J Braz Comput Soc (2010) 16: 147–162
DOI 10.1007/s13173-010-0009-z

ORIGINAL PAPER

Comparative evaluation of static gesture recognition techniques based on nearest neighbor, neural networks and support vector machines

Alexandre Savaris · Aldo von Wangenheim

Received: 27 November 2009 / Accepted: 29 April 2010 / Published online: 9 June 2010

Abstract It is a common behavior for human beings to use gestures as a means of expression, as a complement to speaking, or as a self-contained communication mode. In the field of Human–Computer Interaction, this behavior can be adopted to build alternative interfaces, aiming to ease the relationship between the human element and the computational element. Currently, various gesture recognition techniques are described in the technical literature; however, the validation studies of these techniques are usually performed isolatedly, which complicates comparisons between them. To reduce this gap, this work presents a comparison between three well-established techniques for static gesture recognition, using Nearest Neighbor, Neural Networks, and Support Vector Machines as classifiers. These classifiers evaluate a common dataset, acquired from an instrumented glove, and generate results for precision and performance measurements. The results obtained show that the classifier implemented as a Support Vector Machine presented the best generalization, with the highest recognition rate. In terms of performance, all methods presented evaluation times fast enough to be used interactively. Finally, this work identifies and discusses a set of relevant criteria that must be observed for the training and evaluation steps, and its relation to the final results.

A. Savaris · A. von Wangenheim
Department of Informatics and Statistics (INE), Federal University of Santa Catarina (UFSC), Campus Universitário Trindade, 88040-900, Florianópolis, SC, Brazil

A. Savaris
e-mail: asavaris@inf.ufsc.br

A. von Wangenheim (✉)
e-mail: awangenh@inf.ufsc.br

Keywords Human–computer interaction · Gesture recognition · Alternative interfaces · Nearest neighbor · Neural networks · Support vector machines

1 Introduction

The discipline of Human–Computer Interaction (HCI) addresses the design, implementation, and evaluation of techniques to build interfaces between the human element and the computational element [1]. It is a multidisciplinary study area that involves knowledge from computer science, psychology, sociology, anthropology, and industrial design, among other knowledge sources. The combination of these knowledge sources allows the creation of interactive techniques, which ease the relationship between people (individually or collectively) and computers (through hardware devices and application software).

In the last decades, the study and development of interaction techniques have established landmarks that guided the relationship between man and machine. Among these landmarks, Myers [2] highlights the following: Graphical objects (presentation and manipulation), mouse as a pointing device, window-based interfaces, text editors, spreadsheets, computer-aided design software (CAD), hypertext, and video games. Virtual reality, natural language recognition and 3D object representation can also be considered the building blocks for a number of different technologies used today.

For a long time, interactivity was restricted to the keyboard/mouse pair; all operations performed by users were mapped to a set of mouse motions or keyboard keys. Currently, there is a tendency to use more intuitive actions to perform the same operations. For instance, head movements can change the field of view in an immersive application, and multitouch surfaces [3, 4] can provide input/output, allowing users to interact using their fingertips combined with hand movements. Also, free gestures can be interpreted and translated to application events, with few or even no restrictions imposed by former input devices [5–7].

The development of gesture-based interfaces presents a challenge: How to correctly interpret gestures, avoiding false positive and false negative recognitions? Some questions surround this challenge, and must be answered during the project phase of such interfaces: Is there a technique suitable for a wide range of gesture vocabularies? Is a given technique fast enough to be applied in a reasonable time? How extensive can be the vocabulary, and what are the characteristics of the gestures to be used?

The literature presents different approaches for the interpretation of gestures, but these approaches are generally validated only in a very superficial way, without the presentation of quantitative data. A systematic, quantitative comparison between different approaches lacks altogether. In this context, the objective of this work is to answer some of the questions stated above, systematically comparing a number of common gesture recognition techniques against each other and presenting quantitative and reproducible results in terms of *precision* and *performance*. The chosen techniques are largely used in gesture recognition studies, and were adapted to interpret a set of static hand gestures. As its main contribution, this work identifies the method that can be best applied to recognize gestures, like those that make up the vocabulary used on validation sessions. Besides, in order to allow the results of this work to be tested against other approaches or re-validated by other researchers, we made the instrumented data publicly available.[1]

The paper is organized as follows: Basic concepts on static and dynamic gestures, as well as details about the gesture recognition process, are presented in Sect. 2. Section 3 lists related research efforts on measuring the accuracy of techniques for gesture recognition and interpretation. Section 4 describes the experimental environment and Sect. 5 presents the obtained results. Finally, Section 6 presents a discussion about the obtained results and Sect. 7 highlights the conclusions and future work.

2 Gestures and the gesture recognition pipeline

As interface components, gestures can be classified according to their type (*static* or *dynamic*). This classification organizes the vocabulary used by applications, and guides the choice of appropriate methods for gesture recognition. This section presents the differences between these two types of gestures, and how these differences guide the organization and implementation of a gesture recognition process.

2.1 Gesture types

According to [8], a *static gesture* (or *posture*) is a static movement, which occurs at a specific moment in time. The posture can be formed by the entire body or one of its parts (e.g., an arm or a hand). In turn, a *dynamic gesture* (or simply *gesture*) is a dynamic movement, which spans a time interval and delineates a specific trajectory. Just as static gestures, their dynamic counterpart can be executed by the entire body or by a part of it.

The static and dynamic gestures (as well as the combination of both types) are used in a number of different works that aim to improve the interaction experience. Iwai et al. [9], for example, recognize mimics of musical instruments and greetings from the Japanese signal language through dynamic gestures using static gestures as start and stop marks. Lee et al. [10] control avatars through static and dynamic gestures, and Tani et al. [5] relate static and dynamic gestures to events for manipulation of radiological images. In all cases, the interaction technique (gestures) is the same; the difference is in the application context.

2.2 The gesture recognition process

In order to contextualize our work, we will first offer a short review on the structure of the gesture recognition process as a whole, addressing the weaknesses and caveats involved in each step and their implications.

To use gestures as an interaction technique, a set of steps must be followed. The number of steps varies according to the hardware used, the type of gesture, and the application context, as can be seen in Kjellström et al. [11] and Kim et al. [12]. Despite the particularities, it is possible to identify a sequential arrangement of these steps in a pipeline and their organization in three actions: *Data acquisition*, *data interpretation,* and *software integration*.

The data acquisition step collects and stores data pertinent to gestures. There are two common sources for these data: Video streams/sets of static images (used on computer vision-based interfaces) and posture/motion signals (used on instrumented interfaces). The former uses a camera—or a set of cameras—to acquire visual information of a gesture; the latter uses signals generated by instrumented gloves and motion trackers, which can be translated to postures and gestures executed in an n-dimensional space.

Both types of data (visual and instrumented) are predisposed to different types of noise. The presence of noise does not hinder the recognition process, but influences the results obtained by the data interpretation step. To minimize this problem, two optional actions can be performed on the collected data: Normalization and filtering. Through normalization, time and space limits are established, allowing sparse gestures to fit predefined intervals. Filtering, in turn,

[1] http://www.lapix.ufsc.br/index.php/gil-gesture-interaction-layer.

cleans the original data by suppressing irrelevant information.

After the acquisition step, the obtained data are submitted to a technique (or a set of techniques) that compose the data interpretation step. At this point, the vocabulary of gestures is known, and the algorithms that implement the techniques are trained to classify the new data. The training process is based on a set of gestures previously recorded (called *training set*), composed by examples of each gesture presented in the vocabulary.

The algorithms used for gesture recognition are chosen to be employed in the data interpretation step. Among a number of relevant criteria, the type of available data defines the behavior of algorithms. For instance, the classification of a static gesture uses a set of characteristics in a specific moment in time; on the other hand, the classification of a dynamic gesture uses various sets of characteristics related to a time interval. In both cases, there is one gesture to be classified; however, the input data are different, and the chosen algorithms must deal with this difference.

Finally, in the software integration step, the results of data interpretation are translated to application events—which can be the execution of a particular function, the updating of the system's current state or the change of the application's interaction mode [13]. The rules for the translation of gestures to events are set individually for each application, according to its context and usability.

At this point, the static and dynamic gestures can be treated together (as complementary gestures) or individually. When treated together, static gestures usually mark the start and end of an interaction, which is completed by the dynamic gesture. The mouse motion in a 2D application and the navigation in a 3D virtual environment are examples of this combination. When considered individually, static and dynamic gestures can be translated to atomic events (like mouse clicks or commands to open menus or operate widgets).

3 Related work

Different techniques to perform accurate gesture recognition have been described in the literature. This section briefly presents related work considered relevant, focusing on the most common methods.

3.1 Neural networks

Fausett [14] sates that a neural network is an information-processing system, characterized by an *architecture*, a *learning process* and an *activation function*. It is composed by simple processing elements called *neurons*, which are connected to other neurons by means of connection links. Each link has an associated weight, which represent the information used by the network to solve a problem. The whole structure can be used to recognize and classify both static and dynamic gestures, due to its generalization characteristics. Xu et al. [6] developed a virtual training application of artillery, Self-Propelled Gun (SPG), operated through static gestures, and hand translations and rotations. The recognition of static gestures was performed by a feed-forward neural network with a single hidden layer, with 18 inputs (from an instrumented glove), 15 outputs (one for each gesture class), trained with the backpropagation algorithm. Their setup achieved a recognition rate of 98.0%, using a training set of 200 gestures and a test set of 100 gestures, performed by five different people. Stergiopoulou and Papamarkos [15] used an unsupervised neural classifier (SGONG—Self-Growing and Organized Neural Gas) to classify static gestures whose data were acquired from a camera, respecting the following assumptions: Gestures are executed with the right hand, with the arm in vertical position, palm facing the camera and fingers raised or not. Also, the technique considers a plain and uniform background. A recognition rate of 90.45% was achieved, using a set of 31 different gestures, with a mean processing time of 1.5 seconds per gesture. Bailador et al. [16] recognized dynamic gestures using a set of Continuous Time Recurrent Neural Networks (CTRNN). The gesture data were generated by accelerometers, and the neural networks were used to predict future acceleration values from current ones; the gesture related to the neural network with the smallest prediction error wins. Eight different classes of gestures made up the vocabulary, with a training set of 40 gestures and a test set of 120 gestures (all executed by one person). The method achieved a recognition rate of 94.0% in a controlled environment (with rest positions between the gestures), and a recognition rate of 63.6% in a realistic environment (with gestures been executed between routine activities, like sitting and standing up, among others).

3.2 Support vector machines

A Support Vector Machine is a classifier based on the mapping of characteristics extracted from instances—the feature vectors—to points in space [17]. In the training phase, the points are organized in different classes, divided by a clear gap that is as wide as possible, represented by a *hyperplane*. In the classification phase, new feature vectors are mapped to points and, subsequently, predicted to belong to a class established during the training phase. This type of classifier provides a binary classification, generating by default two classes divided by an ideal hyperplane; however, the definition of multiple evaluation classes is possible through the combination of multiple binary problems to a multiclass problem, based on strategies like *one-versus-all* or *one-versus-one*.

Ren and Zhang [18] used a Support Vector Machine combined with Minimum Enclosing Ball, a method they called MEB-SVM, to classify static gestures acquired from a video camera. After the image acquisition, image segmentation, contour selection, and classification, their work achieved a mean recognition rate of 92.89%. Liu et al. [19] used an SVM with Hu moments to classify hand postures acquired from a camera, automating the verification of hand integrality for the Chinese Driver Physical Examination System. An error rate of 3.5% was generated after tests were executed with data from 20 people. Chen and Tseng [20] developed a robotic visual system to recognize static gestures for finger guessing games. An SVM classifier was implemented and configured to be robust enough to work regardless of hand angles and skin colors. In their tests, their setup achieved a correct recognition rate of 95.0% for the paper, rock, and scissors game, using data from four people. Meng, Pears, and Bailey [21] presented a method to recognize human actions from video streams, using a linear SVM as classifier, trained with data acquired from Motion History Image (MHI) and Hierarchical Motion History Histogram (HMHH). Using examples of walking, jogging, running, boxing, hand clapping, and hand waving, recorded from 25 people in four different scenarios, the method achieved a maximum recognition rate of 93.1%.

3.3 Simple pattern recognition techniques

Other methods, commonly based on pattern similarity, can be adapted to the context of gestures. It is the case of nearest neighbor techniques, which perform the classifications based on distance metrics [22]. For each instance to be classified, the distance of its feature vectors to the feature vectors of already known classes is calculated, and the minor distance (or k minor distances, in the case of k-nearest neighbor) defines the result. The methods can be implemented with different data structures and distance metrics, achieving good evaluation times.

Deller et al. [7] made a simple comparison between data acquired from an instrumented glove and a library of static gestures to classify sequences of static gestures and wrist rotations. Their method calculates the distance between the acquired data and the training dataset; if the distance fits within a predefined threshold, the gesture is recognized. Unfortunately, their work does not present quantitative results. Stefan et al. [23] used a nearest neighbor technique to classify dynamic gestures based on their feature vectors, with data acquired from video streams. A recognition rate of 96.3% was achieved, based on gestures representing digits from 0 to 9. Ziaie et al. [24] used a weighted k-nearest neighbor combined with a naïve Bayes approach to recognize three different static hand gestures. With tests executed in the domain of the JAST human-robot dialog system, the approach obtained more than 93.0% of correct classifications. Tarrataca et al. [25] used a k-nearest neighbor and a Hidden Markov Model to build a gesture recognition system for smartphones, classifying static gestures and sequences of static gestures. Using the phone's camera for image acquisition, the system was trained to recognize three different gestures, achieving an average recognition rate of 83.3%. Kjellström et al. [11] controlled the grasp action of a robot arm using the recognition of a hand shape, made using the Locality Sensitive Hashing (LSH) approach—an approximation of the k-nearest neighbor technique. The recognition performance achieved was comparable to the human performance, with 75.0% correct classifications. Wang and Popović [26] developed a system to facilitate 3D articulated user input, using a colored glove and a single camera. The proposed method, based on nearest neighbor, compares the color pattern of the glove with the color pattern of different poses previously recorded and stored in a database. The work does not present quantitative results, but describes proof-of-concept applications where the method was applied.

3.4 Criticism

The research works mentioned above share some characteristics: Each work evaluates a single gesture recognition technique through the use of one or more algorithms, using a particular dataset for training and testing, and measure the accuracy of each technique in terms of its recognition rate. The methods and the dataset were chosen in a more or less *ad hoc* manner, without offering well-founded justification for either and also without presenting an objective comparison of the obtained results against other techniques.

A more objective approach, based upon public datasets and quantitative and comparable results is necessary. In the present work, three different techniques for static gesture recognition are evaluated. This is achieved using the same dataset for training and testing, and measuring the precision and the performance for each method. Furthermore, the work aims to identify the best method from the set of tested approaches, which could be used to build highly responsive and intuitive interfaces. Our datasets are made publicly available, in order to allow the research community to test other approaches against the results we present here.

4 Experimental environment

The evaluation of precision and performance of gesture recognition techniques demands a hardware and software setup, as well as a vocabulary of static gestures to be used as common input for the different algorithms. This section describes the experimental environment for the current work, through the specification of hardware devices and software components.

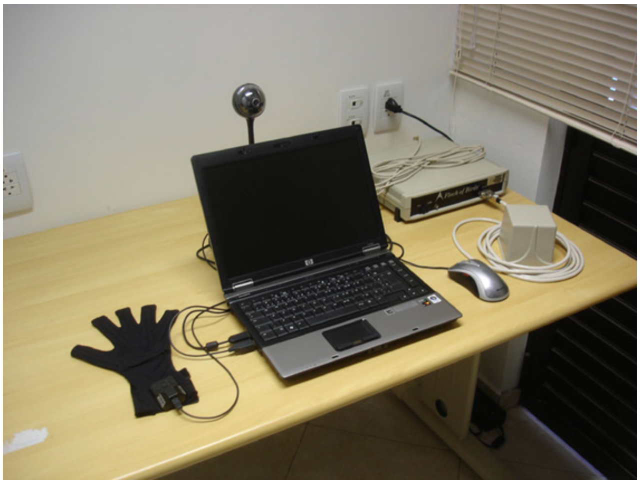


Fig. 1 Hardware setup—data glove, laptop/webcam and motion tracker (not used for the current experiments)

Table 1 5DT Data Glove 5 Ultra specifications [27]

Item	Specification
Material	Black Stretch Lycra
Sensor Resolution	12-bit A/D (typical range 10 bits)
Flexure Sensors	Fiber Optics Based 5 Sensors in total 1 Sensor per finger, measures average of knuckle and first joint
Computer Interface	Full-speed USB 1.1 RS-232 (via optional serial interface kit)
Power Supply	Via USB Interface
Sampling Rate	Minimum 75 Hz

4.1 Hardware setup

The experimental environment for this work simulates an instrumented interface, which uses data acquired from a data glove. This hardware setup composes the first step of the gesture recognition pipeline, generating data for classification algorithms. The hardware used for data acquisition is shown in Fig. 1.

Static gestures are represented by a set of values, which were acquired using an instrumented glove from Fifth Dimension Technologies.[2] The glove used—model 5DT Data Glove 5 Ultra,[3] for the right hand—offers a set of five fiber optic sensors, one for each finger, disposed on the top of the glove. Each sensor measures the flexure of an individual finger. A more detailed specification is shown in Table 1.

Table 2 Example of static gesture data

Sensor description		Sensor data	
Position	Driver Sensor Index	Raw	Scaled
Thumb	0	3807	0.149635
	1	3807	0.149635
	2	0	0
Index finger	3	3086	0.964564
	4	3086	0.964564
	5	0	0
Middle finger	6	3447	0.729363
	7	3447	0.729363
	8	0	0
Ring finger	9	2702	0.69604
	10	2702	0.69604
	11	0	0
Little finger	12	2148	0.145
	13	2148	0.145
	14	0	0
	15	0	0
Pitch angle	16	2048	0
Roll angle	17	2048	0

The access to the glove's hardware is made through an Application Programming Interface (API), which provides a set of functions to initialize and shutdown the glove, calibrate sensors, and retrieve values from sensors (individually or collectively). For each sensor, it is possible to retrieve two types of data: *Raw* (basic) values and *scaled* (normalized) values. Raw data range from 0 to 4,095, and correspond to values acquired directly from sensors, without any modification; scaled data correspond to values acquired from sensors, normalized to fit the interval [0, 1]. The higher the value of a sensor, the higher is the flexure degree of the corresponding finger. Table 2 shows an example of a static gesture (depicted in Fig. 2) through the acquired sensor values.

The manufacturer of the data glove used in this work provides a generic communication driver, which can be used with different glove models. This driver encapsulates the acquired data in a data frame, whose structure allows the storage of data from a number of sensors, as well as data from sensors of different types. For the 5DT Data Glove 5 Ultra model, only the sensors identified by indexes 0, 3, 6, 9 e 12 (highlighted in gray on Table 2) are considered for training and evaluation; the indexes 2, 5, 8, 11, 14, 15, 16, 17 are discarded because there are no data available for the glove model, and the indexes 1, 4, 7, 10, 13 are ignored (since their values are duplicates of values acquired from sensors identified by indexes 0, 3, 6, 9 e 12).

[2] http://www.5dt.com/.

[3] http://www.5dt.com/products/pdataglove5u.html.

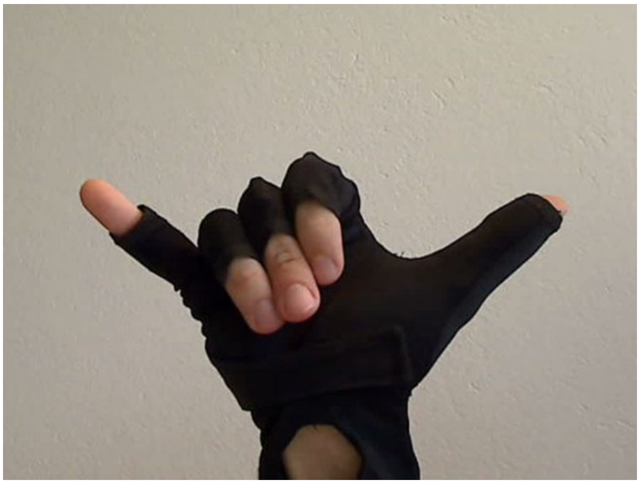


Fig. 2 Example of static gesture

The experiments were performed on a laptop with an AMD Turion™X2 Dual Core processor (2.0 GHz), 2 Gb RAM, running Microsoft Windows®XP Professional SP3.

4.2 Software for data acquisition

The data for the training and evaluation steps were acquired through an application specially developed for this purpose. In this application, the screen is divided in two parts: On the left, a media player is responsible for exhibiting short videos of static gestures (according to an order defined by a playlist), at a 24 frames per second (fps) rate; on the right, a webcam captures the user actions, providing an immediate visual feedback. All the controls needed to configure and use the application are located in a menu bar (at the top of screen) and on a button bar (at the bottom of screen), as can be seen in Fig. 3.

After the configuration of the communication port for the data glove, the image adjusting for the webcam and the stipulation of the number of iterations to be performed by each user, the data acquisition process starts with the calibration of the glove. The calibration procedure adjusts the glove's sensors to the shape of users' hands, allowing the retrieval of fine-tuned data. To calibrate the glove, each user follows a video with a sequence of hand poses. At the end of the calibration process, the glove driver stores minimum and maximum flexure values for each user's finger. These values are used to compute the final data.

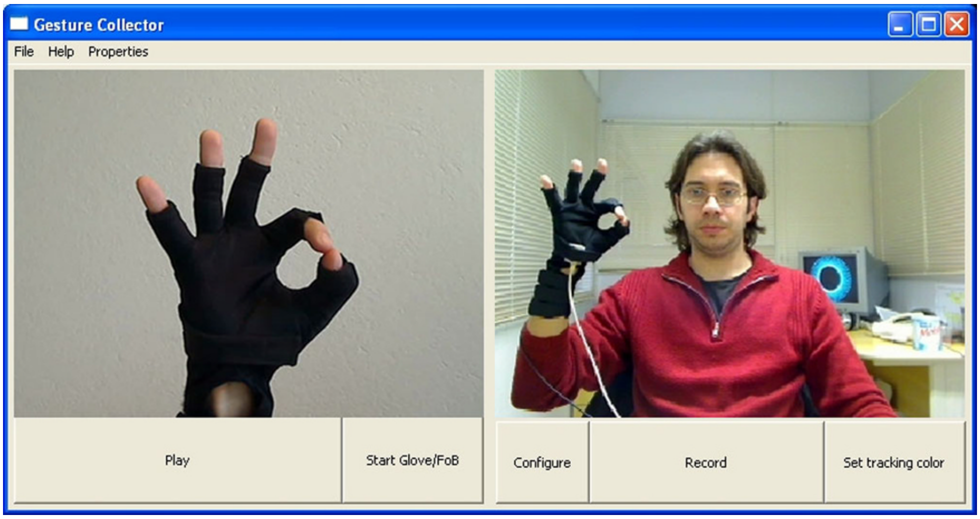
The application adopts the same sequence of steps to acquire data for static gestures. In the beginning of each iteration, the playlist is shuffled, thus guaranteeing that a given posture is not stressed by sequential execution. The user watches, then, a short video of a static gesture to be executed. At this point, a click on the "Record" button captures the raw and scaled data from the glove. The collected data are stored in text files, identified by the user number, type of data (raw/scaled), posture number and iteration number. Also, screenshots are taken from postures. These steps are repeated until the end of the playlist execution.

In order to allow the users to receive a visual feedback, the webcam is turned on at the application startup, and stays online while the application is executed. This strategy helps the users on execution of gestures, offering visual comparison capabilities between the gesture model and the gesture that is been executed.

4.3 Gesture recognition techniques

The choice for the gesture recognition techniques evaluated in this work was made considering the relevance of related work listed in Sect. 3, as well as the recognition rates achieved by experiments. For the static gesture recognition, three techniques were chosen: A Nearest Neighbor algorithm based on K-D Trees, implemented using the Approxi-

Fig. 3 Data acquisition software during execution



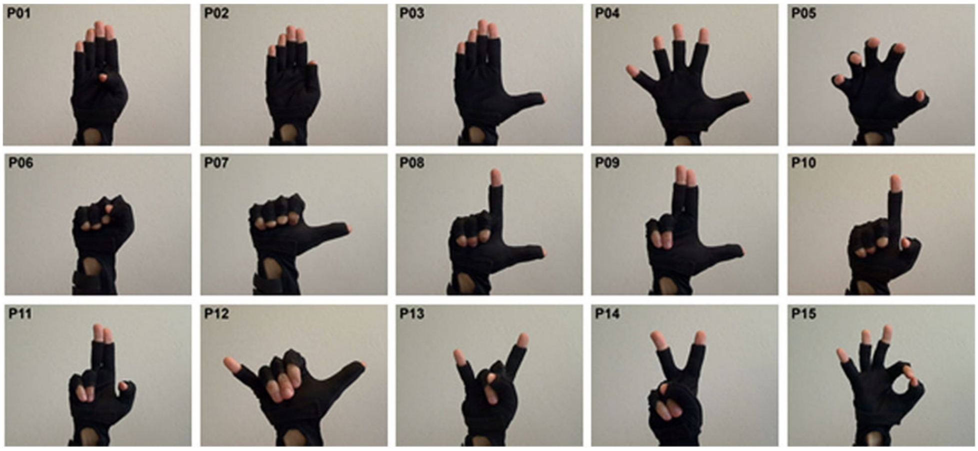


Fig. 4 Vocabulary of static gestures

mate Nearest Neighbor Searching (ANN) library[4]; a Neural Network, implemented using the Fast Artificial Neural Network Library (FANN)[5]; and a Support Vector Machine, implemented using the LIBSVM[6] library. All techniques were adapted to use the text files generated by the data acquisition application as input for training and evaluation, considering raw and scaled data types. The configuration parameters used were defined empirically through a number of tests executed with subsets of the original data. The K-D Tree used for nearest neighbor classification was built directly from data files, without specific configuration parameters. The neural network was configured with five layers: An input layer with five neurons, three hidden layers with 18 neurons each and an output layer with 15 neurons. The training was performed with a number of epochs varying from 100 to 2,500, using a learning rate of 0.5. The SVM was configured as a classification SVM type 1 (C-SVM), using two different kernels: Radial Basis Function (RBF) and SIGMOID. All libraries and software were built using Microsoft Visual C++ 2005.

5 Results

Through the use of the hardware and software components described in Sect. 4, data from static gestures were collected and evaluated using the gesture recognition techniques previously chosen. This section presents the obtained results, focusing on the precision and performance of each technique.

The data for the experiments were collected from 33 users, using the application described in Sect. 4. A vocabulary of 15 static gestures (shown in Fig. 4) was defined and presented to the users, guiding their behavior. Each user performed a set of five enactments of the whole gesture set. The gesture order of each repetition was random and determined by the playlist mentioned before, the first iteration being supervised by a specialist (to solve questions about the procedure). A total of 2,475 static gestures were collected, composing the experimental dataset.

The training and evaluation steps were performed using the repeated random sub-sampling validation. For each iteration (from a total of 10), one third of the dataset was selected as the training set, and the remainder of the original dataset was used as the validation set; in concrete numbers, 825 instances were used as training set and 1,650 instances were used as validation set.

Aiming to identify the importance of the partitioning method adopted to split the original dataset for training and validation, and its relation to the obtained results, two different strategies were used. The first partitioning strategy randomly selected as the training set a gesture set composed of one third of the entire dataset, selecting random gestures uniformly distributed between all gesture classes. The second partitioning strategy randomly selected one third of the users involved, using all the gesture instances recorded by these users as the training set. In both cases, the remainder of the original dataset was used as the validation set. From now on, these strategies will be referred to, respectively, as PS-1 and PS-2.

The results obtained in this manner were organized as confusion matrices, like the ones presented in Table 3 and Table 4. These matrices show the results obtained for the nearest neighbor recognition technique, using the PS-1 and PS-2 partitioning strategies, respectively. Predicted gestures are organized in rows, and evaluated gestures are organized in columns. The cells on the main diagonal (highlighted in gray) display the true positive occurrences, while the other cells display the false positive evaluations. Each cell stores two values: The first one is the result of the evaluation using the glove's raw data, and the second one is the result of

[4] http://www.cs.umd.edu/~mount/ANN/.

[5] http://leenissen.dk/fann/.

[6] http://www.csie.ntu.edu.tw/~cjlin/libsvm/.

Table 3 Nearest Neighbor trained/evaluated employing the PS-1 strategy

		Actual (raw/scaled)															Precision (%) (raw/scaled)
		P01	P02	P03	P04	P05	P06	P07	P08	P09	P10	P11	P12	P13	P14	P15	
Predicted (raw/scaled)	P01	770/828	123/132	49/30	58/33	41/18	0/0	1/0	0/0	0/0	0/0	0/0	0/0	0/0	0/0	58/59	70.00/75.27
	P02	145/142	446/553	273/247	204/135	18/11	0/0	0/2	0/0	1/2	0/0	0/0	0/0	0/0	1/0	12/8	40.55/50.27
	P03	95/39	263/222	469/518	248/302	14/5	0/0	0/0	0/0	1/6	0/0	0/0	0/0	0/0	1/0	9/8	42.64/47.09
	P04	58/47	206/140	247/315	526/548	24/9	0/0	0/0	4/4	3/6	4/4	0/0	0/0	0/0	0/0	28/27	47.82/49.82
	P05	40/27	16/28	5/7	21/7	873/932	17/10	5/0	6/0	2/0	2/8	4/0	0/5	13/0	0/0	96/76	79.36/84.73
	P06	1/0	0/0	0/0	0/0	14/21	905/1053	57/25	0/0	4/0	42/1	14/0	30/0	8/0	25/0	0/0	82.27/95.73
	P07	0/1	0/4	0/0	0/2	14/2	75/40	900/1035	34/0	23/0	23/0	2/0	24/16	4/0	1/0	0/0	81.82/94.09
	P08	0/0	0/0	0/0	1/0	3/0	6/0	52/1	861/973	20/16	130/98	6/0	10/0	10/12	1/0	0/0	78.27/88.45
	P09	0/0	0/0	0/0	1/0	4/1	2/0	15/2	10/7	817/958	4/0	128/77	1/0	0/0	118/55	0/0	74.27/87.09
	P10	0/0	0/0	0/0	0/1	1/13	64/1	10/0	120/90	2/2	836/964	9/3	0/0	52/19	6/7	0/0	76.00/87.64
	P11	0/0	1/0	0/0	0/0	4/1	13/0	0/0	0/0	81/51	3/0	678/720	0/0	0/0	320/328	0/0	61.64/65.45
	P12	0/0	0/0	0/0	0/0	8/0	14/10	35/25	0/0	0/0	8/0	0/0	998/1064	37/1	0/0	0/0	90.73/96.73
	P13	0/0	0/0	0/2	0/3	12/2	2/0	2/1	5/14	0/0	59/17	0/0	36/3	984/1058	0/0	0/0	89.45/96.18
	P14	0/0	0/0	5/0	2/0	1/5	11/0	0/0	0/0	70/34	16/2	278/336	0/0	0/0	717/723	0/0	65.18/65.73
	P15	77/58	24/7	10/2	56/11	80/85	0/0	2/0	0/0	0/2	0/0	0/0	0/0	0/0	0/0	851/935	77.36/85.00
Median precision (%) (raw/scaled)																	70.49/77.95

Table 4 Nearest Neighbor trained/evaluated employing the PS-2 strategy

		Actual (raw/scaled)															Precision (%) (raw/scaled)
		P01	P02	P03	P04	P05	P06	P07	P08	P09	P10	P11	P12	P13	P14	P15	
Predicted (raw/scaled)	P01	405/784	152/171	87/28	104/36	123/32	1/0	1/1	0/0	0/0	0/0	2/0	1/0	6/0	1/0	217/48	36.82/71.27
	P02	154/169	228/454	215/240	292/198	94/15	0/0	2/4	0/0	0/4	0/0	1/0	0/0	2/6	1/0	111/10	20.73/41.27
	P03	73/69	185/234	301/428	330/345	89/6	1/0	4/0	0/0	0/9	0/0	0/2	1/0	3/3	1/0	112/4	27.36/38.91
	P04	40/61	186/166	241/340	393/472	104/16	0/0	1/0	6/5	2/15	0/0	0/3	0/0	0/0	0/0	127/22	35.73/42.91
	P05	79/68	28/32	26/7	38/15	537/812	22/19	31/3	14/0	2/1	11/40	2/1	18/9	70/0	6/0	216/93	48.82/73.82
	P06	0/0	0/0	0/0	0/0	57/28	597/996	81/43	9/0	17/0	118/3	38/0	117/30	16/0	50/0	0/0	54.27/90.55
	P07	1/0	1/3	1/0	0/3	72/9	141/37	580/1022	105/0	45/0	25/1	0/0	117/25	2/0	10/0	0/0	52.73/92.91
	P08	0/0	0/0	0/0	3/0	31/1	30/0	93/9	654/958	36/11	123/110	2/0	7/0	107/11	14/0	0/0	59.45/87.09
	P09	3/0	0/1	0/0	0/4	17/0	22/0	99/0	39/14	592/986	24/0	159/43	0/0	2/0	143/52	0/0	53.82/89.64
	P10	0/0	0/0	0/0	1/0	10/12	118/0	11/18	124/104	3/3	641/918	14/0	3/0	140/37	35/8	0/0	58.27/83.45
	P11	0/0	0/0	0/0	0/0	19/3	99/1	11/0	12/0	76/105	34/2	527/580	1/0	4/0	317/409	0/0	47.91/52.73
	P12	0/0	0/0	0/0	1/0	16/29	57/9	52/47	23/0	1/0	11/0	0/0	802/1015	136/0	1/0	0/0	72.91/92.27
	P13	2/1	0/3	0/7	1/6	16/5	18/0	3/2	18/47	0/0	122/43	0/0	107/4	813/982	0/0	0/0	73.91/89.27
	P14	0/0	2/0	1/0	2/0	3/3	75/0	6/0	11/0	81/74	54/3	372/424	2/0	7/0	482/596	2/0	43.82/54.18
	P15	123/124	100/10	35/3	60/27	254/134	0/0	4/0	0/0	0/5	0/0	0/0	0/0	0/0	1/0	523/797	47.55/72.45
Median precision (%) (raw/scaled)																	48.94/71.52

the evaluation using the glove's scaled data. The precision of the nearest neighbor technique can be seen in the last column, individually for each gesture and collectively for all the vocabulary. The precision for each gesture is calculated dividing the true positive occurrences (stored in the main diagonal) by the sum of true positive occurrences and false positive occurrences (resulting in 1,100 evaluation instances per gesture); the overall precision, in turn, is calculated by the mean of individual precisions of all gestures.

The results for the technique based on the neural network are shown in Table 5 and Table 6. Compared to the nearest neighbor technique, the neural network presented a slightly poorer precision classifying the scaled data–a difference of 6.4% and 0.55% for PS-1 and PS-2 strategies, respectively. But the main difference between the two techniques was the precision achieved classifying raw data. In the best case, the nearest neighbor technique achieved a precision of 70.49% against 17.98% achieved by the neural network. A possible explanation for this behavior is the range for raw and scaled data. As stated in Sect. 4, the glove's raw data range from 0 to 4,095, while the glove's scaled data range from 0 to 1. According to Sarle [28], large attribute values might cause numerical problems, which can interfere on network evaluation; this occurs because attributes with greater numerical ranges can dominate attributes with smaller numerical ranges, distorting the acquired results. The obtained results, related to their respective training epochs, can be seen on Fig. 5 and Fig. 6.

Table 7 and Table 8 show the results obtained using an SVM. This technique presented the best and the worst results for the classification of scaled and raw data, respectively, in numbers, 79.13% and 6.81%. The behavior of the SVM is quite similar to the neural network, since the same numerical restrictions can be applied to it. A comparison between the results obtained with different types of kernel can be seen in Fig. 7.

Through the analysis of the confusion matrices, it is possible to identify some characteristics common to all methods:

- Evaluation of scaled data—the evaluation of scaled data presents better results than the evaluation of raw data. Two factors contribute to this result: The calibration process helps by adjusting the glove's sensors to the user's hand, allowing a fine-tuned gesture acquisition; and the difference regarding the ranges for raw and scaled data, which influences the results of the neural network and the SVM techniques;
- Influence of the partitioning strategy—the PS-1 strategy presents better results than the PS-2 strategy. This is because the PS-1 strategy does not limit the training data to gesture particularities of a specific set of users, allowing a more heterogeneous set of gesture instances. The PS-2 strategy, in turn, generates a more homogeneous training

Table 5 Neural Network trained/evaluated employing the PS-1 strategy

		Actual (raw/scaled)															Precision (%) (raw/scaled)
		P01	P02	P03	P04	P05	P06	P07	P08	P09	P10	P11	P12	P13	P14	P15	
Predicted (raw/scaled)	P01	239/891	110/34	122/29	184/12	0/22	27/0	31/0	75/0	0/0	0/0	41/8	85/1	3/6	4/1	179/96	21.73/81.00
	P02	229/327	110/174	123/318	178/156	1/21	27/0	40/0	74/0	0/36	1/0	47/3	87/1	4/11	3/0	176/53	10.00/15.82
	P03	235/108	110/61	120/561	181/270	1/13	27/0	40/3	74/0	0/50	1/0	46/7	87/2	5/4	3/2	170/19	10.91/51.00
	P04	217/59	109/35	124/471	183/313	0/48	27/0	43/6	71/8	0/43	1/0	40/0	86/10	4/2	3/0	192/105	16.64/28.45
	P05	174/36	100/7	114/5	111/16	0/694	28/114	98/12	81/0	0/1	10/3	32/0	144/45	20/9	6/0	182/158	0.00/63.09
	P06	71/0	45/0	110/0	13/0	1/7	43/1074	167/4	122/0	30/0	133/0	71/0	163/10	3/5	100/0	28/0	3.91/97.64
	P07	31/0	38/0	107/0	13/0	0/1	38/79	248/998	117/0	29/0	108/0	55/0	190/15	6/0	76/0	44/7	22.55/90.73
	P08	45/0	14/0	73/0	3/0	0/0	44/1	138/13	255/975	31/13	234/96	35/0	130/0	28/2	65/0	5/0	23.18/88.64
	P09	13/1	27/0	103/0	8/0	0/0	47/8	79/5	96/13	140/975	77/1	199/71	84/0	3/0	224/26	0/0	12.73/88.64
	P10	78/0	22/0	82/0	6/0	1/3	42/18	123/23	218/59	21/3	239/946	37/2	128/1	20/45	73/0	10/0	21.73/86.00
	P11	18/1	34/0	107/0	17/0	0/6	46/8	68/0	96/0	118/61	71/25	215/701	96/0	3/0	208/298	3/0	19.55/63.73
	P12	45/0	12/0	104/0	1/0	1/8	26/10	123/27	113/0	0/0	60/0	4/0	502/1046	49/9	12/0	48/0	45.64/95.09
	P13	72/0	15/0	69/0	1/0	0/0	26/0	84/0	164/33	0/0	132/33	14/0	265/8	183/1026	17/0	58/0	16.64/93.27
	P14	23/7	30/0	101/0	14/0	0/3	47/4	62/1	106/3	133/106	80/58	196/451	90/0	8/6	208/461	2/0	18.91/41.91
	P15	184/63	110/0	114/2	200/6	0/54	27/3	38/1	34/0	0/0	0/1	13/0	99/0	0/0	0/0	281/970	25.55/88.18
Median precision (%) (raw/scaled)																	17.98/71.55

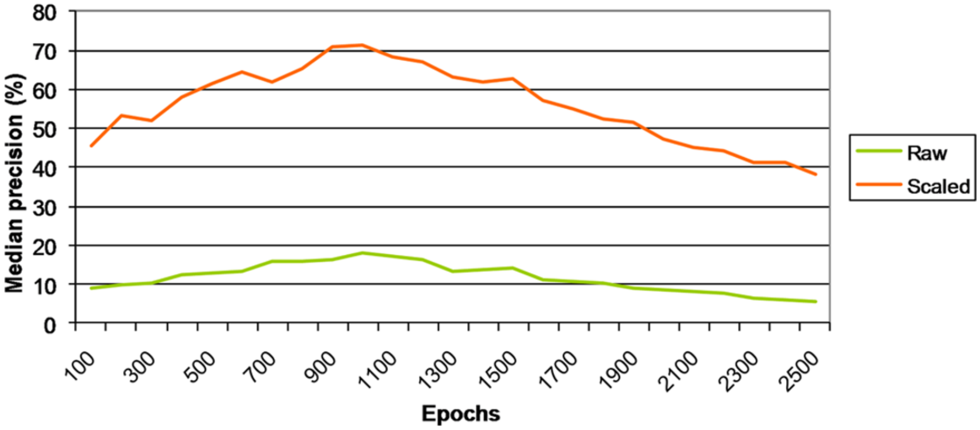


Fig. 5 Median precision × training epochs (PS-1 strategy)

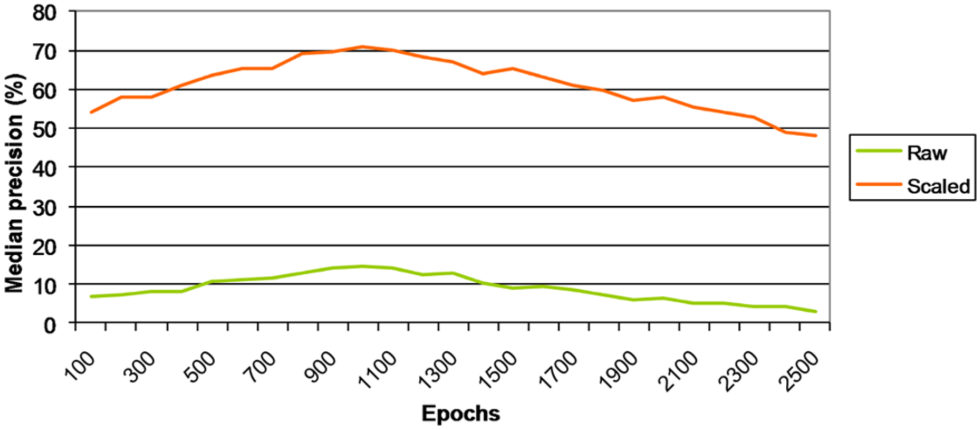


Fig. 6 Median precision × training epochs (PS-2 strategy)

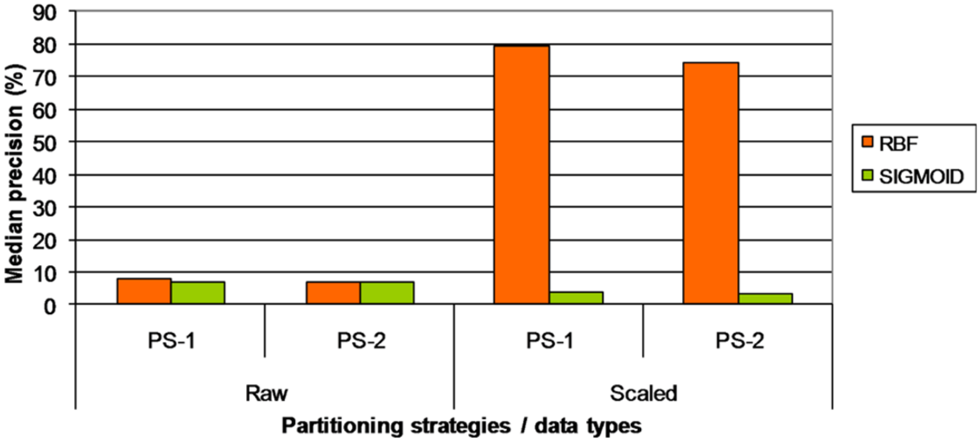


Fig. 7 Median precision × kernel option (SVM technique)

set (through the selection of a number of users), which limits the generalization of the recognition techniques;

- Specific gestures with low recognition rate—specific gestures in the vocabulary were shown to yield a low recognition rate. This can be explained by the similarities between these gestures. The gestures labeled P02, P03, P04, P11, and P14 have the worst recognition rates. In Fig. 4, it is possible to observe that gestures P02, P03, and P04 are quite similar (in terms of finger flexures), as well as the gestures P11 and P14. Similar finger flexure configurations generate similar patterns, which can confuse the classifiers, hence generating false positives. Tables 9, 10, and 11 show the results obtained by grouping these gestures together, using the PS-1 partitioning strategy. The grouping of similar gestures improves the individual recognition rate for these gestures and, consequently, makes for a better overall precision. A gain of 11.8%, 14.34%, and 11.83% were obtained, respectively, for the nearest neighbor, neural network, and SVM methods considering the scaled data. Figure 8 resumes the acquired

Table 6 Neural Network trained/evaluated employing the PS-2 strategy

		Actual (raw/scaled)															Precision (%) (raw/scaled)
		P01	P02	P03	P04	P05	P06	P07	P08	P09	P10	P11	P12	P13	P14	P15	
Predicted (raw/scaled)	P01	93/910	0/29	3/28	196/21	0/33	39/2	1/0	0/0	68/1	22/4	116/4	46/0	218/0	149/1	149/67	8.45/82.73
	P02	82/322	0/189	0/307	197/146	0/54	36/0	6/0	0/0	68/32	15/7	123/1	47/0	225/6	149/0	152/36	0.00/17.18
	P03	81/115	0/62	1/592	193/222	0/26	36/0	4/0	2/2	70/50	11/8	123/7	44/0	226/7	151/2	158/7	0.09/53.82
	P04	76/55	1/34	0/520	208/291	0/75	31/0	10/3	1/5	68/41	8/10	104/0	42/2	221/2	141/0	189/62	18.91/26.45
	P05	29/78	0/1	0/11	162/15	0/667	75/101	23/4	0/0	65/0	20/17	119/1	61/35	242/21	155/0	149/149	0.00/60.64
	P06	0/0	0/0	0/0	6/0	0/5	158/1080	13/2	21/0	115/0	33/9	223/0	48/3	214/1	264/0	5/0	14.36/98.18
	P07	0/0	0/0	0/0	9/1	0/3	157/94	49/982	39/1	109/0	22/0	177/0	49/16	218/0	252/0	19/3	4.45/89.27
	P08	0/0	1/0	0/0	19/0	0/0	138/1	2/8	91/968	118/15	50/98	167/1	27/0	267/8	216/1	4/0	8.27/88.00
	P09	0/1	0/0	0/3	0/0	0/0	103/8	0/1	15/9	180/967	33/2	255/64	26/0	168/0	315/45	5/0	16.36/87.91
	P10	1/0	1/0	0/0	15/0	0/4	147/15	0/10	59/56	109/4	56/981	176/2	17/0	276/25	242/3	1/0	5.09/89.18
	P11	3/5	0/0	0/0	0/0	0/2	107/18	0/0	5/0	188/60	34/25	258/653	13/0	158/0	329/337	5/0	23.45/59.36
	P12	0/0	0/0	0/0	60/0	0/18	187/7	12/37	4/0	29/0	5/3	53/0	172/1031	394/4	107/0	77/0	15.64/93.73
	P13	12/0	0/0	0/0	88/0	0/3	158/2	2/0	12/27	39/0	27/36	60/0	61/5	492/1027	136/0	13/0	44.73/93.36
	P14	1/17	0/0	0/3	0/0	0/1	109/5	1/0	4/1	162/116	28/45	252/427	22/0	196/2	324/483	1/0	29.45/43.91
	P15	39/89	0/0	3/4	200/5	0/95	12/4	13/6	0/0	41/0	15/0	107/4	88/4	119/0	123/0	340/889	30.91/80.82
Median precision (%) (raw/scaled)																	14.68/70.97

Table 7 Support Vector Machine trained/evaluated employing the PS-1 strategy

		Actual (raw/scaled)															Precision (%) (raw/scaled)
		P01	P02	P03	P04	P05	P06	P07	P08	P09	P10	P11	P12	P13	P14	P15	
Predicted (raw/scaled)	P01	11/884	0/114	0/10	4/28	0/23	0/0	0/0	0/0	0/0	0/0	0/0	0/0	0/0	0/0	1085/41	1.00/80.36
	P02	0/89	0/605	5/226	0/156	0/20	0/0	0/0	0/0	0/0	0/0	0/0	0/0	0/0	0/0	1095/4	0.00/55.00
	P03	0/42	4/179	6/549	0/326	0/4	0/0	0/0	0/0	0/0	0/0	0/0	0/0	0/0	0/0	1090/0	0.55/49.91
	P04	2/43	0/124	0/363	3/520	0/16	0/0	0/0	0/4	0/2	0/1	0/1	0/0	0/2	0/0	1095/24	0.27/47.27
	P05	0/26	0/27	0/8	0/6	12/934	0/8	0/0	0/0	0/0	0/9	0/0	0/5	0/2	0/0	1088/75	1.09/84.91
	P06	0/0	0/0	0/0	0/0	0/21	43/1053	2/18	0/0	0/0	0/1	0/0	0/7	0/0	0/0	1055/0	3.91/95.73
	P07	0/0	0/0	0/0	0/6	0/0	2/42	20/1028	0/0	0/0	0/3	0/0	0/21	0/0	0/0	1078/0	1.82/93.45
	P08	0/0	0/0	0/0	0/0	0/0	0/0	0/5	19/1008	0/6	0/71	0/0	0/0	0/10	0/0	1081/0	1.73/91.64
	P09	0/0	0/0	0/0	0/0	0/1	0/0	0/1	0/6	3/979	0/0	0/64	0/0	0/0	0/49	1097/0	0.27/89.00
	P10	0/0	0/0	0/0	0/0	0/5	0/0	0/6	0/89	0/0	13/969	0/0	0/0	1/29	0/2	1086/0	1.18/88.09
	P11	0/0	0/0	0/0	0/0	0/4	0/0	0/0	0/0	0/73	0/0	10/786	0/0	0/0	2/237	1088/0	0.91/71.45
	P12	0/0	0/0	0/0	0/0	0/0	0/14	0/20	0/0	0/0	0/0	0/0	14/1066	0/0	0/0	1086/0	1.27/96.91
	P13	0/0	0/0	0/3	0/2	0/0	0/1	0/0	0/7	0/0	4/18	0/0	0/4	21/1065	0/0	1075/0	1.91/96.82
	P14	0/0	0/0	0/0	0/0	0/6	0/0	0/0	0/2	5/27	0/3	2/414	0/0	0/0	6/648	1087/0	0.55/58.91
	P15	0/84	0/3	0/0	0/12	0/39	0/0	0/0	0/0	0/0	0/0	0/0	0/0	0/0	0/0	1100/962	100.00/87.45
Median precision (%) (raw/scaled)																	7.76/79.13

Table 8 Support Vector Machine trained/evaluated employing the PS-2 strategy

		Actual (raw/scaled)															Precision (%) (raw/scaled)
		P01	P02	P03	P04	P05	P06	P07	P08	P09	P10	P11	P12	P13	P14	P15	
Predicted (raw/scaled)	P01	0/857	0/92	0/22	1/42	0/34	0/0	0/0	0/0	0/0	0/0	0/0	0/0	0/0	0/0	1099/53	0.00/77.91
	P02	0/145	0/528	2/218	0/179	0/25	0/0	0/0	0/0	0/1	0/0	0/0	0/0	0/0	0/0	1098/4	0.00/48.00
	P03	0/67	3/208	5/493	0/320	0/7	0/0	0/0	0/0	0/3	0/0	0/0	0/0	0/1	0/0	1092/1	0.45/44.82
	P04	4/63	0/163	0/383	0/431	0/21	0/0	0/0	0/7	0/12	0/0	0/0	0/0	0/0	0/0	1096/20	0.00/39.18
	P05	0/68	0/30	0/5	0/27	0/835	0/21	0/0	0/1	0/3	0/31	0/1	0/8	0/4	0/1	1100/65	0.00/75.91
	P06	0/0	0/0	0/0	0/0	0/32	5/1023	3/13	0/0	0/0	0/6	0/0	0/24	0/2	0/0	1092/0	0.45/93.00
	P07	0/0	0/1	0/0	0/6	0/17	2/69	0/963	0/2	0/2	0/4	0/0	0/36	0/0	0/0	1098/0	0.00/87.55
	P08	0/0	0/0	0/0	0/0	0/0	0/0	0/7	0/967	0/14	0/97	0/0	0/0	0/15	0/0	1100/0	0.00/87.91
	P09	0/0	0/0	0/0	0/2	0/1	0/0	0/1	0/6	0/953	0/0	0/80	0/0	0/0	2/57	1098/0	0.00/86.64
	P10	0/0	0/0	0/0	0/0	0/4	0/0	0/11	0/78	0/0	5/946	0/2	0/0	0/46	0/13	1095/0	0.45/86.00
	P11	0/0	0/0	0/0	0/0	0/5	0/0	0/0	0/0	0/55	0/1	0/767	0/0	0/0	0/272	1100/0	0.00/69.73
	P12	0/0	0/0	0/0	0/0	0/16	0/9	0/24	0/0	0/0	0/0	0/0	0/1045	0/6	0/0	1100/0	0.00/95.00
	P13	0/2	0/0	0/4	0/6	0/6	0/0	0/2	0/29	0/0	5/50	0/0	0/4	8/997	0/0	1087/0	0.73/90.64
	P14	0/0	0/0	0/0	0/0	0/3	0/0	0/0	0/0	3/46	0/3	0/468	0/0	0/0	0/580	1097/0	0.00/52.73
	P15	0/124	0/10	0/1	0/30	0/102	0/2	0/0	0/0	0/4	0/0	0/0	0/0	0/0	0/0	1100/827	100.00/75.18
Median precision (%) (raw/scaled)																	6.81/74.01

Table 9 Precision of recognition techniques with similar gestures grouped—Nearest Neighbor

Nearest Neighbor		Actual (raw/scaled)												Precision (%) (raw/scaled)
		P01	P02/P03/P04	P05	P06	P07	P08	P09	P10	P11/P14	P12	P13	P15	
Predicted (raw/scaled)	P01	770/828	230/195	41/18	0/0	1/0	0/0	0/0	0/0	0/0	0/0	0/0	58/59	70.00/75.27
	P02/P03/P04	298/228	2882/2980	56/25	0/0	0/2	4/4	5/14	4/4	2/0	0/0	0/0	49/43	87.33/90.30
	P05	40/27	42/42	873/932	17/10	5/0	6/0	2/0	2/8	4/0	0/5	13/0	96/76	79.36/84.73
	P06	1/0	0/0	14/21	905/1053	57/25	0/0	4/0	42/1	39/0	30/0	8/0	0/0	82.27/95.73
	P07	0/1	0/6	14/2	75/40	900/1035	34/0	23/0	23/0	3/0	24/16	4/0	0/0	81.82/94.09
	P08	0/0	1/0	3/0	6/0	52/1	861/973	20/16	130/98	7/0	10/0	10/12	0/0	78.27/88.45
	P09	0/0	1/0	4/1	2/0	15/2	10/7	817/958	4/0	246/132	1/0	0/0	0/0	74.27/87.09
	P10	0/0	0/1	1/13	64/1	10/0	120/90	2/2	836/964	15/10	0/0	52/19	0/0	76.00/87.64
	P11/P14	0/0	8/0	5/6	24/0	0/0	0/0	151/85	19/2	1993/2107	0/0	0/0	0/0	90.59/95.77
	P12	0/0	0/0	8/0	14/10	35/25	0/0	0/0	8/0	0/0	998/1064	37/1	0/0	90.73/96.73
	P13	0/0	0/5	12/2	2/0	2/1	5/14	0/0	59/17	0/0	36/3	984/1058	0/0	89.45/96.18
	P15	77/58	90/20	80/85	0/0	2/0	0/0	0/2	0/0	0/0	0/0	0/0	851/935	77.36/85.00
Median precision (%) (raw/scaled)														81.45/89.75

results in terms of partitioning strategies, data types, vocabulary differences, and median precision.

The performance evaluation of the recognition techniques was executed for both the training and the recognition step. The time spent for the training step is measured through the calculation of the Mean Training Time (MTT), which comprehends the process of loading patterns from text files to memory, the construction of a data structure to store data according to the selected classifier, and the execution of a training algorithm. For the nearest neighbor classifier, specifically, there is no training algorithm (since K-D Tree organizes itself during the construction step). The time spent for the evaluation step, in turn, is measured through the calculation of the Mean Evaluation Time (MET), which comprehends the process of loading a gesture data to memory and its classification. Table 12 and Table 13 summarize the MTT and MET values obtained for each classifier, including the Standard Deviation (SD). All times were collected during the training and evaluation steps performed to obtain the precision values presented in this section, and are expressed in seconds (s), milliseconds (ms) and microseconds (μs).

The results in Table 12 and Table 13 show that, despite the type of data (raw/scaled), training and evaluation times are balanced. There is, however, one exception: The SVM technique presents a low training and evaluation time when scaled data are used. In this case, a maximum reduction of 57.84% for training time and 42.27% for evaluation time were achieved. Figure 9 shows a comparison between the evaluation times, considering the partitioning strategies and the data types.

6 Discussion

The results presented in Sect. 5 summarize the achievements in terms of precision and performance for the analyzed gesture recognition methods. This section discusses these results, highlighting important aspects that will subsidize the final conclusions.

The data acquisition process used to acquire the users' behavior has shown to be robust enough, which can be confirmed by the stable results obtained for a number of gestures. The dynamic of videos showing static gestures from different angles, as well as the webcam providing immediate visual feedback, helped users to correct posture problems, which reflect on the quality of acquired data. The presence of a specialist on the first iteration, answering questions and guiding users through the interface, helped the understanding of the entire process by the users.

The partitioning strategies used in this work, PS-1 and PS-2, have a direct impact on the obtained results. Training sets composed by instances from a number of randomly chosen users improve the recognition rate, since the heterogeneity of instances implies a better generalization and avoids the systematic error which is introduced by the gesture specificities of a limited user set. The selection of specific users limits the diversity of training data, considering that a user executes gestures of a same class in a similar way.

The choice between raw and scaled data, as shown in the confusion matrices, plays an important role in the effectiveness of a classifier. In all cases, scaled data generate better results than raw data, which justifies the normalization in conjunction with the execution of a calibration routine for each user. Despite the fact that all users execute the same gestures, each one has their own finger flexure capabilities, which reflects directly on the acquired data. It is also possible to store the calibration values for a specific user, using these values as a *calibration identity* for that user.

The choice of gestures to compose a vocabulary has a direct impact on the choice of hardware to be used, and vice versa. Gestures P02, P03, and P04, for instance, have a distinct visual representation; however, their finger flexure patterns are quite similar. The same occurs with gestures P11 and P14. These gestures produce the worst classification rates, since the data glove used restricts the acquisition of data to simple flexure values. For a correct classification of these gestures in the context of the present work, abduction measures between fingers are required or the gestures must be grouped and resumed to a single finger flexure pattern.

Finally, the performance obtained for all techniques allows their use in conjunction with interactive interfaces. In this case, the evaluation time can be even smaller, since there is no need to retrieve data from text files: Sensor values can be retrieved and dispatched directly to the classifiers, avoiding I/O handling. The training time, in turn, can be minimized by storing the already trained structure for each classifier, loading it whenever necessary.

7 Conclusions and future work

This work presents an evaluation of static gesture recognition techniques, aiming to identify the best method in terms of precision and performance. The data used for training and evaluation of the selected classifiers were collected from a number of users, and different combinations of data types and partitioning strategies were tested.

Through the analysis of obtained results, it became clear that the choice of a gesture recognition technique must consider a set of variables. Each variable influences the results in a specific way, and the final result must be subsidized by the whole variable set.

The importance of normalization is attested by the differences presented in the confusion matrices, where the results

Table 10 Precision of recognition techniques with similar gestures grouped—Neural Network

	Nearest Neighbor	Actual (raw/scaled)												Precision (%) (raw/scaled)
		P01	P02/P03/P04	P05	P06	P07	P08	P09	P10	P11/P14	P12	P13	P15	
Predicted (raw/scaled)	P01	239/891	416/75	0/22	27/0	31/0	75/0	0/0	0/0	45/9	85/1	3/6	179/96	21.73/81.00
	P02/P03/P04	681/494	1238/2359	2/82	81/0	123/9	219/8	0/129	3/0	142/12	260/13	13/17	538/177	37.52/71.48
	P05	174/36	325/28	0/694	28/114	98/12	81/0	0/1	10/3	38/0	144/45	20/9	182/158	0.00/63.09
	P06	71/0	168/0	1/7	43/1074	167/4	122/0	30/0	133/0	171/0	163/10	3/5	28/0	3.91/97.64
	P07	31/0	158/0	0/1	38/79	248/998	117/0	29/0	108/0	131/0	190/15	6/0	44/7	22.55/90.73
	P08	45/0	90/0	0/0	44/1	138/13	255/975	31/13	234/96	100/0	130/0	28/2	5/0	23.18/88.64
	P09	13/1	138/0	0/0	47/8	79/5	96/13	140/975	77/1	423/97	84/0	3/0	0/0	12.73/88.64
	P10	78/0	110/0	1/3	42/18	123/23	218/59	21/3	239/946	110/2	128/1	20/45	10/0	21.73/86.00
	P11/P14	41/8	303/0	0/9	93/12	130/1	202/3	251/167	151/83	827/1911	186/0	11/6	5/0	37.59/86.86
	P12	45/0	117/0	1/8	26/10	123/27	113/0	0/0	60/0	16/0	502/1046	49/9	48/0	45.64/95.09
	P13	72/0	85/0	0/0	26/0	84/0	164/33	0/0	132/33	31/0	265/8	183/1026	58/0	16.64/93.27
	P15	184/63	424/8	0/54	27/3	38/1	34/0	0/0	0/1	13/0	99/0	0/0	281/970	25.55/88.18
Median precision (%) (raw/scaled)														22.40/85.89

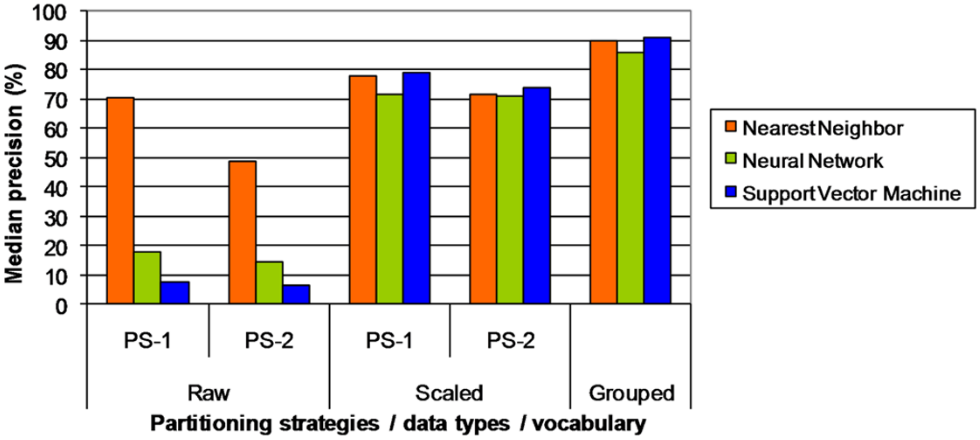


Fig. 8 Precision evaluation results for static gestures

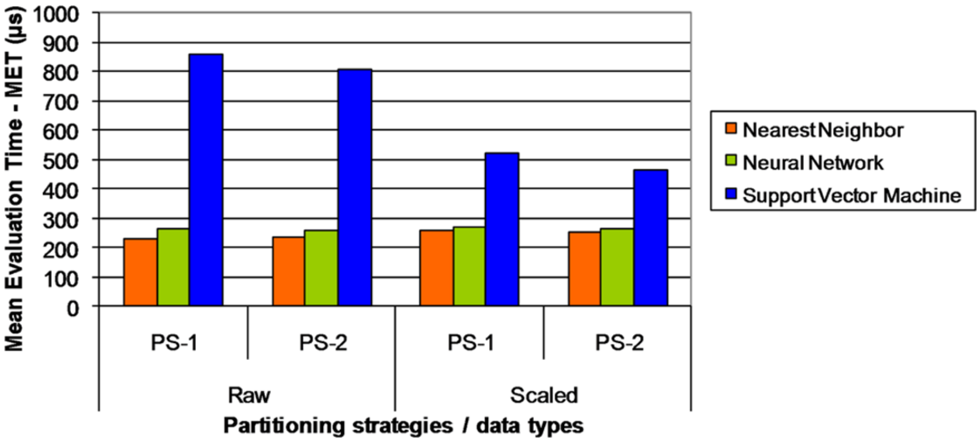


Fig. 9 Performance evaluation results for static gestures

from raw and scaled data were discrepant enough to point out the superiority of scaled data. Partitioning strategies, in their turn, contribute to improve the recognition rates by providing heterogeneous training datasets, and here the close relation between hardware and gesture vocabulary proved to be a relevant aspect. The performance factor contributes

Table 11 Precision of recognition techniques with similar gestures grouped—Support Vector Machine

	Support Vector Machine	Actual (raw/scaled)												Precision (%) (raw/scaled)
		P01	P02/P03/P04	P05	P06	P07	P08	P09	P10	P11/P14	P12	P13	P15	
Predicted (raw/scaled)	P01	11/884	4/152	0/23	0/0	0/0	0/0	0/0	0/0	0/0	0/0	0/0	1085/41	1.00/80.36
	P02/P03/P04	2/174	18/3048	0/40	0/0	0/0	0/4	0/2	0/1	0/1	0/0	0/2	3280/28	0.55/92.36
	P05	0/26	0/41	12/934	0/8	0/0	0/0	0/0	0/9	0/0	0/5	0/2	1088/75	1.09/84.91
	P06	0/0	0/0	0/21	43/1053	2/18	0/0	0/0	0/1	0/0	0/7	0/0	1055/0	3.91/95.73
	P07	0/0	0/6	0/0	2/42	20/1028	0/0	0/0	0/3	0/0	0/21	0/0	1078/0	1.82/93.45
	P08	0/0	0/0	0/0	0/0	0/5	19/1008	0/6	0/71	0/0	0/0	0/10	1081/0	1.73/91.64
	P09	0/0	0/0	0/1	0/0	0/1	0/6	3/979	0/0	0/113	0/0	0/0	1097/0	0.27/89.00
	P10	0/0	0/0	0/5	0/0	0/6	0/89	0/0	13/969	0/2	0/0	1/29	1086/0	1.18/88.09
	P11/P14	0/0	0/0	0/10	0/0	0/0	0/2	5/100	0/3	20/2085	0/0	0/0	2175/0	0.91/94.77
	P12	0/0	0/0	0/0	0/14	0/20	0/0	0/0	0/0	0/0	14/1066	0/0	1086/0	1.27/96.91
	P13	0/0	0/5	0/0	0/1	0/0	0/7	0/0	4/18	0/0	0/4	21/1065	1075/0	1.91/96.82
	P15	0/84	0/15	0/39	0/0	0/0	0/0	0/0	0/0	0/0	0/0	0/0	1100/962	100.00/87.45
Median precision (%) (raw/scaled)														9.64/90.96

Table 12 Performance of training/evaluation using raw data

	Performance (raw)			
	PS-1		PS-2	
	MTT/SD	MET/SD	MTT/SD	MET/SD
Nearest Neighbor	173.4 ms/15.52 ms	229.82 µs/322.78 µs	169.7 ms/14.63 ms	239.55 µs/368.47 µs
Neural Network	40.78 s/0.08 s	264.34 µs/220.20 µs	41.6 s/0.03 s	259.69 µs/188.4 µs
Support Vector Machine	1.12 s/0.08 s	859.32 µs/261.55 µs	1.11s/0.07 s	809.62 µs/311.87 µs

Table 13 Performance of training/evaluation using scaled data

	Performance (scaled)			
	PS-1		PS-2	
	MTT/SD	MET/SD	MTT/SD	MET/SD
Nearest Neighbor	176.1 ms/14 ms	262.32 µs/154.71 µs	177.1 ms/18.72 ms	251.62 µs/204.67 µs
Neural Network	40.68 s/0.05 s	272.19 µs/154.65 µs	41.59 s/0.07 s	268.08 µs/188.48 µs
Support Vector Machine	472.2 ms/14.65 ms	522.68 µs/407.8 µs	468.4 ms/20.52 ms	467.37 µs/409.16 µs

to establish how suitable a technique is for interactive use, since it demands fast response times and adaptability to vocabularies of variable length.

Using precision as a selection criterion, the SVM technique presented the best results for both original and modified dataset, achieving a recognition rate of 79.13% and 90.96%, respectively. In terms of performance, the nearest neighbor was the faster method, with a median evaluation time of 229.82 µs. Considering that all methods presented a good evaluation time, which makes all of them suitable for interactive use, precision was assumed to be the most important criterion to guide the method selection. Therefore, the main conclusion of this work is that the SVM technique is the best recognition method among the selected ones, in a scenario composed by finger-flexure patterns, normalized and calibrated data, and heterogeneity-oriented dataset partitioning.

As future work, the instrumented data[7] can be used as input for other classifiers, allowing a comparison between the results obtained through these classifiers and the results presented on this paper. The visual data, in turn, can be used as an input for visual-based gesture recognition techniques, aiming to identify the best classifiers for this kind of interface. Finally, the SVM-based classifier (which presented the best results) can be validated through its integration to a real application, like GIL-Gesture Interaction Layer, a component of the Cyclops 3D Framework (a generic interac-

[7] http://www.lapix.ufsc.br/index.php/gil-gesture-interaction-layer.

tive visualization and manipulation framework for three-dimensional medical environments) [29, 30].

Acknowledgements We would like to thank Adiel Mittmann and Rodrigo Bittencourt Cabral for their contributions to this work.

References

1. Hewett TT, Baecker R, Card S, et al (1997) Curricula for human–computer interaction. ACM SIGCHI. http://old.sigchi.org/cdg/. Accessed 5 September 2009
2. Myers BA (1998) A brief history of human–computer interaction technology. ACM Interact 5(2):44–54. doi:10.1145/274430.274436
3. Wilson AD, Izadi S, Hilliges O, et al (2008) Bringing physics to the surface. In: Proc 21st annu ACM symp user interface softw technol, pp 67–76. doi:10.1145/1449715.1449728
4. Rick J, Rogers Y (2008) From DigiQuilt to DigiTile: Adapting educational technology to a multi-touch table. In: 3rd IEEE int workshop horiz interact hum comput syst, pp 73–80. doi:10.1109/TABLETOP.2008.4660186
5. Tani BS, Maia RS, Wangenheim Av (2007) A gesture interface for radiological workstations. In: Proc 20th int symp comput-based med syst, pp 27–32. doi:10.1109/CBMS.2007.6
6. Xu D, Yao W, Zhang Y (2006) Hand gesture interaction for virtual training of SPG. In: Proc 16th int conf artif real telexistence, pp 672–676. doi:10.1109/ICAT.2006.68
7. Deller M, Ebert A, Bender M, et al (2006) Flexible gesture recognition for immersive virtual environments. In: Proc int conf inf vis, pp 563–568. doi:10.1109/IV.2006.55
8. LaViola JJ (1999) A survey of hand posture and gesture recognition techniques and technology. Technical Rep CS-99-11, Brown University
9. Iwai Y, Shimizu H, Yachida M (1999) Real-time context-based gesture recognition using hmm and automaton. In: Proc int workshop recognit analysis track faces gestures real-time syst, p 127. doi:10.1109/RATFG.1999.799235
10. Lee C, Ghyme S, Park C, et al (1998) The control of avatar motion using hand gesture. In: Proc ACM symp virtual real softw technol, pp 59–65. doi:10.1145/293701.293709
11. Kjellström H, Romero J, Kragić D (2008) Visual recognition of grasps for human-to-robot mapping. In: IEEE/RSJ int conf intell robot syst, pp 3192–3199. doi:10.1109/IROS.2008.4650917
12. Kim J-H, Roh Y-W, Shin J-H, et al (2005) Performance evaluation of a hand gesture recognition system using fuzzy algorithm and neural network for post PC platform. In: Int workshop soft comput appl, pp 129–138. doi:10.1007/11676935_16
13. Bowman DA, Kruijff E, LaViolla JJ, et al (2005) 3D user interfaces: Theory and practice. Addison-Wesley, Boston
14. Fausett L (1994) Fundamentals of neural networks: Architectures, algorithms, and applications. Prentice-Hall, Upper Saddle River
15. Stergiopoulou E, Papamarkos N (2009) Hand gesture recognition using a neural network shape fitting technique. Eng Appl Artif Intell 22(8):1141–1158. doi:10.1016/j.engappai.2009.03.008
16. Bailador G, Roggen D, Tröster G, et al (2007) Real time gesture recognition using continuous time recurrent neural networks. In: Proc 2nd int conf body area netw, pp 1–8
17. Cristianini N, Shawe-Taylor J (2000) An introduction to support vector machines and other kernel-based learning methods. Cambridge University Press, Cambridge
18. Ren Y, Zhang F (2009) Hand gesture recognition based on MEB-SVM. In: Proc int conf embed softw syst, pp 344–349. doi:10.1109/ICESS.2009.21
19. Liu Y, Gan Z, Sun Y (2008) Static hand gesture recognition and its application based on support vector machines. In: Proc int conf softw eng artif intell netw parallel distrib comput, pp 517–521. doi:10.1109/SNPD.2008.144
20. Chen Y-T, Tseng K-T (2007) Developing a multiple-angle hand gesture recognition system for human machine interactions. In: 33rd annu conf IEEE ind electron soc, pp 489–492. doi:10.1109/IECON.2007.4460049
21. Meng H, Pears N, Bailey C (2007) A human action recognition system for embedded computer vision application. In: Conf comput vis pattern recognit, pp 1–6. doi:10.1109/CVPR.2007.383420
22. Duda RO, PE Hart, Stork DG (2000) Pattern classification. Wiley-Interscience, New York
23. Athitsos V, Stefan A, Alon J, et al (2008) Translation and scale-invariant gesture recognition in complex scenes. In: Proc 1st int conf pervasive technol relat assist env, pp 1–8. doi:10.1145/1389586.1389595
24. Ziaie P, Müller T, Foster ME, et al (2008) Using a naïve Bayes classifier based on *k*-nearest neighbors with distance weighting for static hand-gesture recognition in a human-robot dialog system. In: Proc 13th int CSI comput conf
25. Tarrataca L, Santos AC, Cardoso JMP (2009) The current feasibility of gesture recognition for a smartphone using J2ME. In: Proc ACM symp appl comput, pp 1642–1649. doi:10.1145/1529282.1529652
26. Wang RY, Popović J (2009) Real-time hand-tracking with a color glove. In: Int conf comput graphics interact tech, pp 1–8. doi:10.1145/1576246.1531369
27. 5DT data glove ultra series. http://www.5dt.com/downloads/dataglove/ultra/5DTDataGloveUltraDatasheet.pdf. Accessed 7 November 2009
28. Sarle WS (1997) Neural network FAQ. ftp://ftp.sas.com/pub/neural/FAQ.html. Accessed 2 November 2009
29. Tani BS, Nobrega T, Santos TR, et al (2006) Generic visualization and manipulation framework for three-dimensional medical environments. In: Proc 19th int symp comput-based med syst, pp 27–31. doi:10.1109/CBMS.2006.91
30. Silva AFB, Nobrega THC, Carvalho DDB, et al (2009) Framework for interactive medical imaging applications. In: Colloq comput Brazil. INRIA coop adv chall, pp 126–129