This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement n: 761999

EasyTV: Easing the access of Europeans with disabilities to converging media and content.

D3.8 Remote control with gesture/gaze controls final version

EasyTV Project

H2020. ICT-19-2017 Media and content convergence. – IA Innovation action.

Grant Agreement n°: 761999

Start date of project: 1 Oct. 2017

Duration: 30 months

Document. ref.: Deliverable 3.8

Disclaimer

The EasyTV Consortium consists of the following partners:

	Partner Name	Short name	Country
1	Universidad Politécnica de Madrid	UPM	ES
2	Engineering Ingegneria Informatica S.P.A.	ENG	IT
3	Centre for Research and Technology Hellas/Information Technologies Institute	CERTH	GR
4	Mediavoice SRL	MV	IT
5	Universitat Autònoma Barcelona	UAB	ES
6	Corporació Catalana de Mitjans Audiovisuals SA	CCMA	ES
7	ARX.NET SA	ARX	GR
8	Fundación Confederación Nacional Sordos España para la supresión de barreras de comunicación	FCNSE	ES

PROGRAMME NAME:	H2020. ICT-19-2017 Media and content convergence – IA Innovation action
PROJECT NUMBER:	761999
PROJECT TITLE:	EASYTV
RESPONSIBLE UNIT:	CERTH
INVOLVED UNITS:	CERTH, UPM
DOCUMENT NUMBER:	D3.8
DOCUMENT TITLE:	Remote control with gesture/gaze controls final version
WORK-PACKAGE:	WP3
DELIVERABLE TYPE:	Report
CONTRACTUAL DATE OF DELIVERY:	30-09-2019
LAST UPDATE:	30/09/2019
DISTRIBUTION LEVEL:	PU

Distribution level:

PU = *Public,*

RE = *Restricted to a group of the specified Consortium,*

PP = *Restricted to other program participants (including Commission Services),*

CO = *Confidential, only for members of the LASIE Consortium (including the Commission Services)*

Document History

VERSION	DATE	STATUS	AUTHORS, REVIEWER	DESCRIPTION
v.0.5	01/09/2019	Draft	Dimitrios Konstantinidis (CERTH)	Table of Contents definition and document structure
v.1.0	23/09/2019	Final	Dimitrios Konstantinidis, Kosmas Dimitropoulos and Petros Daras (CERTH)	First full version
v.1.1	25/09/2019	Review	Jordi Fabregat (CCMA)	Revision of document from CCMA
v.1.2	27/09/2019	Review	Pablo Callega-Ibanez (UPM)	Revision of document from UPM
v.2.0	30/09/2019	Final	Dimitrios Konstantinidis, Kosmas Dimitropoulos and Petros Daras (CERTH)	Final version after revision from CCMA and UPM

Definitions, Acronyms and Abbreviations

ACRONYMS / ABBREVIATIONS	DESCRIPTION
DOW	Description of Work
RGB	Red-Green-Blue
3D	3-Dimensional
LDS	Linear Dynamical System
LSTM	Long Short-Term Memory
SLR	Sign Language Recognition
HMM	Hidden Markov Model
CNN	Convolutional Neural Network
CPM	Convolutional Pose Machine
GPD	Grassmannian Pyramid Descriptor
FC	Fully Connected
HoGP	Histogram of Grassmannian Points
2D	2-Dimensional
AV	Average Voting
MV	Majority Voting
DSC	Dynamic Score Combination
PSO	Particle Swarm Optimisation
DWA	Deep Weight Averaging
SDK	Software Development Kit
PC	Personal Computer
FPS	Frames Per Second
DTW	Dynamic Time Warping
HCRF	Hidden Conditional Random Field

Table of Contents

List of Figures

List of Tables

1. EXECUTIVE SUMMARY

In this document, we present and analyse technical information and specifications regarding the remote control of the TV set based on gesture/gaze information, as described in Task 3.4 of the EasyTV Description of Work (DOW) [1]. More specifically, the gesture/gaze remote control will be integrated in the universal accessible remote control, allowing people with disabilities to easily control the TV set. After taking into consideration the technical requirements and end-users needs, established in T1.1 and T1.2 and the preliminary version of the proposed gesture/gaze remote control system, proposed in D3.3, we present in this deliverable the final version of the gesture/gaze remote control, as it was formed after two years of the EasyTV project.

The document is structured in 5 main chapters:

Chapter 2 introduces the scope and purpose of the document, along with its interconnection to other tasks and deliverables. Moreover, Chapter 2 presents the refinements and improvements in the gesture/gaze remote control that led from the preliminary to the final version.

Chapter 3 deals with the problem of motion capturing, feature extraction and action recognition from people performing body and hand movements. These features can assist a gesture classification system to successfully control the TV set.

Chapter 4 provides a thorough description of the final version of the EasyTV gesture/gaze remote control, including the different functionalities and modes of operation it supports.

Finally, Chapter 5 concludes the document by presenting an overview of the theoretical and technical work performed during the first two years of the EasyTV project.

2. INTRODUCTION

This deliverable describes the final version of the gesture/gaze remote control according to the project objectives, presented in the EasyTV DOW, the user requirements, presented in the deliverable D1.1 [2], the system specifications, defined in D1.2 [3] and the preliminary version of the gesture/gaze remote control, presented in D3.3 [4].

2.1. Purpose and scope

The purpose of this deliverable is to present the final version of the gesture/gaze remote control that will be employed by people with disabilities in order to control their TV sets. To be able to design such a system, extensive research was performed not only on available hardware setups, but also on which features can be extracted from the performed gestures. Such information is vital for a gesture recognition system in order to reliably identify gestures. Furthermore, gesture recognition systems of TV manufactures were taken into account in order to guide us towards the implementation of the proposed set of gestures.

The proposed gesture/gaze remote control is part of a universal remote control that is developed in the framework of the EasyTV project in order to ease the access of people with disabilities to the TV. The gesture/gaze remote control is implemented on a PC in order to take advantage of the processing power of modern PC systems for the processing of video sequences and classification of gestures. As far as the communication between the modules of the EasyTV universal remote control and the TV set is concerned, the HbbTV protocol is adopted in order for the proposed remote control to be suitable for TV sets with new technologies.

2.2. Relation to other deliverables

The gesture/gaze remote control, proposed in this document, is heavily based on deliverables D1.1 and D1.2 that describe the user needs and system specifications and D3.3 that describes the first version of the gesture/gaze remote control. Moreover, a connection with the other modules of the EasyTV platform can be observed in D1.4 [5] (Final release of the EasyTV system architecture). The gesture/gaze remote control and the speech remote control, defined in D3.4 and D3.6 will constitute the universal remote control, proposed in the framework of the EasyTV project. As a result, these two controls will share technology with each other in order to be able to communicate with the TV set as a universal system. Finally, this document will describe potential improvements to the final version of the gesture/gaze remote control that will enhance users' experience and improve the usability of the service.

2.3. Improvements with respect to the preliminary version

There have been a number of significant improvements and refinements to the gesture/gaze remote control during the previous year of the project. These improvements enhanced the usability of the gesture/gaze remote control by adding new functionalities and making the graphical interface more intuitive to the users. These improvements are briefly presented below, while they are discussed in detail in later sections.

- Multi-language Graphical User Interface (GUI) for the gesture/gaze remote control (i.e., English, Greek, Spanish, Italian and Catalan).
- Integration of additional hand gestures to support new functionalities (i.e., language change and skeleton visualization).
- Integration of eye tracker sensor for gaze control of the TV set as a third mode of operation.
- Introduction of new functionalities for the display of skeleton, depth and color information to the users.
- Code refactoring to facilitate the addition of new sensors.
- Re-design of the GUI of the gesture/gaze remote control to be more intuitive to the users.

3. GESTURE RECOGNITION

In this chapter, we first present a literature review of motion capturing systems and video processing and classification techniques in the context of human action recognition and sign language classification. Then, we describe our research during the first two years of the EasyTV project, in the context of feature extraction from video sequences that can enable us to accurately and robustly identify human actions, signs and gestures. Finally, we present results of applying our proposed methodology in well-known action and sign language recognition datasets, along with ways to extend our work towards gesture classification tasks.

3.1. Motion capturing technologies

Motion capturing systems can be classified in two major categories: marker-based and marker-less. Marker-based systems consist of markers placed in the body of a person. These markers can provide either optical (retroreflective, colour, light emitting markers) or inertial (Micro-Electro-Mechanical systems) information, such as position, velocity and orientation.

Figure 1: Marker-based motion capturing systems.

The main advantage of such systems is their high accuracy and this is the reason they are usually employed in 3D films and video games. The disadvantages of such systems are their constraints as the person performing actions should wear costume to hold the markers and their high price that renders them inapplicable for everyday usage.

Marker-less motion capturing systems is an economic alternative to the marker-based systems and this is what we are going to use in the context of the EasyTV project. Sensors that belong to this category are RGB cameras that provide just colour information (see Figure 2) and RGB-D sensors that provide both colour and depth information (see Figure 3). RGB-D sensors can usually provide additional information, such as body skeletal joints and face points. Well-known RGB-D sensors that can be employed for gesture recognition in the context of the EasyTV project are Kinect v2 [6], ORBBEC [7] and Intel RealSense [8]. Finally, there are also depth sensors that provide only depth information, such as Camboard Pico Flexx [9]. Such sensors can also be considered for gesture recognition tasks.

Figure 2: RGB sensors provide colour images and video sequences.

Figure 3: RGB-D sensors provide colour, depth and usually skeletal information. From left to right: Kinect v2, ORBBEC and Intel RealSense.

3.2. Feature extraction techniques

In this project, we are mainly concerned with the extraction of features for human action and sign language recognition (SLR) as gesture recognition can be considered as a subcategory of these problems. Human action recognition refers to the identification of specific body movements performed by humans and has been widely applied for surveillance, video retrieval and human machine interaction. On the other hand, sign language recognition is a quite different problem from human action recognition as it involves the identification of a structured set of hand gestures with a specific meaning that is employed from hearing impaired people in order to communicate in everyday life. Sign language recognition is a challenging problem due to the fact that sign language features thousands of signs, sometimes differing only by subtle changes in hand motion, shape or position and involving significant finger overlaps and occlusions. Combined also with differences in the signing style between individuals, SLR can be very challenging for current computer vision algorithms. Finally, the unavailability of large sign language datasets and the fact that sign language is not universal but presents significant variations based on the ethnicity of signers pose challenges to the development of an accurate and robust SLR system.

For such tasks, feature extraction techniques can be categorised based on the data acquisition method, resulting in either direct measurement or vision-based approaches. Direct measurement methods rely on motion data acquired by data gloves, sensors or motion capturing systems [10][11]. The extracted motion data can provide accurate tracking of hands, fingers and other body parts, leading to the development of robust recognition methodologies, at the expense of costly setups and obtrusive systems as the movements of a person are severely restricted from wearing the input devices.

On the other hand, vision-based approaches have been traditionally employed in order to extract discriminative spatial and temporal features from Red-Green-Blue (RGB) video sequences. Although unobtrusive, such methodologies present inaccuracies due to hand and finger overlaps. Most vision-based SLR methods attempt to initially track and extract hand regions prior to their

classification to gestures. Hand detection has been achieved by semantic segmentation and skin colour detection as skin colour is usually easy to distinguish [12][13]. However, due to the fact that other body parts (e.g., face and arms) can be erroneously recognised as hands, recent hand detection methods rely also on face detection and subtraction and background subtraction to identify only the moving parts of a scene[14][15]. To achieve accurate and robust hand tracking, especially in cases of occlusions, previous methods employ filtering techniques, such as Kalman and particle filters [15][16].

However, the sensitivity of the RGB data to illumination changes, background clutter and occlusions and the technological advances in depth sensors has led to the introduction of skeletal data (i.e., set of joints in the 3-Dimensional (3D) space) for human action recognition as they have proven to be robust to illumination variations, human scale and viewpoint. Although modern depth sensors can reliably extract 3D joint coordinates, skeleton-based action recognition remains a challenging problem due to variations in the way people perform actions and joint self-occlusions. Several authors in the literature employed raw skeletal data or similar feature representations in an attempt to improve the accuracy of skeleton-based action recognition [17][18][19][20][21]. Other works construct actionlets [22] or gesturelets [23] by capturing the local interactions and kinematic information between skeleton joints respectively. Taking a different approach and based on Linear Dynamical Systems (LDS), which have been widely used in the past for dynamic texture classification [24], Dimitropoulos et al. proposed the representation of skeleton action sequences as clouds of points in a Grassmannian manifold [25]. Lately, Zhang et al. proposed the extraction of several geometric features from skeleton joints [26], while Wang et al. in [27] proposed Joint Trajectory Maps that compactly encode spatio-temporal information of 3D skeleton sequences into multiple 2D images.

Lately, deep learning techniques enabled the extraction of skeletal data from RGB video sequences. One such algorithm and the one that we are employing for the proposed action and sign language recognition methodologies in the framework of the EasyTV project is OpenPose [28]. OpenPose accepts RGB images as input and outputs a set of keypoints for the body, hands and face. Each keypoint detector in OpenPose is a special type of Convolutional Neural Network (CNN) called Convolutional Pose Machine (CPM). CPMs have the ability to learn long-range dependencies among images and multi-part cues, and also, inherit a modular sequential design. These features enable the networks to learn feature representations for both image and spatial context directly from data. In the first stage, the convolutional pose machine predicts part beliefs from only local image evidence, while the convolutional layers in the subsequent stage allow the classifier to freely combine contextual information by picking the most predictive features. OpenPose is capable of computing a total of 135 keypoints (18 body joints, 21 joints per hand and 70 face points).

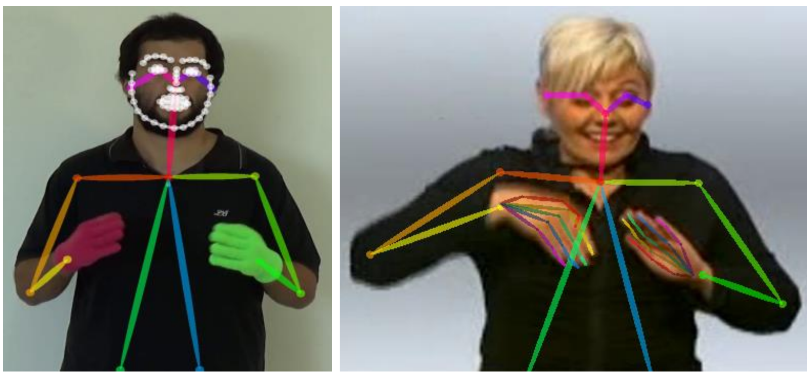


Figure 4: Detected body, hand and face features for a set of signers using the OpenPose algorithm.

3.3. Review of recognition techniques

In this section, we present a brief review of algorithms that can be employed for action, gesture and sign language recognition. We compare these algorithms and present their advantages and disadvantages with respect to the problem of gesture recognition.

3.3.1. Dynamic time warping

The Dynamic Time Warping (DTW) [29] algorithm is considered one of the earliest approaches to classify signals of varied lengths and it has been extensively used in isolated word speech and gesture recognition. The DTW algorithm finds a template or a prototypical version of each class that characterises all training signals that belong to this class and then uses these templates to find the closest match for a queried signal. The simplest form for a template can be a sequence of feature vectors.

In the case of gesture recognition, the template can be a single instance of gesture selected to be typical by some process; for example, by choosing the template which best matches a cohort of training utterances. The total distance between a queried signal and a template could be computed as the sum or the mean of the individual distances between their feature vectors. However, since the lengths of the queried signal and the template may vary, the matching process needs to compensate for length differences and take into account the non-linear nature of the length differences within the gestures. Therefore, any distance (Euclidean, Manhattan etc), which aligns the *i*-th point on one time series with the *i*-th point on the other will produce a poor similarity score. A non-linear (elastic) alignment produces a more intuitive similarity measure, allowing similar shapes to match even if they are out of phase in the time axis (Figure 5).

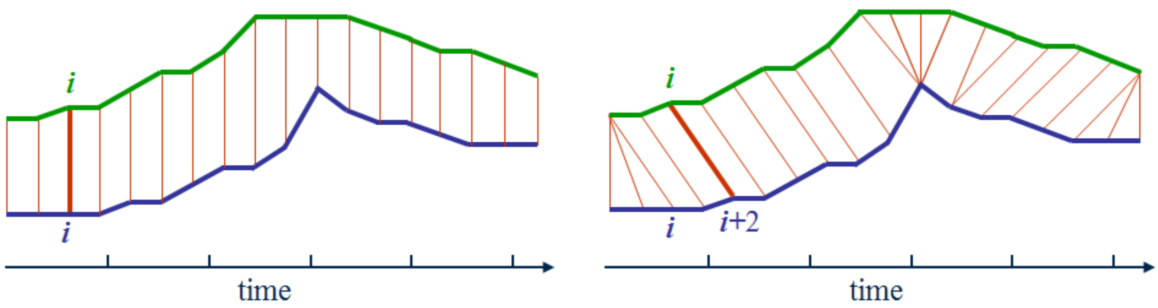


Figure 5: Depiction of a linear (left) and non-linear or elastic (right) alignment of two time series.

As a result, DTW aims at aligning two sequences of feature vectors by warping the time axis iteratively until an optimal match (according to a suitable metric) between the two sequences is found. The aligning process is subsequently described (see Figure 6). The two sequences can be arranged on the sides of a grid, with one on the top and the other up the left hand side. Both sequences start on the bottom left of the grid. Inside each cell a distance measure can be placed, comparing the corresponding elements of the two sequences. To find the best match or alignment between these two sequences one need to find a path through the grid which minimizes the total distance between them. The procedure for computing this overall distance involves finding all possible routes through the grid and for each one of them to compute the overall distance. The overall distance is the minimum of the sum of the distances between the individual elements on the path. It is apparent that for any considerably long sequences the number of possible paths through the grid will be very large.

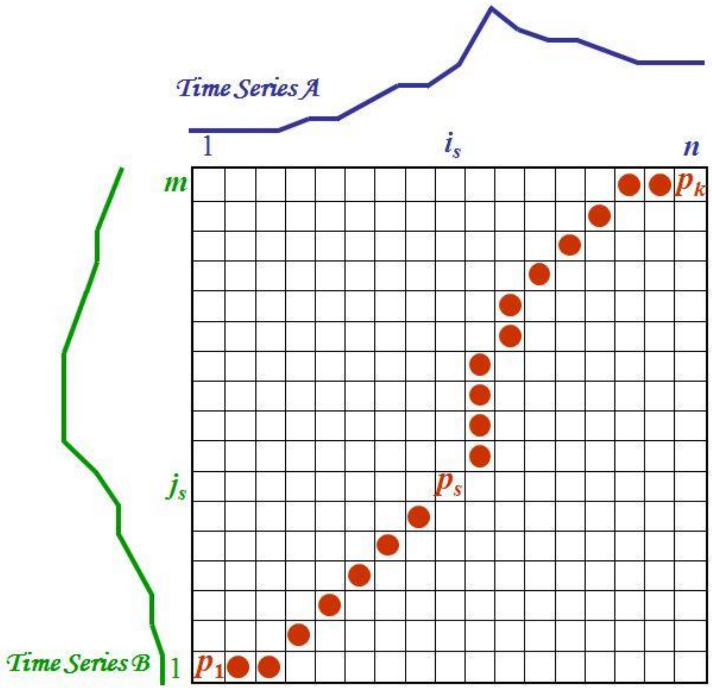
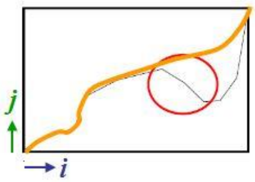
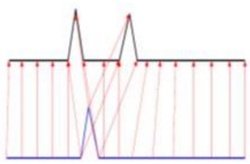
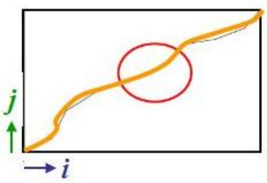
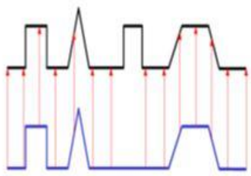
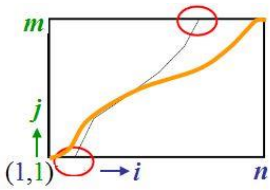
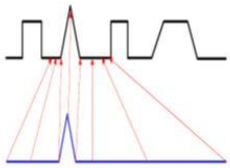
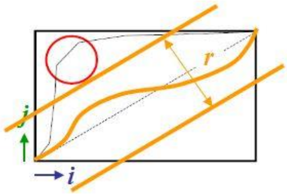
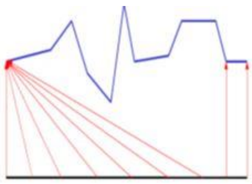
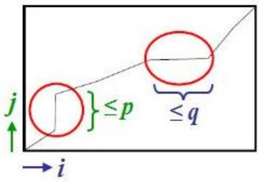
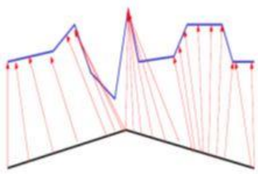


Figure 6: Two time series arranged on the sides of a grid. The path depicted is a possible alignment between the two series.

Nevertheless, there are some constraints that could be applied in order to minimize the number of considered paths. These optimisations or constraints of the DTW algorithm arise from the observations on the nature of acceptable paths through the grid and they are presented in Table 1.

Table 1: Constraints of the DTW algorithm.

Constraint	Description	Optimisation
Monotonicity	The alignment path does not go back in time index. 	Guarantees that features are not repeated in the alignment. 
Continuity	The alignment path does not jump in time index. 	Guarantees that the alignment does not omit important features. 

Boundary conditions	The alignment path starts at the bottom left and ends at the top right.  	Guarantees that the alignment does not consider partially one of the sequences. 
Warping window	A good alignment path is unlikely to wander too far from the diagonal.  	Guarantees that the alignment does not try to skip different features and gets stuck at similar features. 
Slope constraint	The alignment path should not be too steep or too shallow.  	Prevents that very short parts of the sequences are matched to very long ones. 

The advantages of the DTW algorithm are its computational efficiency and its ability to allow new classes to be introduced or changed on the fly as the algorithm does not require training. Furthermore, DTW can operate even when a single instance of a class is present. However, DTW is sensitive to the selection of a representative template for a class, which can be hard to find in some cases. Additionally, DTW is not too robust to noisy sequences or outliers.

3.3.2. **Hidden Markov models**

Different from the sequence matching DTW algorithm, Hidden Markov Models (HMMs) belong to the category of classifiers. In tasks, such as human activity and gesture recognition, HMMs excel as they can accurately classify temporal sequences. Gestures and signs often present a complex underlying temporal structure and models that incorporate hidden structures have proven to be advantageous for recognition tasks. Several existing approaches employ a HMM or a suitable variant (e.g. a factored or coupled state model) to model gestures or signs [30][31][32]. Below, we will briefly introduce HMM and examine their functionality.

Assume that there is a system that can be described using a set of N different states, where random transitions are produced over time, according to a given probability distribution for each state. The state on the system on each moment depends on the state that it was in the previous moments. This kind of stochastic process is called “Markov Model”. Additionally, if the present state of the system cannot be observed (i.e., it could be only measured by an effect that it produces), the system is called “Hidden Markov Model”.

As a result, a HMM is a generative probabilistic model that can be used to generate hidden states from observable data. The main goal of the model is to determine the hidden state sequence

$(x_1 x_2 \ldots x_t)$ that corresponds to the output sequence of observations $(y_1 y_2 \ldots y_t)$. Furthermore, the model should reliably learn its parameters (i.e., state transition and emission probabilities) from the history of observed output sequences. This can be achieved using a special version of an expectation-maximization algorithm, called Baum-Welch algorithm [33]. Figure 7 shows a graphical representation of a HMM.

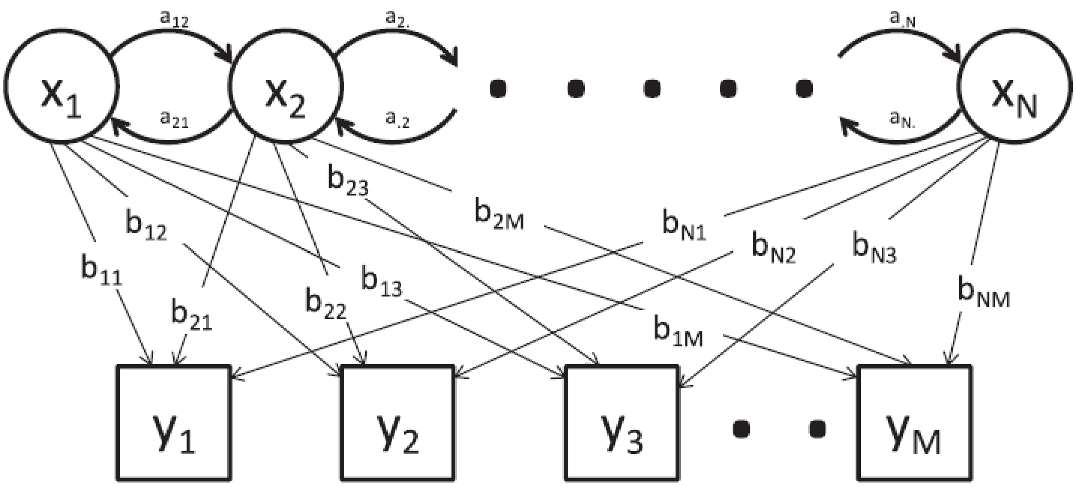


Figure 7: Hidden Markov Model (x – hidden states, y – observations, a – state transition probabilities, b – emission probabilities).

A HMM classifier is a computationally efficient algorithm that can be trained with just a few training samples. However, a significant limitation of this classifier for gesture recognition tasks is the requirement of conditional independence of observations. In addition, hidden states in a generative model are selected to maximize the likelihood of generating all the examples of a given gesture class, which is not necessarily optimal for discriminating the gesture class against other gestures.

3.3.3. Hidden conditional random fields

To overcome the limitations of HMMs, Hidden Conditional Random Fields (HCRFs) were proposed in [34]. The goal of HCRFs is to learn a mapping between a set of observations $(x_1 x_2 \ldots x_t)$ and the class label y, while also learning a set of parameters and hidden states $(h_1 h_2 \ldots h_t)$. Figure 8 shows a graphical representation of a HCRF.

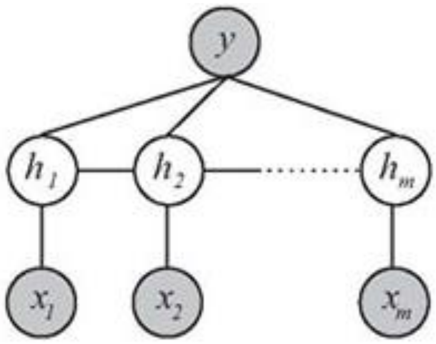


Figure 8: Hidden conditional random field (x – observations, h – hidden states, y – class label)

HCRFs are well-suited to the problem of gesture recognition as they are capable of capturing both spatial dependencies between hidden object parts and temporal dependencies across frames. Furthermore, new variants have also been proposed that can incorporate long range dependencies.

3.3.4. Recurrent neural networks

The outstanding performance of deep learning techniques on several computer vision tasks has led to their adoption for gesture and sign language recognition as well. A specific type of deep networks that deal with temporal signals is the recurrent neural networks. A special type of recurrent neural networks is the Long Short-Term Memory (LSTM) unit. The advantage of a LSTM unit over other temporal classifiers is its ability to capture long temporal dependencies among the input sequences and achieve high accuracy, while also maintaining a small number of parameters with respect to other recurrent or convolutional networks.

LSTMs and other convolutional neural networks (CNNs) were highly employed in the literature for sign language recognition and this is another reason why LSTMs seem fitted for gesture recognition as well. More specifically, Koller et al. proposed a hybrid SLR system based on a convolutional neural network (CNN) and a HMM, where the CNN was employed in order to identify the hand shape and its probabilistic output was then fed to a HMM in order to guide its inference [35]. The same authors subsequently improved their SLR methodology by additionally employing bidirectional recurrent neural networks, in the form of LSTM units [36]. On the other hand, Huang et al. proposed the use of 3D CNNs that can automatically capture both temporal and spatial information from the raw video sequences, without the need for designing features [37].

A LSTM unit is shown in Figure 9 and it is composed of a cell state and input, output and forget gates that are more analytically presented below.

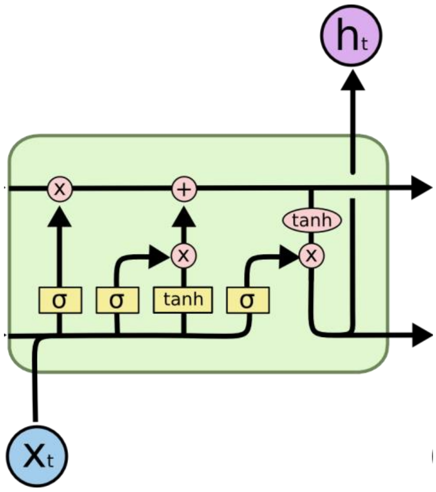


Figure 9: Diagram of a LSTM unit.

The first step in a LSTM unit is the **forget gate** that is responsible for removing information from the cell state (Figure 10). The information that is no longer required for the LSTM to understand things or the information that is of less importance is removed via a sigmoid layer. This is required for optimizing the performance of the LSTM network.

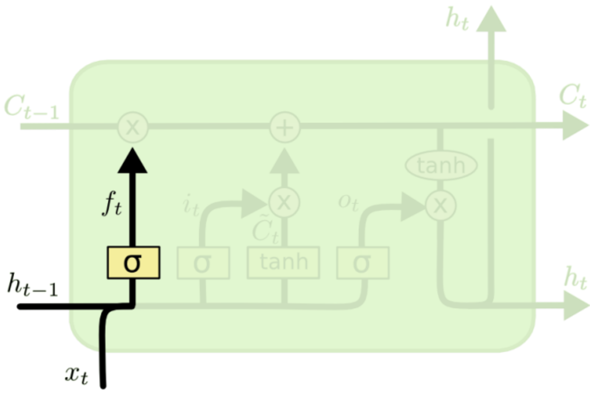


Figure 10: Diagram of the forget gate of a LSTM unit.

The forget gate takes two inputs; the current input x_t and the hidden state or output h_{t-1} from the previous unit. The output of the forget gate is given as follows:

$$f_t = \sigma\left(W_f \cdot [x_t, h_{t-1}] + b_f\right)$$

,where W_f and b_f are the weight matrix and bias of the forget gate respectively. The output of the forget gate is a vector with values ranging from 0 to 1, corresponding to each number in the cell state. Basically, the sigmoid function is responsible for deciding which values to keep and which to discard. If a '0' is output for a particular value in the cell state, it means that the forget gate wants the cell state to forget that piece of information completely. Similarly, a '1' means that the forget gate wants to remember that entire piece of information.

The next step is to decide what new information we are going to store in the cell state. This has two parts. First, a sigmoid layer called the **input gate** decides which values we want to update (Figure 11). Afterwards, a hyper-tangent (tanh) layer creates a vector of new candidate values $\widetilde{C_t}$ that could be added to the state.

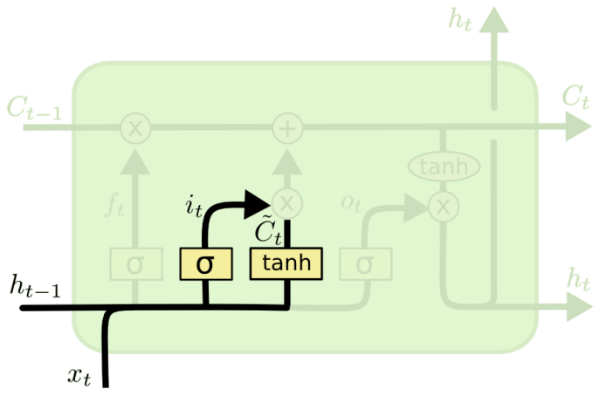


Figure 11: Diagram of the input gate of a LSTM unit.

The input layer is similar to the forget layer as it takes the input $\boldsymbol{x_t}$ and the hidden state $\boldsymbol{h_{t-1}}$ from the previous unit and outputs:

$$\boldsymbol{i_t = \sigma(W_i \cdot [x_t, h_{t-1}] + b_i)}$$

,where $\boldsymbol{W_i}$ and $\boldsymbol{b_i}$ are the weight matrix and bias of the input gate respectively. The update of the cell state $\boldsymbol{\widetilde{C_t}}$ is computed as shown below, along with the way the new cell state $\boldsymbol{C_t}$ is calculated given the cell state $\boldsymbol{C_{t-1}}$ of the previous LSTM unit and the outputs of the input and forget gates.

$$\boldsymbol{\widetilde{C_t} = tanh(W_c \cdot [x_t, h_{t-1}] + b_c)}$$
$$\boldsymbol{C_t = f_t * C_{t-1} + i_t * \widetilde{C_t}}$$

The part of the LSTM unit that is responsible for updating the cell state is presented in Figure 12. Finally, we need to decide what we are going to output from the LSTM unit. This output will be a filtered version of the cell state and is produced via the **output gate** (Figure 13). To compute this output, we initially need a sigmoid layer to decide what parts of the cell state should be outputted. Afterwards, we pass the cell state through a hyper-tangent layer in order to push the values to be between −1 and 1 and multiply it by the output of the sigmoid gate, so that we only output the parts we decide to.

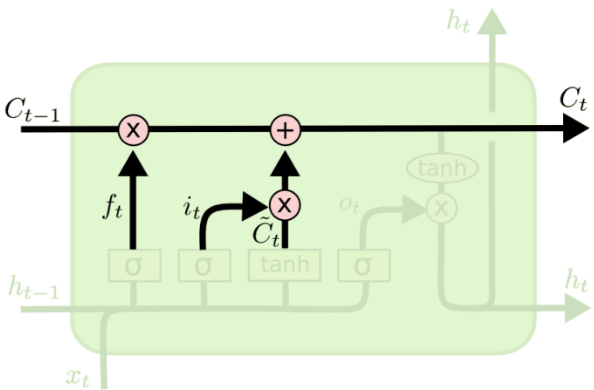


Figure 12: Update of the cell state of a LSTM unit.

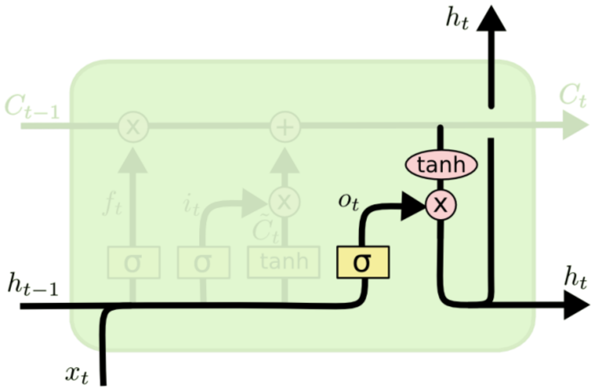


Figure 13: Diagram of the output gate of a LSTM unit.

The output gate takes as inputs the current input x_t and the hidden state or output h_{t-1} from the previous unit and outputs:

$$o_t = \sigma(W_o \cdot [x_t, h_{t-1}] + b_o)$$

$$h_t = o_t * tanh(C_t)$$

In the previous set of equations, W_o and b_o are the weight matrix and bias of the output gate respectively, while o_t and h_t are the output of the output gate and the hidden state of the LSTM unit respectively. The hidden states of the LSTM units are gathered and form the feature vector that is passed as input to subsequent deep network modules.

3.4. Proposed methodology

In this section, we describe the proposed methodologies in order to tackle the problems of action and sign language recognition. More specifically, we present how certain types of features can be extracted from video sequences and how deep learning can be employed in order to design accurate and robust action and sign language recognition methodologies.

3.4.1. Action recognition

As far as action recognition is concerned, our work is concentrated on the processing and classification of skeletal features using deep learning [38]. The main contributions of our work are: (a) a novel four-stream deep neural network that takes advantage of four different temporal skeleton representations to achieve accurate and robust action recognition results, (b) a novel

Grassmannian Pyramid Descriptor (GPD) that captures dynamics of actions from different temporal levels and (c) the use of a meta-learner, which is a network that exploits meta knowledge from the various streams to improve classification accuracy. The proposed action recognition methodology is presented in Figure 14.

In our methodology, we employ two skeletal spatial features. The first type of spatial features is the 3D joint coordinates that are computed based on a common preprocessing scheme, applied to the raw joint coordinates [21][26]. More specifically, all 3D joint coordinates are initially transformed from the world to a person-centric coordinate system by placing the hip centre at the origin. Afterwards, the body part lengths of all skeletons in a dataset are normalised (without changing joint angles) with respect to the corresponding lengths of a reference skeleton that is randomly chosen from the dataset. Finally, the skeletons are rotated in a way that the ground plane projection of the left to right hip vector is parallel to the global x-axis. Such a preprocessing makes skeletons invariant to the absolute location of the human in the scene, scale-invariant and view-invariant respectively.

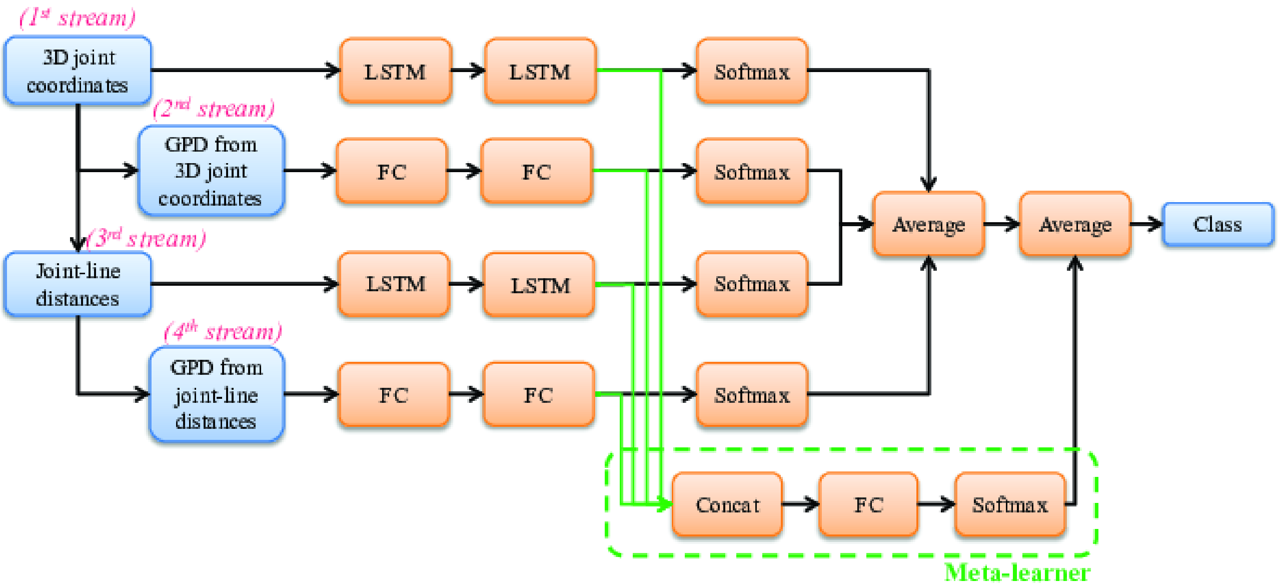


Figure 14: Proposed action recognition methodology [38].

The second type of spatial features that are employed is the joint-line distances [26]. Joint-line distances model the distances from each joint to its projections on the lines formed by every other skeleton joint pair. Given three different joints of a skeleton J_1, J_2 and $J_3 \in R^3$, the distance $d(J_1, J_2 \rightarrow J_3)$ between J_1 and the line formed by J_2 and J_3 is given by employing Heron's formula as follows:

$$d(J_1, J_2 \rightarrow J_3) = \frac{2\sqrt{s(s-d(J_1,J_2))(s-d(J_2,J_3))(s-d(J_1,J_3))}}{d(J_2,J_3)} \quad (1)$$

,where $d(*,*)$ denotes the distance between two 3D joint coordinates and $s = 0.5(d(J_1,J_2)+d(J_2,J_3)+d(J_3,J_1))$. The motivation behind the selection of joint-line distances lies in the fact that they constitute an alternative spatial representation that models the relationship between skeleton joints. As a result, joint-line distances can complement 3D joint coordinates, forming a very descriptive representation that can significantly improve action recognition results. Based on these two spatial features, we propose a four-stream deep neural network, in which the 1st and 3rd streams process the spatial features by directly feeding them to stacked LSTMs in order to derive temporal information, while the 2nd and 4th streams process the GPD features that

are extracted from the spatial features and model the temporal dynamics of these multi-dimensional signals. Afterwards, the GPD features are processed with fully connected (FC) layers in order to enhance their discrimination abilities.

The GPD features are inspired by the LDS theory, which states that the stochastic modelling of both signal dynamics (represented as a time-evolving hidden state process $x(t) \in R^n$) and appearance ($y(t) \in R^d$, where d is the length of the input signal per frame) is encoded by the following two stochastic processes:

$$x(t+1) = Ax(t) + Bv(t) \quad (2)$$

$$y(t) = \bar{y} + Cx(t) + w(t) \quad (3)$$

In Eqs. (2) and (3), $A \in R^{nxn}$ is the hidden state transition matrix, while $C \in R^{dxn}$ represents the mapping of the hidden state to the output of the system. The quantities $w(t) \propto N(0,R)$ and $Bv(t) \propto N(0,Q)$ are the measurement and process noise respectively, while $\bar{y} \in R^d$ is the mean value of the observed data. The LDS descriptor $MLDS = (A,C)$ contains both the appearance information of the observed data modelled by C and its dynamics that are represented by A. Dimitropoulos et al. [25] proposed a higher-order LDS, where a temporal sequence is split in segments, the LDS descriptor of each segment is mapped to a point in the Grassmannian manifold and these points are then clustered to form a Histogram of Grassmannian points (HoGP). The proposed GPD is an extension of the HoGP descriptors as it consists of three levels, where in each subsequent level both the temporal sequence and the window size that splits the sequence in segments are halved (see Figure 15). Individual HoGP descriptors are formed for all seven Grassmannian manifolds before these descriptors are concatenated into a large GPD representation. The motivation behind GPDs is the construction of a temporal representation with the ability to capture dynamics of a multi-dimensional signal (i.e., 3D joint coordinates and joint-line distances in this case) in different temporal resolutions and of different segments. As a result, a temporal sequence can be represented both in coarser levels, achieving robustness to noise, and in finer levels, paying more attention to details. Moreover, the proposed GPD representation can effectively handle temporal scale variations.

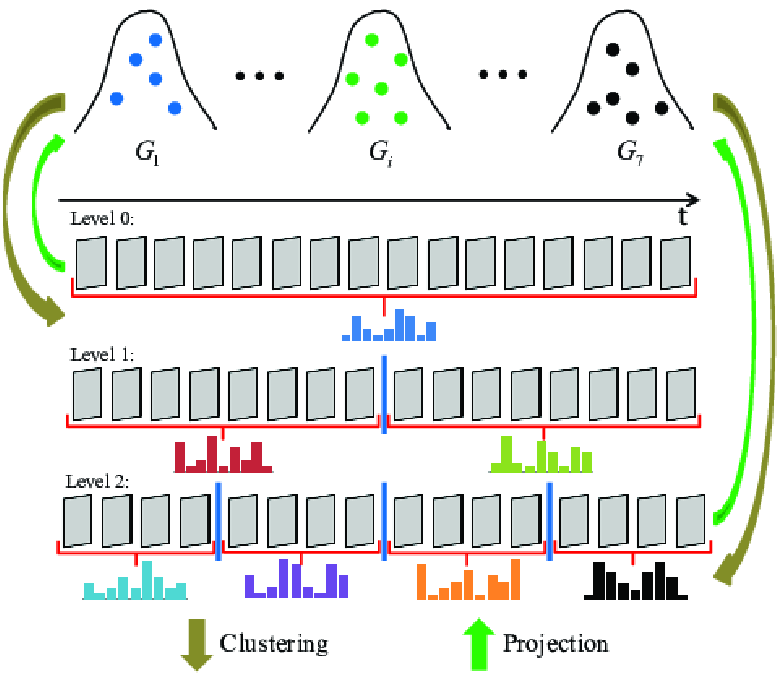


Figure 15: Construction of GPD representation.

The processed skeletal and GPD features are then fed to softmax classifiers in order to derive probabilities for each class, before these probabilities are fused (i.e., averaged) to get an overall prediction. The motivation behind the proposed network is the construction of four different temporal representations of the same skeleton sequence. Given that a single temporal representation may not be descriptive enough for each tested dataset, the use of four complementary temporal representations within a deep network that can weigh them accordingly can assist in improving action recognition results both in the same and across different datasets. Finally, a meta-learner (see dotted outline in Figure 14) is employed that concatenates the temporal features computed from the four streams of the proposed deep network and processes them in order to derive even more discriminative features. These features are then fed to another softmax classifier before the predictions from the meta-learner and the fusion of the four streams are averaged to give the final prediction. Different from previous uses of meta-learner in literature [39], we apply the meta-learner on the features and not the predictions of the deep model. The motivation behind the use of a meta-learner is the fact that a classifier introduces inductive bias, meaning that the classifier's assumptions about a problem and the data can make it effective only on similar types of problems. Although this work deals specifically with skeleton-based action recognition, the variations in skeleton acquisition procedures, number of joints and types of actions introduced by the different datasets can significantly affect classifiers rendering them unable to perform optimally across all datasets. Furthermore, features should be weighted differently when applied on different datasets as their contribution to the action recognition task usually varies depending on the current set of actions that needs to be identified. Thus, the proposed meta-learner is integrated in the proposed deep model, assisting in its optimisation during the training phase. In this way, we enhance the learning procedure and improve the discrimination and generalisation ability of the proposed action recognition methodology.

3.4.2. Sign language recognition

As far as sign language recognition is concerned, our initial work attempts to bridge the gap between direct measurement and vision-based approaches, thus taking advantage of both methods and overcoming their limitations. More specifically, our proposed SLR methodology is based on the processing of video sequences in order to extract accurate body and hand skeletal data that will then be employed for the classification of signs (Figure 16). As a result, our proposed SLR methodology is unobtrusive as it is based only on video sequences without the need for data gloves or other sensors that limit the movements of signers and accurate since it relies on highly discriminative skeletal data.

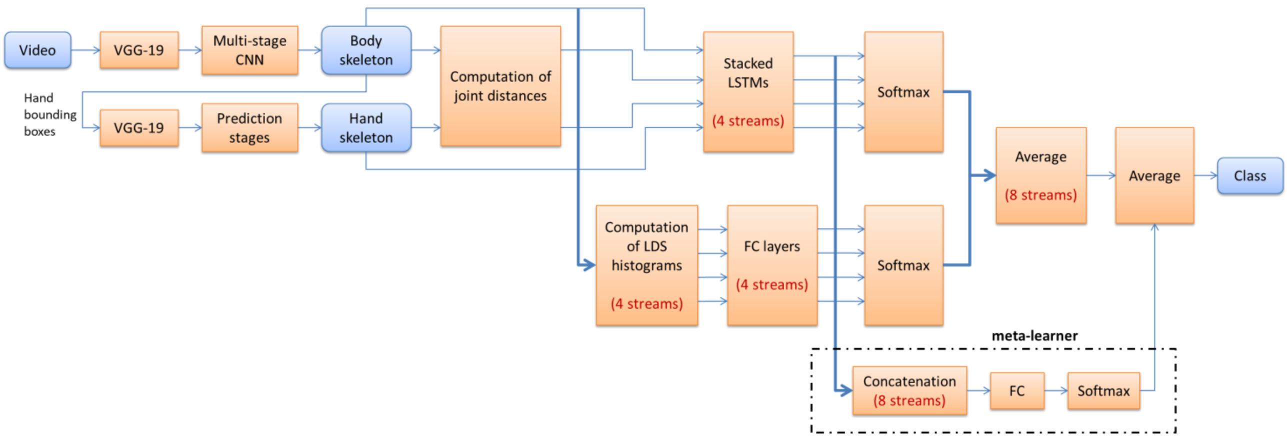


Figure 16: Proposed SLR methodology [40]

The initial processing of the video sequences for the extraction of skeletal data is based on the algorithm developed by [28][41] and known as OpenPose, while their further analysis and classification is based on a proposed robust deep learning architecture, similar to the one

employed in the action recognition problem. More specifically, a pretrained on ImageNet VGG-19 network [42] up to conv4_4 is employed as feature extractor for hand skeleton detection, while the first 10 layers of the same network are employed for body skeleton detection. The outputs of the body and hand skeleton detection networks are 18 body and 21 hand 2D joints, accompanied by confidence scores. As the hand skeleton detector requires a bounding box around the hand, the wrist and elbow positions, computed from the body skeleton detector are employed in order to get an approximate position of the hand location and generate a bounding box.

In our proposed SLR methodology, we employ 12 out of the 18 extracted body skeleton joints as shown in Figure 17. This is due to the fact that i) the signers of a sign language dataset are usually sited and thus the leg skeleton joints are not visible and ii) the leg joints do not provide any valuable information for SLR tasks. Although the employed body skeleton detector produces coordinates for non-visible joints as well, their confidence score is low and thus they are deemed inappropriate by our methodology for robust classification. On the other hand, all hand joints are considered although some of these joints may be occluded by other parts of the hand and thus their confidence scores are low. Another problem that has to be dealt with in the context of sign classification is the fact that some of the gesture classes are signed with the right hand, while others are signed with both hands. To overcome this problem, we consider only the right hand joints for our proposed SLR system. Finally, there are also instances, where the hand skeleton detector is not able to recover the joints of the right hand in some of the frames of a video sequence. In such occasions, we employ the hand joint coordinates of the previous frames in order to fill the missing information.

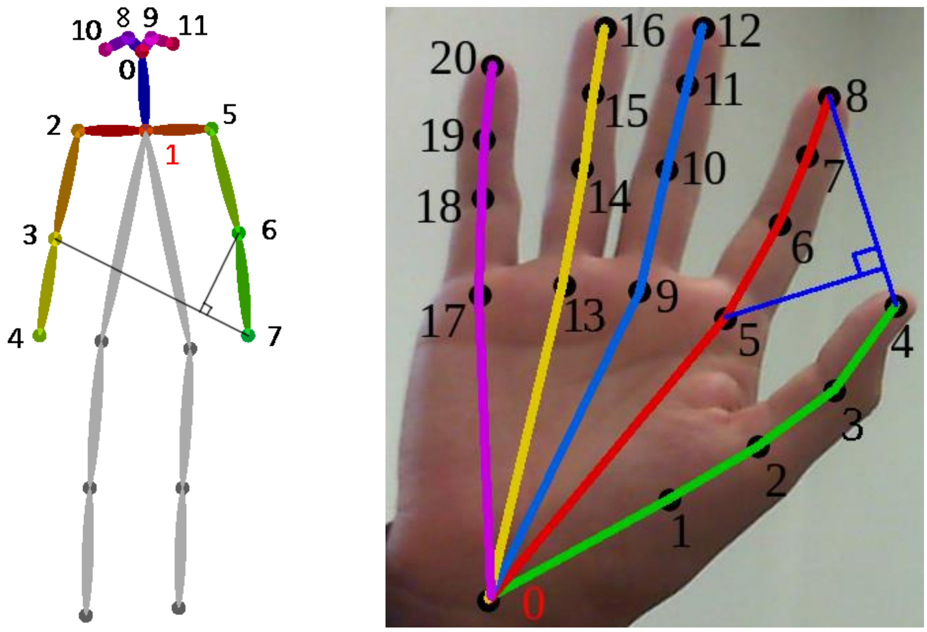


Figure 17: Body and right hand skeleton joints employed by our proposed SLR methodology. The red numbers correspond to the local coordinate system chosen for invariance to human position on the image, while the unused joints are greyed out. Examples of how joint-line distances (Eq. (1)) are computed are also presented.

Before introducing the skeletal features to the proposed skeleton classification network, a preprocessing is required. More specifically, all 2D joint coordinates are initially transformed from the image to a local coordinate system by placing the neck of the body skeleton and wrist of the hand skeleton at the origin (see red colour joints in Figure 17). The purpose of this preprocessing is to make skeletal data invariant to the absolute location of the human in the scene. The skeleton classification network of the proposed SLR methodology is based on two types of spatial features (i.e. relative joint coordinates and joint-line distances of right hand and body skeletons) and a type

of temporal features (GPDs derived from joint coordinates and joint-line distances of right hand and body skeletons). As a result, body and hand joint coordinates and joint-line distances form a four-stream deep neural network that consists of stacked LSTM layers, having as a task to produce descriptive temporal information from the spatial features. The GPD representations form four additional streams that are processed with FC layers in order to derive more discriminative features. The resulting eight streams are finally fed to softmax layers so as each stream produces its own probabilities of a given video sequence to belong to a certain class. These probabilities are averaged and a new probability per class is produced taking into consideration all streams of the proposed skeleton classification network. A meta-learner is additionally employed (see dotted outline in Figure 16), as is the case with the proposed action recognition methodology, in order to further improve the network's accuracy by combining the eight different streams appropriately and producing more discriminative features. In this way, we enhance the learning procedure and improve the discrimination and generalisation ability of the proposed SLR methodology. The probabilities per class computed by the meta-learner are fused (i.e., averaged) with the average class probabilities of the rest of the skeleton classification network, leading to the selection of the most probable class for a given video sequence.

In order to further improve the accuracy and robustness of a SLR methodology, we extended our previous methodology by adding, apart from body and hand skeletal data, image, optical flow and face features [43]. Furthermore, we investigated alternative fusion schemes in order to identify the optimal one that allows for the reliable detection and classification of signs. The new proposed SLR methodology is depicted in Figure 18 and is analysed in detail below.

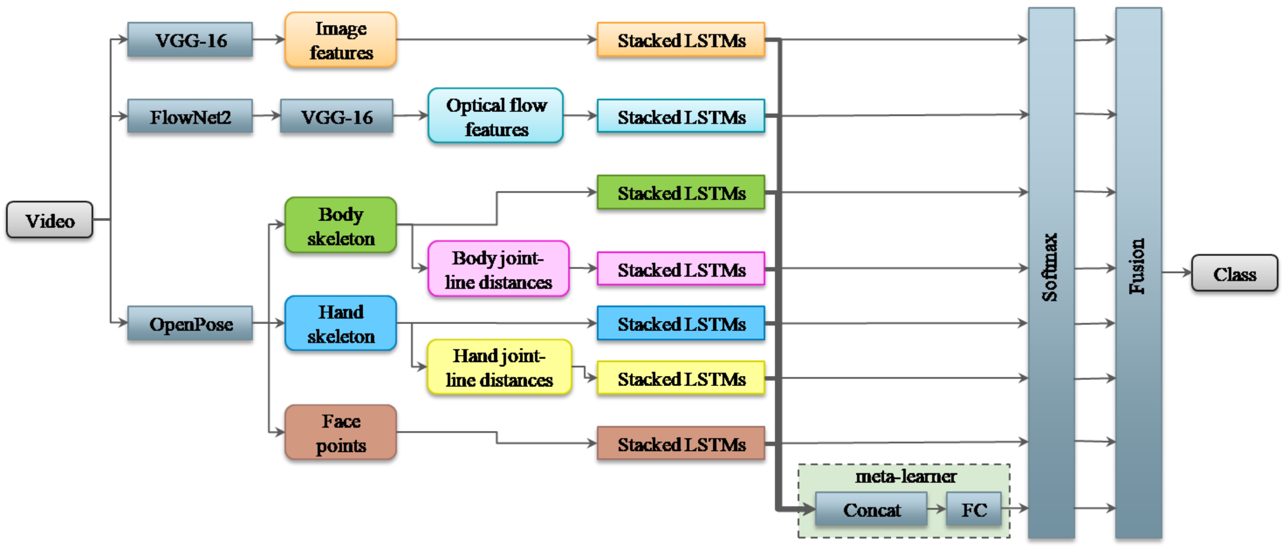


Figure 18: Proposed SLR methodology [43]. Each data stream is shown with a different colour.

The proposed methodology relies on the extraction of video (i.e., image and optical flow) and skeletal (i.e., body, hand and face) features from video sequences. To derive video features, we employ the VGG-16 network [44], pre-trained on ImageNet on both the raw video sequences (i.e., image features) and the optical flow images (i.e., optical flow features). In order to obtain the optical flow images, we employ the well-known and accurate optical flow deep network, FlowNet2 [45]. Regarding the skeletal features, we employ the OpenPose algorithm [28], as in our previous methodology. OpenPose is a deep network capable of producing not only hand and body skeleton joints, but also face points by processing raw videos. The positions of the provided by OpenPose skeletal data are presented in Figure 19.

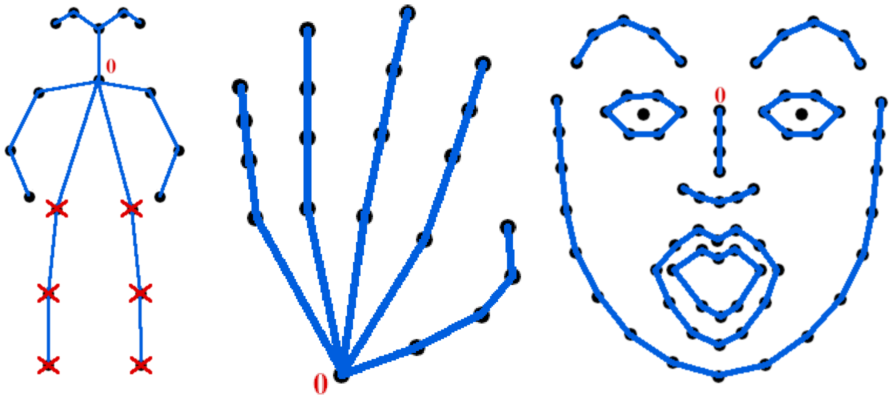


Figure 19: Body and hand skeleton joints and face points extracted from OpenPose [28]. We denote with red zeros the joints chosen as origins of the local coordinate systems and with red crosses the joints that are not taken into account in our proposed SLR method.

The output of OpenPose is 18 body and 21 hand 2D joints and 69 2D face points. Subsequently, we discard 6 body joints, as in our previous method, because firstly a signer is usually sited and thus the leg joints are not visible and secondly the leg joints do not participate in the signing process and thus they do not carry any valuable information. Furthermore, only the right hand skeleton joints are extracted as the right hand is the main signing hand in most tested datasets, although there are signs executed by both hands. Finally, before employing the skeletal data, we normalise their positions by transforming them from image to local coordinate systems. The origins of the local coordinate systems are chosen to be the neck, wrist and upper nose point for the body, hand and face skeleton, respectively (see Figure 19). The purpose of this transformation is to make the skeletal data invariant to the absolute location of the signer in the scene.

As with all previous methodologies presented in this deliverable, we also compute another type of spatial features, namely joint-line distances by employing Eq. (1). We found that joint-line distances can complement the other feature representations, forming an additional descriptive spatial representation that models the relationship between joints. However, we do not compute joint-line distances for the face points as the number of computed face points is quite large, thus leading to an enormous amount of face joint-line distances (i.e., over 150k distances). As a result, the computational complexity for the processing and classification of such enormous vectors would be really high. Therefore, seven data streams are created from the processing of the raw video sequences. These streams are fed to stacked LSTMs, which are several LSTM units put one after the other. These units are individually optimised to achieve best performance for the SLR problem at hand. A meta-learner is also employed, as in the previous methodologies, and concatenates the features computed from the stacked LSTMs before processing them a bit further to derive even more powerful and discriminative features by using a FC layer. In this way, we enhance the learning procedure and improve the discrimination and generalisation ability of the proposed SLR system. The seven data streams, along with the meta-learner stream, form a set of eight classifiers that are then fed to softmax layers so as each of these classifiers produces its own probabilities that a given video sequence belong to a certain class.

To fuse the aforementioned probabilities, in our initial SLR methodology [40], we proposed the averaging of the data streams, the computation of an overall probability and then again, the averaging of this probability with the probability of the meta-learner in order to obtain the final probability per class. In the extended SLR method [43], we not only employ the aforementioned averaging fusion scheme, which we name AV in short notation, but we also investigate other fusion schemes so as to find the optimal way to combine the eight streams and improve the performance of the proposed SLR methodology.

To this end, we test majority voting (MV), which is a well-known technique that accepts as class of a tested video sequence, the one with the most votes from the employed classifiers. Furthermore, inspired by the work of [24], we employ Dynamic Score Combination (DSC) [46][47] and Particle

Swarm Optimisation (PSO) [48]. DSC attempts to combine the individual probabilities in a way that the combined probability distribution exhibits a larger separation than the probability distribution produced by the individual classifiers. Since DSC is employed in two-class optimisation problems and our problem is multi-class, we had to adapt DSC slightly so that each class is compared against all other classes. Given the probability of a classifier for a single class p_i, with $i = 1...8$, the overall probability based on DSC is given by:

$$p_{DSC} = (1-\beta)\min_i \{p_i\} + \beta \max_i \{p_i\}$$

where β is the combination weight, defined by the mean rule:

$$\beta = \frac{1}{m}\left(\sum_{i=1}^{m} p_i\right)$$

and m is equal to the number of classifiers (i.e., $m = 8$ in our case). Another fusion scheme is the PSO, which is a global optimisation algorithm, motivated by social behaviour of organisms such as bird flocking and fish schooling. In this context, the overall probability of a class is given by the weighted aggregation of the individual probabilities as:

$$p_{PSO} = \sum_{i=1}^{m} w_i p_i$$

The PSO algorithm considers each single solution w_i, with $i = 1...8$, as a particle in the search space and associates this particle with a fitness value and velocity, which direct the movement of the particle. In each iteration, the algorithm tries to improve a candidate solution with regard to a given measure of quality (i.e., the fitness function to be optimised), while the particles move in the problem space by following the current optimum particle. In our case, the fitness function is the accuracy of a solution w_i when applied on the training set of a given dataset. Finally, we propose another fusion scheme that we call Deep Weight Averaging (DWA). This scheme attempts to optimise a set of weights, exactly like in the previous equation, but instead of a PSO algorithm, a deep network is employed that accepts as inputs the individual probabilities and learns a set of weights that can optimise the overall probability.

3.5. Results

In this section, we present and analyse the results of employing our methodologies in benchmark action and sign language recognition datasets. Furthermore, we compare the performance of our proposed methodologies with other state-of-the-art methods and draw conclusions.

3.5.1. Action recognition

In the context of action recognition, four datasets are employed for the evaluation of our proposed methodology. The selection of these datasets is based on their different characteristics (i.e., types of actions, number of joints, etc.) that pose challenges to a general action recognition method. Furthermore, the small size of two of these datasets introduces difficulties to the training of a deep neural network that usually requires an abundance of training samples. The tested datasets and their characteristics are shown below:

UT-Kinect dataset [20]: This dataset consists of 10 actions performed twice by 10 different subjects. Each skeleton consists of 20 joints. For the evaluation of this dataset, we follow the

cross-subject test setting of [21], in which 10 folds are created, where half of the subjects are used for training and the remaining half for testing.

Florence3D dataset [17]: This dataset consists of 9 actions performed two or three times by 10 different subjects. Each skeleton consists of 15 joints. For the dataset evaluation, we follow the cross-subject test setting of [21], in which 10 folds are created, where half of the subjects are used for training and the remaining half for testing.

G3D dataset [49]: This dataset consists of 20 actions performed three times by 10 different subjects. Each skeleton consists of 20 joints. For the evaluation of this dataset, we follow the protocol of [25], in which the first instance of each action per subject is used for training and the other two instances are used for testing.

MSRC-12 dataset [50]: This large dataset consists of 30 actions performed by 12 subjects. Each skeleton consists of 20 joints. For the dataset evaluation, we follow two protocols; a cross-subject protocol [27], where the odd subjects are used for training and the even subjects for testing and a modality-based "leave-persons-out" protocol [23], in which all but one subjects are used for training and the remaining subject for testing for each modality (i.e., video, image, text, video-text, image-text). The ground truth annotation of this dataset is based on [18].

The skeleton sequences of all datasets are processed so that they are composed of 64 frames either by removing intermediate frames in the case of larger sequences or by adding interpolated intermediate frames in the case of smaller sequences. The parameters that affect our proposed action recognition methodology (i.e., size of LSTM and FC layers, dropout percentage, learning rate, etc.) are determined after experimentation on the UT-Kinect dataset and kept fixed for the other datasets. In this way, we want to point out the advantages of the proposed features and the meta-learner on the performance of our methodology, no matter which dataset is employed. More specifically, the two-layer LSTMs consist of 1024 and 256 neurons and dropout/recurrent dropout equal to 0.1 and 0.2 respectively. Furthermore, the fully connected layers (FCs) that are fed with the GPDs consist of 512 and 128 neurons respectively, while the FC layer of the meta-learner consists of 128 neurons. The window size for the computation of GPDs is halved in each subsequent level from 16 to 4 frames. Finally, the network is implemented in Keras-Tensorflow framework and trained using the Adam optimiser with batch size of 32 and learning rate equal to 0.0001.

Our proposed methodology is compared with 15 state-of-the-art action recognition methods across four datasets. Table 2 and Table 3 evaluate the performance of our methodology on the large MSRC-12 dataset. It can be observed that our method outperforms all other state-of-the-art methods in both evaluation settings and in all modalities, achieving a significant boost on the accuracy. More specifically, our methodology improves the state-of-the-art results by 1.53% and 3.4% when the cross-subject and modality-based "leave-persons-out" protocol are employed respectively.

Table 2: Classification accuracy on MSRC-12 using cross-subject protocol [27].

Method	Accuracy
ConvNet+JTM [27]	**93.12%**
Ker-RP [50]	**92.3%**
Cov3DJ [18]	**91.7%**
ELC-KSVD [19]	**90.22%**
Proposed [38]	94.65%

Table 3: Classification accuracy on MSRC-12 using "leave-persons-out" protocol [23].

Modality \ Method	Sharaf et al. [52]	Meshry et al. [53]	Patrona et al. [23]	Proposed [38]
Video	0.669 ± 0.082	0.895 ± 0.068	0.927 ± 0.009	**0.969 ± 0.069**
Image	0.598 ± 0.082	0.858 ± 0.086	0.894 ± 0.010	**0.944 ± 0.091**
Text	0.558 ± 0.092	0.788 ± 0.139	0.851 ± 0.012	**0.871 ± 0.165**
Video-Text	0.684 ± 0.074	0.921 ± 0.126	0.983 ± 0.008	**0.992 ± 0.024**
Image-Text	0.687 ± 0.099	0.894 ± 0.085	0.905 ± 0.007	**0.956 ± 0.089**
Overall	0.639	0.871	0.912	**0.946**

Similar performance improvement is noticed for the other tested datasets as well, although their small sizes pose challenges to the accurate training of our proposed deep neural network. From Table 4 and Table 5, it can be observed that our proposed action recognition methodology outperforms the Lie Group method [21] by 0.61% and 0.24% on the UT-Kinect and Florence3D datasets respectively. Moreover, Table 6 shows that our proposed methodology outperforms the Sh-LDS-HoGP method and other classification approaches by at least 1.63%, meaning that the proposed features are more descriptive of the underlying actions of the G3D dataset than the HoGP features. The superb performance of the proposed methodology across all tested datasets reveals the splendid ability of the meta-learner to weigh the different features in a way that makes our method achieve similar performance irrespective of the tested dataset. At this point, it is worth noting that the hyper-parameters of the proposed deep network are kept fixed after their optimisation with respect to the UT-Kinect dataset. As a result, we can conclude that the proposed methodology generalises well on other datasets without requiring additional hyper-parameter tuning.

Table 4: Classification accuracy on UT-Kinect dataset.

Method	Accuracy
Lie Group [21]	97.08%
Histogram of 3D joints [20]	90.92%
Random forests [54]	87.9%
Proposed [38]	**97.69%**

Table 5: Classification accuracy on Florence3D dataset.

Method	Accuracy
Lie Group [21]	90.88%
Multi-Part Bag-of-Poses [17]	82.00%
Proposed [38]	**91.12%**

Table 6: Classification accuracy on G3D dataset. All methods were taken from [25].

Method	Accuracy
Sh-LDS-HoGP [25]	90.75%
Restricted Boltzmann Machine	84%
Hidden Markov Model	77.4%
Conditional Random Fields	69.25%
Dynamic Time Warping	57%
Proposed [38]	**92.38%**

Moreover, we analyse the effect of our contributions on the classification accuracy of our proposed action recognition methodology on the UT-Kinect dataset. From studying Table 7, we can observe the huge boost on the classification accuracy of the proposed methodology when the novel GPDs are employed. More specifically, an improvement of 18.6% is observed when the HoGP features are substituted with the GPDs extracted from the 3D joint coordinates. A similar improvement is noticed in the case of GPDs extracted from joint-line distances. This means that the proposed GPDs are successful in their task of enhancing the discrimination ability of the proposed deep network. Finally, the introduction of the meta-learner in the proposed deep network leads to a better exploitation of the meta knowledge derived from the four network streams and improves the classification accuracy of the proposed methodology by almost 1.35%.

Table 7: Experimentation with proposed contributions on UT-Kinect dataset.

Contributions	Accuracy
HoGP [25] from 3D joint coordinates	68.55%
GPD from 3D joint coordinates	**81.31%**
HoGP [25] from joint-line distances	60.60%
GPD from joint-line distances	**78.80%**
Proposed [38] without meta-learner	96.38%
Proposed [38] with meta-learner	**97.69%**

3.5.2. Sign language recognition

In the context of sign language recognition, two datasets are employed for the evaluation of our proposed methodologies, both the initial version [40] and the extended one [43]. The tested datasets and their characteristics are shown below:

LSA64 dataset [32]: This dataset is a large Argentinean sign language dataset that consists of 10 subjects, executing 5 repetitions of a total of 64 different types of signs. As a result, the LSA64 dataset comprises 3200 videos of different length (i.e., number of frames). For the experimental evaluation of the proposed SLR methodologies, all video sequences are processed so that they are composed of 48 frames each. This is achieved by employing a spline interpolation technique among the given frames of a video sequence. The experimental setup for the LSA64 dataset is based on [55]. More specifically, the dataset is split randomly in a training set consisting of 80% of

all samples and a test set consisting of the remaining 20% of the samples. This procedure is repeated 5 times, where in each iteration, a different split of the dataset is performed.

RWTH-PHOENIX dataset [56]: This dataset was basically created for continuous SLR. However, there is a part of the dataset, called Signer03 Cut-out Gloss Recognition that allows for experiments in the context of isolated SLR. We take advantage of this setup, but we process the dataset by discarding a few samples in order to be more suitable for deep learning training. More specifically, we discard classes with fewer than 10 samples and we also discard samples from classes with over 50 samples. Furthermore, we process all video sequences in order to consist of 10 frames each. As a result, the final processed dataset that we employ consists of 50 classes with 10-50 samples per class and 1297 and 238 training and test video sequences respectively. The video sequences of the RWTH-PHOENIX dataset are already split in training and test sets. In our experimental setup, we repeat the training of our proposed SLR methodologies for 5 times, where the weights of the deep network are randomly re-initialised after each repetition.

The reason behind the selection of these datasets lies in their special characteristics for a deep learning training. The LSA64 dataset is a large and balanced sign language dataset with several frames per video sequence and thus it is suitable for a deep learning framework. By utilising this dataset, we want to unravel the full potential of a deep network. On the other hand, despite our changes, the second dataset remains a highly unbalanced dataset with few frames per video sequence and cases where some of these frames are blurry. As a result, the second dataset is quite challenging for a deep learning algorithm and by using it we want to observe how well our proposed SLR system can cope with problematic datasets.

The optimisation of the hyper-parameters that affect the performance of the proposed SLR methodologies is performed after experimentation on the training sets of the LSA64 and RWTH-PHOENIX datasets individually. These hyper-parameters define the size and number of stacked LSTM units, size of the FC layer, dropout percentage, batch size and learning rate. More specifically, one- or two-layer LSTMs are considered, consisting of 128, 256, 512 or 1024 neurons, while the dropout percentage is in the range [0.0-0.5]. Similarly, the size of the FC layer is selected after experimentation among the values of 128, 256, 512 and 1024. These hyper-parameters vary significantly based on the features fed on each stream and the dataset. Furthermore, for the image and optical flow features, we get the output of the last layer of the VGG-16 network, which is a 1024-element vector. Finally, the network is implemented in Keras-Tensorflow framework and trained using the Adam optimiser with batch size of 32 and learning rate equal to 0.0001.

Our proposed methodologies are assessed on the two previously mentioned datasets and compared against three state-of-the-art SLR methods. Furthermore, the contributions of the various employed features to the performance of the proposed SLR methodologies are evaluated. Table 8 presents the contributions of the features employed from our initial SLR methodology [40] on its performance on the LSA64 dataset.

Table 8: Evaluation of the employed features on the LSA64 dataset.

Method	Accuracy(Mean ± Std)
Body features	93.91 ± 1.24
Hand features	91.64 ± 1.01
Deep Network without meta-learner	97.16 ± 0.57
Proposed Deep Network [40]	**98.09 ± 0.59**

From Table 8, a few conclusions can be drawn. Firstly, the body features constitute a slightly better representation than the hand features for sign language recognition since they achieve a 2.27% increase in sign language recognition on the LSA64 dataset. This is attributed to the fact that the

body joints are more reliably and robustly detected than the hand joints. Accurate hand joint detection suffers from occlusions and overlaps between the fingers and as a result, no detector can reliably infer the locations of non-visible joints. This can also be observed by the low confidence scores the employed hand skeleton detector produces. However, the employment of both hand and body skeletal features is beneficial for the SLR task. Hand skeletal joints contain valuable knowledge that can complement the information body skeletal joints provide and therefore, their combined use gives a boost to the performance of our initial proposed SLR methodology as shown by the increase of 3.25% in the accuracy achieved on LSA64 dataset. It is also worth mentioning that although our initial proposed SLR methodology does not employ any information from the left hand, it manages to successfully classify both one-handed and two-handed signs of the LSA64 dataset, demonstrating the discrimination power of the employed features.

Moreover, the use of a meta-learner is beneficial to the performance of our initial proposed SLR methodology. This can be attributed to the construction of highly discriminative features by the employed meta-learner based on the corresponding features each data stream produces. In this way, the meta-learner exploits the derived meta-knowledge, enhances the learning procedure and improves the discrimination and generalisation ability of the initial proposed SLR methodology [40].

Next, we extend the initial SLR methodology and propose a new SLR methodology [43] by adding additional image and optical flow features and investigating several fusion schemes. In Table 9, we present the experimental evaluation of the individual feature representations, the meta-learner and the proposed fusion schemes in the LSA64 and RWTH-PHOENIX datasets in order to identify the optimal way of combining the information from the different data streams and the meta-learner.

Table 9: Evaluation of individual feature representations and fusion schemes in the performance of the proposed SLR methodology [43].

Feature	Dataset results (mean ± std)	
	LSA64	RWTH-PHOENIX
Image	**99.37 ± 0.25**	59.66 ± 1.39
Optical flow	98.81 ± 0.49	39.24 ± 2.11
Body skeleton	91.06 ± 1.09	33.28 ± 2.48
Body joint-line distances	93.34 ± 2.23	42.52 ± 2.82
Hand skeleton	85.88 ± 1.48	29.58 ± 2.27
Hand joint-line distances	95.19 ± 0.36	44.79 ± 1.5
Face	18.22 ± 1.54	19.66 ± 1.77
Meta-learner	97.94 ± 1.03	**60.76 ± 3.21**
Fusion		
AV	99.19 ± 0.47	64.29 ± 2.93
MV	99.81 ± 0.17	64.87 ± 1.8
DSC	**99.84 ± 0.19**	67.98 ± 1.86
PSO	99.8 ± 0.06	66.49 ± 2.32
DWA	99.72 ± 0.26	**69.33 ± 1.57**

From Table 9, it can be deduced that the most discriminative features are the image and the optical flow features, revealing the power of the pre-trained VGG-16 network. Furthermore, the joint-line distances seem to constitute more powerful representations than the raw skeleton joints, thus justifying our choice to employ them for our proposed SLR methodology. On the other hand, it can be observed that the face features perform poorly in the SLR task. This is something to be

expected as the face features alone are not adequate enough to differentiate signs. Their purpose is mostly complementary in order to enhance the performance of other more descriptive feature representations, such as hand and body skeletal data. Finally, the meta-learner is successful in its task of combining the various data streams in an attempt to produce even more powerful features. This can be more clearly observed in the RWTH-PHOENIX dataset, where the features that the meta-learner provides outperform all individual feature representations.

The evaluation of the proposed fusion schemes cannot give a clear view of the optimal one. From Table 9, one can observe that the AV fusion scheme that was proposed in [40] under-performs with respect to the other fusion schemes. On the other hand, MV has the limitation that it does not take into account the accuracy of the individual classifiers. As a result, although it performs quite well in the LSA64 dataset because all classifiers are really accurate, it performs relatively poorly in the RWTH-PHOENIX dataset, where all classifiers have mediocre performance. The DSC fusion scheme performs optimally in the LSA64 dataset, while our proposed DWA method performs optimally in the RWTH-PHOENIX dataset. This shows the power of deep learning in not only producing discriminative features, but also weighing features appropriately in order to achieve improved results. It is also worth noting that the PSO algorithm is quite sensitive to its initialisation and therefore, we executed it 5 times and obtained its mean accuracy.

Additionally, both our initial and extended SLR methodologies are compared with the state-of-the-art SLR methods presented in [55]. In [55], the authors proposed a SLR system based on the output of two classifiers, one for each hand. The classifier for each hand receives as input a sequence of cropped hand regions and normalised hand positions and employs three sub-classifiers that each use position, movement and hand-shape information. The outputs of these sub-classifiers are merged to a final probability, stating in which class a given hand gesture sequence belongs to. The authors developed their sub-classifiers in a way to be sequence agnostic, meaning that they do not rely much on the correct sequence of the hand gestures and they called their method ALL. Furthermore, they employed two more variants of their method, one of which employs HMMs with Gaussian Mixture Models output probabilities (ALL-HMM) and the other transforms their features to binary ones and then employs one-versus-all multi-class Support Vector Machines (ALL-BF-SVM). Table 10 compares the performance of our SLR methodologies with the SLR methods, proposed in [55], on the LSA64 dataset.

Table 10: Experimental evaluation of proposed methodologies on the LSA64 dataset.

Method	Accuracy (Mean ± Std)
ALL-BF-SVM [55]	95.08 ± 0.69
ALL (sequence agnostic) [55]	97.44 ± 0.59
ALL-HMM [55]	95.92 ± 0.95
Initial SLR methodology [40]	98.09 ± 0.59
Extended SLR methodology [43]	**99.84 ± 0.19**

Although the RWTH-PHOENIX dataset has been evaluated in the context of continuous SLR, no isolated SLR method has been applied yet. This reveals again one of the problems of SLR, which is the unavailability of significant experimental evaluation on the same dataset. To overcome this problem, we test both proposed SLR methodologies on the RWTH-PHOENIX dataset and present the results in Table 11.

Table 11: Experimental evaluation of proposed methodologies on the RWTH-PHOENIX dataset.

Method	Accuracy (Mean ± Std)
Initial SLR methodology [40]	56.13 ± 2.33
Extended SLR methodology [43]	**69.33 ± 1.57**

An analysis of Table 10 and Table 11 reveals that both proposed methodologies significantly outperform all other state-of-the-art methods in the LSA64 dataset and, in fact, the extended SLR method [43] reaches an almost perfect accuracy. This reveals the power of employing several alternative feature representations and a reliable fusion scheme that can boost the performance of a classification procedure even further. Similar conclusions can be drawn from the performance of the proposed methodologies in the RWTH-PHOENIX dataset, where we observe that the use of additional features and a more appropriate fusion scheme is beneficial to the performance of a SLR algorithm.

A comparison of the performance of our extended SLR method between the two datasets can also be performed. One can observe that our method reaches an almost perfect accuracy in a balanced and large dataset (i.e., LSA64 dataset), but it achieves mediocre performance in a problematic dataset, such as the RWTH-PHOENIX. However, although the RWTH-PHOENIX dataset is not so suitable for a deep network, our SLR method does a fine job classifying it, achieving almost 70% accuracy among 50 classes. This fact reveals the discrimination power of the proposed feature representations, meta-learner and fusion schemes and it is the main reason we chose to test our SLR method in such a dataset.

Finally, Figure 20 visualises hand, body and face features extracted from frames of video sequences in the LSA64 and RWTH-PHOENIX datasets by employing the OpenPose algorithm. On the other hand, Figure 21 presents examples of image and optical features extracted by employing the VGG-16 network [44] in the RWTH-PHOENIX dataset. All these features constitute the input to our proposed SLR methodologies before the LSTM, FC and softmax units are employed for feature processing, new feature extraction and sign language classification.

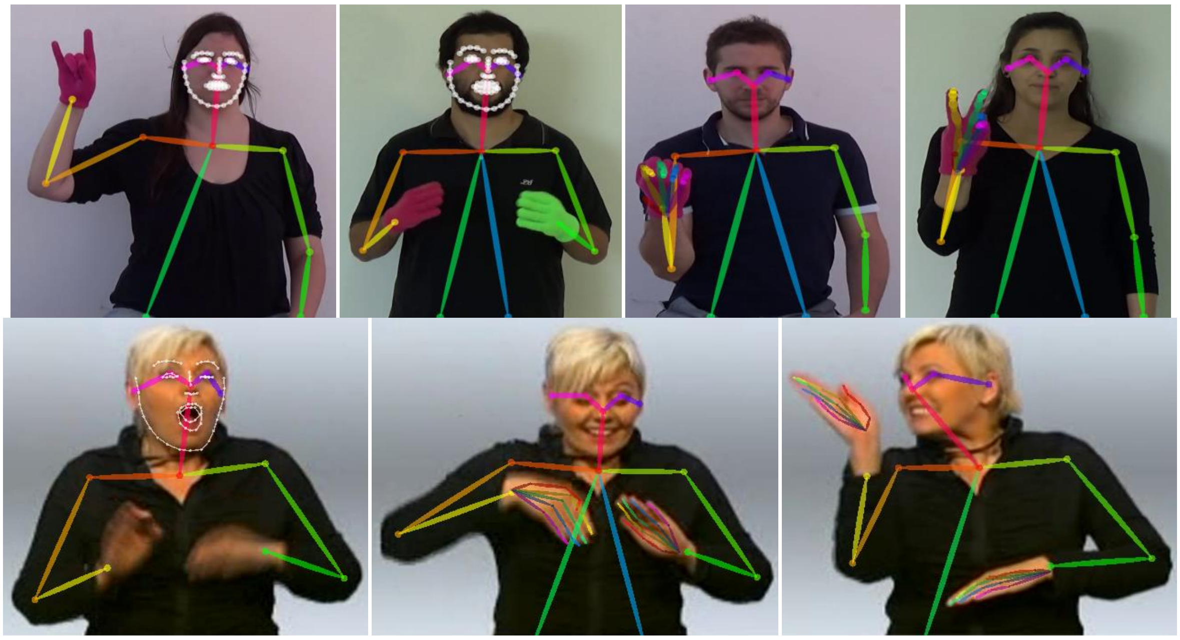


Figure 20: Extracted body/hand and body/face features from LSA64 (first row) and RWTH-PHOENIX (second row) datasets as the computational complexity of detecting all three skeletal features simultaneously was too heavy for our setup.

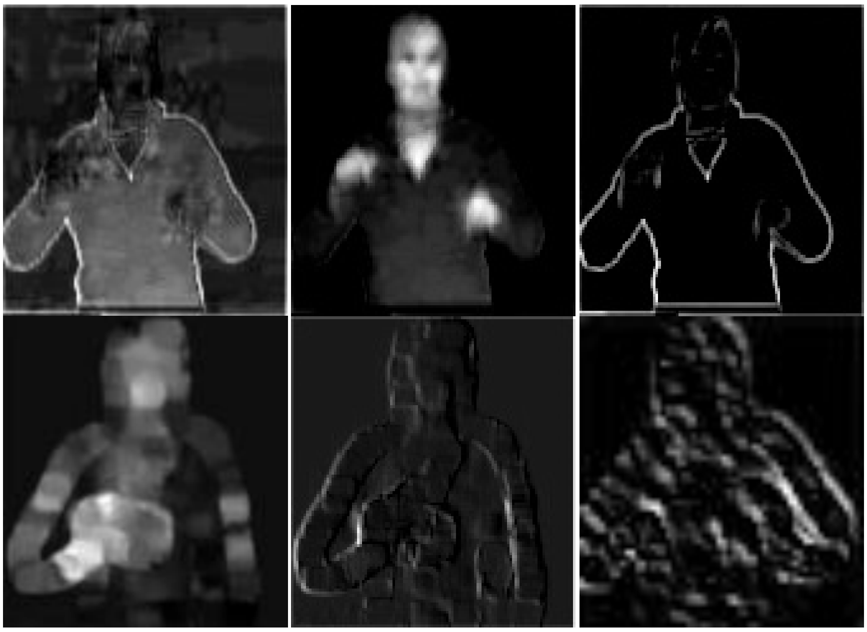


Figure 21: Image (first row) and optical flow (second row) features after the processing with the VGG-16 network [44] from the RWTH-PHOENIX dataset.

3.6. Discussion

This deliverable presents our work during the first two years of the EasyTV project regarding the tasks of feature selection and extraction for the problems of action and sign language recognition. To this end, we propose various methodologies to extract image, optical flow and skeletal features from video sequences and analysed several fusion schemes that can be employed in order to optimally fuse the information derived from the various features in order to improve the performance of action and sign language recognition methodologies.

The purpose of this research is to study the advantages and limitations of the various features that can be extracted from video sequences in order to find out which of these features can be employed to build a reliable and accurate sign language and gesture recognition system. Furthermore, we analysed various deep learning network architectures and we found out that the proposed meta-learner is beneficial to the performance of a classifier. As a result, a meta-learner can also be employed in the context of developing an accurate gesture remote control system. On the other hand, although skeletal features are usually more reliable and discriminative than image features, their extraction using OpenPose is too computationally expensive and thus inappropriate for the development of a real-time gesture recognition system. Therefore, in the case we want to employ skeletal information, we should either rely on RGB-D sensors (i.e., Kinect, ORBBEC, Intel RealSense, etc.) that can provide body, but not hand skeletal features or develop a faster version of the OpenPose algorithm. Finally, we also investigated fusion schemes in the case that we want to rely on more than one features for the task of gesture recognition. In conclusion, the knowledge acquired during the development of the proposed action and sign language recognition methodologies can be of paramount importance to the development of a gesture remote controller, as it is required in the framework of the EasyTV project.

4. THE EASYTV GESTURE/GAZE REMOTE CONTROL

In this section, we initially present a literature review of sensors capable of capturing gesture and gaze information and we then describe the final version of the proposed gesture/gaze remote control, developed in the framework of the EasyTV project. The proposed gesture/gaze remote control detects accurately hand and eye movements, which translates to commands used for the control of the TV set.

4.1. State-of-the-art hand and eye movement detection sensors

To accurately and robustly detect and classify hand movements, RGB-D sensors are usually employed. These sensors are capable of extracting colour and depth information from the scene. Two well-known RGB-D sensors that are usually used for the development of gesture recognition applications, are presented in Figure 22. Additionally, either directly or by using sophisticated machine learning algorithms, such as OpenPose [28], the extraction of skeletal data is possible. The need for capturing data with a nominal framerate of 30 fps and the high computational burden of machine learning algorithms for skeleton extraction limit the use of such sensors to PCs with at least mediocre system specifications (i.e., Windows 10, at least dual-core CPU and a GPU with at least 2 GB of internal memory or vram).

Figure 22: Microsoft Kinect v2 (left) and Intel RealSense (right) RGB-D sensors.

As far as eye movement detection is concerned, two main categories of algorithms can be identified. The first category consists of sophisticated gaze tracking hardware setups that include commercially available sensors that use infrared illumination to achieve accurate and time efficient gaze tracking. A few representative sensors are Tobii [57], SMI [58], EyeTech [59] and Mirametrix [60]. A significant advantage of such sensors is their robustness to the presence of glasses, the eye characteristics (i.e., colour, age) and the environmental lighting. Unfortunately, a prohibitive factor for the extensive use of such specialized equipment is their significantly high cost. Lately, other gaze tracking sensors, such as the myGaze Eye Tracker [61], have been proposed that are more affordable and thus they can be more easily adopted for everyday purposes.

Figure 23: Commercial gaze tracking sensors: Tobii (up left), EyeTech (up right) and myGaze Eye Tracker (down).

In order to satisfy the needs for low-cost eye movement detection and portability, the second category of gaze recognition algorithms consists of methodologies that rely on the use of a single camera and image processing techniques. Such methodologies rely on the extraction of facial features or landmarks from the person before the eye locations are identified and classified using template-based or appearance-based techniques. Wang et al. perform real-time gaze recognition by initially performing 3D face reconstruction and iris and pupil detection [62]. Similarly, Vincente et al. in [63] detects facial features that are then used for 3D head pose estimation and eye centre detection. On the other hand, Zhang et al. in [64] compute 3D head rotation and iris position by detecting facial landmarks and fitting them on a generic face model. Afterwards, they employ CNNs on eye images for gaze estimation. Gaze recognition methods that are based on image processing from single cameras are more flexible but they are sensitive to noise and occlusions and thus they usually cannot achieve the high accuracy of dedicated gaze tracking sensors, thus sacrificing performance for affordability.

4.2. Set of predetermined gestures

As far as the commands that should be interpreted with gestures are concerned, we are aware of the presence of the two following standards:

- H.IPTV-EUIF.1 "Enhanced UI framework for IPTV terminal device – Gesture Control Interface": Updated Draft (Macau, China, 16-27 October 2017)
- ISO/IEC 30113-11:2017(en) "Information technology – Gesture-based interfaces across devices and methods – Part 1: Single-point gestures for common system actions"

Unfortunately, there are two main reasons these standards are hard to be applied in the EasyTV project. Firstly, these standards are not publically available and can be acquired either by paying or by being in a specific network of related businesses. Secondly, we performed a research on available gesture recognition systems for TV control and we found out that there is little consistency among TV manufacturers as each manufacturer implements their own set of commands with different gestures. An example is which gesture TV manufacturers use to define the opening of the TV, as shown in Figure 24. LG asks to lift our hand near the face with the index finger pointing upwards [65]. On the other hand, Samsung asks to raise our hand and wave at the TV [66], while Sony asks to swipe our hand over their developed remote controller [67]. As a result, we decided to implement our own set of gestures, getting inspiration from TV manufacturers, performing tests on EasyTV users and relying on our experience and expertise to propose an intuitive set of simple hand gestures that every user can easily perform with a high level of accuracy.

Figure 24: Proposed gestures to open TV. From left to right, LG, Samsung and Sony propositions [65][66][67].

4.3. Proposed gesture/gaze remote control

The purpose of the proposed gesture/gaze remote control is to comprise an interface for the remote control of the TV set. To this end, the proposed remote control sets up a client application that transmits TV commands as JSON messages to the server application, which is the TV by utilizing the HbbTV protocol. As a result, the gesture/gaze remote control can be set up on a PC, which consists a second screen for the communication with the TV. The gesture/gaze remote control ıs developed in C#. The remote control detects, processes and classifies skeletal and gaze

information, captured by supported sensors, translates them to predefined TV commands and transmits them to the TV set. The main architecture of the gesture/gaze remote control is presented in Figure 25.

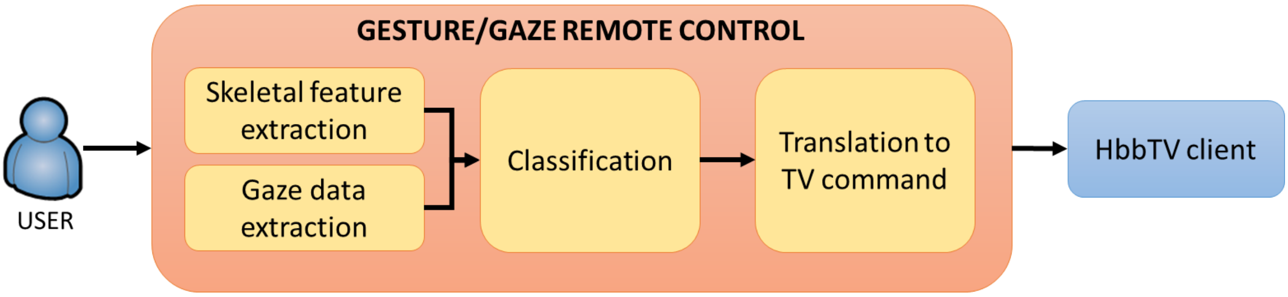


Figure 25: Main architecture of the proposed gesture/gaze remote control.

The proposed remote control currently supports Kinect v2 and RealSense RGB-D sensors, which can provide skeletal information either directly (Kinect v2) or by using commercial software, such as NuiTrack (RealSense). As far as eye tracking is concerned, the proposed application currently supports the myGaze eye tracking sensor in order to take advantage of the high accuracy and time efficiency for real-time applications. However, the remote control application has been rewritten with respect to the preliminary version in order to facilitate the addition of new RGB-D and eye tracking sensors with ease and thus the currently supported sensors do not limit the usability of the proposed gesture/gaze remote control.

The cost of the current setup in order for the gesture/gaze remote control to be fully operational depends on the cost of the sensors and the software used for the extraction of skeletal data. The cost of a RealSense sensor is at the range 149-199 euros, while the cost of the myGaze eye tracker is 499 euros. Additionally, NuiTrack software library is necessary for skeleton extraction from RealSense, as Intel does not currently support such functionality. To this end, a yearly licence of the NuiTrack software costs 40 euros, while a perpetual licence costs 100 euros. As a result, the final cost of the current setup for the gesture/gaze remote control is at the range 688-798 euros.

The GUI of the final version of the gesture/gaze remote control has significantly changed with respect to the preliminary version and it has been designed to be intuitive and well organised for the users. This means that buttons with relative functionalities have been placed in groups, one next to the other. A view of the GUI of the gesture remote control is presented in Figure 26 and all functionalities are explained in detail below.

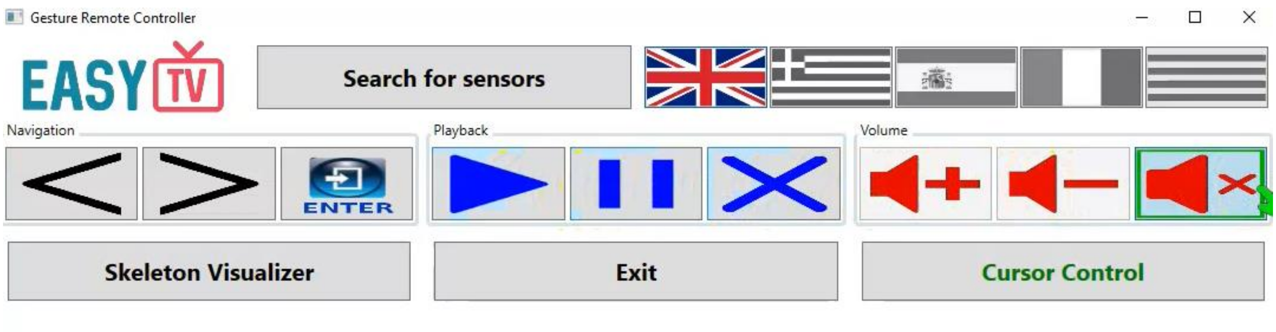


Figure 26: View of the GUI of the final version of the gesture/gaze remote control.

The name of the application, along with the logo of the EasyTV project are clearly presented in the upper left corner of the GUI of the gesture/gaze remote control. The first row of buttons consists of the button “Search for sensors” that allows the application to search for supported RGB-D and eye tracking sensors that are connected to the PC in order to be utilized for the remote control of the TV set. Furthermore, there are also buttons, disguised as flags, in the first row that allows the selection of the language of the user. The gesture/gaze remote control has been designed to be multi-language, thus allowing users speaking different languages to take advantage of the application. Currently, the gesture/gaze remote control supports 5 different languages (i.e., English,

Greek, Spanish, Italian and Catalan), although other languages can be easily imported. The selection of a language can be performed even while the application is running, allowing the on-the-fly change of every text that appears on the GUI (i.e., name of the application and the names on the buttons and the groups of buttons) to the specified language, as shown in Figure 27.

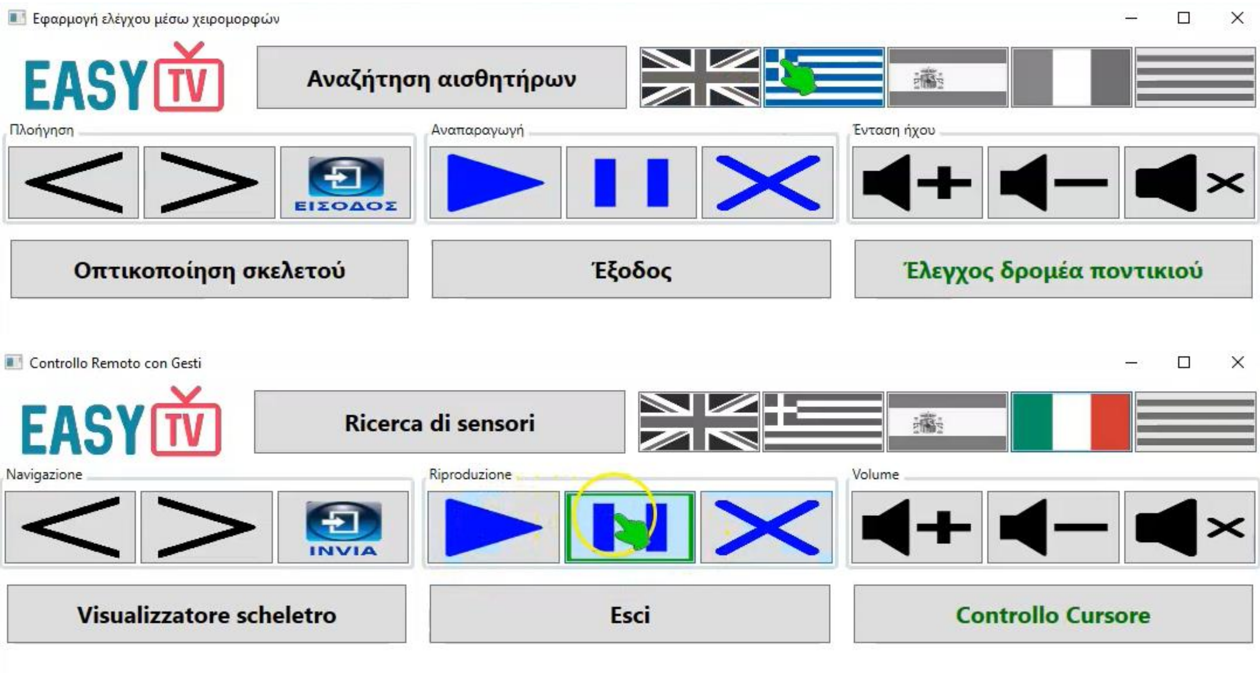


Figure 27: Selecting a language changes all texts on the GUI to the specified language.

The second row of buttons include all major functionalities of the gesture/gaze remote control. These buttons have been split in three groups of three buttons each in order to have an intuitive design for the user with buttons with relative functionalities grouped together. The first group of buttons is called "Navigation" and it includes buttons for navigation left, navigation right and enter in the selected movie. These buttons function only in the movie selection area, as shown in Figure 28.

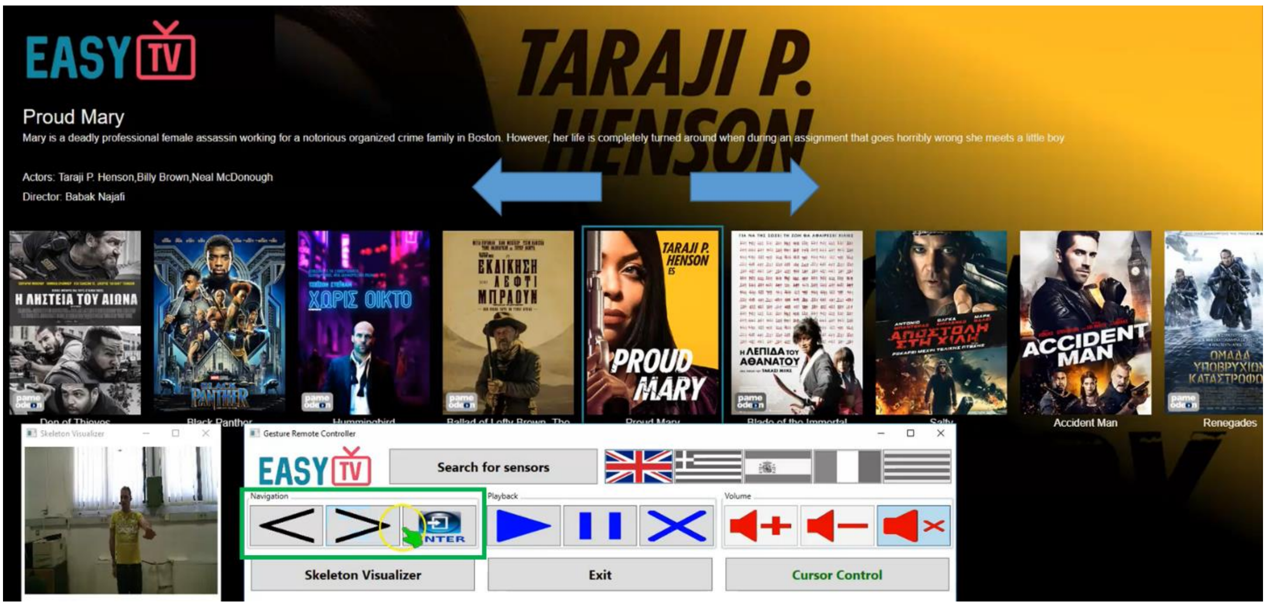


Figure 28: The buttons of the "Navigation" group are responsible for navigation and selection of movie in the movie selection area.

The second group of buttons is called "Playback" and the respective buttons function when we are

in a movie environment in order to play, pause or close the movie, as shown in Figure 29. Finally, the third group of buttons is called "Volume" and the respective buttons increase, decrease or mute the sound of the movie, as shown in Figure 29. All buttons of the second row are connected with specific commands for the TV set that are transmitted to the TV whenever a button is pressed. As a result, the EasyTV gesture/gaze remote control currently supports 8 TV commands, which are presented in the first 8 rows of Table 12.

Figure 29: The buttons of the "Playback" and "Volume" groups function inside the movie environment in order to play, pause and close the movie, as well as increase, decrease or mute the sound.

The third row of buttons consist of the button "Skeleton Visualizer" that allows the visualization of skeletal, colour and depth information to the user, the button "Exit" that terminates the gesture/gaze remote control" and the rightmost button that is responsible for alternating between different modes of operation and displaying the one that is currently active. The modes of operation define the way the user can interact with the application and thus the TV set.

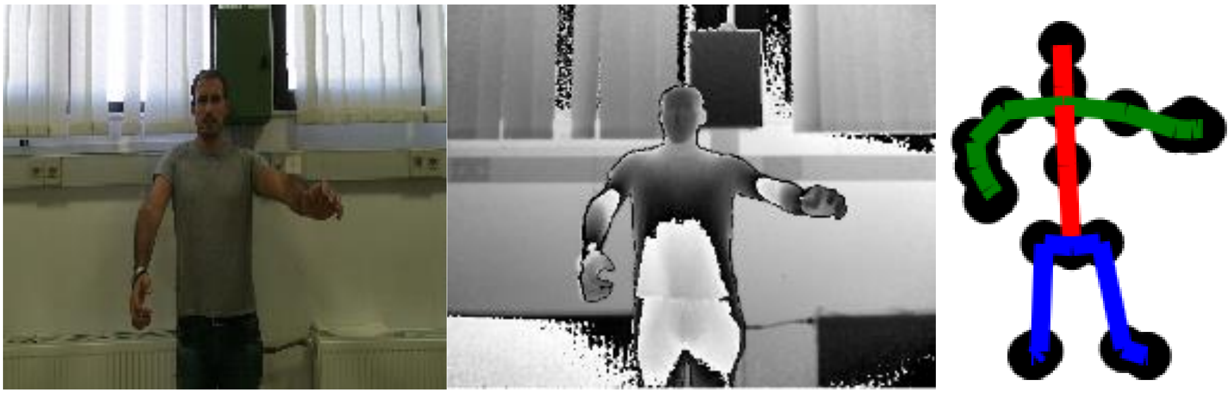


Figure 30: Colour, depth and skeletal information visualized by pressing the corresponding button.

The final version of the gesture/gaze remote control supports three modes of operation, namely hand to cursor control, hand gestures to commands and gaze to cursor control, enhancing the two modes of the preliminary version with the addition of the gaze to cursor control. Furthermore, the application allows easy alternation between the modes with a simple click of a button or issue of a command even while the application is running under a specific mode of operation. The three modes of operation are described in detail below:

- **Hand to cursor control:**
  The hand to cursor control mode involves the control of the mouse functionalities using right and left hand movements. More specifically, the wrist joint of the right hand is used to indicate the position of the mouse cursor on the PC screen. Each displacement of the wrist joint of the right hand with respect to its previous position is translated to a corresponding displacement of the mouse cursor on the PC screen, allowing an easy navigation on the gesture/gaze remote control interface. On the other hand, the left hand is used to indicate the selection of a button (i.e., mouse click). To this end, the positions of the wrist joint of the left hand and the joint of the left shoulder are monitored and the event of the wrist joint of the left hand being higher in the vertical direction than the joint of the left shoulder is interpreted as mouse click on the location of the mouse cursor. Furthermore, by keeping the left wrist raised over the left shoulder, continuous clicking of specific buttons is possible, thus facilitating the continuous increase or decrease of volume and the left or right navigation in the movie selection area.

- **Hand gestures to commands:**
  The hand gestures to commands mode involves the direct issuing of commands via button clicking based on the recognition of predefined hand gestures. More specifically, in this mode, the gesture/gaze remote control is able to automatically extract sequences of skeletal data starting at and finishing to the “hold” state, in which the user has his hands down and close to his/her body. Afterwards, the extracted sequence of skeletal data is compared with predetermined sequences of skeletal data using DTW. The DTW algorithm is chosen due to its ability to identify similar sequences of different temporal length in real-time and with only a single sample. The latter means that the DTW algorithms does not need to be trained with hundreds or thousands of samples from different users in order to achieve high accuracy, making it possible for the user to introduce his/her own new sequences and correlate them with specific commands. Based on the comparison between the executed hand gesture and the saved and predefined hand gestures, the DTW algorithm is able to identify the closest match, meaning that it classifies the executed hand gesture to one of the save gestures, which corresponds to a specific TV command. After the classification, the correct button from the gesture/gaze remote control interface is pressed and the corresponding TV command is sent from the HbbTV client to the HbbTV server, located in the TV set. The predetermined hand gestures were selected in a way to be intuitive to the users based on their meaning (i.e., TV command they correspond to) and to be distinctive enough for the DTW to achieve a high classification accuracy. The predetermined hand gestures that the proposed gesture/gaze remote control currently supports are shown in Table 12.

Table 12: Description of the predetermined hand gestures that the gesture/gaze remote control currently supports.

Command	Image	Description
Navigate left		The user raises his/her left hand parallel to his/her body

Navigate right		The user raises his/her right hand parallel to his/her body
Enter / OK		The user raises his/her right hand in front of his/her body
Play / Resume playback		The user raises both his/her hands in front of his/her body
Close play		The user brings both his/her hands close to his/her chest
Volume up		The user brings his/her right hand index finger in front of him/her and then moves it to the right
Volume down		The user brings his/her right hand index finger in front of him/her and then moves it to the left

Set Mute		The user brings his/her right hand index finger in front of him/her and then moves it close to his/her mouth
Change language		The user brings both his/her wrists close to his/her shoulders
Open / Close "Skeleton Visualizer"		The user raises his/her left hand in front of his/her body
Exit application		The user raises both his/her hands parallel to his/her body

❖ **Gaze to cursor control:**

The gaze to cursor control mode involves the detection of gaze data (i.e., horizontal and vertical location on the screen where each eye is fixating) and their translation to mouse functionalities (i.e., cursor movement and mouse click). To achieve this, a user should stand close to the eye tracking sensor (around 60 cm distance) in order for the sensor to achieve accurate and robust eye tracking results. A useful and important step before eye tracking initiates is calibration. Calibration is used in order to correctly and accurately configure the parameters of the eye tracking sensor based on the current user. If calibration is not performed, the eye tracking sensor will deliver suboptimal accuracy during the gaze data extraction. Calibration requires around 10-15 seconds and it involves the user fixating with his/her eyes on 9 specific targets (i.e., red dots) that are moving around the PC screen, as shown in Figure 31.

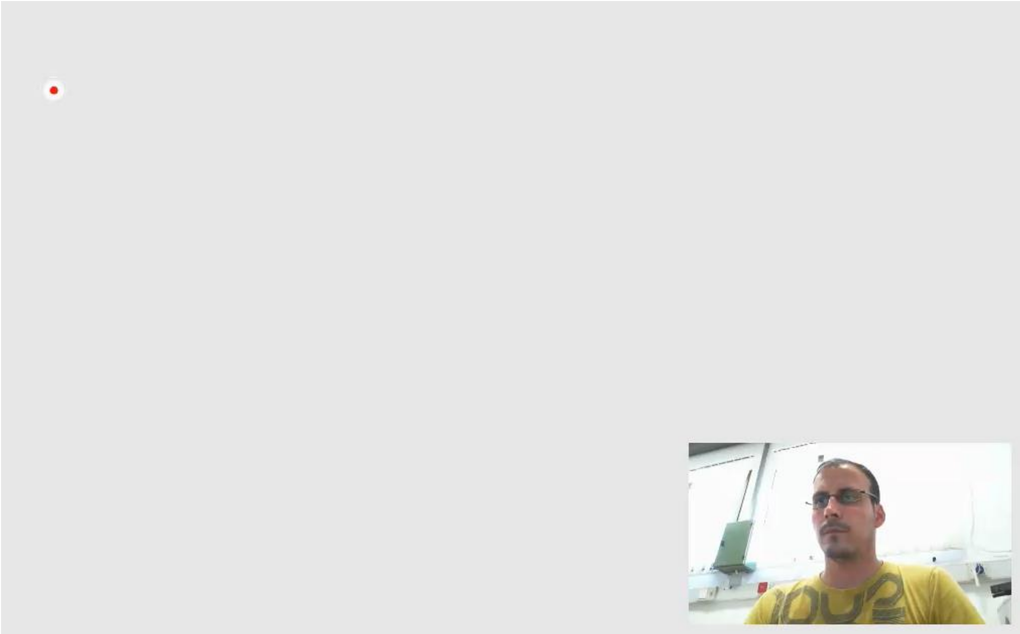


Figure 31: Calibration of the eye tracking sensor by fixating on the red dots.

After calibration is performed, the gesture/gaze remote control is ready to start detecting gaze data and fixations on the PC screen. More specifically, the gesture/gaze remote control is able to detect the location of the screen that a user looks at and move the mouse cursor to that location, effectively implementing the control of the mouse cursor with the user's gaze. Furthermore, the mouse click is performed by measuring the amount of time a user fixates on a specific location of the screen. When a user fixates for more than 2 seconds on a specific location of the screen, then the gesture/gaze remote control translates this incident to his/her inclination to press the specific button that is located in the region in which, a user looks at, as shown in Figure 32. The gesture/gaze remote control executes the button press and the corresponding TV command is sent via the HbbTV client to the HbbTV server, located on the TV set.

Figure 32: A user fixates with his eyes on the "Close" button in order to stop the video from playing.

4.4. Communication with TV set

As far as the communication with TV set is concerned, the final version of the proposed gesture/gaze remote control is developed using Node.js and follows the HbbTV protocol that is

employed for the other EasyTV applications as well. More specifically, the gesture/gaze remote control sets up a HbbTV client that discovers HbbTV terminals on the same network and gets their websocket URLs. This means that the TV set should have a HbbTV application up and running and ready to accept messages using HTML5 technology. The gesture/gaze remote control can then connect to the websocket and start exchanging messages with the HbbTV application, as shown in Figure 33. These messages are in a JSON format (shown in Figure 34) and consist of commands, such as the ones defined in the previous section, that are interpreted by the HbbTV application and employed for the successful communication with the TV set.

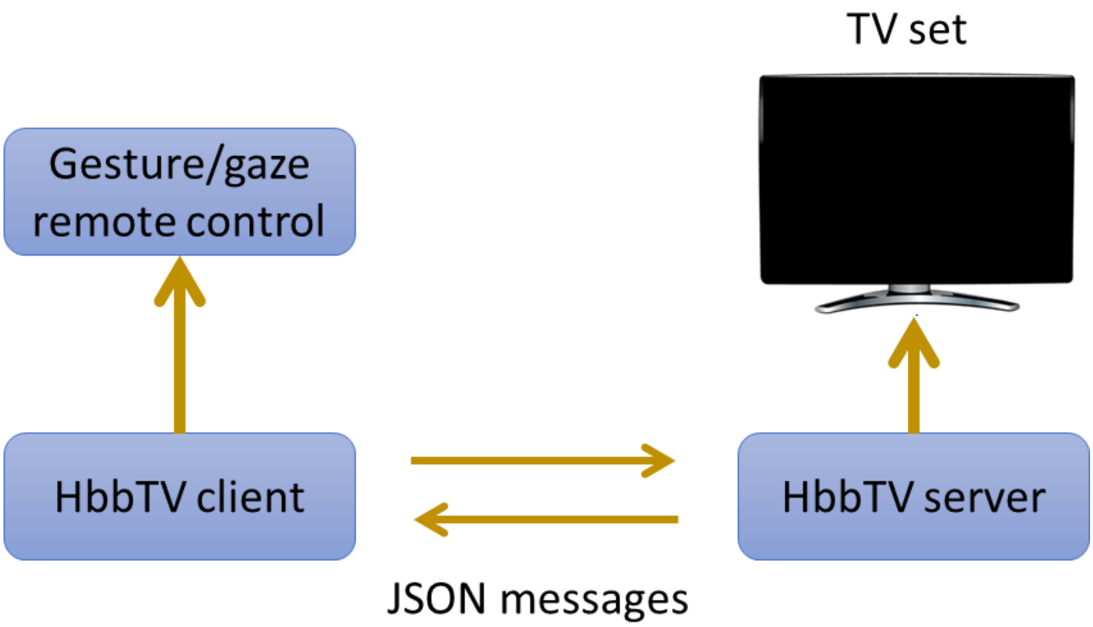


Figure 33: Communication protocol between the proposed gesture remote control and the TV set through a HbbTV application.

```
"{\"action\":\"NAV_RIGHT\"}"

"{\"action\":\"SET_VOLUME\",\"volume\":\"0.3\"}"
```

Figure 34: Example of messages exchanged between the gesture/gaze remote control and the HbbTV server on a TV set. The first command is translated to "Navigation Right", while the second command sets the volume of the movie to 30% of the highest possible volume.

4.5. Refinements and integration with other services of the EasyTV platform

Until the end of the project, some minor refinements to the final version of the EasyTV gesture/gaze remote control are expected. More specifically, these refinements will be concentrated on the GUI of the gesture/gaze remote control and they will involve changes in the images of some buttons in order to become more intuitive to the users.

Additionally, the gesture/gaze remote control will be integrated with the EasyTV personalization service in order to allow some parameters of the GUI of the gesture/gaze remote control to be changed according to the user preferences. These parameters will include changes to the cursor size, shape and colour, font size and style and language of the application, empowering users will low vision or colour blindness and users knowing different languages to take advantage of the gesture/gaze remote control as well. To achieve this, a "Settings" window will be designed in the remote control to allow the definition and update of user preferences, while the remote control will

also be able to read a file that contains default user preference, sent by the EasyTV personalization service.

5. CONCLUSIONS

In conclusion, this document describes the work done as part of the Task 3.4 of the WP3 of the EasyTV project. The goal of this task is to design and develop a gesture/gaze remote control that will be part of a universal accessible remote control. The universal remote control will allow people with disabilities to easily control their TV set using hand movements and gaze data. More specifically, this document presents theoretical and technical details behind the implementation of the final version of the gesture/gaze remote control, as it was defined in the framework of the EasyTV project, the user requirements gathered during the first year of the project and the preliminary version of the gesture/gaze remote control, presented in D3.3. Special attention was given for the design of an application that will conform to specific communication protocols and set of commands in order to facilitate its integration into a universal application, as well as be clear and intuitive for users to use.

Initially, this document describes the theoretical work on action and sign language recognition, along with the publications derived from this work. In this work, we evaluate several deep learning architectures and features that can be extracted from video sequences and we come up with powerful deep learning methodologies that can take full advantage of the provided features to achieve accurate action and sign language recognition results. Such methodologies can be applied for gesture recognition as well, since gestures can be considered special cases of actions and signs. Afterwards, technical information about the gesture/gaze remote control are provided, including detailed presentation of the supported hardware (i.e., sensors and PC configuration), the GUI, the main functionalities, the different modes of operation and the communication protocol. The lack and/or public unavailability of standards for the manipulation of the TV set using gestures has led us to propose a set of intuitive and easy to perform gestures for the EasyTV gesture/gaze remote control. The final version of the gesture/gaze remote control is significantly improved with respect to the preliminary version, presented in the end of the first year of the EasyTV project, while a few final refinements and integration steps, especially with the EasyTV personalization service, are expected till the end of the EasyTV project.

6. REFERENCES

[1] EasyTV – Annex I "Description of Work"
[2] D1.1 User scenario and requirements definition (http://easytvproject.eu)
[3] D1.2 EasyTV system requirements specification (http://easytvproject.eu)
[4] D3.3 Remote control with gesture/gaze controls preliminary version (http://easytvproject.eu)
[5] D1.3 First release of the EasyTV system architecture (http://easytvproject.eu)
[6] Kinect v2 sensor: https://www.windowscentral.com/kinect-windows-v2-sensor-sales-end-developers-can-use-xbox-one-version (Accessed: 25/09/18)
[7] ORBBEC 3d sensor: https://orbbec3d.com/ (Accessed: 25/09/18)
[8] Intel RealSense 3D sensor: https://realsense.intel.com/stereo/ (Accessed: 25/09/18)
[9] Camboard pico flex sensor: https://pmdtec.com/picofamily/flexx/ (Accessed: 26/09/18)
[10] G. Fang, W. Gao and D. Zhao, "*Large vocabulary sign language recognition based on fuzzy decision trees*," IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans, vol. 34, no. 3, pp. 305-314, May 2004.
[11] W.W. Kong and S. Ranganath, "*Signing exact English (SEE): Modelling and recognition*," Pattern Recognition, vol. 41, no. 5, pp. 1638-1652, 2008.
[12] S.S. Rautaray and A. Agrawal, "*A real time hand tracking system for interactive applications*," in International Journal of Computer Applications, vol. 18, no. 6, pp. 28-33, March 2011.
[13] Z. Zhang and F. Huang, "*Hand tracking algorithm based on superpixels feature*," in International Conference on Information Science and Cloud Computing Companion, Guangzhou, December 2013, pp. 629-634.
[14] K.M. Lim, A.W.C. Tan and S.C. Tan, "*Block-based histogram of optical flow for isolated sign language recognition*," Journal of Visual Communication and Image Representation, vol. 40, part B, pp. 538-545, 2016.
[15] K.M. Lim, A.W.C. Tan and S.C. Tan, "*A feature covariance matrix with serial particle filter for isolated sign language recognition*," Expert Systems with Applications, vol. 54, pp. 208-218, 2016.
[16] Y.F.A. Gaus and F. Wong, "*Hidden markov model-based gesture recognition with overlapping hand-head/hand-hand estimated using Kalman filter*," in Third International Conference on Intelligent Systems Modelling and Simulation, Kota Kinabalu, 2012, pp. 262-267.
[17] L. Seidenari, V. Varano, S. Berretti, A. Del Bimbo, and P. Pala, "*Recognizing actions from depth cameras as weakly aligned multi-part bag-of-poses*," in IEEE Conference on Computer Vision and Pattern Recognition Workshops, June 2013, pp. 479–485.
[18] M. E. Hussein, M. Torki, M. A. Gowayyed, and M. El-Saban, "*Human action recognition using a temporal hierarchy of covariance descriptors on 3d joint locations*," in Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI). AAAI Press, 2013, pp. 2466–2472.
[19] L. Zhou, W. Li, Y. Zhang, P. Ogunbona, D. T. Nguyen, and H. Zhang, "*Discriminative key pose extraction using extended LC-KSVD for action recognition*," in International Conference on Digital Image Computing: Techniques and Applications (DICTA), Nov 2014, pp. 1–8.
[20] L. Xia, C. Chen, and J. Aggarwal, "*View invariant human action recognition using histograms of 3d joints*," in IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2012, pp. 20–27.
[21] R. Vemulapalli, F. Arrate, and R. Chellappa, "*Human action recognition by representing 3d skeletons as points in a Lie group*," in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2014, pp. 588–595.
[22] J. Wang, Z. Liu, Y. Wu, and J. Yuan, "*Learning actionlet ensemble for 3d human action recognition*," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 36, no. 5, pp. 914–927, May 2014.
[23] F. Patrona, A. Chatzitofis, D. Zarpalas, and P. Daras, "*Motion analysis: Action detection, recognition and evaluation based on motion capture data*," Pattern Recognition, vol. 76, pp. 612–622, 2018.
[24] K. Dimitropoulos, P. Barmpoutis, and N. Grammalidis, "*Higher order linear dynamical systems for smoke detection in video surveillance applications*," IEEE Transactions on Circuits and Systems for Video Technology (TCSVT), vol. 27, no. 5, pp. 1143–1154, 2017.
[25] K. Dimitropoulos, P. Barmpoutis, A. Kitsikidis, and N. Grammalidis, "*Classification of multidimensional time-evolving data using histograms of Grassmannian points*," IEEE Transactions on Circuits and Systems for Video Technology (TCSVT), vol. 28, no. 4, pp. 892–905, 2018.
[26] S. Zhang, X. Liu, and J. Xiao, "*On geometric features for skeleton-based action recognition using multilayer LSTM networks*," in IEEE Winter Conference on Applications of Computer Vision (WACV), March 2017, pp. 148–157.
[27] P. Wang, Z. Li, Y. Hou, and W. Li, "*Action recognition based on joint trajectory maps using*

convolutional neural networks," in Proceedings of the 2016 ACM on Multimedia Conference. ACM, 2016, pp. 102–106.

[28] T. Simon, H. Joo, I. Matthews and Y. Sheikh, "*Hand keypoint detection in single images using multiview bootstrapping*," IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, 2017, pp. 4645-4653.

[29] H. Sakoe and S. Chiba, "*Dynamic programming algorithm optimization for spoken word recognition*", IEEE Trans. Acoustics, Speech and Signal Processing, vol. ASSP-26, no. 1, pp. 43-49, 1978.

[30] N. Tanibata, N. Shimada and Y. Shirai, "*Extraction of hand features for recognition of sign language words*," in International Conference on Vision Interface, 2002, pp. 391-398.

[31] X. Ni, G. Ding, X. Ni, X. Ni, Q. Jing, J. Ma, P. Li and T. Huang, "*Signer-independent sign language recognition based on manifold and discriminative training*," in Information Computing and Applications, 2013, pp. 263-272, Springer Berlin Heidelberg.

[32] F. Ronchetti, F. Quiroga, C. Estrebou, L. Lanzarini and A. Rosete, "*LSA64: A dataset of Argentinian sign language*," XX II Congreso Argentino de Ciencias de la Computación (CACIC), 2016.

[33] L. R. Rabiner, "*A tutorial on hidden Markov models and selected applications in speech recognition*," Proc. IEEE, vol. 77, no. 2, pp. 257–285, Feb. 1989.

[34] S. Wang, A. Quattoni, L.P. Morency, D. Demirdjian and T. Darrell, "*Hidden Conditional Random Fields for Gesture Recognition*," Computer Vision and Pattern Recognition, vol. 2, pp. 1521-1527, 2006.

[35] O. Koller, S. Zargaran, H. Ney, and R. Bowden, "*Deep sign: Hybrid CNN-HMM for continuous sign language recognition*," in Proceedings of British Machine Vision Conference (BMVC), 2016.

[36] O. Koller, S. Zargaran and H. Ney, "*Re-sign: Re-aligned end-to-end sequence modelling with deep recurrent CNN-HMMs*," IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, 2017, pp. 3416-3424.

[37] J. Huang, W. Zhou, H. Li and W. Li, "*Sign language recognition using 3d convolutional neural networks*", in IEEE International Conference on Multimedia and Expo (ICME), 2015, pp. 1-6.

[38] D. Konstantinidis, K. Dimitropoulos and P. Daras, "*Skeleton-based action recognition based on deep learning and Grassmannian pyramids,*" 26th European Signal Processing Conference (EUSIPCO 2018), Rome, Italy, September 2018.

[39] S. Amiri, M. Pourazad, P. Nasiopoulos, and V. Leung, "*Human action recognition using meta learning for RGB and depth information*," in International Conference on Computing, Networking and Communications (ICNC), Feb 2014, pp. 363–367.

[40] D. Konstantinidis, K. Dimitropoulos and P. Daras, "*Sign language recognition based on hand and body skeletal data*," 3DTV Conference, Stockholm-Helsinki, June 2018.

[41] Z. Cao, T. Simon, S.-E. Wei and Y. Sheikh, "*Real-time multi-person 2d pose estimation using part affinity fields*," IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 1302-1310.

[42] K. Simonyan and A. Zisserman, "*Very deep convolutional networks for large-scale image recognition*," in International Conference on Learning Representations (ICLR), 2015.

[43] D. Konstantinidis, K. Dimitropoulos and P. Daras, "*A deep learning approach for analyzing video and skeletal features in sign language recognition*," IEEE International Conference on Imaging Systems and Techniques (IST), Krakow, Poland, October 2018.

[44] K. Simonyan and A. Zisserman, "*Very deep convolutional networks for large-scale image recognition*", in Proc. International Conference on Learning Representations, 2014.

[45] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy and T. Brox, "*FlowNet 2.0: Evolution of optical flow estimation with deep networks*", IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.

[46] L. Piras, R. Tronci and G. Giacinto, "*Diversity in ensembles of codebooks for visual concept detection*," in International Conference on Image Analysis and Processing (ICIAP), Naples, Italy, 2013.

[47] R. Tronci, G. Giacinto, and F. Roli, "*Dynamic score combination: A supervised and unsupervised score combination method*," in Machine Learning and Data Mining in Pattern Recognition (Lecture Notes in Computer Science), Springer Berlin Heidelberg, vol. 5632, 2009, pp. 163-177.

[48] S. K. Tchomté and M. Gourgand, "*Particle swarm optimization: A study of particle displacement for solving continuous and combinatorial optimization problems*," International Journal of Production Economics, vol. 121, no. 1, pp. 57-67, 2009.

[49] V. Bloom, D. Makris, and V. Argyriou, "*G3D: A gaming action dataset and real time action recognition evaluation framework*," in IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, June 2012, pp. 7–12.

[50] 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.

[51] L.Wang, J. Zhang, L. Zhou, C. Tang, and W. Li, "*Beyond covariance: Feature representation with nonlinear kernel matrices*," in IEEE International Conference on Computer Vision (ICCV), Dec 2015, pp. 4570–4578.

[52] A. Sharaf, M. Torki, M. E. Hussein, and M. El-Saban, "*Real-time multi-scale action detection from 3d skeleton data*," in IEEE Winter Conference on Applications of Computer Vision, Jan 2015, pp. 998–1005.

[53] M. Meshry, M. E. Hussein, and M. Torki, "*Linear-time online action detection from 3d skeletal data using bags of gesturelets*," in IEEE Winter Conference on Applications of Computer Vision (WACV), Jan 2016, pp. 1–9.

[54] Y. Zhu, W. Chen, and G. Guo, "*Fusing spatiotemporal features and joints for 3d action recognition*," in IEEE Conference on Computer Vision and Pattern Recognition Workshops, June 2013, pp. 486–491.

[55] F. Ronchetti, F. Quiroga, C. Estrebou, L. Lanzarini and A. Rosete, "*Sign language recognition without frame-sequencing constraints: A proof of concept on the Argentinian sign language*," in Advances in Artificial Intelligence - IBERAMIA 2016, 2016, pp. 338-349, Springer International Publishing.

[56] J. Forster, C. Schmidt, T. Hoyoux, O. Koller, U. Zelle, J. Piater and H. Ney, "*RWTH-PHOENIX-Weather: A large vocabulary sign language recognition and translation corpus*," in Language Resources and Evaluation (LREC), 2012, pp. 3785-3789.

[57] Tobii gaze tracking sensor: http://www.tobii.com/ (Accessed: 26/09/18)

[58] SMI gaze tracking sensor: http://www.smivision.com/ (Accessed: 26/09/18)

[59] EyeTech gaze tracking sensor: http://www.eyetechds.com/ (Accessed: 26/09/18)

[60] Mirametrix gaze tracking sensor: http://mirametrix.com/ (Accessed: 26/09/18)

[61] MyGaze Eye Tracker sensor: http://www.mygaze.com/products/mygaze-eye-tracker/ (Accessed: 26/09/18)

[62] C. Wang, F. Shi, S. Xia and J. Chai, "*Realtime 3d eye gaze animation using a single rgb camera,*" ACM Transactions on Graphics (TOG), vol. 35, no. 4, 2016, 118.

[63] F. Vicente, Z. Huang, X. Xiong, F. De la Torre, W. Zhang and D. Levi, "*Driver gaze tracking and eyes off the road detection system*" IEEE Transactions on Intelligent Transportation Systems, vol. 16, no. 4, 2014-2027, 2015.

[64] X. Zhang, Y. Sugano, M. Fritz and A. Bulling, "*Appearance-based gaze estimation in the wild,*" in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, p. 4511-4520.

[65] LG gesture control: https://www.lg.com/us/lgeai/HDTV/resources/UserGuides/webOS1/contents/control/motionrecognition_all/enga/w14__control__motionrecognition_all__enga.html (Accessed: 12/09/18)

[66] Samsung gesture control: https://www.samsung.com/au/support/tv-audio-video/using-gesture-control/ (Accessed: 12/09/18)

[67] Sony LF-S50G gesture control: https://www.sony.co.uk/electronics/support/articles/00187472 (Accessed: 12/09/18)