcap.read()
        if(not success):
            break
```

Figure 4.2: Code for extract frames from video.

In this project to recognise hand gestures, there needed video frames or real-time frames. To get frames in video there was a code for dividing video into frames. The normal frames number was four, because the every video time is

different and some frames were not read. So, the optimal number was 4. The first type dataset and second type dataset used these method to extract frames from video (Figure 4.2).

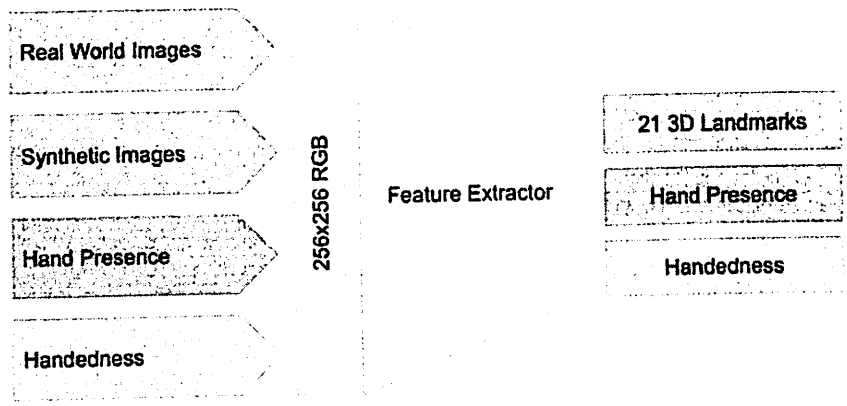


Figure 4.3: Architecture of research hand landmark model. The model has three outputs sharing a feature extractor. Each head is trained by correspondent datasets marked in the same color [8].

Overall there are several trained models perform together : A palm detector model (called BlazePalm) that operates on the full image and returns an oriented hand bounding box. A hand landmark model that runs on the cropped image region defined by the palm detector and returns high fidelity 2.5-3D hand keypoints. A gesture recognizer for classification [2].

Each frame has 21 numbers of landmarks in x, y, z position. So, in general there are 63 points. There are 252 point data for one gesture and only one ended video motion frames, it has got from 63*4 points. All these data points help to train a model for hand gesture recognition. For example, for "zoom in gesture" the program will need more x, y, values from 2 points and collect them into a one dimensional array and classify them. Also, no need to keep 252 points in the dataset, the researchers try to make model simple and fast using only 4 x 4 points, and classification integers (0 - zoom in, 1 - zoom out).

4.1 First Type Dataset

There are many videos of hand gestures: zoom in, zoom out, left, right, down, up. That helps us to collect frames automatically, but the number of frames are different. In general the frame numbers of video was 4. The program divided

it into 4 and read each frame of gesture to get landmarks of hand. Each x, y position of each gesture is written in key-points CSV file only with "x"-position, "y"-position. For this experiment there are using only 2 types of classes: 1st type - "zoom in" gesture; 2nd type - "zoom out" gesture. The representation of the dataset is in the following Figure 4.4.

y	x1	x2	x3	x4	x5	x6	x7	x8
0	0.4656699896	0.6679420471	1.43E-07	0.435313791	0.6894289851	-0.0039743799	0.4129841626	0.7382674813
0	0.4459566474	0.6216173172	2.00E-07	0.4160716236	0.6516026258	0.000133170655	0.4027231932	0.7048466206
0	0.41309008	0.5858129859	2.03E-07	0.387958765	0.6024846435	-0.00290606916	0.373493731	0.6454461217
0	0.248724103	0.6528689861	-1.19E-07	0.2498227656	0.6696017981	-0.01805447415	0.2429991961	0.6888879538
0	0.4006622732	0.6006212831	1.94E-07	0.3765863478	0.6155281663	-0.00453025661	0.3617106378	0.65635252
0	0.3991965353	0.5903832316	2.06E-07	0.3756053746	0.6051010489	-0.00171740155	0.3629561365	0.6456369758
0	0.2392995358	0.6568716168	-6.98E-08	0.2463079393	0.666030705	-0.02198293246	0.2467184663	0.6840019226
0	0.2413506359	0.6552639008	-1.19E-07	0.2462570965	0.6599337459	-0.02310369164	0.2441249639	0.6749482155
0	0.2508418858	0.6505835056	-3.16E-08	0.2481936365	0.6840232015	-0.00415551103	0.2390512079	0.7074927688
0	0.2400315851	0.663644433	-4.96E-08	0.2439057529	0.6703468561	-0.01841712743	0.2459716201	0.6835813522
0	0.2399062961	0.65391922	-1.55E-07	0.2463133335	0.6696051955	-0.02165785059	0.2429064214	0.6902439594
0	0.4205651879	0.577627182	1.73E-07	0.3967759013	0.6033682823	-0.00077187758	0.3858606815	0.6457927227
1	0.4543712437	0.6770622134	2.59E-07	0.426425457	0.699211657	-0.00327548058	0.4108588398	0.7530849576
1	0.4180338681	0.6049672365	2.59E-07	0.3978458047	0.6218616366	-0.0016572166	0.3859153986	0.6621724367
1	0.248724103	0.6528689861	-1.19E-07	0.2498227656	0.6696017981	-0.01805447415	0.2429991961	0.6888879538
1	0.4006622732	0.6006212831	1.94E-07	0.3765863478	0.6155281663	-0.00453025661	0.3617106378	0.65635252
1	0.390517503	0.5929225683	1.79E-07	0.3684381247	0.6050209403	-0.00099005107	0.3546185195	0.6417354941
1	0.3978922963	0.5676781535	1.47E-07	0.4085707664	0.5964772701	-0.01635865867	0.4098788202	0.6342664361
1	0.2508418858	0.6505835056	-3.16E-08	0.2481936365	0.6840232015	-0.00415551103	0.2390512079	0.7074927688

Figure 4.4: Example of 0-zoom in and 1-zoom out type gesture dataset in x positions dataset file.

There are all x, y, z, values are in one file, not divided. This dataset was the first experiment. The researchers have experimented and analyzed it with SVM and KNN classifiers. The accuracy was about 0.4 and 0.2, what means the weak result (Figure 4.5). So, this analyze claimed make research for another dataset. The second type dataset.

```
Testing SVC with 9 tuples...

              precision    recall  f1-score   support

           0       0.00      0.00      0.00         2
           1       0.50      0.67      0.57         3
           2       1.00      0.50      0.67         4

    accuracy                           0.44         9
   macro avg       0.50      0.39      0.41         9
weighted avg       0.61      0.44      0.49         9

0.4444444444444444
```

```
              precision    recall  f1-score   support

    accuracy                           0.22         9
   macro avg       0.13      0.22      0.17         9
weighted avg       0.13      0.22      0.17         9
```

Figure 4.5: SVM and KNN algorithms result.

4.2 Second Type Dataset

This second type of dataset was made after bad results of first typed dataset. For this dataset firstly user should take a video. Then divide it into 4 frames and fill out the dataset by hand the results of hand detection, hand recognition. It takes more time, even days. In this dataset x values were in one file, y values divided into another file, z dataset were in another file. In general, there are three files with almost 3000 values. You can see the x dataset file in the figure below 4.6, where all the x values of landmarks from of the frame. So, from type of dataset the researches have got accuracy about 0.8 from KNN and SVM algorithms. Even if the values are divided to each file, the result was almost the same. Because in such case in the program the data frame will be one, with combined values, because in the program the dataframe cannot be more. If you have question why, as mentioned before the hand gesture types have values as integer and every x, y, z values are of one type hand gesture. That's why, every files' values have combined by gesture types, what was given by researcher.

55 0.232726544141769

A	B	C	D	E	F	G	H	I	J	K	L	M	
0	0.4543712437	0.426425457	0.4108588398	0.4162932336	0.4239361286	0.3988850713	0.3946071267	0.3932300806	0.3923881352	0.4230559468	0.4163655043	0.4128119948	
0	0.4656699896	0.435313791	0.4129841626	0.4078397155	0.408931911	0.4004037082	0.392552942	0.3901266754	0.3889143467	0.4231524467	0.4146359861	0.4102204144	0
0	0.4459566474	0.4160716236	0.4027231932	0.407250613	0.4160924852	0.3974172771	0.3919265568	0.3894048035	0.3872882128	0.4194025397	0.4135022759	0.4088102247	
0	0.41309008	0.387956765	0.373493731	0.3720935583	0.3740748465	0.3671050072	0.3804907691	0.3578589261	0.3562986851	0.3864867389	0.378331989	0.372413367	
0	0.248724103	0.2498227656	0.2429991961	0.232699424	0.2230489403	0.2099872082	0.1945504248	0.2043594718	0.2152679414	0.2025433183	0.1826719493	0.1947725713	
0	0.4006622732	0.3765863478	0.3617106378	0.3585689068	0.361392498	0.360419184	0.3545922637	0.353125453	0.3527674377	0.3783597946	0.3709584773	0.3661530018	
0	0.3991965353	0.3756053746	0.3629561365	0.3623859286	0.3865465415	0.3563411534	0.3492245972	0.3466182947	0.3454945083	0.3741928339	0.3659413159	0.3609440327	
0	0.2392995358	0.2463079393	0.2467184663	0.2397360653	0.230714038	0.2169297785	0.2088960856	0.2055130601	0.204285875	0.2062334567	0.1976872385	0.1936940988	
0	0.2413506359	0.2462570965	0.2441249639	0.2376695424	0.2294101417	0.2100898325	0.1998802274	0.2095999122	0.2213263661	0.2019999176	0.1902374774	0.2027766258	
0	0.2508418858	0.2481936365	0.2390512079	0.2277115881	0.2170853764	0.2126885504	0.1934648156	0.1883651018	0.1871967316	0.2083649337	0.1888156533	0.1884932369	
0	0.2400315851	0.2439057529	0.2459716201	0.2467508763	0.2458229512	0.2104065418	0.2030847371	0.212055698	0.2223683628	0.2016068399	0.182698369	0.1895389706	
0	0.2399062961	0.2463133335	0.2429064214	0.2340668887	0.2261386365	0.2032822669	0.2051753551	0.2158907205	0.2241621166	0.1933014393	0.1900203526	0.2026115358	
0	0.4205651879	0.3967759013	0.3858606815	0.3879730999	0.3944121003	0.3826110383	0.379352212	0.3789899647	0.3795287907	0.4003779292	0.3969768584	0.3940210938	
1	0.4462751746	0.4238047004	0.409937799	0.4092714489	0.4133048654	0.4019307196	0.394480288	0.3909564018	0.3885523975	0.4194789529	0.4107421339	0.4047159255	
1	0.4412176013	0.418343842	0.404360652	0.4055868089	0.4122733474	0.3981377482	0.3918523788	0.389482528	0.3882677555	0.4162988067	0.4086665511	0.4043217003	
1	0.4294313788	0.4061459601	0.3942221999	0.3954281211	0.4007623494	0.3875432909	0.3829304278	0.3810124993	0.3796889518	0.4054305553	0.3992821276	0.3953082263	
1	0.4329059124	0.4095700085	0.3957645297	0.3932023048	0.3955094516	0.3930574059	0.3874101937	0.3841297925	0.3814509213	0.410531044	0.4032924175	0.397562623	
1	0.4139830172	0.3919493854	0.3795950413	0.3812508881	0.3889667392	0.37084499	0.3626160324	0.3579287529	0.3541684151	0.3852522969	0.3752734363	0.3673521578	
1	0.4280828705	0.4069594741	0.3929995298	0.3921847045	0.3962005973	0.3859568834	0.3763295412	0.3704967499	0.3657648563	0.401380988	0.3896047473	0.3813711405	
1	0.4214346707	0.3881760144	0.384980619	0.3859823644	0.391720295	0.3779471219	0.3698312342	0.364787668	0.3610219359	0.3935884237	0.3821129501	0.3739138842	
1	0.4478681359	0.4254416823	0.4067575336	0.3989507258	0.3984848857	0.3977075815	0.3785659969	0.3668899238	0.3560377896	0.4079839289	0.3844339848	0.3686499894	
1	0.4076613188	0.3844555318	0.3722045124	0.3739978969	0.3806601167	0.3678596318	0.3612894118	0.3576439023	0.3544160128	0.3838760853	0.3758277893	0.3689171374	
1	0.3994223475	0.3793494701	0.3683522642	0.3718351126	0.3805373311	0.3635037839	0.3580493629	0.3552515507	0.3532904685	0.3785718381	0.3728233576	0.3678480685	
1	0.399474144	0.3798441589	0.3655359149	0.3625349998	0.3656280041	0.3635920882	0.3568830788	0.3535649776	0.3512517214	0.3771697581	0.3695767224	0.3640348911	
1	0.404147923	0.3820161223	0.368948102	0.3678408861	0.372590661	0.3650845885	0.357239306	0.3525720537	0.3489371836	0.380528599	0.3722440302	0.3642877042	
1	0.4059427381	0.3837258518	0.3719451725	0.3720921874	0.3775572777	0.3671790063	0.3585220575	0.3540593982	0.3512124121	0.3818826973	0.3727217317	0.3654271662	
1	0.4110940695	0.3905284405	0.3772624731	0.3748688698	0.3778574765	0.3725649714	0.3624704778	0.3561358154	0.3513247073	0.3662960339	0.3747807741	0.3652775586	
1	0.4126040637	0.3912768662	0.3799184859	0.3830299973	0.3913843036	0.3739543855	0.3656657934	0.3609845936	0.3577570319	0.3877792656	0.3778961301	0.3696080446	

Figure 4.6: Example of x-dataset.

4.3 Third Type Dataset

Another type of data collection is to only segment 2 fingers from the hand. Once the target classes are found with two fingers, it can modify and speed up the

algorithm using fewer features. However, the MediaPipe framework offers all 25 points at your fingertips, and the complexity of finding points remains the same. Otherwise, the memory of the dataset will be significantly reduced by about 12 times. Data set collection is simplified and ensures that the dataset fits well to certain classes. The data collection example is done by selecting the layer for the gesture and pressing the 'Space' button while clicking on the frame of the hand gesture, and the program will detect the hand and find the landmarks of the hand gesture. The frames are converted to matrix points and the values are stored in a temporary variable. Then, after iterations, the values are passed into the dataset file. The program includes only recognized hand gesture values of the class. Thus, the dataset is updated every time the user shows a new hand gesture. Models will also be recycled. As the data increases, the accuracy of the trained model also increases. This method of data collection is very comfortable and efficient, because the program reads the frame in real time and automatically divides it into frames. The detection and recognition of algorithms will not be manual. The user will do nothing else and the dataset and model will be trained and updated itself. It doesn't take much longer than collecting the first and second data sets. The idea of using this kind of dataset collection method came about after research that needed more data set values.

4.4 Feature scalability. Data normalization and pre-processing.

Normalizing the range of independent variables or qualities of data is accomplished by feature scaling. In machine learning, object scaling is one of the most critical data pre-processing procedures. To increase the efficiency of machine learning and deep learning models, algorithms that calculate distances between features are more prone to numeric values, especially if the input is not scaled.

Normalization is the most frequent scaling approach, yet it may also be the most perplexing.

Example. Features are normalized or scaled using Min-Max normalization or scaling. The following formula is used to determine new points::

This sets the range to be between 0 and 1, or occasionally between 1 and 1. The transformation brings the n-dimensional data down to an n-dimensional unit

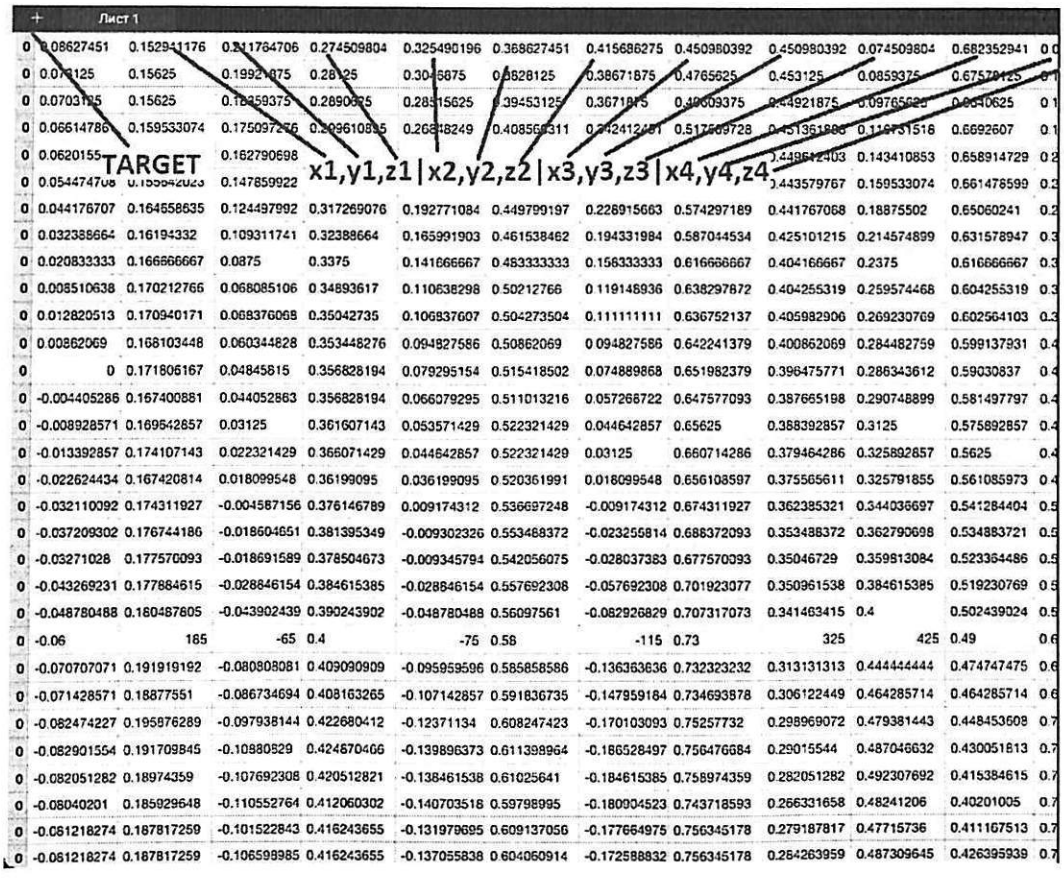


Figure 4.7: Example of dataset-3.

```
X_new = (X - X_min)/(X_max - X_min)
```

Figure 4.8: Normalization

hypercube, which is a geometric representation of the reduction. Normalization is most beneficial in situations in which there are no exceptions since it is unable to deal with them.

The process of transforming characteristics into a Z-score involves first removing the mean from the total, then dividing that number by the standard deviation. The Z-score is the name that most people give to this number.

```
X_new = (X - mean)/Std
```

Figure 4.9: Standardization

When the data follows a Gaussian distribution, normalization might be beneficial. However, there is no guarantee that this is the case. In terms of geometry, it moves the mean vector data from the original to the origin, and it either replaces

or extends the points depending on whether or not the accompanying standard deviation is 1. It is clear to us that all we need to do is alter the value. In a normal normal distribution, both the mean and the standard deviation are always normal, which means that the form of the distribution is unaffected by either value.

The absence of a predetermined range for the modified characteristics ensures that outliers have no impact on the normalization process.

As a result of the photographs having varying scales, the normalizing approach is utilized in this piece of writing. For the target classes, the range of the x, y, and z coordinates is somewhat far. The normalization procedures cut the amount of time needed for model training significantly.

The following is an illustration of the final output of the program that was discussed before, which can be seen in the screenshot.

```
[[0.11785113 0.1767767  0.29462783 0.35355339 0.41247896 0.23570226
  0.47140452 0.41247896 0.35355339]]
```

Figure 4.10: Normalization example

5. Algorithms

The dataset were collected from three typed methods and it has hot from video and real-time case. Every video divided into 4 frames. There where hand detection and hand recognition algorithms.

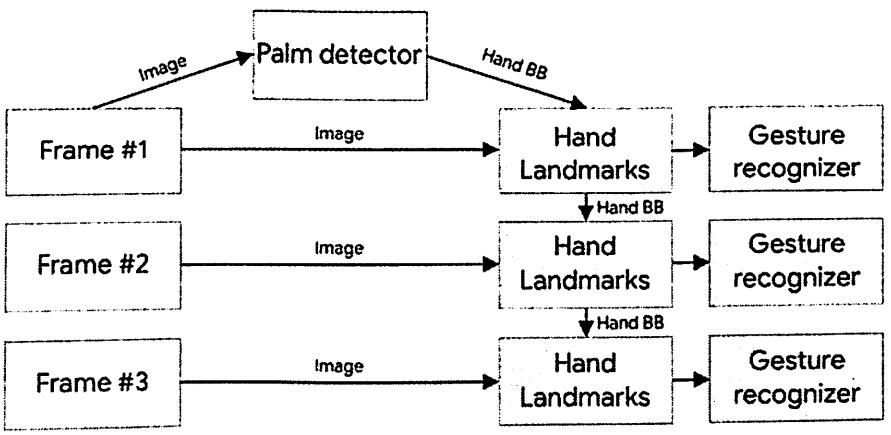


Figure 5.1: Main Algorithm of Hand Gesture Recognition

5.1 Image Processing

5.1.1 Noise removal and Image smoothening

Frame in RGB color, converted to BGR color frame. It is cropped to the dimensions of 300 pixels by 300 pixels in the RGB color space from the original image. Afterwards, it's turned into a grayscale picture. Cropping and grayscale conversion of an RGB picture as an input Webcam images are often plagued with noise, which may be characterized as a random change in brightness or color information. This noise should be eliminated from the image since it is unappealing. A Gaussian filter is used in this process. With each point in the input array,

the Gaussian kernel is rotated by the same amount. The output array is then constructed by adding the elements.

Playing real-time video in a live frame window is as simple as the code shown below. (Figure 5.2).

```
    min_tracking_confidence=0.5) as hands:
while(True):
    ret, image = cap.read()
    if not ret:  # Image was not successfully read!
        print('\rNo image!  Is a webcam available?', '', end='')
        continue

    raw_frame = copy.deepcopy(image)

    # frame_count += 1

    image = cv2.flip(image, 1)
    image_height, image_width, _ = image.shape
    hand_movements.draw_mouse_rectangle(image)
```

Figure 5.2: Code For Reading The Real Time Frame.

5.1.2 Thresholding

Thresholding is a straightforward segmentation technique. To convert a grayscale image to a binary image, a threshold must be applied. Each pixel's intensity (I) value is compared to a preset threshold value using the thresholding technique (T). If it's not, a white pixel is used in its place. We utilized a threshold value (T) of 127 to categorize grayscale picture pixel intensities. If a pixel in an image exceeds the threshold value, the maximum pixel value of 255 is utilized. Inverted binary thresholds and Otsu's thresholds are two types of thresholds that may be used in the implementation. There are white pixels set at maxVal when a binary threshold is inverted. Otsu's approach was handed down to us by Nobuyuki Otsu. Using clustering, we may derive an image threshold. In the case of a bimodal picture, the histogram has two peaks, and Otsu binarization uses this information to automatically determine the threshold value. See if you can locate a cutoff point in Otsu's approach that has the least intradiametric variance (variance within the class).

It is possible to repeatedly calculate the probability and the average of the class. This might result in a useful algorithm. A threshold was applied to the binary picture before the edges were discovered in order to improve accuracy.

5.1.3 Contour Extraction

There are many applications for contours in image processing. To distinguish the hand from its surroundings, our frameworks relied on contours. Curves that link points of the same hue in a straight line are called contour lines. OpenCV's initial step in detecting a white item against a dark backdrop is determining the contour. As a result, the threshold was created using an inverted binary threshold. Afterward, the contours may be utilized to design any form if the boundary points have been established.

5.2 MediaPipe Library

MediaPipe Hands is a high-fidelity solution for tracking fingers and hands. It infers 21 3D hand landmarks from a single picture using machine learning. While sophisticated techniques of today rely mostly on powerful desktop systems for inference, our method achieves real-time performance on smart phones and may even be used by a large number of individuals.

MediaPipe Hands employs a machine learning (ML) pipeline comprised of many models operating in concert: a palm identification model that operates on the entire picture and provides a hand-directed bounding box. The palm landmark model acts on the picture region recognized by the palm detector and delivers 3D palm markers with excellent fidelity.

The one-shot detector model tailored for real-time mobile application detects the starting locations of the hand in a manner similar to the face recognition model in MediaPipe Face Mesh. Hand identification is a difficult task: both the light model and the complete model must function on varied hand sizes at a big scale (20x) to the image frame and be able to recognize hands that are stuck and self-clogged. While faces exhibit high-contrast patterns, such as in the eye and mouth regions, the absence of such patterns on hands makes their recognition based on feature alone reliable. Vision is rather challenging. Rather, supplying extra information, such as arm, body, or human characteristics, facilitates the identification of accurate hand position.

This strategy addresses the aforementioned obstacles using a variety of ways. It is significantly simpler to estimate the bounding boxes of hard objects such as

palms and fists than it is to recognize the hand with knuckles and fingers. In addition, because the palm is a smaller item, the non-decimal suppression approach is also effective for two-handed self-matching, such as handshakes. Alternately, palms can be represented using square bounding boxes (anchors in ML parlance) while disregarding other aspect ratios, hence lowering the number of anchors by a factor of 35. Secondly, even for small objects, an encoder-decoder feature extractor is employed to improve context awareness (similar to the RetinaNet approach). Minimize the loss of concentration during training owing to large-scale variation in order to accommodate a large number of anchors.

```
results = hands.process(cv2.cvtColor(image, cv2.COLOR_BGR2RGB))

# Print handedness and draw hand landmarks on the image.
print('Handedness:', results.multi_handedness)
if not results.multi_hand_landmarks:
  continue
image_height, image_width, _ = image.shape
annotated_image = image.copy()
```

Figure 5.3: Code For Draw Handedness and Finding Landmarks

After the palm has been found in the whole image, our posterior palm landmark model finds the key points of the 21 3D coordinates of the knuckles in the hand regions found by regression. This is a direct match prediction. The internal hand posture learning approach is suitable with partially visible and self-occlusive hands and is incredibly resilient.

To obtain ground truth data, manually mark 30K photos of the actual world with 21 3D locations. To better understand various hand postures and give extra monitoring of the nature of the hand shape, a high-quality synthetic hand model should be created across a variety of backdrops and lighting conditions. Figure 5.4 illustrates this process. AI Infrastructure for Hand Tracking

- BlazePalm is a form of palm detector that operates on the whole picture and provides an orientated hand bounding box.

- A palm marker model operates on the picture region cut by the palm detector and returns 3D palm markers with excellent fidelity.

A gesture recognizer sorts pre-calculated keypoint profiles into a distinct set of motions.

This design resembles that of our newly revealed facet mesh ML channel and other channels used for camouflage estimates. Providing accurately cropped palm

pictures to the palm landmark model eliminates the requirement for data augmentation (e.g., rotations, translations, and scales) and enables the network to maximize its capacity to coordinate the accuracy of predictions.

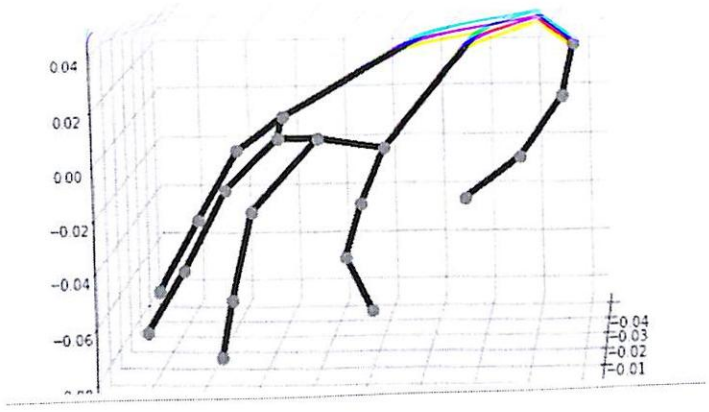


Figure 5.4: Frame Handedness and Hand Landmarks

5.3 k-Nearest Neighbor Algorithm

K-Nearest Neighbor is a basic method for supervised learning.

The KNN method presupposes similarity between incoming data instances and old locations. The model is able to develop a consistent internal hand posture representation and is resilient even when the hands are only partially visible or when they self-occlude.

To gather ground truth data, 30,000 photos of the actual environment were manually labeled with 21 3D coordinates. Also produce a high-quality synthetic hand model over various backdrops, translate it to the correct 3D coordinates, and depict handedness (Figure 5.4).

ML piplenes to hand tracking.

- BlazePalm is a palm detector model that acts on the whole picture and provides an orientated hand bounding box.

- A hand landmark model that acts on the picture region cut by the palm detector and delivers 3D hand keypoints with excellent accuracy.

- A gesture recognizer that categorizes the calculated keypoint configuration into a distinct set of gestures.

- This architecture is comparable to the one utilized by our newly disclosed face mesh ML pipeline and by other posture estimation systems. Providing the

palm picture that has been precisely cropped to the hand landmark model greatly minimizes the requirement for data augmentation (e.g. rotations, translations, and scaling) and enables the network to devote the majority of its resources on the accuracy of coordinate prediction. Choose the category that best corresponds to the given options.

The KNN algorithm saves all available data and scores a new data point based on its similarity to other data points. This implies that fresh data may be easily categorized into a suitable category using the K NN method.

The KNN method may be used for classification and regression, however it is primarily employed for classification tasks.

The KNN method is non-parametric, meaning it makes no assumptions about the underlying data.

K-Nearest Neighbors (KNN) is a basic, supervised machine learning technique that may be used to both classification and regression tasks. It is simple to construct and comprehend, but as the quantity of the data used rises, its performance becomes much slower.

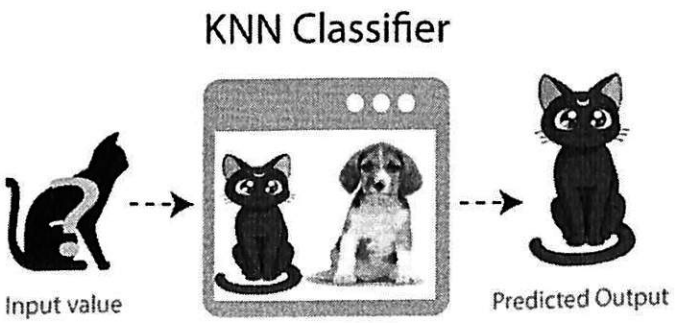


Figure 5.5: KNN Classifier

Assuming KNN as a classifier, it begins by calculating the distance between your query and all the data instances, then picking out the K examples (or closest ones) that come closest to your query. Finally, it either votes for the most frequent label, or averages all the labels (in the case of regression). The KNN algorithm immediately informs us, "Tell me your pals, and I'll tell you who you are."

Train algorithms contain both train and test modules. To train and evaluate the model, the dataset should be separated into two sections with the proportions 80% and 20%. The code below partitions the dataset into train and test sets.(Figure 5.6).

```
# train test split
Y = df['y'].values
xcols = list(df.columns)
#xcols.remove('hand')
xcols.remove('y')
X = df[xcols].values
#X = np.nan_to_num(X)
#Y = np.nan_to_num(Y)
#df.replace(np.nan, 0)
X_train, X_test, Y_train, Y_test = train_test_split(
    X, Y, test_size=test_size, random_state=42)
print(f'Train Size {len(X_train)}')
print(f'Test Size {len(X_test)}')
```

Figure 5.6: Train and Test Values Of One Dataset

5.4 Support Vector Machine

The term "support vector machine" refers to an approach for supervised machine learning that can be applied to problems involving classification as well as regression. However, its primary application is in categorization difficulties. In the SVM algorithm, each piece of data is represented as a point in n-dimensional space (where n is the number of features you have), with the value of each feature being the coordinate specificity value. Then, we classify the data by locating the hyperplane that best separates the two classes. How does the algorithm function?

Determine the proper hyperplane: Here, there are three hyperplanes (A, B, and C). Now, determine the appropriate hyperplane for classifying stars and circles. (Figure 5.7).

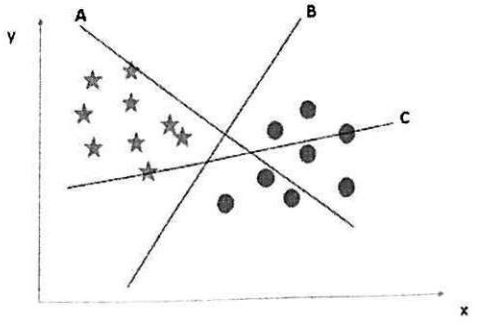


Figure 5.7: SVM Given Data

To determine which hyper-plane is best, we need to maximize the distance between the closest data point (either class) and the hyper-plane. Margin is the name given to this separation. (Figure 5.8).

In comparison to A and B, hyperplane C has a larger margin than A and B.

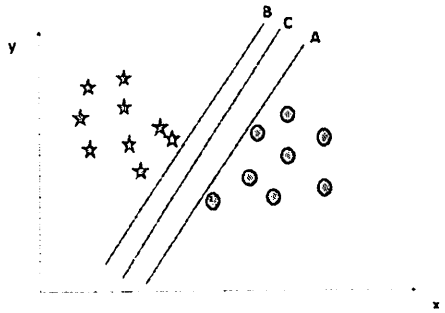


Figure 5.8: SVM Next Step

So we designated the hyperplane C. Robustness is a further argument to select the hyperplane with the larger margin. If we select a hyperplane with a low rate of return, there is a high likelihood of misclassification.

Define the appropriate hyperplane. Use the rules presented in the preceding section to determine the hyperplane accurately.

Some of you may have selected hyperplane B since it is more lucrative than A. However, there is a solution. Before profit maximization, SVM chooses the hyperplane that accurately classifies the classes. Here, hyperplane B contains misclassification, whereas hyperplane A has accurately misclassified everything. Therefore, A is the correct hyperplane.

A star at the opposite end of the stellar class resembles the class's peripheral. The SVM algorithm has the capability to disregard outliers and locate the hyperplane with the highest profit. We can therefore conclude that the SVM classification is appropriate for outliers.

SVM is able to resolve this issue. Simply, this issue is resolved by the implementation of a new functionality. Here, we will add a new feature $z = x^2 + y^2$. By charting the points on the x and z axes.

In the preceding graph, it is important to note that all z values are always positive because z is the squared sum of both x and y. In the original plot, red circles appear close to the origin of the x and y axes, resulting in a lower z value, whereas stars appear comparatively far from the origin, resulting in a larger z value. (Figure 5.9).

The SVM classifier makes it easy to have a linear plane between the classes. However, any other pressing question that arises to manually include this option for a hyperplane. No, the set of SVM rules includes a method known as the kernel trick. The SVM kernel is a feature that turns low-dimensional input space

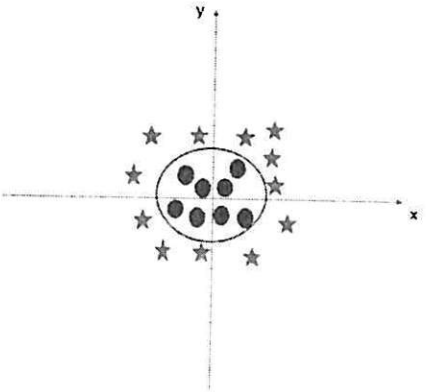


Figure 5.9: SVM Result Data

into higher-dimensional input space, or low-dimensional input space into higher-dimensional input space. It is frequently advantageous in nonlinear separable problems. Simply simply, it performs a number of exceptionally sophisticated data transformations and then exposes the method to divide the information relying on the labels or outputs they've specified.

A SVC code exists for training a dataset. It trains with and without pipeline. The dataset is separated into X train and Y train, and the scaler is a HandPoseTransform. According to scikit-learn, the definition of the pipeline class is to progressively apply a set of transforms to the final estimator. The pipeline's intermediate phases should apply fit and transform approaches, while the final estimator should implement fit.

```
# Train SVC inside of a Pipeline
scale_hands = True
if scale_hands:
    svc_model = Pipeline([
        ('scaler', HandPoseTransform()),
        ('svc', SVC(C=1, kernel='linear', random_state=42))
    ])
else:
    svc_model = SVC(C=1, kernel='linear', random_state=42)

print(f'Training SVC with {len(X_train)} tuples...')
svc_model.fit(X_train, Y_train)
print('Train Completed', end='\n\n')
```

Figure 5.10: SVM With Pipeline Code For Train

5.5 Neural Networks Algorithm

The human brain serves as an inspiration for a type of computer model known as an artificial neural network. Recent years have seen a great deal of progress achieved in the field of artificial intelligence. These advancements include voice recognition, picture recognition, and robotics that make use of artificial neural networks. Artificial neural networks are computer simulations that are inspired by biological systems and are used to carry out a variety of activities. Some examples of these tasks include the following:

- Clustering stage
- Classification stage
- Pattern recognition stage

neurons already existent in the human body, including the brain and various organs and tissues. A series of algorithms called neural networks confirms the underlying links in a data set in a manner that is analogous to how the human brain operates. The input may be improved with the assistance of neural networks, which in turn allows the network to provide superior output without requiring the procedure to be redesigned.

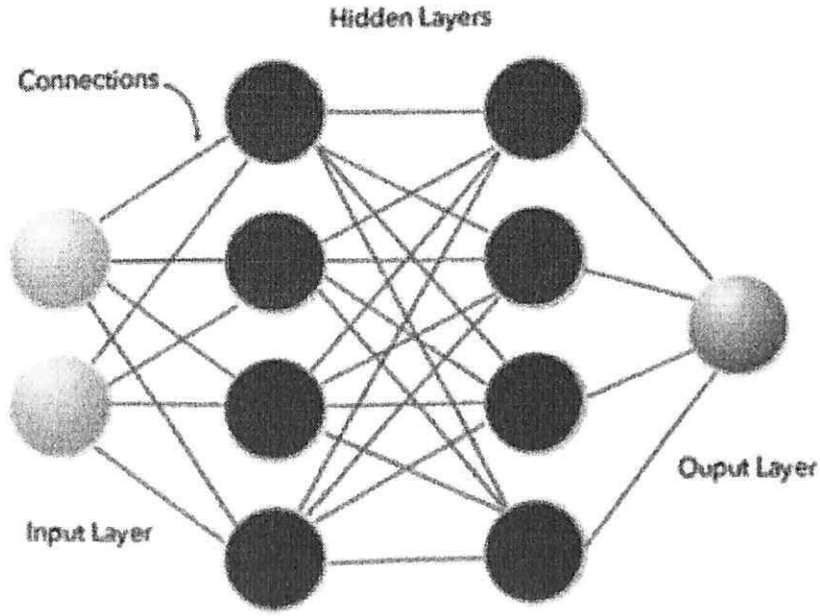


Figure 5.11: Neural Networks Algorithm

The field of research known as pattern recognition examines how computers can monitor their surroundings, learn to differentiate patterns of interest from the

background, and then make conclusions that are logical and acceptable regarding the categories that the patterns fall into. Fingerprint pictures, a handwritten text, a human face, or an audio signal are examples of patterns. Recognizing a given input pattern entails the following steps:

Supervised classification is concerned with the input model's membership in a preset class.

Classification without supervision involves assigning a pattern to an unknown class. Therefore, this recognition problem is mostly a classification or categorization work. Typically, the design of pattern recognition systems includes the following three components:

1. Data gathering and pre-processing
2. Data representation
3. Decision Making

In conclusion, the code contains an algorithm that compares the values of accuracy, F1-score, and errors of the trained model to determine the optimal parameters and model for it.

```
# Get Best Parameters
best_parameters_svc = svc_model_optimized.best_params_
del svc_model_optimized, predict_optimized_svc_model
print('Best Parameters:')
print(best_parameters_svc, end='\n\n')
```

Figure 5.12: The code for finding the best trained model.

There are below codes are for test the model. First it calls the hand detect algorithm, detects the hand landmarks. Then call the trained model, which saved in pickle format and recognises the hand gesture (5.13).

The below code is finding the landmarks and draw them in real team on hand frame, and shows the parameters for the hand gesture recognition (Figure 5.14).

5.6 Deep Learning

Deep learning is a model based on machine learning that asks computers to accomplish tasks that a person would likely perform. Deep learning, for example, is

```
hand_detect = HandDetect(detect_threshold=args.detect_threshold)
hand_pose = HandPoses(pose_threshold=args.pose_threshold,
                      name_classifier=args.path_classifier)
hand_movements = HandMovements(screen_proportion=args.screen_proportion, len_
delay = Delay(hand_pose.classifier.classes_, moving_average=args.moving_averag

cap = cv2.VideoCapture(0)
# start_time = time.time()
```

Figure 5.13: Hand Detect and Hand Recognition Codes.

```
for (pose, confidence), (lm, mp_lm) in hand_detect.detect_hand(hands=hands,
                                                               image=raw_frame,
                                                               hand_pose=hand_pose,
                                                               delay=delay):
    if args.show_lm:
        hand_detect.mp_drawing.draw_landmarks(
            image, mp_lm, hand_detect.mp_hands.HAND_CONNECTIONS)

    if pose is not None:
        cv2.putText(image, f"{pose}: ({confidence:.2f})",
                    (30, 30), cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 255, 100), 2)

        hand_movements.execute_movement(
            pose=pose, lm=lm, delay=delay, frame=image)

    else:
        cv2.putText(image, f"Idle", (30, 30),
                    cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 255, 100), 2)
        if delay.ignore_frames:
            cv2.putText(image, f"Position locked", (30, 60),
                        cv2.FONT_HERSHEY_SIMPLEX, 0.7, (255, 0, 100), 2)
key = (cv2.waitKey(10) & 0xFF)

image = cv2.resize(image, (int(image_width * .6),
                           int(image_height * .6)), interpolation=cv2.INTER_AREA)
cv2.imshow('frame', image)
```

Figure 5.14: Code For Showing The Landmarks and Hand Recognition Parameters.

a key component of autonomous vehicles' ability to detect traffic signals and people. Sound and speech recognition in numerous gadgets, such as mobile phones and tablets, are based on this principle. For the first time, deep learning can do tasks that were previously impossible. Images, text, and audio may all be used in the deep learning model since the classifiers used are so tiny and diverse. An AI system may provide results that are almost human in accuracy and even transcend human speed. Huge datasets and machine learning approaches such as CNN or ANN are used to train these models, which comprise a large number of classifiers. In machine learning, the system demonstrates how to appropriately apply the model to visuals, audio, and text. The accuracy of deep learning algorithms is even better than that of humans. Artificial neural systems with huge classes of categorized data may be created using these models, which are based on the input data. User expectations may now be met with increased accuracy and precision thanks to deep learning. Autonomous vehicles, for example, make

good use of it. In recent years, advances in artificial intelligence have shown that it can even surpass humans in the classification of photographs. A vast amount of categorized data is needed for deep learning. Develop, for example, hundreds of thousands of photographs and movies of autonomous autos. In order to master deep learning, you will need an enormous amount of energy. Equivalently built high-performance GPUs capable of deep learning. As a result of cloud computing and clusters working together, it now takes less time than previously.

5.7 Convolutional Neural Network

A convolutional neural network, often known as a CNN, is a specific type of artificial neural network that was developed with the express purpose of doing image recognition. A patterned hardware or software system may be used to create a neural network, which is responsible for monitoring the activity of neurons in the human brain. CNN is also described as a distinct category of multi-layer neural network, and each layer of a CNN uses a function to convert one level of activation into another. Deep learning makes use of a specialized architecture known as CNN. CNNs are frequently utilized for a wide variety of tasks, including the recognition of scenes and objects, image identification, extraction, and segmentation. CNNs may be broken down into two distinct phases: the formation phase and the inference phase. Convolutional layers, group layers, and fully connected layers are the three primary types of layers that are utilized in the process of developing a CNN-based architecture. The first layer of the CNN is a composite layer, which serves as the primary component of the network. It applies a number of filters to a particular image and generates a variety of distinct trigger characteristics in the image. The output will be determined by the size of the window, while the input will come from the nonlinear activation. The final layer has all of its connections made, and this is the layer in which the target is defined to decide the kind of the final output. The picture data is learnt directly from the CNN as a result of the three layers, which remove the necessity of manually extracting features using various image processing methods. CNN makes sure that the findings of the reconnaissance are original and that it is simple to retrain it for new reconnaissance missions while still allowing it to rely on the network that was there before. The significance of the usage of CNNs in the past few years

may be attributed to all of the following considerations. It is possible for the CNN to successfully capture a picture as long as the appropriate filter is applied to the time and space dependency of the image. When the amount of the input is increased, a neural network that contains completely connected neurons will see a rapid growth in the number of parameters, also known as weights.

5.8 Summary

In this chapter, the image processing model methods are explained. Image processing is the application of specific procedures to an image. The CNN technique is used in image processing to analyze the hand's structure. Using neurons in deep learning algorithms, photos are appropriately identified and passed on for processing, and palm and finger points are recovered.

6. Results and Discussion

Skeletal data is used to recognize hand gestures. For example, in the figure 6.1, an animation of the hand can be seen as an animated time series of the skeleton's movements. It illustrates the gesture sequence's hand movements and shapes. For every image t in a sequence, the camera space positions of each match are represented by three coordinates, e.g. .j.i(t) = [x.i(t) y.i(t) z.i(t)]. The skeleton at frame t is represented by the 3N.j dimension row vector, where N.j is the number

$$s(t) = [\, x_1(t)\ y_1(t)\ z_1(t)\ \ldots\ x_{N_j}(t)\ y_{N_j}(t)\ z_{N_j}(t)\,]$$

Figure 6.1:

of joints constituting the hand skeleton and N.f is the number of frames in the sequence. The final representation of a sequence is a matrix with the dimensions N.f 3N.j, where each line t corresponds to the row vector s(t): This new data

$$\mathcal{M} = \begin{bmatrix} s(1) \\ \vdots \\ s(N_f) \end{bmatrix}$$

Figure 6.2:

type stores a great deal of information regarding the motion and shape of the hand on the string. To properly describe the gesture, we propose capturing the modification of the hand's shape according to the skeleton's joints, as well as the motion direction and rotation of the hand in space, using three unique features.

separate. There are six hand gestures: swipe right, swipe left, zoom in, zoom out, up, down.

Initially, they tested the KNN algorithm, SVM. The first SVM has an accuracy of 0.4 and a KNN of 0.22. The decision was to go deeper and modify the algorithm for the best classification. It is believed that the results of KNN are too bad, because of fewer data sets. To increase the likelihood, they decided to focus on two goals for classification: zoom in and out. The best results are obtained from the Deep Learning algorithm. As expected almost perfect for 2 layers, zoom in and out. Good accuracy in an ideal environment (same person, from a trained dataset) shows better than 90% accuracy. They used a powerful Keras framework for training and prediction. Keras is a Python-based API for deep learning that runs on the machine learning platform TensorFlow. It was created with the goal of facilitating rapid experimentation. This is an example of our training model. (Figure 6.3).

```
model = tf.keras.models.Sequential([
    tf.keras.layers.Input((10, )),
    tf.keras.layers.Dropout(0.2),
    tf.keras.layers.Dense(20, activation='relu'),
    tf.keras.layers.Dropout(0.4),
    tf.keras.layers.Dense(5, activation='relu'),
    tf.keras.layers.Dense(NUM_CLASSES, activation='softmax')
])

model.compile(
    optimizer='adam',
    loss='sparse_categorical_crossentropy',
    metrics=['accuracy']
)
```

Figure 6.3: Prediction Model

The results obtained from the training Deep Learning model was pretty good from the 30 epochs:

Epoch 1/1000 1/27 [>.....] - ETA: 0s - loss: 1.1295 - accuracy: 0.3203

Epoch 00001: keeping/saving model to model/classifier/classifier.hdf5

27/27 [==========] - 0s 11ms/step - loss: 1.1004 - accuracy: 0.3602 - valloss: 1.0431 - valaccuracy: 0.5220
Epoch 2/1000
1/27 [>.....] - ETA: 0s - loss: 1.0440 - accuracy: 0.4844
Epoch 00002: keeping/saving model to model/classifier/classifier.hdf5 27/27 [====
- 0s 3ms/step - loss: 1.0503 - accuracy: 0.4297 - valloss: 0.9953 - valaccuracy: 0.6397
Epoch 3/1000
1/27 [>.....] - ETA: 0s - loss: 1.0043 - accuracy: 0.5312
Epoch 00003: keeping/saving model to model/classifier/classifier.hdf5 27/27 [====
- 0s 4ms/step - loss: 1.0210 - accuracy: 0.4582 - valloss: 0.9545 - vaaccuracy: 0.6523
Epoch 4/1000
1/27 [>.....] - ETA: 0s - loss: 0.9503 - accuracy: 0.5625
Epoch 00004: keeping/saving model to model/classifier/classifier.hdf5
...
Epoch 28/1000
Epoch 00028: keeping/saving model to model/classifier/classifier.hdf5
Epoch 29/1000 29/29 [==========] - 0s 3ms/step - loss: 0.6833 - accuracy: 0.7196 - val_loss : 0.3877 $-val_accuracy$: 0.9390

The data used for testing model are 2 points from fingers, for train and future prediction (Figure 6.4).

There is a test result for zoom in: [9.95287−011.1321−022.99215−044.11445−05, 5.77433−011.112111−022.2328216e−046.6191799e−05, 3.005e−011.639245−022.2328216e−044.6191799e−05, 8.8105639e−011.02−022.2328216e−048.90012−05], 0 − *zoomin*

Firstly, to create the best model, data collection plays a key role, so a simple collection scenario developed in the project will make it easy to collect categorical data, while data is collected in real time by pressing the space button. The second major impact of the script is to prevent false positives from retraining the model by adding a new dataset immediately. Hand gesture recognition is found using well-known classification algorithms in machine learning. The effectiveness of the method was evaluated on the hand image dataset. Experimental results will be presented after the research approach.

Figure 6.4: Zoom in hand gesture detection with the two fingers.

The performance of their method will depend heavily on the results of hand detection on a horizontal surface and the collected data set. If there are moving objects with similar skin color, the objects persist in the hand detection results and then degrade the performance of hand gesture recognition as in previous works. That's why it's better to write the wrapper function in a separate main and then pass the copy of the new image to the MediaPipe. The researchers hope that in the future, machine learning methods and 2D cameras can be used to solve the complex background problem and improve the hand detection problem. For future research: should add horizontal hand gesture logic recognition and add some algorithm, by learning, to make it smarter.

```
model.fit(
    X_train,
    y_train,
    epochs=1000,
    batch_size=128,
    validation_data=(X_test, y_test),
    callbacks=[stop_callback]
)
```

Figure 6.5: Model fit

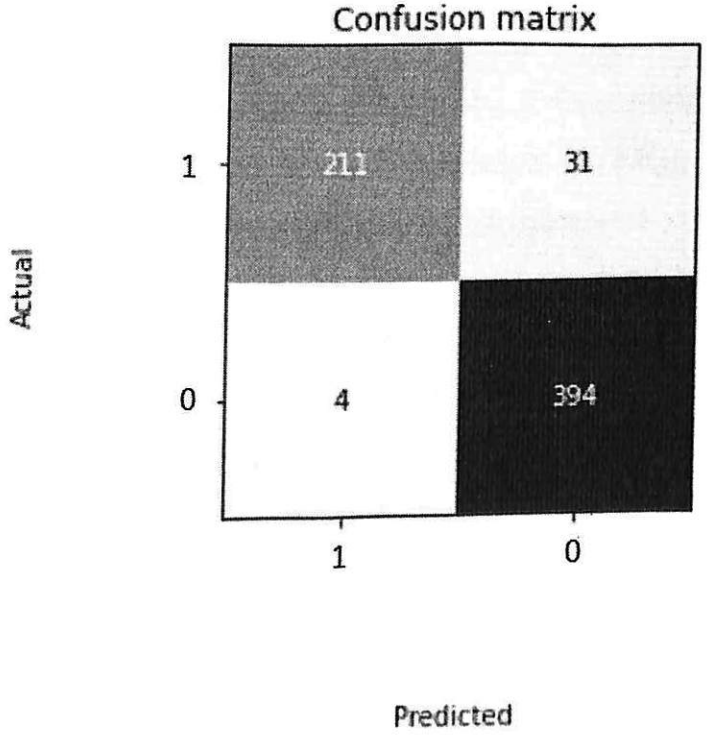


Figure 6.6: Confusion matrix for Deep Learning model.

7. Conclusion and Future work

7.1 Conclusion

In this study, the authors addressed many challenges related to the recognition of dynamic hand gestures from numerical data, a topic that has been intensively studied due to its wide range of applications. Initially, they intended to do dynamic hand gesture detection on a heterogeneous set of gesture types extracted from video or real-time photos. Indeed, the hand is a complex topological entity. There are an endless number of ways the hand can perform the same move. In human-computer interaction programs, for instance, the use of hand movements to control pan and zoom is prevalent. In order to execute a comprehensive recognition procedure, it is necessary to consider the shape of the hand and the variations in its movement throughout a gesture. The standard number of frames is four, as the duration of each movie differs and certain frames are skipped. Thus, four is the best number. There are indicators and finger locations at the end of the process that can be used to recognize hand motions from animation (list of key points) and execute certain commands. In addition, the MediaPipe framework proposed a real-time system for extracting the 3D locations of a human skeleton from a depth image in the field of gesture and action detection. Some of the symbols mentioned in the literature suggest that the position, velocity, and orientation of the joints can serve as appropriate descriptive symbols for characterizing human actions. Following these principles, we attempted to conduct dynamic hand gesture recognition using the properties of the hand frame, as depicted in the figure's finger and palm points. Because hand gesture recognition employing skeleton features is in its infancy. As a result, they developed an algorithm that collects a Hand Gesture dataset containing the hand skeleton sequencing of specific photos. First, data gathering is essential for developing the best model, thus a simple collect-

ing scenario designed during the project will make it simple to collect categorical data, while data is taken in real time by pressing the space bar. Adding a new dataset immediately prevents false positives from retraining the model, which is the second main effect of the script. Hand gesture recognition is discovered utilizing well-known machine learning classification algorithms. On the basis of the hand picture dataset, the method's efficacy was evaluated. The experimental results will be provided following the study methodology. To evaluate the difficulty of recognition from a diverse array of motions, participants were instructed to produce gestures utilizing both one finger and the entire hand. They devised a method for recognizing dynamic hand gestures utilizing three gesture parameters computed from the skeletal hand sequence: finger directions from known finger points. Using the skeletal properties of the fingers and palm for hand gesture identification is shown to have a bright future based on an evaluation of the approach. The evaluation findings illustrate the efficacy of our method for image-based granular descriptors. However, the results also demonstrate an inaccuracy in representing the dynamics of complicated hand motions when compared to the feature learning capabilities of contemporary deep learning techniques. Each type of data set employs the NN, KNN, and SVM methods. The findings for accuracy vary, and the precision number is less than 0.8, which indicates a low result. Using a Deep Neural Network technique, the optimal precision value is close to 0.90. All of these methods are required to train a model, evaluate it, and identify gestures. Recent research has revealed the exceptional efficacy of deep neural networks in numerous fields of study. This is because deep learning algorithms rely on the specific data provided, and as more and more data is uploaded, the effectiveness of the systems improves. However, it is tough to comprehend, therefore they made data collection extremely user-friendly and developed two ways for data collection. First, they offer a recognition solution that enables the system to identify the presence of a gesture in an unsegmented video stream and recognize the gesture type before the video concludes, a capacity required for practical applications. Second, they managed the entirety of the recognition process, from hand posture estimation through classification, and used the ability of deep learning models to improve the system's efficiency and robustness. The final frame is mostly composed of three steps. First, they read and extract relevant hand characteristics from the frame using MediaPipe. The test results have demonstrated that the

suggested method is capable of recognizing hand motions and outperforming current methods. In addition, studies have demonstrated that the new framework can detect the presence of a gesture and recognize it well before it ends, making our system efficient for real-time applications with other users. Using a transfer learning strategy enables us to surpass contemporary deep learning techniques while employing fewer parameters than the base model. However, they have not yet mastered human performance, as motions with striking similarity still display perplexity.

7.2 Future works

Their system's performance will depend heavily on the outcome of manual discovery on horizontal faces and settled datasets. Nonetheless, if there are moving items with the same color as the skin, like in the previous workshop, the presence of these things degrades the performance of the hand gesture detection algorithm. Therefore, it is preferable to construct a wrapper function that separates the hand and passes the duplicate of the new picture to MediaPipe. The researchers anticipate that in a future workshop, machine literacy styles and 2D cameras will be utilized to address the complex background problem and improve the hand detection difficulty. Future investigation should include horizontal hand gesture recognition and the addition of some algorithms to make it smarter through training. The paper samples demonstrated that the suggested solutions ensure an effective recognition of dynamic hand gestures, but do not surpass human performance. First, the study of hand gesture identification employing temporal learnt characteristics on sequences of hand finger and palm points is insufficient. High-parallelism movements may be distinguished in novel ways if temporal modeling is refined by include intermittent layers. For instance, the system may automatically identify the fritters that produce the most reliable data and focus on them to do gesture recognition. It indicates that the MediaPipe algorithm must be more scalable. They continue to struggle with collisions involving rapidly moving hands or fingers.

7.3 Suggestions for the next steps

After conducting all the trials and reviewing the results, the following recommendations are made for future research:

- Increase the number of hand gestures to include all typical motions, such as left click, right click, hold, and more, in order to create a strong role model for those who have suffered a stroke and are unable to perform such duties. These gestures can be simply learnt to improve interpersonal communication.

- Implement alternative deep learning algorithms, such as RNN, because the data utilized in the current testing may be useful. A comparison of the impact of CNN and RNN on training and testing precision.

References

[1] V. Bazarevsky, A.Tkachenka F. Zhang A.Vakunov, and G. Sung. *MediaPipe Hands: On-device Real-time Hand Tracking.* URL: `https://arxiv.org/pdf/2006.10214.pdf`. (accessed: 2020).

[2] V. Bazarevsky and F. Zhang. *On-Device, Real-Time Hand Tracking with MediaPipe.* URL: `https://ai.googleblog.com/2019/08/on-device-real-time-hand-tracking-with.html`. (accessed: 2019).

[3] M. Van den Bergh D. Uebersax J. Gall and L. Van Gool. "Real-time sign language letter and word recognition from depth data". In: In Proceedings of the IEEE International Conference on Computer Vision Workshops (ICCV '11) (November 2011), pp. 383–390.

[4] P. Benölken D. Wickeroth and U. Lang. "Markerless gesture based interaction for design review scenarios". In: In Proceedings of the 2nd International Conference on the Applications of Digital Information and Web Technologies (ICADIWT '09) (August 2009), pp. 682–687.

[5] V. Frati and D. Prattichizzo. "Using Kinect for hand tracking and rendering in wearable haptic". In: In Proceedings of the IEEE World Haptics Conference (WHC '11) (June 2011), pp. 317–321.

[6] H. Park J. Choi and J.-I. Park. "Hand shape recognition using distance transform and shape decomposition". In: In Proceedings of the 18th IEEE International Conference on Image Processing (ICIP '11) (September 2011), pp. 3605–3608.

[7] Y. Sun J. Zeng and F. Wang. "A natural hand gesture system for intelligent human-computer interaction and medical assistance". In: In Proceedings of the 3rd Global Congress on Intelligent Systems (GCIS '12) (November 2012), pp. 382–385.

[8] Y. Wang L. Ge Z. Ren and J. Yuan. "3d hand shape and pose estimation from a single rgb image". In: In Proceedings of the IEEE conference on computer vision and pattern recognition (2019), pp. 10833–10842.

[9] N. Pugeault and R. Bowden. "Spelling it out: real-time ASL fingerspelling recognition". In: In Proceedings of the IEEE International Conference on Computer Vision Workshops (ICCV '11) (November 2011), pp. 1114–1119.

[10] S. Sarkar R. Yang and B. Loeding. "Handling movement epenthesis and hand segmentation ambiguities in continuous sign language recognition using nested dynamic programming". In: IEEE Transactions on Pattern Analysis and Machine Intelligence 32.3 (2010), pp. 462–477.

[11] Matthias Rehm. "Human-Centric Interfaces for Ambient Intelligence". In: ().

[12] L. Silva Santos and M. Aranda Espíndol. *Gesture Hand Controller.* URL: `https://github.com/luizhss/Gesture_Hand_Controller`. (accessed: 2021).

[13] L. Silva Santos and M. Aranda Espíndol. *Gesture Hand Controller Dataset.* URL: `https://github.com/luizhss/Gesture_Hand_Controller/blob/master/dataset_train.csv`. (accessed: 2021).

[14] L. Silva Santos and M. Aranda Espíndol. *Gesture Hand Controller Video.* URL: `https://www.youtube.com/watch?v=OKRuiNP62Qc`. (accessed: 2021).

[15] T.-D. Tan and Z.-M. Guo. "Research of hand positioning and gesture recognition based on binocular vision". In: In Proceedings of the IEEE International Symposium on Virtual Reality Innovations (ISVRI '11) (March 2011), pp. 311–315.

[16] Wikipedia. *Computer Vision.* URL: `https://en.wikipedia.org/wiki/computer-vision`.

[17] Wikipedia. *Dataset.* URL: `https://en.wikipedia.org/wiki/Data_set`.

[18] Wikipedia. *Human-Computer Interaction.* URL: `https://en.wikipedia.org/wiki/Human%E2%80%93computer_interaction`.

[19] Wikipedia. *Machine Learning.* URL: `https://www.ibm.com/topics/machine-learning`.

[20] Wikipedia. *Python*. URL: https://en.wikipedia.org/wiki/Python_(programming_language).

[21] T. Starner Z. Zafrulla H. Brashear and H. Hamilton. "American sign language recognition with the kinect". In: In Proceedings of the 13th ACM International Conference on Multimodal Interfaces (ICMI '11) (November 2011), pp. 279–286.