A Sliding Window Approach to Natural Hand Gesture Recognition using a Custom Data Glove

Granit Luzhnica[*]
Know-Center GmbH

Jörg Simon[†]
Know-Center GmbH

Elisabeth Lex[‡]
Knowledge Technologies Institute, Graz Univ. of Technology & Know-Center GmbH

Viktoria Pammer[§]
Knowledge Technologies Institute, Graz Univ. of Technology & Know-Center GmbH

ABSTRACT

This paper explores the recognition of hand gestures based on a data glove equipped with motion, bending and pressure sensors. We selected 31 natural and interaction-oriented hand gestures that can be adopted for general-purpose control of and communication with computing systems. The data glove is custom-built, and contains 13 bend sensors, 7 motion sensors, 5 pressure sensors and a magnetometer. We present the data collection experiment, as well as the design, selection and evaluation of a classification algorithm. As we use a sliding window approach to data processing, our algorithm is suitable for stream data processing. Algorithm selection and feature engineering resulted in a combination of linear discriminant analysis and logistic regression with which we achieve an accuracy of over 98.5% on a continuous data stream scenario. When removing the computationally expensive FFT-based features, we still achieve an accuracy of 98.2%.

Index Terms: C.3 [Special-Purpose and Application-Based Systems]: Signal processing systems I.5.2 [Design Methodology]: Classifier design and evaluation I.5.2 [Design Methodology]: Feature evaluation and selection I.5.2 [Design Methodology]: Pattern analysis I.5.4 [Applications]: Signal processing H.5.2 [User Interfaces]: Input devices and strategies H.5.2 [User Interfaces]: Interaction styles

1 INTRODUCTION

Gesture recognition has been an active field of research for more than two decades in human computer interaction. Initially, the motivation was to detect and recognise sign language [1, 14, 33, 41]. The goal mostly was to develop computing systems that could understand and translate sign language. More recently, gesture recognition has gained interest as basis for gesture based interaction in a wide range of use cases, such as crisis management [45], TV remote controlling [34], interacting with computer [18, 24], gaming interfaces [23, 26, 45, 52], augmented reality applications [17, 43, 48, 50], hands-free interaction in car driving [27], providing virtual training for car driving [50] or detecting a driver's fatigue [25]. In the medical area, robot nurses are envisioned to detect surgeon's hand gestures and to assist with necessary surgical instrument [45]. In another type of use case, computer systems detect gestures in order to understand user activities. For instance, robots have been envisioned to analyse gestures in order to track which tasks are already completed in order to be able to seamlessly take over with the next steps [6, 35]. Sometimes, it is useful to only observe and document the gestures, as in the case of assembly lines to document the work for quality assurance [40]. More in general, the goal to detect assembly line tasks is an area of active research [20, 32, 46, 51]. Gesture recognition has also been explored in the context of logging activities of daily life: In [38], the authors explore the possibility to detect eating habits via recognising the gestures for eating and drinking (bringing the hand to the mouth). In [39], activity logging based on both smartwatch and smartphone sensing is used to detect drinking too much coffee or not eating.

With this work we contribute to the field of gesture recognition by exploring the recognition of natural and interaction-oriented hand gestures based on sensors worn on users' hands. To that end, we designed a custom data glove equipped with sensors that capture both motion and state of the hand and fingers. We concentrated on gestures that are widely known and that can reasonably be adopted to control and communicate with computing systems. Our envisioned use case is that of mapping out a general-purpose gesture alphabet. It should be easy to learn for users, and should be able to replace some of the interactions with computing systems (selecting, browsing, etc.) that are currently performed via mouse or smartphone gestures.

We approached this goal by conducting a data collection experiment in which multiple users performed such gestures. In parallel to sensing, the gestures were manually annotated with gesture names. This resulted in a labelled set of hand gestures, which we used to extract representative features and to train a supervised learning algorithm. Then, we evaluated the performance of our algorithm "online", i.e. on a continuous sensor data stream. The contributions of this work are three-fold:

- A data set of natural hand gestures, which were gathered in a data collection experiment with 18 adults, and are manually annotated with gesture names.
- Features selection - We identified characteristic features for gestures and investigated similarities between gestures.
- Algorithm selection - We identified a performant algorithm for classifying gestures in a continuous sensor data stream.

2 RELATED WORK

We identified two strands of research that are relevant for our work: firstly, research that deals with vision based systems for gesture recognition and secondly research that deals with wearable sensors for gesture recognition. In the first case, the gesture recognition relies on an infrastructure built into the environment (e.g., using Kinect or webcam) whereas in the second case, the gesture recognition relies on wearable sensor technologies like data gloves, armbands or smartwatches.

Vision based systems for gesture recognition. Typically a camera that is mounted in the environment records human hands and the system extracts features from the individual frames of the recording. Sometimes there is a filtering process involved which removes unwanted objects like e.g heads from the image or video [7]. Typically, postures are predicted [5, 7] and then a grammar is con-

[*]e-mail:gluzhnica@know-center.at
[†]e-mail:jsimon@kow-center.at
[‡]e-mail: elex@know-center.at
[§]e-mail: viktoria.pammer@tugraz.at

IEEE Symposium on 3D User Interfaces 2016
19–20 March, Greenville, SC, USA
978-1-5090-0842-1/16/$31.00 ©2016 IEEE

structed to recognise gestures, where a gesture is defined as a sequence of postures [9]. For instance, in [7], the authors first detect and track hands and then recognise ten postures with an accuracy of 96% in camera images with a multi-class SVM. Similarly, in [5] a single web cam is used as source, from which the authors extract Haar-like features and use AdaBoost to discriminate between four postures: two finger, palm, fist, little finger. The authors achieve an accuracy of over 90%. In [1], the authors achieve an accuracy of 99% by using PCA and a Euclidian distance based classifier to recognise 25 international hand alphabet postures from images of the gestures. An alternative vision based approach is to use coloured gloves in which different parts of the hand are marked with different colours, making it much easier to track gestures [12].

Wearable sensor based systems for gesture recognition. The majority of wearable sensor systems for gesture detection are gloves equipped with sensors. In most research endeavours, gloves are custom-built. In [29], a recurrent neural network (RNN) is used to recognise the following Japanese sign language gestures: father, mother, brother, sister, memorise, forget, like, hate, skilled and unskilled. The gestures are constructed using 42 previously recognised postures representing Japanese letter alphabet with an accuracy of 96%. The data were generated using VPLDataGlove. In [50], the authors use a feed-forward neural network capable of distinguishing between 15 gestures with an accuracy of 98%. Data were recorded with a CyberGlove with 18 sensors. More recently, a feed-forward neural network was used to construct a hand gesture recognition system for interacting with robots [30] . Using data from CyberGlove II (providing 22 joint-angle measurements), the authors were able to recognise 10 different artificial gestures with an accuracy of 99.8% and 30 gestures with an accuracy of 96.3%. In this work, the authors initially performed segmentation by identifying whether each of the data readings belongs to a gesture. These segments were used as input to the machine learning algorithm. In [52], data from EMG and a wrist-worn accelerometer were used to build a system that recognises 18 gestures with an accuracy of 91.7%. The defined gestures were used to play a virtual rubic's cube game. In [36], authors used a list of 22 natural hand gestures. The gestures originally stem from [10]. While analysing the data, the authors first resampled and interpolated the data. They then used LDA to discriminate between the resampled segments with an accuracy of 92.8%. This shows that there is a clear separability between those gestures. However, in a live recognition system, such resampling and interpolation is not possible unless the start and end time of the gesture that needs to be recognised can be detected.

Recently, there has been a growing research interest in gesture recognition based on consumer good technology like smart watches (e.g., [11, 44, 49, 53]). In [49], the authors report the classification of 37 interaction oriented gestures, i.e. gestures that are intended to be used for controlling other devices (turning the arm, simulating a click, pinch to zoom, etc..). The gestures are detected based on smartwatch sensor data only, and with an accuracy of 98% by using Naive Bayes algorithm. The authors report different numbers in a subsequent demo paper, namely 27 gestures with 96% accuracy using Logistic Regression or Decision Trees [53]. However, the data in the latter paper were collected only by a single participant; and the participant performed gestures from a fixed arm position.

When comparing two approaches, vision based systems are more sensitive to the environment. Lighting conditions, scene and background details are issues that affect such systems [45]. In the case of cameras, there might be also privacy issues; and different countries have different regulations concerning video recordings in non-private environments. Wearable sensor based systems, especially glove based ones, can be uncommfortable [45] or even pose a hygienic problem [19]. On the other hand, wearable technologies provide in principle the possibility for higher privacy as the data are a priori more anonymous than pictures or videos. When it comes to accuracy, many authors report very high accuracies (in the higher 90s) for the selected set of gestures using either technique [1, 7, 30, 50].

In this paper, we take the wearable sensors approach. In contrast to some other works, we emphasise capturing the dynamics of the gestures, and present gesture recognition using a custom data glove. Our work also differs from previous work since we take a sliding window approach in combination with dual labelling in the test set. Sliding window is a technique for data preprocessing in which information is extracted (statistics, aggregates, features, etc...) over a "sliding window" that contains a fixed number of samples. This enables us to use representative features that aggregate sensor data over time. Interestingly, the sliding window approach, to the best of our knowledge, has not been used in gesture recognition system even though it very common approach in activity recognition [21, 22, 31].

3 Gesture Detection System

In this section, we describe the design of the overall gesture detection system: What kind of gestures should our system detect, and what kind of information/sensors are required for this purpose?

3.1 Interaction-Oriented and Natural Hand Gestures

As described in the introduction, we are interested in a general-purpose gesture alphabet with which to control computers and communicate with them. Essentially, it would be possible to develop a completely novel gesture language for such a purpose. A study looking at inventing custom gestures [15] showed however, that a user can only remember a very limited number (about two) of such artificial gestures. Therefore, we are looking at gestures that are widely known, even though there may be cultural differences regarding their popularity and meaning. Additionally, there should be a plausible relationship between the gesture and an interaction between human and computer.

These criteria resulted in the following 31 hand gestures (see Table 1). Our gesture set was initially based on the list of 22 natural gestures described in [10]. We added the following: The numbers one to five, as they would be useful to select items; popular touch-based swipe gestures such as swipe left, right and down (up was already on the original list), as these would be useful for navigation. Finally, we added lateral grasp (Grasp 2) and palmar grasp (Grasp 1) gestures, as we think that grasping objects would be useful in interaction with 3D virtual objects. After the first trial, the gesture walk was discarded as it was difficult to perform due the IMU chips on the fingers.

3.2 Custom Data Glove

The above gestures vary a lot in their dynamics: Some gestures contain a lot of complex motions (e.g continue) whereas some are very close to a posture (e.g. numbers one, two, ...).

We planned a data glove that emphasises motion detection of the fingers (which implies that we would have motion sensors on the fingers); as well as hand postures (which implies that we would use bend sensors). The glove is depicted in Figure 1.

We placed two bend sensors on each finger. The upper sensor measures the bending (which translates to angle) of the finger relative to the hand, whereas the lower sensor measures the bending between middle segment and base segment of the finger. Another bend sensor is placed between the thumb and index finger in order to measure the distance/angle between them. Two more bend sensors (in opposite direction) are placed on the wrist in order to be able to measure wrist flexion/extension. Overall, this gives 13 bend sensors. Additionally, each finger tip is equipped with a pressure sensor (5 pressure sensors). Furthermore, 7 IMUs[1] are placed, one

[1]IMU (inertial measurement unit) chip contains a gyroscope and an accelerometer

Gesture	Description
(1) One	Number one by extending index finger
(2) Two	Number two by extending index and middle finger
(3) Three	Number three by extending index, middle and ring finger
(4) Four	Number four by extending all fingers except thumb
(5) Five	Number five by extending all fingers
Thumbs up	Thump stretched pointing up, other fingers form fist
Thumbs down	Thump stretched pointing down, other fingers form fist
Point to self	Pointing at self with thumb
Shoot	Hand in form of a gun and then vibrate
Scissor	Stimulating scissors with two fingers
Cutthroat	Using index finger
Continue	Waving like circular motion with the flat hand
Knocking	Forming a fist and moving the fist up and down
Waving	Shaking the flat hand left and right
Come here	Flat hand with palm upwards: Simultaneous flexing the all fingers but the thumb
Go away	Hand with palm downwards, all fingers but thumb flexed. Simultaneous stretching them
Push away	Flat hand with palm pointing fore wards, then moving the whole hand forward
Never mind	Flat hand with palm pointing left above the head, then moving the whole hand left
Talking	Thumb and 4 fingers pointing forward. Then moving 4 fingers up and down
Calling	Hand is a fist, but thumb and small finger are extended
Walking	Hand is a fist, but making a walking motion with the index and middle finger
Shoulder pat	patting with the open hand on a virtual shoulder
Point	Pointing in front with index finger
Swipe left	Stretched hand with palm pointing left, flexing it completely to the right first, then flexing it to the left, in a circular motion
Swipe right	swiping with palm pointing right, and left to right motion
Swipe up	swiping with palm pointing up, and bottom to top motion
Down	swiping with palm pointing down, and top to bottom motion
Turn	Hand rotation
Zoom	Reverse pinch using index finger and thumb
Grasp 1	Palmar grasp (in the experiment we used a glass)
Grasp 2	Lateral grasp (in the experiment we used a pen)

Table 1: List of 31 interaction-oriented hand gestures.

at the top of each finger, one on the back of the hand and one on the wrist. The wrist IMU is placed exactly at the position where a watch would be. This allows the data recorded with the glove to also be treated as if it came from a smartwatch by simply ignoring the input from other sensors. Finally, a magnetometer is placed on the back of the hand.

At the beginning of our study, various data gloves had already been available commercially. All of them emphasise bend sensors in fingers, and thus focus on hand postures. In contrast, our glove contains both bend and motions sensors (gyroscope + accelerometer) on each finger, thus focussing more on hand motion, i.e. the dynamic aspects of gestures. The MiMu Glove[2] is used to produce music by some means of gesture detection. It employs one IMU at the wrist, 4 bend sensors at the fingers, and vibrators at the underarm to provide haptic feedback. Fifth Dimensional Technologies[3] offers two gloves that are equipped with bend sensors and abduction sensors between fingers. CyberGlove Systems [4] offers CyberGlove II equipped with two bend sensors on each finger, four abduction sensors, sensors measuring thumb crossover, palm arch, wrist flexion and wrist abduction. Virtual Labs[5] offers a range of data glove products (VMG Lite, VMG 10, VMG30, VMG 30 Plus), all of them equipped with bend sensors on the fingers, 9-DOF orientation sensors for hand and wrist, as well as tactile feedback vibrators.

[2]http://mimugloves.com
[3]http://www.5dt.com/
[4]http://www.cyberglovesystems.com
[5]http://www.virtualmotionlabs.com/

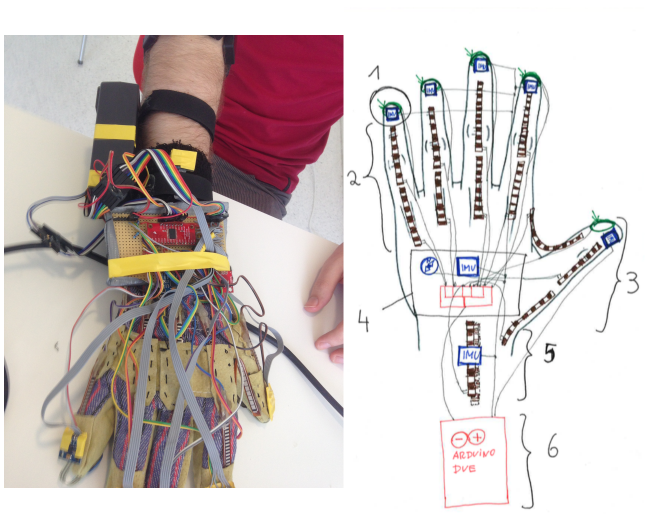

Figure 1: Custom data glove, and scheme of sensor positions. 1: An IMU and a pressure sensor are placed on each finger tip. 2: Two bend sensors cover the two main joints of each finger. 3: The thumb is special as it has 3 bend sensors. 4: An IMU and a Magnetometer is on the top of the hand. Here also an analog multiplexer is mounted to combine all the bend sensors. 5: An IMU is placed on the wrist. Additionally, one bend sensor at the top of the wrist and one at the bottom of the wrist give the angle of the hand to the forearm. 6: All sensors are connected to an Arduino board to collect the data and send it to a computer.

Our custom data glove is a hardware prototype and as such it has some limitations, mainly regarding usability: For long-term wearing, the glove should for instance be made of more comfortable material, be made of smaller and not visible electronic components, should be available in different sizes, and be wirelessly connected to the computing unit.

4 Data Collection Experiment

We collected sensory data annotated with gesture names in the subsequently described data collection experiment.

4.1 Participants

We collected data from 18 healthy adults: 11 males and 7 females. Participants were aged between 24 and 40 years.

4.2 Procedure

Before starting with the data recording, the purpose and procedure of the experiment were explained. Participants were asked to remain seated during the experiment in an office chair. In front of them (on the desk), a monitor was placed. The monitor was used to display the instructions of the experiment. The overall setup is shown in Figure 2.

For each gesture, the following steps were performed in the given sequence:

1. Name of the gesture was shown on the screen ($2s$)
2. A video was displayed; it showed an actor performing the gesture (without glove; $6s$-$7s$)
3. A counter was shown on the screen alarming the participants that the recording was about to start ($3s$)

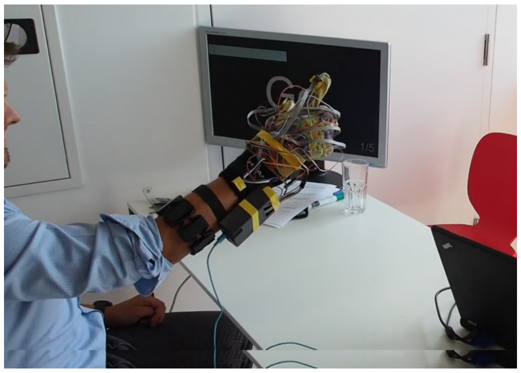
Figure 2: Participant performs "One (1)" gesture while the progress bar is on the screen

4. The participant was asked (audio and text on screen) to start performing the gesture. A progress bar was displayed on screen, indicating the time the participant had to finish the gesture (3*s*). The appearance of the progress bar started the time window called "automatic labelling" (4) in Figure 3.
5. When the participant actually started the gesture, the experiment observer pressed a button on the keyboard. This indicated the start point of the time window called "manual labelling" (5) in Figure 3.
6. When the participant ended the gesture, the experiment observer released the button. This was the end point of the time window called "manual labelling" (6) in Figure 3.
7. When the progress bar ended, the time window called "automatic labelling" (7) in Figure 3 was ended.

The timeline of one gesture is illustrated in Figure 3. Every gesture was performed several times by every participant (5 or 10 times depending on willingness of participant) in a row. The gesture name and the video of the actor performing the gesture (steps 1 and 2) were shown only for the first repetition of the gesture, whereas the counter, progress bar and labelling (steps 3-7) were the same in every repetition.

4.3 Data Annotation

Figure 3 illustrates how the data collection experiment procedure relates to the continuous sensor signals that we recorded. Here we comment on two things regarding data annotation: Firstly, we used *manual labels* as the ground truth, i.e. the labels we refer to in training and testing. *Automatic labels* are used to perform sanity check on *manual labels* e.g. sometimes it happened that the experimenter forgot to label a gesture. We discarded such data. Secondly, when sliding windows are moved over the continuous sensor signals, then there are windows with no gesture in it (window 1 in Figure 3), with partial gestures in it (windows 1 and 5 in Figure 3), and windows with full gestures in it (windows 2 and 3 in Figure 3). For algorithm design (Section 5) only windows with no or full gestures were used, while for evaluation of the selected algorithm in realistic settings, the algorithm was also evaluated on windows that contain a partial gesture (Section 6).

5 Algorithm Design

In this section, we describe the process of selecting the best performing supervised learning algorithm, and the optimum configuration. By configuration we mean selecting parameters for window slicing and parameters for spectral components of the window. Overall, we prepared a list of all possible configurations and

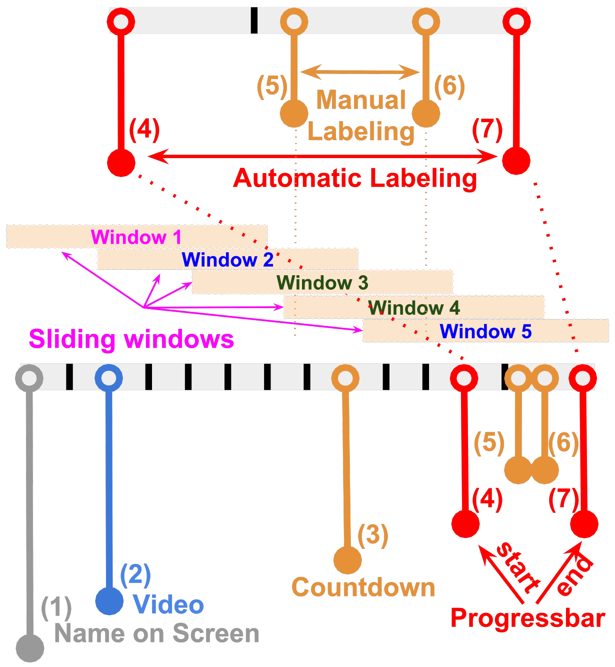

Figure 3: Experiment timeline for a single repetition and sliding windows construction

cross-validated each of them against the set of all chosen learning algorithms. Finally, in order to choose the best configuration, we averaged the cross validation results over all algorithms and considered configurations with the highest average scores. The details of each step are given below.

5.1 Data Pre-processing

Since in the accelerometer readings the gravity component is present and we only need to know the real acceleration value, we first removed the gravity component from the data readings. The gravity component was removed using a complementary filter [13], which typically gives satisfying results and is computationally less expensive than a Kalman Filter [16]. In addition, in order to get independent axis accelerator and gyroscope values, we computed the norm ($norm = \sqrt{x^2+y^2+z^2}$) of accelerometer and gyroscope vector for each of our IMUs. Magnetometer values were discarded as their values provide information related to the absolute location of the hand whereas gesture recognition should work regardless of the hand location. Furthermore, all data dimensions are normalised with zero mean and a standard deviation of 1.

5.2 Window Length and Step-size

As basic unit for classification we use sliding windows, i.e. data windows of fixed sample size that constitute snapshots of the continuous data stream. Features are computed per window. Sliding windows are a well-established method of feature extraction used in many domains (speech to text [37], activity recognition [21, 22, 31], etc..). Their advantage is that the extracted features can be used with almost any algorithm [8]. They typically have two configuration parameters: size and step. For parameter selection, we cross validated the data with several window sizes (140, 160, 180, 200 samples, where 1 second contains 85-87 samples).

As for the labels, we consider one window to have a gesture label only if the window contains the whole gesture. Otherwise we label it as idle class. We used steps of 20, 30, 40 and 50 samples and again used cross validation to select a value for this parameter. The details of cross validation are given in Section 5.4.

5.3 Feature Engineering

The recorded data set contains the following dimensions: (x, y, z, *norm*) values of gyroscope, (x, y, z, *norm*) values of accelerometer, values of pressure sensors, values of bending sensors. For each data dimension we used the following descriptive statistic as features: minimum, maximum, range, average, standard deviation and signal energy from the sliding windows. Minimum and maximum values of the bend sensors should contribute to capturing the static part (posture) of the gesture. On the other hand, the derived values from the norm value of the gyroscope and the accelerometer should capture orientation independent motions aspects, as the norm is just the intensity of the accelerometer or gyroscope in any direction.

For the gyroscope and accelerometer values, we also used the spectrum features, namely the amplitude of Fast Fourier Transform (FFT) coefficients for the signal in the given window. For FFT, one has to decide which and how many coefficients are used. Typically, either the n first or the n largest (by amplitude) coefficients are used [28]. Figure 4 shows that in our case, on average, the amplitudes of FFT coefficients decrease monotonically. This means that the first coefficients are the largest ones, which in turn means that the lower frequencies are dominant. Therefore, using the amplitudes of the first n coefficients is a good way to proceed. For selecting a suitable number of first FFT coefficients, we use again cross-validation to choose amongst the following options: $n \in \{5, 6, 10, 15\}$.

In total we extract 78 statistical features from bending sensors, 30 statistical features from pressure sensors, 336 statistical features and $n \times 56$ FFT features for IMUs, where n is the number of first FFF components used for the window. It is worth pointing out that the majority of the features come from motion sensors (IMUs).

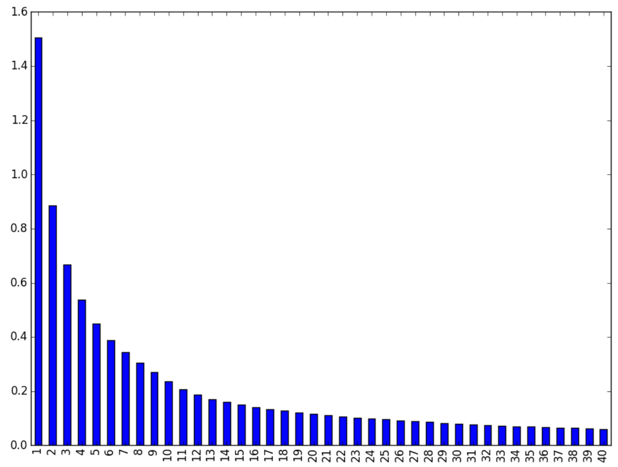

Figure 4: Average amplitudes (over all signals) of the first FFT coefficients (excluding the zeroth) for all 200 frame windows

To avoid correlated features, we calculated correlations between features and automatically removed the features that highly correlate with each other (with absolute Pearson correlation index more or equal to 0.99). Finally, extracted features are normalised with zero mean and a standard deviation of 1.

5.4 Procedure for Algorithm and Parameter Selection

In this section, we describe our procedure to select the best performing algorithm and parameters such as length and step-size for sliding windows, and the number of FFT coefficients to be used as features. We evaluated window sizes of $140, 160, 180, 200$, and window step-sizes of $20, 30, 40, 50$, as well as a number of FFT coefficients of $5, 6, 10, 15$. The choice of these options is discussed in Sections 5.2 and 5.3.

As previously mentioned, similar to activity recognition solutions, we emphasise the motion sensors and follow an approach (sliding windows) that is frequently used in activity detection. Therefore, we chose classification algorithms that have proven to provide robust performance on activity recognition using wearable sensors [21, 22, 46, 47], namely:

- K Nearest Neighbours (KNN)
- Linear Discriminant Analysis Classifier (LDAC)
- Support Vector Machines (SVM) with a linear kernel
- Logistic Regression (LR)

According to the survey presented in [3], discriminative classification algorithms are very effective in identifying features that mostly contribute to discriminations between activities using wearable sensors. Therefore, our discriminative classification algorithms (in our case SVM and LR) should work very well in case the gesture is well captured by extracted features of windows. LDAC is suitable when a linear transformation (LDA) of the data yields in linearly separable classes (in the transformed space). On the other hand, KNN uses the notion of distance in feature space and it can perform good even when linear separability is not possible.

Considering the large number of features we have (724 when using 5 FFT components, 1284 when using 15 of them), we were concerned about overfitting. Therefore, we employed dimensionality reduction techniques prior to training. We used Principal Component Analysis (PCA) which applies an orthogonal linear transformation of the data, in an unsupervised manner, resulting in a maximised variance of data in the transformed space. On the other hand, we also used the supervised linear transformations, namely Linear Discriminant Analysis (LDA) and also it's state-of-the-art alternative Spectral Regression Discriminant Analysis (SRDA) [4]. The latter methods utilise class labels for minimising the within-class variance and maximising between-class variance (in the transformed space). An extensive analysis on how the used algorithms and dimensionality reduction work, including the mathematics behind it can be found in [2]. In our cross-validation process, the classifiers have been trained in both ways: without any dimensionality reduction and with prior dimensionality reduction transformations.

For each window and step-size, we prepared the dataset as follows: We selected only those windows from the complete dataset that are "unambiguous windows", i.e. windows that contain either no gesture, or a full gesture (see Figure 3, where window 1 contains no gesture, windows 2 and 5 contain a partial gesture and are not part of the training data set, and windows 3 and 4 contain a full gesture). The rationale was that the classifier should only learn the full gestures, not parts of gestures.

Moreover, the classes in our data are unbalanced as the majority of the windows are labeled as "idle class" (no gesture). Balancing was achieved by the following procedure: First we calculate the average number of windows per gesture which we will denote by k. Then we removed, before splitting to train and test set, all idle class windows except k random number of idle class windows from the data set. It is worth mentioning that, the balancing was used only during the algorithm and parameter selection process but not during the algorithm evaluation process (Section 6).

The respective training data set was then 80% randomly chosen windows, and the test data set the remaining 20%. We used the following procedure to select the winner combination of configuration (parameters) and algorithm:
First, we compute the performance of each combination "configuration/algorithm" by 5-fold cross validation over the training data set.

Then, for each configuration we compute the average performance over all algorithms to select the winner configuration. The winner algorithm would then be the best-performing algorithm for this configuration. The rationale for this procedure was that we wanted to have the configuration to be as robust as possible in relationship to an algorithm in order to avoid overfitting, i.e. we did not want to select a configuration that only works for a single algorithm.

5.5 Algorithm and Parameter Selection Results

The procedure of selecting and parametrising a classification algorithm that we described above in Section 5.4 yielded the following: The best configuration is the one with a window length of 200 frames, step-size of 20 and 15 FFT coefficients with an average cross-validation (across all compared algorithms) score of 95.6%. Another configuration with less computationally intensive parameters, namely window length of 200 frames, window steps for the sliding windows of 50 and only 5 FFT coefficients had an average cross validation score of 95.3%. Considering that the score difference is minimal whereas the computation efficiency is higher, we selected the latter configuration. For this configuration, the best performing algorithm was LDA+LR with an cross-validation f_1 score of 99.8%. Here, initially LDA was used to perform dimensionality reduction to 32 components and then a logistic regression algorithm was trained and tested on the dimensionally reduced data.

6 Algorithm Evaluation

In this section, we report on accuracies on the full dataset for the selected algorithm and configuration, which constitutes a realistic scenario of continuous data stream analysis.

6.1 Algorithm Performance on Continuous Sensor Data

As realistic algorithm performance, we consider its performance on the following dataset: All windows are used in the test data set, which includes those with a partial gesture windows. A partial gesture window is the one that contains only a portion of a gesture (see window 2 and 3 in Figure 3). Moreover, there is no balancing (neither in the training nor test data set) but class weighting is used when training in order to prevent bias towards the larger classes. This corresponds to the data that would be available in a real world continuous sensor data stream. For windows that contain a partial gesture, we assume the algorithm prediction is correct when the classification outcome is either the idle class or the correct gesture class that is partially in the window. We refer to this strategy as dual labelling in the test set.

On this test set, the LDA+LR algorithm with a window size of 200, a step size of 50, and with only the 5 first FFT gestures in the feature set, performs with an 98.5% f_1 score. The confusion matrix is given in Table 2 and the receiver operating characteristic (ROC) curve is visualised in Figure 5. Here, from 9581 windows, 9440 were classified correctly. From the correctly classified windows, 1618 contained full gestures, 2802 partial gestures and 5020 contained no gesture at all (belonged to the idle class). On the other hand, 141 windows were misclassified from which 6 contained full gestures, 106 partial gestures and 29 came from the idle class.

6.2 Algorithm Performance without FFT Components

Removing FFT calculations during the gesture extraction can speed up the processing the data stream. The rationale details for such an optimisation is discussed in Section 7.2 below. Removing the FFT components from all accelerometer and gyroscope dimensions results in a recognition f_1 score (when considering dual labelling of the ambiguous windows in test set) of 98.2%.

7 Discussion

As our results reveal, we achieve a high classification accuracy in general. As presented in Figure 5, the prediction confidence is also high. It is important to stress that our results were achieved by including in the test set the windows that contain partial gestures. In a live gesture recognition system, there is no way of excluding them. More specifically, in a live scenario, we need to get a sliding window over a stream of data, as visualised in Figure 3, and since we don't know when a gesture starts and when it ends, we can't know beforehand whether a partial gesture is in a window. In the test set, we used the dual labelling strategy which delivers an accuracy of 98.5% (see Table2 and Figure 5). We argue that dual labelling is acceptable for a live system: In the end, it is just a matter of how fast the recognition system realises that the gesture is being performed. In case it predicts the correct gesture class (the one that is partially contained in window), then we can recognise the gesture even before it is completed. Otherwise, if the classifier predicts it to be a idle class window, then the next window (or previous one) will be a window with the full gesture in it and will be classified correctly.

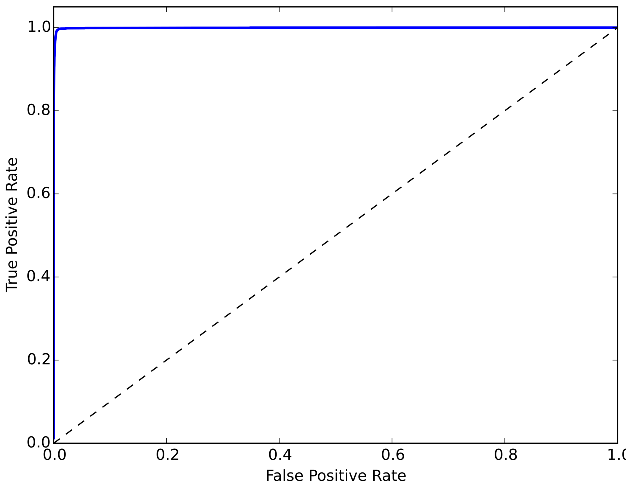

Figure 5: ROC curve for weighted average of all classes

7.1 Analysis of Confusions between Classes

Although overall performance is really good, it is not perfect: some windows are misclassified. In this section, we analyse the confusions between classes. As presented in Section 6, there are 141 misclassifications out of 9440 windows, of which 106 misclassified windows contain partial gestures. We have only 6 misclassifications from windows that contained full gestures. In the following discussion, we focus on windows that contain only partial gestures on them. Several such windows are being misclassified as another gesture and this phenomenon affects mainly the following 10 classes: Down, Swipe Left, Push Away, Swipe Left, Grasp 2, Point at Self, Swipe Left, Go Away, Grasp 1 and Zoom.

A further investigation shows that, there are some systematic misclassification for such windows. For the **"Down"** gesture, nine misclassifications come from windows that partially contain the "Shoulder pat" gesture, five from ones that partially contain the "Continue" gesture and six from windows that contain partial "Swipe left" gestures. When looking on how the gestures movements are performed, the "Shoulder pat" starts and ends very similarly to the "Down" gesture. So, in case the start or the end of the gesture is missing in a window, it seems quite logical that such a confusion can happen. "Continue" could either start as "Down" or as a "Swipe Left" gesture (depending from person to person). Looking into each of those windows reveals that seven out of nine confusions with "Shoulder pat"contain the beginning (on average 40%) of the of the "Shoulder pat" gesture whereas only two out of

	Nothing	(1) One	(2) Two	(3) Three	(4) Four	(5) Five	Thumbs up	Thumbs down	Point to self	Shoot	Scissor	Cutthroat	Continue	Knocking	Waving	Come here	Go away	Push away	Never mind	Talking	Calling	Walking	Shoulder pat	Point	Swipe left	Swipe right	Swipe up	Down	Turn	Zoom	Grasp 1	Grasp 2
Nothing	6024	2	1	5		4	3	1	7	1	1	4	2		1		6	12	2		2		3	5	15	3	4	28	2	7	6	8
(1) One		107																														
(2) Two			113																													
(3) Three				110																									1			
(4) Four					116																											
(5) Five						97																										
Thumbs up							108																									
Thumbs down								116																								
Point to self									117																							
Shoot										89																						
Scissor											126																					
Cutthroat												110																				
Continue													96												1							
Knocking														123																		
Waving															133																	
Come here																111																
Go away																	93															
Push away																		111														
Never mind																			80													
Talking																				107												
Calling												1									128											
Walking																						110										
Shoulder pat																							113									
Point																								112								
Swipe left																									103							
Swipe right																										95						
Swipe up																											108					
Down	1																								1			107				
Turn																													127			
Zoom																														95		
Grasp 1																															138	1
Grasp 2																																117

Table 2: Confusion matrix of classification using dual labelling in test set. Note that zero values (no misclassification) have been removed from the table for better readability

nine contain the end (on average 66%) of the gesture. However for both cases, the prediction confidence is not particularly high: 66% respectively 46%. Confusions of the "Down" gesture with the gesture "Continue" are predicted with an average confidence is 73% and all of them happen in windows that contain only the beginning (on average 30%) of the "Continue" gesture. Similarly, all five windows that are confused with the "Swipe left" gesture contain only the beginning of the "Swipe left" gesture (on average 29%) and the prediction confidence is 55%. Only one window that contains the end of the gesture (51%) is confused with "Down", with a confidence of 73%.

"Swipe Left" class has nine confusions with windows that partially contain "Continue", all of them containing only the end (35%) of the gesture. Two more confusions happen with windows that partially contain "Never Mind" gesture and all of them contain only the last part of the gesture (70%). Interestingly, those confusions have a relative high prediction confidence (average: 90%). Three more confusions happen with windows that contain only the last part (32%) of the "Waving" gesture with a relatively quite low average prediction confidence (58%).

For the **"Push away"**, four misclassifications come from windows with partial "Continue" gesture. Three of them contain the beginning (44%) of the gesture and are misclassified with an average confidence of 44%, whereas the other one contains the last

Classified	Full/Partial Gesture	Confidence
Correctly	Full	98%
Correctly	Partial	89%
Incorrectly	Full	67%
Incorrectly	Partial	57%

Table 3: Average classification confidence depending on whether the window contains the whole gesture

(73%) part of the gesture is predicted with a confidence of 26%. Two other confusions happen with partial "Never mind" gestures. The both contain only the last part (39%) of the gesture and are predicted with average confidence of 46%.

It therefore seems that the gestures: "Down", "Push away", "Continue" and "Shoulder pat" are very similar either at the beginning or at the end of the gesture. For the other gestures there does not seem to be a systematic misclassification as the confusions are distributed among most of the other classes.

To address the partial window misclassification problem, we investigate whether there is a difference in prediction confidence for such misclassification in comparison to other windows. From Table 3 we can see that the misclassified windows that contain only partial gestures have quite a low average prediction confidence

(57%). One way to minimise the number of misclassifications would therefore be to accept only predictions with higher confidence. According to Table 3, such a setting would mainly affect misclassified windows and especially those with partial gestures on them. This would be acceptable in a live recognition system as when a gesture is performed, there will be window(s) with the whole gesture on it where prediction confidence would be higher. Therefore, the practical classification performance would not be affected since the gesture would still be detected either before or afterwards. The only practical implication is how early a gesture can be recognised.

7.2 Computing efficiency

There are several performance indicators to consider when providing a gesture recognition system for human computer interaction. Besides the accuracy, which is the most important, recognition speed (delays hurt the user experience [45]) and power consumption important. Recognition speed and power consumption directly relate to computational cost. Both are of particular relevance if the gesture recognition is planned to be carried out in a mobile or embedded device, which seems very practical especially for wearable sensor based systems for gesture recognition. the Section 5.4, we mentioned that we chose the sliding window step of 50 and the number of FFT coefficients to be 5, even though the configuration was not the best one in terms of cross-validation score. The rationale behind reducing the number of FFT coefficients was that it reduces the number of features and therefore the computational costs and therefore power consumption. Choosing a step-size of 50 frames instead of 20, enables us to perform predictions less often (only after each 50 frames). On the other hand of course, a larger window step means a larger delay: In our case, 50 frames window size means a delay of 0.625 seconds, which we deemed acceptable for the time being. In the end, this is a trade-off in system design of course.

Removing all FFT components (as described in Section 6.2), on the other hand, results in a more significant reduction of computational costs. During the feature extraction process, descriptive statistical calculations like min, max, average, energy and standard deviation can be calculated with a time complexity of $O(n)$, where n is the number of points (window size). On the contrary, the computation of FFT has a time complexity of $O(nlogn)$ and it is performed for every data dimension d that contains accelerometer or gyroscope values ($d = 56$). In addition, for every such calculation the amplitudes of the first k coefficients need to be extracted with a time complexity of $O(k)$. By having only descriptive statical features that are calculated in $O(n)$, the feature extraction process results in a total complexity of $O(fn)$, where f is the number of features. It is worth mentioning that removing FFT components also results in 280 less features. Moreover, descriptive statistic can easily be computed incrementally for a sliding window in a data stream [42] (see the online algorithm[6]). To our knowledge, there is no way of calculating FFT incrementally in a sliding window.

Removing FFT also reduces the complexity in the classification process, though only by a constant factor. First, LDA performs a matrix multiplication (of dimensions $f \times c$) to transform data into the new space. For a single window (of dimensions $1 \times f$), this transformation has a time complexity of $O(fc)$, where c is the dimension of the new space, which in our case equals the number of classes. Note that $c \leq f$. After this transformation, the multinomial logistic regression is applied for classification. Given the reduced dimensions of the window ($1 \times c$), it has a time complexity of $O(c^2)$ (where c is the number of classes), making the whole classification complexity of $O(fc)$. By removing FFT components we reduce f by 280 which impacts the computational complexity of classification. Note that we do not discuss the time complexity of training process as the training can be done offline and therefore it does not play a role in the live system performance.

[6]https://en.wikipedia.org/wiki/Algorithms_for_calculating_variance

8 Conclusion

In this paper, we presented a gesture recognition system built for recognising 31 natural and interaction-oriented hand gestures. Our feature extraction is based on statistics and spectral properties of a sliding window over the data stream. We show that our features are highly discriminative for natural hand gestures and we achieve an accuracy of 98.5% with our gesture recognition system, which relies on linear discriminant analysis for dimensionality reduction and logistic regression for classification. Moreover, accuracy does not significantly suffer (98.2%) when the computationally expensive FFT features are removed. The main contribution of this paper lies in showing that all selected gestures can be recognised very well, given the sensors on the custom data glove and selected features extracted using sliding window approach.

This result is relevant for gesture-based interfaces, as it means that continuous gesture detection based on continuous sensing is accurate enough; and can be implemented in a computationally efficient manner. Computational efficiency is particularly important in wearable systems, considering the mobile nature of such systems. One direction of future work following up this line of argumentation will be an implementation of the recognition system on smartphone (wirelessly connected to the glove), making it a complete mobile solution. Further directions of future work on computing efficiency will include an exploration of which sensors are irrelevant and could be completely removed without degrading the recognition accuracy. Less sensors means less features, and hence more computational efficiency, but also fewer sensors to supply with power. Spreading out from the core of gesture recognition, it will of course be interesting to design interactions with computer systems using such natural and interaction-oriented gestures that can be recognised well.

Acknowledgements

Work on this paper is partially funded by MoreGrasp (H2020-ICT-2014-1 643955). The Know-Center is funded within the Austrian COMET Program - Competence Centers for Excellent Technologies - under the auspices of the Austrian Federal Ministry of Transport, Innovation and Technology, the Austrian Federal Ministry of Economy, Family and Youth and by the State of Styria. COMET is managed by the Austrian Research Promotion Agency FFG.

We thank Eduardo Veas for his advice for building the data glove and planning the experiment study, and Christoffer Öjeling for his work on the data glove and his help in preparing and carrying out the experiment.

References

[1] H. Birk, T. B. Moeslund, and C. B. Madsen. Real-time recognition of hand alphabet gestures using principal component analysis. In *In 10th Scandinavian Conference on Image Analysis*, 1997.

[2] C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[3] A. Bulling, U. Blanke, and B. Schiele. A tutorial on human activity recognition using body-worn inertial sensors. *ACM Comput. Surv.*, 46(3):33:1–33:33, Jan. 2014.

[4] D. Cai, S. Member, X. He, J. Han, and S. Member. Srda: An efficient algorithm for large-scale discriminant analysis. *IEEE Transactions on Knowledge and Data Engineering*, pages 1–12, 2008.

[5] Q. Chen, N. D. Georganas, and E. Petriu. Real-time vision-based hand gesture recognition using haar-like features. In *Instrumentation and Measurement Technology Conference Proceedings, 2007. IMTC 2007. IEEE*, pages 1–6, May 2007.

[6] E. Coupet, F. Moutarde, and S. Manitsaris. Gesture recognition using a depth camera for human robot collaboration on assembly line.

Procedia Manufacturing, 3:518 – 525, 2015. 6th International Conference on Applied Human Factors and Ergonomics (AHFE 2015) and the Affiliated Conferences, {AHFE} 2015.

[7] N. Dardas and N. D. Georganas. Real-time hand gesture detection and recognition using bag-of-features and support vector machine techniques. *Instrumentation and Measurement, IEEE Transactions on*, 60(11):3592–3607, Nov 2011.

[8] T. Dietterich. Machine learning for sequential data: A review. In T. Caelli, A. Amin, R. Duin, D. de Ridder, and M. Kamel, editors, *Structural, Syntactic, and Statistical Pattern Recognition*, volume 2396 of *Lecture Notes in Computer Science*, pages 15–30. Springer Berlin Heidelberg, 2002.

[9] P. Garg, N. Aggarwal, and S. Sofat. Vision based hand gesture recognition. *World Academy of Science, Engineering and Technology*, 49(1):972–977, 2009.

[10] P. Glomb, M. Romaszewski, S. Opozda, and A. Sochan. Choosing and modeling the hand gesture database for a natural user interface. In *Proceedings of the 9th International Conference on Gesture and Sign Language in Human-Computer Interaction and Embodied Communication*, GW'11, pages 24–35, Berlin, Heidelberg, 2012. Springer-Verlag.

[11] J. Han, S. Ahn, and G. Lee. Transture: Continuing a touch gesture on a small screen into the air. In *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems*, CHI EA '15, pages 1295–1300, New York, NY, USA, 2015. ACM.

[12] M. M. Hasan and P. K. Mishra. Hand gesture modeling and recognition using geometric features: a review. *Canadian Journal on Image Processing and Computer Vision*, 3(1):12–26, 2012.

[13] W. Higgins. A comparison of complementary and kalman filtering. *Aerospace and Electronic Systems, IEEE Transactions on*, AES-11(3):321–325, May 1975.

[14] C.-L. Huang and W.-Y. Huang. Sign language recognition using model-based tracking and a 3d hopfield neural network. *Mach. Vision Appl.*, 10(5-6):292–307, Apr. 1998.

[15] J.-F. Jego, A. Paljic, and P. Fuchs. User-defined gestural interaction: A study on gesture memorization. In *3D User Interfaces (3DUI), 2013 IEEE Symposium on*, pages 7–10, March 2013.

[16] R. E. Kalman. A new approach to linear filtering and prediction problems. *Journal of Fluids Engineering*, 82(1):35–45, 1960.

[17] M. Kavakli, M. Taylor, and A. Trapeznikov. Designing in virtual reality (desire): A gesture-based interface. In *Proceedings of the 2Nd International Conference on Digital Interactive Media in Entertainment and Arts*, DIMEA '07, pages 131–136, New York, NY, USA, 2007. ACM.

[18] H. Kenn, F. V. Megen, and R. Sugar. A glove-based gesture interface for wearable computing applications. In *Applied Wearable Computing (IFAWC), 2007 4th International Forum on*, pages 1–10, March 2007.

[19] E. C. Kiruthika, E. Coimbatore, and N. Kumar. Survey on hand gesture recognition. *International Journal of Engineering Research and Technology*, 3(2):943–946, 2014.

[20] H. Koskim"aki, V. Huikari, P. Siirtola, and J. R"oning. Behavior modeling in industrial assembly lines using a wrist-worn inertial measurement unit. *Journal of Ambient Intelligence and Humanized Computing*, 4(2):187–194, 2013.

[21] H. Koskimaki, V. Huikari, P. Siirtola, P. Laurinen, and J. Roning. Activity recognition using a wrist-worn inertial measurement unit: A case study for industrial assembly lines. In *Control and Automation, 2009. MED '09. 17th Mediterranean Conference on*, pages 401–405, June 2009.

[22] N. C. Krishnan and D. J. Cook. Activity recognition on streaming sensor data. *Pervasive Mob. Comput.*, 10:138–154, Feb. 2014.

[23] A. Kulshreshth and J. J. LaViola, Jr. Exploring 3d user interface technologies for improving the gaming experience. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, pages 125–134, New York, NY, USA, 2015. ACM.

[24] P. Kumar, J. Verma, and S. Prasad. Hand data glove: A wearable real-time device for human-computer interaction. *International Journal of Advanced Science and Technology*, 43, 2012.

[25] B.-G. Lee, B.-L. Lee, and W.-Y. Chung. Wristband-type driver vigilance monitoring system using smartwatch. *Sensors Journal, IEEE*, 15(10):5624–5633, Oct 2015.

[26] T. Martins, C. Sommerer, L. Mignonneau, and N. Correia. Gauntlet: a wearable interface for ubiquitous gaming. In *MobileHCI '08: Proceedings of the 10th international conference on Human computer interaction with mobile devices and services*, page 367, New York, New York, USA, Sept. 2008. ACM Request Permissions.

[27] P. Molchanov, S. Gupta, K. Kim, and K. Pulli. Multi-sensor system for drivers hand-gesture recognition. In *IEEE Conference on Automatic Face and Gesture Recognition*, pages 1–8, 2015.

[28] F. Mörchen. Time series feature extraction for data mining using dwt and dft. Technical report, Math and Computer Science Department, Philipps University, Marburg, Germany, 2003.

[29] K. Murakami and H. Taguchi. Gesture recognition using recurrent neural networks. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '91, pages 237–242, New York, NY, USA, 1991. ACM.

[30] P. Neto, D. Pereira, J. Norberto Pires, and A. Moreira. Real-time and continuous hand gesture spotting: An approach based on artificial neural networks. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 178–183, May 2013.

[31] J. Ortiz Laguna, A. Olaya, and D. Borrajo. A dynamic sliding window approach for activity recognition. In J. Konstan, R. Conejo, J. Marzo, and N. Oliver, editors, *User Modeling, Adaption and Personalization*, volume 6787 of *Lecture Notes in Computer Science*, pages 219–230. Springer Berlin Heidelberg, 2011.

[32] J. Ou, Y. Shi, J. Wong, S. R. Fussell, and J. Yang. Combining audio and video to predict helpers' focus of attention in multiparty remote collaboration on physical tasks. In *Proceedings of the 8th International Conference on Multimodal Interfaces*, ICMI '06, pages 217–224, New York, NY, USA, 2006. ACM.

[33] C. Oz and M. C. Leu. American sign language word recognition with a sensory glove using artificial neural networks. *Eng. Appl. Artif. Intell.*, 24(7):1204–1213, Oct. 2011.

[34] G. Ren and E. O'Neill. Freehand gestural text entry for interactive tv. In *Proceedings of the 11th European Conference on Interactive TV and Video*, EuroITV '13, pages 121–130, New York, NY, USA, 2013. ACM.

[35] A. Roitberg, N. Somani, A. Perzylo, M. Rickert, and A. Knoll. Multimodal human activity recognition for industrial manufacturing processes in robotic workcells. In *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction*, ICMI '15, pages 259–266, New York, NY, USA, 2015. ACM.

[36] M. Romaszewski, P. Glomb, and P. Gawron. Natural hand gestures for human identification in a human-computer interface. In *Image Processing Theory, Tools and Applications (IPTA), 2014 4th International Conference on*, pages 1–6, Oct 2014.

[37] T. J. Sejnowski and C. R. Rosenberg. Parallel networks that learn to pronounce english text, 1987.

[38] S. Sen, V. Subbaraju, A. Misra, R. K. Balan, and Y. Lee. The case for smartwatch-based diet monitoring. In *Pervasive Computing and Communication Workshops (PerCom Workshops), 2015 IEEE International Conference on*, pages 585–590, March 2015.

[39] M. Shoaib, S. Bosch, H. Scholten, P. J. Havinga, and O. D. Incel. Towards detection of bad habits by fusing smartphone and smartwatch sensors. In *Pervasive Computing and Communication Workshops (PerCom Workshops), 2015 IEEE International Conference on*, pages 591–596, March 2015.

[40] T. Stiefmeier, D. Roggen, G. Ogris, P. Lukowicz, and P. Lukowicz. Wearable activity tracking in car manufacturing. *Pervasive Computing, IEEE*, 7(2):42–50, April 2008.

[41] T. Takahashi and F. Kishino. Hand gesture coding based on experiments using a hand gesture interface device. *SIGCHI Bull.*, 23(2):67–74, Mar. 1991.

[42] K. Tangwongsan, M. Hirzel, S. Schneider, and K.-L. Wu. General incremental sliding-window aggregation. *Proc. VLDB Endow.*, 8(7):702–713, Feb. 2015.

[43] T.-H. Tsai, C.-C. Huang, and K.-L. Zhang. Embedded virtual mouse system by using hand gesture recognition. In *Consumer Electronics - Taiwan (ICCE-TW), 2015 IEEE International Conference on*, pages

352–353, June 2015.

[44] W. Van Vlaenderen, J. Brulmans, J. Vermeulen, and J. Schöning. Watchme: A novel input method combining a smartwatch and bimanual interaction. In *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems*, CHI EA '15, pages 2091–2095, New York, NY, USA, 2015. ACM.

[45] J. P. Wachs, M. Kölsch, H. Stern, and Y. Edan. Vision-based hand-gesture applications. *Commun. ACM*, 54(2):60–71, Feb. 2011.

[46] J. Ward, P. Lukowicz, G. Troster, and T. Starner. Activity recognition of assembly tasks using body-worn microphones and accelerometers. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(10):1553–1567, Oct 2006.

[47] J. A. Ward. *Activity monitoring: Continuous recognition and performance evaluation*. PhD thesis, Swiss Federal Institute of Technology (ETH) Zürich, 2006.

[48] J. Weissmann and R. Salomon. Gesture recognition for virtual reality applications using data gloves and neural networks. In *Neural Networks, 1999. IJCNN '99. International Joint Conference on*, volume 3, pages 2043–2046 vol.3, 1999.

[49] C. Xu, P. H. Pathak, and P. Mohapatra. Finger-writing with smartwatch: A case for finger and hand gesture recognition using smartwatch. In *Proceedings of the 16th International Workshop on Mobile Computing Systems and Applications*, HotMobile '15, pages 9–14, New York, NY, USA, 2015. ACM.

[50] D. Xu. A neural network approach for hand gesture recognition in virtual reality driving training system of spg. In *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on*, volume 3, pages 519–522, 2006.

[51] P. Zappi, T. Stiefmeier, E. Farella, D. Roggen, L. Benini, and G. Troster. Activity recognition from on-body sensors by classifier fusion: sensor scalability and robustness. In *Intelligent Sensors, Sensor Networks and Information, 2007. ISSNIP 2007. 3rd International Conference on*, pages 281–286, Dec 2007.

[52] X. Zhang, X. Chen, W.-h. Wang, J.-h. Yang, V. Lantz, and K.-q. Wang. Hand gesture recognition and virtual game control based on 3d accelerometer and emg sensors. In *Proceedings of the 14th International Conference on Intelligent User Interfaces*, IUI '09, pages 401–406, New York, NY, USA, 2009. ACM.

[53] Y. Zhao, P. H. Pathak, C. Xu, and P. Mohapatra. Demo: Finger and hand gesture recognition using smartwatch. In *Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services*, MobiSys '15, pages 471–471, New York, NY, USA, 2015. ACM.