Hand Detection and Segmentation Schemes for Trajectory-guided Gesture Recognition

A

Thesis Submitted

in Fulfilment of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

By

Debajit Sarma

Department of Electronics and Electrical Engineering

Indian Institute of Technology Guwahati

Guwahati, India.

October, 2022

Dedicated To

Lord Almighy

for His blessings

My guide **Prof. M.K. Bhuyan**

for his guidance and inspiration

My beloved wife **Dr. Trishna Devi**

for her unconditional heartfelt love and sacrifices

&

My **parents** and **parents-in-law**

for their blessings

Certificate

This is to certify that the thesis entitled **"Hand Detection and Segmentation Schemes for Trajectory-guided Gesture Recognition,"** submitted by **Debajit Sarma** (156102003), a research scholar in the *Department of Electronics and Electrical Engineering, Indian Institute of Technology (IIT) Guwahati*, for the award of the degree of **Doctor of Philosophy**, has been carried out by him under my supervision and guidance. The thesis has fulfilled all requirements as per the regulations of the institute and in my opinion has reached the standard needed for submission. The results embodied in this thesis have not been submitted to any other University or Institute for the award of any degree or diploma.

Dated:

Guwahati.

Dr. M.K. Bhuyan
Professor,
Dept. of Electronics and Electrical Engineering,
Indian Institute of Technology Guwahati,
India - 781039.

Acknowledgements

I am obliged to God for His divine guidance and blessings. I would also like to thank all those people who made this dissertation possible.

First of all, I would like to express my profound respect and gratitude to my supervisor, Dr. M.K. Bhuyan, who has been the guiding force behind this work. I am greatly indebted for his guidance, constant encouragement, and valuable comments on my work. I am fortunate enough to have such an advisor who gave me the freedom to think independently and explore new ideas. More importantly, I would like to thank for the patience he has shown in carefully reading and commenting on the manuscripts, and countless revisions of this dissertation. His commitments and dedication to research have been and will continue to be a constant source of inspiration for me. I have no doubts that finishing my degree in a proper and timely manner would have been impossible without his help. I am highly privileged for getting an opportunity to work with such a wonderful person.

I would also like to thank my doctoral committee members Dr. P. Guha, Dr. S. Sundaram and Dr. A. Anand for their invaluable suggestions, encouragements, and moral supports that helped me to improve my research work. I am also thankful to the Head, other faculty members and non-teaching staff of Department of Electronics & Electrical Engineering for their kind help extended during my academic studies.

On a personal note, I would like to thank my wife Dr. Trishna Devi for her constant support and for being with me in every aspect of my life. My special thanks to my friends-cum-moral adviser Mr. Pradipta Sasmal, Dr. Tilendra Choudhury and Miss Shikha Baghel, for their guidance and insightful comments during the entire journey of my PhD life.

My special thanks go to my seniors Dr. Biplab Ketan Chakraborty , Dr. Sunil Kumar, Dr. Amit Vishwakarma, Dr. Santhosh Yadav, Dr. Gaurav Kumar, for their motivation and support. I had a great time with many of my friends at IIT Guwahati, including (but not limited to) Dr. Aniruddha, Mr. Nayan, Mr. Pallab, Mr Allen, Mr. Snehil, Mr. Shakhanil, and Mr. Soumayan, Mr. Sandeep, Mr. Sumon, Miss Saswati, Miss Sukanya, Dr. Deepika, Dr.

Subhalaxmi, Miss Moa, Dr. Anirban Bhowal, Mr. Samarjeet, Miss Mouchumi, Mr. Prabhakar, and Dr. Vineeta. I would like to thank them for their support and encouragement.

I am grateful to my grandparents, parents, brothers and their families and the entire family of my in-laws whose love, encouragement, and support made this research work possible. I am thankful to IIT Guwahati for providing the research environment and the MHRD scholarship to undertake my PhD research. A kind thanks to all the doctors, nurses, and staff members of the Institute Hospital who have taken care of my health timely. I would like to thank each and every person who share a part of my life in the journey of my Ph.D. Finally, I would like to thank the Almighty God for bestowing me this opportunity and showering his blessings on me to come out successful against all odds.

Debajit Sarma

Abstract

One very interesting field of research that has gained much attention in recent times is Gesture Recognition. In the context of human-computer non-verbal communication, the visual interface becomes important in establishing communication via understanding human intention from their behaviour, such as facial expressions, hand gestures, etc. With the increasing interest in human-computer interaction (HCI), there has been rapid growth in studies related to vision-based gesture recognition in recent years. Hand gesture recognition from visual images finds applications in areas like human-computer interaction, machine vision, sign language, virtual reality, augmented reality, and so on.

Gesture recognition may be accomplished either by capturing gestural motion using sensor devices or by analyzing gesture images/videos using computer vision techniques. Vision-based gesture recognition typically depends on three stages: (a) gesture acquisition, detection and preprocessing; (b) gesture representation and feature extraction; and (c) recognition or classification. The acquisition includes capturing the gestures using various kind of imaging devices whereas detection and preprocessing includes segmentation of gesturing body parts from images and videos as accurately as possible. The accurate detection of gestures is significantly affected by physical movement, variations in illumination and shadows, presence of skin-like colors in background, occlusion, background complexity, and different other factors. The complex articulated shape of the hand makes it harder to model the hand appearance for both static and dynamic gestures. Variation of gesture parameters due to spatio-temporal variance in dynamic gesture makes the recognition process more difficult. Different classifiers with the existing features used for vision-based gesture

recognition may not be capable of simultaneously handling all the gesture classification problems. Each one has some drawbacks limiting the overall performance. Various researchers have developed different methods to overcome such problems. However, there are still many shortcomings with the algorithms developed so far. This has motivated us to carry out research in this field.

In general, there are three types of dynamic hand gestures – (1) Gestures with local motion only where only the fingers and the palm move without any movement of the whole hand/arm, (2) Gestures with global motion only where the hand as a whole move differently in the 3D space to make different gestures, and (3) Gestures with both local and global motions where the fingers and palm make different hand poses while moving the arm in space. In this research work, an attempt has been made to recognize the second category of dynamic hand gestures having only global motion with different spatio-temporal and motion characteristics which are called trajectory-based gestures. So, here in this dissertation, we are basically going to look into different approaches to recognize trajectory-based gestures and try to propose some models for the betterment of the same. For this purpose, various methods have been adopted to extract the trajectory of the gesturing hand. Another prime objective is to make these methods color, shape and size invariant which makes these methods more generalized. In these methods, deep neural architectures have been used that can learn inherent features automatically in a hierarchical manner from local to global with multiple layers of abstraction from a vast number of sample images. The application of deep neural networks has also helped us to remove some of the prevailing constraints of hand-crafted features to a better extent. Experimental results show that the proposed trajectory extraction methods have achieved better performance compared to state-of-the-art methods in hand gesture recognition.

Contents

List of Figures

List of Tables

List of Acronyms

ANN	Artificial Neural Network
CAMShift	Continuous Adaptive Mean-shift
CNN	Convolutional Neural Network
C3D	3D-CNN
CRF	Conditional Random Field
DCT	Discrete Cosine Transform
DI	Dynamic Image
DTW	Dynamic Time Warping
FSM	Finite-State Machine
HCI	Human-Computer Interaction
HMM	Hidden Markov Model
HOG	Histogram of Oriented Gradient
LSTM	Long Short-Term Memory
MCC	Motion Chain Code
MEI	Motion-Energy Image
MHI	Motion-History Image
MRF	Markov Random Field
OFMT	Optical Flow-guided Motion Template
PCA	Principal Component Analysis
ReLU	Rectified Linear Units
RNN	Recurrent Neural Network
RST	Rotation-Scaling-Translation

SGD	Stochastic Gradient Descent
SIFT	Scale-Invariant Feature Transform
SURF	Speeded Up Robust Features
SVM	Support Vector Machine
VGR	Vision-based Gesture Recognition

1

Introduction

The ability of computers to recognize hand gestures visually is essential for the future development of systems in the human-computer interaction (HCI) community. Hand gesture recognition is an important research area in computer vision with many applications. The major applications of gesture recognition cover various domains, ranging from sign language to medical assistance to virtual reality. The initial task of a hand gesture-based HCI system is to acquire raw data which can be achieved mainly by two approaches: sensor-based and vision-based. The sensor-based approach requires the use of sensors or instruments physically attached to the arm/hand of the user to capture data. Whereas vision-based approaches require the acquisition of images or videos of the hand gestures through a video camera. However, vision-based recognition is extremely challenging not only because of its diverse contexts, multiple interpretations, and spatio-temporal variations but also because of the complex non-rigid properties of the human hand. Here, in this thesis, we will basically talk about vision-based dynamic hand gesture recognition through different trajectory extraction methods which are mostly invariant to shape, size and color of the hand. Experimental results on publicly available databases show the effectiveness of the proposed approaches. This introduction chapter gives an idea of different types of gestures, an overview of gesture acquisition and recognition system, its application and major challenges in this research. It also includes the research motivation and finally, the organization of the thesis is presented at the end.

1.1 Gestures for Human-Computer Interaction (HCI)

In this era of technology, where we are deep into the information age, technological advancement has reached such a point that almost everyone in every nook and corner of the world irrespective of any discipline, has come in contact with computers in some way or the other. But in general, a common user should not have to acquire computer literacy to use computers for common tasks in everyday life. *Human-computer interaction* (HCI) concerns "the design, evaluation, and implementation of interactive computing systems for human use and with the study of major phenomena surrounding them" [14], especially "in the context of the user's task and work" [15]. Basically, *HCI* is a field of study which aims to facilitate the interaction of users, whether experts or novices, with computers in an easy way. It improves user experience by identifying factors that helps reduce the learning curve for new users and also provides provisions such as keyboard short-cuts and other navigational aids for common users. Moreover, as the world adapts to the new changes after the COVID-19 pandemic, touch-less technology can be the 'new normal' in such situations to minimize the risk of a global health crisis. For instance, in airports, if cameras and hardware are already embedded, passengers can take benefit from hand tracking and gesture recognition to control menus without physically touching a platform. Though there are some other touch-less technologies such as voice recognition, language and pronunciation become a barrier in many instances. Moreover, people are focusing on using smartphones to minimize contact when it comes to aspects such as check-in. However, with smartphones, passengers still often have to touch a screen, which gives a chance of risk. Additionally, at airport border control, it is often forbidden to use a smartphone. So, there are further limits to these existing features. In addition, on roads, drivers can control auto navigation through simple in-air movements. In such cases, hand-tracking and gesture recognition technology can provide a hardware-agnostic solution to these problems. Gestures can be made universal and users can apply user-friendly gestures in place of multi-step interactions for communication. With a worldwide focus on reducing the risk of spreading bacteria and viruses, this sort of solution would undoubtedly be welcomed by all. Hence, the need has increased for interfaces that support effective *human-computer interaction* (HCI). In striving toward these ends, *HCI* research builds on progress in several related fields, as shown in Fig. 1.1. It is focused not only on enhancing the usability, reliability, and functionality of present-day interfaces

but also on the development of novel, innovative interfaces that can be used in natural, lifelike ways. Such interfaces are in demand for interacting with virtual environments in computer games and virtual reality, for teleoperation in robotic surgery, and so on. Active interfaces, intelligent adaptive interfaces, and multi-modal interfaces are all gaining prominence.

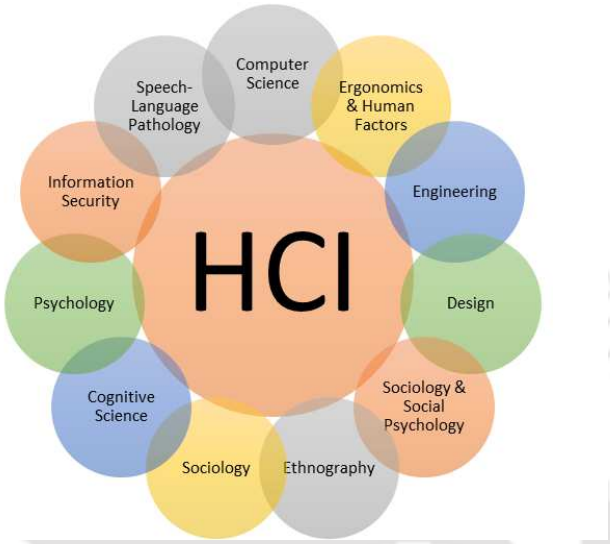


Figure 1.1: Human-computer interaction and related research fields.

With the increased interest in human-computer interaction, research related to gesture recognition has grown rapidly. Along with speech, they are the obvious choice for natural interfacing between a human and a computer. Human gestures constitute a common and natural means for nonverbal communication. A gesture-based *HCI* system enables a person to input commands using natural movements of the hand, head, and other parts of the body [16] (Fig. 1.2). And since the hand is the most widely used body part for gesturing apart from face [17], hand gesture recognition from visual images forms an important part of this research. Generally, hand gestures are classified as static gestures or postures and dynamic or trajectory-based gestures. Again, dynamic or trajectory gestures can be isolated or continuous.

1.1.1 Gesture acquisition

The primary task of a hand gesture-based *HCI* system is to acquire raw data which can be achieved mainly by two approaches [18]: contact-based or wearable sensor-based systems (Fig. 1.3a) and vision-based systems (Fig. 1.3b).

Sensor-based approaches require the use of sensors or instruments physically attached to the

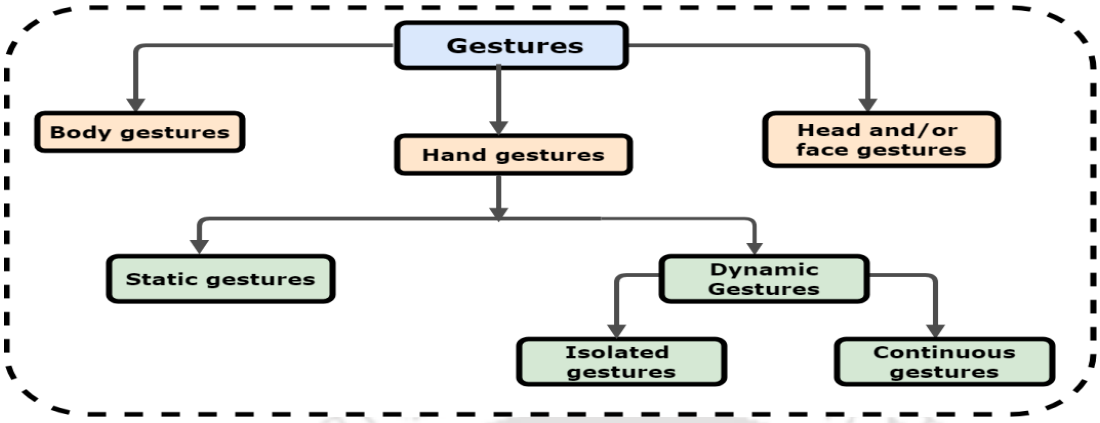


Figure 1.2: Classification of different gestures based on used body-part.

(a) **(b)**

Figure 1.3: Human-Computer Interaction using: (a) CyberGlove-II, (b) Vision-based system [1].

arm/hand of the user to capture data consisting of position, motion and trajectories of fingers and hand. Sensor-based methods are mainly as follows:

(i) Glove-based approach measures acceleration, position, degree of freedom and bending of the hand and fingers. This includes the use of flex sensors, gyroscope and accelerometer.

(ii) Electromyography (EMG) measures human muscle's electrical pulses and harness the bio-signal to detect finger movements.

(iii) WiFi and Radar use radio waves, broad beam radar or spectrogram to detect in-air signal strength changes.

(iv) Others utilize ultrasonic, mechanical, electromagnetic and other haptic technologies.

Vision-based approaches require the acquisition of images or videos of the hand gestures through video cameras.

(i) Single camera-webcam, video camera and smart-phone camera.

(ii) Stereo-camera and multiple camera-based systems — a pair of standard color video or still cameras capture two simultaneous images to give depth measurement. Multiple monocular cameras can better capture the 3D structure of an object.

(iii) Light coding techniques - projection of light to capture the 3D structure of an object. Such devices include PrimeSense, Microsoft Kinect, Creative Senz-3D and Leap Motion Sensor etc.

(iv) Invasive techniques-body markers such as hand color, wrist bands, and finger marker. But the term vision-based is generally used for capturing images or videos of the bare hand without any glove and/or marker. The sensor-based approach reduces the need for pre-processing and segmentation stage, which is essential to classical vision-based gesture recognition systems. But for contact or wearable sensor-based systems, the user needs to be accustomed to these devices for accurate usage. Moreover, vision-based gesture interfaces are preferred to data gloves because of their simplicity, contact free and more natural way of interaction and low cost.

Again based on the number of input channels used in the system, a HCI system can be classified as *unimodal* or *multimodal* [19] (Fig. 1.4). Unimodal systems can be a) *vision-based* (*e.g.*, body movement tracking [20], facial expression recognition [21,22], gaze detection [23], and gesture recognition [24]), b) *audio-based* (*e.g.*, auditory emotion recognition [25], speaker recognition [26], and speech recognition [27]), or c) *based on other types of sensors* [28]. The most researched unimodal HCI systems are vision-based. People typically use multiple modalities during human-human communication. Therefore, to assess a user's intention or behaviour comprehensively, HCI systems should integrate information from multiple modalities as well [29]. Multimodal interfaces can be setup using combinations of inputs, such as gesture and speech [30] or facial pose and speech [31] etc.

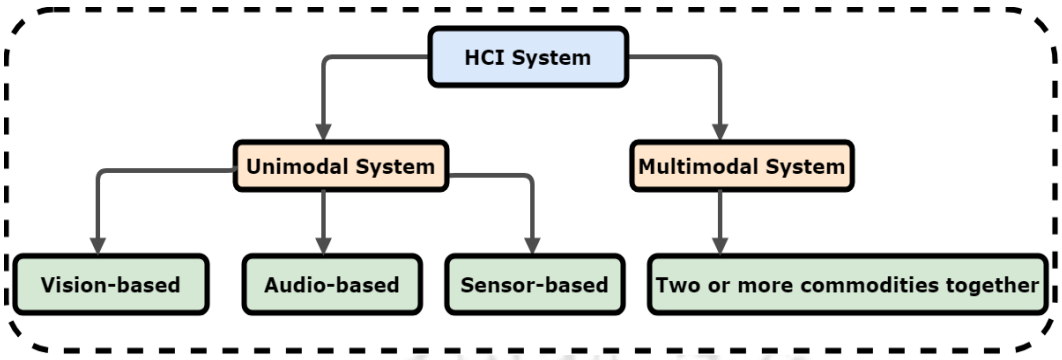


Figure 1.4: General taxonomy of HCI system based on input channel/channels.

1.2 Overview of Vision-based Hand Gesture Recognition (VGR) System

The primary task of vision-based interfaces is to detect and recognize visual information for communication. A vision-based approach is more natural and convenient than a glove-based approach. It is easy to deploy and can be used anywhere within a camera's field of view. The straightforward approach to vision-based gesture recognition (VGR) is to acquire visual information of a person in a certain environment and try to extract the necessary gestures. This approach must be performed in a sequence, namely, acquisition, detection and pre-processing; gesture representation and feature extraction; and recognition (Fig. 1.5).

(i) ***Acquisition, detection and pre-processing:*** The acquisition and detection of the gesturing body part is crucial because the accuracy of the VGR system depends on it. The acquisition includes capturing gestures using imaging devices. Detection and pre-processing segments the gesturing body parts from images or videos as accurately as possible.

(ii) ***Gesture representation and feature extraction:*** The next stage in a hand gesture recognition task is to choose a mathematical description or representation of the gesture. The scope of a gestural interface is directly related to the proper representation of hand gestures. After gesture modeling, a set of features needs to be extracted for gesture recognition. Many authors have identified different features for representing particular kinds of gestures [32].

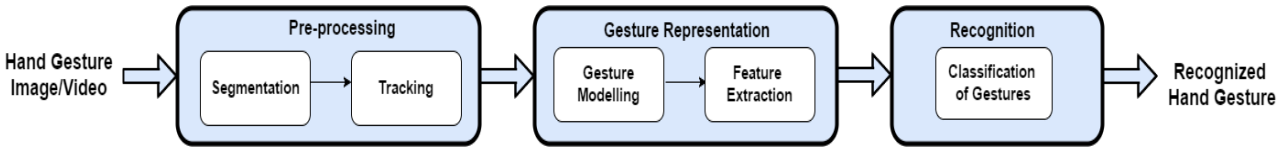


Figure 1.5: The basic architecture of a typical gesture recognition system.

(iii) ***Recognition:*** The final stage of a gesture recognition system is recognition or classification. A suitable classifier recognizes the incoming gesture parameters or features and groups them into either predefined classes (supervised) or by their similarity (unsupervised) [24]. There are many classifiers used for both static and dynamic gestures, each with its own limitations.

1.3 Application and Recent Advancement of VGR Systems

A vision-based approach is more natural and convenient compared to other glove-based approaches used in HCI. It can be used anywhere in the field of view of a camera. It does not require special hardware that the operator needs to master and, thus, is easier to deploy. A vision-based approach also enables a variety of gestures to be used that can be updated in the software. Computer vision methods can enable human-computer interaction that is difficult or impossible to achieve with other modalities. Visual information is important in human-human communication because meaning is conveyed through identity, facial expression, posture, gestures, and other visually observable attributes. Therefore, intuitively it is possible to have natural human-computer interaction by sensing and perceiving these visual cues from video cameras placed appropriately in the environment. The major advantage of vision-based gesture recognition is that it requires cheap input devices. A digital camera can be integrated with a single chip. Mass-production is therefore much easier to realize compared to other input-devices with mechanical parts, such as a data glove. In addition, the cost of image processing hardware can be saved because most computers now have a central processing unit and graphics processing unit fast enough to perform this computer vision task. More importantly, computer vision is versatile. While other input devices such as a mouse, joystick, and trackpad are limited to a specific function; computer vision offers a whole range of possible future applications not only in human-computer interaction but also in user authentication, video conferencing, and

distance education. Another important advantage of computer vision is that it is non-intrusive. Cameras are open input devices that do not require direct contact with the user to sense actions. The user can interact with the computer without wires and without manipulating intermediary devices. Moreover, humans are more comfortable in communicating with body postures or gestures as compared to using some mechanical techniques like clicking the mouse or pressing the keyboard or touching a touch-sensitive screen and thus experience more comfortable and better natural interactions than with traditional interaction techniques. They have major advantages, including a natural, contact-free method of interaction. However, vision-based gesture interfaces also have major disadvantages, include user fatigue, cultural differences, the requirement of high-speed processing, and noise sensitivity. Nevertheless, it is more difficult to use because state-of-the-art computer vision algorithms are still limited in processing such highly articulated, non-convex, and flexible objects as the human hand. Vision-based recognition is extremely challenging not only because of its diverse contexts, multiple interpretations, and spatio-temporal variations but also because of the complex non-rigid properties of the human hand. The existing classifiers used for vision-based gesture recognition are not capable of simultaneously handling all the gesture classification problems. Each of them has one or more drawbacks limiting the overall performance of the gesture recognition methods.

Despite all the drawbacks, the number of VGR systems is assumed to increase more in daily life; and as such, interactive technology needs to be designed effectively to provide a more natural way of communication. Therefore, in recent years, vision-based gesture recognition has become a key research topic in HCI and there are many real-life applications of VGR. More specifically hand gestures based VGR systems can provide a noncontact input modality. The widespread use of gesture-based interfaces for vision-based HCI is possible due to the advantages mentioned above. One of the breakthroughs in VGR is the introduction of Microsoft Kinect® as a contact-less interface [33]. The Kinect has significant potential in various applications, such as healthcare [34], education [35], etc. However, its poor outdoor performance and depth resolution limit its usability. Recently, SoftKinetic's Gesture Control Technology is incorporated in BMW cars to allow drivers to navigate the in-vehicle infotainment system with ease [36]. Most recently implemented and some proposed applications of VGR include sign language recognition [37], virtual reality [38], virtual game [39], augmented reality [40], smart video conferencing [41], smart home and office [42], healthcare and medical assistance (MRI navigation) [34], robotic

surgery [43], wheelchair control [44], driver monitoring [45], vehicle control [46], interactive presentation module [47], and virtual classroom [48], e-commerce [49], and so on. Some of the major applications (see Fig. 1.6) of hand-gesture based HCI applications are illustrated below:

- **Augmented reality and virtual reality:** Hand gestures can be very useful for realistic manipulations of virtual objects in virtual environments [38] and as an interface for virtual gaming [39]. Many problems like detection, registration and tracking can be solved using augmented reality techniques [40].

- **Sign language recognition:** Hand gestures are useful for sign language recognition for the deaf-mute community [37]. The system mainly acts as an interpreter between the deaf/mute and others.

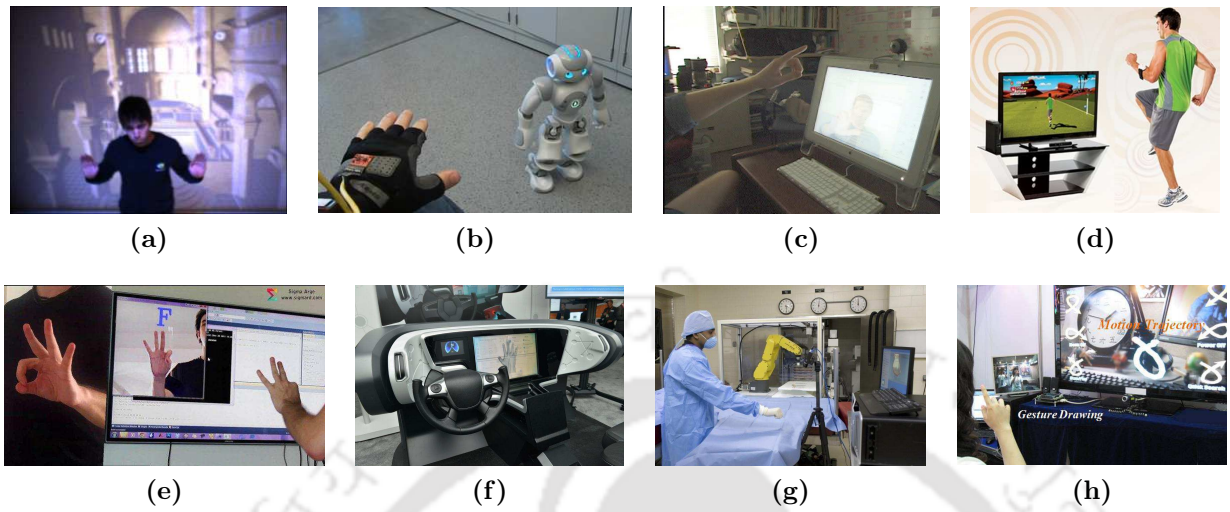
- **Vehicle monitoring and vehicle control:** Gesture-based interfaces may be used to operate a vehicle [46], and also for driver monitoring [45].

- **Healthcare & Medical assistance:** Gesture-based interfaces have many applications in healthcare and medicine, for example, MRI navigation in the operating room [34], and medical volume visualization tasks, browsing radiology images may be some of the possible applications. Gestures can also be used to train physicians in robotic surgery [43] and medical assistance for physically disabled persons, including hand gesture-based wheelchair control [44].

- **Information retrieval:** Gesture-based interfaces can also be used for day-to-day information retrieval from the internet [50].

- **Education:** Gesture interfaces for controlling presentations (*e.g.,* powerpoint®) is helpful for teachers [47]. Gesture-based interfaces can be used for window menu activation.

- **Desktop, television control and tablet PC applications:** Gesture interfaces can be useful in controlling desktop, television, etc. and also for tablet PC applications [42].



(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 1.6: Applications of hand gesture recognition systems: (a) Virtual reality, (b) Gesture-based interaction with robots, (c) Desktop computing application, (d) Virtual computer games using gesture, (e) Sign language recognition, (f) Vehicle control, (g) Gesture controlled robotic surgery and (h) Television and desktop controlling.

1.4 Major Challenges in VGR Systems

The ability of computers to recognize hand gestures visually is essential for the future development of vision-based HCI. Static gesture recognition or pose estimation of the isolated hand, in constrained conditions, is roughly a solved problem. However, vision-based recognition of hand gestures, especially dynamic hand gestures, poses an onerous difficult interdisciplinary challenge mainly for three reasons [51]:

- Hand gestures are diverse, have multiple meanings, and vary spatio-temporally;
- The human hand is a complex non-rigid object making it difficult to recognize; and
- Computer vision itself is an ill-posed problem.

A gesture recognition system relies on a series of subsystems, as explained previously. Because the subsystems are connected in series, the overall accuracy of the system is dependent on the accuracy of each subsystem. Thus, overall performance is highly affected by a subsystem that is the "weakest link". Gesture recognition has applications ranging from sign language to medical assistance to virtual reality. But all these applications are dependent on the ability of the device to read gestures efficiently and correctly. Applications using the human hand

as a human-computer interface motivate researchers for hand gesture recognition. The major challenges present in the process of hand gesture recognition are - constraints related to segmentation, problems in spotting the hand gestures perfectly in a continuous stream of gestures, problems related to two-handed gesture recognition, difficulties associated with extracted features and difficulties related to the articulated shape of the hand.

- **Challenges in segmentation:** Accurate segmentation of the hand or the gesturing body part from the captured videos or images still remains a challenge in computer vision for many constraints like illumination variations, background complexity, and occlusion. The variations in illumination affect the accuracy of skin color segmentation methods immensely. Poor illumination may change the chrominance properties of the skin colors, and the skin color will appear different from the original color. Biplab *et al.* has used a fusion-based image specific model for skin segmentation to handle the problem of segmentation under varying illumination conditions [52]. A major challenge in gesture recognition is the proper segmentation of skin-colored objects (*e.g.*, hands, face) against a complex static background. The accuracy of skin segmentation algorithms is limited because of objects in the background that are similar in color to human skin. Skin-colored objects present in the background increase false positives. Pisharady *et al.* used biologically inspired features like Gabor wavelet to handle the problem of complex background [16]. Another major challenge is mitigating the effects of occlusion in gesture recognition. Not only may the hand occlude itself, but one hand may occlude the other during two-handed gestures. Both kinds of occlusion affect the appearance of the hand, thus hindering in gesture recognition. Multiple camera-based systems are one solution for this problem [53], but these devices are not purely accurate. View-invariant 3D models or depth measuring sensors can provide some more insight into this problem.

- **Difficulties related to the articulated shape of the hand:** The accurate detection and segmentation of the gesturing hand is significantly affected by variations in illumination and shadows, presence of skin-like colors in the background, occlusion, background complexity, and different other factors. The complex articulated shape of the hand makes it harder to model the appearance of the hand for both static and dynamic gestures. Moreover, in the case of dynamic or trajectory-based gestures, the tracking of physical

movement of the hand is quite challenging due to the varied size, shape and color of the hand. Generally, it is expected that a generic gesture recognition system should be invariant to the shape, size and appearance of the gesturing body-part.

- **Difficulties associated with extracted features:** It is generally not recommended to consider all the image pixel values in a gesture image/video as the feature vector. This will not only be time-consuming but also it would take a great many examples to span the space variation, particularly if multiple viewing conditions and multiple users are considered. The standard approach is to compute some features from each image and concatenate these as a feature vector to the gesture model. A gesture model should consider both the spatial and temporal characteristics of the hand and its movements. No two samples of the same gesture will result in exactly the same hand and arm motions or the same set of visual images *i.e.*, gestures suffer from spatio-temporal variation. There exists spatio-temporal variation when a user performs the same gesture at different times. Every time the user performs a gesture, the shape and the speed of the gesture generally vary. Even if the same person tries to perform the same sign twice, a small variation in speed and position of the hands may occur. Therefore, extracted features should be rotation-scaling-translation (RST) invariant. But various image processing techniques have their own constraints to produce RST-invariant features. Another difficulty is that the processing of a large amount of image data is time-consuming and so real-time recognition may be difficult.

- **Gesture spotting problem:** Gesture spotting means locating the starting point and the endpoint of a gesture in a continuous stream of gestures. Once gesture boundaries have been determined, the gesture can be extracted and classified. But spotting meaningful patterns from a stream of input gestures isa highly difficult task mainly due to two aspects of signal characteristics: segmentation ambiguity and spatiotemporal variability. For sign language, the recognition engine must support natural gesturing to enable the user's unrestricted interaction with the system. Because non-gestural movements often intersperse a gesture sequence, these movements should be removed from the video input before the gesture sequence is identified. Examples of non-gestural movements include *"movement epenthesis"*, the movement that occurs between gestures and *"gesture co-articulation"*, the

effect of the end of a sign and the beginning of the next sign has on each other. In some cases, a gesture could be similar to a sub-part of a longer gesture, referred to as the *"sub-gesture problem"* [10]. Though static hand gesture recognition problem [54–58] is almost a solved one, but till date, there are only a handful works [59–64] that deal with these three problems of continuous hand gesture recognition system.

- **Problems related to two-handed gesture recognition:** The inclusion of two-handed gestures in a gesture vocabulary can make human-computer interaction more natural and expressive for the user. It can greatly increase the size of the vocabulary because of the different combinations of left and right-hand gestures. Previously proposed methods include template-based gesture recognition with motion estimation [65] and two-hand tracking with colored gloves [66]. Despite its advantages, two-handed gesture recognition faces some major difficulties:
  - **Computational complexity:** The inclusion of two-handed gestures can be computationally expensive because of their complicated nature.
  - **Inter-hand overlapping:** The hands can overlap or occlude each other, thus impeding recognition of the gestures.
  - **Simultaneous tracking of both hands:** The accurate tracking of two interacting hands in a real environment is still an unsolved problem. If the two hands are clearly separated, the problem can be solved as two instances of the single-hand tracking problem. However, if the hands interact with each other, it is no longer possible to use the same method to solve the problem because of overlapping hand surfaces [67].

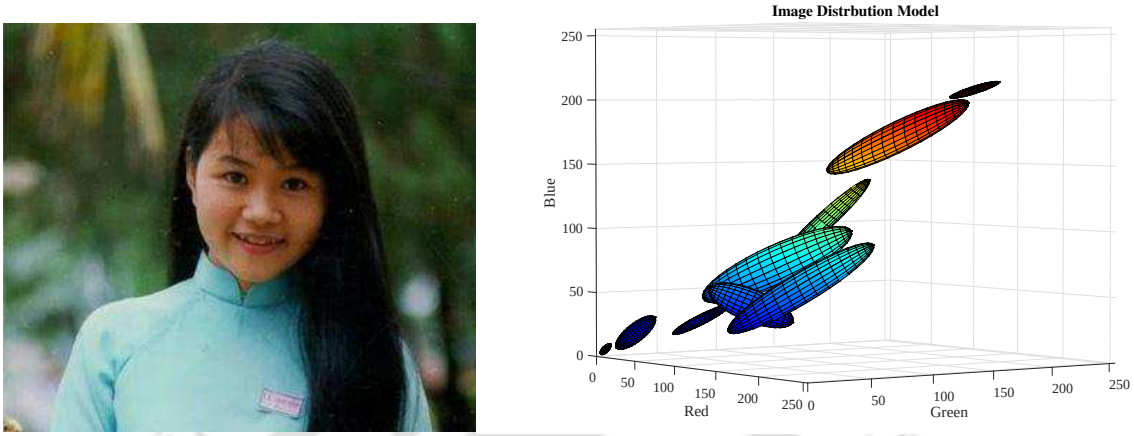
1.5 Research Motivation

From the previous section, it is evident that a significant amount of work is needed for realizing an efficient hand gesture recognition system under various environmental conditions. In this research work, an attempt has been made to recognize the trajectory-based dynamic hand gestures having only global motion with different spatio-temporal and motion characteristics. The motivation behind this dissertation is to address some of the major issues related

to trajectory-based hand gestures mentioned in the previous section. One major issue is the detection and segmentation of the hand region from the gesture videos. So, the primary goal of the dissertation is around the detection of the hand in different gesture videos especially to give some insight into the problems related to hand segmentation. Additionally, detecting hand regions in videos with different shapes, sizes and appearances is another important task. The development of an effective hand gesture recognition system considering all the requirements into one algorithm is a major challenge. Accordingly, this thesis looks into several aspects using chromatic as well as trajectory information of the hand region and aims at developing suitable algorithms that can take care of at least some of the limitations of the existing methods. The motivations behind this research work are given below:

(i) Accurate segmentation of the hand from the captured images/videos remains a challenge for many preoccupied constraints like illumination variations, background complexity, occlusion etc. So, in our first work, we would try to mitigate these problems as far as possible with the help of both colors as well as motion information.

Typically a chromatic and/or textural discrimination is observed between the skin and the non-skin regions of an image. Moreover, human skin color does not fall randomly in a given color space but clustered at a small area in the color space [68] as shown in Fig. 1.7. So, in skin-color-based segmentation, the color needs to be represented in a color space where the skin color is most compact so that it can be modelled into a single class tightly. The RGB color-space is not perceptually uniform, which means distances in the space do not linearly correspond to human perception. In addition, RGB color space does not separate luminance and chrominance parts, and the R, G, and B components of RGB are highly correlated. The luminance of a given RGB pixel is a linear combination of the R, G, and B values. Therefore, changing the luminance of a given skin patch affects all the R, G, and B components. Hence, there is a need for color-spaces with an adjustable range to suit the color of the hand for effective and efficient segmentation. Even after the use of proper color-spaces, there may be cases when accurate segmentation is not possible due to reasons like the variations in illumination and presence of skin-colored objects, occlusion, complex background *etc.* In such cases, there is a requirement of some additional techniques other than skin segmentation which will be explored here.



(a) **(b)**

Figure 1.7: Illustration of image pixel distribution: a) Original image, b) Image pixel distribution.

Skin segmentation becomes very challenging when the proper color information is not available or when there are multiple skin-colored scenes in the background. Basically, when prior knowledge of the color of the moving object is not available, pixel-level change can provide powerful motion-based cues for detecting and localizing objects. Here along with skin-color segmentation, motion information can also be used to segment the moving hand. Tracking of the segmented region in blurring scenes or in presence of occluding objects is another challenge. Some additional measures can be incorporated to handle such situations.

(ii) Automatic detection of moving objects is a key motive in visual surveillance and tracking system. In the process of hand gesture recognition, proper detection and tracking of the moving hand in a cluttered background plays an important role due to the varied shape and size of the hand. Segmentation and tracking become a very challenging task due to the articulated shape of the hand and the presence of skin-color-like objects in the background. The majority of skin detection algorithms use skin color as a primary feature. However, the use of some other features like texture information or depth information along with colour features generally improves segmentation accuracy. But the availability of this information is not guaranteed with each gesture image/video. For trajectory-based gestures, motion information can be another added information useful for segmentation.

Basically, when prior knowledge of the moving hand like appearance, color and shape is not known, motion-based cues can still provide effective information for detecting and localizing objects. To get rid of the inherent problems of skin segmentation, this information can be used in tracking trajectory-based gestures with varied shapes and sizes of the hand.

(iii) But in some cases segmentation becomes an unavoidable process, and then researchers generally opt for different types of segmentation techniques like semantic or instance segmentation. Popular deep learning methods like *convolutional neural network* (CNN) have come out as a magnificent tool for classification, whose benefit can also be exploited for image segmentation tasks. Recently, attention mechanisms are widely used in computer vision to extract better visual features. Attention not only tells where to focus, but it also improves the representation of interests. Moreover, benefiting from the attention mechanism, segmentation can be made more efficient and effective. Here, the objective is to develop a gesture recognition scheme applicable to both static as well as dynamic hand gestures.

1.6 Organization of the Thesis

To address the issues mentioned in the previous section, this thesis work is organized into six chapters. The content of each chapter is summarized as follows:

- Chapter 2 reviews several existing methods for hand gesture recognition under various conditions. The review section is presented in three parts according to the stages of a VGR system: acquisition & pre-processing, gesture representation & feature extraction, and recognition. The recognition section is again discussed in three subsections: conventional methods on RGB data, depth-based methods on RGB-D data, and deep-learning-based methods. The summary of the review and the scope for this thesis work is discussed in the last section of the chapter.

- In Chapter 3, a hand gesture recognition framework for isolated dynamic gestures using a convolutional neural network (CNN) is presented. In the preprocessing step, a two-level segmentation process with compensation for the illumination variations and a

double-tracking system with occlusion handling ability are used for tracking the gesture trajectory. Through this pre-processing step, each isolated dynamic gesture is converted into single image consisting of the contour of the gesture trajectory which we call hand-trajectory-based-contour-images. The feature learning capability of CNN architecture has been used here and it has shown outstanding results on three different datasets.

- In Chapter 4, a two-stream fusion model for hand gesture recognition is proposed. The two-stream network consists of two layers - a 3D convolutional neural network (C3D) that takes gesture videos as input and a 2D-CNN that takes OFMT images as input. C3D has shown its efficiency in capturing spatio-temporal information of a video. As input to the second layer, a motion template guided by optical flow (OFMT) is proposed which helps to eliminate irrelevant gestures providing additional motion information. Though each stream can work independently, they are combined with a fusion scheme to boost the recognition results. We have shown the efficiency of the proposed two-stream network on two databases. Here the major contribution is OFMT images that can track the moving hand irrespective of the shape, size, and color of the hand.

- In Chapter 5, a deep-learning method is used for static and dynamic hand gesture recognition. The ability to discern the shape of hands can be a vital issue in improving the performance of static hand gesture recognition. Segmentation itself is a very challenging problem having various constraints like illumination variations, complex background *etc.* The objective of this work is to incorporate the perception of semantic segmentation into a classification problem and make use of the deep neural models to achieve improved results. Attention-based methods have been proved to be effective ways to obtain important contextual information in different segmentation methods like semantic segmentation. This work utilizes the attention-based UNET architecture to obtain the semantically segmented mask of the input, which is then given to a classifier for recognition. The data augmentation process is used in preprocessing to generate a sufficient number of training images for training the CNN-based model and it has been able to achieve a significant and improved recognition performance.

- Finally, we draw our conclusions in Chapter 6 by highlighting the strengths and shortcomings of our schemes and outlining possible extensions.

2

Vision-based Gesture Recognition System - A Review

This chapter reviews the various methods presented in the literature pertaining to hand gesture recognition. These methods are presented in different sections depending on the similarity in the approach taken. Special attention is given to classify the schemes/approaches at various stages of the gesture recognition system for a better understanding of the topic to facilitate further research in this area. A detailed discussion is provided on feature extraction and major classifiers in current use including deep learning techniques. Finally, the chapter concludes with a brief summary of the literature review and the scope for the present work.

2.1 Overview of Vision-based Hand Gesture Recognition (VGR) System

The primary task of vision-based interfaces is to detect and recognize visual information for communication. A vision-based approach is more natural and convenient than other sensor-based approaches. It is easy to deploy and can be used anywhere within a camera's field of view. The straightforward approach to vision-based gesture recognition (VGR) is to acquire visual information about a person in a certain environment and try to extract the necessary gestures. A typical VGR system constitutes three stages that have to be performed in a sequence:

(i) ***Acquisition, detection and pre-processing.***

(ii) ***Gesture representation and feature extraction.***

(iii) ***Recognition.***

2.1.1 Acquisition, detection and pre-processing

Gesture acquisition involves capturing images or videos using imaging devices. The detection and classification of moving objects present in a scene is an important research area in action/gesture recognition. The most important research challenges are segmentation, detection, and tracking of moving objects from a video sequence. The detection and pre-processing stage mainly deals with localizing gesturing body parts in images or videos. Since dynamic image analysis consists of all these subtasks, so this very portion can be subdivided into segmentation and tracking or combining both of them together.

(i) *Segmentation:* Segmentation is the process of partitioning images into multiple distinct parts and thereby finding the region of interest (ROI), which is hand in our case. Accurate segmentation of hand or body parts from the captured images remains a challenge in computer vision for many preoccupied constraints such as illumination variations, background complexity, and occlusion due to the articulated shape of the hand. Most of the segmentation techniques can be broadly classified as follows (Fig. 2.1): a) Skin color-based segmentation b) Region-based c) Edge-based d) Otsu thresholding etc. The easiest way to detect skin regions of an image is through an explicit boundary specification for skin

color in a specific color space *e.g.*, RGB [69], HSV [70], YCbCr [71] or CMYK [12]. Many researchers dropped the luminance component and used only the chrominance component since chrominance cues contain skin color information and it is less sensitive to illumination changes in the hue-separation space as compared to RGB color space [72]. However, color cues show variations in the skin color in different illumination conditions, and also skin color changes with the change in human races, and so segmentation is restricted due to the presence of skin-colored objects in the background. Occlusion also leads to many issues in the segmentation process. In order to improve the detection accuracy, many researchers have used parametric and non-parametric model-based approaches for skin detection. For example, Yang *et al.* [73] used a single multivariate Gaussian to model skin color distribution. But, skin color distribution possesses multiple co-existing modes. So, the Gaussian mixture model (GMM) [74] is more appropriate than a single Gaussian function. Lee and Yoo [75] proposed an elliptical modeling-based approach for skin detection. The elliptical modeling has less computational complexity as compared to GMM modeling. However, many true skin pixels may get rejected if the ellipse is small. Whereas if the ellipse is sufficiently large, many non-skin pixels may be detected as skin pixels. Out of non-parametric model-based approaches for skin detection Bayes skin probability map (Bayes SPM) [76], self-organizing map (SOM) [77], k-means clustering [78], artificial neural network (ANN) [79], support vector machine (SVM) [69], random forest [80] are noteworthy. The region-based approach involves region growing techniques, region splitting and region merging techniques. Rotem *et al.* [81] combined patch-based information with edge cues under a probabilistic framework. In an edge-based technique, basic edge-detecting approaches like Prewitt filter, Canny edge detector, Hough transforms are used. Otsu thresholding is a clustering-based image thresholding method that converts a gray level image to a binary image using any edge detecting or tracking technique so that we have only two objects *i.e.* one is hand and the other is background [9]. In the case of videos, all these methods can be applied with dynamic adaptation.

(ii) *Tracking:* Tracking can also be considered as a part of pre-processing in the hand detection process as both tracking and segmentation together help to extract the hand from the background. Though skin-segmentation is one of the most preferred methods for segmen-

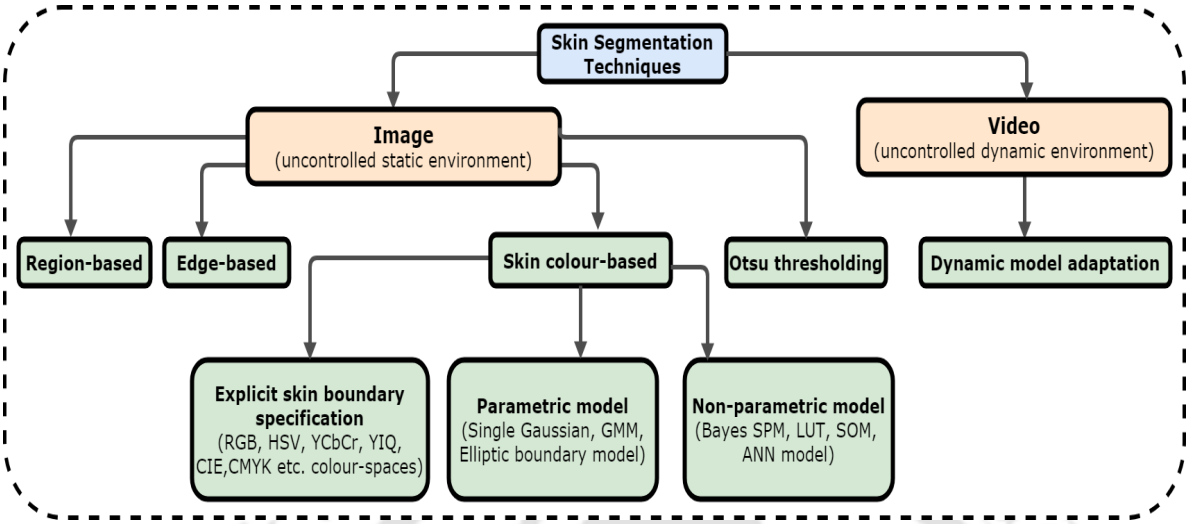


Figure 2.1: Different skin segmentation schemes.

tation or detection, still it is not so effective for various constraints like scene illumination variations, background complexity, and occlusion [72]. Basically, when prior knowledge of moving objects like appearance and shape is not known, pixel-level change can still provide effective motion-based cues for detecting and localizing objects. Various approaches for moving object detection using pixel-level change can be background subtraction, interframe difference, or three-frame difference [82]. Stabilized background detection is always a costly matter making it vulnerable for long and varied video sequences [82]. Apart from this, the choice of temporal distance between frames is a tricky question. It basically depends on the size and speed of the moving object. Though interframe difference methods can easily detect motion, it shows poor performance in localizing the object. The three-frame difference [83] approach uses previous, current and future frames to localize the object in the current frame. The use of future frames introduces a lag in the tracking system, but this lag is acceptable only if the object is far away from the camera or moves slowly relative to the high capture rate of the camera.

Tracking of the hand can be difficult as the movement of the hand can be very fast and its appearance can change vastly within a few frames. In such cases, model-based algorithms like mean-shift [84], Kalman filter [85], particle filter [86] are some of the methods used for tracking. The mean-shift is a purely non-parametric mode-seeking algorithm that iteratively shifts a data point to the average of data points in its neighbourhood

(similar to clustering). However, tracking often converges to an incorrect object when the object changes its position very quickly in the two neighbouring frames. Because of this problem, a conventional mean-shift tracker fails to position a fast-moving object. [72, 87] modified the mean-shift algorithm to continuous adaptive mean-shift (CAMShift) where the window size is adjusted so as to fit the gesture area reflected by any variation in the distance between the camera and the hand. Though CAMShift performs well with objects that have a simple and constant appearance, it is not robust in more complex scenes. The motion model for the Kalman filter is based on the assumption that the velocity is relatively small when objects are moving, and therefore, it is modeled by a zero mean and low variance white noise. One limitation of the Kalman filter is the assumption that the state variables are based on Gaussian distribution, and thus the Kalman filter will give incorrect estimations of state variables that do not follow a linear Gaussian environment. The particle filter is generally a better method than the Kalman filter because it can consider non-linearity and non-Gaussianity. The main idea of the particle filter is to apply a weighted sample particle set to approximate the probability distribution, i.e., the required posterior density function is represented by a set of random samples with associated weights and estimation is done on the basis of these samples and weights. Both Kalman filter and particle filter have the disadvantage of the requirement of previous knowledge in modeling the system. Kalman filter or particle filter can be combined with the mean shift tracker for precise tracking. In [88], authors have detected hand movement using Adaboost with the histogram of gradient (HOG) method.

(iii) *Combined segmentation and tracking:* Here the first step is object labeling by segmentation and the second step is object tracking. Accordingly, an update for tracking is performed by calculating the distribution model with different label values. Skin-segmentation and tracking together can give quite a good performance [89], but researchers have adopted other methods too where skin-segmentation is not so efficient.

2.1.2 Gesture representation and feature extraction

According to its spatio-temporal properties, gestures are broadly classified as static or dynamic. Static gestures are defined by the pose or orientation of a body part in the space (*e.g.*,

hand pose); that's why sometimes simply called as posture. Whereas dynamic gestures are defined by trajectory or temporal deformation of body parts (*e.g.*, shape, position, motion). Again dynamic gestures can of single isolated trajectory type or continuous type, occurring in a stream, one after another.

2.1.2.1 Gesture representation

To recognize a gesture, it must be represented using a suitable model (Fig. 2.2). Based on feature extraction methods, the following are the types of gesture representations: model-based and appearance-based.

(i) *Model-based:* Here gestures can be represented using either a 2D model or 3D model. The 2D model basically depends on either various color models like RGB, VSB, YCbCr, etc, or silhouettes or contours obtained from 2D images. The deformable Gabarit model depends on the formation of active deformable contouring. On the other hand, 3D models can be categorized into mesh model [90], geometric model, volumetric models and skeletal models [91]. The volumetric model represents hand gestures with high accuracy. The skeletal model reduces the hand gestures into a set of equivalent joint angle parameters with segment length. For example, Rehg and Kanade [92] used a 27 degree-of-freedom (DOF) model of a human hand in their system called 'Digiteyes'. Local image-based trackers are employed to align the projected model lines to the finger edges against a solid background. The work of Goncalves *et al.* [93] promoted three-dimensional tracking of the human arm against a uniform background using a two cone arm model and a single camera. One major disadvantage of model-based representation using a single camera is self-occlusion [93] that frequently occurs in articulated objects like a hand. To avoid it, some systems employ multiple/stereo cameras and restrict the motion to small regions [92]. But it also has its own disadvantages like precision, accuracy, etc [51]. Generally, static gestures can be represented using a 2D model or 3D model.

(ii) *Appearance-based:* The appearance-based model tries to identify gestures either directly from visual images/videos or from the features derived from the raw data. Parameters of such models may be either the image sequences or some features derived from the images which can be used for hand-tracking or simple gesture classification. For example,

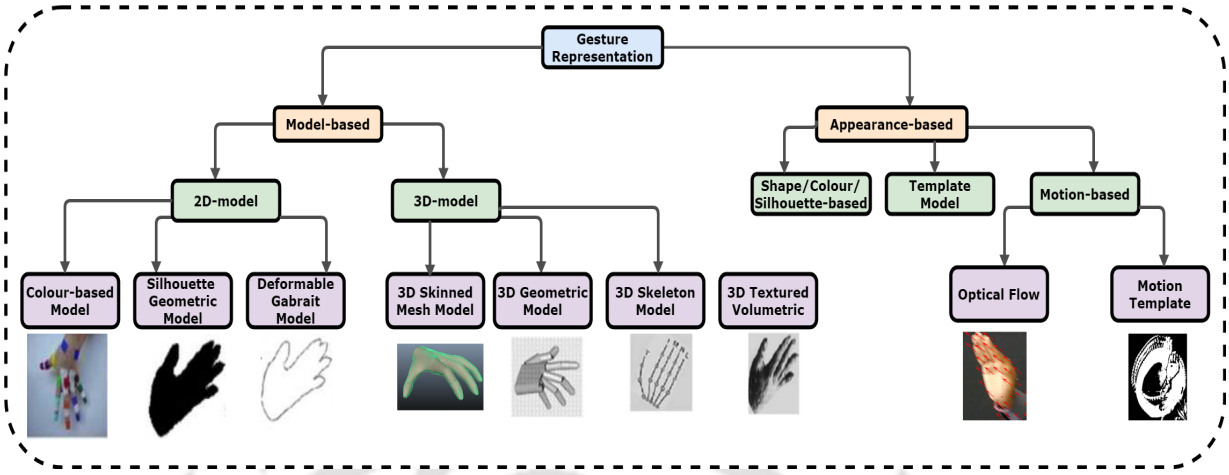


Figure 2.2: Different hand models for hand gesture representation.

Wilson and Bobick [94] presented results using actions, mostly hand gestures, where the actual gray-scale images (with no background) are used in action representation. Rather than using raw gray-scale image, Yamato *et al.* [95] used body silhouettes, and Akita [96] employed body contours/edges. Yamato *et al.* [95] utilized low-level silhouettes of human actions in a Hidden Markov Model (HMM) framework, where binary silhouettes of background-subtracted images are vector quantized and used as input to the HMMs. In Akita's work [96], the use of edges and some simple two-dimensional body configuration knowledge were used to determine the body parts in a hierarchical manner (first, find legs, then head, arms, trunk) based on stability. While using two-or-three dimensional structural information, there is a requirement of individual features or properties to be extracted and tracked from each frame of the image sequence. Hence, motion understanding is really accomplished by recognizing a sequence of static configurations that require previous detection and segmentation of the object. And since early days, sequential state-space models such as generative hidden Markov models (HMMs) [97] or discriminative conditional random fields (CRFs) [98] have been proposed to model dynamics of action/gesture videos. Temporal ordering models like dynamic time warping (DTW) [10] have also been applied in the context of dynamic action/gesture recognition where matching of an incoming gesture is done to a set of pre-defined representations.

In most literature, *e.g.* [99], it is mentioned that gestures are represented by either model-based or appearance-based model. The motion-based methods are also generally included in the

appearance-based methods. But here we want to discuss the motion-based methods separately. This is because the shape and appearance of the body/body-part depends on many factors *e.g.* illumination variations, image resolution, skin color, clothing etc. But estimation of motion is invariant to shape and appearance (at least in theory) and can be used directly to describe human gesture/action [100]. Optical flow and motion-templates are the two main motion-based representation methods.

(i) **Optical flow:** Optical flow is the apparent motion or displacement of objects/pixels as perceived by an observer. Optical flow indicates the change in image velocity of a point moving in the scene, also called as motion field. Here the goal is to estimate the motion field (velocity vector) which can be computed from horizontal and vertical flow fields. Ideally, the motion field represents the $3D$ motion of the points of an object across $2D$ image frames for a definite frame-interval. Out of different optical flow techniques found in the literature, the most common methods are: (a) Lucas-Kanade [101] (b) Horn-Schunk [102] (c) Brox 04 [103] and (5) Brox 11 [104] (d) Farneback [105]. The choice of the optical flow method primarily depends on the power of the resulting histogram of optical flow (HOF) or motion boundary histogram (MBH) descriptor. HOF gives the optical flow displacement vectors in horizontal and vertical directions. The intuitive idea of MBH is to represent the oriented gradients computed over the vertical and the horizontal optical flow components. Once horizontal and vertical optical flow components are obtained, histograms of oriented gradients are computed on each image component. The outcome of this process is a pair of horizontal (MBHx) and vertical (MBHy) descriptors. Laptev *et al.* [106] implemented a combination of HOG-HOF for learning realistic human action from movies. [107] also proposed to calculate changes of optical flow that focuses on optical flow differences between frames (motion boundaries). Yacoob and Davis [108] used optical flow measurements to track predefined polygonal patches placed on interest regions for facial expression recognition. [109] presented an integrated approach where the optical flow is integrated frame-by-frame over time by considering the consistency of direction. In [110], optical flow was used to detect the direction of motion along with the RANSAC algorithm which in turn helped to further localize the motion points.

(ii) **Motion-templates:** Basically motion-templates are the compact representation of a

gesture video where the motion information of a gesture is encoded into a single image. These templates are compact representations of videos useful for video analysis where a single image summarizes the appearance and dynamics of the whole video sequence. Hence, these images are named as motion fused images or temporal templates or motion-templates. There are three widely used motion fusion strategies [100]: motion-energy-image (MEI) and motion-history-image (MHI) [2, 111], dynamic images (DI) [3] and methods based on PCA [4]. The main disadvantage of all the three motion-template methods is in representing static gestures or when a user remains static while performing some gesture/action in the video.

- ***MEI-MHI:*** MEI and MHI are two major motion-templates proposed to represent the motion evaluation of an object in a video where all the frames in the video sequence are projected onto one image across the temporal axis. This is the starting of a novel approach where people thought of converting a complete dynamic video/video-frames with motion templates into a single image. MEI represents where motion has happened in a sequence of frames; whereas MHI represents how an object is moving (Fig.2.3). MEI describes the motion-shape and spatial distribution of motion, and MHI is the function of the intensity of motion of each pixel at that location. Moreover, MEI can be generated by thresholding the MHI above zero. So, MEI-MHI basically squeezes the time scale of human actions/gestures by encoding a bundle of frames into a single image. To make the system view-invariant, authors of [2] have used seven Hu moments [112] which are translation-and scale-invariant. For each view of each movement, a statistical model of the moments (mean and covariance matrix) is generated for both the MEI and MHI. Mahalanobis distance is calculated between the moment description of the input and each of the known movements to recognize an input movement. Gray-scale MHI is sensitive to the direction of motion, unlike the MEIs, and hence better suited for discriminating between actions of opposite directions (e.g., 'standing up' versus 'sitting down'). Several modifications are there on MEI-MHI-based implementation in the literature [111]. Though the MEI-MHI method is simple and computationally not expensive, still it has some crucial problems [111]. First, it fails to separate the motion information when there

is self-motion-occlusion or overwriting of prior information like if a person sits down, and then stands up. Second, the change of the standing position of a person while executing an action may produce false recognition for an action. Third, the MEI-MHI method is not suitable for dynamic background with its basic representation (which is based on background subtraction or image differencing approaches). There is always a requirement of having stationary objects in the background. Also, it is unable to discriminate among similar motions. So, it is always crucial of employing the MEI-MHI images for recognition purposes, because the MEI-MHI method takes into account the global motion calculation of the image frames which is dependent on the variances in movement duration. MEI-MHI is always a choice of representation for action recognition, only when temporal segmentation is available, actors are fully visible and can be separated from each other. MEI-MHI can be implemented by the following algorithm and the outcome is shown in Fig.(2.3)

MEI-MHI Algorithm [2]:

- **Image sequences**

$$I(x, y, t) = (I_1, I_2, ..., I_n). \tag{2.1}$$

- **Image binarization**

$$B(x, y, t) = |I(x, y, t) - I(x, y, t-1)|. \tag{2.2}$$

$$\text{where, } B(x, y, t) = \begin{cases} 1 & \text{if } B(x, y, t) > \xi \\ 0 & \text{otherwise} \end{cases}$$

- **MEI**

$$E_\tau(x, y, t) = \bigcup_{i=0}^{\tau-1} B(x, y, t-i). \tag{2.3}$$

- **MHI**

$$H_\tau(x, y, t) = \begin{cases} \tau & \text{if } B(x, y, t) = 1 \\ max(0, H_\tau(x, y, t-1) - \delta & \text{otherwise} \end{cases} \tag{2.4}$$

where τ decides the temporal extent of the motion (in terms of frames) and δ is the

decay parameter.

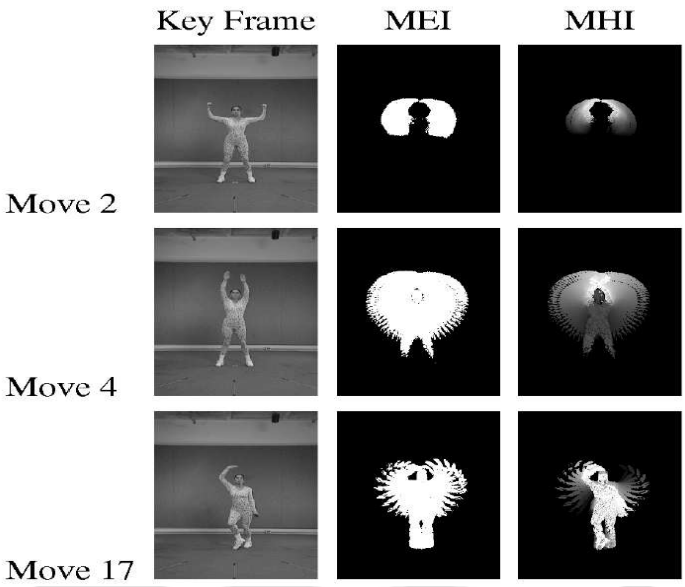


Figure 2.3: MEI and MHI example from [2].

- ***Dynamic images:*** Dynamic image (DI) [3] (shown in Fig(2.4)) is a novel video-wide temporal evolution representation. It basically captures the video-wide temporal dynamics of a video converting into a single image, suitable for action/gesture recognition. It is observed that if the execution time of actions varies greatly, the temporal ordering is typically preserved. So dynamic image generally uses a technique called rank pooling which is the process of ranking the frame content used to capture the video-wide temporal evolution and pooling the whole video into a single image [113]. Major advantages of rank pooling are: (a) rank pooling is useful and robust for encoding video-wide, temporal information, (b) since it does not extract any trajectories or other more sophisticated features, so it is computationally not expensive. So this novel dynamic image is a simple, efficient, compact, and very powerful method to extract video-wide temporal evolution into a single image, particularly useful in the context of deep learning. Another notable advantage of DI compared to other classical methods is that it performs quite well for both fast/slow and short/long actions. Normally classical methods are applicable to only slow (< 30 frames per second) and short (only a few seconds) videos. So in such cases, the dynamic image method is applicable where there exist characteristic motion patterns

Figure 2.4: Dynamic images summarizing the actions and motions that happen in (from left to right and top to bottom): blowing hair dry, band marching, balancing on beam, golf swing, fencing, playing the cello [3].

and dynamics [3]. In [114], authors have mentioned MHI as a direct competitor to the dynamic image method. Here, authors have shown that DIs provide a more detailed representation of the videos, as the range of intensity values is not limited to the number of frames as in MHIs. Second, DIs are more robust to moving viewpoints, long-range and background motion. Finally, in contrast to DIs, MHIs can only represent the motion gradient in object boundaries. In [115], authors have presented a hierarchical rank pooling method which consists of a network of non-linear operations and rank pooling layers. It has shown substantial performance improvement over other temporal encoding and pooling methods such as max-pooling [116], average pooling [116], rank pooling [114], temporal pyramids [116], and LSTMs [117].

Dynamic Image (DI) Algorithm [3]:

- **Image sequences**

$$V = (I_1, I_2, ..., I_T). \tag{2.5}$$

- **Time-average feature**

$$\psi(t) = \frac{1}{t}\sum_{\tau=1}^{t}\psi(I_\tau) \tag{2.6}$$

– **Dynamic image**

$$d^* = (I_1, I_2, ..., I_T; \psi) = argminE(d). \tag{2.7}$$

– **Optimization problem**

$$E(d) = \frac{\lambda}{2}||d||^2 + \frac{2}{T(T-1)} \sum max(0, 1 - S(k|d) + S(l|d)) \tag{2.8}$$

where, $k > l \Rightarrow S(k|d) > S(l|d)$ *i.e.* later times are given larger score.

- ***PCA and Robust PCA using PCP method:*** The use of principal component analysis (PCA) as a foreground-detection technique is well-known in various applications like object detection [118], pedestrian detection [119], video surveillance. But there are only few instances when PCA-based method is used for gesture [4] (shown in Fig(2.5) or activity [120] recognition. Robust PCA is a matrix factorization method that decomposes the input matrix I into the sum of two matrices *i.e.* $I = L + S$, where L is low-rank matrix and S is sparse matrix. The background sequence is then modeled by a low-rank subspace that generally changes gradually over time, while the moving foreground objects are constituted by the correlated sparse matrix. This is done by solving the following optimization problem called principal component pursuit (PCP)

$$\min ||L||_* + \lambda||S||_1 \quad s.t. \quad L + S = I \tag{2.9}$$

where $||\ ||_*$ and $||\ ||_1$ are the nuclear norm (which is the l_1-norm of singular value) and l_1-norm, respectively, and $\lambda > 0$ is an arbitrary balanced parameter. The major advantages of PCA-based methods are [4]: (a) it performs quite well in both RGB as well as depth video and (b) it is particularly well suited for the case when motion happens in different location of the image stream.

2.1.2.2 Feature extraction

After gesture modeling, a set of features needs to be extracted for gesture recognition. Features for static gestures are derived from image information like color and texture, or pose

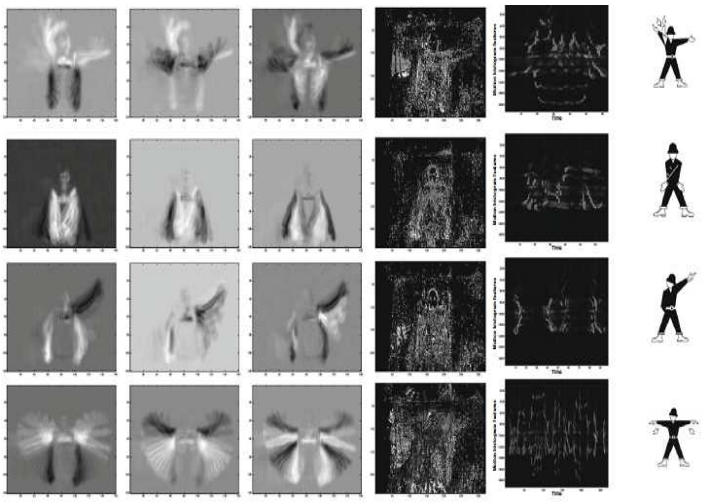


Figure 2.5: Principal motion components for the gesture dataset of helicopter signals: Each row is associated with a different gesture, the first three columns of each row display top 3 principal motion components of the gesture; columns 4-6 show the MHI, motion maps and a visual description of the corresponding gesture, respectively [4].

information like orientation, shape, etc. There are three basic features for spatio-temporal patterns of dynamic gesture namely location, orientation and velocity [121], based on which various features or descriptors are used in the state-of-the-art methods. For example, features are based on motion and/or deformation information like position, skewness, and the velocity of hands. Features for dynamic hand gestures are spatio-temporal patterns. A static hand gesture may be viewed as a special case of a dynamic gesture with no temporal variation of the hand shape and position. A gesture model should consider both the spatial and temporal characteristics of the hand and its movements. No two samples of the same gesture will result in exactly the same hand and arm motions or the same set of visual images *i.e.*, gestures suffer from spatio-temporal variation. There exists spatio-temporal variation when a user performs the same gesture at different times. Every time the user performs a gesture, the shape and the speed of the gesture generally vary. Even if the same person tries to perform the same sign twice, a small variation in speed and position of the hands will occur. Therefore, extracted features should be rotation-scaling-translation (RST) invariant. Various features or descriptors are used in the state-of-the-art methods for VGR systems. These features can be broadly classified based on their method of extraction, such as spatial domain features, transform domain features, curve fitting-based features, histogram-based descriptors, and interest point-

based descriptors. Moreover, the classifier should be able to handle spatio-temporal variations. Recently, feature extraction techniques based on deep learning have often been applied for gesture recognition. Kong *et al.* [122] proposed a view-invariant feature extraction method using deep learning for multi-view action recognition. Table 2.1 gives a brief survey of the properties of different features used for both static and dynamic gesture recognition.

2.1.3 Recognition

The final stage of a VGR system is the recognition stage where a suitable classifier recognizes the incoming gesture parameters or features and groups them into predefined classes (supervised) or by their similarity (unsupervised). Here the process of hand gesture recognition has been tried to divide into some categories for easy understanding. And based on the type of input data and the method, hand gesture recognition process can be broadly categorized into three sections:

- Conventional methods on RGB data
- Depth-based methods on RGB-D data
- Deep learning techniques

2.1.3.1 Conventional methods on RGB data

Vision-based gesture recognition typically depends on three stages. The third stage of the gesture recognition module consists of a classifier, which classifies the input gesture. However, each classifier has its own advantages as well as limitations. Here we discuss the conventional methods of classification of static and dynamic gestures on RGB data.

- ***Static gesture recognition:*** Static gestures are basically finger-spelled signs in still images without any time frame. Unsupervised k-means and supervised k-NN, SVM, ANN are the major classifiers for static gesture recognition.
  - *k-means:* This algorithm is an unsupervised classifier that determines k centre points to minimize clustering error defined by the sum of the distances of all data points

Table 2.1: Features used for gesture recognition

Feature type	Examples	Static	Dynamic	Advantages	Limitations
Spatial domain (2D)	Fingertips location, finger direction, and silhouette [54]	✓	✓	• Easy to extract • Rotation invariant.	• Unreliable under occlusion or varying illumination. • Object view-dependent. • Distorted hand trajectory distorts MCC also.
	Motion chain code (MCC) [97, 98]		✓		
Spatial domain (3D)	Joint angles, hand location, surface texture and surface illumination [123]	✓	✓	• 3D modelling can most accurately represent the state of a hand, and thus can give higher recognition accuracy.	• Difficult to accurately estimate 3D shape information of a hand.
Transform domain	Fourier descriptor [124], DCT descriptor [125], Wavelet descriptor [126]	✓	✓	• RST invariant	• Not able to perfectly distinguish different gestures.
Moments	Geometric moments, orthogonal moments [55]	✓	✓	• Moments can be used to derive RST invariant global features.	• Moments are in general global features. So, moments cannot effectively represent an occluded hand.
Curve fitting-based	Curvature scale space [127]		✓	• RST invariant. • Resistant to noise.	• Sensitive to distortion in the boundary.
Histogram-based	Histogram of gradient (HoG) features [56]	✓	✓	• Invariant to geometry and illumination changes.	• Performance is not so satisfactory for images with a complex background and noise.
Interest point-based	Scale-invariant feature transform (SIFT) [128], Speeded up robust features (SURF) [129]	✓	✓	• RST and illumination invariant	• They are not the best choice for real-time applications because they are computationally expensive.
Mixture of features	Combined features [57]	✓	✓	• Incorporates the advantages of different types of features.	• Classification performance may degrade due to *curse of dimensionality*.

to their respective cluster centres. For a set of observations $(\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n)$, in a d-dimensional real vector space, k-means clustering partitions the n observations into a set of k clusters or groups $S = \{S_1, S_2, \ldots, S_k\}$ $(k \leq n)$ and their centers are given by -

$$\arg\min_{\mathbf{S}} \sum_{i=1}^{k} \sum_{\mathbf{x}_j \in S_i} \left\| \mathbf{x}_j - \mu_i \right\|^2 \tag{2.10}$$

The classifier randomly locates k cluster centres in the feature space. Each point in the input dataset is assigned to the nearest cluster centre, and their locations are updated to the average location value for each cluster. This process is then repeated until a stopping condition is met. The stopping condition could be either a user-specified maximum number of iterations or a distance threshold for the movement of the cluster centres. Ghosh and Ari [58] used a k means clustering-based radial basis function neural network (RBFNN) for static hand gesture recognition. In this work, k means clustering is used to determine the RBFNN centres.

- *k-nearest neighbours (k-NN):* k-NN is a non-parametric, supervised learning algorithm. Data in the feature space can be multidimensional. The training data consists of a set of labelled feature vectors. The number k determines how many neighbours (nearby feature vectors) influence the classification. Typically, an odd value of k is chosen for two-class classification. Each neighbour may be given the same weight or those closer to the input data may be given more weight (*e.g.*, by applying a Gaussian function). In uniform voting, a new feature vector is assigned to the class to which the plurality of its neighbours belongs. Hall *et al.* assumed two statistical models (Poisson and binomial) for the sample data to obtain the optimum value of k [130]. The k-NN can be used in different applications such as hand gesture-based control media player control [131], sign language recognition [132], etc.

- *Support vector machine (SVM):* An SVM is a supervised classifier for both linearly separable and nonseparable data. This method non-linearly maps the input data (if not linearly separable in current feature space) to some higher dimensional space, where the data can be linearly separated. This mapping from lower to higher dimensional spaces makes the classification of the input data simpler and more accurate. SVMs are often used for hand gesture recognition [90, 128, 133, 134]. SVMs were originally designed for two-class classification, and an extension for multi-class classification is necessary for gesture recognition. Weston and Watkins [135] proposed

an SVM structure to solve a multi-class pattern recognition problem using a single optimization stage. Dardas *et al.* [128] used this method along with bag-of-features for hand gesture recognition. However, their single optimization procedure found out to be very complicated to be implemented for real-life pattern classification problems [136]. Instead of using a single optimization stage, multiple binary classifiers can be used to solve multi-class classification problems, such as "one-against-all" and "one-against-one" methods. Murugeswari and Veluchamy [137] used "one-against-one" multi-class SVM for gesture recognition. It was found that the "one-against-one" method performs better than the rest of the methods [136].

- *Artificial neural network (ANN):* An ANN is a biologically inspired statistical learning algorithm for functional approximation, pattern recognition and classification. ANNs can be used as a supervised classifier for gesture recognition. Training is performed using a set of labeled input patterns. The ANN classifies new input patterns within the labeled classes. ANNs can be used to recognize static and continuous hand gestures and gestures using a 3D articulated hand model [138]. They have used a dataset collected through Kinect® sensor [139]. A limitation of classical ANN architecture is its inability to handle temporal sequences of features efficiently and effectively [99]. Mainly, it is unable to compensate for changes in temporal shifts and scales, especially in real-time applications [140]. Out of several modified architectures, multi-state time-delay neural networks [141] can handle such changes to some extent using dynamic programming. Fuzzy-based neural networks have also been used to recognize gestures [142].

- ***Dynamic gesture recognition:*** Dynamic gestures or trajectory-based gestures are basically gestures with trajectory having temporal information in terms of video-frames. Dynamic gestures can be either a single isolated trajectory type or continuous type occurring one after another in a stream. Recognition performance of dynamic gestures, especially the continuous gestures, is basically dependent on gesture spotting methods that can be classified as direct or indirect [10]. Direct approaches first detect the time boundaries of the performed gestures and then apply standard isolated gesture recognition. Typically, motion cues (*e.g.*, velocity, acceleration, and trajectory curvature [121])

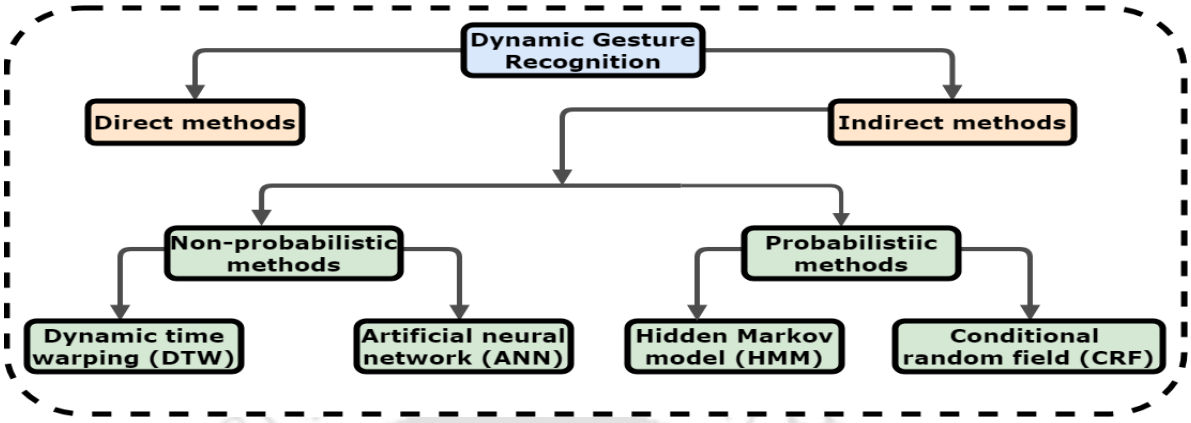


Figure 2.6: Conventional dynamic gesture recognition techniques.

or specific start and end marks [10], an open/closed palm can be employed for boundary detection. On the other hand, in the indirect approach temporal segmentation is intertwined with recognition. Indirect methods for temporal gesture segmentation detect gesture boundaries by finding, in the input sequence, intervals that give good recognition scores when matched with one of the gesture classes. Such methods are highly prone to recognition errors and false positives since they have to deal with two fundamental problems of continuous gesture recognition [24]: 1) spatiotemporal variability, *i.e.*, a user cannot reproduce the same gesture at the exact same shape and duration and 2) segmentation ambiguity, *i.e.*, problems caused by erroneous boundary detection. Through indirect methods, we try to minimize these problems as much as possible. Indirect methods can be of two types (Fig. 2.6): non-probabilistic *i.e.* a) Dynamic programming/ Dynamic time warping, b) ANN; and probabilistic *i.e.* c) HMM and other statistical methods, d) CRF and its variants. Some other common techniques are eigenspace based methods [143], curve fitting [144], finite-state machine (FSM) [62, 98] and graph-based methods [145].

– *Dynamic programming/ Dynamic time warping (DTW):* Dynamic time warping (DTW), a template matching application of dynamic programming, has been widely used in isolated gesture recognition. DTW can find the optimal alignment of two signals in the time domain. Each element in a time series is represented by a feature vector. Hence, the DTW algorithm calculates the distance between each possible

pair of points in two time series in terms of their feature vectors. The steps in a DTW are as follows:

* Two time series P and Q:

$$\mathbf{P} = \mathbf{p_1}, \mathbf{p_2}, ..., \mathbf{p_M}$$
$$\mathbf{Q} = \mathbf{q_1}, \mathbf{q_2}, ..., \mathbf{q_N}$$

where $\mathbf{q}_i$, $\mathbf{p}_i$ are feature vectors for the i^{th} element of the corresponding time sequences.

* Construct $N \times M$ matrix D with distances $D_{ij} = d(\mathbf{p}_i, \mathbf{q}_j)$.
* Warping path W is a contiguous set of matrix elements $w_k = (i, j)_k$
* Define warping between **P** and **Q**

$$W = w_1, w_2, ..., w_K$$

where $\max(M, N) \leq K \leq M + N - 1$

* Find:

$$DTW(P, Q) = \min \sqrt{\sum w_k}$$

DTW has been used for gesture recognition by several authors [10, 37, 63, 146]. Alon *et al.* [10] has proposed a DTW based method that can handle sub-gesture problem. Lichtenauer *et al.* [37] introduced a hybrid approach by using Statistical DTW (SDTW) only for time warping and a separate classifier on the warped features.

– *Hidden Markov Model (HMM):* Though HMM originally emerged in the field of speech recognition, now, it is one of the most widely used techniques for gesture recognition with its numerous variants. HMM is extensively used because it can be applied for modeling the spatiotemporal variability of the gesture videos. Since dynamic gesture is a sequence of images, so there is a need for past knowledge to help the system to recognize gestures and an HMM can help us in this. Before we elaborate on HMM, let us understand a traditional Markov process. A stochastic process has the n^{th} order Markov property if the current event's conditional probability density depends solely on the n most recent events. For $n = 1$, the process is

called a first-order Markov process, where the current event depends solely on the previous event. This is a useful assumption for hand gestures, where the positions and orientations of the hands are treated as events. HMM has two special properties for encoding hand gestures - a) it assumes a first-order model *i.e.* it encodes the present time (t) in terms of the previous time ($t-1$) - the Markov property of underlying unobservable finite-state Markov process and b) a set of random functions, each associated with a state, that produces an observable output at discrete intervals. In this way, an HMM is a "doubly stochastic" process [147]. The states in the hidden stochastic layers are governed by a set of probabilities:

i. The state transition probability distribution $\mathbf{A}$, which gives the probability of transition from the current state to the next possible state.

ii. The observation symbol probability distribution $\mathbf{B}$, which gives the probability of observation for the present state of the model.

iii. The initial state distribution $\mathbf{\Pi}$, which gives the probability of a state being an initial state.

An HMM is expressed as $\lambda = (\mathbf{A}, \mathbf{B}, \mathbf{\Pi})$ and is described as follows:

* Let there be a set of N states $\{s_1, ..., s_N\}$; with a sequence of states $Q = \{q_1, ..., q_T\}$, where $t = 1, ..., T$. For a gesture with M observable states, the set of observed symbol or feature is given by $O = \{o_1, ..., o_T\}$.

* The state-transition matrix is $\mathbf{A} = \{a_{ij}\}$, where a_{ij} is the state-transition probability from state $q_t = s_i$ at time t to state $q_t = s_j$ at time $t+1$.

$$\mathbf{A} = \{a_{ij}\} = P(q_{t+1} = s_j | q_t = s_i), \text{ for } 1 \leq i, j \leq N.$$

* The observation symbol probability matrix $\mathbf{B} = \{b_{jk}\}$, where b_{jk} is the probability of symbol o_k at state s_j.

$$b_j(k) = P[o_k \text{ at t} | q_t = s_j], \text{ for } 1 \leq j \leq N, 1 \leq k \leq M$$

* The initial probability distribution $\Pi = \{\pi_j\}$, where

$$\pi_j = P\left[q_1 = s_j\right], \text{ for } 1 \leq j \leq N$$

The modeling of a gesture sequence involves two phases - feature extraction and HMM training. In the first phase, a particular gesture trajectory is represented by a set of feature vectors. Each of these feature vectors describes the dynamics of a hand corresponding to a particular state of a gesture. The number of such states depends on the nature and complexity of a gesture. In the second phase, the vector set is used as an input to HMM. The global HMM structure is formed by connecting in parallel the trained HMMs $(\lambda_1, \lambda_2, ..., \lambda_G)$, where G is number of gestures to be recognized. For dynamic gestures, temporal components like the start state, the end state, and the set of observation sequences (*e.g.,* position) are mapped by an HMM classifier using a set of boundary conditions.

For a given observation sequence, the key issues of HMM are,

* **Evaluation:** Given the model $\lambda = (\mathbf{A}, \mathbf{B}, \mathbf{\Pi})$. What is the probability of occurrence of a particular observation sequence $O = \{o_1, ..., o_T\} = P(O|\lambda)$? This is the heart of the classification/recognition problem. Determination of the probability that a particular model will generate the observed sequence when there is a trained model for each of a set of classes (forward–backward algorithm).
* **Decoding:** Optimal state sequence to produce an observation sequence $O = \{o_1, ..., o_T\}$ Determination of the optimal state sequence that produces the observation sequence (Viterbi algorithm).
* **Learning:** Determine model λ, given a training set of observations *i.e.* find λ, such that $P(O|\lambda)$ is maximal. Train and adjust the model to maximize the observation sequence probability such that HMM should identify a similar observation sequence in the future (Baum–Welch algorithm).

HMMs are often used for dynamic gesture recognition [50, 97, 148, 149]. But the main disadvantage of HMM is that every gesture model has to be represented and trained separately considering it as a new class, independent of anything else already

learned.

– *Conditional random field (CRF):* CRF is basically a variant of the Markov model with some added advantages. HMM requires strict independence assumptions across multivariate features and conditional independence between observations. This is generally violated in continuous gestures where observations are not only dependent on the state, but also on the past observations. The other disadvantage of using HMM is that the estimation of the observation parameters requires a large amount of training data. The difference between HMM and CRF is that HMM is a generative model that defines a joint probability distribution to solve a conditional problem thus focusing on modeling the observation to compute the conditional probability. Moreover, one HMM is constructed per label or pattern where HMM assumes that all the observations are independent. On the other hand, CRF is a discriminative model that uses a single model of the joint probability of the label sequence to find conditional densities from the given observation sequence. CRFs seamlessly represent contextual dependencies and have computationally attractive properties. CRFs support efficient recognition using dynamic programming, and their parameters can be learned using convex optimization.

  Both HMM and CRF can be used for labeling sequential data. For this, we define a statement for a given observation sequence x that, we want to choose a label sequence y^* such that the conditional probability $P(y|x)$ is maximized, that is:

$$y^* = argmax_y P(y|x) \tag{2.11}$$

  Maximum entropy Markov models (MEMMs) are discriminative models, where each state has an exponential model that takes the observation sequence as input and outputs a probability distribution over the next possible states.

$$P(y|x) = \prod_{t=1}^{T} P(y_t|y_{t-1}, x) \tag{2.12}$$

  Each of the $P(y_t|y_{t-1}, x)$, is an exponential model of the form:

$$P(y|x) = \frac{1}{Z(x_t, y_{t-1})} \exp(\sum_a \lambda_a f_a(x_t, y_t)) \tag{2.13}$$

where Z is a normalization constant and the summation is overall features. But MEMM suffers from *Label Bias Problem*, i.e., the transition probabilities of leaving a given state are normalized for only that state (local normalization). MEMMs have a non-linear decision surface because the current observation is only able to select what successor state is selected, but not the probability mass transferred to that state. In order to avoid this effect, a CRF employs an undirected graphical model that defines a single log-linear distribution over the joint vector of an entire class label sequence given a particular observation sequence(thus the model has a linear decision surface). Let $G = (V, E)$ be a graph such that $Y = (Y_v)$, $v \epsilon V$ so that Y is indexed by vertices of G. Then (X, Y) is a conditional random field in case, when conditioned on X, the random variables Y_v obey the Markov property with respect to the graph: $p(Y_v|X, Y_w, w \neq v) = p(Y_v|X, Y_w, w \sim v)$, where $w \sim v$ means that w and v are neighbors in G. Given by Hammersley and Clifford, it states that the probability distribution of x satisfies the Markov property with respect to graph $G(V, E)$ if and only if, it can be factored according to G:

$$P(x) = \frac{1}{Z} \prod\nolimits_C \psi_C \tag{2.14}$$

where Z is the normalization constant and ψ_C is the potential function over clique C.

$$P(x) = \frac{1}{Z} \prod_C \exp(\lambda_C^T f(C)) = \frac{1}{Z} \exp(\sum_C \lambda_C^T f(C)) \tag{2.15}$$

where $f(.)$ is the feature vector defined over the clique and λ is the corresponding weight vector for those features. Bhuyan *et al.* [98] proposed a classification technique based on CRFs on a novel set of motion chain code features. Sminchisescu *et al.* [5] have compared performance analysis applying algorithms based on CRF and MEMMs for recognizing human motion in monocular video sequences. Undirected conditional model CRF and directed conditional model MEMMs with different

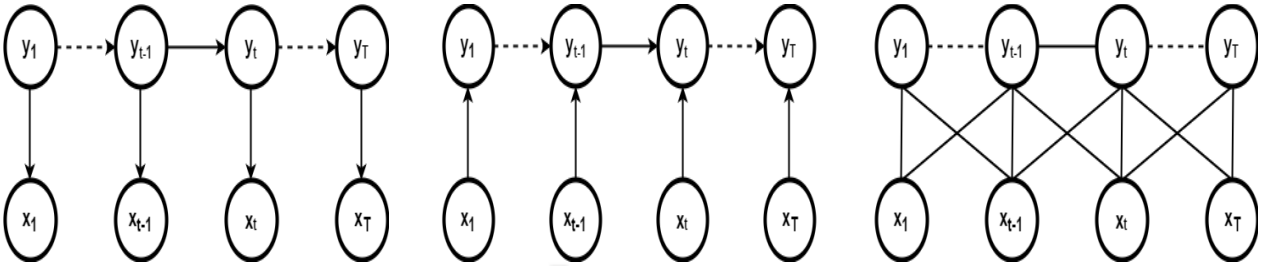


Figure 2.7: (a) HMM (b) A directed conditional model or MEMM (c) A Conditional Random Field accommodates arbitrary overlapping features or long-term dependency of observation sequence [5].

windows of observations are compared with HMM. Both MEMM and HMM have difficulty in accounting for long-range observation dependencies that appear useful in discriminating among different gestures. We observed that CRFs have improved recognition performance over MEMMs, which in turn, typically outperformed competing HMMs. This is because CRF uses an undirected graphical model to overcome the problem of label bias present in maximum entropy Markov models (MEMMs) where states with low-entropy transition distributions effectively ignore their observations. The main disadvantage of CRF is that training is more time consuming ranging from several minutes to several hours for models having longer windows of observations (as opposed to seconds for HMMs, or minutes for MEMMs), on a standard desktop PC.

– *Some other classification methods:* Here we discuss some other classification techniques that have also been used in the classification of gestures. Patwardhan and Roy [143] proposed an eigenspace based framework to model dynamic hand gestures containing both shape and trajectory information which is rotation, scaling and translation (RST) invariant. Shin *et al.* [144] proposed a curve-fitting based geometric method using Bezier curves for the trajectory analysis and classification of dynamic gestures. Gestures are recognized by fitting the curve to the 3D motion trajectory of the hand. The gesture speed is incorporated into the algorithm to enable accurate recognition from trajectories having variations in speed. Bhuyan *et al.* [62,98] represented the keyframes of a gesture trajectory as an ordered sequence of

states in the spatial-temporal space, which constitutes a finite state machine (FSM). The recognition of gestures can be performed using the trained FSM. Graph-based techniques are also used as a powerful tool for pattern representation and classification but have been practically left unused for a long period of time due to its high computational cost. [145] used graphs for gestures matching in an eigenspace to handle hand occlusion.

2.1.3.2 Depth-based methods on RGB-D data

Depth information is largely invariant to illumination variations and skin colors and offers a quite clear segmentation from the background. So, the major problems in segmentation like illumination variations and occlusion can be handled nicely with the help of depth data to a great extent. Due to these advantages, depth cameras have been used in computer vision for several years. However, the applicability of depth cameras was limited due to its high price and poor quality. The release of low-cost color-depth (RGB-D) cameras like Kinect® by Microsoft, Leap Motion Controller (LMC) by Leap Motion, Intel RealSense®, Senz3D® by Creative and DVS128® by iniLabs, has created a revolution in gesture recognition by providing high-quality depth images, addressing issues like complex backgrounds and illumination variations. Out of all these, hand gesture recognition on Kinect®-based dataset and 'one-shot learning' with RGB-D data, are the prominent methods discussed mostly in depth-based hand gesture recognition.

- *Kinect® based methods:* Kinect® has a combination of RGB and IR camera along with depth sensor [33]. It uses the infrared projector and sensor for depth computation; not the RGB camera. The infrared projector projects a known pattern on the object and a CMOS sensor captures the deformations in the reflected pattern. Depth information is then calculated by mapping a three-dimensional view of the scene obtained from the deformation information. Kinect® obtain RGB-D data by combining structured light with two classic computer vision techniques: depth from focus and depth from the stereo. The skeletal data obtained from these RGB-D sensors are converted to more meaningful and high-level features, and algorithms are developed for the robust classification of gestures. Recognition of hand gestures is especially challenging due to the complex articulation and relatively smaller area of hand region. Kinect® is useful in addressing these fundamental

problems in computer vision [139, 150, 151]. It has also diverse applications ranging from gaming to classroom [35, 48].

- *Other depth sensor-based methods:* Leap motion controller (LMC) and Intel RealSense® are the most used RGB-D sensor for HCI applications apart from Kinect®. RealSense® is more robust to self-occlusions and it can capture pinching gestures. LMC is another RGB-D sensor and its objective is to determine 3D fingertip positions instead of whole-body depth information as with the Kinect® sensor. The sensor can detect only fingertips lying parallel to the sensor plane, but with high accuracy. In [61] feature vector with depth information is computed using a leap motion sensor and fed into the hidden conditional neural field (HCNF) classifier to recognize dynamic hand gestures. Leap motion sensors can be used in various applications, *e.g.*, virtual environments [152] and sign language recognition [153].

- *One-shot learning methods on RGB-D data:* Using Deep Learning, human-level performance has become achieve-able on complex image classification tasks. However, these models rely on supervised training paradigm and their performance heavily depends on the availability of labeled training data. Also, the classes that the models can recognize are limited to those they were trained on. This makes these models less useful in realistic scenarios where enough labeled data is not available for all classes during training. Also, since it is practically not possible to train on images of all possible objects, so the model is expected to recognize images from classes with a small amount of data in the training phase or precisely with a single example. So, in the case of a small dataset, 'one-shot learning' may be very useful. Various researchers [154–156] have used one-shot learning in both deep learning and non-deep learning paradigm for recognition of hand gestures, especially with RGB-D data. Wu *et al.* [154] proposed a system to learn gestures from only one learning example per class, namely 'one-shot learning'. Features are extracted based on extended-motion-history-image (Extended-MHI) and the gestures are classified by calculating the maximum correlation coefficient. The extended-MHI is proposed to improve the performance of MHI by compensating for the non-moving regions and repetitive actions. Multi-view spectral embedding (MSE) algorithm is used to fuse the RGB and depth data in a physically meaningful manner. The MSE algorithm discovers the intrinsic

relationship between RGB and depth features, improving the recognition rate of the algorithm. In [157], authors applied an approach combining MHI with statistical measures and frequency domain transformation on depth images for one-shot-learning hand gesture recognition. Due to the availability of the depth information, the background-subtracted silhouette images were obtained using a simple mask threshold.

2.1.3.3 Deep learning techniques

Though the idea of artificial intelligence (AI) is quite ancient, modern AI first came into the picture around the mid-20^{th} century. The AI aims at developing intelligence in machines so as to make them work and respond like humans. This can be achieved when the machines are made to have certain traits, e.g., reasoning, problem solving, perception, learning, etc. Machine learning (ML) is one of the cores of AI. There are a large number of applications of ML in many aspects of modern human society. Consumer products like cameras and smartphones are the best examples where ML techniques are being employed increasingly. In the area of computer vision, ML techniques have been widely applied in tasks such as object detection, image classification, face recognition, gesture and activity recognition, semantic segmentation, and many more. In conventional ML, engineers and data scientists have to identify useful features and they have to handcraft the feature extractor manually which requires considerable engineering skills and domain knowledge. In order to identify important and powerful features, they must have considerable domain expertise. The issue of "handcrafting features" can be addressed if good features can be learned automatically. This automatic learning of features can be done by a learning method called "representation learning". It is a set of methods that enables a machine to automatically learn the representations that are crucial for detection or classification.

Recently, deep learning has irrupted in action and gesture recognition fields achieving outstanding results and outperforming "non-deep" state-of-the-art methods. Deep learning, a subfield of ML, is based on representation learning methods having multiple levels of representation. Deep learning is a set of algorithms in ML, in which learning of multiple levels of representation is carried out to model complex relationships among data. In several fields, such as computer vision, deep learning methods have been proved to have much better performance than con-

ventional ML methods. The main reason for deep learning having an upper hand over ML is the fact that the feature learning mechanism at these different levels of representation is fully automatic, thereby allowing the computational model to implicitly capture intricate structures embedded in the data. In deep learning, higher-level features are defined in terms of lower-level features. The deep learning methods are said to have deep architecture because of the non-uniform processing of information at different levels of abstraction. This has motivated the development of learning robust and effective representations directly from raw data and deep learning provides a plausible way of automatically learning multiple level features, by using multiple processing layers to learn image representations with multiple levels of feature abstraction. Deep networks are capable of finding salient latent structures within unlabelled and unstructured raw data and can be used for both feature extraction as well as classification [158]. The recent popular deep learning methods like *convolutional neural network* (CNN), *recurrent neural network* (RNN) and *long short-term memory* (LSTM) have demonstrated competitive performance in both image/video representation as well as classification. But deep learning approaches have mainly two inherent requirements: huge data for training purposes and expensive computation. But in this modern era, the abundance of high quality, easily available labeled datasets from different sources along with parallel graphics processing unit (GPU) computing, also played a key role in the success of deep learning by fulfilling its requirements. We will see all these methods one by one, but before that let's talk about one major problem of deep learning which is the requirement of huge data and how various researchers have tried to overcome it through the data augmentation process when the database is limited.

- ***The need for data augmentation in deep learning methods:*** Indifference to handcrafted features, there is a growing trend toward feature representations learned by deep neural networks [8, 158–177]. But, in deep learning techniques, the main requirement is lots of database samples. Several authors have emphasized the importance of using many diverse training examples for CNNs/RNNs [158]. For datasets with limited diversity, they have proposed data augmentation strategies to prevent CNNs/RNNs from overfitting in the training phase. Krizhevsky *et al.* [158] employed multiple data augmentation strategies in the training images for classification task into 1000 categories. Simonyan and Zisserman [159] employed similar spatial augmentation on each video frame to train

CNNs for video-based human activity recognition. However, these data augmentation methods were limited to spatial variations only. To add variations to video sequences containing dynamic motion, Pigou *et al.* [160] temporally translated video frames in addition to applying spatial transformations. In order to avoid overfitting, Molchanov *et al.* [161] applied a 3D-CNN on the whole video sequence and introduce space-time video augmentation techniques.

- ***Convolutional neural networks (CNN):*** In 1962, D.H. Hubel and T.N. Weisel proposed the model of Cat's visual cortex, which later on helped in the development of CNNs. The first neural network model for visual pattern recognition was proposed by K. Fukushima in 1980 and was given a nickname "neocognitron" [178]. This network was based on unsupervised learning. Finally, in the late 90s, Yann LeCunn and his collaborators developed CNN which showed exciting results in various recognition tasks [8]. But till 2012, CNN was not that much evolved due to the requirements of deep learning methods mentioned above. After the work of Krizhevsky *et al.* [158], various researchers applied CNN in various domains for classification as well as other purposes. Generally, 2D-CNN is used in the case of images that can access only spatial information, whereas, for video processing, 3D-CNN (C3D) is quite effective which can extract both spatial as well as temporal information. A combined fusion-based method with CNN as trajectory shape extractor as a gesture feature and CRF as temporal feature recognition is proposed by [179]. In [72], the authors used CNN for recognition of hand gesture using trajectory-to-contour-based images obtained through skin segmentation and tracking method. In [180], authors used pseudo-color based MHI images as input to convolutional networks. [**?**] proposed a framework for the recognition of isolated gestures where the trajectory of the moving hands with different shapes, sizes and colors are detected through optical flow, and the proper hand gesture is recognized using a VGG16 CNN architecture.

- ***3D-CNN (C3D) model:*** 2D-CNNs are such a type of deep network that can act directly on the raw images *i.e.* limited to handling 2D images. Whereas 3D-CNN models, also called as C3D, act on videos for action/gesture recognition. This model extracts features from both spatial and temporal dimensions by performing 3D convolutions, thereby capturing the spatial as well as motion information encoded in multiple adjacent frames

of the video. [167] introduced a C3D network for human action recognition. To analyze a series of short video clips and average the network's responses for all clips, Tran *et al.* [168] employed a 3D-CNN to learn the spatio-temporal features from sliced video clips and then fuse these features to make the final classification. [169] used a temporal segment network that works on video-segments called snippets for spatio-temporal evaluation in action recognition. 3D-CNN (C3D) is quite effective which can extract both spatial as well as temporal information in less expense of both data and processing computation compared to RNN/LSTM [170, 181].

- ***Two-stream model:*** Ciregan *et al.* [162] has shown that the use of multi-column deep CNNs with multiple parallel networks can improve recognition rates of single networks by 30-80% for various image classification tasks. Similarly, for large scale video classification, Karpathy *et al.* [163] have shown the best results on combining CNNs trained with two separate streams of the original and spatially cropped video frames. Simonyan and Zisserman [159] proposed separate CNNs for the spatial and temporal streams that are late-fused and that explicitly use optical flow in the context of action recognition. To recognize sign language gestures, Neverova *et al.* [164] employed CNNs to combine color and depth data from hand regions and upper-body skeletons. Two stream model with two C3D layers that takes RGB and optical flow computed from the RGB stream as inputs were used by [170] for action recognition. [171] used a hidden two-stream CNN model which takes only raw video frames as input and directly predicts action classes without explicitly computing optical flow. Here the network predicts the motion information from consecutive frames through a temporal stream CNN that makes the network 10× faster [171], without computing optical flow which is time-consuming.

- ***Long-term video prediction–RNN/LSTM/GRU:*** CNN can exploit limited local temporal information, and hence, the researchers have moved towards RNN, which can process temporal data using recurrent connections in hidden layers [172]. However, the main drawback of RNN is its short-term memory, which is insufficient for real-life temporal variations in gestures. To solve this problem, long short-term memory (LSTM) [174] was proposed which can tackle longer-range temporal variations. Gestures or actions, in a video sequence, can be considered as a sequential temporal evaluation of body/body-

part in a space-time representation. So, 3D-CNN/RNN/LSTM is the network generally applied in action/gesture recognition. In addition to 3D-CNNs, recurrent neural networks have also been applied for dynamic hand gesture classification [165, 173]. [166] has extracted hand trajectory and hand posture features from RGB-D data and then a two-stream recurrent neural network (2S-RNN) is used to fuse multi-modal features. The spatio-temporal graphs are well known for modelling of a spatio-temporal structure. Hence, a combination of high-level spatio-temporal graphs and RNN can also be used to solve the spatio-temporal modelling problem of RNN [182]. The long short-term memory problem and vanishing/exploding problem of RNN can be handled to some extent by adding 'gates' in LSTM. Hence LSTM-based deep networks can be used for efficient modelling of dynamic gestures [175–177]. However, in RNN/LSTM, the problem of vanishing/expanding gradient is much acute compared to CNN and they become more data-hungry. Gated recurrent units (GRU) are simplified LSTM units with adaptive gate parameters with fewer parameters which makes the training process faster. [183] proposed a skeleton-based dynamic hand gesture recognition method that divides geometric features into multiple parts and uses a gated recurrent unit-recurrent neural network (GRU-RNN) for each feature part. Because each divided feature part has fewer dimensions than an entire feature, the number of hidden units required for optimization is reduced. As a result, the scheme achieved competitive recognition performance with fewer parameters.

So, in a nutshell, deep learning techniques can give outstanding performance in both feature extraction and recognition owing to their built-in feature learning capability. The effective and efficient algorithms of deep networks are capable of solving complex optimization tasks.

2.2 Summary and Scope for Present Work

In this chapter, a brief review is presented addressing different approaches for vision-based hand gesture recognition. The literature shows that the accuracy of gesture recognition depends on the different stages of the system. And, recognition rate falls significantly in the presence of background noise, variations in illumination and shadows, presence of skin-like colors in the background, occlusion, complex background and varied shape and size of the hand. A particular

method may not be able to compensate for all the different variabilities present in the gesture recognition system.

Handling the problem of illumination variations requires methods like illumination suppression techniques. So, we would like to propose a skin segmentation method with an illumination compensation technique in different color-spaces. Again, occlusion handling itself is a vast research topic in this field. We would like to address such situations of occlusion or blurring incorporating some additional measures. The performance of a gesture recognition system depends on the accuracy of a particular module used in each step. We would also try to upgrade it if the used scheme/step has any shortcomings in achieving our goal.

Another target is to propose such a model which is invariant to the shape and size of the hand in recognition of a gesture. In achieving these motives, our primary target will be to represent a gesture by extracting the motion information confined in the trajectory of a dynamic gesture. Moreover, it is seen that the use of different streams with informative input data helps to increase the performance in the recognition accuracy. So, in this part, we will try to fuse different input information to propose a fusion framework.

Lastly, we would also like to explore some deep learning method that is applicable to both static and dynamic hand gestures. Popular deep learning methods like convolutional neural networks (CNN) have demonstrated competitive performance in both image representation as well as classification. Recently, attention mechanisms are widely used in computer vision to extract better visual features. So, our main goal is to combine these two for hand gesture recognition.

3

Hand Detection by Two-Level Segmentation with Double-Tracking and Gesture Recognition using Deep-Features

Vision-based hand gesture recognition involves a visual analysis of handshape, position and/or movement. Most of the previous approaches require complex gesture representation as well as the selection of robust features for proper gesture recognition. To eliminate this problem, a simple model-based framework has been presented here using a deep network for hand gesture recognition. The model is fed with 'hand-trajectory-based-contour-images'. These images represent the motion trajectory of the hand for isolated trajectory gestures obtained via pre-processing steps - a two-level segmentation process and a double-tracking system. Deep features extracted from these images are used for estimating the hand gestures. Conventional machine learning methods involve tedious feature engineering schemes, while deep learning approaches can learn image features hierarchically from local to global with multiple layers of abstraction from a vast number of raw sample images. The feature learning capability of CNN architecture has been used here and it has shown outstanding results on three different datasets.

3.1 Introduction

One contemporary goal in human-computer interface design is to enable effective and engaging interaction. For example, vision-based hand gesture recognition (VGR) systems can enable contactless interaction in sterile environments such as hospital surgery rooms, or simply provide engaging controls for entertainment and gaming applications. However, VGR is not as robust as a standard keyboard and mouse-based interaction. Issues such as sensitivity to size and speed variations of the gesturing body part, varying luminance conditions in the scene, a poor performance against complex backgrounds and the reliable detection of the gesturing phase due to various reasons like blurring or occlusion, etc, have limited the use of hand gestures as a reliable tool in interface design [51].

Numerous attempts have been taken by various researchers to deal with these inherent problems in a vision-based hand gesture recognition system. In this process, it is seen that a classical vision-based hand gesture recognition system usually consists of three main stages: first, acquisition, detection and pre-processing; second, gesture representation and feature extraction; and finally recognition [51]. Among them, proper segmentation of the hand and robust and effective feature representation are two major challenges. Accurate segmentation of the hand from the captured images/videos remains a challenge for many preoccupied constraints like illumination variations, background complexity, and occlusion due to the articulated shape of the hand. Here in this work, the major issues that have been tried to address are -

Firstly, to get proper segmentation, the effect of illumination variations has been tried to minimize as much as possible. Various researchers have shown significant attention towards skin color information owing to its computational efficiency yet, robustness against rotations, scaling and partial occlusions. Generally, the chrominance cue information of skin color is less sensitive to illumination changes as compared to the luminance counterpart. Human skin color does not scatter randomly in a given color space, but clusters in a minimal region of a given color space. However, the color representation is not segregated into luminance and chrominance components in RGB color space, and all the three color components R, G, B are highly correlated to each other. Due to this reason, HSV and YCbCr color spaces are used in the skin segmentation process where the luminance component has been dropped, and only the chrominance component is used. Along with skin-color segmentation, motion information has

also been used to segment the moving hand through the three-frame differencing method. These two methods are applied separately on the video frames and finally, the intersected portion is retained after a double check on the region of interest and the next process is carried out on this segmented hand region.

Secondly, tracking is done on the segmented region through a particle filter tracker which has been modified to handle the problem of blurring or occlusion to a great extent. Moreover, another CAMShift tracker tracks the centroid of each tracked hand region in each frame. In this way, a double-tracking system gives a single image called *hand-trajectory-based-contour-images* that describes the trajectory of the gesture video. These images are used in feature extraction and recognition.

Lastly, the manual handcrafted features usually demand the user to have some prior knowledge and some preprocessing such as image transformation, etc, for proper recognition by a classifier. This has motivated the development of learning robust and effective representations directly from raw data and deep learning provides a plausible way of automatically learning multiple level features, by using multiple processing layers to learn image representations with multiple levels of feature abstraction. The recent popular deep learning methods like *convolutional neural network* (CNN) have demonstrated competitive performance in both image representation as well as classification. This work utilizes CNN to extract robust hand gesture features that can be used to recognize the hand gestures more precisely.

The main contribution of this work is to present a hand gesture recognition model for isolated trajectory-based gestures using CNN architecture which is more suitable in spatial and semantic representation. Especially, in order to enhance the hand gesture representation, a technique for converting a gesture video into a single image representing the trajectory of the gesture is introduced in the pre-processing step and we call these images hand-trajectory-based-contour-images. Here the pre-processing scheme removes the constraint of variable illuminations inherently present in the original gesture videos. Through a double-tracking framework, the problem of blurring or occlusion has also been tried to minimize. The experimental results demonstrate the effectiveness of the model achieved through preprocessing steps that remove the illumination variations and occlusion problems cooperating in raw data. And, the salient features learned through CNN have shown improved results for hand gesture recognition.

3.2 Background and Related Work

3.2.1 Pre-processing - color-based skin segmentation, motion-based segmentation and tracking

The unique color characteristic of human skin makes color-based skin segmentation very useful. However, the compactness of the skin colored clusters is not the same for all the color spaces. Multiple color spaces are investigated in the literature for skin detection. The choice of color spaces affects the shape of the skin cluster. This eventually affects the detection accuracy. Most color-based skin segmentation depends on some explicit boundary specification. Boundary specification for skin color depends on a set of thresholds and conditions which could be either defined in the same color space, (*e.g.* RGB) or in a transformed color space, such as YCbCr, HSV etc.

RGB is the most primitive color space in computer vision. However, in RGB the color representation is not segregated into luminance and chrominance components, and all the three color components R, G, B are highly correlated to each other (shown in Fig. 1.7). For example, changing the luminance component (average of the three colors) also changes the RGB values of pixels. Therefore, a varying illumination can affect all the RGB elements of a pixel belonging to a skin patch. This eventually results in a change of location of the skin cluster in the RGB space.

In skin detection, another type of perceptual color space is HSV. In this color space, color is represented using three components Hue (H), Saturation (S), and brightness (V). The brightness components V is independent of chromatic components H and S. Hence, it can be dropped to reduce the effect of illumination change on skin color during skin detection. One of the major advantages of using this color space in skin detection is that a skin color boundary with respect to H and S can be specified intuitively by a user [184]. One of the earliest methods of skin detection is proposed by Sobottka and Pitas [70]. They proposed a skin detection boundary along S and H channels in HSV color space as $S \in [0.23, 0.68]$ and $H \in [0, 50]$. Later, Tsekeridou and Pitas proposed a modification [185] to this method for face region segmentation in an image watermarking system [186]. The corresponding boundary rule in the HSV color space is as follows:

Table 3.1: Most popular examples of color spaces used in skin detection: RGB, YCbCr, HSV [12].

color space	Range of components	Restrictions for skin color
RGB	R, G, B: [0, 255]	$R > 95 \wedge G > 40 \wedge B > 20 \wedge \{\max(R,G,B) - \min(R,G,B) > 15\} \wedge \lvert R - G \rvert > 15 \wedge R > G \wedge R > B$
YCbCr	Y, Cb, Cr: [0, 255]	$Y > 80 \wedge 77 < Cb < 127 \wedge 133 < Cr < 173$
HSV	H: [0°, 360°], S, V: [0, 1]	$0° < H < 50° \wedge 0.1 < S < 0.68 \wedge 0.35 < V < 1$

$$\begin{cases} (0 \leq H \leqslant 25) \vee (335 \leqslant H \leqslant 360) \\ (0 \leqslant S \leqslant 0.6) \wedge (0.4 \leqslant V) \end{cases} \tag{3.1}$$

According to Hsu *et al.* [68], the skin color cluster is more compact in YCbCr than in any other color space. That means YCbCr has the smallest overlap between the skin and non-skin data under various illumination conditions. Hsu et al. [68] proposed a boundary rule based on YCbCr color space. The authors observed that the shape of the skin tone cluster in *Cb*-*Cr* space can be approximated as an elliptical structure where the cluster location depends on luminance Y. They performed a non-linear modification to C_b and C_r values if $Y < 125$ or $Y > 188$. Subsequently, the skin pixel cluster is modelled as an ellipse in a transformed space $Cb'Cr'$. Kukharev and Nowosielski proposed another set of skin detection rules [187] using RGB and YCbCr color spaces as follows:

$$\begin{cases} (R > G) \wedge (R > B) \\ \{(G \geqslant B) \wedge (5R - 12G + 7B \geqslant 0)\} \vee \{(G < B) \wedge (5R + 7G - 12B \geqslant 0)\} \\ \{Cr \in (135, 180)\} \wedge \{Cb \in (85, 135)\} \wedge (Y > 80) \end{cases} \tag{3.2}$$

In [188], Shaik *et al.* compared HSV and YCbCr spaces for skin detection using a boundary-based method. Table 3.1 represents the range for three popular color spaces used in skin detection.

Skin segmentation becomes very challenging when the color information is not available or

when there are multiple skin-colored scenes in the background. Basically, when prior knowledge of the color of the moving object is not available, pixel-level change can provide powerful motion-based cues for detecting and localizing objects. Various approaches for moving object detection using pixel-level change can be background subtraction, inter-frame difference or three-frame difference. The background subtraction method is extremely sensitive to the changes in illumination. Moreover, stabilized background detection is always a costly affair making it vulnerable for long and varied video sequences [82]. Apart from this, the choice of temporal distance between frames is a tricky question. It depends on the size and speed of the moving object. Conversely, inter-frame difference methods can easily detect motion but show poor performance in localizing the object. The three-frame difference approach uses previous, current and future frames to localize the object in the current frame. Using future frames introduces a lag in the tracking system, but this lag is acceptable if the object is far away from the camera or moves slowly relative to the high capture rate of the camera. But three-frame difference method has some other advantages for which we adopted this method and it will be discussed in the implementation section.

Tracking of the hand in a gesture video can be very difficult as the movement of the hand can be very fast and its appearance can change vastly within a few frames. In such cases, model-based algorithms like mean-shift [84], Kalman filter [85], particle filter [86] are some of the methods used for tracking. The mean-shift is a purely non-parametric mode-seeking algorithm that iteratively shifts a data point to the average of data points in its neighbourhood (similar to clustering). However, tracking often converges to an incorrect object when the object changes its position very quickly in the two neighbouring frames. Because of this problem, a conventional mean-shift tracker fails to position a fast-moving object. In [87], a modified version of mean-shift algorithm has been used to continuously adapt mean-shift (CAMShift) where the window size is adjusted so as to fit the gesture area reflected by any variation in the distance between the camera and the hand. Though CAMShift performs well with objects that have a simple and constant appearance, it is not robust in more complex scenes. The motion model for the Kalman filter is based on the assumption that the velocity is relatively small when objects are moving, and therefore, it is modeled by a zero mean and low variance white noise. One limitation of the Kalman filter is the assumption that the state variables are based on Gaussian distribution, and thus the Kalman filter will give incorrect estimations of state variables that do

not follow a linear Gaussian environment. The particle filter is generally a better method than the Kalman filter because it can consider non-linearity and non-Gaussianity. The main idea of the particle filter is to apply a weighted sample particle set to approximate the probability distribution, i.e., the required posterior density function is represented by a set of random samples with associated weights and estimation is done on the basis of these samples and weights. Both Kalman filter and particle filter have the disadvantage of the requirement of previous knowledge in modeling the system. Kalman filter or particle filter can be combined with the mean shift tracker for precise tracking.

Comaniciu et at. [189] proposed a model based on the color histogram for tracking hands. The color histogram of the hand detected from the video was used initially as the information for shifting the hand region and track it in the consecutive frames of the video sequences. The limitation of the system was that it can track the object if the background color has a different color other than the tracked object. To remove this difficulty, a method was proposed by Guo et at. which combined two additional information i.e. AdaBoosting and background removal along with the color information [190]. The limitation of this approach was that the background model has to be known beforehand and the tracking object should not be included in the background model. In [191], the CAMShift algorithm was proposed for tracking objects, but the constraint faced by this algorithm was that it was not able to handle occlusion. Later Shi and Tomasi [192] used minimum Eigen feature points to track the target object. The advantage of this model was that it does not depend on the color information. But this tracker faced difficulty in tracking objects with the long video sequences. This was because the feature points went on decreasing with consecutive frames of the video may be due to the change in illumination or shape of the hand or occlusion. KLT tracker was used by Kolsch and Turk [193] to track the hand. The major problem was that it was not able to track if the target object suffers from any shape transformations. Asaari et at. [194] incorporated eigenhand with the adaptive Kalman filter for tracking hand. They proved that their algorithm could be suitable for challenging environments, but it fails when there are large-scale variations and changes in poses of the gesturing body part. In [88], authors have detected hand movement using Adaboost with the histogram of oriented gradient (HOG) method. But this tracking algorithm is very sensitive to illumination variations.

Here in this work, we have segmented the moving hand through skin segmentation and

motion-based segmentation through the frame-differencing scheme. The luminance counterpart in skin segmentation is also suppressed for segmentation under varying illumination conditions. To minimize some of the prevailing tracking problems like changes in the hand shape or blurring or occlusion, a double-tracking framework consisting of CAMShift and particle filter has been adopted. For the measurement model of particle filter, a histogram of oriented gradients (HOG) has been applied. The details of the implementation are given in the methodology section.

3.2.2 Feature extraction and classification

Vision-based hand gesture recognition for natural human-computer interaction is still an active research field [10, 56, 97, 98, 124, 195]. A lot of early works prevailed in state-of-the-art hand gesture recognition that requires prior knowledge for designing hand-crafted features [56,124,195]. Different spatio-temporal descriptors such as Fourier descriptor (FD) [124], scale-invariant feature transform (SIFT) [195], histogram of oriented gradients (HOG) [56] are used to recognize gestures. Different template matching approaches like dynamic time warping (DTW) [10] and various state-space models like hidden Markov models (HMM) [97] and conditional random fields (CRF) [98] have been widely used as gesture classifiers. SVM is another famous gesture recognition classifier [56, 195]. A problem with many of these approaches is that a large variety of gestures executed by different people is very difficult to match. Hence, robust classification of gestures under widely varying lighting conditions with complex backgrounds, and from different subjects is still a challenging problem.

Indifference to hand-crafted features, there is a growing trend towards feature representations learned by deep neural networks. The main reason for deep learning having an upper hand over ML is the fact that the issue of "handcrafted features" can be addressed by deep learning. The feature learning mechanism of deep learning at different levels of representation is fully automatic, thereby allowing the computational model to implicitly capture intricate structures embedded in the data. In deep learning, higher-level features are defined in terms of lower-level features. 2D-CNNs are such a type of deep network that can act directly on raw images *i.e.* it can handle 2D images. Whereas 3D-CNN models, also called C3D, act on videos for action/gesture recognition. Recently, deep convolutional neural networks have been successful for both feature extraction and classification in various recognition challenges [8,158–160,162,163].

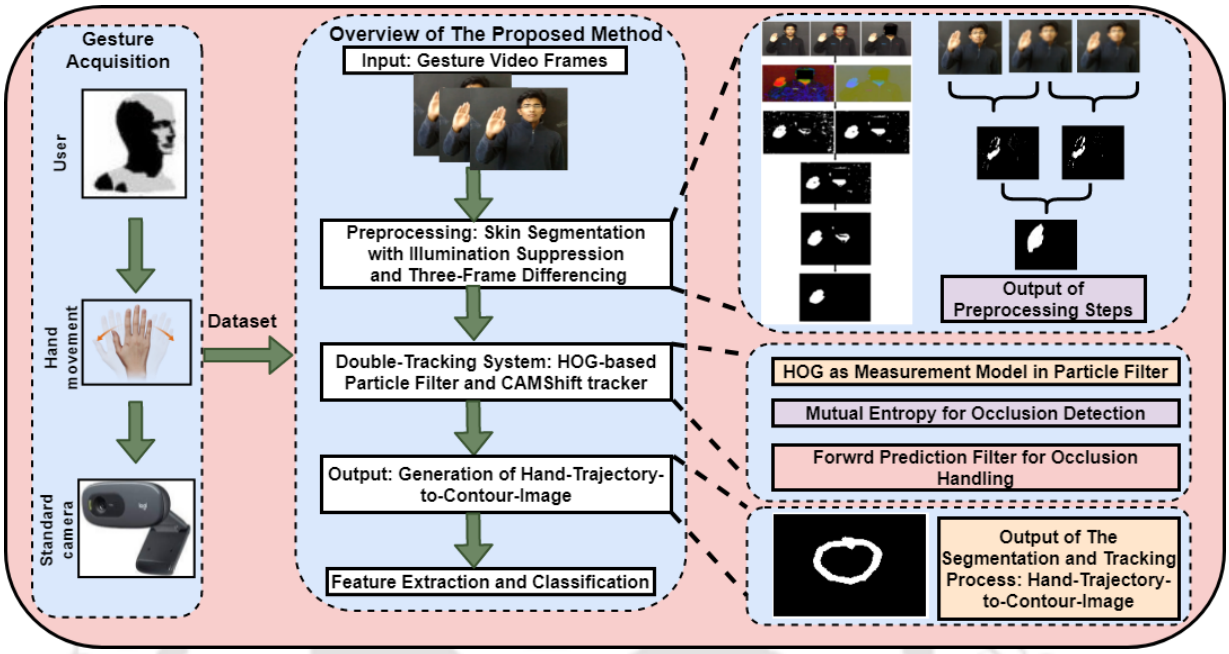


Figure 3.1: Block diagram of our proposed hand gesture recognition framework.

In this work, we are using a 2D-CNN model instead of 3D-CNN with a focus on gesture video recognition. This is because the whole gesture video is converted into a single image called hand-trajectory-based-contour-image through pre-processing steps. The details are given in the next section.

3.3 Proposed Hand Gesture Recognition Methodology

This work presents a model for the recognition of isolated hand gestures under constrained environments like varying illumination, occlusion, or blurring conditions (shown in Fig. 3.1). This has been one of the substantially challenging tasks in gesture recognition. The main aim of this work is to build a model that can correctly classify an instance of the test gesture that actually belongs to one of the pre-defined classes. For this model, the first step is to segment the hand region effectively followed by tracking the moving hand. From these steps, some images are generated which basically represent the trajectory of the hand for the whole gesture. Thus the preprocessing part yields some 2D images called hand-trajectory-based-contour-images from gesture videos. In the next stage, CNN is chosen as feature extractor due to its inherent ability to learn features directly from raw images. It generally gives improved recognition performance

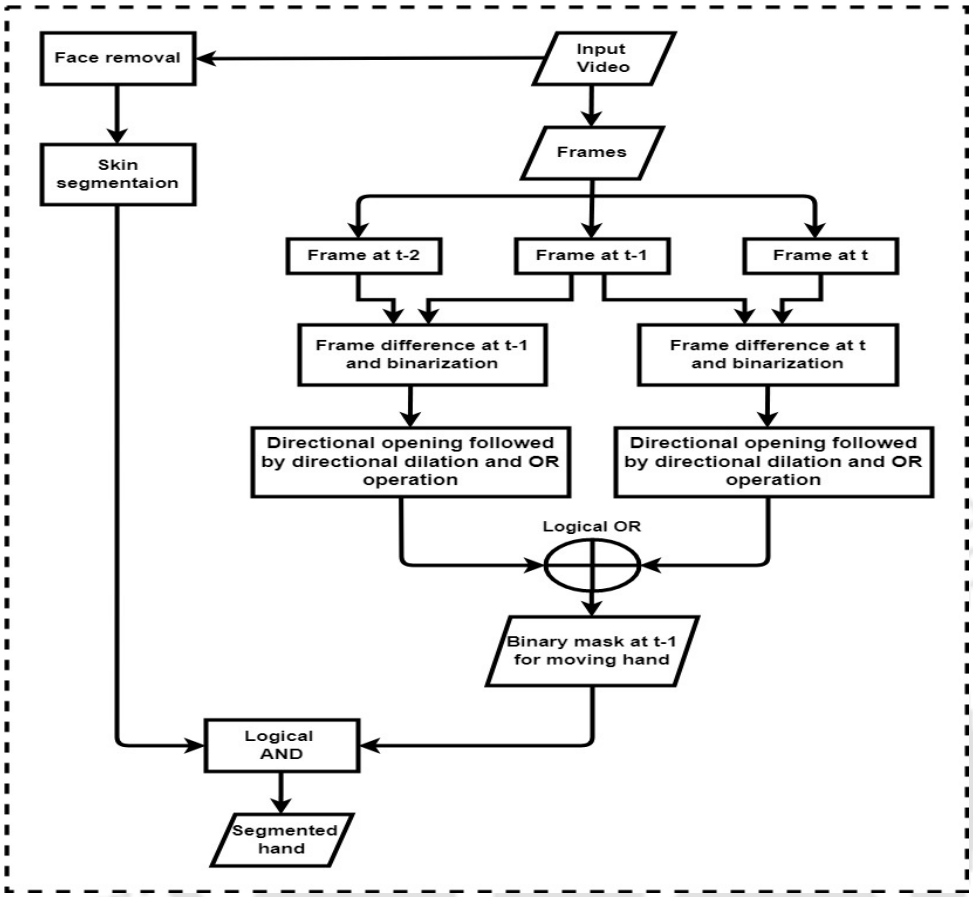


Figure 3.2: Flowchart for hand segmentation using skin segmentation and three-frame differencing.

if there is no over-fitting. In case of over-fitting various approaches like regularization, increase in training samples through data augmentation, etc, are adopted. But we have adopted another method which will be explained in the experiment section pertaining to a specific dataset.

3.3.1 Pre-processing

In the pre-processing stage, detection, segmentation and tracking all-together play a crucial role, because the accuracy of the VGR system depends on it. A gesture in our case is the motion of the human hand particularly the palm region along with fingers either in folded or open mode. So basically the palm region is the region of interest (ROI) in our case. A process for segmenting the hand region using skin color detection and motion-based segmentation followed by a tracking algorithm is discussed here. Segmentation refers to picking only the palm region leaving the rest part in each frame and thus creating a binary image. A flowchart for hand

segmentation is shown in Fig. 3.2. Tracking here refers to locating the palm in each frame. The task at hand is to effectively build an approach to track the segmented palm and to learn the temporal evolution of its motion. Temporal evolution is about the trajectory of the ROI for the entire gesture. Thus, the entire trajectory of a gesture is converted into a single image consisting of the contour of the gesture trajectory through the simultaneous process of segmentation and tracking.

3.3.1.1 Skin segmentation

The objective of this stage is to extract the hand region from the image frame. The hand region is obtained in this stage by the following steps: face detection and removal, change of color-space for illumination compensation, hand region detection, morphological filtering and smoothening followed by retrieval of the largest connected area *i.e.* the hand (shown in Fig. 3.3). Identifying the range of skin color of the person in the image frame is a problem as the pixel intensities are subjected to change in illumination. Variation in size and color among people also poses a problem. Moreover, human skin color does not fall randomly in a given color space but clustered at a small area in the color space [68]. So, the color needs to be represented in a color space where the skin class is most compact in order to be able to tightly model the skin class. The RGB color-space is not perceptually uniform, which means distances in the space do not linearly correspond to human perception. In addition, RGB color space does not separate luminance and chrominance, and the R, G, and B components are highly correlated. The luminance of a given RGB pixel is a linear combination of the R, G, and B values. Therefore, changing the luminance of a given skin patch affects all the R, G, and B components.

To tackle the illumination variations problem, RGB images are transformed into HSV and YCbCr color spaces. HSV and YCbCr are two color spaces that separate the chrominance ([H S] or [Cb Cr]) and luminance (V or Y) components [71]. One of the major advantages of using these color spaces in skin detection is that a skin color boundary can be specified intuitively by a user. The HSV based detection is well suited for images with uniform background. In the case of YCbCr color space, transformation and efficient separation of color and intensity information is easy as compared to HSV. The skin color cluster shows more compactness in YCbCr space as compared to other spaces [68]. Moreover in YCbCr space, skin and non-skin colors show

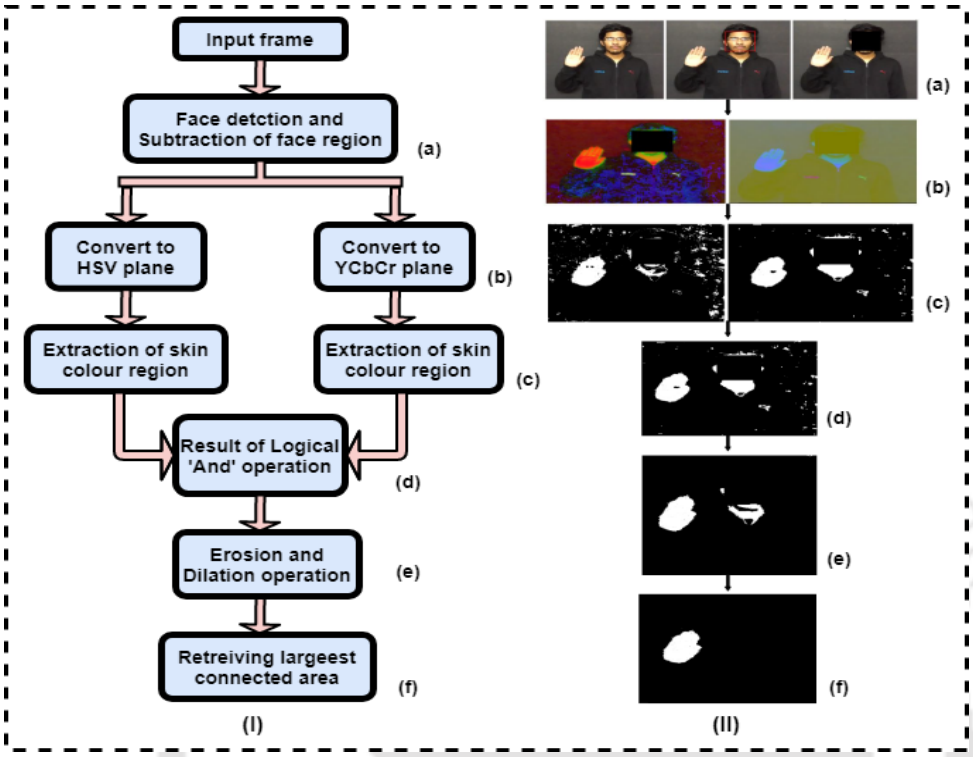


Figure 3.3: Hand segmentation steps: (I) Processing steps and (II) Corresponding outputs where (a) Face detection and removal, (b) RGB to HSV and YCbCr, (c) Extraction of skin, (d) Logical 'AND', (e) Erosion and dilation, and (f) Smoothening and retrieval of the largest connected area.

minimum overlap under different illumination conditions. So YCbCr color space works for the complex color images with uneven illumination. Here, both the color-spaces are used with an adjustable range to suit the color space of the person for segmentation in effective and efficient way. The output images in each steps of segmentation process is shown in Fig. 3.3. First, the face region in the frame is removed by using an inbuilt Viola-Jones [196] face detection algorithm (using haar cascade filter). After removing the face region, the RGB image is converted to HSV and YCbCr images, and sub-sequently thresholding is done to the chrominance components of both HSV and YCbCr color spaces. The threshold is determined by computing histogram of all the components. By ignoring the luminance channel, the variation in background illumination factor can be reduced, and thus making it suitable to be used for skin color segmentation. A logical 'AND' operation is performed between the images to obtain the skin likely region. After getting the skin color segmented regions, two approaches are tried to get the ROI: in first approach, morphological operations like erosions and dilations of convex hull technique are performed on the segmented region. Erosion first deletes the small white spots in the image

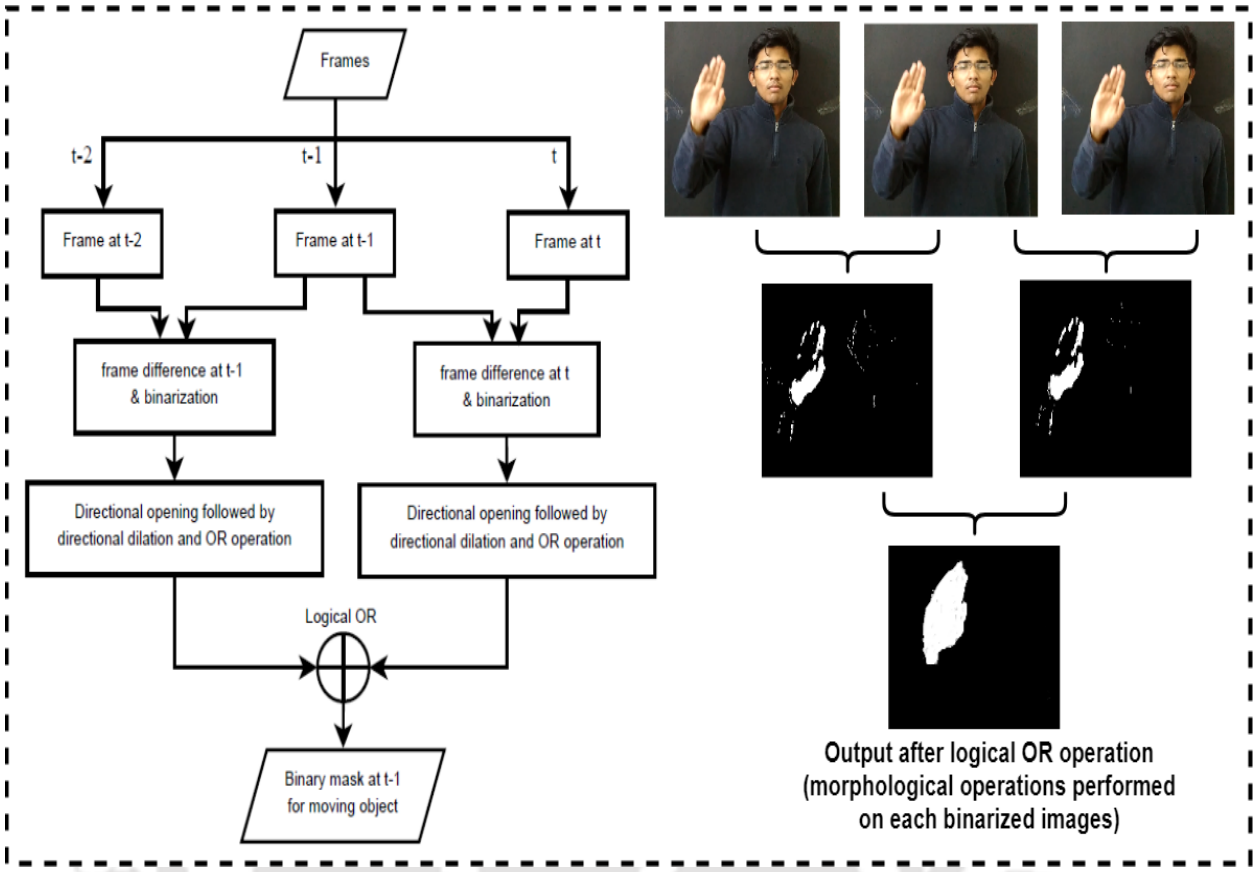


Figure 3.4: Flowchart of three-frame difference method with corresponding outputs for motion-based segmentation of the hand.

followed by dilation which fills the gaps. The largest connected segment obtained corresponds to the palm region of the hand; and in second approach a median filter is used to remove the *salt and pepper* noise. The median filter replaces each pixel in the image with the median value from the square neighbourhood. To remove Gaussian noise from the input image, a two dimensional Gaussian filter is used. Then the resulting image is binary thresholded using Otsu thresholding [197] where ROI is assigned white color. After the contours are obtained from the previous step, the region with maximum connected area is chosen as the palm region.

3.3.1.2 Motion-based segmentation

Assuming the background is static, the frame difference method is one of the easiest ways to detect moving objects in a frame. However, the frame difference algorithm suffers from some limitations like – the occurrence of ghost foreground regions and foreground apertures.

Ghost foreground regions occur due to the motion of the objects. During frame differencing, an ambiguity may occur between real foreground regions and ghost foreground regions. The other drawback of frame difference is the occurrence of foreground object aperture (FOA). FOA occurs if the object is texture-less and/or the intensity gradient of the image is significantly less. The probability of occurrence of FOA is significantly high in the case of moving skin regions as they have less texture and intensity gradient. In order to avoid the occurrence of ghost foreground regions, Kameda and Michihiko [83] proposed a 'double-difference of frame' (DDF) or three-frame difference method. In this method, three frames at time $t-2$, $t-1$, and t are selected. The DDF method performs a logical 'AND' operation over thresholded difference frames between frames at $t-2$ and $t-1$, and frames at $t-1$ and t. The DDF algorithm produces a narrow region for a moving object if the object has less texture and/or intensity gradient. The use of morphological operations can reduce FOA in a difference frame. A dilation along the direction of motion of an object can reduce FOA with the inclusion of ghost foreground regions. Motivated by this fact, a morphological enhancement-based three-frame difference method is proposed to detect moving hand regions (shown in Fig. 3.4). In our proposed method, morphological dilation is applied to each of the thresholded difference frames. In order to reduce inclusion of background regions in foreground regions of a thresholded difference frame, dilation should be performed in the direction of motion of the foreground objects. However, in case of articulated objects like hands, foreground motion could be complex. We approximate complex movements as a combination of motions in four directions – $0°, 45°, 90°, 135°$ with respect to the horizontal direction. Directional opening can be used to select a region in a particular direction. After directional opening, a dilation in the perpendicular direction of the opening process grows a region in the direction of its motion. Finally, a logical 'OR' operation is performed on the two morphologically enhanced thresholded difference frames to obtain a moving hand mask. The logical 'OR' operation helps in including more moving skin regions between the consecutive frames.

3.3.1.3 Tracking of the hand using a double-tracking system

The next step is to track the hand after successful hand detection through the segmentation process. Existing tracking algorithms suffer from various flaws and therefore there is a need to develop an algorithm that overcomes some of the existing constraints like changing the shape

Algorithm 1: Proposed double-tracking algorithm

Data: Read the initial video frame along with the segmented hand region
Output: Hand-trajectory-based-contour-images
Initialize: Initialize the particle filter tracker according to the HOG features of the segmented region and
Set: Set optimum number of particles (Thr) of the particle filter to represent the hand region
Initialize: Initialize N particles of the particle filter with associated weight $1/N$ in the segmented area
Loop:
if $N \geq Thr$ **then**
- Change the detected area according to the segmented hand position
- Apply CAMShift algorithm to find and save the centroid of the detected area

else
- Double the searching window
- Do segmentation through skin and motion segmentation after the hand detection
- Detect the new region-of-interest and find HOG features
- Initialize the particle filter according to the new HOG features and insert N particles
- Count the number of particles
- Go to **Loop** till the end of the video frames

end

of the hand, a fixed searching window for the target hand, occlusion or blurring of the hand while tracking, etc.

In traditional tracking algorithms like CAMShift or particle filter, the initial tracking region needs to be selected manually. But to make a robust system, automatic selection of tracking regions is necessary. In this proposed system, initialization of the first tracking window has been made automatic by considering the detected hand as the initial tracking window. Detection should be proper to set the tracking window properly as the initial tracker will decide the accuracy of the complete trajectory of the gesture video. After the initialization of the tracking region, a proper selection of features is essential. Here we are using a particle filter for tracking purposes along with a histogram of oriented gradients (HOG) features as a measurement model. As the video progresses, detected particles of particle filter may start decreasing and a time comes when tracking is lost due to the loss of all particles. This is due to the change in hand shape or occlusions or blurring of the target hand. To reduce these challenges, a modified system is defined using the re-detection of the hand according to the newly defined area. So, detection is performed again to eliminate the issues of tracking as soon as the number of observable particles decreases. Until detection, the searching area is doubled so as to enclose a greater

area to find the occluded hand portion through the visible particles and then skin filtering and three-frame differencing are done. Logical 'AND' operation is performed between the skin-segmented and three-frame differenced images to get the target hand. Then new particles are generated according to the HOG features of the newly identified region-of-interest and particles are inserted again. If the number of visible particles is greater than a threshold, then the particle filter tracker is run again, and the whole process continues till the end of all the video frames. To form the gesture trajectory, centroids of the detected hand regions are calculated and marked each time through CAMShift tracker and then the trajectory is generated by joining the smoothed centroids.

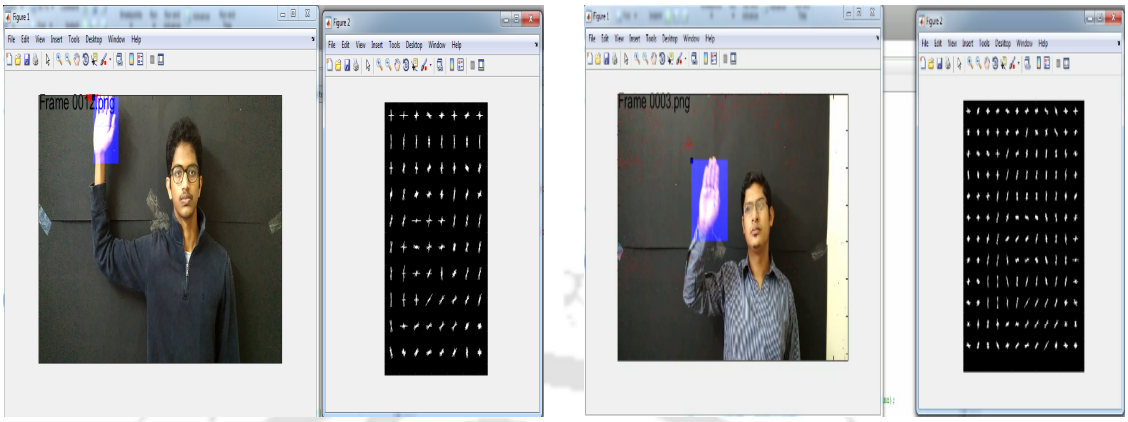
(i) **Modified particle filter framework:** During the gesture trajectory, the captured video sequences may suffer from blurring and/or occlusion. In such a scenario, tracking an object becomes very difficult. Detection of such events during trajectory is very crucial in gesture recognition. Whenever the particles lose the target, they gradually increase their search area until they find the target again. To control such types of scenes of occlusions or blurring, an uncertainty factor is introduced to increase the search space of the particles of the particle filter tracker. This part of the work is motivated by works like [6] and [198]. Cumulative measure by all the particles in the particle filter is taken for each frame of the gesture video sequence and mutual entropy is proposed as a measure. Mutual particle entropy, H is given by:

$$H = \sum_{i=1}^{N} w_k^i log(w_k^i) \tag{3.3}$$

where, w_k is the weight of particle at k^{th} instant. When the particles converge to the target, mutual entropy tends to decrease, and it increases when the particles lose the target either because of occlusion or blurring. Thus, an uncertainty factor is introduced, which is related to mutual entropy given by:

$$E = 1 - e^{-H^{\gamma}} \tag{3.4}$$

where, γ is a constant laying in between 0 and 1, H is the mutual particle entropy. The purpose of this function is to keep the value of the uncertainty factor close to 0 when



(a) (b)

Figure 3.5: Figure showing hand region detected through HOG-based particle filter and sample of HOG features for different subjects.

entropy is low and close to 1 when entropy is high thus scaling the measurement noise accordingly.

- *Particle update:* For a particle filter framework the state update equation is given by:

$$\overrightarrow{X}_k = \overrightarrow{X}_{k-1} + k\overrightarrow{\mathcal{N}} \tag{3.5}$$

where, $\overrightarrow{\mathcal{N}}$ represents the measurement of noise and k is a scaling constant. In this work, the modified framework has the state update equation given by:

$$\overrightarrow{X}_k = \overrightarrow{X}_{k-1} + E\overrightarrow{\mathcal{N}} \tag{3.6}$$

where, $E = 1 - e^{-H^{\gamma}}$. Thus, for a very confident measurement, the noise gets scaled down to a low value, and the search space becomes small and vice versa.

- *Measurement model:* The weight assigned to each particle depends on the closeness to the neighborhood target features. If the measurement is closer to the target, then higher weights will be given to the particles. These weights give the probability of the presence of the target in the neighborhood of each particle. Different measurement models have been proposed in the literature. In this work, histograms of oriented

gradients (HOG) features are generated from the segmented hand portion for each frame and used for measurement. HOG basically returns the counts of occurrences of the gradient orientation in localized portions of an image. This portion is called a cell. The cells representing the HOG features are shown in Fig. 3.5. This represents the image-oriented gradients and it is different for the hands of different persons due to varied shapes and sizes. Even the HOG features change along the trajectory of the gesture due to the variation in shape and size of the moving hand. By this measurement model, we try to represent the hand model along the gesture trajectory. The weights assigned to the particles in each frame in the neighborhood of the hand regions are higher than the weights of the particles associated with the non-region-of-interest.

- *Occlusion handling:* When occlusion is detected with the use of uncertainty measure, a forward prediction filter is activated. It predicts the next state as a combination of a few previous states. The details of the method can be found in the paper [6]. Gang Yu et al. [7] suggested object tracking in case of occlusion by using an incremental principle component analysis (PCA)-based method. In our work, the traditional particle filter has been modified to handle the problem of occlusion or blurring scenes in the video. But the gesture databases used in this work have only illumination variations and complex background (sown in Fig. 3.13), but no such cases of occlusion. So, to get an idea of the effectiveness of our proposed method, we tested our method on one of the databases used by Ref [7] which is a classic PCA-based method. Fig. 3.6 shows the effectiveness of the proposed method compared to other methods like the incremental PCA-based approach.

- *Re-sampling of particles:* This is the final step in every iteration of the particle filter algorithm. Here, N particles are randomly picked from the existing particle set according to the updated weights. Thus the particles with lower weights will be less picked and will finally die out. Whereas the particles with higher weights will be picked more than once and another set of N particles will be chosen from the existing particles.

(ii) **CAMShift algorithm:** The CAMShift algorithm uses a color histogram of the moving

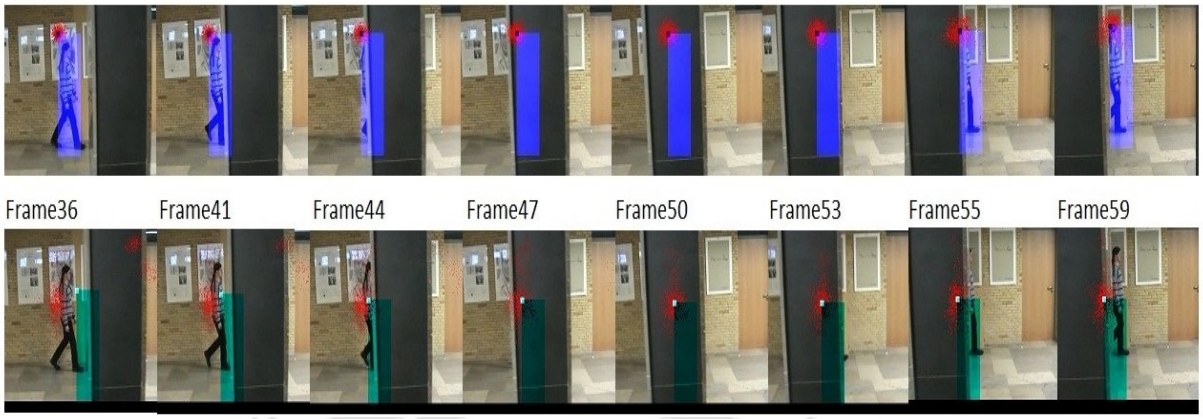


Figure 3.6: Tracking results using the proposed method (first row) [6] and PCA-based method (second row) [7].

target as target mode and is thus known as the target tracking algorithm. CAMShift algorithm when used as a mean shift algorithm has advantages like simplicity and fast speed to process and converge. The window size of this algorithm is constant so that the object location cannot be exactly detected when the size of the object changes and hence tracking loses the path sometimes. Moreover, it also loses its path when there is some skin-colored object that causes occlusion. To mitigate these problems, we have to go for a double-tracker system. As explained above, the particle filter tracker is used to keep track of the segmented hand region frame after frame according to the hand shape defined by HOG features. And, CAMShift is used to capture the trajectory of the centroid of the palm in the sequence of frames in the input gesture video to generate the gesture trajectory. The centroid is calculated from the first-order moments of the pixels (white) of the maximum connected area obtained from the previous steps. The end of the gesture is detected by the absence of a connected area greater than the threshold.

The CAMShift algorithm with related equations [98] are given below.

- Computation of 0^{th} and 1^{st} moment: The 0^{th} and the 1^{st} order moments are defined as:

$$M_{00} = \Sigma\Sigma I(x,y),\ M_{10} = \Sigma\Sigma x I(x,y),\ M_{01} = \Sigma\Sigma y I(x,y) \tag{3.7}$$

In the above equations, *I(x,y)* is the pixel value at the position *(x,y)* in the image.

- Compute the new center of feature: As the image is binary, the centroid of the hand in a frame is also the centroid of the total frame which is given by:

$$x_c = \frac{M_{10}}{M_{00}} \quad \text{and} \quad y_c = \frac{M_{01}}{M_{00}} \tag{3.8}$$

(iii) **Smoothening of the obtained gesture trajectory:** The gesture trajectory is traced out by joining all the calculated centroids in a sequential manner. The trajectory obtained by joining the centroid points could be noisy due to various reasons. Some common reasons are - the points being too close, varying the hand shape/orientation could lead to isolated points far from the trajectory, some unintentional movements and trembling of hand *etc.* The final gesture trajectory is obtained by a technique of considering an average of the mean values of three successive video frames which reduces various noises to a great extent.

$$(\hat{x}_t, \hat{y}_t) = \left(\frac{x_{t-1} + x_t + x_{t+1}}{3}, \frac{y_{t-1} + y_t + y_{t+1}}{3} \right) \tag{3.9}$$

Thus, a dynamic hand gesture (DG) can be interpreted as a set of points joined together in a spatio-temporal space (Fig. 3.7):

$$DG = \{(\hat{x}_1, \hat{y}_1), (\hat{x}_2, \hat{y}_2), \cdots\cdots, (\hat{x}_t, \hat{y}_t)\} \tag{3.10}$$

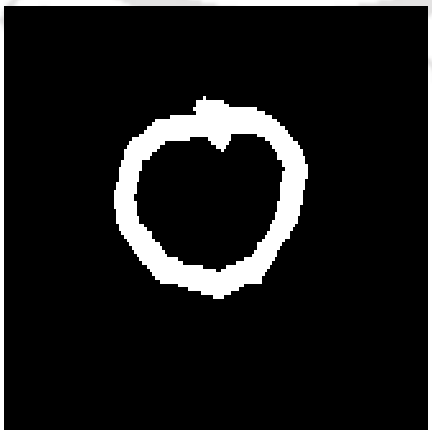


Figure 3.7: Tracking gesture for English alphabet 'O'.

3.3.2 CNN Network Architecture and Training

A convolutional neural network (CNN) architecture is used for feature extraction and gesture recognition on the segmented and tracked images obtained in the image processing section. A simple custom structure with two convolution layers and two fully-connected layers is chosen here since more complex networks with more layers can become overfitting and they try to memorize a particular work making it less generic. For training purposes, we have used EMNIST (letters) dataset [199] as our training dataset. And, for testing, we have used three databases. The details of network architecture and training are given below.

3.3.2.1 Network Architecture

The CNN architecture used for this work is shown in Fig.(3.8). The first layer of the network is a convolutional layer. The design of this layer comprises 32 filters, with a kernel size of 5×5 each. The activation function used is *Rectified Linear Units* (ReLU) which generally outperforms sigmoid and tangent in speed for training without any tradeoff for accuracy. ReLU activation introduces non-linearities and offers less saturation. Efficient gradient back-propagation can be achieved by ReLU along with momentum [163]. After that, we use a max-pooling layer. A 2×2 box non-overlapping max pooling with stride 2 in both horizontal and vertical directions is applied. The next layer is again a convolution layer which consists of 64 filters, each with the size of 7×7 and ReLU activation function. The output of this step undergoes 2×2 box max-pooling. The weight of all the filters of these convolution layers gets updated during the training phase. The output feature maps can be seen to contain a few evident features from input as training proceeds. To avoid overfitting while learning, there is a dropout layer. The parameter for this layer is set to 0.4 which means the layer randomly excludes 40% of neurons (and weights associated with it) in the layer. It is configured to speed up the training process and reduce over-fitting. The 2D matrix data is converted to a column vector (one dimensional) in order to be processed by fully connected dense layers. A fully connected layer with 1024 neurons and ReLU activation function process the result from the previous step. A softmax activation function is used on the output fully connected layer to turn the outputs into probability-like values. These values are used for prediction.

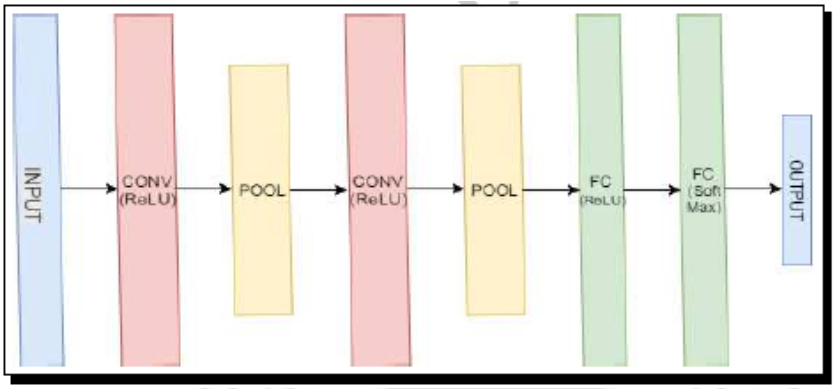


Figure 3.8: CNN architecture for hand gesture recognition (inspired by LeNet [8]) CONV: Convolutional, FC: Fully connected.

3.3.2.2 Training

As already mentioned, the EMNIST dataset (letters) has been used as our training dataset. Here categorical cross-entropy (logarithmic) loss function is used as a cost function and *stochastic gradient descent* (SGD) with momentum is used as an optimization technique for convergence of the model. Momentum in gradient descent is a method in which some fraction from the previous update is added to the current update to compound the effect of repeated updates in a particular direction. As a result, the descent in the desired direction is faster. The learning rate of the model is slowed down with an increase in the number of epochs in order to allow the model to converge more accurately. With a deeper network, there is a problem of loss in training and testing. This is not because of overfitting, but due to the slow learning of a deep network. So, generally, the learning rate is varied in different steps to get rid of this. The choice of learning rate is 0.01 if the epoch count is less than 25 and is reduced to 0.001 for epoch count up to 50. To further promote slow learning, values of $1e^{-4}$ and $1e^{-5}$ are used, till 75 and 100 epochs respectively.

3.4 Experimental Results

3.4.1 Databases and Experimental Set-up

The performance of the model has been tested on two publicly available datasets and our own in-house dataset. The publicly available datasets are: EMNIST (letters) dataset [199] and

NITS hand gesture database (with no variation in gesticulation speed or pattern) [9]. Our in-house dataset is a limited dataset of English upper case letters and numerals 0-9 mimicking the structure of the EMNIST dataset. The dataset is created with the help of three subjects and the background is kept simple. In this experimentation, datasets that define the same classes which are upper case letters of the English alphabet are used. So, the same model that has been trained using EMNIST (letters) training dataset, is used for testing which reduces the burden of training requirements again and again. For testing purposes, EMNIST (letters) test images and the images obtained (through the process explained in the image processing section) from NITS hand gesture database and our own video dataset are used and the results are discussed in the next subsections. So, in a nutshell, our goal is to recognize the images of the English upper case alphabets that have been obtained from the three different databases. All experiments are performed in a workstation with Intel® Core™i5-4570 CPU at 3.2 GHz and 8 GB in RAM without any GPU usage.

3.4.2 Results using EMNIST (letters) dataset

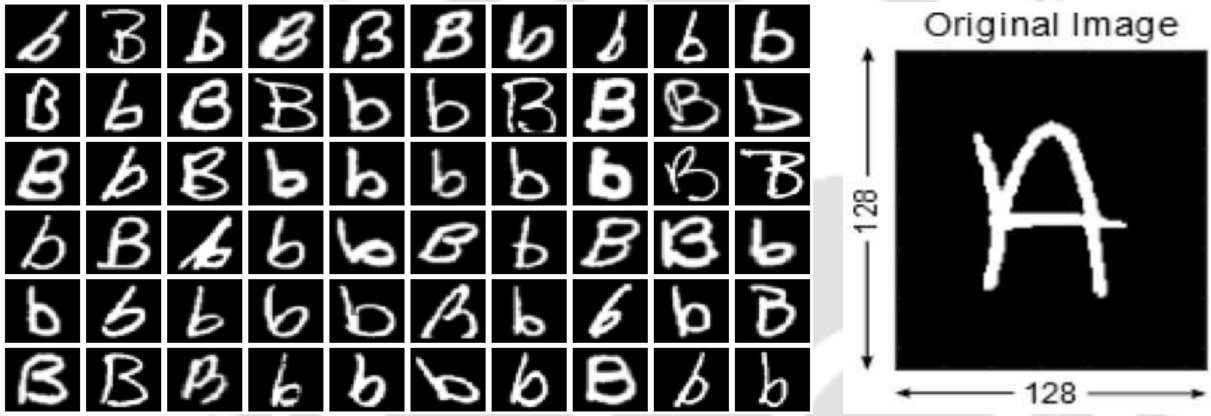
The EMNIST (Extended MNIST) [199] dataset is a set of handwritten character/digits derived from the NIST (National Institute of Standards and Technology) [8] database converted to a 28×28-pixel image format and dataset structure that directly matches the MNIST (Modified NIST) [8] dataset. The MNIST database is a large database of handwritten digits that is commonly used for training and testing various image processing and machine learning systems. The MNIST database of handwritten digits has a training set of 60,000 examples and a test set of 10,000 examples. It is a subset of a larger set available from NIST. The digits have been size-normalized to 28×28 and centered in a fixed-size image. The MNIST dataset was extended including handwritten letters by the name EMNIST which was published in 2017. It contains 2,40,000 training images and 40,000 testing images of handwritten digits and 1,24,800 and 20,800 handwritten images for training and testing respectively for letters. There are six different splits provided in this dataset and each is provided in two formats: binary and CSV (combined labels and images). The description of six categories of the EMNIST dataset is given in Table 3.2.

EMNIST (letters) contains 1,45,600 characters of 26 balanced classes (shown in Fig. 3.9a

Table 3.2: Categories of EMNIST dataset [13]

Categories	Classes	Training	Testing	Valid.	Total
By class	62 unbal[1]	697,932	116,323	No	814,255
By merge	47 unbal	697,932	116,323	No	814,255
Balanced	47 bal[2]	112,800	18,800	Yes	131,600
Digits	10 bal	240,000	40,000	Yes	280,000
Letters	26 bal	124,800	20,800	Yes	145,600
MNIST	10 bal	60,000	10,000	Yes	70,000

[1]unbal = *unbalanced*; [2]bal = *balanced*



(a) **(b)**

Figure 3.9: (a) Different variants of 'B' and 'b' from EMNIST (letters) dataset, (b) One particular sample of 'A' from EMNIST (letters) dataset.

and Fig. 3.9b). Our model is trained on 1,24,800 samples and tested on 20,800 samples (800 test samples per class). The accuracy and loss obtained during training and testing phases are listed in Table 3.3.

The training and testing categorical cross-entropy losses and accuracies as a function of a number of epochs are shown in Fig. 3.10 and Fig. 3.11 respectively.

The comparison among state-of-the-art results for all variants of the EMNIST dataset with our result on the portion of the letters of the EMNIST dataset is given in Table 3.4. [13] has obtained a better result compared to our method. There are two reasons for this improvement: First, the incorporation of Markov random field (MRF) based filter banks along with DCT and Gabor filters as prefixed convolutional operators to derive the feature maps. By combining Markov random field models, to extract salient information from the raw data, along with the convolutional neural networks, to implicitly learn high-level features, it has combined the

Table 3.3: Results on the training and testing set of EMNIST (letters) dataset

Parameters	Epoch 1	Epoch 20	Epoch 50	Epoch 100
Learning Rate	0.01	0.01	0.0001	0.0001
Training Loss	0.9451	0.0965	0.0464	0.0454
Training Acc.	0.7556	0.9628	0.9819	0.9825
Testing Loss	0.3071	0.2461	0.2809	0.2836
Testing Acc.	0.8999	0.9338	0.9354	0.9360

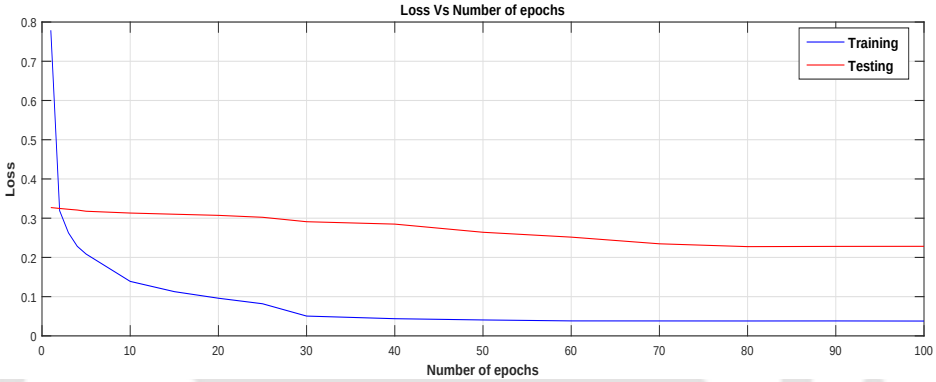


Figure 3.10: Training and testing loss as a function of number of epochs for EMNIST (letters) dataset.

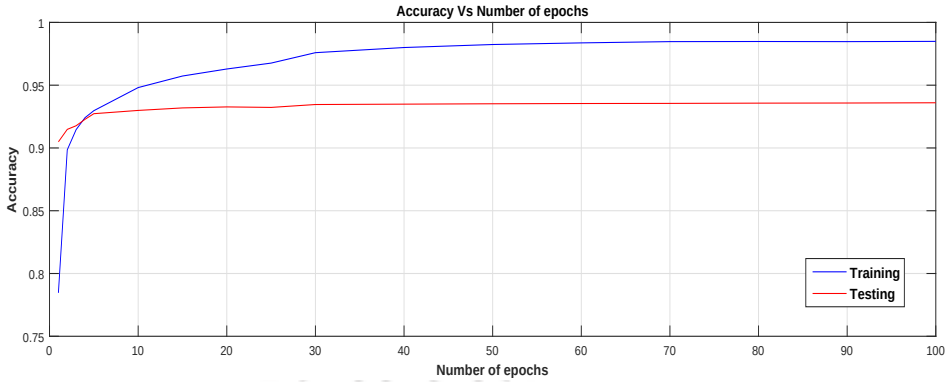


Figure 3.11: Training and testing accuracy as a function of number of epochs for EMNIST (letters) dataset.

respective strengths of MRFs and CNNs to bring the expressive power of a deep architecture. Another major difference is that [13] has used a deeper convolutional layer structure compared to our two-convolutional layer model. Our model has been kept simple so that it can also be applied in resource constraint environments without the use of GPUs. This comparison is shown here to make it clear that though our model is not state-of-the-art, still it is good enough to carry out our recognition procedure on the hand-trajectory-based-contour-images derived from

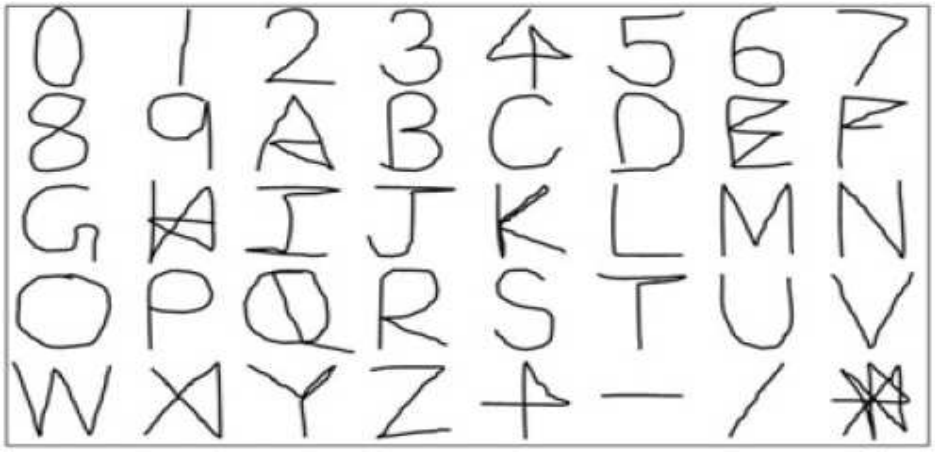


Figure 3.12: Gesture set of NITS hand gesture database [9].

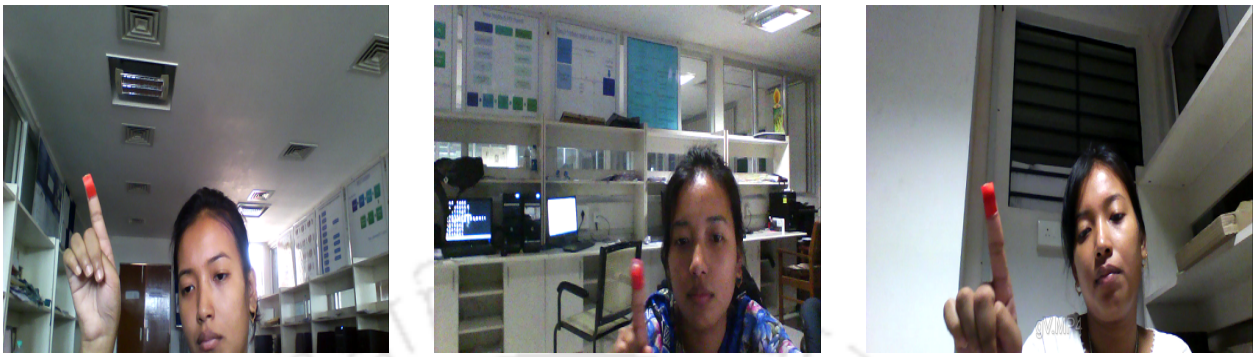
real gesture databases. Due to the limited number of data samples in the gesture databases, the model is trained with the EMNIST image dataset and it is saved in the hdf5 format to be used directly in the recognition system. Here data augmentation and transfer learning type approaches are not adopted due to limited sample size in the real gesture databases and are used only for testing.

Table 3.4: Comparison of error rate (%) with state-of-the-art methods on EMNIST dataset

EMNIST dataset	Linera classi-fier [199]	OPIUM classi-fier [199]	MRF-CNN [13]	Our method
Balanced	49.17	21.98	9.71	–
By class	48.19	30.29	12.33	–
By merge	49.49	27.43	9.06	–
Letters	44.22	14.85	4.56	6.40
Digits	15.30	4.10	0.25	–
MNIST	14.89	3.78	0.33	–

3.4.3 Results using NITS hand gesture database

NITS hand gesture database [9] is developed by the Speech and Image Processing Lab of NIT, Silchar, India in seven different categories with different variations in pattern and speed. Here, the database I have been used as it resembles our training dataset. The database consists of 40 gestures, which are alphabets (A-Z), numbers (0-9) and mathematical operators (+, -, /, *) (Fig. 3.12). Out of all these, we have used the alphabet gestures for testing purposes. The



(a) (b) (c)

Figure 3.13: Screenshots from NITS hand gesture database showing varying illumination conditions.

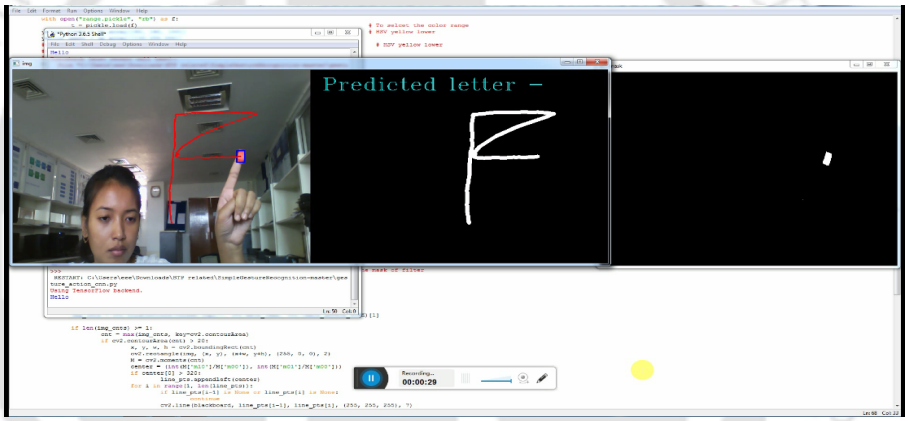
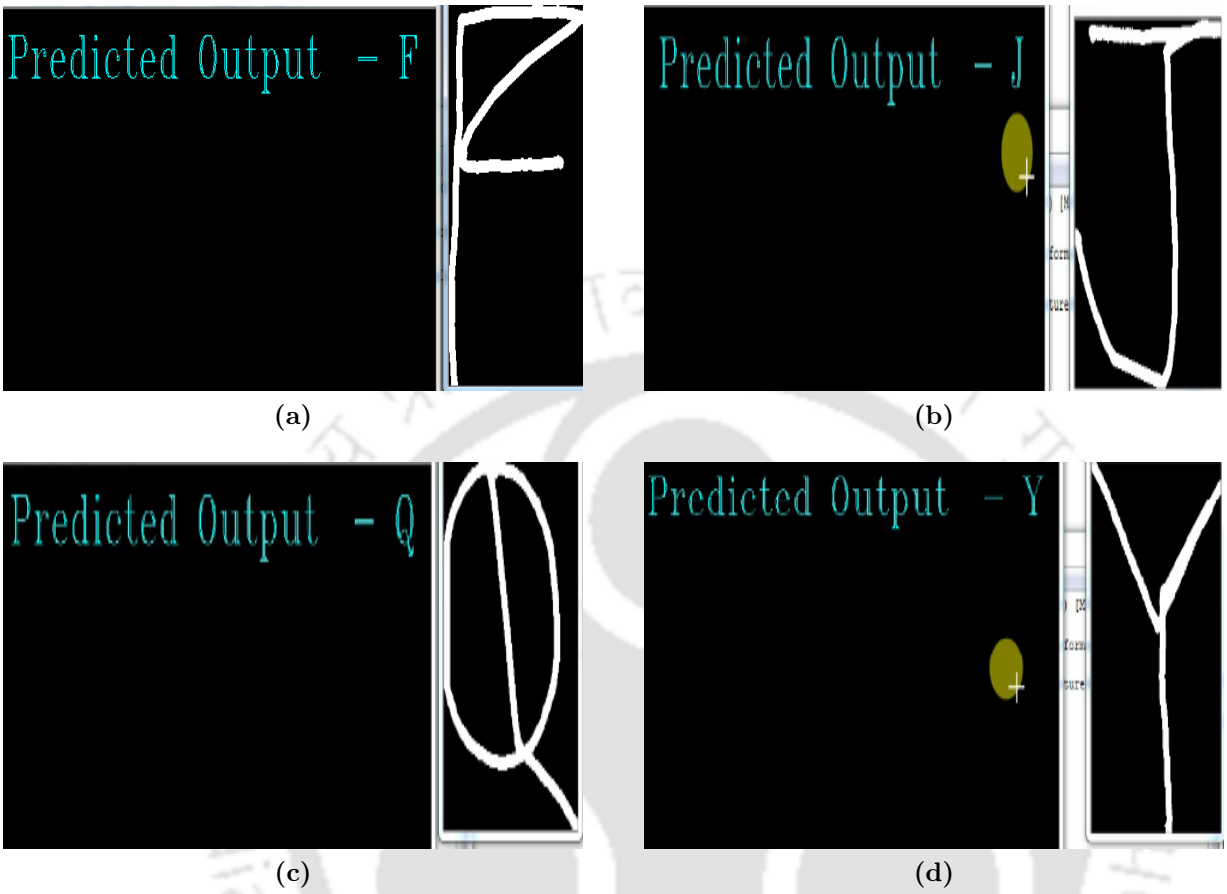


Figure 3.14: A snapshot of the gesture recognition system that has been used for user interface.

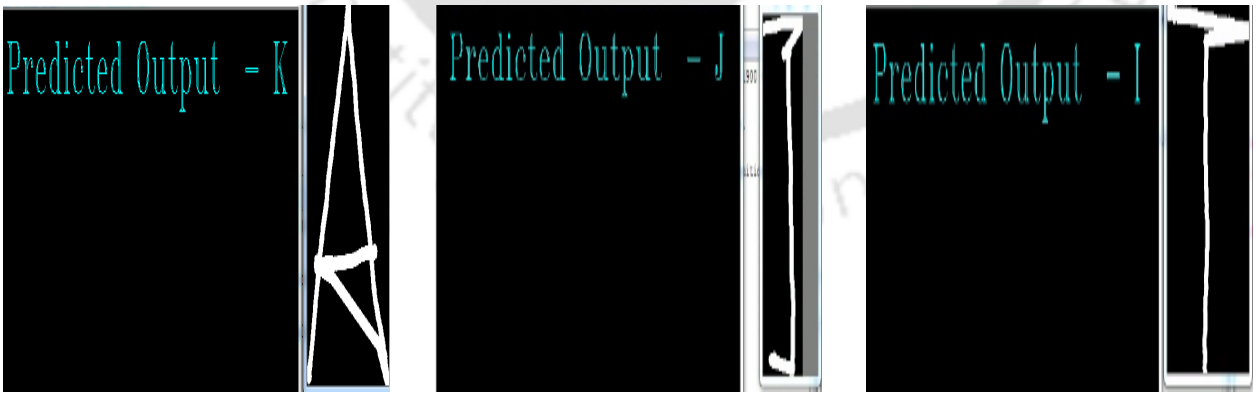
database is recorded in different sessions with multiple participants. Gestures are captured in an uncontrolled environment with both simple and complex backgrounds, and under varying illumination conditions (shown in Fig. (3.13)). However, color cues show variations in the skin color in different lighting conditions, and also skin color changes with the change in human tribes, leading to many segmentation issues and also create restrictions due to the presence of skin-colored objects in the background. To overcome the disadvantage of skin color detection, participants of the dataset have used markers on hand or fingers in the dataset which enhances segmentation accuracy leading to better performance.

In Fig. 3.14, a snapshot of the gesture recognition system has been shown that is used as a handy user interface in the experimentation. Our model has been able to recognize most of the English alphabet gestures (shown in Fig. 3.15). However, there are a few gesture examples where the CNN classifier has misclassified as shown in Fig. 3.16. The model predicts the testing



(a) (b) (c) (d)

Figure 3.15: Correct predictions of different test samples: (a) 'F', (b) 'J', (c) 'Q', (d) 'Y'.



(a) (b) (c)

Figure 3.16: False Classifications: (a) Prediction of test sample 'A' as 'K' due to its resemblance in training samples of EMNIST (letters) dataset, (b) Prediction of test sample 'I' as 'J', (c) Prediction of test sample 'T' as 'I'.

samples 'A' as 'K' shown in Fig. 3.16a. This is due to the structural difference of the testing sample 'A' with the training one as shown in Fig. 3.12 and Fig. 3.9b. The same is the case with gestures 'I' and 'T' where the system has not been able to predict correctly mainly due to the similarity of the gesturing alphabets with some other training examples (Refer Fig. 3.16b and Fig. 3.16c). It is also seen that there is some minor structural difference in the test samples of 'E', 'F', 'H' and 'X' compared to training samples, still, our model has been able to recognize them correctly (shown in Fig. 3.15a). So, in this way, our prediction accuracy of the whole system with CNN, both as feature extractor and classifier, stands at 93.02% due to the above-mentioned misclassifications.

One easy way of increasing the accuracy performance by removing such type of ambiguity is the addition of different variants of the misclassified test samples in the training set. In the case of a limited database, an increase in training samples through data augmentation can be a very good option that basically diversifies the training process. However, in our case, the model has been trained on the EMNIST dataset. And, the saved trained model has been used in the recognition system to test some other gesture databases like the NITS gesture dataset and our in-house dataset. Though the training-testing accuracy curves on the EMNIST dataset suggest that the model is not overfitting, but through the experimentation, it is quite clear that it has tried to memorize only the particular training samples making it less generic. To overcome this problem, we have used the CNN structure only for feature extraction purposes and used a support vector machine (SVM) classifier for the recognition task. It is expected that this hybrid CNN–SVM model will show better performance compared to each individual classifier based on the fact that the hybrid system compensates the limits of individual classifiers by incorporating the merits of both classifiers. Since the hypothetical learning technique for CNN is equivalent to that for the MLP, so CNN is just an extension model of the MLP. The learning calculation of MLP depends on the Empirical Risk Minimization, which endeavours to limit the errors in the training set. At the point when the first isolating hyperplane is found by the back-propagation calculation, regardless of whether it is the local or the global minima, the training process stops and the calculation doesn't keep on further developing the isolating hyperplane arrangement. Thus, the speculation capacity of MLP is lower than that of SVM. Then again, the SVM classifier tries to minimize the generalization error on the concealed information with a fixed distribution on the training set, by utilizing the Structural

Risk Minimization standard. Here the isolating hyperplane is a global ideal arrangement. It is determined by tackling the quadratic programming problem, and the margin between the two classes of training samples attains its maximum. Subsequently, the generalization ability of SVM is maximized to improve the performance. Through this process, we have been able to increase the recognition accuracy to a great extent and achieved satisfactory performances compared to the state-of-the-art methods. This is explained in the next section.

3.4.3.1 Classification using SVM

An SVM is a supervised classifier for both linearly separable and linearly nonseparable data [200]. In a linearly nonseparable case, the input data is mapped to some higher dimensional space, where the data can be separated linearly. To do so, a non-linear mapping is done through kernels. This mapping from lower to higher dimensional spaces makes the classification of the input data simpler and more accurate. There are many kernels used for this operation and we have chosen different SVM kernel functions such as linear, polynomial (quadratic), Gaussian and radial basis function (RBF) to compare the performance measure. For linear and polynomial auto kernel scale and for Gaussian and RBF, 0.79 and 1 are used as kernel scale and their accuracy performance is shown in Table 3.5. The same is shown in graph form in Fig. 3.17. In a CNN architecture, the convolutional layers extract the features and the fully connected layers are responsible for recognition. The feature vector of dimension 2048 extracted from the output of the second fully-connected layer of the CNN structure is given as input to the SVM recognition classifier after L_2 normalization. It can be observed that SVM with RBF kernel provides the highest accuracy of 99.09% for the alphabet gestures and hence, we have chosen RBF kernel in the final model. The RBF kernel on two samples x_i and x_j, represented as a feature vector in some input space, is given by -

$$K(x_i, x_j) = \exp\left(-\frac{(\|x_i - x_j\|)^2}{2\sigma^2}\right), \quad \sigma > 0 \tag{3.11}$$

which can be written as -

$$\boldsymbol{K}(x_i, x_j) = \exp(-\beta\|x_i - x_j\|^2), \quad \beta > 0 \tag{3.12}$$

β is a parameter of a Gaussian RBF kernel to handle non-linear classification. In the

Table 3.5: Results (accuracy in %) with different SVM kernels

SVM Kernel Function	Kernel Scale	Classification Accuracy (%)
Linear KF	auto	95.93%
Quadratic KF	auto	97.67%
Gaussian KF	0.79	97.13%
RBF KF	1	99.09%

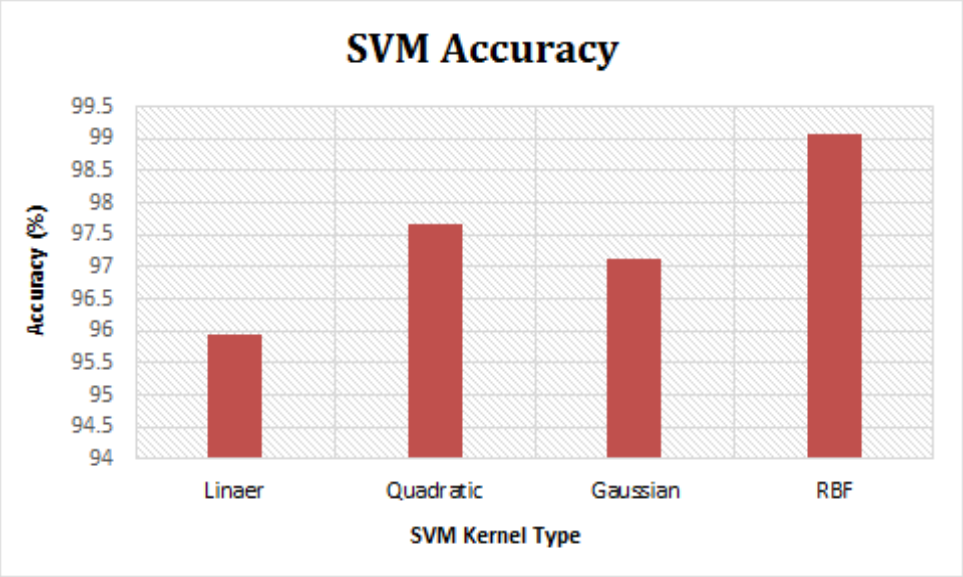


Figure 3.17: Graph showing accuracy of SVM classifier with different kernels.

experiments, we have used $\sigma = 0.7$ or $\beta = 1$ as the basis for trial and error. C is another parameter for the soft margin cost function, which controls the influence of each individual support-vectors. A large C gives low bias and high variance, whereas a small C gives higher bias and lower variance. So, this process involves trading error penalties for stability. The RBF function is straightforward that can achieve nonlinear mapping, and uses relatively few parameters. In our experiment, we have used a non-linear support vector machine (SVM) with a Gaussian radial basis function kernel with C as a parameter with values 1 and 10. The kernel trick is a strength of SVM. The risk of overfitting is less in SVM and has good generalization. There are basically two methods to use SVM for a multiclass problem [136]. The first approach is called "one-versus-one" (OVO) that constructs one classifier per pair of classes and combines binary classifiers in a way to form a multi-class classifier by selecting the most voted class. So, $N(N-1)/2$ binary SVM classifiers are needed, each of them is trained on the samples of the two

corresponding classes. The second method is called "one-versus-all" (OVA) and it considers all the classes of data in one optimization problem. In fact, for each classifier, the considered class is fitted against all the other classes, so for N number of classes, N SVM classifiers are required. In this work, we have used support vector machines (SVMs) that implement a one-versus-all (OVA) multiclass approach.

3.4.3.2 Comparison with state-of-the-art methods

A comparison among state-of-the-art results for the NITS hand gesture database with our result is given in Table 3.6. In [201], bare hand English alphabet gestures are classified with the help of handcrafted features using different classifiers like kNN, SVM and ANN. [9] has used handcrafted features in SVM and ANN classifiers for classification. Ours is the first work on this dataset where deep network features are used along with an SVM classifier and it has been able to achieve state-of-the-art performance.

Table 3.6: Comparison with other methods for NITS database

Work	Dataset	Features	Classifier	Performance
[201]	Bare hand gestures, English alphabets	Handcrafted features	kNN, SVM, ANN	87.82%, 88.31%, 90.58%
[9]	English alphabets	Handcrafted features	ANN, SVM	96.41%, 96.95%
Our method	Red marker, Bare hand, English alphabets	Deep features	SVM classifier	**97.76%** (Bare hand), **99.09%** (Red marker)

3.4.4 Results using in-house dataset

One limited in-house dataset has been created with the help of three participants and is mainly used for testing purposes. It consists of 78 videos with 26 classes of English upper case letters. For simplicity, a very simple black background is kept with full-sleeve attire worn by the

Figure 3.18: Sample frames from our own in-house dataset.

subjects as shown in Fig. 3.18. Due to the simple background and similarity with the training set, the segmentation of the hand is very accurate which consequently gives us a recognition accuracy of 100% for our small dataset. So, this is a clear indication that a simple background results in good segmentation which subsequently gives better recognition performance.

3.5 Summary

In this work, a template-based framework in a spatial domain has been presented to recognize trajectory-based isolated hand gestures. Here a technique for converting a gesture video into a single 2D image depicting the contour of the gesture trajectory is introduced as a preprocessing step. This is done by combining the classical preprocessing steps of segmentation and tracking. These preprocessing steps are adopted in such a way as to eliminate the constraints of illumination variations and occlusion in gesture videos. The output image of the image processing step is fed to a pre-trained deep network that is robust at learning shape features. Thus, the extracted deep features have been applied to an SVM classifier to classify different gestures. Due to the built-in feature learning capability of deep networks, considerably improved performance is obtained in the results.

The advantages of this framework are - (a) skin-segmentation is carried out along with motion-based segmentation and the outputs of both the methods are logical 'AND'ed to properly segment out the hand portion, (b) through the double-tracking system, the problem of blurring or occlusion has also been tried to minimize and (c) template-based methods are relatively simple and suitable for some specific user interfaces with limited gesture classes.

But, a high variety of gestures with wide spatio-temporal variations executed by the same or different people is difficult to match by template matching approaches. Another major disadvantage of this model may occur in the segmentation process, especially with complex background scenes. Accurate segmentation of hand or body parts from the captured images/videos still remains a challenge in computer vision for many preoccupied constraints such as illumination variations, background complexity, skin-color variation, occlusion and the articulated shape of the hand. So, the performance of the model mainly depends on the segmentation process of the hand which is basically dependent on the selection of a proper threshold for the color cues. Moreover, a high variety of gestures with wide spatio-temporal variations executed by the same or different people is difficult to match by template matching approaches. In our next work, one major objective is to skip the hand segmentation module to eliminate such types of problems. The objective is to propose a tracking framework by considering the challenges due to the different shapes, sizes and colors of hands. For trajectory-based gestures, motion cues can be another added information useful for hand detection. This cue has motivated us to develop an effective representation of hand movement directly from raw gesture videos. Another modification may be the fusion of different streams or layers which can capture the temporal variations along with the spatial variations more effectively. All these attempts will be carried out in the next chapter.

4

Two-Stream Fusion Model using 3D-CNN and 2D-CNN via Video-Frames and Optical Flow Motion Templates for Hand Gesture Recognition

In the process of hand gesture recognition, proper detection, segmentation and tracking of the moving hand become challenging due to the varied shape, size and color of the hand. Here the objective is to track the movement of the hand irrespective of the shape, size and color. And, for this, a motion template guided by optical flow (OFMT) is proposed. OFMT is a compact representation of the motion information of a gesture encoded into a single image. In the experimentation, different datasets using bare hand with an open palm, and folded palm wearing green-glove are used, and in both cases, we could generate the OFMT images with equal precision. Recently, deep network-based techniques have shown impressive improvements as compared to conventional hand-crafted feature-based techniques. Moreover, in the literature, it is seen that the use of different streams with informative input data helps to increase the performance in the recognition accuracy. Hence, in this work, we propose a two-stream fusion model for hand gesture recognition where the network consists of two layers - a 3D convolutional neural network (C3D) that takes gesture videos as input and a 2D-CNN that takes OFMT images as input. Though each stream can work independently, they are combined with a fusion scheme to boost the recognition results. The efficiency of the proposed two-stream network has been shown on two databases.

4.1 Introduction

In spite of having several advantages of color-based skin segmentation methods, accurate segmentation of the hand or any other body part is still a big challenge. The accuracy of color-based skin detection methods is severely affected by the presence of skin-like colors in the background. In our previous chapter, the skin segmentation method with illumination compensation has been used to segment the hand portion from the background. Apart from skin-color-like objects in the background, segmentation may be difficult due to some other reasons like variation in shape and appearance of the hand, clothing of the gesture signer, image resolution, *etc.* Moreover frame-difference method is also not that suitable for a small-sized target object.

So, one major objective in this work is to skip the hand segmentation module. The objective is to propose a tracking framework by considering the challenges due to the different shapes, sizes and colors of hands. For trajectory-based gestures, motion cues can be another added information useful for hand detection. This cue has motivated us to develop an effective representation of hand movement directly from raw gesture videos. This has motivated the development of an effective representation directly from raw gesture videos and we propose a motion template called *optical flow-guided motion template* (OFMT). OFMT is a compact representation of the motion information of a gesture encoded into a single image. In the experimentation, different datasets using a bare hand with an open palm, and folded palm wearing green gloves are used, and in both cases, we could generate the OFMT images with equal precision. Recently, deep network-based techniques have shown impressive improvements as compared to conventional hand-crafted feature-based techniques. Moreover, in the literature, it is seen that the use of different streams with informative input data helps to increase the performance in the recognition accuracy. This work basically proposes - a two-stream fusion model for hand gesture recognition and a compact yet efficient motion template based on optical flow. Specifically, the two-stream network consists of two layers - a 3D convolutional neural network (C3D) that takes gesture videos as input and a 2D-CNN that takes OFMT images as input. C3D has shown its efficiency in capturing spatio-temporal information of a video. Whereas OFMT helps to eliminate irrelevant gestures providing additional motion information. Though each stream can work independently, they are combined with a fusion scheme to boost

(a) (b) (c) (d) (e) (f)

Figure 4.1: Various approaches for moving object detection: (a),(b) A pair of consecutive video frames from our in-house dataset, (c) Optical flow, (d) Binarized difference image, (e) Morphological operation on the binarized difference image, (f) Inter-frame difference image.

the recognition results. We have shown the efficiency of the proposed two-stream network on two databases.

4.2 Background and Related Work

Automatic detection of moving hand is a key motive in the hand gesture recognition system. Segmentation and tracking become very challenging when color information is not available, moving object is small, the background is complex, or when the scene is changing frequently. Various approaches for moving object detection using pixel-level change are: background subtraction, frame difference and optical flow [82]. Stabilized background subtraction is always a noisy affair making it vulnerable for long and varied video sequences [82]. Though frame difference methods can easily detect motion, it shows poor performance in localizing the object. Apart from this, the choice of temporal distance between frames is a tricky question. It basically depends on the size and speed of the moving object. In such cases, when prior knowledge of moving objects like appearance and shape is not available, basically optical flow can still provide effective motion-based cues for the detection and localization of objects. All the above-mentioned motion detection schemes are applied as shown in Fig. (4.1) and it is seen that optical flow gives the proper needful information.

For achieving good performance, the hand gesture recognition system should be independent of different textures like shape, size and colour of the hand for tracking purpose. Out of various methods, the estimation of the motion field is invariant to shape and appearance (at least in theory) and can be used directly to describe human gestures/actions. Optical flow and motion-templates are the two main motion-based representation methods used for this purpose [100].

Generally, both these methods are used separately for motion estimation since both have their own advantages. Motion templates like motion-energy-image (MEI) and motion-history-image (MHI) [2] give a global aspect of motion without the requirement of segmentation of the moving object. It is computationally very efficient making it suitable for real-time applications [111]. In [157], authors applied an approach combining MHI with statistical measures and frequency domain transformation on depth images for one-shot-learning hand gesture recognition. Due to the availability of the depth information, the background-subtracted silhouette images were obtained using a simple mask threshold. Whereas in [180], authors used pseudo-color based MHI images as input to convolutional networks. On the other hand, optical flow is obtained from the movement of the target object in a video scene. Though it is computationally a little expensive, still it has the advantage that it can produce good results even in the presence of a bit of camera movement. [107] also proposed to calculate changes of optical flow that focuses on optical flow differences between frames (motion boundaries). Yacoob and Davis [108] used optical flow measurements to track predefined polygonal patches placed on interest regions for facial expression recognition. [109] presented an integrated approach where optical flow is integrated frame-by-frame over time by considering the consistency of direction. In [110], the optical flow was used to detect the direction of motion along with the RANSAC algorithm which in turn helped to further localize the motion points. There are only a few examples like [202] where the optical flow is combined with the motion template. In [202], the authors claimed that the combined technique can give a better discrimination power to describe local motions in a global time-space representation. In this work, we have also proposed a motion template driven by optical flow. Our method is different from [202] in reducing background noises through an update rule. In our work also, better discrimination can be seen for optical flow-guided motion template (OFMT) over conventional motion templates. This combined method can accurately detect the location and thus provide the contour of the moving object just like a tracker. The effectiveness of this method is quite impressive for long and varied video sequences.

Indifference to hand-crafted features, there is a growing trend towards feature representations learned by deep neural networks [72, 159, 162, 170, 180, 203, 204]. Ciregan *et al.* [162] has shown that the use of multi-column deep CNNs with multiple parallel networks improves recognition rates of single networks by $30 - 80\%$ for various image classification tasks. Similarly, for large scale video classification, Karpathy *et al.* [163] have shown the best results on combining

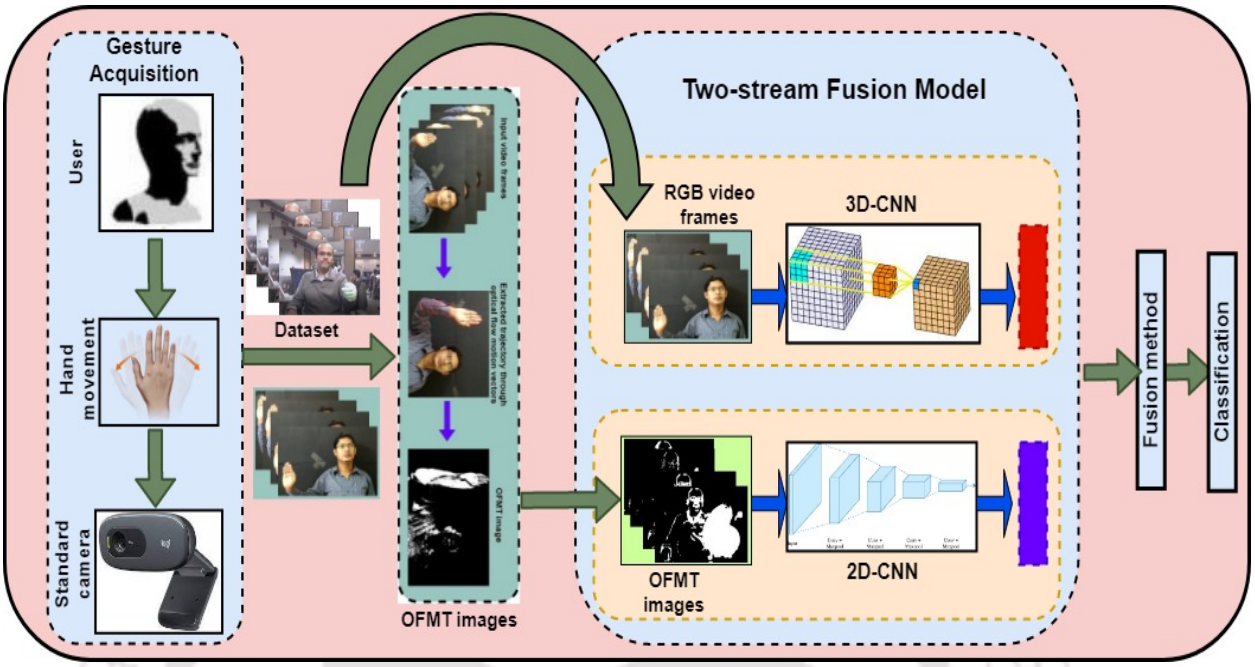


Figure 4.2: Proposed framework for hand gesture recognition.

CNNs trained with two separate streams of the original and spatially cropped video frames. Simonyan and Zisserman [159] proposed separate CNNs for the spatial and temporal streams that are late-fused and that explicitly use optical flow in the context of action recognition. To recognize sign language gestures, Neverova *et al.* [164] employed CNNs to combine color and depth data from hand regions and upper-body skeletons. A two-stream model with two C3D layers that takes RGB and optical flow computed from the RGB stream as inputs were used by [170] for action recognition. [171] used a hidden two-stream CNN model which takes only raw video frames as input and directly predicts action classes without explicitly computing optical flow. Here the network predicts the motion information from consecutive frames through a temporal stream CNN that makes the network $10x$ faster [171], without computing optical flow which is time-consuming. But still, two hidden layers in one stream are computationally not so efficient. Moreover, state-of-the-art performance is achieved through traditional optical flow precomputed for the convolution layer in a two-stream network for action and hand gesture recognition [159, 170, 180]. But this approach of precomputation of optical flow motion vectors through CNN is computationally expensive and storage inefficient [171].

So in this work, the objective is to track the movement of the hand irrespective of the shape, size and color of the hand. Another main intention is to propose a resource-efficient network

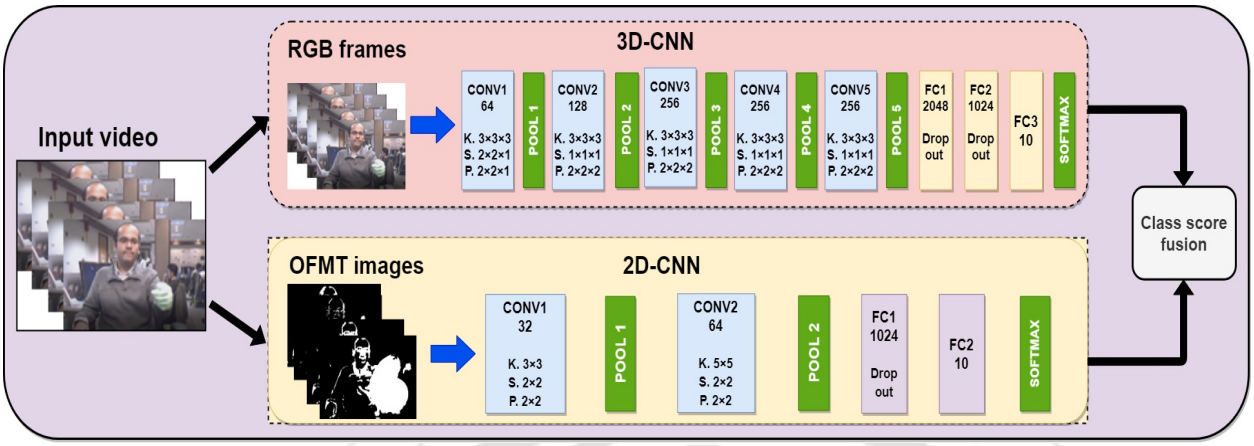


Figure 4.3: Proposed two-stream network for hand gesture recognition (K=kernel size, S=stride size, P=pooling size, max-pooling is used here).

in terms of data and processing power as much as possible without compromising much in its performance. The complete framework is shown in Fig. (4.2). Here we propose a deep learning-based two-stream network that is zoomed in Fig. (4.3). The first layer/stream is a 3D-CNN (C3D) network in the proposed two-stream architecture, which is used to capture the spatial as well as temporal information from RBG video frames. The second layer is a 2D-CNN model where the input is an optical flow-guided motion template (OFMT) image. OFMT is a hybrid representation, proposed in this work that is obtained by combining optical flow with the motion template to get the advantage of both the methods for temporal evaluation analysis. OFMT is used to provide additional motion pattern information which in turn helps to eliminate irrelevant gestures. The output score of both the layers is fused using an ensemble method to boost the final output.

4.3 The Proposed Methodology

As shown in Fig. (4.3), the proposed gesture model is composed of two main streams/layers - the first layer is a 3D-CNN (C3D) network in a two-stream architecture to capture spatial as well as temporal information of a gesture. The second layer is a 2D-CNN network, where the input is an optical flow-guided motion template (OFMT) image. OFMT is a hybrid, compact and robust motion representation, proposed in this work. The OFMT template is obtained by combining optical flow information with the motion template. In this way, we get the advantage

of both the methods for temporal evaluation analysis. The OFMT is used to provide motion pattern information, which in turn eliminates irrelevant gestures. The proposed OFMT can nominally reduce computational complexity and memory requirement as well. The output of a CNN classifier is a class-membership probability for each of the gestures under consideration, and thus the prediction results of 3D-CNN and 2D-CNN networks are fused through a simple probability-based ensemble method at the decision level to boost the final output by taking advantage of both the models.

Here, we will first talk about proposed OFMT images then about the two-stream network.

4.3.1 Proposed Optical Flow-guided Motion Templates (OFMT)

The input to the 2D-CNN is a compact motion template. For this, a hybrid representation is proposed for encoding temporal information of a gesture by combining optical flow motion information with motion templates. This representation takes advantage of both optical flow and motion-energy-image (MEI) and motion-history-image (MHI) templates.

4.3.1.1 Motion-templates

MEI represents where motion has occurred in an image sequence; whereas MHI represents how an object is moving [111]. MEI describes the motion-shape and spatial distribution of motion, and MHI is the function of the intensity of motion of each pixel at that location. MEI-MHI can be implemented by the following algorithm.

MEI-MHI Algorithm [2]:

- **Image sequences**

$$I(x, y, t) = (I_1, I_2, ..., I_n). \tag{4.1}$$

- **Image binarization**

$$B(x, y, t) = |I(x, y, t) - I(x, y, t-1)|. \tag{4.2}$$

$$\text{where, } B(x, y, t) = \begin{cases} 1 & \text{if } B(x, y, t) > \xi \\ 0 & \text{otherwise} \end{cases}$$

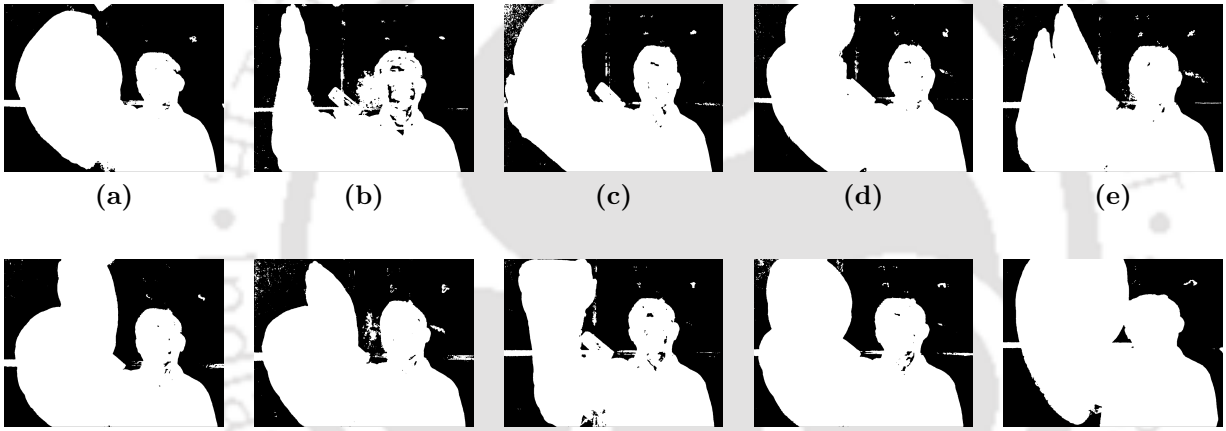
- **MEI**

$$E_\tau(x,y,t) = \bigcup_{i=0}^{\tau-1} B(x,y,t-i). \tag{4.3}$$

- **MHI**

$$H_\tau(x,y,t) = \begin{cases} \tau & \text{if } B(x,y,t)=1 \\ max(0, H_\tau(x,y,t-1)-\delta & \text{otherwise} \end{cases} \tag{4.4}$$

where τ decides the temporal extent of the motion (in terms of frames) and δ is the decay parameter.



(a) (b) (c) (d) (e)

(f) (g) (h) (i) (j)

Figure 4.4: MEI images: (a)-(j) representing gestures 0-9.

(a) (b) (c) (d) (e)

(f) (g) (h) (i) (j)

Figure 4.5: MHI images: (a)-(j) representing gestures 0-9.

4.3.1.2 Optical flow

Optical flow is the apparent motion or displacement of objects/pixels as perceived by an observer. Optical flow indicates the change in image velocity of a point moving in the scene, also called a motion field. Here the goal is to estimate the motion field (velocity vector) which can be computed from horizontal and vertical flow fields. Ideally, the motion field represents the $3D$ motion of the points of an object across $2D$ image frames for a definite frame interval. Out of different optical flow techniques found in the literature, the most common methods are: (a) Lucas-Kanade [101] (b) Horn-Schunk [102] (c) Brox 04 [103] and (5) Brox 11 [104] (d) Farneback [105]. The most widely used techniques for optic flow estimation is the differential methods. Differential techniques can be classified into local methods where they optimize some local energy-like expression, and global schemes which attempt to minimize a global energy functional. Local strategies like Lucas-Kanade [101] generally offer relatively high robustness under noise, but do not give dense flow fields. On the other hand, global techniques like Horn-Schunk [102] yield flow fields with high density, but are experimentally known to be more sensitive to noise [205]. The choice of optical flow method primarily depends on the power of the resulting histogram of optical flow (HOF) or motion boundary histogram (MBH) descriptor. HOF gives the optical flow displacement vectors in horizontal and vertical directions. The intuitive idea of MBH is to represent the oriented gradients computed over the vertical and the horizontal optical flow components. Once horizontal and vertical optical flow components are obtained, histograms of oriented gradients are computed on each image component. The outcome of this process is a pair of horizontal (MBHx) and vertical (MBHy) descriptors. Since MBH represents the gradient of the optical flow, locally constant camera motion is removed and information about changes in the flow field (i.e., motion boundaries) is kept intact. MBH is more robust to camera motion than optical flow [107].

One major problem with optical flow estimation is that it is very sensitive to noise and outlier due to background motion. To get rid of this problem Gaussian smoothing operation is done to image frame $k(x, y, t)$, where (x, y) denotes the location of the pixel and t denotes time. Smoothing is done prior to differentiation, by convolving each frame with some Gaussian kernel $G_\sigma(x, y)$ of standard deviation σ:

$$I(x, y, t) := (G_\sigma * k)(x, y, t), \tag{4.5}$$

The low-pass effect of Gaussian convolution removes noise and other destabilizing high-frequency outliers. In a subsequent procedure, σ also called the 'noise sale', can be chosen of different values. While some moderate pre-smoothing improves the results, great care should be taken not to apply too much pre-smoothing, since this would severely destroy important image structure.

Another problem is tracking points that are moving long distances with a higher speed of motion. This can be mitigated by a course-to-fine optical flow estimation by forming an image pyramid. While applying a single-scale Lucas-Kanade optical flow algorithm, it is assumed that the window has a little motion so that high-order terms in the derivation of Taylor expansion can be ignored. But this assumption fails for an object with long-distance movement in consecutive frames. In this case, the iterative coarse-to-fine method helps a lot which is applied to an image pyramid building multiple copies with different resolutions for each image frame. Each level in the pyramid is one-fourth of the size of the previous higher resolution level. To get rid of the small motion constraint, first, we start from the lowest resolution level. Then the iterative optical flow is used to estimate potential motion velocity at this level and then expand it to a higher resolution level through the warp and upsampling process. This is done because lower resolution images can provide better optical flow for large motion compared to higher resolution images. So we first start with a coarse resolution and warp it to fine resolution through interpolation. But the main problem with this technique is that it makes the computational efficiency a little expensive. In the iterative process, potential optical flow is estimated in one level on the window corresponding to one pixel, then we reapply the estimated vector to warp the image to a new position. This process is repeated for several iterations until the residual motion is sufficiently small. All these steps are shown in Fig. (4.6)

For tracking the motion of the hand, Lukas-Kanade Algorithm (LKA) [101] is applied by extracting the desired features of the gesture. For that purpose, a mouse call back function is adopted to select the desired feature on the first frame that will be tracked successively in the next frames of the video. The optical flow motion vectors track these features during the trajectory of the dynamic gesture. These vectors are tracked on a per-frame basis. By joining

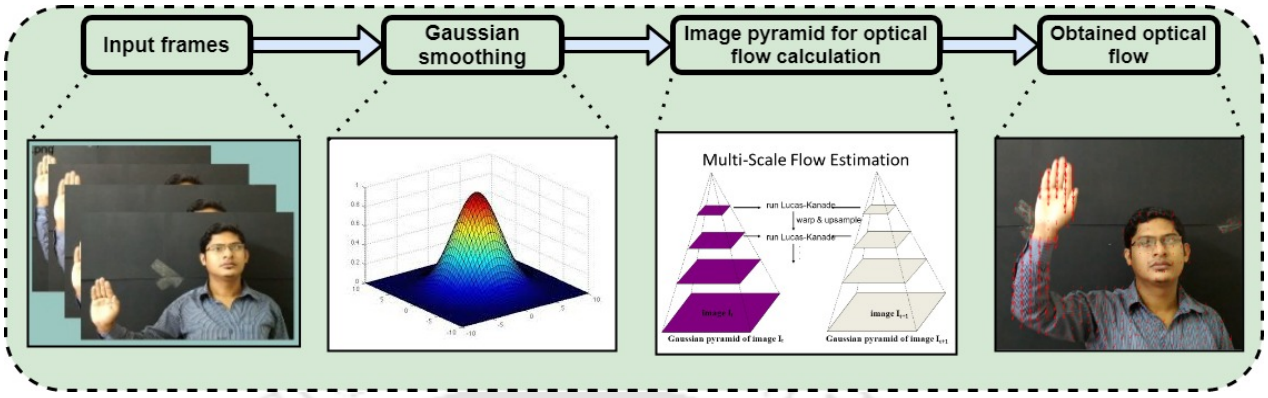


Figure 4.6: Steps to obtain optical flow from input video frames.

these flow vectors on the mask, the trajectory of the gesture is recovered from the continuous motion of the hand.

Optical flow tracking algorithm:

- A customized black window mask is created with the same dimensions as that of the original frame of the video.
- The first frame of the video is read and it is called $frame_{old}$.
- The initial points are selected on the $frame_{old}$ using a mouse call back function (mouse cursor). These points are named $points_{old}$.
- Initialize a new list to store optical flow vectors.
- Initialize Lucas-Kanade parameters with the window size as of 15×15 and pyramid level as 4. Large motion is ignored by the pyramidal model.
- while for the next number of frames:
  $frame_{new}$ = Next frame of the video converted into a gray scale image.
  $points_{new}$ = Generated by optical flow with $frame_{old}$, $frame_{new}$, $points_{old}$ and LK parameters
- copy $frame_{new}$ into $frame_{old}$.
  copy $points_{new}$ into $points_{old}$.
  Append $points_{new}$ to list.

- if length(list) $\geq$ 2:

  Draw line with the last two points in the list on the mask.

- Thus the optical flow vector between two frames is obtained.

4.3.1.3 Optical flow-guided motion templates (OFMT)

Now, here we present a motion template driven by the optical flow method. This combined method can accurately detect the location of the hand and also provide the contour of the moving hand just like an object tracker. MEI and MHI are generated using a binarized image, obtained from frame subtraction, using a threshold ξ as shown in Eq. (4.2). In the motion template representation, all the foreground or moving pixels (*i.e.* $B(x, y, t) = 1$) are considered for creating the templates irrespective of the duration and speed of the individual moving pixel. Motion templates basically describe the global motion of a scene and cannot fully describe the local motions of the target object. Whereas optical flow is generally used for foreground segmentation or to extract moving objects. It can better describe the local motions of the target object. In our method, optical flow is judiciously combined with the motion templates to exploit the advantages of both methods. In the proposed method, the optical flow sequence $O(x, y, t)$ representing the moving regions of the previously smoothed image is accumulated and fused together to form an optical flow-guided motion template as per the following equation:

$$E_{\tau}(x, y, t) = \bigcup_{i=0}^{\tau} O(x, y, t - i - 1) + \lambda . O(x, y, t) \tag{4.6}$$

where, τ indicates the duration of the gesture, and λ is an update parameter.

If the optical flow length $O(x, y, t)$ is small compared to a pre-defined threshold ϵ_s, then it labels the pixel (x, y) as a background point and hence λ value is taken as zero to reduce the effect of background noises. If the optical flow length value is greater than the threshold then it labels a pixel as a foreground moving point, then λ is empirically set to 5 to consider foreground pixels. In this way, the background noise is reduced in our proposed method. The saved moving points from the video frames generate a single image providing the trajectory of the gesture as shown in Fig. (4.7). These steps are performed for all the hand gesture videos and the corresponding OFMT images are pre-obtained as shown in Fig. (4.8). The entire experiment is done in python environment, with an OpenCV tool and 30 frames/s is the frame

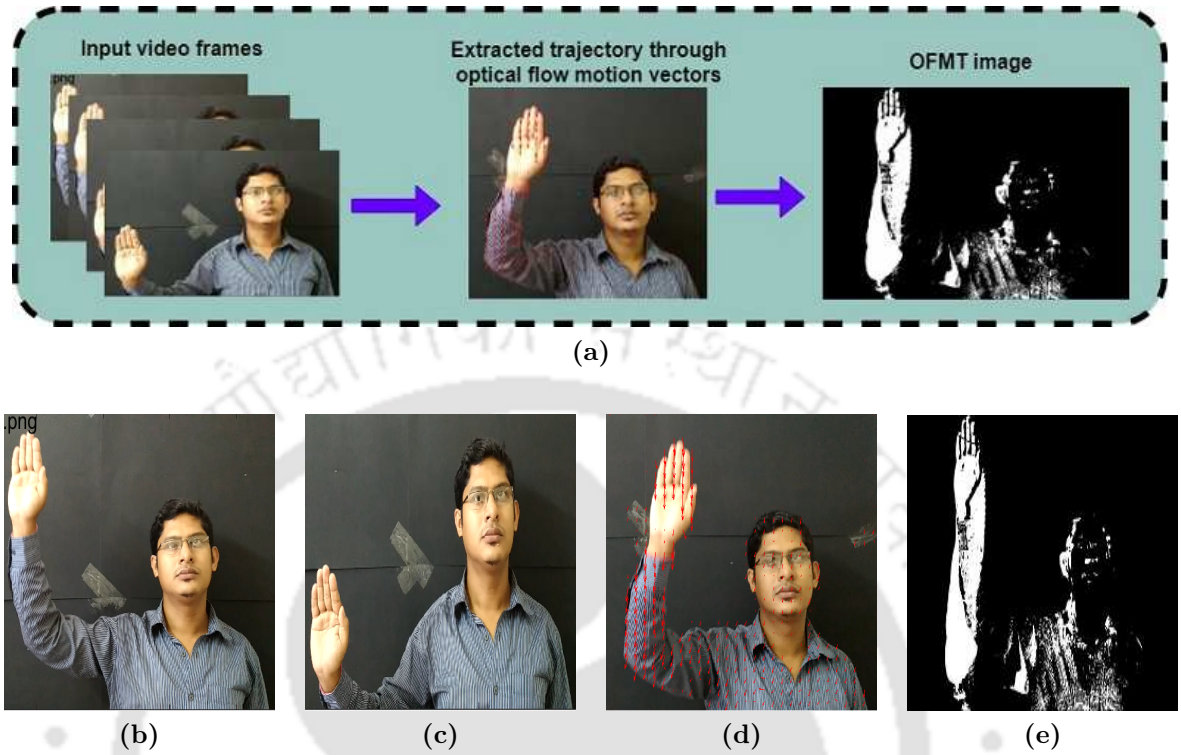


Figure 4.7: (a) Steps to obtain OFMT images, (b-c) Video frames, (d) Extracted optical flow, (e) Obtained OFMT (video frames from in-house dataset).

rate used for generating the OFMT images that are fed to the 2D-CNN network. Another advantage obtained here is that the requirement of segmenting the hand portion from the body is not needed and also, the size, shape, and color of the hand have no effects on the OFMT images.

Through the naked eye, it can be easily noticed that our proposed OFMT images give much better results than the conventional motion templates. Still, for quantitative analysis, we calculate entropy and structural similarity index measurement (SSIM) for each set of images to get a clear idea.

4.3.1.4 Entropy

The entropy of a discrete random variable X with possible values $\{x_0, x_1, x_2, ..., x_N\}$ can be defined as [206]

$$H = -\sum_{k=0}^{N} p_k log_2(p_k) \tag{4.7}$$

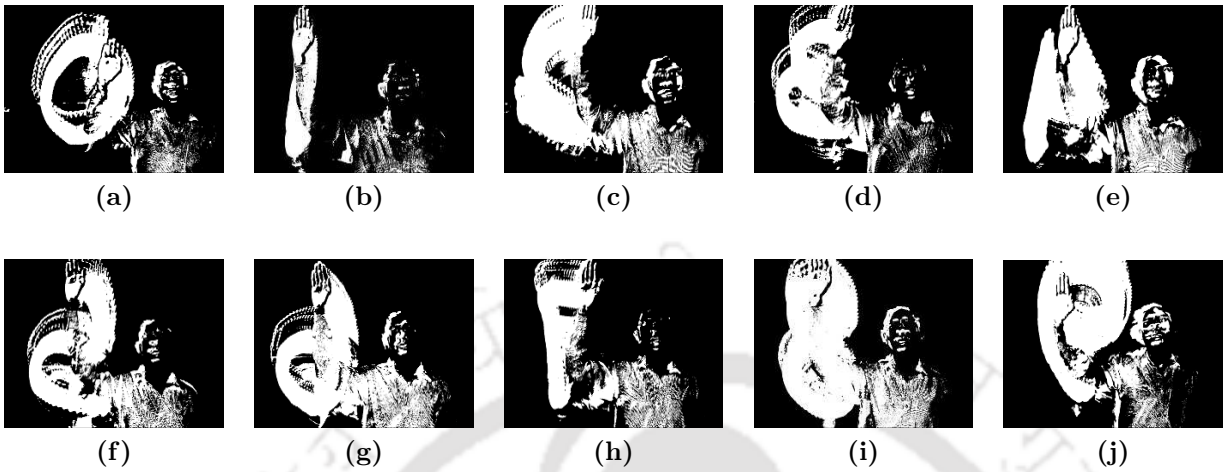


Figure 4.8: Optical Flow-guided Motion Templates (OFMT) images applied on our in-house dataset: (a)-(j) for gesture 0-9.

where, H indicates entropy and p_k is the probability associated with input k.

For a grey-scale image, the intensity value of each pixel varies from 0-255, and the possibility of a particular value occurring is random and varies with the pixel intensity values of the images. Considering an image with dimension $M \times N$ having a total of $W = M \times N$ pixels, the probability of a particular intensity value x_k occurring in the image is $p(x_k) = n_k/W_k$, where n_k is the number of occurrences of x_k among the W pixels. In this case, considering $\sum_k n_k = M$, the entropy of the image can be expressed as

$$H = -\frac{1}{W}\sum_{k=0}^{255} n_k log_2(n_k) \tag{4.8}$$

Table (4.1) shows that most OFMT images have low entropy values compared to MHI or MEI images. This is due to the fact that, from the image compression algorithm viewpoint, entropy tries to discover predictability and each aspect of predictability requires some storage to represent. The more storage that an image requires to represent its predictability, the higher is the entropy the image posses. An image that is all the same is entirely predictable and has low entropy. An image that changes from pixel to pixel might at first thought to be unpredictable, but the change might follow a pattern too. From this point we can say that if our motion template can give some idea about its gesture *i.e.* predictable in nature, it should contain low entropy value.

Table 4.1: Image Entropy and mSSIM Values for Different Motion Templates

Gestures →	0	1	2	3	4	5	6	7	8	9
MEI	1.0128	1.0358	1.0217	1.0119	1.0215	1.0090	1.0300	1.0237	0.9801	0.9227
MHI	2.4577	1.9533	2.5930	2.6417	2.4296	2.4893	2.4033	2.3524	2.6540	2.7437
OF-MT	**0.9258**	**0.6670**	1.0636	**0.9568**	**1.0130**	**0.9902**	1.0339	**0.9235**	1.0811	1.0351
mSSIM	0.8348	0.8509	0.830	0.8532	0.8821	0.8810	0.8890	0.8627	0.9102	0.8642

4.3.1.5 Structural Similarity Index Measurement (SSIM)

There are many algorithms developed to provide an index for image quality analysis. SSIM is a reference image quality indexing algorithm which is why it requires a reference image to estimate the quality of the test image. The parameters that are considered for comparison are the luminance, contrast and structure of the images [207]. These three factors are estimated from the images and a relative score is being provided to the test image. The factors mentioned are some of the important factors used by the human eye to provide a subjective analysis of the images. These physical factors are simulated with the use of the basic statistical parameters like mean, variance and covariance.

Let $x = \{x_0, x_1, x_2, ..., x_N\}$ and $y = \{y_0, y_1, y_2, ..., y_N\}$ be two discrete non-negative signals that have been aligned with each other (e.g., two image patches extracted from the same spatial location from two images being compared, respectively). And, let μ_x, σ_x^2 and σ_{xy} be the mean of x, the variance of x, and the covariance of x and y, respectively. Similarly, we have for y. Approximately, μ_x and σ_x can be viewed as the estimates of the luminance and contrast of x, and σ_{xy} measures the tendency of x and y to vary together, thus an indication of structural similarity. In [5], the luminance, contrast and structure comparison measures were given as follows:

$$l(x, y) = \frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1} \tag{4.9}$$

$$c(x, y) = \frac{2\mu_x\mu_y + C_2}{\mu_x^2 + \mu_y^2 + C_2} \tag{4.10}$$

$$s(x, y) = \frac{\sigma_{xy} + C_3}{\sigma_x\sigma_y + C_3} \tag{4.11}$$

where C_1, C_2 and C_3 are small constants given by $C_1 = (K_1L)^2$; $C_2 = (K_2L)^2$ and $C_3 = C_2/2$; respectively where typical values for K_1=0.01 and K_2=0.03. L is the dynamic range of the pixel values (L = 255 for 8 bits/pixel gray scale images), and $K_1 \ll 1$ and $K_2 \ll 1$ are two scalar constants. The general form of the structural similarity index measurement (SSIM) between signal x and y is defined as:

$$SSIM(x, y) = [l(x, y)]^{\alpha}.[c(x, y)]^{\beta}.[s(x, y)]^{\gamma} \tag{4.12}$$

where α, β and γ are parameters to define the relative importance of the three components. Specifically, we set $\alpha = \beta = \gamma = 1$, and the resulting SSIM index is given by

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \tag{4.13}$$

which satisfies the following conditions:

- **Symmetry:** $SSIM(x, y) = SSIM(y, x)$.
- **Boundedness:** $SSIM(x, y) \leq 1$.
- **Unique maximum:** $SSIM(x; y) = 1$ if and only if $x = y$.

The SSIM indexing algorithm for image quality assessment using a sliding window approach. The window moves pixel-by-pixel across the whole image space. At each step, the SSIM index is calculated within the local window. If one of the images being compared is considered to have a perfect quality, then the resulting SSIM index map can be viewed as the quality map of the other (distorted) image. The distribution of the window can be rectangular or Gaussian distribution. The Gaussian distribution is preferred to avoid the blocking effect which is predominant in a rectangular window. Here, we consider Gaussian distribution with parameters $\mu = 0$ and $\sigma = 1.5$. Finally, a mean SSIM index (mSSIM) of the quality-map is used to evaluate the overall image quality.

Here, we have compared normal motion templates with our OFMT images (Refer Table (4.1)). Generally, the zero value of SSIM indicates the worst case and one the best-case scenario. Visually an SSIM index greater than 0.94 can be considered a good image in comparison with the original image.

4.3.2 Spatio-temporal feature learning through a 3D convolutional (C3D) network

The original C3D [168] was designed for RGB videos. The number of parameters of the networks depends on the resolution of input frames. The original C3D was trained on the large-scale dataset Sport1M [163], which consists of 1.1M videos downloaded from YouTube consisting of 487 sports classes. 2D-CNN is extended to a 3D-CNN by incorporating the temporal dimension of a video sequence. In 2D-CNNs, the dimension of each feature map is $c \times h \times w$, where c represents the number of filters in the convolutional (conv) layer, h and w represents the height and width of the feature map. In 3D-CNNs, the dimension of each feature map is $c \times l \times h \times w$, where additional parameter l represents the number of frames. This network extracts the features which are compact and generic while being discriminative. As we worked on two smaller databases, a slightly different architecture with 5 conv layers is employed which has a smaller number of parameters compared to the original C3D [168] with 8 conv layers. The proposed network has 5 space-time conv layers with 64, 128, 256, 256, 256 kernels. Each conv layer is followed by a rectified linear unit (ReLU) and a space-time max-pooling layer. All 3D convolution kernels are of size $3 \times 3 \times 3$, that gives the best performance [168] with stride $1 \times 1 \times 1$. Max pooling kernels are of size $2 \times 2 \times 2$ except for the first, where it is $2 \times 2 \times 1$ and stride is $2 \times 2 \times 1$. The conv layers are followed by two dense layers with 2048 and 1024 neurons and ReLU as the activation function. To avoid over-fitting while learning, there is a dropout in each dense layers. The parameter of dropout is set to 0.4, which means the layer randomly excludes 40% of neurons. The final dense layer of the classifier has 10 neurons giving us the respective class labels where softmax function is used for activation.

4.3.3 2D motion template CNN model

As illustrated in Fig. (4.3), the proposed 2D motion template CNN model consists of two major parts - motion templates and a 2D-CNN model. The generation of the motion templates is explained in the previous section. The 2D-CNN architecture used in our method is a simple structure based on LeNet [8] (shown in Fig. (4.3)). The network has 2 conv layers with 32 and 64 kernels followed by 2 fully connected layers of size 1024 and 10. The final dense layer of the classifier has 10 neurons giving us the respective class labels. The size of the kernels is 3×3 and

5×5 respectively for the two conv layers. Each conv layer is followed by ReLU and 2×2 box non-overlapping max-pooling layers with stride 2 in both horizontal and vertical directions. A dropout of 40% is used in the dense layer with 1024 neurons to avoid over-fitting.

4.3.4 Proposed Fusion Rule

In [159], the authors used two decision level fusion methods of averaging and SVM fusion, on two identical C3D networks. In the averaging method, two softmax prediction scores are averaged to represent the output class scores. In the SVM fusion method, the features from fully-connected layers of both the streams are stacked and after L2 normalization, are input to an SVM classifier. Whereas in our proposed two-stream model, two non-identical networks are applied with all-together different inputs for each stream with different dimensions. Hence, decision level fusion is preferred here in place of feature-level fusion due to computational overhead. But, in place of just simple averaging fusion, we have formulated an empirical formula given by Eq. (4.14) for output prediction score fusion.

$$p_i = \gamma.p_i^{3D} + (1-\gamma).p_i^{2D} \quad where \;\; i = 1, 2, ..., N \tag{4.14}$$

Here, p_i^{3D} and p_i^{2D} are the prediction class scores of 3D-CNN and 2D-CNN respectively, N is the number of gesture classes, and γ is an empirical parameter. For fusion at the decision level, experiment with different values of γ such as 0.2, 0.3, 0.4, 0.5, 0.6, 0.7 and 0.8 is carried out. We investigated that γ as 0.6 achieves the best performance. This is quite justified since the score given by 3D-CNN has a greater impact than the score given by 2D-CNN. With the user-defined parameter γ, a user can judiciously select the importance to be given to each stream. The final prediction class score S is the one whose value is maximum and it is calculated as given below:

$$S = \underset{1 \le i \le N}{\operatorname{argmax}}\, p_i \quad where \;\; i = 1, 2, ..., N \tag{4.15}$$

4.4 Experimentation and Results

To evaluate the performance of the proposed method, we have carried out experiments on two databases: 1) Palm's Graffiti Digits [10] and 2) Our in-house database [72]. The following sections elaborate on all the details during the implementation and evaluation processes.

4.4.1 Databases

In our work, two datasets are employed, one is Palm's Graffiti Digits dataset [10] and another is the self-collected in-house dataset [72]. The details are described below.

(i) **Palm's Graffiti Digits dataset**: The Palm's Graffiti digits database [10] contains standard RGB 2D videos of ten subjects writing "in the air" the ten Hindu-Western Arabic numerals, 0-9, in a continuous streaming mode with video size 320×240, 30 frames/s as shown in Fig. (4.9). This database is split into three subsets, namely "GreenDigits," "EasyDigits," and "HardDigits" sets. Each one of the first two datasets contains 300 gestures (ten subjects × ten digits × three examples/digit/subject). In both datasets, the subjects used tightly folded palms while performing the gestures. GreenDigits dataset after data augmentation is used for the training phase and EasyDigits and HardDigits sets are used for the testing phase. In the GreenDigits set, subjects wear a green glove, while short-sleeves in EasyDigits. In this dataset, the subjects used tightly folded palms while performing the gestures. GreenDigits dataset after data augmentation is used for the training phase and EasyDigits and HardDigits sets are used for the testing phase. In the GreenDigits set, subjects wear a green glove, while short-sleeves in EasyDigits. HardDigits is intended for the evaluation of hand detection methods under very challenging and uncontrolled scenarios like with movement of people in the background. It has 140 videos (seven subjects × ten digits × two examples/digit/subject). The proposed OFMT images are obtained for these gesture videos and are fed into the 2D-CNN network for classification purposes in our experiments. While obtaining the OFMT images of the gestures, whether the subject is wearing any glove or using a bare hand, there is no effect in detecting and tracking the hand. This indicates that the performance of the system is independent of the shape, size, and color of the hand and it adds robustness to our method.

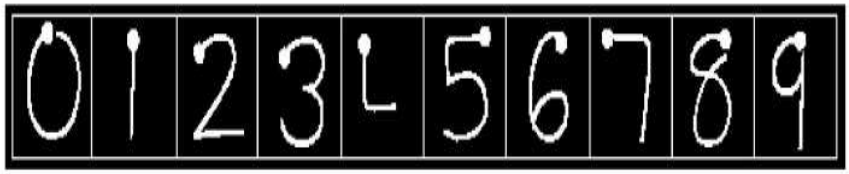


Figure 4.9: Palm's Graffiti digits [10]. The dot point indicates the starting position.

(a) (b) (c)

Figure 4.10: Different scenes of Palm's Graffiti digits dataset [10]: (a) Green glove in GreenDigits, (b) Bare hand in EasyDigits, (c) With moving persons in background.

(ii) **In-house dataset**: One limited in-house dataset [72] has been created with the help of three subjects. Here also the gestures are ten Hindu-Western Arabic numerals, 0-9, but in an isolated mode (320×240, 30 frames/s). Dataset consists of 90 videos for 10 classes (three subjects × ten digits × three examples/digit/subject). For simplicity, a very simple black background is used with full-sleeve attire worn by the subjects and they have used open palm while performing the gestures.

4.4.2 Data Augmentation

Data augmentation plays a vital role in the deep learning approach due to the huge amount of data required in these techniques. It generates more data from a small database using some simple methods like affine transformations. With this, we can increase the diversity of data available for the training model, without actually collecting new data. The generation of new data provides robustness as well as scale, translation and rotation invariance to the system. In this work, data augmentation methods are used on the GreenDigits dataset. Initially, 300 videos from the GreenDigits dataset are preprocessed into 300 OFMT images. These images are then increased to 1500 images after data augmentation. For data augmentation, several transforms are used like rotation up to 20 degrees, width shift (up to 0.2 range), height shift (up to 0.2 range), sheer (up to 0.2 range), zoom mode (up to 0.2), fill mode on nearest data *etc.* Data augmentation techniques like horizontal and vertical flipping are not used on the images as it may lead to confusion between a few pairs of digits like (2, 5), (4, 7), (6, 9) *etc.*

EasyDigits and HardDigits sets are used for the testing phase on which data augmentation is not implemented.

4.4.3 Experimental Set-up

This section gives an idea of the experimental set-up, work performed and the analysis done on the databases to obtain the results. Also, it throws light on the importance of data augmentation process for small databases. The neural network experimentation part is done taking the help of the Google Colab GPU. Other parts of the experiments are performed in a workstation with Intel® Core™i5-4570 CPU with 3.2 GHz and 8GB RAM.

For training the 3D-CNN model, the segmented video clips of isolated gestures from GreenDigits sets are used from the Graffiti dataset. Here stochastic gradient descent (SGD) algorithm is used with the cross-entropy loss function given by Eq. (4.16).

$$Loss(y, \hat{y}) = -\sum_{j=1}^{M}\sum_{i=1}^{N} y_{ij} log \hat{y}_{ij} \tag{4.16}$$

where M is the number of samples, N is the number of classes and $\hat{y}$ is the predicted value for a true value y. The batch size is set to 10 videos and the model is trained with 100 epochs on the training dataset. The choice of learning rate is 0.01 if the epoch count is less than 25 and is reduced to 0.001 for epoch count up to 50. To further promote slow learning, values of $1e^{-4}$ and $1e^{-5}$ are used, till 75 and 100 epochs respectively. Fig. (4.11) and Fig. (4.12) gives the training-testing loss and accuracy curves for the Graffiti dataset. From the training loss and accuracy curves, it can be concluded that the system is not suffering from over-fitting after some tuning of the hyper-parameters. Since the gestures of our limited in-house dataset are the same as the training dataset, hence our in-house dataset is used only for testing purpose which reduces the burden of training requirements again and again.

With regard to the training of the 2D-CNN model, the GreenDigits set from Graffiti Digits is used. SGD algorithm is carried out with a cross-entropy loss function in the training process. The initial learning rate is set to 0.01 if the epoch count is less than 25 and is reduced to 0.001 for epoch count up to 50 with batch size as 32. The training process is stopped after 50 epochs. Fig. (4.13) and Fig. (4.14) gives the training-testing loss and accuracy curves for the Graffiti dataset for the 2D-CNN model.

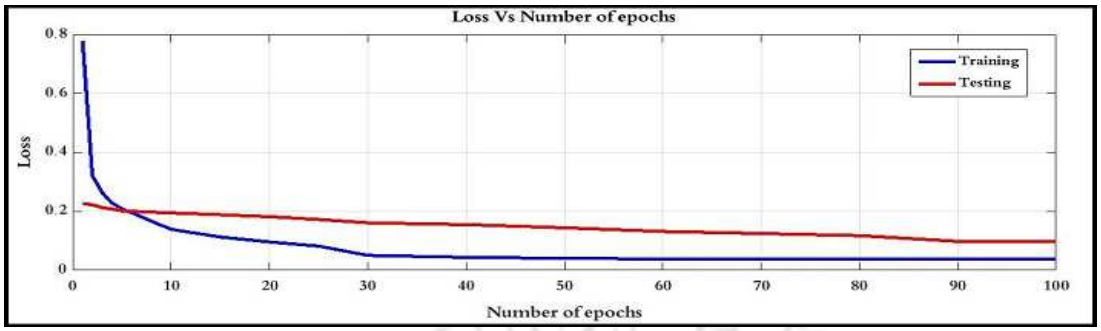


Figure 4.11: Training and testing loss as a function of the number of epochs for 3D-CNN.

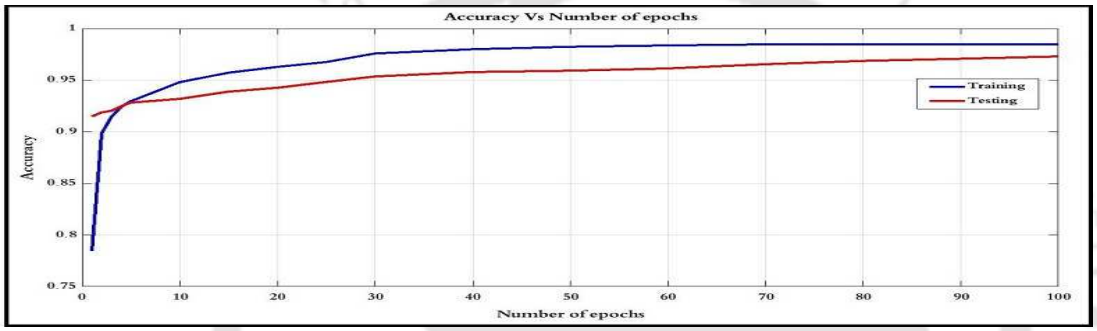


Figure 4.12: Training and testing accuracy as a function of the number of epochs for 3D-CNN.

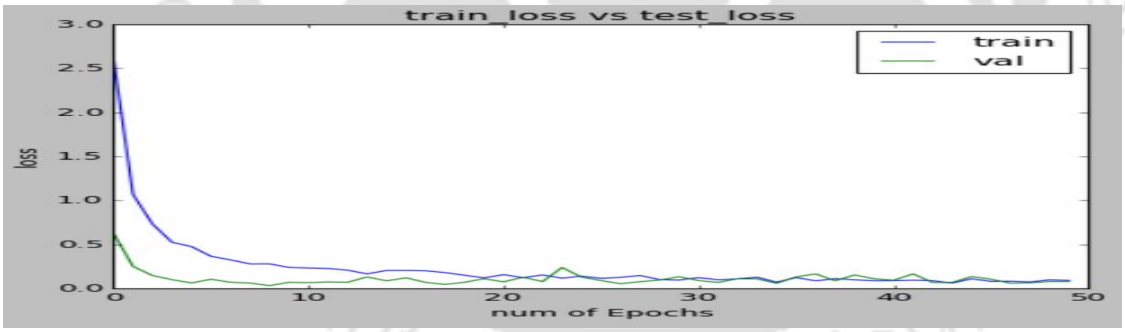


Figure 4.13: Training and testing loss as a function of the number of epochs for 2D-CNN.

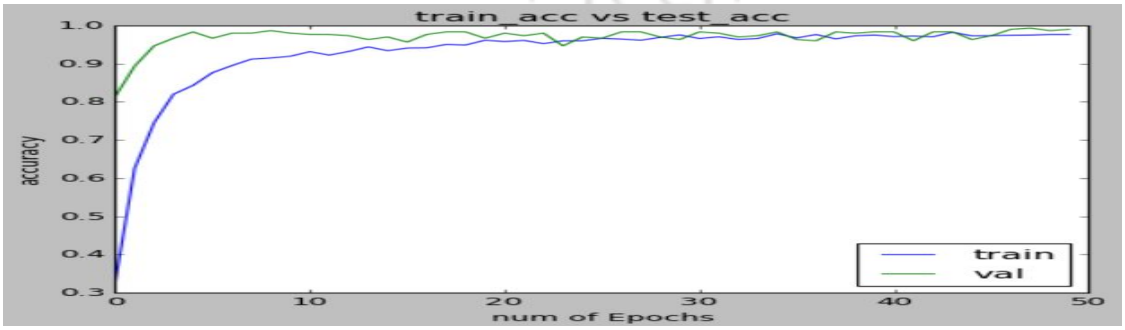


Figure 4.14: Training and testing accuracy as a function of the number of epochs for 2D-CNN.

4.4.4 Results

In this subsection, the performance of the proposed two-stream network is evaluated in three aspects on Graffiti as well as in-house datasets: 2D-CNN model alone with OFMT motion templates as input, 3D-CNN model alone with RGB gesture videos as input, and the combined fusion model. Basically here accuracy, which indicates the proportion of correctly classified samples with respect to the total number of samples, is used as the evaluation index. Since data imbalance is not there in our databases i.e. each class is constituted by an equal number of samples so other performance matrices like precision, sensitivity, or f-score are not considered here.

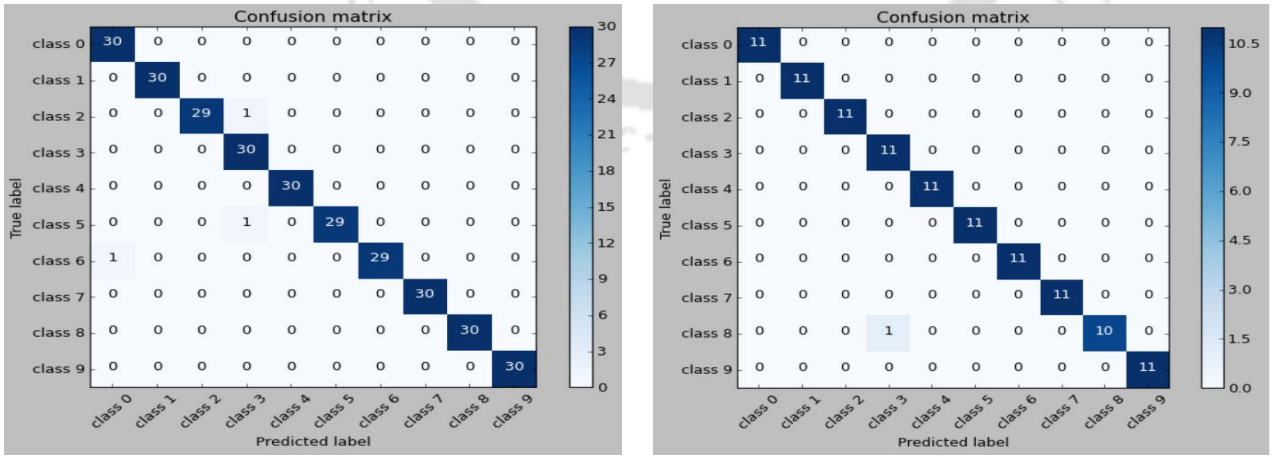
In the proposed model, used 3D-CNN and the 2D motion template CNN architecture can be regarded as two heterogeneous networks that can be used as independent models. So, first, we have evaluated the performance of the independent streams and then the fusion performance as a combined network. In the case of 2D-CNN, confusion occurs mainly in discriminating class '3' with class '5' and class '1' with class '7' due to the similarities in their shapes in the OFMT images which can be seen in Fig. (4.8). Here 3.4% of '5' are misclassified as '3' or vice-versa and 2.8% of '7' are misclassified as '1' or vice-versa and rest 2.2% are various misclassification. Whereas, 3D-CNN has performed quite better in this regard since it also considers the temporal evaluation of the gestures in the video clips. Table (4.2) gives the results of the proposed 2D motion template CNN model, where two cases, without data augmentation and with data augmentation are considered. From Table (4.2), we can conclude that the data augmentation has a great impact on accuracy and can improve the network performance up to a great extent.

Table 4.2: Performance accuracy (%) of 2D-CNN motion template network alone

Dataset	Proposed method (without data augmentation)	Proposed method (With data augmentation)
Graffiti	86.24%	**92.60%**
In-house	81.20%	**89.70%**

To analyze the performance of the 3D-CNN model, the entire EasyDigits set and 110 compatible videos out of 140 videos from the HardDigits set are considered. For the testing phase, stratified 10-fold cross-validation is carried out. The confusion matrices for EasyDigits and HardDigits sets for one fold (i.e. 30 videos from EasyDigits and 11 videos from HardDigits set)

are shown in Fig. (4.15). The mean accuracy for this dataset is obtained at 97.30%. Whereas our in-house dataset has provided an accuracy of 98.67% when tested on the pre-trained network. Lastly, the prediction class/label scores from both the streams are fused for each class. Since both the streams acquire complementary motion information regarding the gesture, so such fusion generally boosts the recognition performance. That is, the two streams complement each-other in acquiring the spatio-temporal information from their respective inputs, and thus certain good output score w.r.t. the target can be achieved, at least by one or the other or by both. In our case also, the same scenario is noticed when both streams are fused at the decision level. Here decision level fusion is chosen since we have used two non-identical networks applied on inputs with different dimensions. Moreover, rather than simple averaging of the prediction class scores, we have formulated a probabilistic ensemble formula given by Eq. (4.14) with γ as fusion parameter. Different values of γ has been tried out and γ as 0.6 has provided us the best fusion accuracy of 99.20%. So, here more weight is given to the score provided by the 3D-CNN since it can capture more subtle spatio-temporal features compared to the other one. This is quite justified from the performance achieved by the individual networks performed on the two databases. Our in-house dataset has achieved a recognition rate of 99% when tested on the fused model. The confusion matrix for EasyDigits and HardDigits sets are also shown in Fig. (4.15).



(a) **(b)**

Figure 4.15: Confusion matrix for: (a) EasyDigits set, (b) HardDigits set.

Table 4.3: Comparison with other methods for pre-segmented Graffiti database

Paper	Feature-type	Features	Classifier	Accuracy
[208]	Hand-crafted	Longest common subsequence (LCS)	HMM, CRF, Most probable LCS (MPLCS)	89.50%, 96.40%, 98.30%
[209]	Hand-crafted	Trajectory matching	Max cosine similarity, fastNN	97.60%
[179]	Both hand-crafted and deep features	CRF-based temporal features	CNN and CRF combined	98.40%
Our method	Deep features	Deep network, Motion template	only 2D-CNN, only 3D-CNN, Late fusion	92.60%, 97.30%, **99.20%**

4.4.5 Comparison with state-of-the-art methods

Our proposed model is compared with three existing methods performed on the same Graffiti dataset. Table (4.3) represents a comparison of performance for the different methods. The first two methods [208, 209] rely on hand-crafted feature representations for gesture classification, while [179] uses CNN to extract gesture features. In [208], the most probable longest common subsequence (MPLCS) is proposed to measure the similarity between the probabilistic template and hand gesture sample. The final decision is based on the probability and length of the extracted subsequences. The method is also compared with HMM and CRF classifiers for performance analysis. Whereas maximum cosine similarity and fastNN is used as a trajectory mapping scheme for digit hand gesture recognition in [209]. A combined fusion-based method with CNN as trajectory shape recognition and CRF as temporal feature recognition is proposed by [179]. From Table (4.3), it can be noticed that our 2D-CNN model is as efficient as the classic HMM model, whereas 3D-CNN has achieved even better results. In [179], fusion-based model with CNN and CRF as components has achieved 98.4% accuracy which is similar to the sequential state-space MPLCS [208] method. On the other hand, our 2D-CNN and 3D-CNN

based fusion model has achieved state-of-the-art results with 99.20% of accuracy. The fusion result at the decision level has outperformed all other methods, which shows the effectiveness of the fusion scheme. The only work done on our in-house dataset is [72], which has a similar accuracy of 99% as this work for alphabet gesture recognition.

4.5 Summary

In this work, we propose a fusion-based two-stream network with 3D-CNN and 2D-CNN as its two streams/layers for hand gesture and in general action recognition. So, the main objective of the model is to detect and recognize isolated dynamic hand gestures with varying shapes, sizes, and colors of the hand. This is possible because of the fact that the system doesn't require the pre-segmentation of the hand portion through various methods like skin-segmentation *etc.* The first stream of the system is a 3D-CNN applied for capturing the spatio-temporal information directly from the RGB gesture videos. The second layer is a 2D-CNN model employed to extract motion-patterns for gesture classification. For this stream, an optical flow-guided motion template (OFMT) is used as input where the temporal motion information of a gesture is encoded into a single image which helps to remove irrelevant gesture patterns. Moreover, the proposed OFMT can nominally reduce computational complexity and memory requirement as compared to more complex networks like double 3D-CNN/RNN/LSTM models. So, our proposed model can be used in a resource constraint environment without affecting much to its performance. For improving results, the prediction scores of the 3D-CNN model and the 2D-CNN model are fused. Since both the streams acquire complementary motion information regarding the gesture, so such fusion generally boost the recognition performance. The main contributions of our proposed model are as follows:

(i) Ground truth flow is required in supervised training for optical flow estimation. But, generally, the ground truth flow is not available except for limited synthetic data [171]. Moreover, computation of optical flow and then learning the mapping from optical flow to action labels is time-consuming as well as storage demanding. So, we have proposed optical flow-guided motion template (OFMT) images as input to the 2D-CNN stream which provides additional temporal information in a resource constraint environment.

(ii) Our method is efficient in terms of computation and storage point of view as we do not need to store the precomputed optical flow. Moreover, the requirement of segmenting the hand portion from the body is not needed and also, the size, shape, and color of the hand have no effects on the OFMT images.

(iii) A late-fusion scheme is proposed to leverage the information containing in both RGB gesture videos and motion template modalities. The advantage of the proposed method is that different deep models can provide complementary motion information. The first layer can capture the spatio-temporal information thorough the 3D deep network, while the motion-patterns are obtained using 2D-CNN through OFMT images.

Though our model is simple, experimental results have demonstrated that it is able to achieve state-of-the-art results. However, the adopted motion template has the limitation that the moving body has to be in a plane perpendicular to the camera. One important research direction would be to investigate more on robust feature learning methods for distinguishing the subtle differences among gestures for the viewpoint-invariant mechanism. In this work, an optical flow-guided motion template (OFMT) has been used as a substitute for the segmentation process in trajectory-based gestures recognition. But in some cases segmentation becomes an unavoidable process, and then researchers generally opt for other types of segmentation techniques. Moreover, segmentation approaches are mostly viewpoint-invariant. So, in our next work, we would go for some segmentation method for detecting the hand across each frame in the hand gesture recognition process. Another target is to develop a gesture recognition scheme applicable to both static as well as dynamic hand gestures.

5

Attention-based Hand Semantic Segmentation and Gesture Recognition using Deep Networks

The ability to discern the shape of hands can be a vital issue in improving the performance of hand gesture recognition. Segmentation itself is a very challenging problem having various constraints like illumination variation, complex background etc. The objective of the work is to incorporate the perception of semantic segmentation into a classification problem and make use of the deep neural models to achieve improved results for both static and dynamic gestures. This work utilizes the UNet architecture with attention-module to obtain the semantically segmented masks of the input images, which are then fed to a classifier for recognition. The concept of attention-mechanism adds to the improvement of segmentation accuracy. In this work, for static gestures, the top classifier layer of the VGG16 model is replaced with a classifier designed specifically for classifying the gestures at hand. For dynamic gestures, 3D-CNN (C3D) architecture is used as a classifier that can capture spatial as well as temporal information of a gesture video. The data augmentation process is used in preprocessing to generate a sufficient number of training images for the aforementioned CNN-based models. Significant and improved recognition has been achieved for both static and dynamic hand gesture databases through the inherent feature learning capability of CNN and refined segmentation.

5.1 Introduction

Accurate segmentation of the hand or the gesturing body part from the captured videos or images still remains a challenge in computer vision for many constraints like illumination variations, background complexity, occlusion and so on [51, 210]. Illumination variations affect the accuracy of skin color segmentation methods. Poor illumination may change the chrominance properties of the skin colors, and the skin color will appear different from the original color. A major challenge in gesture recognition is the proper segmentation of skin-colored objects (*e.g.*, hands, face) against a complex static/dynamic background. The accuracy of skin segmentation algorithms is limited because of objects in the background that are similar in color to human skin. Skin-colored objects present in the background also increase false positives. All these factors make the detection of the hand to be one of the vital stages in the gesture recognition system. In our previous work, an optical flow-guided motion template (OFMT) has been used as a substitute for the segmentation process in trajectory-based gesture recognition. But in some cases segmentation becomes an unavoidable process, then researchers generally go for other types of segmentation techniques for detecting the hand in the gesture recognition process.

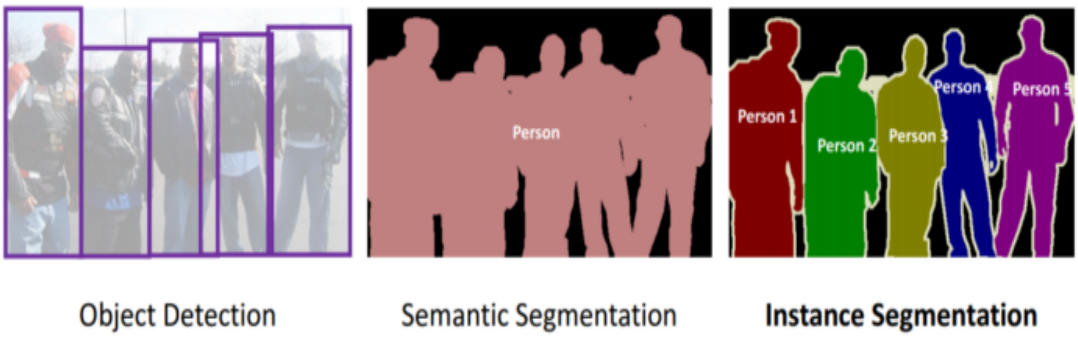


Figure 5.1: Different techniques for the segmentation process.

Due to the above-mentioned constraints, researchers generally opt for different segmentation techniques like object detection by bounding box, semantic segmentation, or instance segmentation as shown in Fig. 5.1. In object detection through a bounding box, people try to locate and classify multiple objects within an image/video, by drawing bounding boxes around them and then classifying what's in the box. One major disadvantage here is that we only get a bounding box covering the object, but we really don't get an idea regarding the shape of the

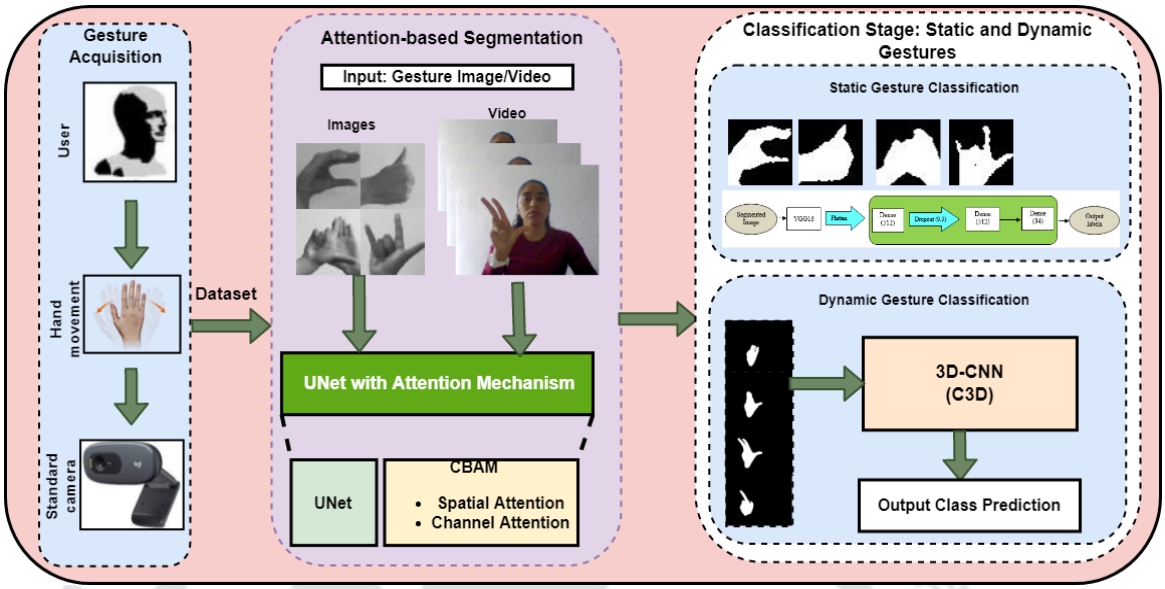


Figure 5.2: Block diagram of our proposed hand gesture recognition framework.

object. Semantic segmentation is more informative where it classifies each and every pixel in the image by assigning an appropriate class label to the pixels and linking similar pixels into a group (class). Instance segmentation is a challenging task that requires the prediction of object instances and their per-pixel segmentation task. This makes it a hybrid of semantic segmentation and object detection. There is no hard and fast rule regarding their adoption and it all depends on the application where one single scheme can be applied using various approaches.

With the advancement in neural networks and computing devices, tasks like image classification, object recognition, and segmentation have been carried out with improved results and much efficiency. Convolutional Neural Networks (CNN) form the backbone of most of the modern-day deep learning models, which have achieved ground-breaking outcomes in regards to the above tasks. It has helped to achieve classification results close to the human level. Also, it is capable of localizing objects by assigning appropriate class labels to the pixels, which is the governing principle of semantic segmentation [211].

Attention mechanism, which plays an important role in human perception, can effectively highlight useful information while suppressing the redundant one. Recently, attention mechanism has been receiving wide attention in a variety of tasks, such as natural language processing for machine translation, natural image classification, salient object detection, natural image segmentation, medical image segmentation and classification in medical image analysis

fields, image captioning etc. There are many attempts that have embedded attention modules into deep neural network architecture for improving the performance of image segmentation, classification, and object detection in computer vision fields.

Meanwhile, UNet [212] has achieved great success in the field of medical image segmentation, and it is also the mainstream of current segmentation methods. The network has a progressively narrowing structure, which tends to encode the input into a fine to the coarse manner, followed by a decoding structure that broadens progressively. However, during the process of downsampling, UNet constantly reduces the dimension of the image, which results in poor segmentation accuracy for the small-scale objects. Considering that attention mechanisms can enhance local feature expression, to solve the insufficient segmentation accuracy, researchers generally adopt attention mechanisms in the various segmentation processes. As a computing resource allocation scheme, the attention mechanism uses limited resource allocation to process more important information to solve the problem of information overload. Generally, the input of a neural network often contains a lot of redundant information and all the information is not needed to be focused on. So, one can pay attention only to something important to improve time and space utilization. Researchers have demonstrated that introducing an attention mechanism into UNet can enhance local feature expression and improve the performance of image segmentation.

It is already mentioned that hand segmentation is a challenging task due to constraints like illumination variations, background complexity, occlusion and so on. Background noise and varying lighting conditions also cause occlusions and clutter, which have to be considered. However, the most accurate approaches that try to mitigate such constraints tend to employ multiple modalities derived from input frames, such as optical flow or depth information [204, 213]. This practice limits real-time performance due to intense extra computational cost. In this work, we avoid depth information or optical flow computation by proposing a hand gesture recognition method based on RGB frames combined with hand segmentation masks. [214] found that the semantic segmentation is more than two times faster than the optical flow, making the semantic segmentation an alternative feasible option for real-time applications as well. Contextual information extraction is difficult with hand-crafted features in conventional machine learning techniques. Attention-based methods have been proved to be effective ways to obtain important contextual information in different segmentation meth-

ods like semantic segmentation. Therefore, in this work, we have proposed a deep learning attention-based segmentation method as a solution to the above-mentioned issues. Here, we aim to explore the effectiveness of a recent attention module called Convolutional Block Attention Module (CBAM) [215] combined with UNet architecture for hand segmentation purposes. The rule-based algorithms of attention-based semantic segmentation for static gesture interpretation have successfully been transferred into dynamic gesture recognition where C3D is used to automatically extract the robust temporal and spatial features to recognize the hand gestures.

5.2 Background and Related Work

5.2.1 Semantic Segmentation

There are several model variants based on Convolutional Neural Networks (FCNs) to enhance contextual aggregation in segmentation. Faster R-CNN [216], R-FCN [217] are used to exploit the region of each instance, and then predict the mask for each region. He et al. [218] proposed Mask R-CNN that is built on the top of Faster R-CNN by adding an instance-level semantic segmentation branch. On the other hand, semantic segmentation, using CNN-based methods, was pioneered by Long *et al.* [219] using Fully Convolutional Network (FCN). This work primarily defined a skip network that combined the information from the coarse upper layer of the deep neural architecture with the lower fine layer, which in turn helped achieve meticulous segmentation results. In [212], Ronneberger *et al.* defined an architecture that apprehended contextual information through a gradually contracting path and localized the concerned objects via a symmetric expanding path. Due to its structure, it was called UNet and it is a classic work for medical image segmentation. The authors trained the network on a few biomedical images with the application of the data augmentation process and achieved state-of-the-art results. UNet++ [220] re-design skip pathways that connect the encoder and decoder networks and adopt deep supervision on the basis of UNet to further improve the segmentation accuracy of the model. Huang et al. [221] proposed a novel model UNet3+ that reconstructs the connections between the encoder and the decoder and internally. The role of the connections between decoders is to capture fine-grained details and coarse-grained semantics from the entire scale.

Apart from UNet, the Deeplab series is also one of the most popular CNN architectures in the field of semantic segmentation. Since 2014, v1 [222], v2 [223], v3 [224] and v3+ [225] series have been successively proposed. Deeplabv2 [223] and Deeplabv3 [224] adopt atrous spatial pyramid pooling (ASPP) to embed contextual information, which consists of parallel dilated convolutions with different dilated rates. Deeplabv3+ [225] is currently the latest neural network structure of the Deeplab series, which is mainly improved based on Deeplabv3. This network mainly borrows the traditional encoder-decoder architecture, expands a simple and effective module for recovering boundary information.

Semantic segmentation requires a significant number of annotated data at the pixel level, and this drawback is addressed in [226]. Souly *et al.* [226] proposed a semi-supervised method of semantic segmentation, based on Generative Adversarial Network (GAN) [227]. Zhang *et al.* [228] proposed a novel model named SegGAN, formed by fitting a pre-trained deep semantic segmentation model into a GAN. This composite network learned features, which reduced the loss between the original images and the generated ones, and eventually arrived at better segmentation masks. In order to make reasonable use of limited visual information processing resources, attention can be used to explain the alignment relationship between input and output data and explain what the model has learned. The reason why the attention mechanism is so popular is that the attention mechanism gives the network the ability to distinguish and focus.

5.2.2 Attention Mechanism

Attention mechanisms are widely used in computer vision to extract better visual features. Attention not only tells where to focus, but it also improves the representation of interests. Our goal is to increase representation power by using an attention mechanism: focusing on important features and suppressing unnecessary ones.

Considering the **number of positions**, attention mechanisms are usually divided into soft attention and hard attention. The soft attention mechanism is easy to implement and it attends to arbitrary input locations using spatial transformer networks [229]. It produces a distribution over input locations, reweight features and feed as input. Soft attention focuses on the image channels and is a deterministic attention mechanism. Its advantage is that the derivative of the function can be differentiated. Thus, the gradient values can be back-propagated through the

neural network. In contrast, hard attention emphasizes the salient areas of the image and is a random prediction process that focuses primarily on dynamic changes. It can't use gradient descent and need reinforcement learning.

According to the **type of architecture**, attention models can be implemented as encoder-decoder [230, 231], transformer [229] and memory networks [232]. An encoder-decoder-based attention model takes any input representation and reduces it to a single fixed length, a transformer network aims to capture global dependencies between input and output, and in the memory networks, facts that are more relevant to the query are filtered out.

Depending on the **type of focus**, there are two types of attention mechanism: spatial attention [233] and channel attention [231]. The spatial attention mechanism makes the network pay more attention to the spatial position of the target, and the channel attention mechanism tends to focus on the size of the target [234].

With respect to **number of sequences**, attention can be of three types, namely distinctive, co-attention and self-attention. While in distinctive attention candidate and query states belong to two distinct input and output sequences, in self-attention [235] the candidate and query states belong to the same sequence. The self-attention mechanism just concerns single rather than multiple cross-modal semantic information, that is, query, key, and value are all obtained from the same semantic information in contrast to spatial transformer networks. Co-attention accepts multiple input sequences as input at the same time and jointly produces an output sequence.

There have been several attempts like [230, 231, 233] to incorporate attention processing to improve the performance of CNNs in large-scale classification tasks. Wang et al. [233] proposed Residual Attention Network which uses an encoder-decoder style attention module. With the refining of the feature maps, the network not only performs well but is also robust to noisy inputs. Instead of directly computing the 3D attention map, channel attention and spatial attention can be learned separately. Hu et al. [231] introduced a compact module to exploit the inter-channel relationship in their Squeeze-and-Excitation (SE) module. They used global average-pooled features to compute channel-wise attention. However, these are suboptimal features in order to infer fine channel attention. They also missed the spatial attention, which plays an important role in deciding 'where' to focus as shown in [236]. Li et al. [237] and Yu et al. [238] feed the features of deep layers with stronger semantics into SE-like attention

block to provide high-level category information, which helped to precisely recover details in the upsampling stage of image segmentation.

5.2.3 Attention-based Methods for Hand Gesture Recognition

The first noted visual attention-based work to recognize hand postures in the complex background was given by [16]. The proposed method was simple without any deep architecture and utilized a Bayesian model of visual attention generating a saliency map to detect and identify the hand region. Feature-based visual attention was implemented using a combination of high-level (shape, texture) and low-level (color) image features. Using deep networks, a multi-channel method was proposed by [239] with spatial attention focused on the hands, and different channels were fused using a sparse network. [240] extended MaskRCNN with a novel attention mechanism to incorporate contextual cues that captures non-local dependencies between features. [241] proposed a stacked 3D attention-based residual network (Res3ATN) with convolution, residual and attention blocks in a sandwich manner layer after layer. The multiple attention blocks can generate different features at each attention block. [242] used a transformer-based neural network for dynamic hand gesture recognition. [243] proposed an attention-model based on Inception CNN for extracting spatial features and Bi-LSTM (long short-term memory) for temporal feature extraction in Arabic sign language classification. [244] applied transformer-based self-attention mechanism to collect features from cropped input frames and combined through mutual-attention feature fusion to produce a joint RGB-D representation.

Most of these attention mechanisms, however, do not consider spatial locality. But locality is essential for hand detection in a scene. Furthermore, most of them are defined based on similarity instead of semantics, ignoring the contextual cues obtained by reasoning about the spatial relationships between semantically related entities. So, here, we have designed a method for hand segmentation in images as well as videos using attention-based semantic segmentation and subsequent recognition through deep networks.

5.3 Methodology

This section describes the workflow of the proposed method. The proposal has two sections: attention-based semantic segmentation and classification. Here, we have semantically

segmented the static (still images) or dynamic (video frames or image sequences) gestures, and segmented masks are subsequently fed to a classifier for recognition. The following subsections shed light on the models as well as the work process for semantic segmentation and classification independently.

5.3.1 Semantic Segmentation

As already mentioned, the objective of semantic segmentation is to assign labels to each pixel and then link the similar pixels into a group. Here we employed UNet architecture for obtaining a segmented mask image, where the hand portion is segmented from the background.

5.3.1.1 UNet structure

The network has a progressively narrowing structure, which tends to encode the input into a fine to the coarse manner, followed by a decoding structure that broadens progressively. The decoder adds skip connections. This makes the segmented result more accurate block after block, as the finer details from the earlier layers of the encoding structure coalesce with the layers of the decoding path (up6 with dropout4, up7 with conv3, up8 with conv2 and up9 with conv1).

The architecture consists of convolutional blocks each on the encoding path as well as on the decoding path. Each block of the encoding path contains two convolutional layers with a receptive field of size (3×3) followed by a max-pooling layer with a (2×2) window. Zero padding is done for the convolutional operation to maintain the same spatial dimensions of the output feature map as the input. The layer succeeding the downsampling (max-pooling) layer contains twice the number of channels in the previous layer. Also, a dropout layer is included in the last and penultimate block of the encoder, which is basically to prevent the development of the co-dependencies amongst neurons.

In the decoder, the max-pooling layer is replaced by the upsampling layer of window size (2×2). After each upsampling layer, there is a convolutional layer to match the dimension of the feature map of the layer on the encoding path, which is concatenated with this layer. This convolutional layer reduces the feature channel to half of the number of channels in the previous layer. This is basically for the skip connection. The activation function for each convolutional

layer is 'ReLU' except the last layer, which is 'Sigmoid'.

5.3.1.2 Re-designed skip path with attention module

In the original UNet, the skip-connected feature maps of the decoder are received straight from the encoder. But, here, we have made some modifications in skip connection by inserting an attention unit called Convolutional Block Attention Module (CBAM) between encoder and decoder. Dataflow passes through the chain of convolutional layers using CBAM in the skip-connections. With the insertion of the attention module, the semantic distance between the encoder and the decoder maps is likely to decrease. The architecture for the modified network is shown in Fig. 5.3.

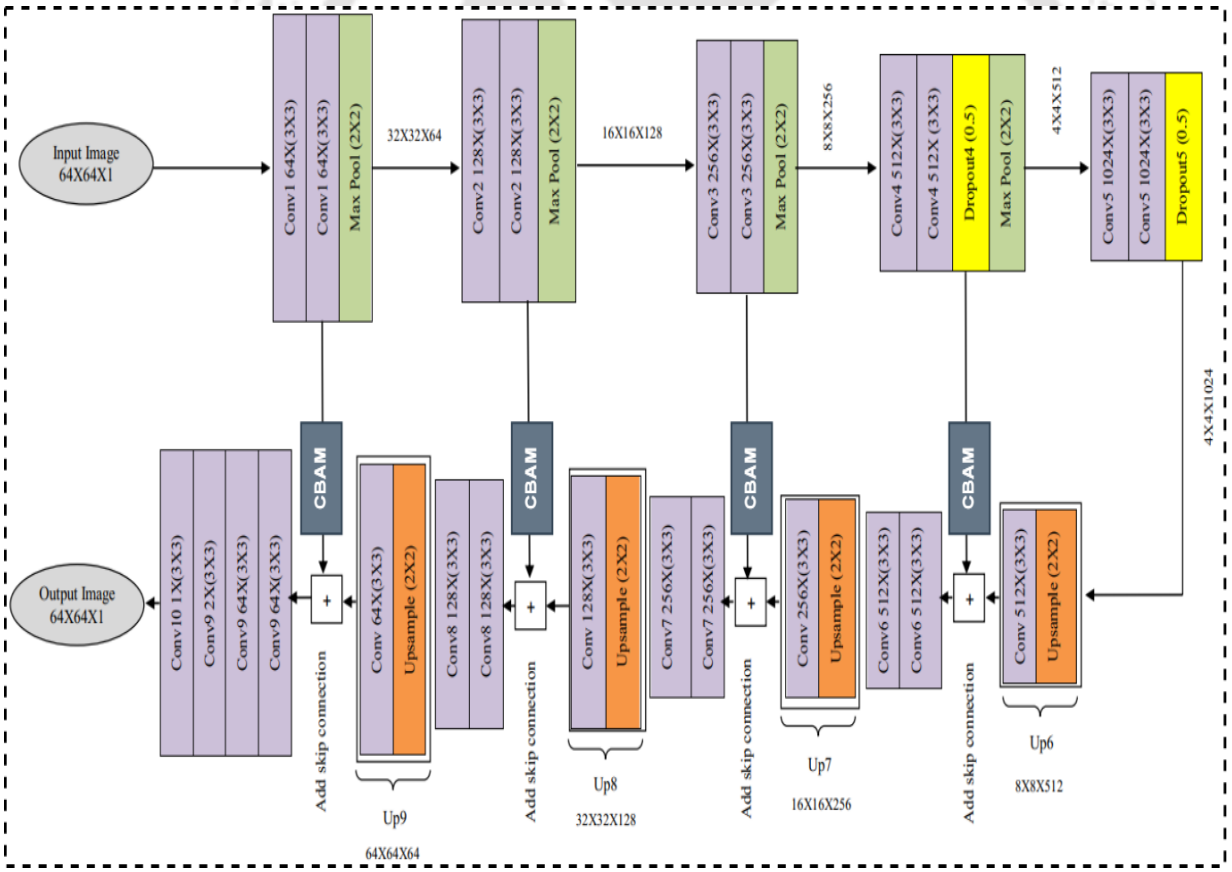


Figure 5.3: UNet architecture used for semantic segmentation with attention mechanism.

5.3.1.3 Convolutional Block Attention Module (CBAM)

Usually, the attention mechanism is placed after the convolutional layer; then, the features to which the attention network pays attention and the features extracted by the neural network are input into the next convolutional layer. So, an attention network can be understood as a weighting operation that operates on different feature regions. Convolutional Block Attention Module (CBAM) [215] emphasize meaningful features along those two principal dimensions: channel and spatial axes. To achieve this, we sequentially apply channel and spatial attention modules, so that each of the branches can learn 'what' and 'where' to attend in the channel and spatial axes respectively. Given an input image, two attention modules, channel and spatial compute complementary attention, focusing on 'what' and 'where' respectively. The channel attention block is proposed to integrate the interaction among the inter-channel feature maps. It is employed to enhance the vital information of a feature map of the object i.e. hand. The spatial attention mechanism focuses on the local regions of a feature map. Thus, this module is employed to preserve the location of the hand information (ROI) in the feature maps. Considering this, two modules can be placed in a parallel or sequential manner. We found in the literature that the sequential arrangement (as shown in Fig. 5.4) gives a better result than a parallel arrangement [215].

Given an intermediate feature map $F \in \mathbb{R}^{C\times H\times W}$ as input, CBAM sequentially infers a 1D channel attention map $M_c \in \mathbb{R}^{C\times 1\times 1}$ and a 2D spatial attention map $M_s \in \mathbb{R}^{1\times H\times W}$ as illustrated in Fig. 5.4. The overall attention process can be summarized as:

$$F' = M_c(F) \otimes F, \tag{5.1}$$

$$F'' = M_s(F') \otimes F' \tag{5.2}$$

where $\otimes$ denotes element-wise multiplication. During multiplication, the attention values are broadcasted (copied) accordingly: channel attention values are broadcasted along the spatial dimension, and vice versa. F'' is the final refined output. The zoomed section of the channel and the spatial portion in Fig. 5.4 depicts the computation process of each attention map. The following describes the details of each attention module.

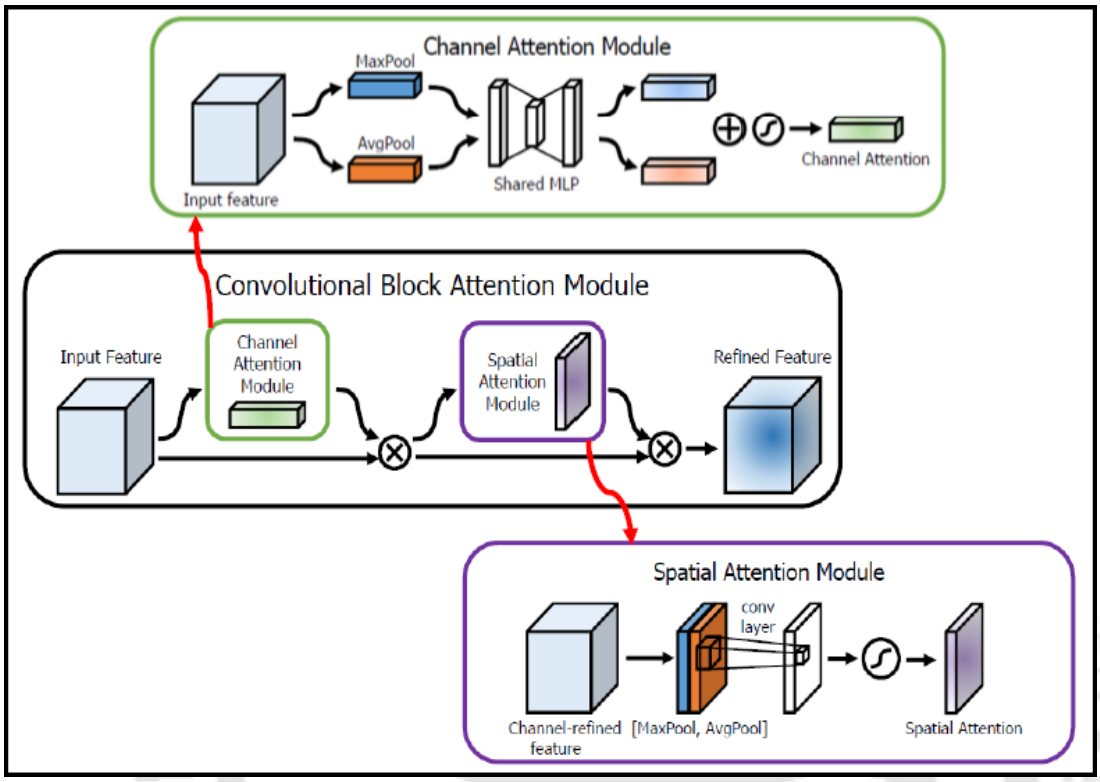


Figure 5.4: CBAM architecture used for attention mechanism.

(i) **Channel attention module:** A channel attention map is exploiting the inter-channel relationship of features. The spatial information of a feature map is aggregated by using both average-pooling and max-pooling operations, generating two different spatial context descriptors: F^c_{avg} and F^c_{max}, which denote average-pooled features and max-pooled features respectively. Both descriptors are then forwarded to a shared network to produce our channel attention map $M_c \in \mathbb{R}^{C\times1\times1}$. The shared network is composed of a multi-layer perceptron (MLP) with one hidden layer. To reduce parameter overhead, the hidden activation size is set to $\mathbb{R}^{C/r\times1\times1}$, where r is the reduction ratio. After the shared network is applied to each descriptor, we merge the output feature vectors using element-wise summation. In short, the channel attention is computed as:

$$M_c(F) = \sigma(MLP(AvgPool(F)) + MLP(MaxPool(F)) \tag{5.3}$$

$$= \sigma(W_1(W_0(F^c_{avg})) + W_1(W_0(F^c_{max}))) \tag{5.4}$$

where σ denotes the sigmoid function, $W_0 \in \mathbb{R}^{C/r \times C}$, and $W_1 \in \mathbb{R}^{C \times C/r}$. Note that the MLP weights, W_0 and W_1, are shared for both inputs and the ReLU activation function is followed by W_0.

(ii) **Spatial attention module:** The spatial attention map is generated by utilizing the inter-spatial relationship of features. On the concatenated feature descriptor, we apply a convolution layer to generate a spatial attention map $M_s(F)^{H \times W}$ which encodes where to emphasize or suppress. For this, we aggregate channel information of a feature map by using two pooling operations, generating two 2D maps: $F^s_{avg} \in \mathbb{R}^{1 \times H \times W}$ and $F^s_{max} \in \mathbb{R}^{1 \times H \times W}$. Each denotes average-pooled features and max-pooled features across the channel. Those are then concatenated and convolved by a standard convolution layer, producing our 2D spatial attention map. In short, the spatial attention is computed as:

$$M_s(F) = \sigma(f^{7\times7}[(AvgPool(F); (MaxPool(F)]) \tag{5.5}$$

$$= \sigma(f^{7\times7}[F^s_{avg}; F^s_{max}]) \tag{5.6}$$

where σ denotes the sigmoid function and $f^{7\times7}$ represents a convolution operation with the filter size of 7×7.

5.3.2 Static Hand Gesture Recognition

For static gestures, above mentioned attention-based UNet architecture is employed to obtain the segmented masks from the still images. Then these segmented images are fed to a neural network for feature extraction and finally extracted features are fed to a classifier for recognizing the gestures. The workflow is shown in Fig. 5.5. Though this model is simple, experimental results have demonstrated that it is able to achieve state-of-the-art (SOTA) results. The following sections also throws light into the dataset used and the importance of data augmentation process.

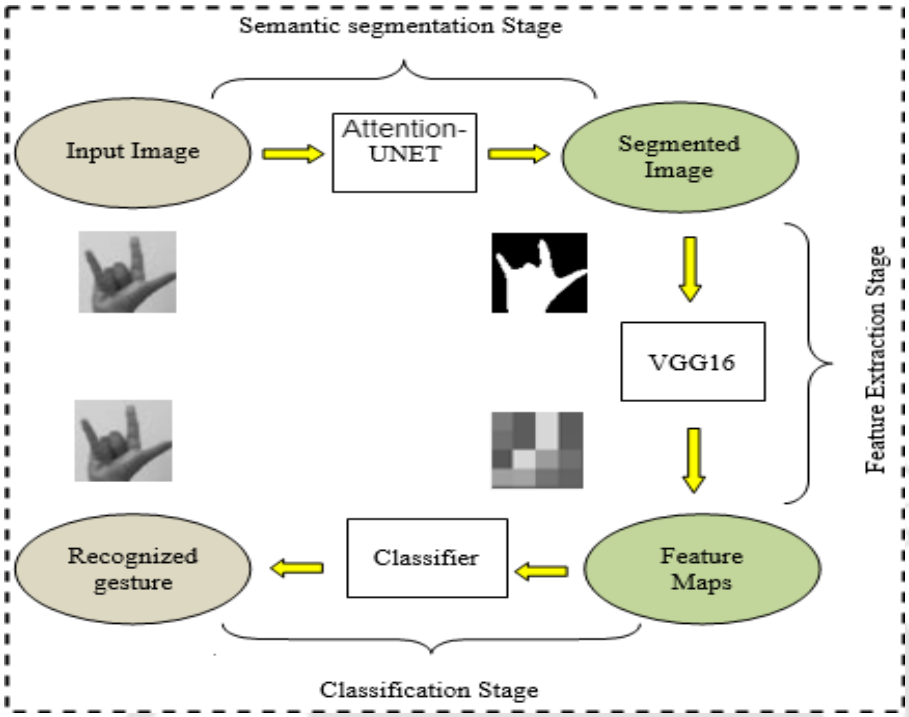


Figure 5.5: Block diagram of the workflow for static hand gesture recognition.

5.3.2.1 Dataset

The dataset used for static hand gesture recognition is the Brazilian Sign Language (Libras) dataset [11]. The official sign language of Brazil is called Libras. It consists of a total of 9600 images, evenly distributed among 40 classes. The different classes represent letters of Libras alphabets (22 classes), numbers (6 classes), and a few words (12 classes). Though there are 40 classes in the original Brazilian Sign Language database, the experiments were carried out for 34 compatible classes. Each class contains the segmented masks depicting the skin region of the gesture. Each image in the dataset has a resolution of $50 \times 50 \times 3$, and the images are captured considering small variations in the illumination as well as the hand posture and size. The background is kept uniform, without any cluttering objects.

5.3.2.2 Data Augmentation

Data augmentation plays a crucial role in deep learning approaches, as the number of data samples required in deep learning techniques is very high. Data augmentation generates more training data out of the few training samples available, generally employing affine transforma-

tion to the samples. Thus, the model is exposed to every possible aspect of the distribution of the data samples and helps it generalize the new data. For data augmentation, several transforms are used like rotation up to 20 degrees, width shift (up to 0.2 range), height shift (up to 0.2 range), sheer (up to 0.2 range), zoom mode (up to 0.2), fill mode on nearest data, etc. This newly generated data also contributes to the required robustness against the variation of scale, translation and rotation. The training sample size for the classifier network is increased to 106800 images after data augmentation.

5.3.2.3 Generation of Segmented Masks

Till now, the architecture of the model has been discussed, and in this section, the process of generation of the segmented masks is highlighted. Speaking of the procedure, the original images are passed through the attention-based UNet architecture, and the output images are obtained through the final convolutional layer of the decoder part of the trained structure. After a sufficient number of images are generated through the data augmentation process, the images are fed to the UNet model for training. For the training process, the input images are arranged into two sections, one containing the original images, and the other containing the segmented masks (included in the dataset in grayscale). During the training process, the architecture learns to focus on the ROI portion and when the test images are fed, it can segment out the hand portion. The segmented image has its advantage, as it has only two regions (*i.e.*, the hand and the background), and it is free from variations in the intensity values within the same region (shown in Fig. 5.6). The shading and the depth information are also not present, which may increase the complexity and in turn, the time to process the images in the next stage i.e. classification.

Since this dataset consists of two regions - the hand (*i.e.*, the foreground) and the background, it is, in fact, a sort of binary segmentation problem, assigning a certain range of intensity values to the foreground and the rest to the background. During the training process, the parameters are optimized using the Adam Optimization method [245], where the learning rate is kept at 0.0001 and the hyperparameters β_1 and β_2 are kept at 0.9 and 0.999 respectively. The network is trained for twenty-five epochs with the loss function being the binary

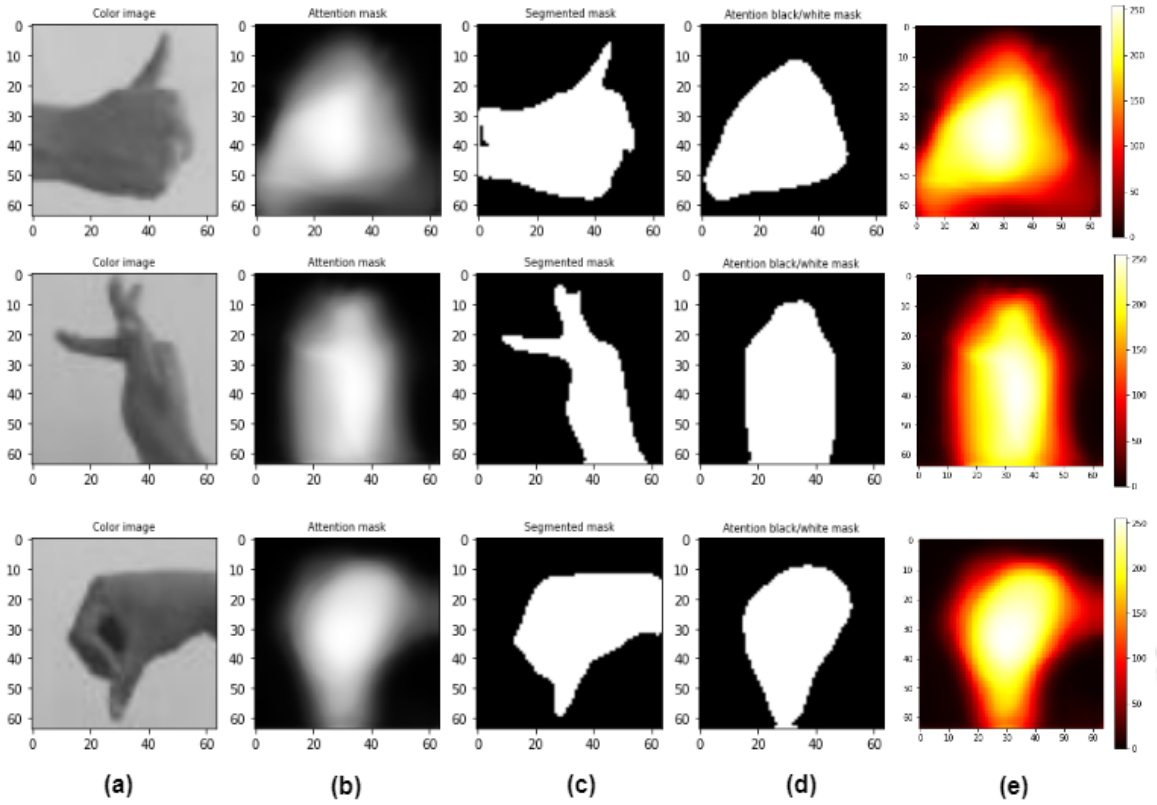


Figure 5.6: Semantic segmentation results showing attention masks for the Brazilian dataset: (a) Shows the gesture images, (b) Shows the attention masks, (c) Shows the segmented masks, (d) Shows the black/white attention masks and (e) Shows the heat-map of the attention masks.

cross-entropy, which is given as:

$$Loss(y, \hat{y}) = -\frac{1}{N}\sum_{i=1}^{N}[ylog\hat{y}_i + (1-y)log(1-\hat{y}_i)]) \tag{5.7}$$

where, N is the number of samples and $\hat{y}$ is the predicted value.

5.3.2.4 Kernel Initialization

For kernel initialization, he-normal initialization [246] is adopted. The reason for using he-normal initialization is that, in this method, the weights are initialized, keeping in mind the size of the previous layer, which helps in attaining the global minimum of the cost function faster and efficiently. Also, the weights differ in range depending on the size of the previous layer, which provides a controlled initialization. Above all, the initialization takes into account the non-linearity induced by the ReLU activation function. In he-normal initialization, the

primary interest is on the variance of the response of each layer. For a convolutional layer, the output is given as: $Y = WX + b$, where W is the weight matrix of the filter, and b is the bias vector. X is the input vector with n number of neural connections.

Let the elements in X and W are mutually independent and have identical distribution. Then,

$$\begin{aligned} var[Y] &= var[WX] + 0 \\ &= E[W^2X^2] - (E[WX])^2 \end{aligned}$$

We know $cov[X, Y] = E[XY] - E[X]E[Y]$ and $var[X] = E[X^2] - (E[X])^2$

$$\begin{aligned} \therefore var[Y] &= (cov[W^2, X^2] + E[W^2]E[X^2]) - (cov[W, X] \\ &\quad + E[W]E[X])^2 \\ &= 0 + (var[W] + E[W]^2)(var[X] + E[X]^2) - \\ &\quad (0 + E[W]E[X])^2 \\ &= var[W]var[X] + E[W]^2var[X] + E[X]^2var[W] \end{aligned}$$

Let y_i, w_i and x_i are the independent random variables of Y, W and X respectively, and also, let w_i have zero mean. i represents the i^{th} convolutional layer. Then

$$var[y_i] = n_i E[x_i]^2 var[w_i] \tag{5.8}$$

Now if w_{i-1} has a symmetric distribution around zero, then y_{i-1} has zero mean and a symmetric distribution. This implies $E[x_i]^2 = \frac{1}{2}var[y_{i-1}]$

$$\therefore var[y_i] = \frac{1}{2} n_i var[y_{i-1}] var[w_i] \tag{5.9}$$

For all the N layers,

$$var[y_N] = var[y_1] \prod_{i=2}^{N} \frac{1}{2} n_i var[w_i] \tag{5.10}$$

This product is the principal factor in initialization, which should not reduce or increase the amplitude of a signal exponentially. Thus, $\frac{1}{2}n_i var[w_i] = 1$.

Hence, we can conclude that he normal initialization draws samples from a truncated Gaussian distribution centred at zero with variance $\frac{2}{\text{number of input units in the weight tensor}}$.

5.3.2.5 Classification

The classification problem in this work is a multi-class problem with a limited amount of data. Hence, we opted for a pre-trained network [247], *i.e.*, VGG16 [248], trained on ImageNet dataset. The final images obtained after segmentation have two regions – the hand and the background. Since the network is trained on RGB images, so these segmented images are converted into 3-channel images using the pseudo coloring method. It is a minor pre-processing step before feeding them into the classification stage, which would recognize the different gestures. The objective of using the pre-trained network is to exploit the spatial hierarchy of features learned by the network, which can be considered as generic and reusable representation of data. Then the classifier on top of the VGG16 model is replaced by our classifier to learn the specific features of the classes of used database. This classifier consists of a dense layer containing 512 neurons and ReLU as the activation function, which is followed by a dropout layer to fight the situation of overfitting. The final layer of the classifier is a dense layer, with 34 neurons giving us the respective class labels with softmax function being used for activation. The block diagram of the classifier is shown in Fig. (5.7). The softmax activation function returns the probability score of the different classes where the largest value gives the class predicted. It is defined as:

$f: \mathbb{R}^N \rightarrow \mathbb{R}^N$, such that $f(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{N} e^{x_j}}$, for $i = 1, ..., N$ and $\forall x_i \in \mathbb{R}^N$

Similar to the semantic segmentation step, the weights are optimized similarly, but the performance measure is being changed from binary to categorical cross-entropy, and it is given as:

$$Loss(y, \hat{y}) = -\sum_{j=1}^{M}\sum_{i=1}^{N} y_{ij} log\hat{y_{ij}} \tag{5.11}$$

where, N is the number of samples, M is the number of classes and $\hat{y}$ is the predicted value.

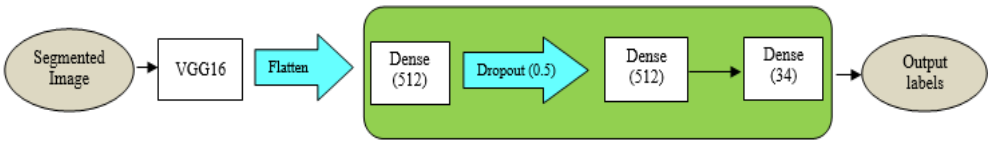


Figure 5.7: Block diagram for the classification process for static gestures.

5.3.3 Dynamic Hand Gesture Recognition

This attention-based model is modified and extended for dynamic hand gesture recognition as well. A dynamic hand gesture video can be considered as a sequence of several static hand gestures. This sequence contains enough information to be used for dynamic hand gesture recognition. So, in this section, we first get the segmented masks for the sequential images after some pre-processing steps. We have chosen a 3D-CNN (C3D) network as a classifier that can capture spatial as well as temporal information of a video. The workflow is shown in Fig. 5.8.

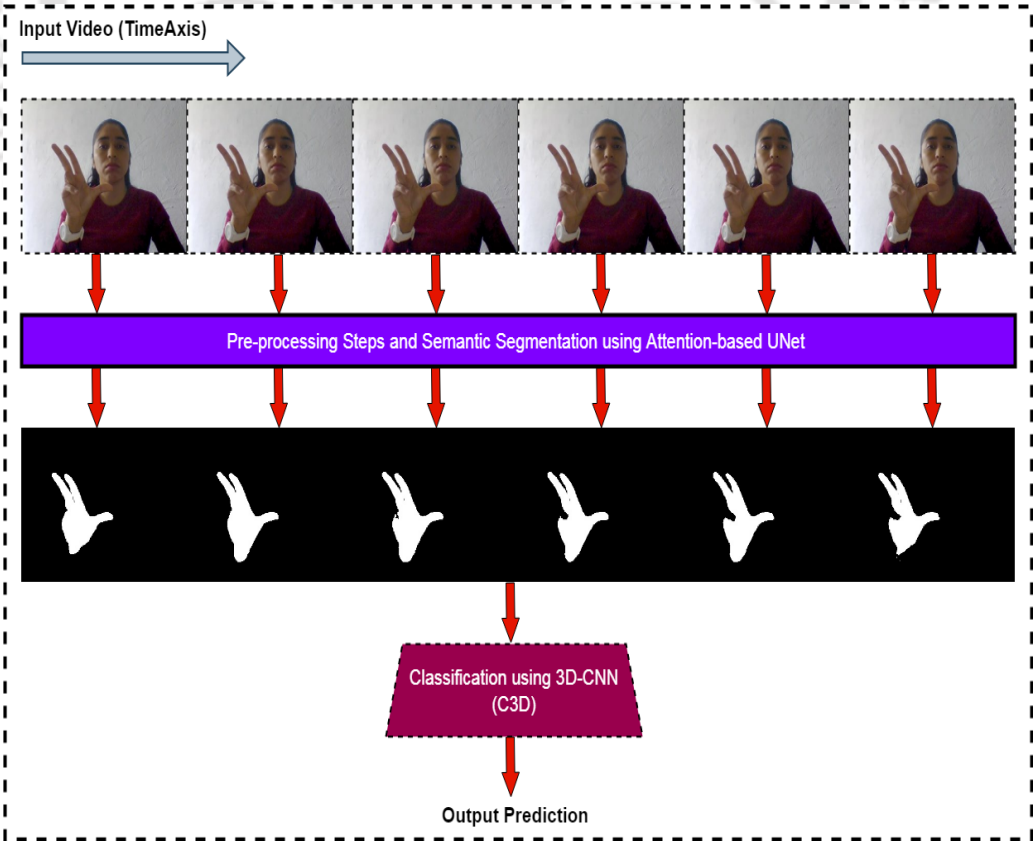
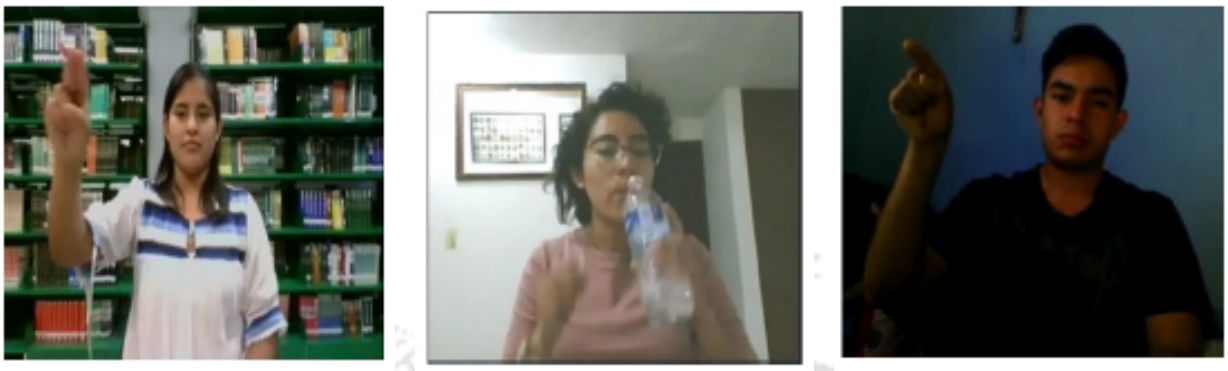


Figure 5.8: The workflow for dynamic hand gesture recognition.



(a) (b) (c)

Figure 5.9: Some examples showing the challenges of IPN hand dataset: (a) Clutter backgrounds, (b) Natural interaction with objects, (c) Weak illumination conditions.

5.3.3.1 Dataset

The dataset used for dynamic hand gesture recognition is the IPN hand dataset [214]. It is a recent dataset with 13 static and dynamic gesture classes for interaction with touchless screens. It contains 4,218 gesture instances and 800,491 frames from 50 subjects in 28 diverse scenes. The recordings have different clutter backgrounds with varying illumination in both static/dynamic scenes as shown in Fig. 5.9. All videos were recorded using a normal PC or laptop in the resolution of 640×480 at 30 fps. The recorded gestures can be used to control the pointer and actions focused on the interaction with touchless screens. Apart from the RGB frames, real-time optical flow and hand segmentation results are also available with the database.

5.3.3.2 Pre-processing

To effectively train the CNNs, we have adopted a few pre-processing steps as performed by [249]. To reduce the chances of CNNs being trained on noisy data some filtering operations are also performed. It is also noticeable that preprocessing is only done with the training data to reduce the elements resulting in degraded performance and it is a prior expense of time. The various pre-processing stages are outlined below-

(i) Each gesture video is first converted into several frames, then each frame is processed individually. Each frame is normalized to [0, 1] to reduce the computation.

(ii) The unwanted noise and spots in the frame are removed using median filtering.

(iii) The illumination variations in the frame are canceled out using Histogram equalization.

(iv) The hand portions in the frames are then segmented out using previously explained attention-based UNet structure.

(v) The segmented frames are then combined again to form the video sequence for training the 3D-CNNs.

5.3.3.3 Segmentation

Static gesture is represented by single still image and dynamic gesture is nothing but a sequence of images. The segmentation method used for dynamic gestures is exactly the same as for the static gesture. The only difference with dynamic gesture is that the semantic segmentation mask is obtained for each streaming frame. Finally, all the segmented frames are combined into a single video to be processed by the next classification stage.

5.3.3.4 Classification

The original C3D [168] was designed for RGB videos. The number of parameters of the networks depends on the resolution of input frames. The original C3D was trained on the large-scale dataset Sport1M [163], which consists of 1.1M videos downloaded from YouTube consisting of 487 sports classes. 2D-CNN is extended to a 3D-CNN by incorporating the temporal dimension of a video sequence. In 2D-CNNs, the dimension of each feature map is $c \times h \times w$, where c represents the number of filters in the convolutional (conv) layer, h and w represents the height and width of the feature map. In 3D-CNNs, the dimension of each feature map is $c \times l \times h \times w$, where additional parameter l represents the number of frames. This network extracts the features which are compact and generic while being discriminative. As we worked on two smaller databases, a slightly different architecture with 5 conv layers is employed which has a smaller number of parameters compared to the original C3D [168] with 8 conv layers. The proposed network has 5 space-time conv layers with 64, 128, 256, 256, 256 kernels. Each conv layer is followed by a rectified linear unit (ReLU) and a space-time max-pooling layer. All 3D convolution kernels are of size $3 \times 3 \times 3$, that gives the best performance [168] with stride

$1 \times 1 \times 1$. Max pooling kernels are of size $2 \times 2 \times 2$ except for the first, where it is $2 \times 2 \times 1$ and stride is $2 \times 2 \times 1$. The conv layers are followed by two dense layers with 2048 and 1024 neurons and ReLU as the activation function. To avoid over-fitting while learning, there is a dropout in each dense layers. The parameter of dropout is set to 0.4, which means the layer randomly excludes 40% of neurons. The final dense layer of the classifier has 13 neurons giving us the respective class labels where softmax function is used for activation.

5.4 Experimental Results

This section gives an idea of the work performed and analysis of the results obtained. The entire experiment was done in a python environment, taking the help of the NVIDIA Tesla K80 GPU in Google Colab.

5.4.1 Results for Static Gestures

The experimentatation is carried out on the Brazilian Sign Language dataset given by Bastos *et al.* [11] for static hand gesture recognition. The results for this section is given in two subsections namely for segmentation and classification.

5.4.1.1 Results for segmentation stage

The first step of the proposed method is to find the segmented masks of the dataset samples. The input image is first resized into $64 \times 64 \times 1$ image, and then, it is passed through the UNet architecture along with the corresponding segmented mask provided by the dataset. The training process was executed for 25 epochs in a regular PC of 8 GB RAM and 3.5 GHz processor speed. It took about 413 seconds, i.e., 103 ms/image to segment 4000 test images with an accuracy of 96.33 %. The comparison of the segmented masks provided by [11] and the generated segmented masks are shown in Fig. (5.10). From the figure, it is evident that, the segmentation results obtained in this work is better than the ones obtained by [11]. It may be subjective, hence, in order to support the results obtained, mean Jaccard Similarity Index and the mean PSNR values are calculated.

Jaccard Similarity Index, also known as Intersection over Union (IoU), computes the sim-

ilarities between the elements of two sets. It ranges in the interval [0,1], with 0 referring the sets to be disjoint and 1 signifying the exact match. Mean Jaccard Similarity Index is defined as:

$$J = \frac{1}{M}\sum_{j=1}^{M}\frac{\sum_i min(I_i, G_i)}{\sum_i max(I_i, G_i)} = \frac{I \cap G}{I \cup G} \tag{5.12}$$

where M is the number of classes and I and G are the vectorized segmented mask and ground truth respectively. It measures the overlap between two bounding boxes I and G as the ratio of the total covered area.

Another measure used is PSNR, the ratio of peak signal power to noise power. The mean PSNR measure is given as:

$$PSNR = \frac{1}{M}\sum_{j=1}^{M} 10log(peak^2/MSE(f, gt)) \tag{5.13}$$

$$MSE = \frac{1}{N_1 N_2}\sum_{x=1}^{N_1}\sum_{y=1}^{N_2}(f(x,y) - gt(x,y))^2 \tag{5.14}$$

where, MSE stands for Mean Square Error and f and gt represents the image and the ground truth respectively. N_1 and N_2 are the number of rows and columns of the image.

Table 5.1: Comparison of the segmentation performance measures for the Brazilian Sign Language (Libras) dataset

	Bastos *et al.* [11]	Proposed method (without attention module)	Proposed method (with attention module)
Jaccard (IoU)	0.76	0.89	**0.98**
PSNR	15.11	15.62	**17.32**

Table 5.1 gives a comparison between the segmentation result of this work and the skin segmentation using multilayer perceptron adopted by [11]. The table shows a slightly better result for the PSNR measure, but the proposed work has achieved a significant improvement in the Jaccard Similarity performance measure which justifies the betterment in the subjective results shown in Fig. (5.10).

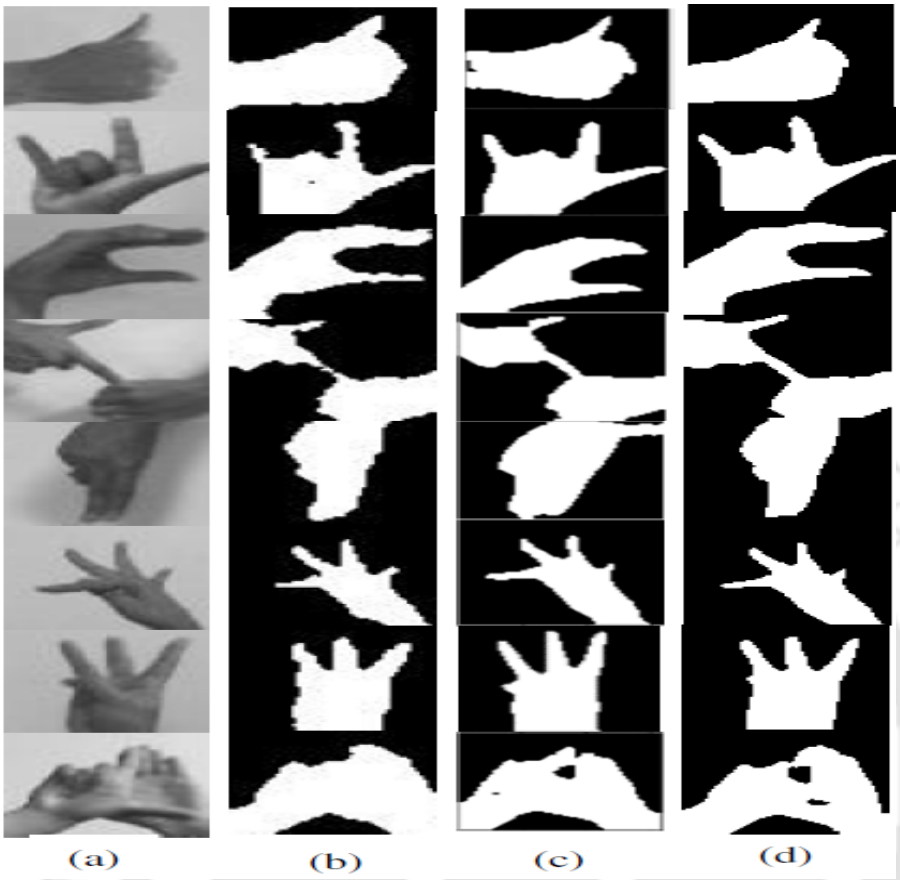


Figure 5.10: Comparion among semantic segmentation outputs for static gestures: (a) shows the gesture images, (b) shows the segmented masks obtained by [11], (c) shows the segmented masks obtained through UNet without attention mechanism and (d) shows the segmented masks obtained through attention-based UNet.

5.4.1.2 Results for classification stage

After segmentation, the images are passed through the classification stage. In [11], the feature vector comprising of two shape descriptors - Histogram of Oriented Gradients (HOG) and Zernike Invariant Moments (ZIM), are used for training and testing a two-stage Multi-Layer Perceptron (MLP) classifier. This method produced a high recognition rate. Since the proposed method also uses the same dataset, the work in [11] is used for comparing the classification results. The original gesture images could also have been passed through the classifier without going through the segmentation stage. But, through experimentation, it is found that with the inclusion of the attention-based semantic segmentation the accuracy of classification has increased from 93.28% to 99.50% (Table 5.2). This has also helped us achieve much better results compared to the prior work by Bastos *et al.*

Table 5.2: Comparison of Accuracy Performance (%) for the Brazilian Sign Language (Libras) dataset

	Without pre-segmentation	With pre-segmentation (without attention module)	With pre-segmentation (with attention module)
Bastos *et al.* [11]	–	97.14%	–
Our method	93.28%	98.97%	**99.50%**

The classifier training was done on 106800 images for 20 epochs and the model with the best accuracy was saved. It took about 280 seconds for each epoch in the NVIDIA Tesla K80 GPU in Google Colab. For testing, 200 images were considered from 34 classes each in a 10-fold cross-validation pattern. Fig. (5.11) depicts the training and the testing accuracy of the model. Figure 5.12 gives the detailed confusion matrix of the testing phase. The yellow colored cells show the true positives while the brown-colored cells represent the misclassified samples. Table 5.3 shows the comparison of the results by Bastos *et al.* and the proposed method.

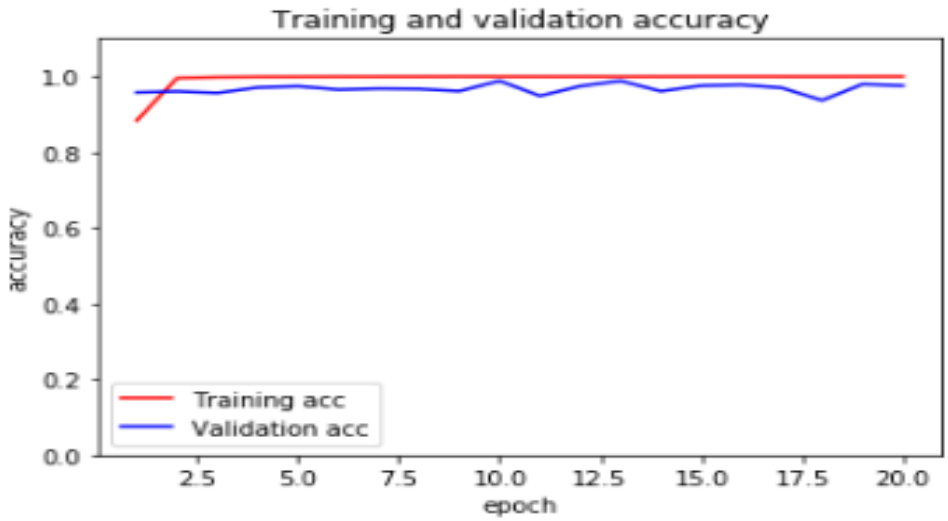


Figure 5.11: Plot depicting training and validation/testing accuracy for 20 epochs.

5.4.2 Results for Dynamic Gestures

The experimentation is carried out on the IPN hand dataset [214] for dynamic hand gesture recognition. The databases used in previous chapters do not contain the ground truth segmented masks. So, here we are using a new dataset for dynamic gesture since it contains the ground truth segmented masks which are required for training purposes. The results for this section are given in two subsections namely for segmentation and classification.

Table 5.3: Table showing the comparison between Bastos *et al.* and the proposed method for static gestures

Class	Bastos *et al.* [11]	Proposed Method
1	100	100
2	95.83	100
4	99.16	100
5	100	100
7	96.67	100
Adult	95.83	100
America	100	100
Plane	100	100
C	90	100
House	100	100
D	96.67	100
E	98.33	100
G	99.16	100
Gas	100	100
I	89.16	100
Identity	98.33	96
Together	98.33	100
L	95	94
Lei	99.16	100
N	99.16	100
O	96.67	96
P	100	100
Word	100	100
Stone	99.16	100
Little	90.83	100
Q	98.83	100
R	96.67	100
T	100	100
U	90.83	100
V	95.83	100
Verb	95	97
W	95	100
X	98.33	100
Y	95	100
Average	97.14	99.50

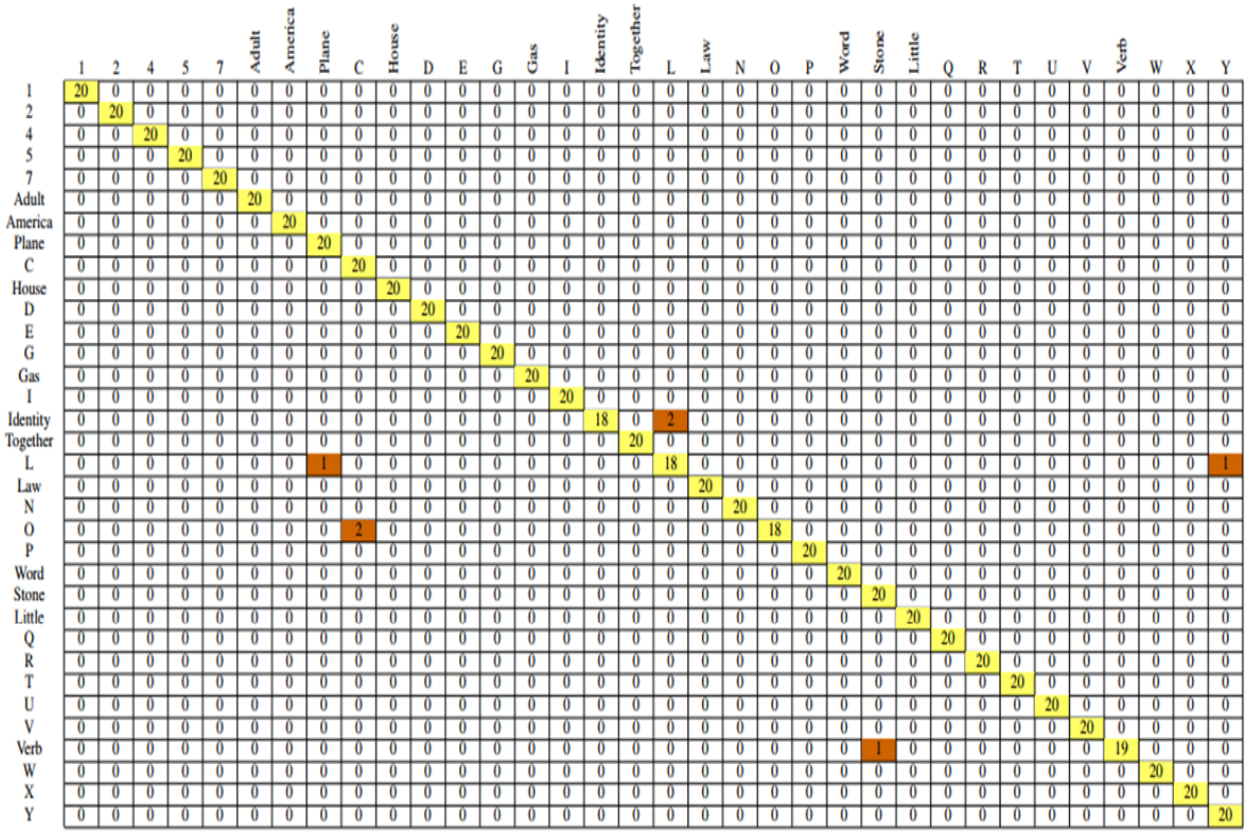


Figure 5.12: Confusion Matrix for Static Gestures

5.4.2.1 Results for segmentation stage

The continuous video sequences are segmented into isolated gesture samples based on the beginning and ending frames by manual annotation. Since we calculate on a 3D-CNN model, first, the semantic segmentation masks are evaluated for each streaming frame. The subjective comparison of the ground truth segmented mask provided by [214] and the generated segmented masks are shown in Fig. (5.13). It is seen that when the frame is shaky, then segmentation is not so accurate. For quantitative analysis, the mean Jaccard Similarity Index and the mean PSNR values are given in Table 5.4.

Table 5.4: Comparison of segmentation performance measures for IPN hand dataset

	Proposed method (without attention module)	Proposed method (with attention module)
Jaccard (IoU)	0.86	**0.93**
PSNR	13.60	**19.52**

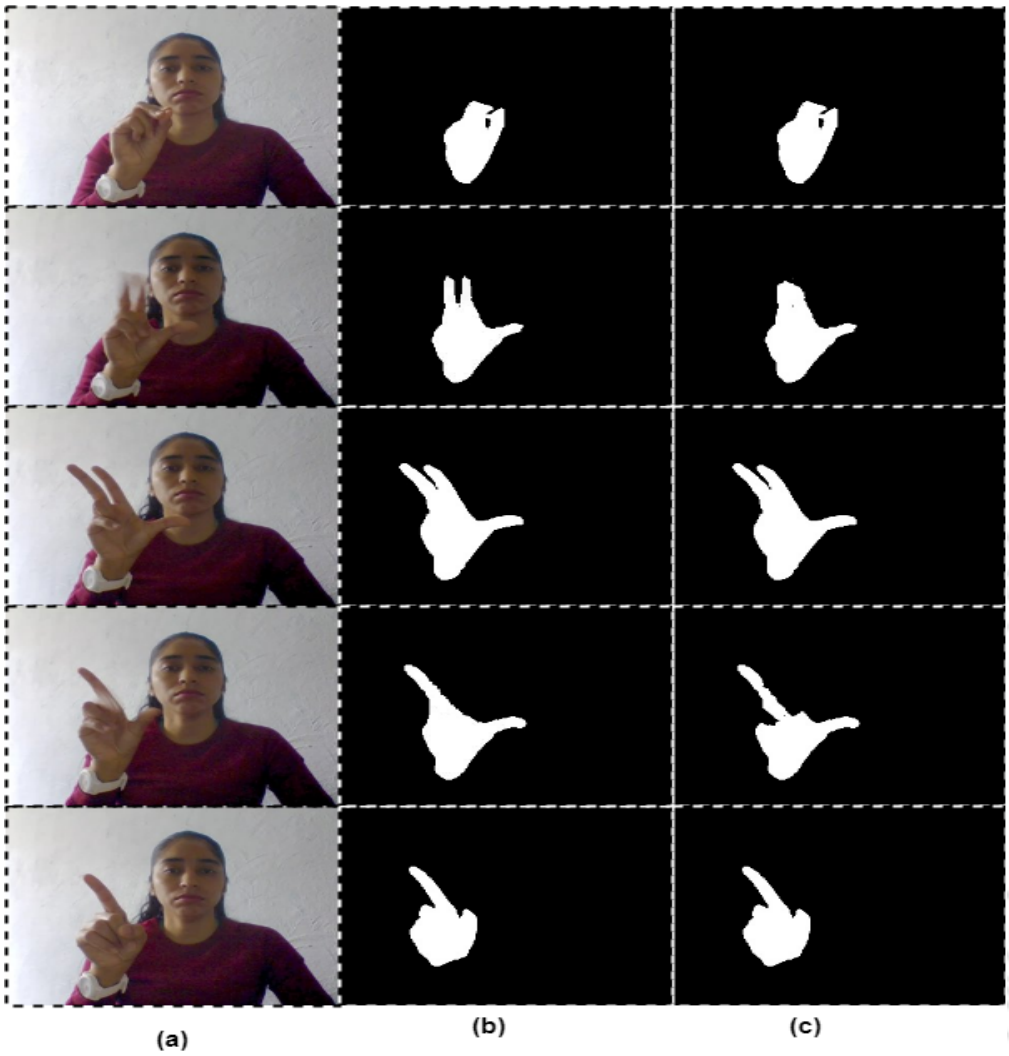


Figure 5.13: Semantic segmentation output for a few dynamic gesture frames from IPN hand dataset: (a) Shows the gesture images, (b) Ground truths, and (c) Shows the corresponding segmented masks obtained by our method.

5.4.2.2 Results for classification stage

The task of the classifier is to predict class labels for each gesture sample as shown in Fig. 5.8. We use classification accuracy, which is the percent of correctly labeled examples as an evaluation metric for this classification task. Table 5.5 shows the individual class accuracy using the proposed method.

We have compared our results in Table 5.6 with [214] where authors have used RGB frames as a single input and RGB frames with segmented masks (RGB-Seg) or RGB frames with optical flow (RGB-Flow) as multi-modal inputs. From Table 5.6, it is clear that our method

Table 5.5: Table showing the individual class accuracy for IPN hand dataset

Class	B0A	B0B	G01	G02	G03	G04	G05	G06	G07	G08	G09	G10	G11
Accuracy (%)	89.21	86.23	90.20	87.30	88.72	87.52	86.80	87.50	88.23	88.50	87.60	87.91	87.75

has achieved SOTA performance. This is due to the effective attention-based segmentation process which has led to better classification results. From the 324 instances of each class, 64 instances i.e. almost 20% is used for testing. Figure 5.14 gives the detailed confusion matrix of the testing phase. The yellow colored cells show the true positives while the brown-colored cells represent the misclassified samples.

Table 5.6: Comparison of performance measures (% accuracy) for Isolated IPN Gestures

Method	**Modality**	**Accuracy (%)**
C3D [214]	RGB	77.75%
ResNeXt-101 [214]	RGB-Flow	86.32%
ResNeXt-101 [214]	RGB-Seg	84.77%
ResNet-50 [214]	RGB-Flow	74.65%
ResNet-50 [214]	RGB-Seg	75.11%
Our Method	Segmented Masks	**87.95%**

	B0A	B0B	G01	G02	G03	G04	G05	G06	G07	G08	G09	G10	G011
B0A	62	0	0	0	0	0	0	0	0	0	0	2	0
B0B	0	59	0	3	0	0	0	0	0	0	2	0	0
G01	0	0	61	0	0	0	0	0	0	3	0	0	0
G02	0	2	0	60	0	0	0	0	0	0	2	0	0
G03	0	0	0	0	62	0	0	0	1	0	0	1	0
G04	0	0	0	0	0	64	0	0	0	0	0	0	0
G05	0	0	0	0	0	0	61	1	0	0	0	2	0
G06	0	0	0	0	0	0	2	60	0	0	0	2	0
G07	0	0	0	0	1	0	0	0	63	0	0	0	0
G08	1	0	2	0	0	0	0	0	0	61	0	0	0
G09	0	0	0	2	0	0	0	0	0	0	62	0	0
G10	0	0	0	0	0	0	0	0	0	0	0	62	2
G11	0	0	0	0	0	0	0	0	0	0	0	1	63

Figure 5.14: Confusion Matrix for Dynamic Gestures

5.5 Summary

Motivated by the success of the attention-based methods, and considering it from the view of focus and region-wise representations, we have embedded an attention-based module in semantic segmentation to capture global contexts from the perspective of space and channel for better feature representations. CNN proves to be a magnificent tool for classification, whose benefit can also be exploited for image segmentation tasks. Hence, two of the CNN-based deep models - UNet and VGG16 are employed in this work concerning semantic segmentation and classification respectively to achieve state-of-the-art results for static hand gesture recognition problems. An attention-based UNet model is used for segmenting the gesture images in the preprocessing stage, which is basically an encoding-decoding structure. It adds fine information to the coarse layers, and thus, helps in improving the segmentation results. Moreover, benefiting from the attention mechanism, UNet can be used more efficiently and effectively than other segmentation methods. The hierarchical pattern learned by the UNet projects accurate visualization of the problem at hand. Speaking of the classification stage, a pre-trained VGG16 network is used to extract the features of the segmented images, and the extracted feature maps are passed through the designed classifier. This same process has also been extended for dynamic hand gesture recognition as well. Here, in place of 2D-CNN, a 3D-CNN is used as a classifier since it can capture more subtle spatio-temporal features. Comprehensive empirical results verify that our proposed model is better than state-of-the-art. The main contributions of our proposed model are as follows:

(i) A deep supervised attention module to focus and guide the learning of information for segmentation in UNet structure.

(ii) We have proposed a novel approach for both static and dynamic hand gesture recognition where the attention technique is used to increase the segmentation performance on similar gestures.

(iii) We have demonstrated how the quality of segmented images impacts the performance of hand gesture classification through experiments on two databases and have proven our network has better results than state-of-the-art on a noisy dataset.

6

Conclusion and Directions for Future Work

There is a humbling amount of past works on vision-based hand gesture recognition. The objective of this thesis is to give some direction towards the development of a vision-based hand gesture recognition system specifically for trajectory-based or dynamic gestures. One simple agenda of a trajectory-based gesture recognition system can be to track the hand to find its trajectory in the gesture video. So, with this motive in hand, this dissertation describes various proposed trajectory-based methods to track the movement of the hand. In our first work, a system is developed where the whole trajectory of the gesture is converted to a hand-trajectory-based-contour-image depicting the contour of the gesture. But in this work, skin color-based segmentation has been applied as a pre-processing step which may fail in many constrained cases like the presence of skin-like colors in the background etc. Hence, in our next work, a fusion-based scheme is adopted that can recognize hand gestures irrespective of the shape, size and color of the hand. Lastly, a deep learning attention-based method has been applied to improve the performance in gesture recognition which is applicable for both static as well as dynamic hand gestures. This chapter reflects on these contributions, discusses future work for gesture recognition, and concludes. We hope that future gesture recognition algorithm developers find our contributions useful, and get benefited from our schemes without having to reinvent their own.

6.1 Summary

At the beginning of this thesis, challenges faced by the computer vision community in recognizing hand gestures are mentioned in the introduction chapter. These challenges include segmentation-related problems, constraints with extracted features and tracking related problems etc. Segmentation-related problems arise due to the presence of skin-like colors in the background, illumination variations, background complexity, occlusion etc. In this research work, we have proposed multiple frameworks to overcome some of the above-mentioned challenges in hand gesture recognition. The main goal of this dissertation is to detect and recognize hand gestures which can be used as a pre-processing step for different HCI applications. In our first work, we have proposed a two-level segmentation process by integrating motion and color information, and a double-tracking approach for tracking the gesture trajectory. In the preprocessing step, a skin segmentation framework suppressing the luminance component to compensate for the illumination variations is used. A double-tracking system incorporating mean-shift and particle filters with occlusion handling ability is applied for tracking the gesture trajectory. But tracking the physical movement of the hand is quite challenging due to the varied size, shape and color of the hand. So, in our second work, a motion template guided by optical flow (OFMT) is proposed which can track the movement of the hand irrespective of the shape, size and color of the hand. It is used in a 2D-CNN network along with a 3D-CNN layer in a fusion scheme to enhance the recognition performance. Most hand-crafted features require prior knowledge which can be avoided by deep learning methods that have a robust and effective feature learning capability directly from the raw data. That is why we have used deep learning methods for both feature extraction as well as classification purposes. Lastly, deep learning methods have also been tried in semantic segmentation incorporating attention module to achieve improved gesture recognition results in both static and dynamic gestures.

The summary of all the chapters of this dissertation is highlighted as follows:

i) **Chapter 1** provides a brief description of a typical gesture recognition system and its different components. It also talks about the importance of gestures in the human-computer interaction (HCI) community and different acquisition techniques. Different applications and some recent advancements of gesture recognition systems are also discussed in this chapter. In Chapter 1, various challenges in realizing a gesture recognition system are

also discussed. Major challenges include - segmentation-related problems, gesture spotting problems, problems related to two-handed gesture recognition, constraints with extracted features and difficulties related to the articulated shape of the hand. Finally, Chapter 1 is ended up with research motivation and the organization of the thesis.

ii) **Chapter 2** reviews several existing methods for hand gesture recognition under various conditions. The review section is presented in three parts according to the stages of a VGR system: acquisition & pre-processing, gesture representation & feature extraction, and recognition. The recognition section is again discussed in three subsections: conventional methods on RGB data, depth-based methods on RGB-D data, and deep-learning-based methods. The summary of the review and the scope for this thesis work is discussed in the last section of the chapter.

iii) **Chapter 3** focuses on developing a hand gesture recognition framework for isolated dynamic gestures using a convolutional neural network (CNN). In the preprocessing step, a two-level segmentation process with compensation for the illumination variations and a double-tracking system with occlusion handling ability is used for tracking the gesture trajectory. In this step, each isolated dynamic gesture is converted into single image consisting of the contour of the gesture trajectory that we call hand-trajectory-based-contour-images. The CNN used for feature extraction has shown competitive performance on three different datasets.

iv) **Chapter 4** describes a proposed two-stream fusion model for hand gesture recognition. The two-stream network consists of two layers - a 3D convolutional neural network (C3D) that takes gesture videos as input and a 2D-CNN that takes novel OFMT images as input. C3D has shown its efficiency in capturing spatio-temporal information of a video. It is seen that skin segmentation may fail in different cases like the presence of skin-like colours in the background. Hence, in this work, a motion template guided by optical flow (OFMT) is proposed which can track the movement of the hand irrespective of the shape, size and color of the hand. OFMT is a compact representation of the motion information of a gesture encoded into a single image. In the experimentation, different datasets using bare hands with an open palm, and folded palms wearing green gloves are used, and in both cases, we could generate the OFMT images with equal precision. OFMT also helps to

eliminate irrelevant gestures providing additional motion information. Though each stream can work independently, they are combined with a fusion scheme to boost the recognition performance. We have shown the efficiency of the proposed two-stream network on two databases. Here the major contribution is novel OFMT images that can track the moving hand irrespective of the shape, size, and color of the hand.

v) **Chapter 5** explores the use of a deep-learning approaches for both static and dynamic hand gesture recognition. Already deep networks have shown their efficiency in action and gesture recognition fields achieving outstanding results and outperforming "non-deep" state-of-the-art methods. So, in this work deep network has been applied to both segmentation as well as classification problem to achieve improved results. The ability to discern the shape of hands can be a vital issue in improving the performance of static hand gesture recognition. Segmentation itself is a very challenging problem having various constraints like illumination variations, complex background *etc.* The objective of the work is to incorporate the perception of semantic segmentation into a classification problem and make use of the deep neural models to achieve improved results for both static and dynamic gestures. This work utilizes the UNet architecture with attention-module to obtain the semantically segmented masks of the input images, which are then fed to a classifier for recognition. The concept of attention-mechanism adds to the improvement of segmentation accuracy. In this work, for static gestures, the top classifier layer of the VGG16 model is replaced with a classifier designed specifically for classifying the gestures at hand. For dynamic gestures, 3D-CNN (C3D) architecture is used as a classifier that can capture spatial as well as temporal information of a gesture video. The data augmentation process is used in preprocessing to generate a sufficient number of training images for the aforementioned CNN-based models. Significant and improved recognition has been achieved for both static and dynamic hand gesture databases through the inherent feature learning capability of CNN and refined segmentation.

6.2 Thesis Contributions

The main contributions of the thesis can be summarized as follows:

- A two-level segmentation process by integrating motion and color information and a double-tracking system with occlusion handling ability is proposed to track the gesture trajectory.
- A method for converting the trajectory of a gesture video into an image called hand-trajectory-based-contour-image is employed depicting the contour of the gesture trajectory.
- A motion template guided by optical flow (OFMT) is proposed which can track the movement of the hand irrespective of the shape, size and color of the hand.
- A fusion rule for a two-stream network is proposed basically for non-identical streams.
- A simple and flexible architecture to effectively learn contextual information through a deep supervised attention-based semantic segmentation module.
- The rule-based algorithms of attention-based semantic segmentation for static gesture interpretation have successfully been extended for dynamic gesture recognition.
- A limited in-house dataset is created for experimentation in a hand gesture recognition system.

6.3 Future Research Directions

Our proposed work has addressed a number of existing issues for the hand gesture recognition system. Based on the outcome of this thesis work, here we provide some of the possible future directions for further research. It points to certain areas which could benefit other researchers.

- Our proposed work addressed a number of existing issues like hand shape, size and color dependencies of a gesture recognition system. Some other issues like complex and dynamic backgrounds are not explicitly addressed in this dissertation.
- In the future, some other direction of research may be in continuous hand gesture recognition where gesture spotting is a big issue.

- We also have not addressed various issues pertaining to two-hand gesture recognition. Here the hands can overlap or occlude each other, thus impeding recognition of the gestures. Simultaneous tracking of both hands is quite a big issue in two-hand gesture recognition that can be addressed by future researchers.

Bibliography

[1] P. Premaratne, *Human computer interaction using hand gestures*. Springer Science & Business Media, 2014.

[2] A. F. Bobick and J. W. Davis, "The recognition of human movement using temporal templates," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 23, no. 3, pp. 257–267, 2001.

[3] H. Bilen, B. Fernando, E. Gavves, A. Vedaldi, and S. Gould, "Dynamic image networks for action recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 3034–3042.

[4] H. J. Escalante, I. Guyon, V. Athitsos, P. Jangyodsuk, and J. Wan, "Principal motion components for one-shot gesture recognition," *Pattern Analysis and Applications*, vol. 20, no. 1, pp. 167–182, 2017.

[5] C. Sminchisescu, A. Kanaujia, and D. Metaxas, "Conditional models for contextual human motion recognition," *Computer Vision and Image Understanding*, vol. 104, no. 2-3, pp. 210–220, 2006.

[6] S. Gupta, M. Bhuyan, and P. Sasmal, "Occlusion robust object tracking with modified particle filter framework," in *2020 IEEE Applied Signal Processing Conference (ASPCON)*. IEEE, 2020, pp. 257–261.

[7] G. Yu, Z. Hu, H. Lu, and W. Li, "Robust object tracking with occlusion handle," *Neural Computing and Applications*, vol. 20, no. 7, pp. 1027–1034, 2011.

[8] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[9] S. Misra, J. Singha, and R. Laskar, "Vision-based hand gesture recognition of alphabets, numbers, arithmetic operators and ascii characters in order to develop a virtual text-entry interface system," *Neural Computing and Applications*, vol. 29, no. 8, pp. 117–135, 2018.

[10] J. Alon, V. Athitsos, Q. Yuan, and S. Sclaroff, "A unified framework for gesture recognition and spatiotemporal gesture segmentation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 31, no. 9, pp. 1685–1699, 2009.

[11] I. L. Bastos, M. F. Angelo, and A. C. Loula, "Recognition of static gestures applied to brazilian sign language (libras)," in *2015 28th SIBGRAPI Conference on Graphics, Patterns and Images*. IEEE, 2015, pp. 305–312.

[12] D. J. Sawicki and W. Miziolek, "Human colour skin detection in cmyk colour space," *IET Image Processing*, vol. 9, no. 9, pp. 751–757, 2015.

[13] Y. Peng and H. Yin, "Markov random field based convolutional neural networks for image classification," in *International Conference on Intelligent Data Engineering and Automated Learning*. Springer, 2017, pp. 387–396.

[14] T. T. Hewett, R. Baecker, S. Card, T. Carey, J. Gasen, M. Mantei, G. Perlman, G. Strong, and W. Verplank, *ACM SIGCHI curricula for human-computer interaction*. ACM, 1992.

[15] A. Dix, "Human-computer interaction," in *Encyclopedia of database systems*. Springer, 2009, pp. 1327–1331.

[16] P. K. Pisharady, P. Vadakkepat, and A. P. Loh, "Attention based detection and recognition of hand postures against complex backgrounds," *International Journal of Computer Vision*, vol. 101, no. 3, pp. 403–419, 2013.

[17] M. Karam, "Phd thesis: A framework for research and design of gesture-based human-computer interactions," Ph.D. dissertation, University of Southampton, 2006.

[18] M. J. Cheok, Z. Omar, and M. H. Jaward, "A review of hand gesture and sign language recognition techniques," *International Journal of Machine Learning and Cybernetics*, pp. 1–23, 2017.

[19] A. Jaimes and N. Sebe, "Multimodal human–computer interaction: A survey," *Computer vision and image understanding*, vol. 108, no. 1-2, pp. 116–134, 2007.

[20] T. B. Moeslund, A. Hilton, and V. Krüger, "A survey of advances in vision-based human motion capture and analysis," *Computer vision and image understanding*, vol. 104, no. 2-3, pp. 90–126, 2006.

[21] G. Sandbach, S. Zafeiriou, M. Pantic, and L. Yin, "Static and dynamic 3d facial expression recognition: A comprehensive survey," *Image and Vision Computing*, vol. 30, no. 10, pp. 683–697, 2012.

[22] S. Kumar, M. Bhuyan, and B. K. Chakraborty, "Extraction of informative regions of a face for facial expression recognition," *IET Computer Vision*, vol. 10, no. 6, pp. 567–576, 2016.

[23] F. Song, X. Tan, S. Chen, and Z.-H. Zhou, "A literature survey on robust and efficient eye localization in real-life scenarios," *Pattern Recognition*, vol. 46, no. 12, pp. 3157–3173, 2013.

[24] S. Mitra and T. Acharya, "Gesture recognition: A survey," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 37, no. 3, pp. 311–324, 2007.

[25] M. El Ayadi, M. S. Kamel, and F. Karray, "Survey on speech emotion recognition: Features, classification schemes, and databases," *Pattern Recognition*, vol. 44, no. 3, pp. 572–587, 2011.

[26] T. Kinnunen and H. Li, "An overview of text-independent speaker recognition: From features to supervectors," *Speech communication*, vol. 52, no. 1, pp. 12–40, 2010.

[27] J. Li, L. Deng, Y. Gong, and R. Haeb-Umbach, "An overview of noise-robust automatic speech recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 4, pp. 745–777, 2014.

[28] P. Kumar, S. S. Rautaray, and A. Agrawal, "Hand data glove: A new generation real-time mouse for human-computer interaction," in *Recent Advances in Information Technology (RAIT), 2012 1st International Conference on*. IEEE, 2012, pp. 750–755.

[29] M. Pantic and L. J. Rothkrantz, "Toward an affect-sensitive multimodal human-computer interaction," *Proceedings of the IEEE*, vol. 91, no. 9, pp. 1370–1390, 2003.

[30] S. Oviatt, "Multimodal interfaces," *The human-computer interaction handbook: Fundamentals, evolving technologies and emerging applications*, vol. 14, pp. 286–304, 2003.

[31] R. M. Jiang, A. H. Sadka, and D. Crookes, "Multimodal biometric human recognition for perceptual human–computer interaction," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 40, no. 6, pp. 676–681, 2010.

[32] L. W. Campbell, D. A. Becker, A. Azarbayejani, A. F. Bobick, and A. Pentland, "Invariant features for 3-d gesture recognition," in *Automatic Face and Gesture Recognition, 1996., Proceedings of the Second International Conference on.* IEEE, 1996, pp. 157–162.

[33] Z. Zhang, "Microsoft kinect sensor and its effect," *IEEE multimedia*, vol. 19, no. 2, pp. 4–10, 2012.

[34] M. Jacob, C. Cange, R. Packer, and J. P. Wachs, "Intention, context and gesture recognition for sterile mri navigation in the operating room," in *Iberoamerican Congress on Pattern Recognition.* Springer, 2012, pp. 220–227.

[35] W. Kumara, K. Wattanachote, B. Battulga, T. K. Shih, and W.-Y. Hwang, "A kinect-based assessment system for smart classroom," *International Journal of Distance Education Technologies (IJDET)*, vol. 13, no. 2, pp. 34–53, 2015.

[36] "Softkinetic's gesture control technology rolls out in additional car model," May 2017.

[37] J. F. Lichtenauer, E. A. Hendriks, and M. J. Reinders, "Sign language recognition by combining statistical dtw and independent classification," *IEEE transactions on pattern analysis and machine intelligence*, vol. 30, no. 11, pp. 2040–2046, 2008.

[38] K. M. Sagayam and D. J. Hemanth, "Hand posture and gesture recognition techniques for virtual reality applications: a survey," *Virtual Reality*, vol. 21, no. 2, pp. 91–107, 2017.

[39] A. Kulshreshth, K. Pfeil, and J. J. LaViola, "Enhancing the gaming experience using 3d spatial user interface technologies," *IEEE computer graphics and applications*, vol. 38, no. 3, pp. 16–23, 2017.

[40] S. Reifinger, F. Wallhoff, M. Ablassmeier, T. Poitschke, and G. Rigoll, "Static and dynamic hand-gesture recognition for augmented reality applications," in *International Conference on Human-Computer Interaction.* Springer, 2007, pp. 728–737.

[41] L. McCowan, D. Gatica-Perez, S. Bengio, G. Lathoud, M. Barnard, and D. Zhang, "Automatic analysis of multimodal group actions in meetings," *IEEE transactions on pattern analysis and machine intelligence*, vol. 27, no. 3, pp. 305–317, 2005.

[42] W. L. Ng, C. K. Ng, N. K. Noordin, and B. M. Ali, "Gesture based automating household appliances," in *International Conference on Human-Computer Interaction.* Springer, 2011, pp. 285–293.

[43] L. Tao, L. Zappella, G. D. Hager, and R. Vidal, "Surgical gesture segmentation and recognition," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2013, pp. 339–346.

[44] N. Kim-Tien, N. Truong-Thinh, and T. D. Cuong, "A method for controlling wheelchair using hand gesture recognition," in *Robot Intelligence Technology and Applications 2012*. Springer, 2013, pp. 961–970.

[45] P. Smith, M. Shah, and N. da Vitoria Lobo, "Determining driver visual attention with one camera," *IEEE transactions on intelligent transportation systems*, vol. 4, no. 4, pp. 205–218, 2003.

[46] C. Pickering, "The search for a safer driver interface: a review of gesture recognition human machine interface," *Computing & Control Engineering Journal*, vol. 16, no. 1, pp. 34–40, 2005.

[47] B. Zeng, G. Wang, and X. Lin, "A hand gesture based interactive presentation system utilizing heterogeneous cameras," *Tsinghua Science and Technology*, vol. 17, no. 3, pp. 329–336, 2012.

[48] B. Hariharan, S. Padmini, and U. Gopalakrishnan, "Gesture recognition using kinect in a virtual classroom environment," in *Digital Information and Communication Technology and it's Applications (DICTAP), 2014 Fourth International Conference on*. IEEE, 2014, pp. 118–124.

[49] Y. Arafa and A. Mamdani, "Building multi-modal personal sales agents as interfaces to e-commerce applications," in *International Computer Science Conference on Active Media Technology*. Springer, 2001, pp. 113–133.

[50] S.-Y. Peng, K. Wattanachote, H.-J. Lin, and K.-C. Li, "A real-time hand gesture recognition system for daily information retrieval from internet," in *Ubi-media computing (U-Media), 2011 4th international conference on*. IEEE, 2011, pp. 146–151.

[51] B. K. Chakraborty, D. Sarma, M. Bhuyan, and K. F. MacDorman, "Review of constraints on vision-based gesture recognition for human–computer interaction," *IET Computer Vision*, vol. 12, no. 1, pp. 3–15, 2017.

[52] B. K. Chakraborty, M. Bhuyan, and S. Kumar, "Combining image and global pixel distribution model for skin colour segmentation," *Pattern Recognition Letters*, vol. 88, pp. 33–40, 2017.

[53] A. Utsumi and J. Ohya, "Multiple-hand-gesture tracking using multiple cameras," in *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, vol. 1. IEEE, 1999, pp. 473–478.

[54] T.-N. Nguyen, D.-H. Vo, H.-H. Huynh, and J. Meunier, "Geometry-based static hand gesture recognition using support vector machine," in *Control Automation Robotics & Vision (ICARCV), 2014 13th International Conference on.* IEEE, 2014, pp. 769–774.

[55] S. P. Priyal and P. K. Bora, "A study on static hand gesture recognition using moments," in *Signal Processing and Communications (SPCOM), 2010 International Conference on.* IEEE, 2010, pp. 1–5.

[56] K.-p. Feng and F. Yuan, "Static hand gesture recognition based on hog characters and support vector machines," in *Instrumentation and Measurement, Sensor Network and Automation (IMSNA), 2013 2nd International Symposium on.* IEEE, 2013, pp. 936–938.

[57] D. K. Ghosh and S. Ari, "Static hand gesture recognition using mixture of features and svm classifier," in *Communication Systems and Network Technologies (CSNT), 2015 Fifth International Conference on.* IEEE, 2015, pp. 1094–1099.

[58] ——, "A static hand gesture recognition algorithm using k-mean based radial basis function neural network," in *Information, Communications and Signal Processing (ICICS) 2011 8th International Conference on.* IEEE, 2011, pp. 1–5.

[59] M. Bhuyan, D. Ghosh, and P. Bora, "Feature extraction from 2d gesture trajectory in dynamic hand gesture recognition," in *Cybernetics and Intelligent Systems, 2006 IEEE Conference on.* IEEE, 2006, pp. 1–6.

[60] ——, "Continuous hand gesture segmentation and co-articulation detection," in *Computer Vision, Graphics and Image Processing.* Springer, 2006, pp. 564–575.

[61] W. Lu, Z. Tong, and J. Chu, "Dynamic hand gesture recognition with leap motion controller," *IEEE Signal Processing Letters*, vol. 23, no. 9, pp. 1188–1192, 2016.

[62] M. Bhuyan, "Fsm-based recognition of dynamic hand gestures via gesture summarization using key video object planes," *International Journal of Computer and Communication Engineering*, vol. 6, pp. 248–259, 2012.

[63] W. Tan, C. Wu, S. Zhao, and J. Li, "Dynamic hand gesture recognition using motion trajectories and key frames," in *Advanced computer control (ICACC), 2010 2nd international conference on*, vol. 3. IEEE, 2010, pp. 163–167.

[64] R. Yang, S. Sarkar, and B. Loeding, "Handling movement epenthesis and hand segmentation ambiguities in continuous sign language recognition using nested dynamic programming," *IEEE transactions on pattern analysis and machine intelligence*, vol. 32, no. 3, pp. 462–477, 2010.

[65] Y. Huang, T. S. Huang, and H. Niemann, "Two-handed gesture tracking incorporating template warping with static segmentation," in *Proceedings of Fifth IEEE International Conference on Automatic Face Gesture Recognition*. IEEE, 2002, pp. 275–280.

[66] O. Aran and L. Akarun, "Recognizing two handed gestures with generative, discriminative and ensemble methods via fisher kernels," in *International Workshop on Multimedia Content Representation, Classification and Security*. Springer, 2006, pp. 159–166.

[67] I. Oikonomidis, N. Kyriazis, and A. A. Argyros, "Tracking the articulated motion of two strongly interacting hands," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2012, pp. 1862–1869.

[68] R.-L. Hsu, M. Abdel-Mottaleb, and A. K. Jain, "Face detection in color images," *IEEE transactions on pattern analysis and machine intelligence*, vol. 24, no. 5, pp. 696–706, 2002.

[69] J. Han, G. Award, A. Sutherland, and H. Wu, "Automatic skin segmentation for gesture recognition combining region and support vector machine active learning," in *Automatic Face and Gesture Recognition, 2006. FGR 2006. 7th International Conference on*. IEEE, 2006, pp. 237–242.

[70] K. Sobottka and I. Pitas, "A novel method for automatic face segmentation, facial feature extraction and tracking," *Signal processing: Image communication*, vol. 12, no. 3, pp. 263–281, 1998.

[71] D. Chai and K. N. Ngan, "Face segmentation using skin-color map in videophone applications," *IEEE Transactions on circuits and systems for video technology*, vol. 9, no. 4, pp. 551–564, 1999.

[72] D. Sarma and M. K. Bhuyan, "Hand gesture recognition using deep network through trajectory-to-contour based images," in *15th IEEE India Council International Conference (INDICON)*, 2018, pp. 1–6.

[73] J. Yang, W. Lu, and A. Waibel, "Skin-color modeling and adaptation," in *Asian Conference on Computer Vision*. Springer, 1998, pp. 687–694.

[74] M.-H. Yang and N. Ahuja, "Gaussian mixture model for human skin color and its applications in image and video databases," in *Storage and Retrieval for Image and Video Databases VII*, vol. 3656. International Society for Optics and Photonics, 1998, pp. 458–467.

[75] J. Y. Lee and S. I. Yoo, "An elliptical boundary model for skin color detection," in *Proc. of the 2002 International Conference on Imaging Science, Systems, and Technology*, 2002.

[76] M. J. Jones and J. M. Rehg, "Statistical color models with application to skin detection," *International Journal of Computer Vision*, vol. 46, no. 1, pp. 81–96, 2002.

[77] D. A. Brown, I. Craw, and J. Lewthwaite, "A som based approach to skin detection with application in real time systems." in *BMVC*, vol. 1. Citeseer, 2001, pp. 491–500.

[78] P. Ng and C.-M. Pun, "Skin color segmentation by texture feature extraction and k-mean clustering," in *Computational Intelligence, Communication Systems and Networks (CICSyN), 2011 Third International Conference on*. IEEE, 2011, pp. 213–218.

[79] L. Chen, J. Zhou, Z. Liu, W. Chen, and G. Xiong, "A skin detector based on neural network," in *Communications, Circuits and Systems and West Sino Expositions, IEEE 2002 International Conference on*, vol. 1. IEEE, 2002, pp. 615–619.

[80] R. Khan, A. Hanbury, and J. Stoettinger, "Skin detection: A random forest approach," in *Image Processing (ICIP), 2010 17th IEEE International Conference on*. IEEE, 2010, pp. 4613–4616.

[81] O. Rotem, H. Greenspan, and J. Goldberger, "Combining region and edge cues for image segmentation in a probabilistic gaussian mixture framework," in *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*. IEEE, 2007, pp. 1–8.

[82] Z. Yin and R. Collins, "Moving object localization in thermal imagery by forward-backward mhi," in *2006 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'06)*. IEEE, 2006, pp. 133–133.

[83] Y. Kameda and M. Minoh, "A human motion estimation method using 3-successive video frames," in *International conference on virtual systems and multimedia*, 1996, pp. 135–140.

[84] K. Fukunaga and L. Hostetler, "The estimation of the gradient of a density function, with applications in pattern recognition," *IEEE Transactions on information theory*, vol. 21, no. 1, pp. 32–40, 1975.

[85] P. Dondi, L. Lombardi, and M. Porta, "Development of gesture-based human–computer interaction applications by fusion of depth and colour video streams," *IET Computer Vision*, vol. 8, no. 6, pp. 568–578, 2014.

[86] Y. Chai, S. Shin, K. Chang, and T. Kim, "Real-time user interface using particle filter with integral histogram," *IEEE Transactions on Consumer Electronics*, vol. 56, no. 2, 2010.

[87] S. M. Nadgeri, S. Sawarkar, and A. D. Gawande, "Hand gesture recognition using camshift algorithm," in *Emerging Trends in Engineering and Technology (ICETET), 2010 3rd International Conference on.* IEEE, 2010, pp. 37–41.

[88] X. Wang, M. Xia, H. Cai, Y. Gao, and C. Cattani, "Hidden-markov-models-based dynamic hand gesture recognition," *Mathematical Problems in Engineering*, vol. 2012, 2012.

[89] J. Han, G. Awad, and A. Sutherland, "Automatic skin segmentation and tracking in sign language recognition," *IET Computer Vision*, vol. 3, no. 1, pp. 24–35, 2009.

[90] C. Keskin, F. Kıraç, Y. E. Kara, and L. Akarun, "Real time hand pose estimation using depth sensors," in *Consumer depth cameras for computer vision.* Springer, 2013, pp. 119–137.

[91] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake, "Real-time human pose recognition in parts from single depth images," in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on.* Ieee, 2011, pp. 1297–1304.

[92] J. M. Rehg and T. Kanade, "Model-based tracking of self-occluding articulated objects," in *Computer Vision, 1995. Proceedings., Fifth International Conference on.* IEEE, 1995, pp. 612–617.

[93] L. Goncalves, E. Di Bernardo, E. Ursella, and P. Perona, "Monocular tracking of the human arm in 3d," 1995.

[94] A. D. Wilson and A. F. Bobick, "Learning visual behavior for gesture analysis," in *Computer Vision, 1995. Proceedings., International Symposium on.* IEEE, 1995, pp. 229–234.

[95] J. Yamato, J. Ohya, and K. Ishii, "Recognizing human action in time-sequential images using hidden markov model," in *Computer Vision and Pattern Recognition, 1992. Proceedings CVPR'92., 1992 IEEE Computer Society Conference on.* IEEE, 1992, pp. 379–385.

[96] K. Akita, "Image sequence analysis of real world human motion," *Pattern recognition*, vol. 17, no. 1, pp. 73–83, 1984.

[97] H.-K. Lee and J.-H. Kim, "An hmm-based threshold model approach for gesture recognition," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 21, no. 10, pp. 961–973, 1999.

[98] M. K. Bhuyan, D. A. Kumar, K. F. MacDorman, and Y. Iwahori, "A novel set of features for continuous hand gesture recognition," *Journal on Multimodal User Interfaces*, vol. 8, no. 4, pp. 333–343, 2014.

[99] V. I. Pavlovic, R. Sharma, and T. S. Huang, "Visual interpretation of hand gestures for human-computer interaction: A review," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 19, no. 7, pp. 677–695, 1997.

[100] D. Sarma and M. Bhuyan, "Optical flow guided motion template for hand gesture recognition," in *2020 IEEE Applied Signal Processing Conference (ASPCON).* IEEE, 2020, pp. 262–266.

[101] B. D. Lucas, T. Kanade *et al.*, "An iterative image registration technique with an application to stereo vision," 1981.

[102] B. K. Horn and B. G. Schunck, "Determining optical flow," *Artificial intelligence*, vol. 17, no. 1-3, pp. 185–203, 1981.

[103] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert, "High accuracy optical flow estimation based on a theory for warping," in *European conference on computer vision.* Springer, 2004, pp. 25–36.

[104] T. Brox and J. Malik, "Large displacement optical flow: descriptor matching in variational motion estimation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 33, no. 3, pp. 500–513, 2011.

[105] G. Farnebäck, "Two-frame motion estimation based on polynomial expansion," in *Scandinavian conference on Image analysis.* Springer, 2003, pp. 363–370.

[106] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld, "Learning realistic human actions from movies," in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*. IEEE, 2008, pp. 1–8.

[107] N. Dalal, B. Triggs, and C. Schmid, "Human detection using oriented histograms of flow and appearance," in *European conference on computer vision*. Springer, 2006, pp. 428–441.

[108] Y. Yacoob and L. S. Davis, "Recognizing human facial expressions from long image sequences using optical flow," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 18, no. 6, pp. 636–642, 1996.

[109] L. Wixson, "Detecting salient motion by accumulating directionally-consistent flow," *IEEE transactions on pattern analysis and machine intelligence*, vol. 22, no. 8, pp. 774–780, 2000.

[110] U. Mahbub, H. Imtiaz, and M. A. R. Ahad, "An optical flow based approach for action recognition," in *14th International Conference on Computer and Information Technology (ICCIT 2011)*. IEEE, 2011, pp. 646–651.

[111] M. A. R. Ahad, J. K. Tan, H. Kim, and S. Ishikawa, "Motion history image: its variants and applications," *Machine Vision and Applications*, vol. 23, no. 2, pp. 255–281, 2012.

[112] M.-K. Hu, "Visual pattern recognition by moment invariants," *IRE transactions on information theory*, vol. 8, no. 2, pp. 179–187, 1962.

[113] B. Fernando, E. Gavves, J. Oramas, A. Ghodrati, and T. Tuytelaars, "Rank pooling for action recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 4, pp. 773–787, 2017.

[114] H. Bilen, B. Fernando, E. Gavves, and A. Vedaldi, "Action recognition with dynamic image networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

[115] B. Fernando, P. Anderson, M. Hutter, and S. Gould, "Discriminative hierarchical rank pooling for activity recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 1924–1932.

[116] Z. Lan, M. Lin, X. Li, A. G. Hauptmann, and B. Raj, "Beyond gaussian pyramid: Multi-skip feature stacking for action recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 204–212.

[117] N. Srivastava, E. Mansimov, and R. Salakhudinov, "Unsupervised learning of video representations using lstms," in *International conference on machine learning*, 2015, pp. 843–852.

[118] L. Malagón-Borja and O. Fuentes, "Object detection using image reconstruction with pca," *Image and Vision Computing*, vol. 27, no. 1-2, pp. 2–9, 2009.

[119] H. Kim *et al.*, "Novel and efficient pedestrian detection using bidirectional pca," *Pattern Recognition*, vol. 46, no. 8, pp. 2220–2227, 2013.

[120] M. Arunraj, A. Srinivasan, and A. V. Juliet, "Online action recognition from rgb-d cameras based on reduced basis decomposition," *Journal of Real-Time Image Processing*, pp. 1–16, 2018.

[121] H.-S. Yoon, J. Soh, Y. J. Bae, and H. S. Yang, "Hand gesture recognition using combined features of location, angle and velocity," *Pattern recognition*, vol. 34, no. 7, pp. 1491–1501, 2001.

[122] Y. Kong, Z. Ding, J. Li, and Y. Fu, "Deeply learned view-invariant features for cross-view action recognition," *IEEE Transactions on Image Processing*, vol. 26, no. 6, pp. 3028–3037, 2017.

[123] M. de La Gorce and N. Paragios, "A variational approach to monocular hand-pose estimation," *Computer Vision and Image Understanding*, vol. 114, no. 3, pp. 363–372, 2010.

[124] P. R. Harding and T. Ellis, "Recognizing hand gesture using fourier descriptors," in *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, vol. 3. IEEE, 2004, pp. 286–289.

[125] I. Akhter, Y. Sheikh, S. Khan, and T. Kanade, "Trajectory space: A dual representation for non-rigid structure from motion," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 7, pp. 1442–1456, 2011.

[126] K.-C. Hung, "The generalized uniqueness wavelet descriptor for planar closed curves," *IEEE Transactions on image processing*, vol. 9, no. 5, pp. 834–845, 2000.

[127] X. Wu, X. Mao, L. Chen, Y. Xue, and A. Rovetta, "Point context: an effective shape descriptor for rst-invariant trajectory recognition," *Journal of Mathematical Imaging and Vision*, vol. 56, no. 3, pp. 441–454, 2016.

[128] N. H. Dardas and N. D. Georganas, "Real-time hand gesture detection and recognition using bag-of-features and support vector machine techniques," *IEEE Transactions on Instrumentation and measurement*, vol. 60, no. 11, pp. 3592–3607, 2011.

[129] R. Zhang, Y. Ming, and J. Sun, "Hand gesture recognition with surf-bof based on gray threshold segmentation," in *Signal Processing (ICSP), 2016 IEEE 13th International Conference on.* IEEE, 2016, pp. 118–122.

[130] P. Hall, B. U. Park, and R. J. Samworth, "Choice of neighbor order in nearest-neighbor classification," *The Annals of Statistics*, pp. 2135–2152, 2008.

[131] T. Marasović and V. Papić, "Feature weighted nearest neighbour classification for accelerometer-based gesture recognition," in *Software, Telecommunications and Computer Networks (SoftCOM), 2012 20th International Conference on.* IEEE, 2012, pp. 1–5.

[132] B. Gupta, P. Shukla, and A. Mittal, "K-nearest correlated neighbor classification for indian sign language gesture recognition using feature fusion," in *Computer Communication and Informatics (ICCCI), 2016 International Conference on.* IEEE, 2016, pp. 1–5.

[133] L. Liu, J. Xing, H. Ai, and X. Ruan, "Hand posture recognition using finger geometric feature," in *Pattern Recognition (ICPR), 2012 21st International Conference on.* IEEE, 2012, pp. 565–568.

[134] K. O. Rodriguez and G. C. Chavez, "Finger spelling recognition from rgb-d information using kernel descriptor," in *Graphics, Patterns and Images (SIBGRAPI), 2013 26th SIBGRAPI-Conference on.* IEEE, 2013, pp. 1–7.

[135] J. Weston and C. Watkins, "Multi-class support vector machines," Citeseer, Tech. Rep., 1998.

[136] C.-W. Hsu and C.-J. Lin, "A comparison of methods for multiclass support vector machines," *IEEE transactions on Neural Networks*, vol. 13, no. 2, pp. 415–425, 2002.

[137] M. Murugeswari and S. Veluchamy, "Hand gesture recognition system for real-time application," in *Advanced Communication Control and Computing Technologies (ICACCCT), 2014 International Conference on.* IEEE, 2014, pp. 1220–1225.

[138] C. Nolker and H. Ritter, "Visual recognition of continuous hand postures," *IEEE Transactions on neural networks*, vol. 13, no. 4, pp. 983–994, 2002.

[139] O. Patsadu, C. Nukoolkit, and B. Watanapa, "Human gesture recognition using kinect camera," in *Computer Science and Software Engineering (JCSSE), 2012 International Joint Conference on.* IEEE, 2012, pp. 28–32.

[140] S. S. Rautaray and A. Agrawal, "Vision based hand gesture recognition for human computer interaction: a survey," *Artificial Intelligence Review*, vol. 43, no. 1, pp. 1–54, 2015.

[141] M.-H. Yang, N. Ahuja, and M. Tabb, "Extraction of 2d motion trajectories and its application to hand gesture recognition," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 24, no. 8, pp. 1061–1074, 2002.

[142] A. R. Várkonyi-Kóczy and B. Tusor, "Human–computer interaction for smart environment applications using fuzzy hand posture and gesture models," *IEEE Transactions on Instrumentation and Measurement*, vol. 60, no. 5, pp. 1505–1514, 2011.

[143] K. S. Patwardhan and S. D. Roy, "Hand gesture modelling and recognition involving changing shapes and trajectories, using a predictive eigentracker," *Pattern Recognition Letters*, vol. 28, no. 3, pp. 329–334, 2007.

[144] M. C. Shin, L. V. Tsap, and D. B. Goldgof, "Gesture recognition using bezier curves for visualization navigation from registered 3-d data," *Pattern Recognition*, vol. 37, no. 5, pp. 1011–1024, 2004.

[145] A. Shamaie and A. Sutherland, "Graph-based matching of occluded hand gestures," in *Applied Imagery Pattern Recognition Workshop, AIPR 2001 30th.* IEEE, 2001, pp. 67–73.

[146] S. M. A. Hussain and A. H.-u. Rashid, "User independent hand gesture recognition by accelerated dtw," in *Informatics, Electronics & Vision (ICIEV), 2012 International Conference on.* IEEE, 2012, pp. 1033–1037.

[147] L. R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[148] J. Kwon and F. C. Park, "Natural movement generation using hidden markov models and principal components," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 38, no. 5, pp. 1184–1194, 2008.

[149] P. Heracleous, N. Aboutabit, and D. Beautemps, "Lip shape and hand position fusion for automatic vowel recognition in cued speech for french," *IEEE Signal Processing Letters*, vol. 16, no. 5, pp. 339–342, 2009.

[150] Z. Ren, J. Yuan, J. Meng, and Z. Zhang, "Robust part-based hand gesture recognition using kinect sensor," *IEEE transactions on multimedia*, vol. 15, no. 5, pp. 1110–1120, 2013.

[151] M. Tang, "Recognizing hand gestures with microsoft's kinect," *Palo Alto: Department of Electrical Engineering of Stanford University:[sn]*, 2011.

[152] H. Regenbrecht, J. Collins, and S. Hoermann, "A leap-supported, hybrid ar interface approach," in *Proceedings of the 25th Australian Computer-Human Interaction Conference: Augmentation, Application, Innovation, Collaboration.* ACM, 2013, pp. 281–284.

[153] A. J. Porfirio, K. L. Wiggers, L. E. Oliveira, and D. Weingaertner, "Libras sign language hand configuration recognition based on 3d meshes," in *Systems, Man, and Cybernetics (SMC), 2013 IEEE International Conference on.* IEEE, 2013, pp. 1588–1593.

[154] D. Wu, F. Zhu, and L. Shao, "One shot learning gesture recognition from rgbd images," in *Computer Vision and Pattern Recognition Workshops (CVPRW), 2012 IEEE Computer Society Conference on.* IEEE, 2012, pp. 7–12.

[155] J. Wan, Q. Ruan, W. Li, and S. Deng, "One-shot learning gesture recognition from rgb-d data using bag of features," *The Journal of Machine Learning Research*, vol. 14, no. 1, pp. 2549–2582, 2013.

[156] J. Konecnỳ and M. Hagara, "One-shot-learning gesture recognition using hog-hof," *Journal of Machine Learning Research*, vol. 15, pp. 2513–2532, 2014.

[157] U. Mahbub, H. Imtiaz, T. Roy, M. S. Rahman, and M. A. R. Ahad, "A template matching approach of one-shot-learning gesture recognition," *Pattern Recognition Letters*, vol. 34, no. 15, pp. 1780–1788, 2013.

[158] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[159] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," in *Advances in neural information processing systems*, 2014, pp. 568–576.

[160] L. Pigou, S. Dieleman, P.-J. Kindermans, and B. Schrauwen, "Sign language recognition using convolutional neural networks," in *Workshop at the European Conference on Computer Vision.* Springer, 2014, pp. 572–578.

[161] P. Molchanov, S. Gupta, K. Kim, and J. Kautz, "Hand gesture recognition with 3d convolutional neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2015, pp. 1–7.

[162] D. Ciregan, U. Meier, and J. Schmidhuber, "Multi-column deep neural networks for image classification," in *Computer vision and pattern recognition (CVPR), 2012 IEEE conference on*. IEEE, 2012, pp. 3642–3649.

[163] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2014, pp. 1725–1732.

[164] N. Neverova, C. Wolf, G. Taylor, and F. Nebout, "Moddrop: adaptive multi-modal gesture recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 8, pp. 1692–1706, 2015.

[165] P. Molchanov, X. Yang, S. Gupta, K. Kim, S. Tyree, and J. Kautz, "Online detection and classification of dynamic hand gestures with recurrent 3d convolutional neural network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 4207–4215.

[166] X. Chai, Z. Liu, F. Yin, Z. Liu, and X. Chen, "Two streams recurrent neural networks for large-scale continuous gesture recognition," in *Pattern Recognition (ICPR), 2016 23rd International Conference on*. IEEE, 2016, pp. 31–36.

[167] S. Ji, W. Xu, M. Yang, and K. Yu, "3d convolutional neural networks for human action recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 1, pp. 221–231, 2012.

[168] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3d convolutional networks," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 4489–4497.

[169] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. Van Gool, "Temporal segment networks: Towards good practices for deep action recognition," in *European conference on computer vision*. Springer, 2016, pp. 20–36.

[170] V.-M. Khong and T.-H. Tran, "Improving human action recognition with two-stream 3d convolutional neural network," in *2018 1st International Conference on Multimedia Analysis and Pattern Recognition (MAPR)*. IEEE, 2018, pp. 1–6.

[171] Y. Zhu, Z. Lan, S. Newsam, and A. Hauptmann, "Hidden two-stream convolutional networks for action recognition," in *Asian Conference on Computer Vision*. Springer, 2018, pp. 363–378.

[172] M. Asadi-Aghbolaghi, A. Clapes, M. Bellantonio, H. J. Escalante, V. Ponce-López, X. Baró, I. Guyon, S. Kasaei, and S. Escalera, "A survey on deep learning based approaches for action and gesture recognition in image sequences," in *Automatic Face & Gesture Recognition (FG 2017), 2017 12th IEEE International Conference on*. IEEE, 2017, pp. 476–483.

[173] R. Zhao, H. Ali, and P. Van der Smagt, "Two-stream rnn/cnn for action recognition in 3d videos," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 4260–4267.

[174] F. A. Gers, N. N. Schraudolph, and J. Schmidhuber, "Learning precise timing with lstm recurrent networks," *Journal of machine learning research*, vol. 3, no. Aug, pp. 115–143, 2002.

[175] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell, "Long-term recurrent convolutional networks for visual recognition and description," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 2625–2634.

[176] E. Tsironi, P. Barros, and S. Wermter, "Gesture recognition with a convolutional long short-term memory recurrent neural network," *Bruges, Belgium*, vol. 2, 2016.

[177] J. Liu, A. Shahroudy, D. Xu, and G. Wang, "Spatio-temporal lstm with trust gates for 3d human action recognition," in *European Conference on Computer Vision*. Springer, 2016, pp. 816–833.

[178] K. Fukushima and S. Miyake, "Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition," in *Competition and cooperation in neural nets*. Springer, 1982, pp. 267–285.

[179] C. Yang, D. K. Han, and H. Ko, "Continuous hand gesture recognition based on trajectory shape information," *Pattern Recognition Letters*, vol. 99, pp. 39–47, 2017.

[180] E. Zhang, B. Xue, F. Cao, J. Duan, G. Lin, and Y. Lei, "Fusion of 2d cnn and 3d densenet for dynamic gesture recognition," *Electronics*, vol. 8, no. 12, p. 1511, 2019.

[181] D. Sarma, V. Kavyasree, and M. Bhuyan, "Two-stream fusion model using 3d-cnn and 2d-cnn via video-frames and optical flow motion templates for hand gesture recognition," *Innovations in Systems and Software Engineering*, pp. 1–14, 2022.

[182] A. Jain, A. R. Zamir, S. Savarese, and A. Saxena, "Structural-rnn: Deep learning on spatio-temporal graphs," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 5308–5317.

[183] S. Shin and W.-Y. Kim, "Skeleton-based dynamic hand gesture recognition using a part-based gru-rnn for gesture-based interface," *IEEE Access*, vol. 8, pp. 50 236–50 243, 2020.

[184] B. Muhammad and S. A. R. Abu-Bakar, "A hybrid skin color detection using hsv and ycgcr color space for face detection," in *Proc. IEEE Int. Conf. Signal and Image Process. App. (ICSIPA)*, Oct 2015, pp. 95–98.

[185] S. Tsekeridou and I. Pitas, "Facial feature extraction in frontal views using biometric analogies," in *Proc. 9th European Signal Process. Conf (EUSIPCO 1998)*, Sept 1998, pp. 1–4.

[186] A. Nikolaidis and I. Pitas, "Robust watermarking of facial images based on salient geometric pattern matching," *IEEE Trans. Multimedia*, vol. 2, no. 3, pp. 172–184, Sep 2000.

[187] G. Kukharev and A. Nowosielski, "Fast and efficient algorithm for face detection in colour images," *MG and V*, vol. 13, no. 4, pp. 377–399, 2004.

[188] K. B. Shaik, P. Ganesan, V. Kalist, B. Sathish, and J. M. M. Jenitha, "Comparative study of skin color detection and segmentation in HSV and YCbCr color space," *Procedia Computer Science*, vol. 57, no. Supplement C, pp. 41–48, 2015.

[189] D. Comaniciu, V. Ramesh, and P. Meer, "Kernel-based object tracking," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 25, no. 5, pp. 564–577, 2003.

[190] J.-M. Guo, Y.-F. Liu, C.-H. Chang, and H.-S. Nguyen, "Improved hand tracking system," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, no. 5, pp. 693–701, 2011.

[191] G. Bradsky, "Computer vision face tracking as a component of a perceptual user interface," in *Workshop on Appl. of Comp. Vision*, 1998, pp. 214–219.

[192] J. Shi and C. Tomasi, "Good features to track," in *1994 Proceedings of IEEE conference on computer vision and pattern recognition*. IEEE, 1994, pp. 593–600.

[193] M. Kolsch and M. Turk, "Fast 2d hand tracking with flocks of features and multi-cue integration," in *2004 Conference on Computer Vision and Pattern Recognition Workshop*. IEEE, 2004, pp. 158–158.

[194] M. S. M. Asaari, B. A. Rosdi, and S. A. Suandi, "Adaptive kalman filter incorporated eigenhand (akfie) for real-time hand tracking system," *Multimedia Tools and Applications*, vol. 74, no. 21, pp. 9231–9257, 2015.

[195] N. Dardas, Q. Chen, N. D. Georganas, and E. M. Petriu, "Hand gesture recognition using bag-of-features and multi-class support vector machine," in *Haptic Audio-Visual Environments and Games (HAVE), 2010 IEEE International Symposium on*. IEEE, 2010, pp. 1–5.

[196] P. Viola and M. J. Jones, "Robust real-time face detection," *International journal of computer vision*, vol. 57, no. 2, pp. 137–154, 2004.

[197] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE transactions on systems, man, and cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.

[198] B.-F. Wu, C.-C. Kao, C.-L. Jen, Y.-F. Li, Y.-H. Chen, and J.-H. Juang, "A relative-discriminative-histogram-of-oriented-gradients-based particle filter approach to vehicle occlusion handling and tracking," *IEEE Transactions on Industrial Electronics*, vol. 61, no. 8, pp. 4228–4237, 2013.

[199] G. Cohen, S. Afshar, J. Tapson, and A. van Schaik, "Emnist: an extension of mnist to handwritten letters," *arXiv preprint arXiv:1702.05373*, 2017.

[200] C. J. Burges, "A tutorial on support vector machines for pattern recognition," *Data mining and knowledge discovery*, vol. 2, no. 2, pp. 121–167, 1998.

[201] J. Singha, A. Roy, and R. H. Laskar, "Dynamic hand gesture recognition using vision-based approach for human–computer interaction," *Neural Computing and Applications*, vol. 29, no. 4, pp. 1129–1141, 2018.

[202] H. Xu, L. Li, M. Fang, and F. Zhang, "Movement human actions recognition based on machine learning," *International Journal of Online and Biomedical Engineering (iJOE)*, vol. 14, no. 04, pp. 193–210, 2018.

[203] D. Sarma and M. Bhuyan, "Hand detection by two-level segmentation with double-tracking and gesture recognition using deep-features," *Sensing and Imaging*, vol. 23, no. 1, pp. 1–29, 2022.

[204] V. Kavyasree, D. Sarma, P. Gupta, and M. Bhuyan, "Deep network-based hand gesture recognition using optical flow guided trajectory images," in *2020 IEEE Applied Signal Processing Conference (ASPCON)*. IEEE, 2020, pp. 252–256.

[205] A. Bruhn, J. Weickert, and C. Schnörr, "Combining the advantages of local and global optic flow methods," in *Joint Pattern Recognition Symposium*. Springer, 2002, pp. 454–462.

[206] X. Fan and T. Tjahjadi, "A dynamic framework based on local zernike moment and motion history image for facial expression recognition," *Pattern recognition*, vol. 64, pp. 399–406, 2017.

[207] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE transactions on image processing*, vol. 13, no. 4, pp. 600–612, 2004.

[208] D. Frolova, H. Stern, and S. Berman, "Most probable longest common subsequence for recognition of gesture character input," *IEEE transactions on cybernetics*, vol. 43, no. 3, pp. 871–880, 2013.

[209] S. Poularakis and I. Katsavounidis, "Low-complexity hand gesture recognition system for continuous streams of digits and letters," *IEEE transactions on cybernetics*, vol. 46, no. 9, pp. 2094–2108, 2015.

[210] D. Sarma and M. Bhuyan, "Methods, databases and recent advancement of vision-based hand gesture recognition for hci systems: A review," *SN Computer Science*, vol. 2, no. 6, pp. 1–40, 2021.

[211] H. P. J. Dutta, D. Sarma, M. K. Bhuyan, and R. H. Laskar, "Semantic segmentation based hand gesture recognition using deep neural networks," in *2020 National Conference on Communications (NCC)*. IEEE, 2020, pp. 1–6.

[212] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

[213] D. Sarma, V. Kavyasree, and M. K. Bhuyan, "Two-stream fusion model for dynamic hand gesture recognition using 3d-cnn and 2d-cnn optical flow guided motion template," *arXiv preprint arXiv:2007.08847*, 2020.

[214] G. Benitez-Garcia, J. Olivares-Mercado, G. Sanchez-Perez, and K. Yanai, "Ipn hand: A video dataset and benchmark for real-time continuous hand gesture recognition," in *2020 25th International Conference on Pattern Recognition (ICPR)*. IEEE, 2021, pp. 4340–4347.

[215] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "Cbam: Convolutional block attention module," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 3–19.

[216] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *Advances in neural information processing systems*, vol. 28, pp. 91–99, 2015.

[217] J. Dai, Y. Li, K. He, and J. Sun, "R-fcn: Object detection via region-based fully convolutional networks," in *Advances in neural information processing systems*, 2016, pp. 379–387.

[218] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961–2969.

[219] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.

[220] Z. Zhou, M. M. R. Siddiquee, N. Tajbakhsh, and J. Liang, "Unet++: A nested u-net architecture for medical image segmentation," in *Deep learning in medical image analysis and multimodal learning for clinical decision support*. Springer, 2018, pp. 3–11.

[221] H. Huang, L. Lin, R. Tong, H. Hu, Q. Zhang, Y. Iwamoto, X. Han, Y.-W. Chen, and J. Wu, "Unet 3+: A full-scale connected unet for medical image segmentation," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 1055–1059.

[222] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Semantic image segmentation with deep convolutional nets and fully connected crfs," *arXiv preprint arXiv:1412.7062*, 2014.

[223] ——, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 4, pp. 834–848, 2017.

[224] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," *arXiv preprint arXiv:1706.05587*, 2017.

[225] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 801–818.

[226] N. Souly, C. Spampinato, and M. Shah, "Semi supervised semantic segmentation using generative adversarial network," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 5688–5696.

[227] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[228] X. Zhang, X. Zhu, N. Zhang, P. Li, L. Wang *et al.*, "Seggan: Semantic segmentation with generative adversarial network," in *2018 IEEE Fourth International Conference on Multimedia Big Data (BigMM)*. IEEE, 2018, pp. 1–5.

[229] M. Jaderberg, K. Simonyan, A. Zisserman *et al.*, "Spatial transformer networks," *Advances in neural information processing systems*, vol. 28, pp. 2017–2025, 2015.

[230] C. Lea, M. D. Flynn, R. Vidal, A. Reiter, and G. D. Hager, "Temporal convolutional networks for action segmentation and detection," in *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 156–165.

[231] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7132–7141.

[232] C. Li, Y. Tan, W. Chen, X. Luo, Y. He, Y. Gao, and F. Li, "Anu-net: Attention-based nested u-net to exploit full resolution features for medical image segmentation," *Computers & Graphics*, vol. 90, pp. 11–20, 2020.

[233] F. Wang, M. Jiang, C. Qian, S. Yang, C. Li, H. Zhang, X. Wang, and X. Tang, "Residual attention network for image classification," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 3156–3164.

[234] J. Fu, J. Liu, H. Tian, Y. Li, Y. Bao, Z. Fang, and H. Lu, "Dual attention network for scene segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3146–3154.

[235] A. Vaswani, P. Ramachandran, A. Srinivas, N. Parmar, B. Hechtman, and J. Shlens, "Scaling local self-attention for parameter efficient visual backbones," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 12 894–12 904.

[236] L. Chen, H. Zhang, J. Xiao, L. Nie, J. Shao, W. Liu, and T.-S. Chua, "Sca-cnn: Spatial and channel-wise attention in convolutional networks for image captioning," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 5659–5667.

[237] H. Li, P. Xiong, J. An, and L. Wang, "Pyramid attention network for semantic segmentation," *arXiv preprint arXiv:1805.10180*, 2018.

[238] C. Yu, J. Wang, C. Peng, C. Gao, G. Yu, and N. Sang, "Learning a discriminative feature network for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 1857–1866.

[239] P. Narayana, R. Beveridge, and B. A. Draper, "Gesture recognition: Focus on the hands," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 5235–5244.

[240] S. Narasimhaswamy, Z. Wei, Y. Wang, J. Zhang, and M. Hoai, "Contextual attention for hand detection in the wild," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 9567–9576.

[241] N. Dhingra and A. Kunz, "Res3atn-deep 3d residual attention network for hand gesture recognition in videos," in *2019 International Conference on 3D Vision (3DV)*. IEEE, 2019, pp. 491–501.

[242] A. D'Eusanio, A. Simoni, S. Pini, G. Borghi, R. Vezzani, and R. Cucchiara, "A transformer-based network for dynamic hand gesture recognition," in *2020 International Conference on 3D Vision (3DV)*. IEEE, 2020, pp. 623–632.

[243] W. Abdul, M. Alsulaiman, S. U. Amin, M. Faisal, G. Muhammad, F. R. Albogamy, M. A. Bencherif, and H. Ghaleb, "Intelligent real-time arabic sign language classification using attention-based inception and bilstm," *Computers & Electrical Engineering*, vol. 95, p. 107395, 2021.

[244] X. Li, Y. Hou, P. Wang, Z. Gao, M. Xu, and W. Li, "Trear: Transformer-based rgb-d egocentric action recognition," *IEEE Transactions on Cognitive and Developmental Systems*, 2021.

[245] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[246] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1026–1034.

[247] F. Chollet, *Deep Learning with Python*. Manning Publication Co., Nov. 2017.

[248] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[249] S. Sharma and K. Kumar, "Asl-3dcnn: American sign language recognition technique using 3-d convolutional neural networks," *Multimedia Tools and Applications*, vol. 80, no. 17, pp. 26 319–26 331, 2021.

[250] P. P. Kumar, P. Vadakkepat, and A. P. Loh, "Hand posture and face recognition using a fuzzy-rough approach," *International Journal of Humanoid Robotics*, vol. 7, no. 03, pp. 331–356, 2010.

[251] A. Betancourt, P. Morerio, E. I. Barakova, L. Marcenaro, M. Rauterberg, and C. S. Regazzoni, "A dynamic approach and a new dataset for hand-detection in first person vision," in *International conference on Computer Analysis of Images and Patterns*. Springer, 2015, pp. 274–287.

[252] M. Matilainen, P. Sangi, J. Holappa, and O. Silvén, "Ouhands database for hand detection and pose recognition," in *Image Processing Theory Tools and Applications (IPTA), 2016 6th International Conference on*. IEEE, 2016, pp. 1–5.

[253] T.-K. Kim, S.-F. Wong, and R. Cipolla, "Tensor canonical correlation analysis for action classification," in *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*. IEEE, 2007, pp. 1–8.

[254] X. Shen, G. Hua, L. Williams, and Y. Wu, "Dynamic hand gesture recognition: An exemplar-based approach from motion divergence fields," *Image and Vision Computing*, vol. 30, no. 3, pp. 227–235, 2012.

[255] J. Triesch and C. Von Der Malsburg, "A system for person-independent hand posture recognition against complex backgrounds," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 12, pp. 1449–1453, 2001.

[256] M. Holte and M. Stórring, "Documentation of pointing and command gestures under mixed illumination conditions: video sequence database," 2004.

[257] A. Just, O. Bernier, and S. Marcel, "Hmm and iohmm for the recognition of mono-and bi-manual 3d hand gestures," IDIAP, Tech. Rep., 2004.

[258] Z. Lin, Z. Jiang, and L. S. Davis, "Recognizing actions by shape-motion prototype trees," in *Computer Vision, 2009 IEEE 12th International Conference on*. IEEE, 2009, pp. 444–451.

[259] F. Dadgostar, A. L. Barczak, and A. Sarrafzadeh, "A color hand gesture database for evaluating and improving algorithms on hand gesture and posture recognition," 2005.

[260] S. Marcel and A. Just, "Idiap two handed gesture dataset," *IDIAP Research Institute, Switzerland*, 2005.

[261] H. Gunes and M. Piccardi, "A bimodal face and body gesture database for automatic analysis of human nonverbal affective behavior," in *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on*, vol. 1. IEEE, 2006, pp. 1148–1153.

[262] M. S. M. Asaari, B. A. Rosdi, and S. A. Suandi, "Intelligent biometric group hand tracking (ibght) database for visual hand tracking research and development," *Multimedia tools and applications*, vol. 70, no. 3, pp. 1869–1898, 2014.

[263] Z. Ren, J. Yuan, and Z. Zhang, "Robust hand gesture recognition based on finger-earth mover's distance with a commodity depth camera," in *Proceedings of the 19th ACM international conference on Multimedia*. ACM, 2011, pp. 1093–1096.

[264] X. Suau, M. Alcoverro, A. López-Méndez, J. Ruiz-Hidalgo, and J. R. Casas, "Real-time fingertip localization conditioned on hand gesture classification," *Image and Vision Computing*, vol. 32, no. 8, pp. 522–532, 2014.

[265] J. Tompson, M. Stein, Y. Lecun, and K. Perlin, "Real-time continuous pose recovery of human hands using convolutional networks," *ACM Transactions on Graphics (ToG)*, vol. 33, no. 5, p. 169, 2014.

[266] E. Kollorz, J. Penne, J. Hornegger, and A. Barke, "Gesture recognition with a time-of-flight camera," *International Journal of Intelligent Systems Technologies and Applications*, vol. 5, no. 3-4, pp. 334–343, 2008.

[267] I. Guyon, V. Athitsos, P. Jangyodsuk, B. Hamner, and H. J. Escalante, "Chalearn gesture challenge: Design and first results," in *Computer Vision and Pattern Recognition Workshops (CVPRW), 2012 IEEE Computer Society Conference on*. IEEE, 2012, pp. 1–6.

[268] S. Fothergill, H. Mentis, P. Kohli, and S. Nowozin, "Instructing people for training gestural interactive systems," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 2012, pp. 1737–1746.

[269] S. Escalera, J. Gonzàlez, X. Baró, M. Reyes, O. Lopes, I. Guyon, V. Athitsos, and H. Escalante, "Multi-modal gesture recognition challenge 2013: Dataset and results," in *Proceedings of the 15th ACM on International conference on multimodal interaction*. ACM, 2013, pp. 445–452.

[270] Y. Song, D. Demirdjian, and R. Davis, "Tracking body and hands for gesture recognition: Natops aircraft handling signals database," in *Automatic Face & Gesture Recognition and Workshops (FG 2011), 2011 IEEE International Conference on*. IEEE, 2011, pp. 500–506.

[271] S. Ruffieux, D. Lalanne, and E. Mugellini, "Chairgest: a challenge for multimodal mid-air gesture recognition for close hci," in *Proceedings of the 15th ACM on International conference on multimodal interaction*. ACM, 2013, pp. 483–488.

[272] L. Liu and L. Shao, "Learning discriminative representations from rgb-d video data." in *IJCAI*, vol. 1, 2013, p. 3.

[273] B.-W. Hwang, S. Kim, and S.-W. Lee, "A full-body gesture database for automatic gesture recognition," in *Automatic Face and Gesture Recognition, 2006. FGR 2006. 7th International Conference on*. IEEE, 2006, pp. 243–248.

[274] M. Chen, G. AlRegib, and B.-H. Juang, "6dmg: A new 6d motion gesture database," in *Proceedings of the 3rd Multimedia Systems Conference*. ACM, 2012, pp. 83–88.

[275] G. Marin, F. Dominio, and P. Zanuttigh, "Hand gesture recognition with leap motion and kinect devices," in *Image Processing (ICIP), 2014 IEEE International Conference on*. IEEE, 2014, pp. 1565–1569.

[276] L. Minto and P. Zanuttigh, "Exploiting silhouette descriptors and synthetic data for hand gesture recognition," 2015.

[277] A. I. Maqueda, C. R. del Blanco, F. Jaureguizar, and N. García, "Human–computer interaction based on visual hand-gesture recognition using volumetric spatiograms of local binary patterns," *Computer Vision and Image Understanding*, vol. 141, pp. 126–137, 2015.

[278] C. Wang, Z. Liu, and S.-C. Chan, "Superpixel-based hand gesture recognition with kinect depth camera," *IEEE transactions on multimedia*, vol. 17, no. 1, pp. 29–39, 2015.

[279] Q. De Smedt, H. Wannous, and J.-P. Vandeborre, "Skeleton-based dynamic hand gesture recognition," in *Computer Vision and Pattern Recognition Workshops (CVPRW), 2016 IEEE Conference on.* IEEE, 2016, pp. 1206–1214.

[280] A. Amir, B. Taba, D. Berg, T. Melano, J. McKinstry, C. Di Nolfo, T. Nayak, A. Andreopoulos, G. Garreau, M. Mendoza *et al.*, "A low power, fully event-based gesture recognition system," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 7243–7252.

[281] S. Yuan, Q. Ye, B. Stenger, S. Jain, and T.-K. Kim, "Bighand2. 2m benchmark: Hand pose dataset and state of the art analysis," in *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on.* IEEE, 2017, pp. 2605–2613.

[282] C. Cao, Y. Zhang, Y. Wu, H. Lu, and J. Cheng, "Egocentric gesture recognition using recurrent 3d convolutional neural networks with spatiotemporal transformer modules," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 3763–3771.

[283] E. Ohn-Bar and M. M. Trivedi, "Hand gesture recognition in real time for automotive interfaces: A multimodal vision-based approach and evaluations," *IEEE transactions on intelligent transportation systems*, vol. 15, no. 6, pp. 2368–2377, 2014.

[284] M. Kawulok, J. Kawulok, and J. Nalepa, "Spatial-based skin detection using discriminative skin-presence features," *Pattern Recognition Letters*, vol. 41, pp. 3–13, 2014.

[285] N. Pugeault and R. Bowden, "Spelling it out: Real-time asl fingerspelling recognition," in *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on.* IEEE, 2011, pp. 1114–1119.

[286] A. Barczak, N. Reyes, M. Abastillas, A. Piccio, and T. Susnjak, "A new 2d static hand gesture colour image dataset for asl gestures," 2011.

[287] R. Wilbur and A. C. Kak, "Purdue rvl-slll american sign language database," 2006.

[288] P. Dreuw, D. Rybach, T. Deselaers, M. Zahedi, and H. Ney, "Speech recognition techniques for a sign language recognition system," in *Eighth Annual Conference of the International Speech Communication Association*, 2007.

[289] P. Dreuw, C. Neidle, V. Athitsos, S. Sclaroff, and H. Ney, "Benchmark databases for video-based automatic sign language recognition." in *LREC*, 2008.

[290] W. Li, Z. Zhang, and Z. Liu, "Action recognition based on a bag of 3d points," in *Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on*. IEEE, 2010, pp. 9–14.

List of Publications

Journal Publications:

1. **D. Sarma** and M.K. Bhuyan, "**Hand Detection by Two-Level Segmentation with Double-Tracking and Gesture Recognition using Deep-Features**," **Sensing and Imaging, Springer**, vol. 23 (1), pp. 1-29, 2022.

2. **D. Sarma**, V. Kavyashree and M.K. Bhuyan, "**Two-Stream Fusion Model using 3D-CNN and 2D-CNN via Video-frames and Optical Flow Motion Templates for Hand Gesture Recognition**," **Innovations in Systems and Software Engineering, Springer**, 1-14, 2022.

3. **D. Sarma** and M.K. Bhuyan, "**Methods, Databases and Recent Advancement of Vision-based Hand Gesture Recognition Systems for HCI: A Review**," **SN Computer Science, Springer**, vol. 2 (6), pp. 1-40, 2021.

4. B. K. Chakraborty, **D. Sarma**, M.K. Bhuyan, and K. F. MacDorman, "**Review of constraints on vision-based gesture recognition for human–computer interaction**," **IET Computer Vision**, vol. 12 (1), pp. 3-15, 2018.

Journals (Under Review/Communicated):

1. **D. Sarma**, H. P. J. Dutta, M.K. Bhuyan, and R. H. Laskar: "**Attention-based Hand Semantic Segmentation and Gesture Recognition using Deep Networks**," (Under Review in **Evolving Systems, Springer**).

Conference Publications:

1. **D. Sarma** and M.K. Bhuyan, "**Hand Gesture Recognition using Deep Network through Trajctory-to-Contour based Images**," In Proceedings of the IEEE India Council International Conference (INDICON), pp. 1-6, 2018.

2. H. P. J. Dutta , **D. Sarma**, M.K. Bhuyan, and R. H. Laskar, "**Semantic Segmentation based Hand Gesture Recognition using Deep Neural Networks**," In Proceedings of the IEEE National Conference on Communications (NCC), pp. 1-6, 2020.

3. **D. Sarma** and M.K. Bhuyan, "**Optical Flow guided Motion Template for Hand Gesture Recognition**," In Proceedings of the IEEE Applied Signal Processing Conference (ASPCON), pp. 262-266, 2020.

4. V. Kavyashree, **D. Sarma** and M.K. Bhuyan, "**Deep Network-based Hand Gesture Recognition using Optical Flow guided Trajectory Images**," In Proceedings of the IEEE Applied Signal Processing Conference (ASPCON), pp. 252-256, 2020.

5. **D. Sarma** and M.K. Bhuyan, "**A Comparison of Deep Network with SVM Classifier for Static Hand Gesture Recognition**," In Proceedings of International Conference on Advances in Communication Technology, Computing and Engineering (ICACTCE), 2021.

6. **D. Sarma**, T. Barman, M.K. Bhuyan and Y. Iwahari, "**Motion-based Representations for Trajectory-based Hand Gestures: A Brief Overview**," In Proceedings of International Conference on Data Electronics and Computing (ICDEC), 2022.

Table 6.1: Summary of hand gesture databases with description

Sl. No.	Dataset	Contents	Description
RGB/Grayscale dataset			
1	NUS hand posture dataset-I, 2010 [250]	10 classes, 1 subject, 240 samples	Both colour and gray scale
2	NUS hand posture dataset-II, 2012 [16]	10 classes, 40 subjects, 2750 samples	Complex natural background
3	UNIGEhands Dataset, 2015 [251]	37.21 and 37.63 minutes of positives and negative video sequencs	Egocentric videos in 5 natural locations (Office, Bar, Kitchen, Bench, Street)
4	OUHANDS hand gesture dataset, 2016 [252]	2150 training and 1000 test images	Different background, contains body gesture, collected by Intel RealSense
5	Cambridge hand gesture dataset, 2007 [253]	9 classes, 2 subjects, 900 image sequences	Different illumination conditions
6	Gesture dataset by Shen *et al.* [254], 2012	10 classes, 15 subjects, 1050 samples	Different poses of thumb, fist, all fingers extended
7	Sebastien Marcel hand posture and gesture datasets, 2001 [255]	Three static datasets, with 10 (gray scale), 12 (color), and 6 (gray scale) classes. One dynamic dataset with 4 classes	Both simple and complex background
8	Aalborg Video Database, 2004 [256]	9 static and 4 dynamic classes	Hand gestures over a wooden table
9	Sebastien Marcel interact play database, 2004 [257]	16 classes, 22 subjects, 50 samples/ subject	Single and both hand dataset
10	Gesture dataset by Yoon *et al.*, 2001 [121]	48 classes, 20 subjects, 9600 samples	Alphabetical gestures containing sequences of xy coordinates
11	Keck gesture dataset, 2009 [258]	Keck gesture dataset, 2009	Military signals with training set in simple background and testing set in complex background
12	Massey gesture dataset, 2005 [259]	6 classes, 5 subjects, about 1500 frames	Different image frames of gestures in different illumination
13	IDIAP two-handed gesture dataset, 2005 [260]	7 classes, 7 subjects	Special colour-glove to differentiate between right and left hand
14	FABO gesture dataset, 2006 [261]	21 classes divided into two sets	Face and body gesture dataset in fixed background
15	IBGHT dataset, 2015 [262]	36 classes, 60 video sequences	0-9 numeric and A-Z alphabatic colour dataset
16	10 Palm Graffiti Digits dataset, 2009 [10]	10 classes, 30 examples per class	0-9 digits in continuous stream, coloured glove in training set, both easy and hard test set
17	NITS hand gesture dataset, 2015 [9]	40 classes, 20 subjects, divided into 7 sets	Gestures collected in lab environment with coloured fingertip
18	The 20BN-jester dataset, 2019 [255]	148,092 videos in total: 118,562 for training, 14,787 for validation and 14,743 for testing	Densely-labeled video clips that show humans performing predefined hand gestures in front of a laptop camera or webcam

Appendix A: List of Databases with Brief Description

Sl. No.	Dataset	Contents	Description
RGB-D dataset			
19	NTU posture dataset by Ren *et al.*, 2011 [263]	10 classes, 10 subjects, 1000 samples	Colour as well as depth maps, cluttered background, recorded with Kinect
20	ColorTip dataset, 2013 [264]	7 subjects, 9 classes, 7 training sequences of between 600 and 2000 depth frames	Fingertips are covered with coloured glove for automatic annotation
21	NYU hand pose dataset, 2014 [265]	72,757 and 8252 frames in training and test sets	2 users, data from 3 Kinects (frontal and 2 sides)
22	General-HANDS data-set, 2014	22 sequences	Different view-points, scales, poses, and occlusions
23	VPU Hand Gesture dataset (HGds), 2008 [266]	12 classes, 11 subjects	One static pose video per gesture (252 grayscale frames); collected by time-of-flight camera
24	ChaLearn gesture data, 2011 [267]	62,000 samples	Hand gestures including body gestures; recorded with Kinect
25	MSRC-12 Kinect gesture dataset, 2012 [268]	12 classes, 30 subjects, 6244 samples	Human movement including body gestures; recorded with Kinect
26	ChaLearn multi-modal gesture dataset, 2013 [269]	20 classes, 27 subjects, 13,858 samples	Including body gestures
27	NATOPS aircraft handling signals database, 2011 [270]	24 classes, 20 subjects, 9600 samples	Including body gestures
28	ChAirGest multi-modal dataset , 2013 [271]	10 classes, 10 subjects, 1200 samples	Recorded with Kinect and inertial motion units
29	Sheffield KInect Gesture (SKIG) dataset, 2013 [272]	10 classes, 6 subjects, 2160 samples	Two illumination condition, recorded with Kinect and RGB cameras
30	Full Body Gesture (FBG) database, 2006 [273]	14 normal gesture of daily life, 10 abnormal gesture classes, 20 subjects	Full body 3D dataset
31	10 3D digit dataset by Berman *et al.*, 2013 [208]	10 classes, 8 subjects	0-9 in continuous stream, dataset collected using PrimeSense 3D camera
32	6D Motion Gesture (6DMG) dataset, 2012 [274]	10 digit classes, 26 upper and lower alphabet classes each	Dataset is recorded by Wii device with trajectories in space, includes some body gestures also
33	Hand gesture datasets, University of Padova, 2014 [275]	10 ASL classes, 14 subjects	Dataset is collected with both leap motion controller and Kinect. First of its kind dataset collected by both.
34	Hand gesture datasets, University of Padova, 2015 [276]	Several static gestures	Collected with Senz3D device
35	Hand gesture datasets, University of Polytechnique, Madrid, 2015 [277]	10 classes, divided into 2 sets with 5 gestures each	Collected with Senz3D device

Sl. No.	Dataset	Contents	Description
36	SP-EMD dataset, 2015 [278]	10 gestures with 20 different poses, 5 subjects	In two different illumination, collected using Kinect
37	DHG-14/28, 2016 [279]	14 classes, 20 subjects	Gestures are collected using Kinect in two ways: using one finger and the whole hand
38	DVS128 gesture dataset, 2017 [280]	11 classes, 29 subjects	3 illumination condition, collected with DSV128
39	BigHand2.2M hand posture dataset, 2017 [281]	2.2 million depth maps	Collected with Intel RealSense, some are egocentric images
40	EgoGesture Dataset, 2017 [282]	83 classes, 50 subjects, 6 scenes, 24161 RGB-D video samples	First-person view gestures, collected using Intel RealSense SR300
41	VIVA dataset, 2014 [283]	19 classes, 8 subjects, 885 RGB-D video samples	Driver hand gestures in single scene, collected using Microsoft Kinect
42	NVIDIA Gesture (nvGesture) dataset, 2016 [165]	25 classes, 20 subjects, 1532 RGB-D video samples	Driver hand gestures collected using SoftKinetic DS325 and a top-mounted DUO 3D sensor to record a pair of stereo-IR streams
Sign language dataset			
43	Dataset by Kawulok *et al.*, 2014 [284]	32 classes, 18 subjects	Gestures from Polish Sign Language and American Sign Language (ASL)
44	ASL Finger Spelling Dataset, 2011 [285]	24 classes, 9 subjects, 65,000 samples	Alphabet depth dataset
45	Massey 2D Static ASL dataset, 2011 [286]	2425 gestures, 5 subjects	Colour ASL dataset
46	Purdue RVL-SLLL ASL Database, 2006 [287]	Different ASL gestures by 14 subjects	Alphanumeric dataset
47	RWTH-BOSTON-104 Database, 2007 [288]	104 signs, 201 videos, about 15000 image frames	Grayscale ASL dataset
48	RWTH-BOSTON-400, 2008 [289]	406 signs; extended upon 2007 dataset	Colour ASL dataset
49	MSR/MSRA Gesture 3D dataset, 2011 [290]	12 ASL gesture classes, 10 subjects	Hand tracking ASL dataset. Some are daily gestures
50	Kaggle Sign Language dataset, 2017	24 classes A-Z excluding J, Z and 10 classes of digits 0-9, mimics EMNIST	ASL image dataset

Table 6.2: Publicly available hand gesture databases with sources

Sl. No.	Dataset	Static(S) and/or Dynamic(D)	Source
RGB/Grayscale dataset			
1	NUS hand posture dataset-I, 2010	S	*http* : *//www.ece.nus.edu.sg/stfpage/elepv/NUS − HandSet/*
2	NUS hand posture dataset-II, 2012	S	*http* : *//www.ece.nus.edu.sg/stfpage/elepv/NUS − HandSet/*
3	UNIGEhands Dataset, 2015	S	*http* : *//alejobetancourt.com/resume/dataset?id* = 1
4	OUHANDS hand gesture dataset, 2016	S	*http* : *//www.ouhands.oulu.fi*
5	Cambridge hand gesture dataset, 2007	S and D	*http* : *//www.iis.ee.ic.ac.uk/˜tkkim/ges_db.htm*
6	Gesture dataset by Shen *et al.*, 2012	S and D	*http* : *//users.eecs.northwestern.edu/˜xsh835/GestureDataset.zip*
7	Sebastien Marcel hand posture and gesture datasets, 2001	S and D	*http* : *//www.idiap.ch/resource/gestures/*
8	Aalborg Video Database, 2004	S and D	*http* : *//www − prima.inrialpes.fr/FGnet/data/*03 − *Pointing/index.html*
9	Sebastien Marcel interact play database, 2004	D	*https* : *//www.idiap.ch/resource/interactplay/*
10	Gesture dataset by Yoon *et al.*, 2001	D	Available on e-mail request to yoonhs@etri.re.kr
11	Keck gesture dataset, 2009	D	*http* : *//users.umiacs.umd.edu/ zhuolin/Keckgesturedataset.html*
12	Massey gesture dataset, 2005	D	*https* : *//www.massey.ac.nz/˜albarcza/gesture*$_d$*ataset*2012*.html*
13	IDIAP two-Handed Gesture Dataset, 2005	D	*https* : *//www.idiap.ch/resource/twohanded/*
14	FABO gesture dataset, 2006	D	*https* : *//mmv.eecs.qmul.ac.uk/fabo/*
15	IBGHT dataset, 2015	D	*http* : *//ibg − usm.org*
16	10 Palm Graffiti Digits dataset, 2009	D	*http* : *//vlm*1*.uta.edu/ athitsos/projects/digits/*
17	NITS dataset, 2015	D	*https* : *//joyeetasingha*26*.wixsite.com/nits − database*
18	The 20BN-jester dataset, 2019	D	*https* : *//*20*bn.com/datasets/jester*
RGB-D dataset			
19	NTU posture dataset by Ren *et al.*, 2011	S	*http* : *//rose*1*.ntu.edu.sg/datasets/actionrecognition.asp*
20	ColorTip dataset, 2013	S	*https* : *//imatge.upc.edu/web/res/colortip*
21	NYU hand pose dataset, 2014	S	*https* : *//jonathantompson.github.io/NYU*$_H$*and*$_P$*ose*$_D$*ataset.htm*
22	General-HANDS data-set, 2014	S	*http* : *//homepages.inf.ed.ac.uk/rbf/CVonline/Imagedbase.htmgesture*
23	VPU Hand Gesture dataset (HGds), 2008	S	*http* : *//www − vpu.eps.uam.es/DS/HGds/*
24	ChaLearn gesture data, 2011	D	*http* : *//gesture.chalearn.org/data*
25	MSRC-12 Kinect gesture dataset, 2012	D	*http* : *//research.microsoft.com/en − us/um/cambridge/projects/msrc*12*/*

Appendix B: List of Databases with Source Links

Sl. No.	Dataset	Static(S) and/or Dynamic(D)	Source
26	ChaLearn multi-modal gesture data, 2013	D	*http://sunai.uoc.edu/chalearn/*
27	NATOPS aircraft handling signals database, 2011	D	*http://groups.csail.mit.edu/mug/natops/*
28	ChAirGest multi-modal dataset , 2013	D	*https://project.heia-fr.ch/chairgest/Pages/Download.aspx*
29	Sheffield KInect Gesture (SKIG) dataset, 2013	D	*http://lshao.staff.shef.ac.uk/data/SheffieldKinectGesture.htm*
30	Full Body Gesture (FBG) Database, 2006	D	*http://gesturedb.korea.ac.kr/*
31	10 3D digit dataset by Berman *et al.*, 2013	D	Available on e-mail request to sigalbe@bgu.ac.il
32	6D Motion Gesture (6DMG) dataset, 2012	D	*http://web.cs.wpi.edu/claypool/mmsys-dataset/2012/6dmg/*
33	Hand gesture datasets, University of Padova, 2014	S	*http://lttm.dei.unipd.it/downloads/gesture*
34	Hand gesture datasets, University of Padova, 2015	D	*http://lttm.dei.unipd.it/downloads/gesture*
35	Hand gesture datasets, University of Polytechnique, Madrid, 2015	S and D	*https://www.gti.ssr.upm.es/data/HandGesture_database.html*
36	SP-EMD dataset, 2015	D	*https://sites.google.com/site/spemdKinect*
37	DHG-14/28, 2016	D	*http://www-rech.telecom-lille.fr/DHGdataset/*
38	DVS128 gesture dataset, 2017	D	*http://research.ibm.com/dvsgesture/*
39	BigHand2.2M hand posture dataset, 2017	S	Available on e-mail request to hands.iccv17@outlook.com
40	EgoGesture dataset, 2017	D	*http://www.nlpr.ia.ac.cn/iva/yfzhang/datasets/egogesture.html*
41	VIVA datase, 2014	D	*http://www.site.uottawa.ca/research/viva/projects/hand_detection*
42	NVIDIA Gesture (nvGesture) dataset, 2016	D	*https://research.nvidia.com/publications*
Sign language dataset			
43	Dataset by Kawulok *et al.*, 2014	S	*http://sun.aei.polsl.pl/mkawulok/gestures/*
44	ASL Finger Spelling Dataset, 2011	S	*http://lifeprint.com*
45	Massey 2D Static ASL dataset, 2011	S	*http://imsc.massey.ac.nz/research/letters/*
46	Purdue RVL-SLLL ASL Database, 2006	D	Available on e-mail request to wilbur@purdue.edu
47	RWTH-BOSTON-104 Database, 2007	D	*http://www-i6.informatik.rwth-aachen.de/dreuw/database.php*
48	RWTH-BOSTON-400, 2008	D	*http://www-i6.informatik.rwth-aachen.de/aslr/*
49	MSR/MSRA Gesture 3D dataset, 2011	D	*https://www.microsoft.com/en-us/research/people/zliu/*
50	Kaggle Sign Language dataset, 2017	S	*https://www.kaggle.com/datamunge/sign-language-mnist*