Design of a Hand Pose Recognition System for Mobile and Embedded Devices

https://doi.org/10.3991/ijes.v10i04.35163

Houssem Lahiani[1,2(✉)], Mahmoud Neji[2,3]
[1]National School of Electronics and Telecommunications, University of Sfax, Sfax, Tunisia
[2]Multimedia Information Systems and Advanced Computing Laboratory, Sfax, Tunisia
[3]Faculty of Economics and Management of Sfax, University of Sfax, Sfax, Tunisia
lahianihoussem@gmail.com

Abstract—Today, smart devices such smart watches and smart cell phones are becoming ever-present in all fields that influence the quality of life of the modern people. These on-board systems have revolutionized the behavior of human beings and especially their way of communicating. In this context and to improve the experience of using these devices, we aim to develop a system that recognizes hand poses in the air by a smart device. In this work, the system is based on Histogram of Oriented Gradient (HOG) features and Support Vector Machine (SVM) classifier. The impact of using HOG and SVM on mobile devices is studied. To carry out this study, we used an improved version of the "NUS I" dataset and obtained a recognition rate of approximately 94%. In addition, we conducted run speed experiments on various mobile devices to study the impact of this task on this embedded platform. The main contribution of this work is to test the impact of using the HOG descriptor and the SVM classifier in terms of recognition rate and execution time on low-end smartphones.

Keywords—android, HOG, SVM, gesture detection, run speed

1 Introduction

Today, smart devices such as smart watches and smart cell phones are becoming ever-present in all fields that influence the quality of life of the modern human being. We could use them for working, playing, shopping [1], and even in the education we use now mobile learning [2].

The majority of today's smartphones have a touch screen. This sort of screen has enhanced the way we interact with mobile devices, but it is expected that future mobile devices, such as Google glasses and smart watches, would incorporate a more interactive and intuitive interaction interface. Some of these devices are equipped with a voice recognition system to control them, but this solution may not be effective in noisy places. Moreover, this voice recognition-based solution is not suitable for the deaf and dumb people, which makes in-air gestures recognition a crucial solution to control these devices. Indeed, many complex problems arise for the development of such a system, and which make the realization of an interactive system to interact with mobile

devices a daring task. That's why we tried through this work to design and develop in-air hand poses detection system for low-end smart devices without using any contact peripheral. Here, we are interested in the recognition of hand poses to interpret the user's intentions using only the camera integrated in the machine.

The most troubling issue in this context is managing the large number of poses, especially with smart low-end machines that have limited computational capabilities. Although there are several algorithms that have shown unprecedented efficiency in the field of human-computer interaction, striking a balance between realism and robustness under different lighting conditions, especially when interacting with a low-end device limited in terms of critical resources remains difficult. The main challenge of recognizing hand gestures by entry-level smartphones is that they have limited computing and power capabilities, and in return vision-based recognition tasks are computationally expensive. To recognize gestures, a manipulation of a considerable number of degrees of freedom and a great variability of the 2D aspect according to the point of view of the camera, even for the same gesture, is required. This task is become more challenging after the release of a new variant of the Android OS (Go edition) which is intended for devices with 2 GB Random Access Memory (RAM) or less [3] making it difficult to perform vision-based tasks in this kind of devices.

The system targeted by this work is intended to work well on Android devices. Thus, Android is the widely used operating system (OS) around the world according to the statistics carried out during the month of August of the year 2022 by Statcounter Global Stats [4]. Additionally, most smartphone manufacturers offer Android-compatible devices [5]. Android devices may have better hardware specifications than Apple's devices, which use iOS, the second most popular mobile OS. Thus, Android devices are better than iOS-equipped ones in terms of cost what makes them more acquirable according to [6].

As we indicated earlier, the biggest dilemma to develop our system is to deal with vision-based tasks which are usually heavy tasks for the processor and RAM and in contrast most of low-end devices are computationally limited according to Lahiani, H. and Neji, M. [7]. In this context, several methods and approaches to recognize hand gestures in the air through an entry-level mobile device have been proposed. Among which there are those based on contact devices, accelerometers, gloves, etc. but which are in contrast annoying for users even if they give good detection results. Other works which are essentially based on vision tasks have been proposed in [8,9,10,11,12,13,14] and which may be considered more comfortable for the user. The faced issue here is to find an effective manner to interact with low-end devices in a more intuitive way than the existing contact-based methods. Another issue to be solved here is to find algorithms and methods which are robust against machine vision problems such as contrast and luminance change and of being at the same time suitable for devices which are computationally limited.

The work presented in this article is based on the Histogram of Oriented Gradient (HOG) features and the Support Vector Machine (SVM) classifier. Our system attained a recognition rate of approximately 94%, which is better than the results attained by the system developed in [15] that is based on Random Forest classifier, and which obtained a recognition rate of 93%. In addition, our system performs better than the system developed in [16] which is based on back propagation neural network, and which had a recognition rate of 91.66%. Therefore, the main contributions of our paper are as

follows: improve the recognition rate compared to other studies and find inexpensive methods for calculation that do not influence the smartphone performance.

2 Literature review

Various studies have been conducted around hand gesture/posture recognition for mobile and embedded devices. This section reviews some of them.

Both HOG and SVM features have been extensively employed to detect objects because most of their performance depends on the learning process rather than changes in lighting. Nowadays, there is a diverse literature in this field. In this context, a mobile app was created by Che, Y. et al. in [17] to develop a hand pose estimation method in 3D space to estimate hand poses per frame. In this system, a trained random forest was implemented to provide a rough initial estimate of the hand pose, after which a gradient-based model fitting is implemented. This hybridization accelerates convergence and restricts tracking loss. With this human-mobile interaction interface, stable and smooth implementation of common operations such as "Touch", "Grasp" and "Hold" with an in-air interface has become possible. However, this system uses a depth sensor, which does not exist in all types of devices, especially low-end devices.

We cite also the work done by Verdadero, M. S. et al. in [18] who constructed a mobile system that aims at recognizing static hand poses using the camera of an Android devices to perform commands on certain household appliances using infrared light. A 100% gesture recognition success rate was observed on the Samsung Galaxy S4 with a recognition time of 2.26451 seconds while there were only three hand gesture detection failures with the Samsung Galaxy Note 3 with a recognition time of 2.6991 seconds. The disadvantage of this system is that it uses infrared which could sometimes be a problem, because recognition will be better if working in natural lighting conditions.

In the same context, Chavan, S et al. put forward a vision-based gesture recognition system in [19] which seeks to find a mixture between image processing and deep learning architecture with less complication to implement the system in mobile applications. This system recognized ten different signs using a Convolutional Neural Network. It achieved a testing rate of 87.5%. The weakness of the aforementioned system is that the recognition rate is a little low compared with other systems.

To recognize hand gestures by a mobile device, Elhenawy, I, and Khamiss, A set forth an application designed for Android operating system that permits mobile devices to recognize ten Arabic letters in [20]. Both decision tree and neural network were used as classification tools. An accuracy rate of 97% was obtained. One of the main weaknesses of this system is the use of a computationally expensive task on the native side.

Hartanto, R. et al. suggested a translation system that focused on Indonesian sign language recognition using a smartphone in [16]. Skin color detection and convex hull algorithm were applied to obtain the contour of the hand. After that convexity defect algorithms have been implemented to extract hand gesture features. A back propagation neural network classifier was also utilized. The proposed system was able to recognize different alphabet signs with an accuracy value of 91.66%. The limitation of this system lies in the use of a resource-intensive contour detector that is not suitable for devices with restricted computational capacities, because the computation of convex hull is time consuming according to [21].

In [22], Qiao, Y. et al. offered a recognition system using a smartphone. This system was based on Principal Component Analysis (PCA). PCA solves such issues as different size of captured gesture image, different angle of gesture rotation and reverse gesture. The results of the experiment show that the average recognition rate of the proposed method is 93.95%. Although PCA avoids the instability of the earliest geometric methods that were developed, it is not precise enough to describe the intricacies of the (geometric) manifolds present in the original image space, which constitute a drawback for this system.

In a study conducted by Sharma, T. et al. in [23], an Android application was designed in order to recognize Air-swipe gestures based on a dataset created by the authors that contains four gestures (up, left, down, right). These gestures that make up the dataset were taken for 5 different people. The system has a recognition rate of 96%. However, the problem with this system is that it is designed to recognize a low number of gestures, it is likely that its recognition rate could be affected in case it was decided to extend the number of gestures to recognize.

In a study conducted by Jimoh et al. in [24], an Android application for hand sign recognition was designed for selected English vocabularies. For this system, a recognition model was designed, implemented, and evaluated on 230 samples of hand gestures. For features extraction FAST and Rotated BRIEF, and principal component analysis where used. For classification and recognition purposes template matching algorithm was used. A recognition rate of about 87% was attained, and an average accuracy value of 88% and 91% for the average recall rate were obtained. However, the use of template matching in this system could affect the recognition rate due to its sensitivity to rotation.

3 System design

We want through this work to create an intuitive gesture interface between the user and the mobile phone without any physical interaction. The system is based on vision so the first step here is to detect the hand so that you can then move on to the feature extraction step and then to the classification step to get a result as mentioned by Figure 1.

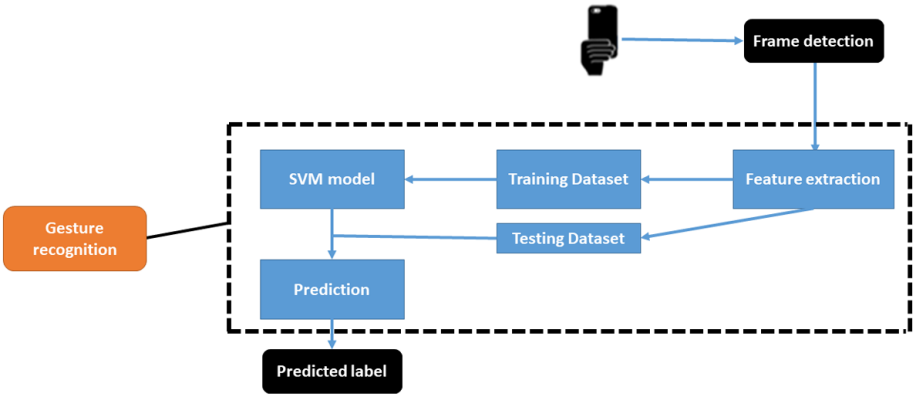

Fig. 1. The flowchart of the system

3.1 Frame detection

In any vision-based system the first step is to detect the region of interest. The region of interest in our case is the user's hand which will have to put his hand in front of the phone camera to be able to start the recognition process.

3.2 Features extractions with HOG descriptor

HOG descriptors are suggested by Dalal and Triggs. HOG is a local feature extraction algorithm. To measure HOG input, a support vector machine classifier with a linear model was voluntarily selected by Dalal and Triggs. In our case, the cropped partial image is divided into blocks, and for each one, with a cell as a unit, the features are extracted. A characteristic vector was computed for each cell within the area by assessing unsigned orientations in "*K*" orientation cells weighted by the amplitude of the gradient. The magnitude of the gradient "*g*" and the angle "*θ*" are computed by utilizing equations 1 and 2, separately as defined by Dalal and Triggs in [25].

$$g(u,v) = \sqrt{g_x(u,v)^2 + g_y(u,v)^2} \tag{1}$$

$$\theta_{(u,v)} = arctan\frac{g_y(u,v)}{g_x(u,v)} \tag{2}$$

In this study, each feature point is connected to a 3 × 3 block of cell descriptors. The size of each cell is 8 × 8 pixels. The HOG feature was reconstructed to achieve a better result. We calculated contrast sensitive features for each cell in the image. We also classified the gradient directions of each pixel into a bin in 18 directions. A normalization process was performed on the 2 * 2 neighborhood following the calculation of HOG features for every cell. Afterwards, we got a set of normalized eigenvalues and 18-dimensional contrast sensitive features "F1". For contrast-insensitive features, however, normalized eigenvalues were summarized and the sum of 9 different directions was normalized. Finally, we obtained 13-dimensional contrast insensitive features "F2". Hence, the set {F1, F2} represents the final HOG functionality.

3.3 Postures classifications with Support Vector Machine

Vapnik and Chervonenkis designed a supervised learning algorithm in [26], called SVM, within the statistical learning field. In the mid-1990s, an SVM implementation was suggested with the introduction of kernel trick and non-separable generalization. SVM is currently one of the most successful learning algorithms. Its ability to estimate sophisticated models for the computational cost of a very simple model has made it a central component of the machine-learning field where it has been particularly successful in image recognition. SVM has gained special attention of both researchers and practitioners according to Burges, J.C in [28]. In this work, we provided manual annotations of each hand pose in the enhanced NUS hand posture dataset I (Figure 2). The classification task can be seen as a multi-class problem. The hand posture recognition

method is based on a Support Vector Machine that uses the kernel function as presented in (3) as defined by Cortes C. & Vapnik V. in [27]:

$$h(x) = \sum_{i=1}^{n} \alpha i \, yi \, k(x, xi) + b \tag{3}$$

Where k is the kernel function, xi stands for the training samples and their respective class labels are represented by yi, the parameters αi and b are obtained after training. We trained our model using LIBSVM wrapper for android [29]. Specifying a training dataset file and a model file is similar to how LIBSVM binaries are used. Information of our trained model are put into that model file.

The training dataset file where our extracted features were saved in a "sparse matrix form" shows the standard format for the training file. This work aims to examine the effect of SVM on mobile and embedded systems. We applied the k-fold cross validation method thanks to its efficiency in decreasing overfitting. The model parameters are optimized when k is equal to 5.

4 Training and classification

4.1 Dataset

To develop and train our model, the NUS hand gesture database I (The NUS Datasets, 2010) [30] was used. The NUS I image database basically contains 240 photos representing postures, these postures are organized in 10 classes, each class is composed initially by 24 images. In order to get a well-trained classification model, we need to have a good number of samples. In this context, we chosen to move forward the NUS I database by including 51 hand gesture pictures to each class. Newly introduced hand gestures photos were taken with diverse individuals and in numerous settings and different lighting conditions. All those images were employed as positive samples to obtain a trained model. We must also utilize different negative pictures that don't incorporate what we need to distinguish to build an accurate classification model. Negative images that don't contain hand gestures (backgrounds, faces, etc) were used. We relegated to each hand sign a letter or a word for recognition purposes (Figure 3). We also performed some refinement operations on the images to get a better result. The NUS dataset was previously utilized by Pisharady, P.K et al. in [31] to recognize hand gestures by using SVM classifier. The result gotten by this framework was 94.36% utilizing cross-validation and not real-time tests.

Fig. 2. Some posture of enhanced NUS I dataset

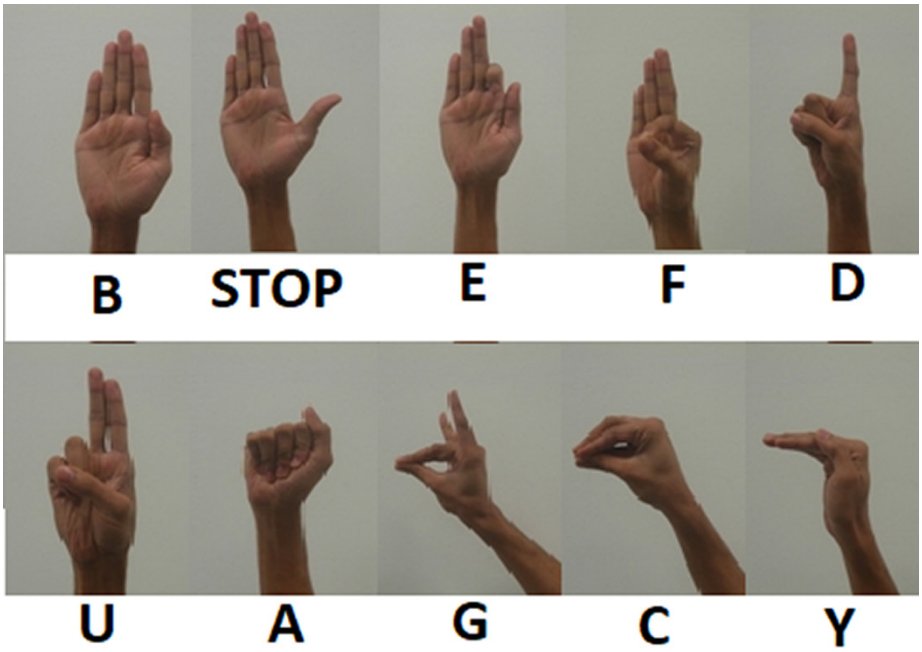

Fig. 3. Grammar to interpret gestures

4.2 System assessment

The proposed system in this work is outlined for real-time use in entry-level mobile systems. In order to measure not only the recognition rate but also the impact of our system on the runtime on a low-end device, we analyze this system's performance based on real-time tests. Different persons were asked to do a series of NUS hand gestures in front of the camera while the program was running in order to assess the real-time recognition rate. Tests have been conducted with various backdrops and lighting setups. Fifty pictures of each hand pose were made in varied lighting and backdrop settings. Figure 4 shows several postures made by individuals that our system accurately identified against various backdrops and lighting conditions. The name associated with each sign is shown in the capture box's upper portion. Our system was put through the identical training and evaluation procedures as the previously created systems in the literature in order to make a comparison. Tables 1–4 present the results of recognition acquired by our system and sate of the art systems.

Fig. 4. Gestures detection with different persons and in different lightning conditions and backgrounds

Table 1. Recognition rate of different state of the art systems

		System in [32]		System in [33]		System in [34]	
Pose	**Frames**	**Identified Correctly**	**Percentage**	**Identified Correctly**	**Percentage**	**Identified Correctly**	**Percentage**
STOP	50	46	92%	45	90%	48	96%
B	50	47	94%	45	90%	48	96%
C	50	45	90%	44	88%	44	88%
D	50	47	94%	44	88%	45	90%
E	50	48	96%	43	86%	45	90%
F	50	48	96%	44	88%	47	94%
G	50	47	94%	42	84%	45	90%
U	50	46	92%	45	90%	44	88%
Y	50	46	92%	43	86%	45	90%
A	50	46	92%	45	90%	47	94%
RECOGNITION RATE			93.2%		88%		91.6%

Table 1 shows that the system created in [32] achieved the best recognition rate. Both SVM and Local Binary Pattern (LBP) features were used by this system. Next, the approach suggested in [34] by Lahiani, H. and Neji, M. utilized the AdaBoost classifier in conjunction with HOG and LBP features. The approach proposed by Lahiani, H. et al. in [33] that employs the LBP features with Adaboost classifier has the lowest recognition rate. To make a comparison, we used training and testing procedures for our system that were identical to those in the table above (Table 1). In contrast, the major results of our system are shown in Table 2.

Table 2. Recognition rate of our system

Distance <= 75 cm			
Feature: HOG			
Classifier: SVM			
Sign	**Captures**	**Recognized**	**Recognition Rate**
B	50	48	96%
STOP	50	47	94%
E	50	48	96%
F	50	47	94%
D	50	48	96%
U	50	45	90%
A	50	46	92%
G	50	47	94%
C	50	46	92%
Y	50	46	92%
Average			93.6%

The findings of several hand gesture recognition systems for mobile devices are reported in Tables 1 and 2. After contrasting these various systems, they provide an overview of the key findings. Based on results from real-time tests, a correct hand gesture recognition may be achieved at up to 75 cm. For each posture, precision and recall (Pr) metrics were generated to assess the performance of our system. The Pr metrics were chosen because, as demonstrated by Howse et al. in [35], they would be superior than the ROC concept (receiver operating characteristic).

- Precision = TP / (TP + FP)
- Recall = TP / (TP + FN)
- F1 Score = 2 * (Recall * Precision) / (Recall + Precision)

With:

TP: true positive, regarded as a reliable detection,

FP: false positive, often known as a detection for which there is no posture,

The terms "false negative" (FN) and "F1 score" (weighted average of Precision and Recall metric scores) refer to gestures for which there is no detection.

The assessment procedure was the same as in [10].

A test set of 35 photos for each class of gesture was elaborated in order to get the results of precision, recall, and F1 score. These photos, which include hand gestures, were collected from the NUS dataset-II. In addition to the outcomes produced by the HOG-SVM system, Tables 3 and 4 also shows our Pr metrics results and the results obtained by the systems created in [32,34] for comparison.

Table 3. Comparison of Precision, Recall and F1 score between our system and system in [34]

Pose	Precision		Recall		F1 Score	
	HOG and SVM	**HOG-LBP and AdaBoost**	**HOG and SVM**	**HOG-LBP and AdaBoost**	**HOG and SVM**	**HOG-LBP and AdaBoost**
A	0.612555	0.45679	0.894234	1	0.727066	0.627119
B	0.826423	0.935484	0.945672	0.828571	0.882035	0.878788
C	0.674818	0.421053	0.954732	0.914286	0.790734	0.576577
D	0.657412	0.617762	0.975125	0.925267	0.785353	0.740874
E	0.834563	0.625471	0.964751	0.954796	0.894947	0.755818
F	0.845823	0.754892	0.927564	0.916582	0.884810	0.827916
G	0.765712	0.584253	0.874982	0.910203	0.816708	0.711682
U	0.712045	0.619892	0.874658	0.996974	0.785019	0.764462
Y	0.694756	0.611458	0.874125	0.997035	0.774187	0.758033
STOP	0.797516	0.618532	0.858742	0.989752	0.826997	0.761300

Table 4. Recall and F1 score between our system and system in [32]

Pose	Precision		Recall		F1 Score	
	HOG and SVM	LBP and SVM	HOG and SVM	LBP and SVM	HOG and SVM	LBP and SVM
A	0.612555	0.65369	0.894234	0.81564	0.727066	0.725739
B	0.826423	0.954789	0.945672	0.81425	0.882035	0.878937
C	0.674818	0.513658	0.954732	0.84985	0.790734	0.640307
D	0.657412	0.787654	0.975125	0.914758	0.785353	0.846461
E	0.834563	0.784526	0.964751	0.841256	0.894947	0.811901
F	0.845823	0.845362	0.927564	0.84369	0.884810	0.844525
G	0.765712	0.645825	0.874982	0.745832	0.816708	0.6922351
U	0.712045	0.694761	0.874658	0.84391	0.785019	0.7621067
Y	0.694756	0.683945	0.874125	0.949123	0.774187	0.7950041
STOP	0.797516	0.756492	0.858742	0.846952	0.826997	0.7991703

What fraction of affirmative identifications was accurately recognized is what the Precision metric tries to determine. Recall thus seeks to provide a response to the following query: What proportion of real positive findings were accurately detected. Metrics for Precision and Recall should be examined in order to properly evaluate the model performance. Consequently, Precision and Recall frequently conflict. Indeed, the increase in Precision has a negative impact on recall, and the opposite is also true. The hardware employed for evaluation in this study consists of two separate cell phones, the first of which has an Android OS running at version 5.1 and a 13 MP camera. This smartphone has an octa-core Central Processor Unit (CPU) running at 1.5 GHz and 1.5 GB of RAM. The second device has Android 4.4 installed on it, a Cortex Dual Core 1.3 GHz processor, 512 MB of RAM, and a 4 MP camera.

4.3 Runtime speed

We have investigated the impact of the various methods used on a low-end mobile device by analysing the processing time. In order to do this, TimingLogger was used to measure the processing time [36]. To put it another way, we looked at how much time each detection task took (Figure 5). The logcat Command-line Tool [37] of Android Studio, the official Integrated Development Environment (IDE) for developing native Android applications, allowed us to display results of the detection time during the processing of the task when the device was connected to the IDE via a USB port. TimingLogger was able to do this for us. To make sure the program is functioning properly, we want to keep track of how it is currently being executed. Android Studio offers a log of current events in the device.

The "Logcat" is where we may display the TimingLogger findings. According to the systems mentioned above in the literature, the systems based on HOG-AdaBoost and LBP-AdaBoost had the best performance in terms of detection and execution time. The execution time for each hand gesture for the two systems was 1 ms. In comparison, employing the AdaBoost algorithm along with the HOG and LBP features, an execution

time of 2 ms for each hand posture detection was attained. This may be understood by the fact that combining two features makes recognition tasks more complex. By taking two milliseconds to identify each gesture, HOG features with SVM classifier exhibit performance comparable to that of HOG-LBP-AdaBoost. In fact, compared to Adaboost, the SVM classifier had reduced testing errors. SVM, however, needs a lot longer time to recognize hand gestures. The second mobile phone's execution time is longer than the one made by the first mobile device. This is caused by the first smartphone's Octa-core CPU, which has eight processing cores. The Octa-core CPU enables smartphones to carry out more complex activities, such as playing high-definition films and graphically demanding games, without consuming battery life and limiting their functionality. A CPU with eight cores can guarantee quick loading times.

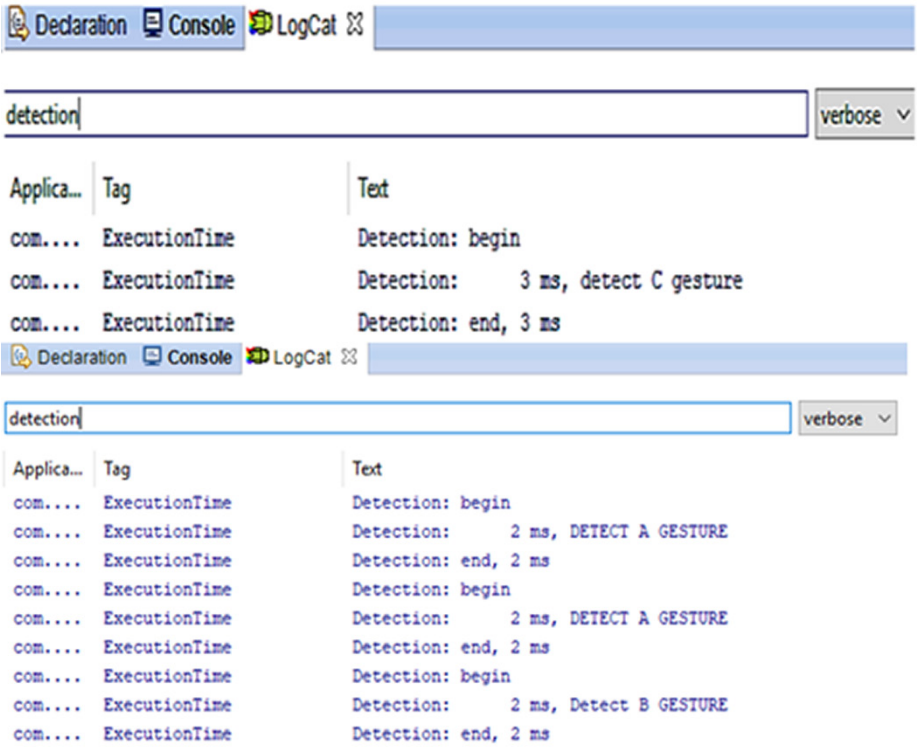

Fig. 5. Required time to detect some gestures

5 Conclusion

In this research, we present a hand gesture-based human-machine interaction system. To find static hand postures, we used a sign recognition algorithm. Despite some issues, the outcomes demonstrated the application's robustness. While certain sample photos from the NUS database II were utilized for assessment, training was conducted using the improved NUS hand gesture database I. SVM is a useful technique for recognizing hand gestures. The recommended technique, which uses HOG features and SVM to

classify 10 signs from the improved NUS database, achieves real-time performance with cutting-edge approaches accuracy even for various signers. The recognition rate of our system is 94%, which represents a respectable rate compared to the work carried out previously, nevertheless the execution time remains a little high which could affect the performance of the device and its energy consumption. Even while SVM achieves a higher recognition rate than the preceding systems, we still need to find a way to shorten execution time. Our future works will put special focus on: (1) comparing the trained model of the two versions of the NUS hand gesture database. In addition, (2) applying deep learning algorithms, by using a Convolutional Neural Network (CNN) for image classification [38], for our task by using smartphones. In this respect, the effect of these algorithms on mobile devices will be also examined. Furthermore, (3) investigating the influence of the previously used approaches on the energy consumption of mobile applications.

6 References

[1] Wohllebe, A., Hübner, D.-S., Kämpf, D., & Podruzsik, S. (2022). "Classification of Mobile App Users in Multi-Channel Retail—an Exploratory Analysis." International Journal of Recent Contributions from Engineering, Science & IT (iJES), 10(1), pp. 4–16. https://doi.org/10.3991/ijes.v10i01.28071

[2] Sulisworo, D., Yunita, L., & Komalasari, A. (2017). "Which Mobile Learning is More Suitable on Physics Learning in Indonesian High School?" International Journal of Recent Contributions from Engineering, Science & IT (iJES), 5(1), pp. 97–104. https://doi.org/10.3991/ijes.v5i1.6494

[3] Introducing Android 9 Pie (Go edition). (2021). [Online] Available: https://www.android.com/versions/go-edition/

[4] Statcounter company. (2022). [Online] Available: https://gs.statcounter.com/os-market-share/mobile/worldwide/#monthly-202204-202204-bar

[5] Android device provisioning services. (2022). [Online] Available: https://developers.google.com/zero-touch/resources/manufacturer-names

[6] Jamdaade, K., Khairmode, A., & Kamble, S. (2016). "A Comparative Study between Android & iOS." International Journal of Current Trends in Engineering & Research (IJCTER), 2(6), pp. 495–501. https://doi.org/10.14569/IJACSA.2017.081123

[7] Lahiani, H., & Neji, M. (2017). "Comparative Study between Hand Pose Estimation Systems for Mobile Devices." Journal of Information Assurance and Security, 12, pp. 218–226.

[8] Lahiani, H., Kherallah, M., & Neji, M. (2016). "Vision Based Hand Gesture Recognition for Mobile Devices: A Review." In proceedings of 16th *International Conference on Hybrid Intelligent Systems* (HIS 2016), Advances in Intelligent Systems and Computing. vol. 552. Springer, Cham, pp. 308–318. https://doi.org/10.1007/978-3-319-52941-7_31

[9] Lahiani, H., & Neji, M. (2020). "A Survey on Hand Gesture Recognition for Mobile Devices." International Journal of Intelligent Systems Technologies and Applications, 19(5), pp. 458–485. https://doi.org/10.1504/IJISTA.2020.111065

[10] Lahiani, H., & Neji, M. (2020). "Hand Pose Estimation System Based on Combined Features for Mobile Devices." International Journal of Intelligent Information and Database Systems, 13(2–4), pp. 436–453. https://doi.org/10.1504/IJIIDS.2020.109465

[11] Lahiani, H., Elleuch, M., & Kherallah, M. (2015). "Real Time Hand Gesture Recognition System for Android Devices." *2015 15th International Conference on Intelligent Systems Design and Applications (ISDA)*, pp. 591–596. https://doi.org/10.1109/ISDA.2015.7489184

[12] Lahiani, H., Elleuch, M., & Kherallah, M. (2016). "Real Time Static Hand Gesture Recognition System for Mobile Devices." Journal of Information Assurance & Security, 11(2), pp. 67–76.
[13] Lahiani, H., Kherallah, M., & Neji, M. (2017). "Hand Pose Estimation System Based on a Cascade Approach for Mobile Devices." In proceedings of *17th International Conference on Intelligent Systems Design and Applications (ISDA)*. Advances in Intelligent Systems and Computing, vol. 736, pp. 619–629. https://doi.org/10.1007/978-3-319-76348-4_60
[14] Lahiani, H., Kherallah, M., & Neji, M. (2016). "Hand Pose Estimation System based on Viola-Jones Algorithm for Android Devices." *2016 IEEE/ACS 13th International Conference of Computer Systems and Applications (AICCSA)*, pp. 1–6. https://doi.org/10.1109/AICCSA.2016.7945717
[15] Song, J., Sörös, G., Pece, F., Fanello, S. R., Izadi, S., Keskin, C., & Hilliges O. (2014). "In-air Gestures around Unmodified Mobile Devices." *27th ACM User Interface Software and Technology Symposium (UIST'14) 2014*, pp. 319–129. https://doi.org/10.1145/2642918.2647373
[16] Hartanto, R., & Kartikasari, A. (2016). "Android Based Real-Time Static Indonesian Sign Language Recognition System Prototype." *2016 8th International Conference on Information Technology and Electrical Engineering (ICITEE)*, pp. 1–6, https://doi.org/10.1109/ICITEED.2016.7863311
[17] Che, Y., Qi, Y., & Song, Y. (2019). "Real-Time 3D Hand Gesture Based Mobile Interaction Interface." *2019 IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct)*, pp. 228–232, https://doi.org/10.1109/ISMAR-Adjunct.2019.00-41
[18] Verdadero, M. S., Martinez-Ojeda, C. O., & Cruz, J. C. D. (2018). "Hand Gesture Recognition System as an Alternative Interface for Remote Controlled Home Appliances." *2018 IEEE 10th International Conference on Humanoid, Nanotechnology, Information Technology,Communication and Control, Environment and Management (HNICEM)*, pp. 1–5. https://doi.org/10.1109/HNICEM.2018.8666291
[19] Chavan, S., Yu, X., & Saniie, J. (2021). "Convolutional Neural Network Hand Gesture Recognition for American Sign Language." In proceedings of *2021 IEEE International Conference on Electro Information Technology (EIT)*, pp. 188–192. https://doi.org/10.1109/EIT51626.2021.9491897
[20] Elhenawy, I., & Khamiss, A. (2014). "The Design and Implementation of Mobile Arabic Fingerspelling Recognition System." International Journal of Computer Science and Network Security, 14(2), 149–155, 2014.
[21] Zhao, J., & An, J. (2013). "An Iterative Convex Hull Approach for Image Segmentation and Contour Extraction." International Journal of Pattern Recognition and Artificial Intelligence 26(7), 1255013. https://doi.org/10.1142/S0218001412550130
[22] Qiao, Y., Feng, Z., Zhou, X., & Yang, X. (2017). "Principal Component Analysis Based Hand Gesture Recognition for Android Phone Using Area Features." *In proceedings of 2nd International Conference on Multimedia and Image Processing (ICMIP)*, pp. 108–112. https://doi.org/10.1109/ICMIP.2017.36
[23] Sharma, T., Kumar, S., Yadav, N., Sharma, K., & Bhardwaj, P. (2017). "Air-Swipe Gesture Recognition using OpenCV in Android Devices." *2017 International Conference on Algorithms, Methodology, Models and Applications in Emerging Technologies (ICAMMAET)*, pp. 1–6. https://doi.org/10.1109/ICAMMAET.2017.8186632
[24] Jimoh, K. O., Ajayi, A. O., & Ogundoyin, I. K. (2020). "Template Matching Based Sign Language Recognition System for Android Devices." FUOYE Journal of Engineering and Technology, 5(1), pp. 42–48. https://doi.org/10.46792/fuoyejet.v5i1.465
[25] Dalal, N., & Triggs, B. (2005). "Histograms of Oriented Gradients for Human Detection." *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, pp. 886–893, vol. 1. https://doi.org/10.1109/CVPR.2005.177

[26] Vapnik, A. Ya. Chervonenkis. (1964). "A Class of Algorithms for Pattern Recognition Learning." Avtomat. i Telemekh., 25(6), pp. 937–945.
[27] Cortes, C., & Vapnik, V. (1995). "Support-Vector Networks." Machine Learning, 20, pp. 273–297. https://doi.org/10.1007/BF00994018
[28] Burges, C. J. A. (1998). "Tutorial on Support Vector Machines for Pattern Recognition." Data Mining and Knowledge Discovery, 2, pp. 121–167. https://doi.org/10.1023/A:1009715923555
[29] LIBSVM Wrapper. (2015). LIBSVM Ported to Android JNI Environment [online] Available: https://github.com/cnbuff410/Libsvm-androidjni
[30] The NUS Hand Posture Datasets. (2010). [online] Available: https://www.ece.nus.edu.sg/stfpage/elepv/NUS HandSet
[31] Pisharady, P. K., Vadakkepat, P., & Loh, A. P. (2013). "Attention Based Detection and Recognition of Hand Postures Against Complex Backgrounds." International Journal of Computer Vision, 101(3), p. 403–419. https://doi.org/10.1007/s11263-012-0560-5
[32] Lahiani, H., & Neji, M. (2019). "Hand Gesture Recognition System Based on LBP and SVM for Mobile Devices." Computational Collective Intelligence. (ICCCI). Lecture Notes in Computer Science, vol. 11683, pp. 283–295. https://doi.org/10.1007/978-3-030-28377-3_23
[33] Lahiani, H., Kherallah, M., & Neji, M. (2017). "Hand Gesture Recognition System based on Local Binary Pattern Approach for Mobile Devices." *17th International Conference on Intelligent Systems Design and Applications (ISDA)*. Advances in Intelligent Systems and Computing, vol. 736, pp. 180–190. https://doi.org/10.1007/978-3-319-76348-4_18
[34] Lahiani, H., & Neji, M. (2018). "Hand Gesture Recognition Method based on HOG-LBP Features for Mobile Devices." *22nd International Conference on Knowledge-Based and Intelligent Information & Engineering Systems (KES)*. Procedia Computer Science, Vol. 126, pp. 254–263. https://doi.org/10.1016/j.procs.2018.07.259
[35] Howse, J. et al. "Object detection performance testing", in Howse, J., 1984, OpenCV 3 Blueprints, Expand your Knowledge of Computer Vision by Building Amazing Projects with OpenCV 3, Print version: OpenCV 3 blueprints, Packt Publishing, Birmingham, UK. 2015
[36] Android Developers TimingLogger. (2021). [Online] Available: https://developer.android.com/reference/android/util/TimingLogger
[37] Logcat Command-line Tool. (2021). [Online] Available: https://developer.android.com/studio/command-line/logcat
[38] Karthikeyan, N. (2022). "Review of Deep Transfer Learning Models for Image Classification." International Journal of Recent Contributions from Engineering, Science & IT (iJES), 10(1), pp. 17–28. https://doi.org/10.3991/ijes.v10i01.29783

7 Authors

Houssem Lahiani is a PhD and engineer in computer science. He is a senior lecturer at the National School of Electronics and Telecommunications of Sfax, University of Sfax, Tunisia. He currently continues his research on pattern recognition and the distant control of mobile and embedded devices. He is also a member of the Multimedia and InfoRmation Systems and Advanced Computing Laboratory (MIRACL).

Mahmoud Neji is a Full Professor at the Faculty of Economics and Management of Sfax, University of Sfax, Tunisia. He is a member of the Multimedia and InfoRmation Systems and Advanced Computing Laboratory (MIRACL).

Article submitted 2022-09-04. Resubmitted 2022-10-25. Final acceptance 2022-11-14. Final version published as submitted by the authors.