GESTURE RECOGNITION

Aristotle University of Thessaloniki

ΑΡΙΣΤΟΤΕΛΕΙΟ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΘΕΣΣΑΛΟΝΙΚΗΣ

Papanastasiou Nektarios, Prof. Ioannis Pitas

January 15, 2021

Summary

In this article a presentation attempt is made for the gesture recognition. In the beginning there is an explaination about what gestures are and what are the different types of them. The sensors which are used for capturing gestures and create modalities of dataset are also discussed. The applications of the gesture recognition and the interaction between a human and a machine or a robot or a UAV is analysed and some of the latest and most important datasets for gesture recognition are later presented. Moreover, the algorithms that have been developed and the problems that have arisen over the gesture recognition are examined. There is a presentation of the structure of the 3D Convolutional Neural Networks and their layers and in addition to them the combination with the Long Short Term Memory models. Lastly, systems for spatiotemporal analysis of data are studied. CNN models for gesture recognition are compared and a proposed skeleton-based architecture is presented. The article has references from 16 scientific papers of Computer Vision and their Open Access versions, provided by the Computer Vision Foundation (CVF) and from 5 articles on websites on the internet.

Contents

Introduction

Gesture recognition is the interpretation of any gesture from a computer through mathematical algorithms so it can execute commands based on those gestures. The user performs gestures to control or interact with a machine without using any physical touch. It is a form of interaction between a human and a computer, a robot or a device and nowadays it is applied more often and it is widespread as a state of art technology. Computer vision is the scientific field that deals with the creation of algorithms for the interpretation of images and videos which are recorded by sensors, thus gesture recognition is one of its objects of study. In this article, an attempt will be made to analyze in detail the way in which gestures are recognized with computer vision algorithms.

Chapter 1

Gesture Types

Gestures can be extensive and comprehensive or small and contained and they can include any type of movement. Different types of gestures will be mentioned below categorized in terms of time, body parts and duration:

1.1 Gesture types (based on the timing)

- Online gestures: Instant interpretation of gestures. They are used to manipulate an object (scaling, rotation).
- Offline gestures: they are implemented after the procedure of the user interaction with the object.

1.2 Gesture types (based on the body parts)

- Hand Gestures: waving goodbye, showing points…
- Head Gestures: nodding, winking…
- Body Gestures: kicking, raise knee or elbow…

1.3 Gesture types (based on duration)

- Static Gesture: the pose in an instant time, a photo or an image of the position of the body part (showing the thumb…).

- Dynamic Gesture: changeable pose over a small period of time (waving palm. . .).

Chapter 2

Evolution of Gesture Evolution Devices

2.1 Data Gloves

Those gloves are worn on the hands and measure the position and the movement. They map every movement of phalange and wrist joints accurately with the sensors they carry. The advantage is that in this way, no data processing step for obtaining descriptors is needed, as in the case of the images from a camera. On the other hand, the disadvantage is that data gloves are expensive and not so comfortable for a user. In 1983, it was the first time that a data glove recognized hand position. There are two types of data gloves:

- The active gloves have a sensor or accelerometer and they are connected to the computer by cables or with wireless technology.
- The passive or non-invasive gloves have colour markers for image identification (2.1).

2.2 EMG-Electromyography Electrodes

Instead of data gloves, there are wearable bracelets with electromyography sensors which measure the electrical signals from the muscles. In this way there are no environmental interferences like in the voice interaction or in computer vision systems. Electromyography is based on the study of the neuromuscular

Figure 2.1: Passive gloves to help differentiate finger position [JES2020] .

system, so with EMG we can detect, analyze and process the electrical signals which are produced by muscles and nerves, by using electrodes. EMG has been used for medical diagnosis, control of prosthetics and for the rehabilitation after severe musculoskeletal injuries.

2.3 Ultrasound

There are two techniques for gesture capturing by ultrasound:

- sonomyography: uses ultrasound images to observe the muscles inside the body on a real-time. in contrast with EMG, sonomyography does not have difficulty of differentiating between individual muscles (for non-invasive techniques), and those lying deeper in the forearm so it captures the variety of wide hand movements.

- A technique based on Doppler Effect uses ultrasonic frequency signals where a device emits ultrasonic continuous tones. This can be done thanks to the fact that our tissues have different acoustic impedances, so when a sound wave passes from one to another with a different acoustic impedance, different amounts of energy are reflected, thus forming an ultrasound image.

2.4 WiFi

WiFi has the ability to perform gesture recognition even under non line of sight (NLOS). There are already researches based on the received signal strength indicator (RSSI), on the signal flight time indicator (ToF) or on the observation of the channel status information (CSI). However, these kinds of systems require either specialized devices, or it can be a modification by already-existing commercial devices, and some of them are excessively susceptible to interference. The most important advantage of this technique is that it is able to capture gestures for providing gesture recognition even without direct vision (NLOS) (2.2)

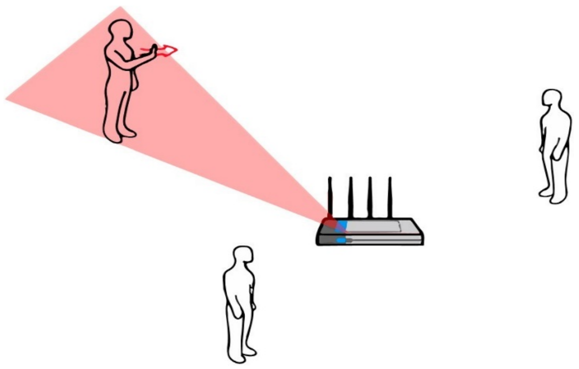

Figure 2.2: WiFi recognition of different hand positions [JES2020] .

2.5 RFID-Radio Frequency Identification

RFID systems generally consist of ultra-high frequency (UHF) commercial readers, which are capable of detecting labels. Information such as phase changes, can be collected from UHF signals and provide potential possibilities for gesture recognition. Identification tags can be carried by a person or without anyone, based on the type of RFID application, and they can work even without a battery (active and passive). RFID tags are usually used for supply chain management and automatic object identification. The advantage of the use of passive RFID technology is the low cost, but on the other hand, since they do not count with their own energy supply, the detection ranges are reduced to centimeters.

2.6 RGB Cameras

The most widely used devices for gesture captures are definitely RGB cameras. Digital cameras capture the light in three main channels of red, green and blue light. The image is saved in an array of pixels, each one obtains the three RGB channels. The object will be detected from the image in a computer-vision algorithm, in which, the order and the intensity of colour of every pixel will provide the information.

2.7 Depth Camera

This camera gives depth data which provides a map for the depth in each pixel. The depth camera has a fourth channel for the depth information, the distance between the focus and the captured object. In this way, the depth map describes the distance of an object. The advantage of this data is that it captures automatic body part segmentation via skeleton tracking so track and segment hands automatically, providing depth data related specifically to the hands. Distortions can be easily caused because of the small lense's flaws (2.3,2.4, 2.5).

Figure 2.3: Microsoft Kinect depth Camera.

2.8 Leap Motion

It is a compact and affordable recognition device, which is capable of 3D tracking forearms, hands and fingers in real time. It contains two infrared cameras, with an angle of 120° and three infrared LEDs. The sensors work at a wavelength

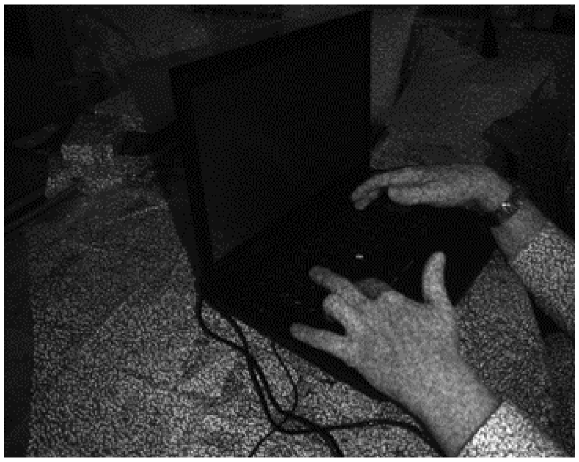

Figure 2.4: Infrared image taken by the Kinect infrared.

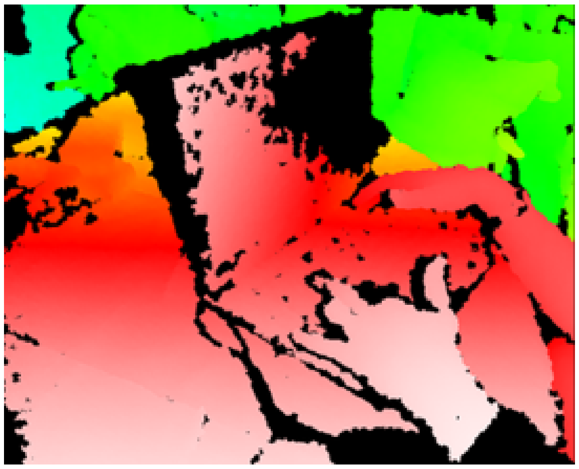

Figure 2.5: Image revisualized into a depth map using color gradients from white (near) to blue (far).

of 850 nanometers, within a non-visible spectrum for the human eye, and use sample speeds of up to 200 fps (frames per second), adapting their lighting to detected light. In addition, sensors are CMOS type, which means that no external electronics are required, because the digitization of each pixel occurs within each cell. Because of this, the capture speed is faster, and there is less use of the hardware space (2.6).

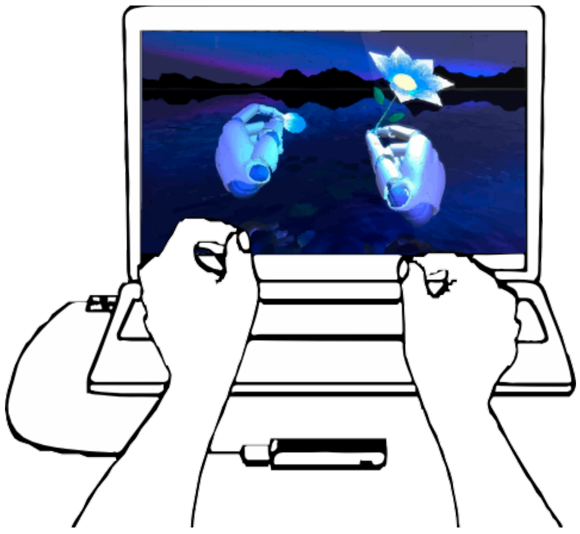

Figure 2.6: Leap Motion operating mode [JES2020]

2.9 VR-Virtual Reality

VR-based devices can create interactive spaces for different fields, such as finger tracking. There have been a lot of efforts on creating spatial awareness, close to real and more accurate, to create a full user experience. Therefore, users want to achieve the head-mounted display (HMD), a mounted screen that leads to the highest possible degree of presence. The tracked movements of the user are transformed into arbitrary coordinate systems and data formats, so the VR environment serves as a kind of meta-layer to integrate and simulate different virtual tracking systems.

Chapter 3

Applications

The gesture recognition is applied in many different fields. Some of the most common application areas of it are:

- Sign Language is one of the most important areas that gesture recognition can be very useful.
- Navigating or/and manipulating in virtual environment is another promising area that gesture recognition will play a basic role.
- In Distance learning also can be used.
- Understand human behavior in human-computer interaction is the field that now we can see the first results.
- Lastly, monitoring devices and machines from a distance is one of the subjects of the computer vision science.

3.1 Human-Machine Interaction

The interaction between a human and a device can be performed with gesture recognition. The paper [ZEN2018], presents a driver–vehicle interaction and the sensor that is used to get the depth data, is the Camboard Nano one of the types of Time-of-Flight (ToF) sensors. The control system is applied on a mobile tablet computer which is on the central console of the vehicle. ToF sensors are able to record hand gestures as depth data input in real time. Camboard nano sensor provides depth images of resolution 165 × 120 pixel at a frame rate of 90 fps.

The extracted three-dimensional point clouds are being preprocessed and serve as input for machine learning methods (3.1).

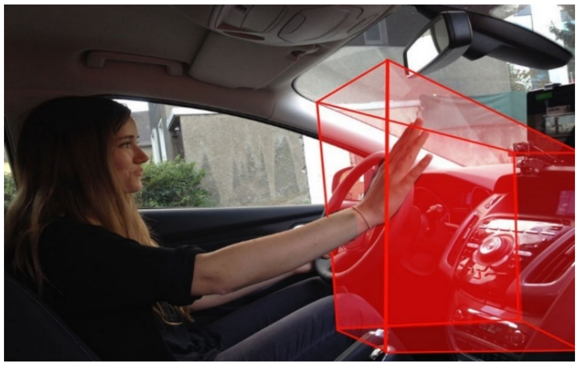

Figure 3.1: Performing a hand gesture in the detection range of time-of-flight (ToF) sensors (red area) [ZEN2018]

3.1.1 Preprocessing data

Depending on the number of hand gesture recognitions and the dimensionality of the input data, there must be robust preprocessing methods for the classification system, thorough hyperparameter tuning to achieve high recognition rates in a low runtime. The data carries irrelevant information for the hand gesture classification. PCA (Principal Components Analysis) reduces the dimensions of high-dimensional data and it is used for cropping the data to the part we want to focus on (3.2). The eigenvector equation for the covariance matrix of the depth images estimates the direction of largest variance. For an n three-dimensional coordinates input vector x, we compute the mean value m: $m = \frac{1}{n} \sum_{i=1}^{n} x_i$. The estimation of the covariance matrix as scatter matrix C: $C = \sum_{i=1}^{n} (x_i - m)(x_i - m)^T$.

In an example of testing the system, a participant drives a simulated vehicle on a predefined course with three lanes at a fixed velocity, and performs a mid-air hand gesture. With the gesture recognition the system interprets the command to change the lane (3.3).

3.2 Human-Robot Interaction

Gesture controlled robots are used in various fields:

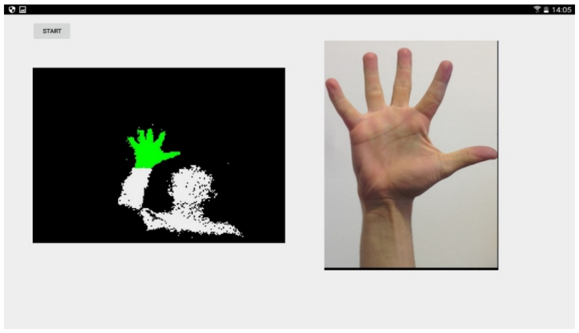

Figure 3.2: Preprocessing data with PCA the green area (left) remains to identify the correct hand posture class (right) [ZEN2018]

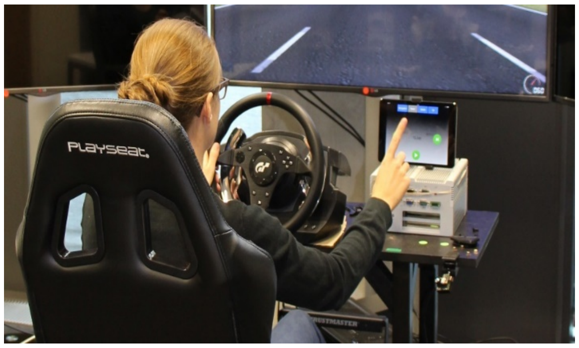

Figure 3.3: Performing the Lane Change Test while controlling the system with gestures [ZEN2018]

- In industry (car factories. . .).
- In medicine (remote surgeries with robots. . .).
- In military (armed robotic vehicles. . .).
- In space (articulated hands. . .).

3.2.1 Robot Programming Using Gestures

In the paper [TSA2016] proposed model, a high-level robot programming method based on body and hand gestures. The method is integrated within an open communication architecture based on Robot Operating System (ROS), so in this way it can be extended with new functionalities. The data formed from two devices. The first device is an RGB-d for the body gestures, and the second is leap motion for the hand gestures. There is a vocabulary of body and hand gestures, allowing the movement of robot in different directions. The body gestures control the motion of a robot arm in 6 different directions +/-x, +/-y and +/-z (3.4). The hand gestures are dynamic motions which involve movements of the fingers and they create the same motions as the body gestures. This method is applied for on-line interaction with an industrial robot, and it can perform in multi-arm robotic systems, enabling the coordinated motions execution.

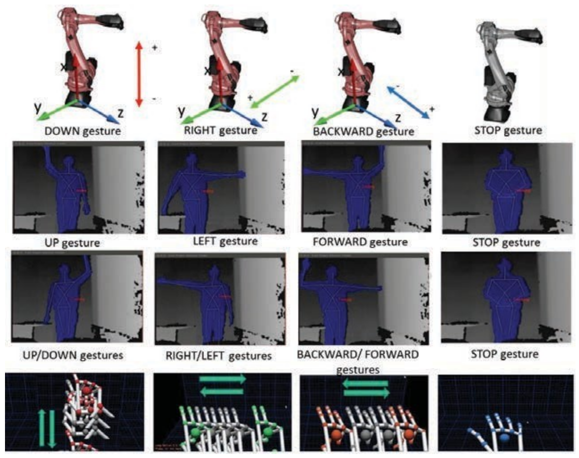

Figure 3.4: Performing body gestures as commands for robot [TSA2016]

3.2.2 The Structure of the System

The two sensors (Kinect and leap motion) are connected to computer. The computer and the robot create a network through an ethernet cable. A vocabulary includes body and hand gestures as high-level commands for the robot. The body gestures are interpreted by the Kinect and the model recognizes 18 human skeleton nodes. Analysis of the nodes ends up to the classification which is based to the vocabulary. The data from the leap motion sensor is used for the recognition of the hand gestures in the same way (3.5).

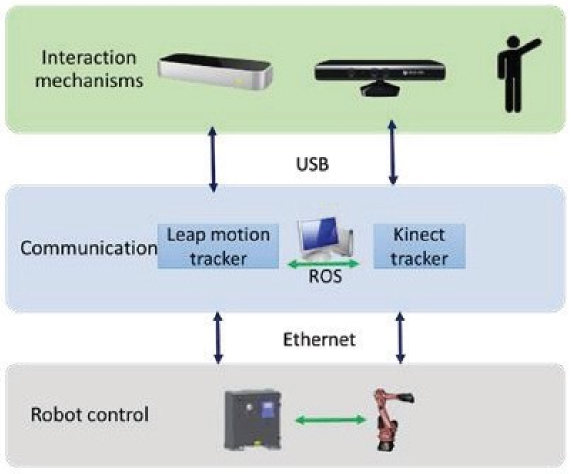

Figure 3.5: The architecture of the hardware [TSA2016]

3.3 UAV-unmanned aerial vehicle

UAV-unmanned aerial vehicle is used in many different purposes:

- In photography (aerial photos…).
- In military (surveillance…).
- In transportation (UAV taxi…).
- In delivery (courier services…).
- In agriculture (spray for pest and fungal infestations…).

3.3.1 Payload of UAVs

Besides its weight, a UAV can carry additional material such as sensors, cameras, packages for deliver etc. Depending on the size and the power of the UAV, different types of sensors can be implemented [AZO]:

- Thermal Sensors: detect the heat that radiates from any observed object and create data with images or videos.
- Chemical Sensors: detect the chemicals of the environment. The main use of them is for the safety of the workers or generally people in unknown chemical present
- Time of Flight (ToF) Sensors: Depth sensors of the ToF that emit a short infrared light pulse and they measure the time in which the infrared light pulse returns back to the drone
- Distance Sensors:
- Light-Pulse Distance Sensing (Laser): A Laser Range Finder (LRF) sends out a laser beam, detects the time that the beam returns after reflecting an object and calculates the distance.
- Radio Detection and Ranging: detect the speed and direction of the object in relation to the drone depending on their frequency shift.
- Sonar-Pulse Distance Sensing (Ultrasonic): send a sound wave at a specific frequency to an object and detect the time that it returns.
- Orientation Sensors:
- Accelerometers: sense movement of the drone.
- Inertial Measurement Sensors: detect changes of direction
- Some of the most used cameras on drones are:
- RGB camera
- RGB-depth camera
- Leap motion camera

- NIR(Near-Infrared) camera
- SWIR (short-wave infrared) camera
- LIDAR (Light Detection And Ranging) camera

3.4 Human-Drone Interaction

The paper [HUA2019] proposed a model that performs a remote control of drones by gesture recognition. The structure of the model has 5 modules (3.6):

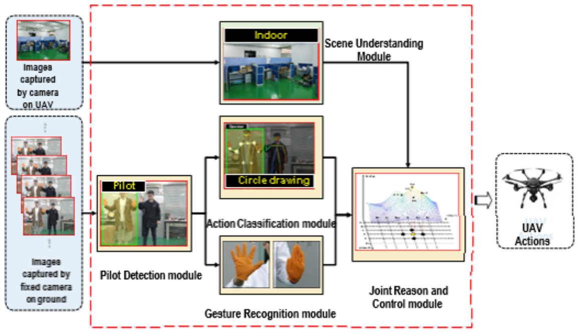
Images captured by camera on UAV
Indoor
Scene Understanding Module
Circle drawing
Action Classification module
Pilot
Pilot Detection module
Images captured by fixed camera on ground
Gesture Recognition module
Joint Reason and Control module
UAV Actions

Figure 3.6: The structure of the model [HUA2019]

- The scene-understanding module plays an important role, so the UAV can be able to execute in different environments and working conditions.
- The pilot detection module detects and isolates the real UAV pilot from any other person, extracts the skeleton points from the pilot, and determines the correct hand regions for gesture recognition.
- The action detection and recognition module recognize human actions by using the key points extracted from the pilot in the pilot detection module, as the input features (3.7). The description of human motion can be recognized by using the key points extracted from the bodies as the input features with the GCN-Graph Convolutional Network

$$f^t_{out}(x_c) = \sum_{h=1}^{K} \sum_{w=1}^{K} f_{in}(S(x_c), h, w)\dot{W}(h, w)$$

S represent the sampling function used to enumerate the neighbors of location x. W represent the weight function which provides a weight vector for computing the inner product with the sampled input feature vectors within the convolutional kernel, h and w are the pixels in the kernel (rows and columns). There are 6 kinds of actions for the control the action of UAV: a_i for $i = 1 \ldots 6$. The computation of the probability of an input action a^{in} to be classified as action a_i with a softmax function $P(a_i|a^{in})$ through M successive frames is:

$$P(a_i|a^{in}) = softmax(\sum_{t=1}^{M}\sum_{c=1}^{N})f_{out}^{t}(x_c))$$

If the maximum output value of this function is over a predefined threshold, then the input action will be considered as the corresponding action.

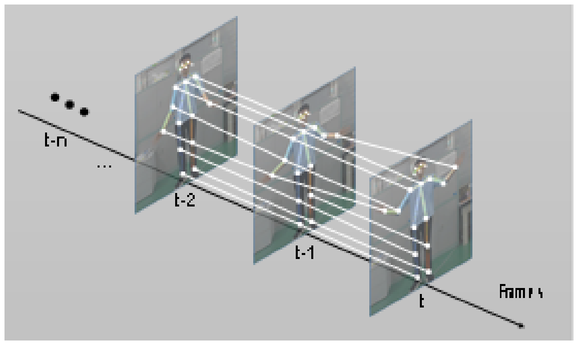

Figure 3.7: The ST-GCN spatial temporal graph of a skeleton sequence which is used to classify the action of a human [HUA2019]

- The gesture recognition module extracts the personal features of each pilot who will control the UAV and sorts them in his personal feature library before the entire system is operating. That helps the model to personalize the gestures and to avoid false interpretations.
- The joint reason and control module is formed to segregate similar gestures. For example, the actions of hands up and that of drawing circle are almost the same (3.8).

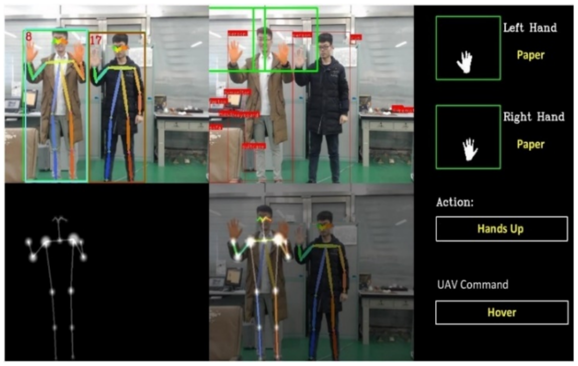

Figure 3.8: Simulation experimental result of the system when two persons perform similar actions [HUA2019]

Chapter 4

Datasets for Gesture Recognition

Recently, deep learning models are the state of the art in the methodology of gesture recognition. Different modalities of data inputs (such as skeleton joints information, human body shape, RGB, optical flow, and depth frames) are combined for the training of these models, therefore there is a significant number of datasets which had been created based on gesture recognition.

4.1 DVS128 Gesture Dataset [DVS]

The CVPR 2017 paper "A Low Power, Fully Event-Based Gesture Recognition System." describes a real time gesture Recognition model which used this dataset for training. The dataset comprises 1,342 instances of a set of 11 hand and arm gestures, grouped in 122 trials collected from 29 subjects under 3 different lighting conditions. Each trial has two files: a data file contains the DVS128 events, and an annotation file contains the start and stop times of each gesture. Filenames identify the subject and illumination condition in each trial. The event-based gesture recognition system in which the data was recorded used two devices: a DVS128 camera to generate input events, and a TrueNorth processor to inspect the input event stream. The DVS128 camera is a 128 × 128-pixel Dynamic Vision Sensor that generates events only when a pixel value changes magnitude by a user-tunable threshold. The IBM TrueNorth chip is a reconfigurable, non-von Neumann processor containing 1 million spiking neurons and 256 million synapses distributed across 4096 parallel, event-driven, neurosynaptic cores.

4.2 Hand Gesture Recognition Using 3D Skeletal Dataset [SKD]

There are 14 hand gestures in this dataset performed in two ways: with only one finger and with the whole hand. There are 2800 sequences, every gesture is performed between 1 and 10 times by 28 participants in both of the ways. All participants are right-handed. The gesture, the number of fingers that was used, the performer and the trial, were all labelled in every sequence. Each frame of sequences contains the coordinates of 22 joints both in the 2D depth image space and in the 3D world space forming a full hand skeleton. The skeletons were captured at 30 frames per second. The training set has 1960 gesture sequences (70% of the dataset) of maximum size of 171 frames and the testing set has 840 gesture sequences (30% of the dataset) of maximum size of 162 frames.

4.3 HGM-4

The HGM-4 dataset [GM4] is built for hand gesture recognition and contains 4,160 color images (1280×700 pixels) of 26 hand gestures captured by 4 cameras at different position. The images were taken indoor at different positions and the background was removed semi-automatically. This dataset can be used for studying the hand-gesture recognition problems in the perception of multiple views. Every image from 4 cameras were combined to be in the training set or in the testing set with all possible combination.

4.4 EgoGesture Dataset

The dataset contains 2.081 RGB-D videos, 24.161 samples and 2.953.224 frames from 50 distinct subjects [EGO2018]. There are 83 classes of static or dynamic gestures performed on interaction with wearable devices. The videos were made in 6 different scenes, 4 indoor and 2 outdoor. There are videos where the gestures were performed by people while they were walking. The two-modality videos, RGB and depth modules, are recorded in a resolution of 640x480 pixel with 30 fps. The gestures performed randomly, thus the videos can be applied to evaluate gesture detection in sequence. The volume of the data is about 46G of RGB-D videos and about 32G of images(320x240) (4.1).

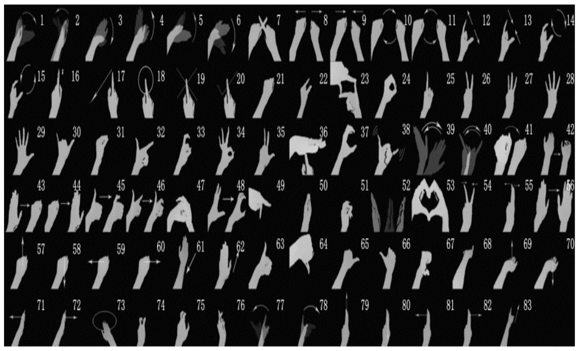

Figure 4.1: The 83 classes of the EgoGesture dataset [EGO2018]

4.5 putEMG AND putEMG-Force

These are databases of surface electromyographic activity recorded from forearm [PEM]. The dataset includes 44 healthy subjects (8 females, 36 males) aged 19 to 37 years old. Each subject participated in the experiment twice, with a minimum one-week interval, performing same procedure. Signals were recorded from 24 electrodes fixed around subject right forearm using 3 elastic bands, creating a 3×8 matrix, data is sampled at 200 Hz rate. The dataset includes 7 active gestures (like hand flexion, extension, etc.) + idle and a set of trials with isometric contractions. They are used for the implementation of algorithms for gesture recognition and for grasp force recognition. For gesture execution ground-truth, a HD Camera providing an RGB feed and a depth camera with a close view of subject's hand were used. Videos and depth images are provided alongside the EMG data (4.2).

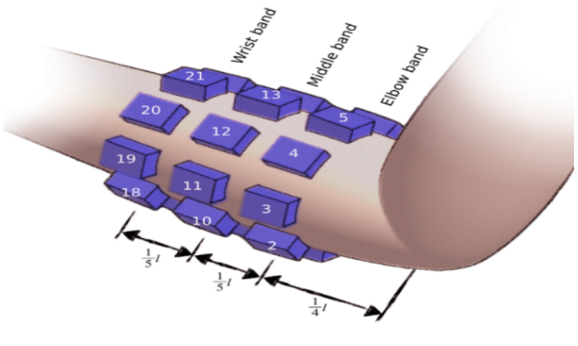

Figure 4.2: Signals are recorded from 24 electrodes fixed around subject right forearm pattern: elbow band [1-8], middle band [9-16], wrist band [17-24] [PEM]

4.6 HMD Gesture Dataset

This dataset contains about 360,000 image pairs for gesture recognition and were taken from 31 participants and 30 different environments [HMG]. The images hand gestures for AR and VR head-mounted display systems and they are captured from a stereo monochrome fisheye pair mounted in front of a typical HMD system. The dataset contains 8 gesture classes in 4 pairs: *left hand- right hand, left peace- right peace, left thumbs up,-right thumbs up, left thumbs down- right thumbs down*. For each image pair, it is provided a gesture class label and bounding box location of hands. Images also may contain cluttered background and exacting lighting conditions.

4.7 WLASL

The dataset, WLASL, is a large-scale signer-independent ASL dataset, 34.404 videos of 3.126 glosses [DON2020]. The meta data has annotations about:

- Temporal boundary: the indication of the start and end frames of a sign.
- Body Bounding-box: the reduction of the noises with the use of YOLOv3 for person detection, to identify the body of the signers in videos.
- Signer Diversity: there are inter-signer variations in the input data (example: signer appearance and signing paces).
- Dialect Variation Annotation: the dialect variations of signs contain different sign primitives, such as hand-shapes and motions, have been annotated for each gloss.

The videos have lengths ranging from 0,36 to 8,12 seconds, and the average length of all the videos is 2,41 seconds. The average intra-class standard deviation of the videos is 0,85 seconds.

4.8 The 20BN-jester Dataset V1

The video data is split into parts of 1 GB and the number of the videos is 148.092 and their total size is 22.8 GB [JES]. Total number of frames is 5.331.312. The videos for the training set are 118.562, for the testing set 14.743, and for the validation set 14.787. The 1.376 actors have recorded a set of 27 actions. The average duration of the videos is 3 second and the average number of videos for every participant is 43. The JPG images from the videos were extracted at 12 frames per seconds and there are 27 classes of the dataset. The subjects performed hand gestures in front of a laptop camera or webcam. The dataset focuses on a small set of action categories that encompass the most commonly performed human gestures in the context of visual human-computer interfaces.

4.9 UAV-Gesture

The data [ASA2018] was collected on a place located in the middle of a wheat field from a rotorcraft UAV (3DR Solo) in slow and low-altitude flight. For video recording, we used a GoPro Hero 4 Black camera with an anti-fish eye replacement lens. The videos have a resolution HD (1920 × 1080) at 25 fps. In these videos, the subject is located in the middle of the frame and execute 13 gestures, each of the gesture is performed five to ten times. When recording the gestures, sometimes the UAV drifts from its initial hovering position due to wind gusts. This adds random camera motion to the videos making them closer to practical scenarios. There are rich variations in the recorded gestures in terms of the phase, orientation, camera movement and the body shape of the actors. Thirteen body joints are annotated in 37.151 frames, namely ankles, knees, hip-joint, wrists, elbows, shoulders and head and each annotation has also the gesture class, the subject identity and the bounding box. The bounding box is created by adding a margin to the minimum and maximum coordinates of joint annotations in both x and y directions.

Chapter 5

Algorithms of Gesture Recognition

There is a variety of different algorithms for gesture recognition (5.1):

- 3D model-based algorithms: an object is interpreted as a collection of vertices and lines in the 3D mesh version.
- Skeletal-based algorithms: models the object but with fewer parameters than the volumetric version.
- Appearance-based models: derive the parameters directly from the images or videos using a template database.
- Electromyography-based models: classify the body movement with data of electrical signals produced by muscles.

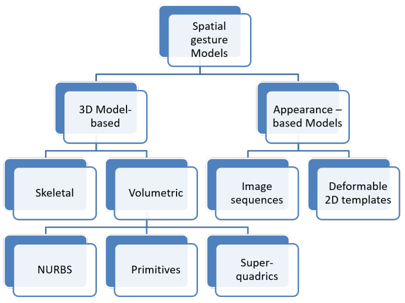

Figure 5.1: The tree of algorithms

The differences of the algorithms of the tree in the image above and how they represent a hand model each one of them are shown below (5.2).

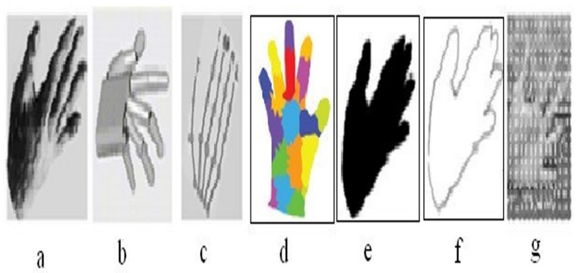

Figure 5.2: Various hand modeling methods to represent hand posture. (a) 3D volumetric model. (b) 3D geometric model. (c) 3D Skeleton model. (d) Colored marker-based model. (e) Non-geometric shape model (Binary silhouette). (f) 2D deformable template model [RAF2012]

Chapter 6

Problems of Gesture Recognition

- Lighting condition: the changes of the light in the image make the extraction of the skin region more difficult.
- Rotation: there is problem when the subject that performs the gesture is rotated in any direction.
- Scaling: the pose of the body or the hand have different sizes in the image of the gesture.
- Interpretation: the changes of the position of the subject in different images, may give false representation of the features.
- Background: when there is complex background with different objects that confuse the detection and lead to misclassification.

Chapter 7

3D CNN-Convolutional Neural Networks

3D CNN architectures can be applied in a sequence of frames as inputs thus they are used for video analysis. In recent years, the CNN approach has been revived owing to the huge advancements on computational hardware such as the general-purpose graphics processing units (GPUs). The CNN differs from the classical FC-NNs by its weights sharing mechanism. 2D CNNs make use of the spatial dependencies and 3D CNNs make use of both the spatial and temporal relations of the frames. Real-time recognition systems require the use of detection of the gesture and classification on the continuous stream of images. Many layers are the same as in the 2D CNN but all the calculations are in the 3 dimensions. Rectified linear unit (ReLU) is one of the most common activation functions, avoiding in this way the problems of the vanishing and exploding gradient descent. The 3D-CNN takes the preprocessed phase voxels as the input. Subsequent multiple convolutional layers serve as the critical composition of the CNN with 3D convolution filters and pooling operation. The way that a convolution and a pooling is computing in the input data of a 3D model is shown below (7.1, 7.2, 7.3).

7.1 Activation Functions

- ReLU -Rectified Linear Unit: $f(x) = max(0, x)$
- Sigmoid function: $f(x) = \frac{1}{1+e^{-x}}$
- Tanh function: $f(x) = tanh(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$

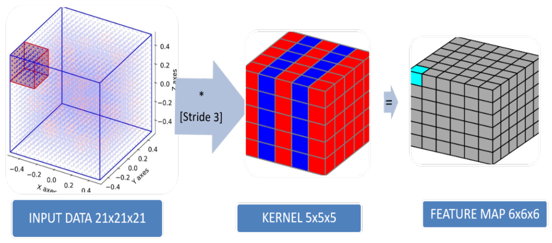

Figure 7.1: Convolution in a 3D CNN architecture [RAO2020]

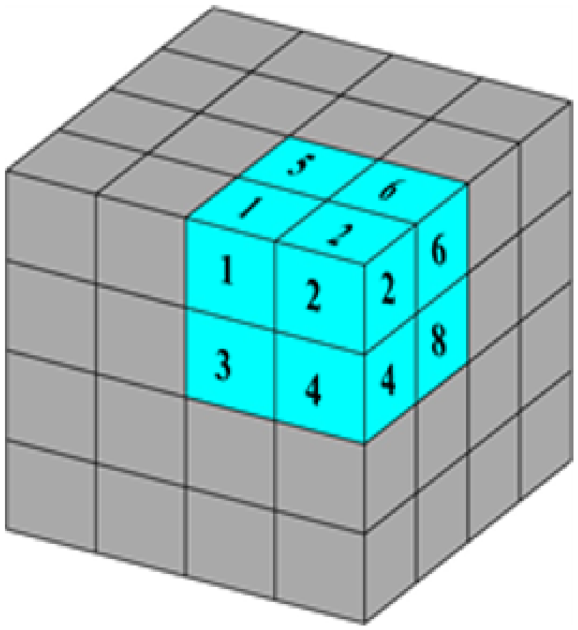

Figure 7.2: Before max pooling in 3 dimensions [RAO2020]

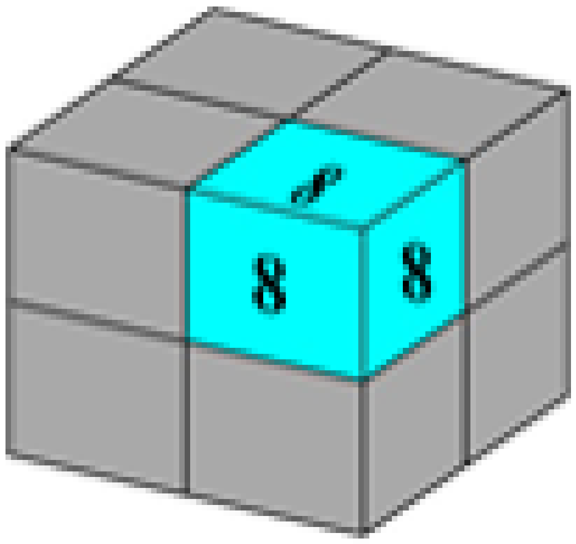

Figure 7.3: After max pooling in 3 dimensions [RAO2020]

7.1.1 ReLU

Mathematical expression of the output value γ at position (x, y, z) on j-th feature map in i-th 3D convolutional layer can be written as:

$$\gamma^i_{j,xyz} = ReLU(b^i_j + \sum_{m=1}^{M^{i-1}} \sum_{p=0}^{P^i-1} \sum_{q=0}^{Q^i-1} \sum_{r=0}^{R^i-1} w^i_{jm,pqr} \cdot \gamma^{i-1}_{m,(x+p)(y+q)(z+r)})$$

b^i_j is the common bias for j-th feature map. $w^i_{jm,pqr}$ is the (p, q, r)-th value of the 3D filter for j-th feature map at i-th layer associated with the m-th feature map in the (i-1)-th layer. M^{i-1} is the number of feature maps at (i-1)-th layer. P^i, Q^i and R^i denotes the size of the 3D filter at i-th layer.

7.2 Hyperparameters of the 3D CNN System

- The number and the size of the filters. AlexNet has 5 filters: 11x11,5x5, 3x3,3x3,3x3.
- The size of the batch input. The batch size defines the number of samples that will be propagated through the network in every step.

- The learning rate. It controls how much to change the model in response to the estimated error each time the model weights are updated

- The number of the hidden layers. When considering the structure of dense layers, there are really two decisions that must be made regarding these hidden layers: how many hidden layers to actually have in the neural network and how many neurons will be in each of these layers.

- The number of the Fully Connected layers and their neurons. The same as for the filters.

- The type of the classifier. A classifier is an algorithm that maps the input data to a specific category.

- The type and the size of the pooling. Pooling, progressively reduces the size of feature maps. the different types of pooling operations are: Maximum Pool, Minimum Pool, Average Pool, Adaptive Pool.

Chapter 8

3D CNN+LSTM-ConvLSTM

In the paper [NOO2019] the proposed model combines 3D convolution neural network connected with long short-term memory (LSTM) as a feature extraction model. Multimodal data, RGB and Depth data are incorporated as the input model. The use of the Finite State Machine (FSM) control is to reduce some gesture flows and to border the recognition classes. Usage of small classes tends to showcase higher accuracy to that of big classes. The reuse of gestures for multiple commands in one application depends on the context of each application itself. Attention is focused on the hand by removing the background and unnecessary pixels. The global side of the hand gesture recognition is analyzed. The input data includes the whole handshape instead of using finger feature for classification (8.1).

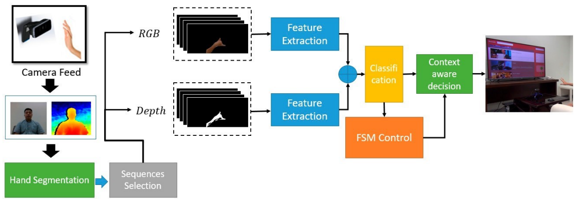

Figure 8.1: The architecture of the proposed system [NOO2019]

8.1 The Dataset of the System

Twenty individuals' data with five different environments and varying lighting conditions were collected as the dataset for training, validating and testing purposes. The dataset contains 2162 videos of users when performing 24 gestures, 13 static and 11 dynamic. Each gesture sequence consists of a three-second length dynamic gesture which contains 120 frames. The sensor for the data was The Real Sense SR300 depth camera thus the dataset includes the RGB and Depth data. To extract the hand, given the whole RGB image I_r, and depth image I_d, fixed distance dt to remove the long-distance background and minimum distance min of I_d as the range filter was defined. Let I_{rb} be the RGB image and $I_d b$ be Depth image, after background removal (8.2). Based on the range [min, dt] the average distance of a point $d_a v$ in $I_d b$ can be calculated as follows: $d_{av} = \frac{\sum_i^n I_{db}^i}{n}$ where $I_{db}^i > min$ and $I_{db}^i < d_t$

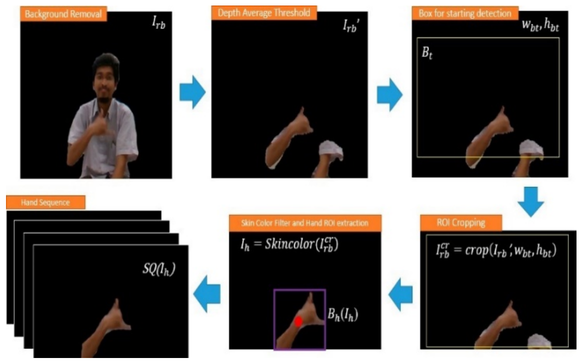
Background Removal
I_{rb}
Depth Average Threshold
I_{rb}'
Box for starting detection
w_{bt}, h_{bt}
B_t
ROI Cropping
$I_{rb}^{cr} = crop(I_{rb}{}', w_{bt}, h_{bt})$
Skin Color Filter and Hand ROI extraction
$I_h = Skincolor(I_{rb}^{cr})$
$B_h(I_h)$
Hand Sequence
$SQ(I_h)$

Figure 8.2: The preprocessing step of hand extraction based on the average depth threshold [NOO2019]

8.2 The Multimodal Architecture

The architecture of the model consists of 3DCNN layers, one stack LSTM layer and, a fully connected layer followed by the softmax layer. Batch normalization was utilized to allow the model to use much higher learning rates. The kernel size of each Conv3D layer is 3 × 3 × 3, the stride and padding are sizes of 1 × 1 × 1. Each Conv3D layer is followed by a batch normalization layer, a ReLU layer and a 3D Max Pooling layer with a pool size of 3 × 3 × 3. Features are extracted from the 3DCNN architecture then fed into the one stack of LSTM with 256 sizes of the unit. Several dropout layers were added in every section with the value of 0.5 and then computed the output probability result using the softmax layer.

8.2.1 Gates of LSTM

- Input gate: $i_t = \sigma(x_t w_{xi} + h_{t-1} whi + c_{t-1} wci + w_{ibais})$
- Forget gate: $f_t = \sigma(x_t w_{xf} + h_{t-1} whf + c_{t-1} wcf + w_{fbais})$
- Output gate: $o_t = \sigma(x_t w_{xo} + h_{t-1} who + c_{t-1} wco + w_{obais})$
- Cell gate: $z_t = tahn(x_t wxz + h_{t-1} whz + w_{zbais})$
- Memory gate: $c_t = z_t \bigotimes i_t + c_{t-1} \bigotimes f_t$
- Output activation (hidden state): $h_t = o_t \bigotimes tahn(c_t)$

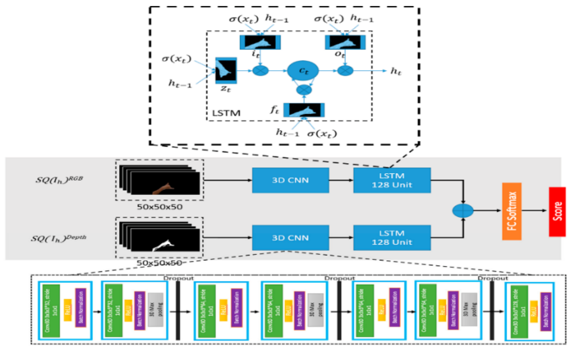

Figure 8.3: The proposed 3DCNN + LSTM architecture [NOO2019]

8.3 Handling the Data

Making fusion of both depth and RGB as the input data might produce a better result rather than using one data stream input. There are three kinds of multimodal types according to their fusion levels (8.4):

- Early fusion: 4 channels,3 channels RGB+1 channel depth, fused before the input to the 3D CNN layers.
- Middle fusion: the output of the separated 3D CNN layers, one from the RGB and the other from the Depth, are fused for the input to LSTM layer.
- Late fusion: in the same way, the fusion takes place after the LSTM layer and before Fully Connected layer.

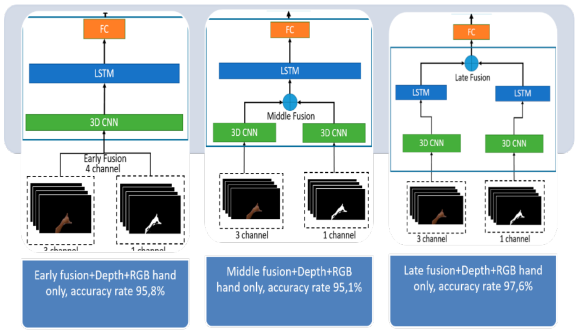
FC
LSTM
3D CNN
Early Fusion
4 channel
3 channel
1 channel
Early fusion+Depth+RGB hand only, accuracy rate 95,8%
Middle Fusion
Middle fusion+Depth+RGB hand only, accuracy rate 95,1%
Late Fusion
Late fusion+Depth+RGB hand only, accuracy rate 97,6%

Figure 8.4: [NOO2019]

8.4 Context-Aware Neural Network

One way to increase the recognition rate in the system is to let the model recognize smaller gesture class. In a Context-Aware recognition control system recognition class was limited in every context of the application. The Finite State Machine (FSM) as a controller machine connects with the Deep learning model and gives restrictions to the softmax decision probability by handling the weight in the last layer. The system in a current context or state communicates with FSM to decide which gesture should be ignored. The pre-defined weights to the last layer's node that connects to the class which is ignored by the FSM are applied, thus just the correct gestures are accepted (8.5, 8.6).

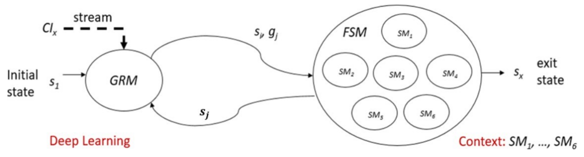

Figure 8.5: the FSM model controller with GRM (Gesture Recognition Machine) [NOO2019]

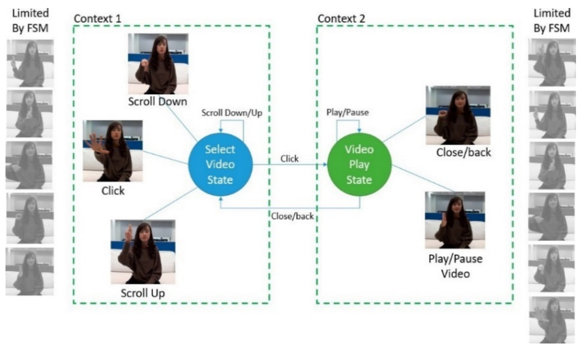

Figure 8.6: An example of two contexts in FSM model [NOO2019]

Chapter 9

Deep Learning Methods for Spatiotemporal Systems

There are 4 ways of modelling for spatiotemporal systems [EGO2018]:

- The 2D ConvNets extract features of one frame. The classifiers are trained to predict the labels of videos based on the frame features.
- The 3D ConvNets extract features of video clips. Afterwards, they aggregate the clip features into video descriptors.
- The usage of recurrent neural networks (RNN) to handle the temporal frames sequences is based on convolutional features.
- Formatting a video as one or multiple compact frames and classify it with a neural network.

9.1 Evaluation of Models

In paper [EGO2018], deep learning models are being compared using the dataset EgoGesture. The data splits into training set (60%) 1.239 videos, in validation set (20%) 411 videos and in testing set (20%) sets 431 videos. For classification, we segment the video sequences is segmented into isolated gesture samples based on the first and the last frames, which are annotated in advance. The learning task is to predict the labels of the class for each gesture sample. The numbers of gesture samples in training, validation and testing splits are 14416, 4768 and

4977. For classification, the video sequences segmented into isolated gesture samples based on the beginning and ending frames annotated in advance. The learning task is the prediction of the class labels for each gesture sample.

9.2 The Deep Learning Models

VGG16 is a 2D convolutional neural network with 13 convolutional and 3 fully-connected layers. **C3D** is a 3D CNN with eight 3D convolutional layers, one 2D pooling layers, four 3D pooling layers and three fully-connected layers. **C3D+hand mask**: a hand segmentation based C3D method since the depth camera eliminate most of the background information and the depth frame is considered as a hand mask. **C3D+LSTM+RSTTM**: a C3D augmented model with a recurrent spatiotemporal transform module (RSTTM). **VGG16+LSTM** a single-layer LSTM with 256 hidden units after the first fully-connected layer of VGG16. **IDMM+CaffeNet** handles spatial and temporal data of a video into an image called improved depth motion map (IDMM)and in this way there can be the use of 2D ConvNets for classification.

9.2.1 The Parameters of the Models

The learning rate and the batch size were set as large as possible. When the loss is steady, the learning rate reduced with a fixed decay factor which is set to 10.

- Stochastic Gradient Descent (SGD) is used for optimization.
- learning rate: VGG16 (0.001), C3D (0.003), VGG16+LSTM (0.0001).
- the step size of learning rate decay: VGG16 (5), C3D (5), VGG16+LSTM (10).
- Batch size: VGG16(60), C3D (20), VGG16+LSTM (20).

In 9.1 the methods are compared based on the accuracy rates and the proposed model **C3D+LSTM+RSTTM** has the best rates. In 9.2, the comparison is made with cross subject setting.

METHOD	RGB	DEPTH	RGB-D
IDMM+CaffeNet	-	0,664	-
VGG16 softmax	0,572	0,579	0,612
VGG16 fc6	0,625	0,623	0,665
VGG16+LSTM softmax	0,673	0,690	0,725
VGG16+LSTM lstm7	0,747	0,777	0,814
C3D fc6, 8 frames	0,817	0,844	0,865
C3D softmax, 16 frames	0,851	0,868	0,887
C3D fc6, 16 frames	0,864	0,881	0,897
C3D+HandMask	-	-	0,872
C3D+LSTM+RSTTM	0,893	0,906	0,922

Table 9.1: Gesture Classification Accuracy of the models with EGOGESTURE Dataset [EGO2018]

Method	Modality	Accuracy without CS	Accuracy with CS	Variance
VGG16 fc6	RGB	0,667	0,625	0,042
VGG16+LSTM lstm7	RGB	0,764	0,689	0,075
C3D fc6, 16 frames	RGB	0,892	0,864	0,028
VGG16 fc6	depth	0,647	0,623	0,024
VGG16+LSTM lstm7	depth	0,801	0,732	0,069
C3D fc6, 16 frames	depth	0,907	0,881	0,026
VGG16 fc6	RGB-D	0,697	0,665	0,32
VGG16+LSTM lstm7	RGB-D	0,826	0,753	0,73
C3D fc6, 16 frames	RGB-D	0,922	0,897	0,025

Table 9.2: CS: cross subject setting, when the training set and the testing set are from different subjects. Classification accuracy with or without CS [EGO2018]

Chapter 10

Comparison of Architectures

Comparison of models which were trained with WLASL dataset [DON2020]:

- 2D CNN+RNN: RNN are used for the temporal relations and 2D CNN for the spatial features of the frames (10.1).

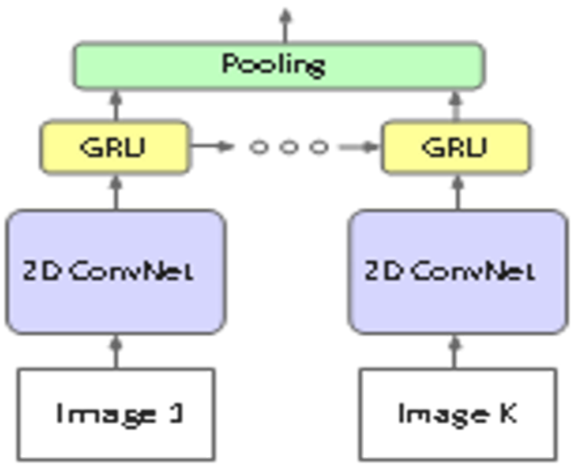

Figure 10.1: 2D Conv. RNN [DON2020]

- 3D CNN can be applied for both the spatial and temporal relationship between the frames (10.2).

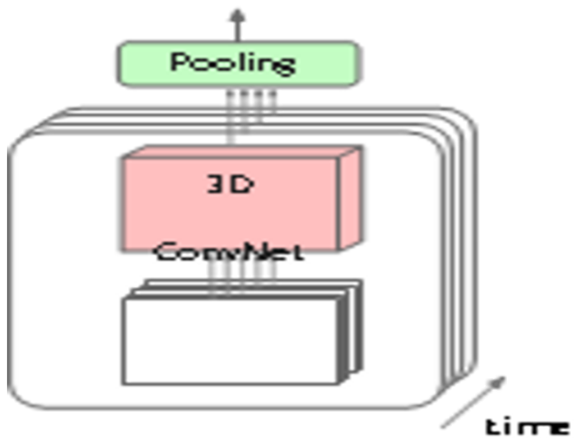

Figure 10.2: 3D CNN [DON2020]

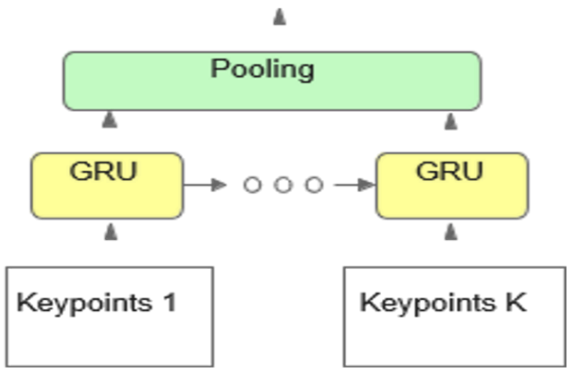

Figure 10.3: Pose RNN the keypoints are the joints of human bodies [DON2020]

- Pose based models: use RNNs to model the pose sequences for analyzing the movements (10.3).

- Temporal Graph Convolution Networks: (TGCN) models the spatiotemporal dependencies of the pose sequence (10.4).

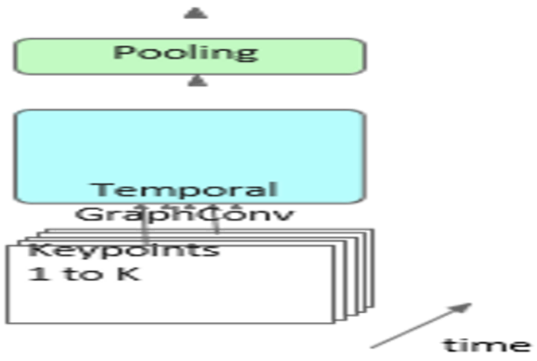

Figure 10.4: Pose TGCN [DON2020]

10.1 Pose Based Temporal Graph NEURAL NETWORKS

In the Temporal Graph Convolution Networks (TGCN), the input pose sequence: $X_{1:N} = \left[x_1, x_2, x_3 \ldots x_N\right]$, where N=frames and $X_i \in \mathbb{R}^K$ are the 2D keypoints in K dimensions. They encode the body movements as a holistic representation of the trajectories of body keypoints. In this way the dependencies among the joints of the human body are represented in a graph network. A residual graph convolutional block stacks two graph convolutional layers. A human body is represented as a fully-connected graph with K vertices and the edges in the graph as a weighted adjacency matrix: $A \in \mathbb{R}^{K \times K}$ In a deep graph convolutional network, the n-th graph layer is a function G_n that take as input features a matrix: $H_n \in \mathbb{R}^{K \times F}$. F is the feature dimension output by its previous layer. The set of trainable weights: $W_n \in \mathbb{R}^{F \times F'}$. A graph convolutional layer is expressed as: $H_{n+1} = G_n(H_n) = \sigma(A_n H_n W_n)$ where A_n is a trainable adjacency matrix for n-th layer and $\sigma(\cdot)$ denotes the activation function tanh$(\cdot)$ (10.5).

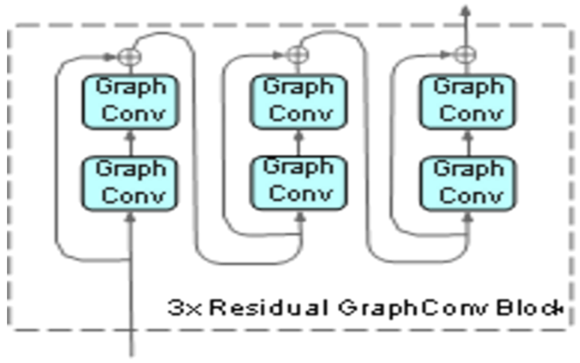

Figure 10.5: Residual Graph Convolution Block [DON2020]

All the 4 systems are compared with different datasets (10.1). In particular, there is a selection of top-K glosses with K = 100, 300, 1000, 2000, and they are organized to four subsets, named WLASL100, WLASL300, WLASL1000 and WLASL2000.

Method	**WLASL 100**	**WLASL 300**	**WLASL 1000**	**WLASL 2000**
Pose GRU	0,856	0,760	0,701	0,613
Pose TGCN	0,876	0,796	0,719	0,622
VGG+GRU	0,639	0,610	0,493	0,325
3D CNN	0,899	0,869	0,843	0,663

Table 10.1: Top-10 accuracy rates by each model on WLASL subsets with different number of glosses [DON2020].

The comparison became with combinations of the training and the testing sets (10.2).

TRAINING SET/ TESTING SET	**WLASL100**		**WLASL300**		**WLASL1000**		**WLASL2000**	
	3D CNN	**TGCN**	**3D CNN**	**TGCN**	**3D CNN**	**TGCN**	**3D CNN**	**TGCN**
WLASL100	0,899	0,876	-	-	-	-	-	-
WLASL300	0,883	0,814	0,869	0,796	-	-	-	-
WLASL1000	0,852	0,775	0,862	0,742	0,843	0,719	-	-
WLASL2000	0,720	0,678	0,711	0,654	0,673	0,645	0,663	0,622

Table 10.2: Top-10 accuracy rates of 3D CNN and Pose-TGCN with different training set (rows) and testing set (columns) on WLASL subsets [DON2020]

Chapter 11

Skeleton-Based Gesture Recognition

The paper [LIU2020] proposed an architecture which is built for a skeleton-based Gesture recognition and its approach is that the gesture is a sequence of complexly composite movements. The innovation of this architecture is that it is combined of two model: one applied on the hand posture variations and the other on the hand movements. The method can simultaneously enhance the expressive power of the posture and motion information. **HPEV** (3D hand posture evolution volume) is the model applied on the posture variations and **HMM** (2D hand movement map) model captures holistic movements. The HPEV integrates spatiotemporal information of hand postures with a 3D CNN, and on the other hand, the HMM uses a 2D CNN model to manipulate hand movement features. A unified two-stream framework learn the decoupled representations of the gestures (11.1).

11.1 FRPV - Fingertip Relative Position Vector

FRPV describes finger motion precisely the relative positions of four fingers and thumb of each frame construct it. For frame t, the four relative positions are concatenated as a vector u_t as follows:

$$u_t = (p_{I,t}, p_{M,t}, p_{R,t}, p_{L,t}) - (p_{0,t}, p_{0,t}, p_{0,t}, p_{0,t})$$

where p (0, t) is the coordinate of thumb of the t-th frame, and *p(I, t), p(M, t), p(R, t) and p(L, t)* are the coordinates of index fingertip, middle fingertip, ring fingertip and little fingertip at frame t respectively. The FRPV is the concatenation of all the u_t of each frame [N= number of frames]: $U_{FRPV} = (u_1, u_2, \ldots, u_t, \ldots, u_N)$

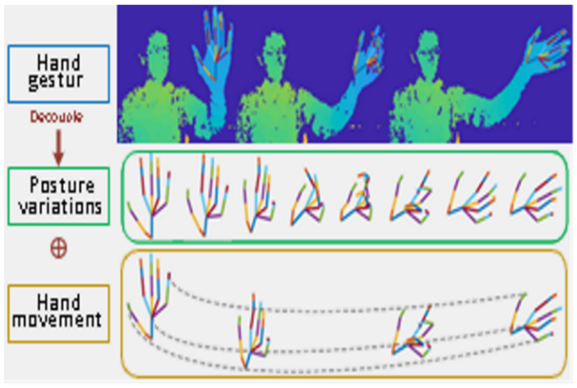

Figure 11.1: Hand gesture separated into hand posture variations and hand movements [LIU2020]

11.2 Architecture of the System

The system consists two main streams and each of them extracts a vector of the features that has built for: HPEV-Net and HMM-Net. HPEV-Net is applied for the hand movement map and uses a 3D CNN for the low-level features with the size of the kernel 7x3x3 and afterwards there is a stack of four bottleneck modules [KAI2016] for the high-level features. In every CNN layer is used batch normalization and the activation function is RELU. Also, the max pooling layers are 4x2x2. The output channels of the four bottleneck modules are 128, 128, 256 and 512. The output of the last bottleneck after global average pooling ends into a feature vector. The Fingertip Relative Position Vector (FRPV), is applied to describe movements of the fingers, because the restrictions on the resolution make very difficult the encoding of these movements. The output of FRPV goes to the next fully connected layer with Batch Normalization and ReLU activation. In the same stream, the outputs of HPEV and FRPV, are concatenated and there is a classification of the hand gesture sequences with a softmax algorithm. On the other stream, Hand Movements Map (HMM), uses a 2D CNN for the for the motion of the hand. HMM-Net is based on the Hierarchical Co-occurrence Network (HCN) the paper's [CHA2018] proposed network, to extract features. In the same way like the HPEV-Net, there is a stack of four bottleneck modules. The output channels of the four bottleneck modules are 128, 128, 256 and 512. The output of the last bottleneck after global average pooling ends into a feature vector (11.2).

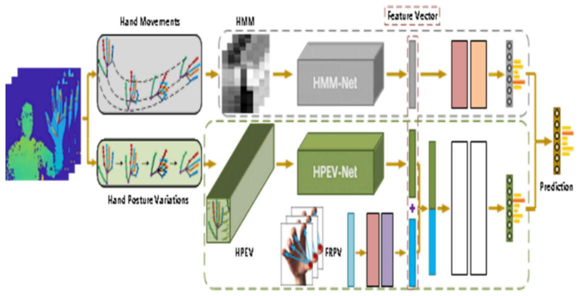

Figure 11.2: The two Neural Networks HPEV-Net+HMM-Net [LIU2020]

11.3 Datasets for Analysis

- SHREC'17 Track: Dataset of the paper [QUE2017] contains 14 gestures. They are performed twice: with one finger and with the whole hand. It includes 2800 sequences, 1960 for the training set and 840 for testing set.
- DHG-14/28: The dataset of the paper [HAZ2016] comprises 14 gestures with 2800 sequences. The DHG-14/28 and the SHREC'17 Track datasets have the same hand joints and the same data collection method.
- FPHA: The last dataset in the paper [GUI2018] provides dynamic hand sequences. It includes 1175 action had sequences, with 45 categories handling 26 different objects in 3 scenarios. It has one less hand joint from the SHREC'17 Track dataset. The training set (600 sequences) and testing set (575 sequences) have almost the same percentage of data (11.3).

A comparison of the separated networks that form the system 11.1
In table 6 (11.2), there is a comparison of some proposed models performing with the SHREC'17 dataset. The paper [LIU2020] proposed model has the best accuracy rate.

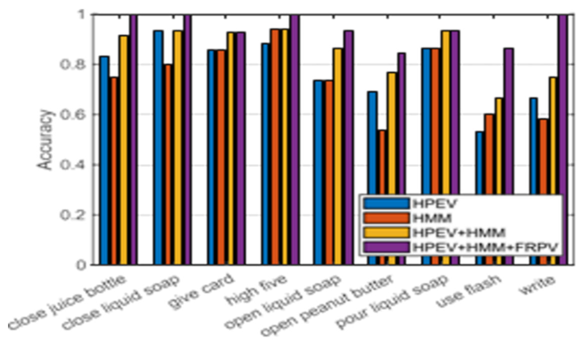

Figure 11.3: Gesture recognition accuracy rates using different input combinations on FPHA dataset [LIU2020]

Method	SHREK		FPHA
	14G	28G	
HPEV	0,734	0,714	0,770
HMM	0,927	0,866	0,677
FRPV	0,628	0,588	0,664
HPEV+HMM	0,944	0,902	0,829
HPEV+HMM+FRPV	0,948	0,922	0,909

Table 11.1: Gesture Recognition accuracy rates for different input combinations on SHREC'17 Track and FPHA dataset.14G:14 gestures, 28G:28 gestures [LIU2020].

METHOD	ACCURACY 14G	ACCURACY 28G
HON4D [OMA2013]	0,785	0,740
SoCJ+Direction+Rotation [SME2017]	0,869	0,842
SoCJ+HoHD+HoWR [HAZ2016]	0,882	0,819
Two-stream 3D CNN [JUA2018]	0,834	0,774
Res-TCN [JIN2018]	0,911	0,873
STA-Res-TCN [JIN2018]	0,936	0,907
ST-GCN [YAN2018]	0,927	0,877
ST-TS-HGR-NET [XUA2019]	0,942	0,894
DG-STA [YUX2019]	0,944	0,907
HPEV+HMM+FRPV	0,948	0,922

Table 11.2: Gesture recognition comparison of some of the latest proposed models with SHREC'17 dataset 14G:14 gestures, 28G:28 gestures [LIU2020]

Bibliography

[ASA2018] Asanka G Perera, Yee Wei Law, , and Javaan Chahl. "uav-gesture: A dataset for uav control and gesture recognition". 2018.

[AZO] `https://www.azosensors.com/article.aspx?ArticleID=1149`.

[CHA2018] Chao Li, Qiaoyong Zhong, Di Xie, and Shiliang Pu. "co-occurrence feature learning from skeleton data for action recognition and detection with hierarchical aggregation". 2018.

[DON2020] Dongxu Li, Cristian Rodriguez Opazo, Xin Yu, and Hongdong Li. "word-level deep sign language recognition from video: A new large-scale dataset and methods comparison". 2020.

[EGO2018] Yifan Zhang, Congqi Cao, Jian Cheng, and Hanqing Lu. "egogesture: A new dataset and benchmark for egocentric hand gesture recognition". 2018.

[GM4] `https://data.mendeley.com/datasets/jzy8zngkbg/4`.

[GUI2018] Guillermo Garcia-Hernando, Shanxin Yuan, Seungryul Baek, , and Tae-Kyun Kim. "first-person hand action bench- mark with rgb-d videos and 3d hand pose annotations". 2018.

[HAZ2016] Quentin De Smedt, Hazem Wannous, and Jean-Philippe Vandeborre. "skeleton-based dynamic hand gesture recognition". 2016.

[HMG] `https://sites.google.com/view/hmd-gesture-dataset/`.

[HUA2019] Bo Chen, Chunsheng Hua, Decai Li, Yuqing He, and Jianda Han. "intelligent human–uav interaction system with joint cross-validation over action–gesture recognition and scene understanding". 2019.

[JES] `https://20bn.com/datasets/jester/v1`.

[JES2020] Jesús Galván-Ruiz, Carlos M. Travieso-González, Acaymo Tejera-Fettmilch, Alejandro Pinan-Roescher, Luis Esteban-Hernández, and Luis Domínguez-Quintana. "perspective and evolution of gesture recognition for sign language: A review". 2020.

[JIN2018] Xinghao Chen Jing-Hao Xue Rui Zhu Huazhong Yang Jingxuan Hou, Guijin Wang. "spatial-temporal attention res-tcn for skeleton-based dynamic hand gesture recognition". 2018.

[JUA2018] Mengyuan Liu Juanhui Tu and Hanying Liu. "skeleton- based human action recognition using spatial temporal 3d convolutional neural networks". 2018.

[KAI2016] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "deep residual learning for image recognition". 2016.

[LIU2020] Jianbo Liu, Yongcheng Liu, Ying Wang, Veronique Prine, Shiming Xiang, and Chunhong Pan. "decoupled representation learning for skeleton-based gesture recognition". 2020.

[NOO2019] Noorkholis Luthfil Hakim, Timothy K. Shih, Sandeli Priyanwada Kasthuri Arachchi, Wisnu Aditya, Yi-Cheng Chen, and Chih-Yang Lin. "dynamic hand gesture recognition using 3dcnn and lstm with fsm context-aware model". 2019.

[OMA2013] Omar Oreifej and Zicheng Liu. Hon4d. "histogram of oriented 4d normals for activity recognition from depth sequences". 2013.

[PEM] `https://biolab.put.poznan.pl/putemg-dataset/`.

[QUE2017] Quentin De Smedt, Hazem Wannous, Jean-Philippe Vande-borre, Joris Guerry, Bertrand Le Saux, and David Filliat. "shrec'17 track: 3d hand gesture recognition using a depth and skeletal dataset". 2017.

[RAF2012] Rafiqul Zaman Khan and Noor Adnan Ibraheem. "comparative study of hand gesture recognition system". 2012.

[RAO2020] Chengping Rao and Yang Liu. "three-dimensional convolutional neural network (3d-cnn) for heterogeneous material homogenization". 2020.

[SME2017] Quentin De Smedt. "dynamic hand gesture recognition-from traditional handcrafted to recent deep learning approaches". 2017.

[TSA2016] Panagiota Tsarouchi, Athanasios Athanasatos, Sotiris Makris, Xenofon Chatzigeorgiou, and George Chryssolouris. "high level robot programming using body and hand gestures". 2016.

[XUA2019] Olivier Le´zoray Xuan Son Nguyen, Luc Brun and Se´bastien Bougleux. "a neural network based on spd manifold learning for skeleton-based hand gesture recognition". 2019.

[YAN2018] Dahua Lin Sijie Yan, Yuanjun Xiong. "spatial temporal graph convolutional networks for skeleton-based action recognition". 2018.

[YUX2019] Xi Peng Jianbo Yuan Yuxiao Chen, Long Zhao and Dimitris N Metaxas. "construct dynamic graphs for hand gesture recognition via spatial-temporal attention". 2019.

[ZEN2018] Nico Zengeler, Thomas Kopinski, and Uwe Handmann. "hand gesture recognition in automotive human–machine interaction using depth cameras".