A face recognition system based on a Kinect sensor and Windows Azure cloud technology

Dan-Marius Dobrea[1], Daniel Maxim[1], Stefan Ceparu[1]
[1] Technical University "Gh. Asachi", Faculty of Electronics, Telecommunications and Information Technology, Iaşi, Romania
mdobrea@etti.tuiasi.ro, danielmaxim88@yahoo.com, ceparu.stefan@gmail.com

***Abstract*— The aim of this paper is to build a system for human detection based on facial recognition. The state-of-the-art face recognition algorithms obtain high recognition rates base on demanding costs – computational, energy and memory. The use of these classical algorithms on an embedded system cannot achieve such performances due to the existing constrains: computational power and memory. Our objective is to develop a cheap, real time embedded system able to recognize faces without any compromise on system's accuracy. The system is designed for automotive industry, smart house application and security systems. To achieve superior performance (higher recognition rates) in real time, an optimum combination of new technologies was used for detection and classification of faces. The face detection system uses skeletal-tracking feature of Microsoft Kinect sensor. The face recognition, more precisely - the training of neural network, the most computing-intensive part of the software, is achieved based on the Windows Azures cloud technology**

I. INTRODUCTION

The computer's or system's ability to sense and respond to the requirements of a specific user helps them to better adapt to the user's needs and also, to develop the naturalness of human-computer communication.

The face identification and face recognition are some of the toughest and challenging problems in the computational intelligence field. Sometimes even a human has difficulty in recognizing faces. It is well known that people have trouble recognizing face differences amongst other people of different races than their own.

One of the main challenges in face recognition algorithms is given by the face detection (i.e. locating the face or faces in an image) – a preliminary, but necessary, step before attempting faces recognition. Various face detection algorithms have been proposed [1], [2], [3], [4], [5], [6]; however, almost all of them have poor performance in the presence of: scale variation [6], face occlusions [1], [4], rotation [3], variation in illumination [2], [6], orientation, variation in skin colors [2], [6], complex backgrounds [5] etc. Another important drawback is the requirement of high computing power in order to run the algorithm in real time.

In the face recognition field, the reported solutions presented in the literature are of rare encountered diversity, but all have a common feature: they are computing intensive – requiring powerful and expensive systems. In connection with the face data acquisition methods, face recognition methods can be classified into three main categories [7]: first the one that operates on intensity images, second one that deals with video sequences and, the last one, that uses other sensory data (such as 3D information).

Comparing the performances obtained in face recognition field with the ones obtained in OCR (optical character recognition) area one can remark that in the field of OCR over forty years were necessary to built acceptable quality algorithms able to recognize written symbols. The face recognition algorithms developments are in this moment in the childhood stage.

Our goal is to develop a low cost, but reliable, face recognition system. The system was developed and tested related to automotive industry applications – security system (to have the authorization to start the engine) and setting the car environment accordingly to the driver necessities (adjust the mirrors, driver seat, the steering wheel etc.).

The classical algorithms (for face detection and recognition) are very complex and, additionally, are data and computational intensive requiring powerful systems and large fast memories. In order to solve these problems we have used the most recent advances in computer vision techniques, computer design, sensors and distributed computing.

As a result, a low cost solution was devised (without any compromise in system's reliability and performances) based on two edge technologies: Windows Azure and Kinect sensor.

Moving the computational load and the complexity of the algorithms on Windows Azure cloud and Kinect sensor we are able to build an embedded system able to work in real time, with maximum performance for face recognition.

Our project has three main sections: face detection, face identification and training the neural network (used to recognize faces).

The system operates in two modes (see Fig. 1): learning mode and recognizing mode.

Since the training of neural network requires a lot of computational resources (it is impossible to be done on a low cost embedded system) and, additionally, must be done only few times, we have decided to let the Windows Azure (the Microsoft's cloud platform) handle this job.

The novelty of our system does not rely on the algorithms used, but in the main concepts of the system: how to use and integrate several new technologies allowing us to obtain results previously unattainable under the same conditions. The resulting build on system is very accurate with a very competitive price.

II. THE HARDWARE SYSTEM

The requirements of the face detection and recognition algorithms are sustained by a PC104+ embedded system. The embedded system is a MOPS-PM system produced by Kontron. The MOPS-PM is an „all in one“ Intel® Pentium Processor Celeron® M (1GHz Hz 512KB L2 cache - Dothan) based PC/104+ board having 1 GByte of RAM, 2 serial ports, 1 LAN port 10/100 BaseT Ethernet, 1 parallel port, 2 USB 2.0, 1 Watchdog timer etc.

The link between the PC104+ system and the Windows Azure cloud infrastructure was a 3G wireless connection.

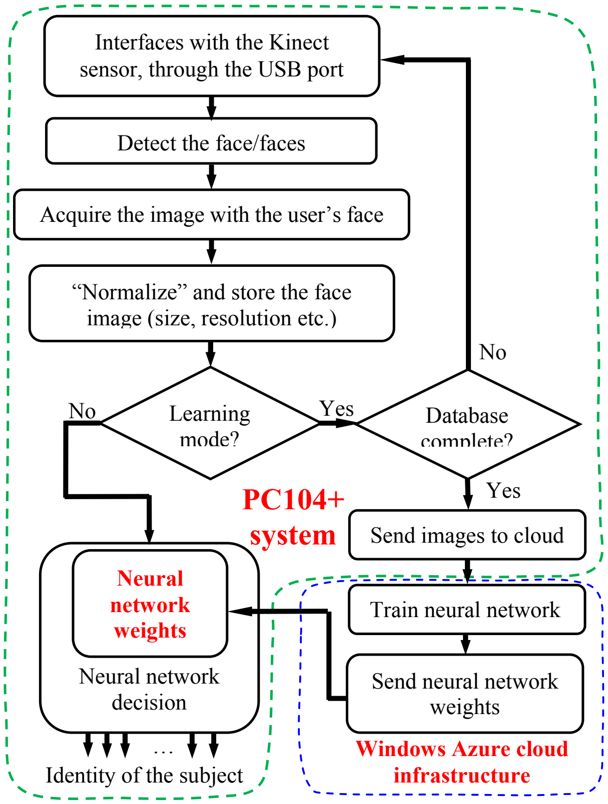

Figure 1. The software modules and the hardware allocated to support them

The Windows Embedded Standard 7 operating system runs on the PC104+ system and supports all the developed software modules, see Fig. 1. Windows Embedded Standard 7 is a componentized form of Windows 7 operating system that allows developers to select only the required components necessary to support their applications.

III. KINECT SENSOR AND FACE DETECTION

The Kinect sensor uses a RGB video camera (to capture an image), an IR projector, an infrared camera sensor (used to build a "depth map" of the area in front of it), four microphones and some special signal processing hardware that is able to preprocess all the data from the sensors – in this mode a part of the potential computational load of the system, that use Kinect sensor, is supported by the sensor itself.

One of the main applications of the Kinect sensor is to recognize and track people standing in front of it. The SDK (Software Development Kit), provided by Microsoft, running on the embedded device is able to: decode the information from sensors, recognize human elements in the images and map them into a body skeleton drew in the RBG space, Fig. 2(a). At the end we have access to different skeleton joints (e.g. head, left hand, right hand, elbow etc.) coordinates in the Cartesian system, having Kinect sensor as a reference.

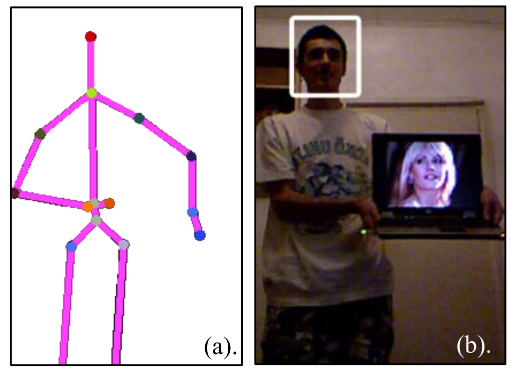

Figure 2. (a). Kinect skeleton and joints, (b). the detected face

Detection of the face is very simple mainly because we have the head coordinates (the center of gravity) in the RGB frame, Fig. 2(a). Even if the detection of the face seems to be trivial (from the point of view of the user of the Kinect SDK) a large part of the face recognition technologies existing in the Kinect is based on two powerful algorithms: learning-base descriptor and pose-adaptive matching technique [8].

The classical techniques for face detection confront with another problem: the false positive detection of faces. For example the OneVision algorithm for face detection developed by Innovation Labs, Microsoft's R&D center in Israel, detects all the faces in a frame, even if the faces are on a poster [9] – resulting low detection accuracy.

Comparing with the classical face detection algorithms (e.g. Viola–Jones or other methods) the Kinect face detection approach bring the following advantages: (1) the embedded system use the computational resources of the Kinect sensor to detect faces, (2) the 2D faces (from pictures, posters etc.) are not detected – see Fig. 2(b) and (3) the system is invariant to had rotation – the explanation will follow.

Having the head coordinates, in the next step we must detect and save only the face information. The face dimension in an image is proportional with the distance between the Kinect sensor and the human body. As a result, depending on the distance from the sensor, the dimension of the rectangle that framed the face was changed accordingly.

Figure 3. Samples with the obtained faces

In the last step, in order to obtain a good face image for the recognition process, the face images were resampled at the desired size (32 x 32 pixels) and converted to the grayscale. In Fig. 3 are presented some samples of the obtained faces images, all having the same resolution, for all face images the distances from the Kinect sensor was different.

IV. Windows Azure Cloud Platform

The facial recognition algorithm, used in this project, is based on an artificial neural network (ANN) structure. In a ANN, the training process is necessary only a few times (the first time when the face database is created or when a new user must be added or removed from the database), otherwise the system will work in the "recognition mode". In the "recognition mode" the system is able to detect a person, based on its face image and on the neural network weights (that embed knowledge acquired in the training stage). As a result, almost all the time (when the system is in the "recognition mode") the computation load is very low, with very rare spikes – when the system goes in "learning mode" (e.g. for 4 users, 13 imagines/user, the training process of ANN require around 140 hours, if we use the PC104+ system), see Fig. 1. These computational characteristics fit with cloud elasticity feature – ability to elastically provision and respectively free new resourced, in order to scale them rapidly outward and inward in a direct relation with the requirements of the application [10].

On a practical consumer scenario (for automotive industry, smart house application and security systems) a new user must be recognized by the system – as a result: (a) several imagine are acquired and (b) the embedded system must to obtain the ANN weights in order to get to the "recognition mode". Base on Azure cloud infrastructure the training process of ANN is reduced from hundreds of hours to 5 minutes.

A. *Neural network*

In this research we used an auto-associative neural network (AANN) [11]. The algorithm used in the classification process affect directly the recognition rate. In our project **the AANN was used only to demonstrate the feasibility of the system concepts.** Other neuronal network structures (with superior performances, like support vector machine – SVM) can be used without any system changes, only replacing directly the AANN.

In an AANN the input layer is linked to the output layer through the associated weights. In the training process, the input face image and the output face image are the same. After "storing" a set of prototype patterns in the memory (weights) based on auto-associative learning process the unknown face image is presented at the neural networks inputs. The recognition phase is based on computing differences (the errors) between the reconstructed image (obtained from the input unknown image going through the neural network weights) and each image prototype (7 for each user of the system). The selected candidate is founded based on the greatest degree of similarity. After a large number of tests, we establish a recognition threshold of 60%. The subjects who were not in the database obtained a percentage of similarity of about 45%. As a result, any picture which received a value of similarity smaller than 60% will be classified as intruder.

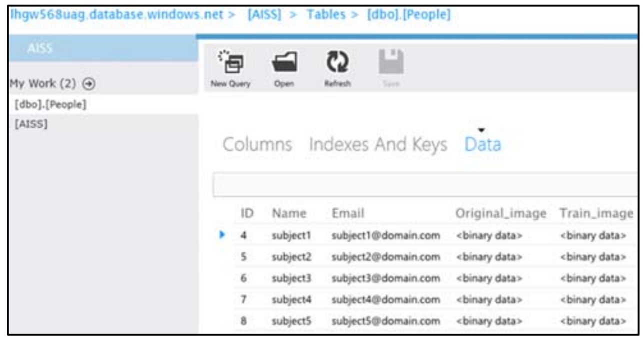

Figure 4. SQL Azure database view

B. *Windows Azure*

In order to train, in cloud, the neural network we need the following components: (1) Cloud SQL Database – to store the training face images, Fig. 4, (2) Cloud Worker – used to train the neural network, Fig. 5, and (3) Cloud Blob Storage – to hold the results of the training process.

The face images database is located in cloud (our choice to speed up the training process).

A Cloud Worker (named Cloud Service) is similar to a Windows Service and represents a container used to host any applications. Essentially it can run any code – our program is developed in C#. It can react to outside stimuli (e.g. by polling from the Azure Queue service) but can also open communication channels, query databases etc.

In our project the Cloud Worker handles several main functions as follows. First, handle the client-server communication: we have chosen the asynchronous

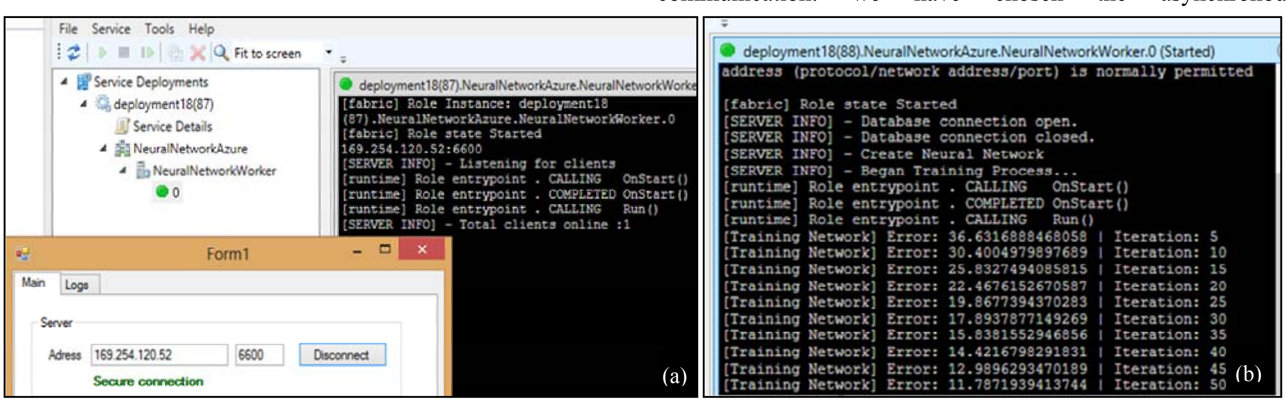

Figure 5. Cloud Worker states: (a) when a client just connected and (b) training the neural network

communication technology in order to minimize the costs and, in this mode, the worker can deal with multiple clients at the same time. Second, Cloud Worker manage the server-database communication used to get the pictures from the database and, finally, when the training process is completed, to store the best weights of the auto-associative neural networks. Third, Cloud Worker train the neural network based on the image faces retrieved from the database, Fig. 5(b). The algorithm used to train the network was backpropagation. Fourth, finally, when the training process is completed, the neural network is saved into a blob storage container.

V. RESULTS

To test the ability of the system to recognize correctly different faces, we have used 4 subjects, each of them having 7 face image prototypes based on which we have trained the auto-associative neural network. Even if it seems that the number of subject is small, for an automotive applications this number is adequate – it represent the potentially car/truck drivers. The recognition rate was 93%. To obtain such higher correct classification rate we have used some parameters provided by the SDK that helps to find the head orientation – the head rotation angle. As a result, the system grabbed the face images only when the face position is suitable for a good recognition – frontal faces and less than 20 degrees off position. All the tests took place in a room with normal natural light – not directly exposed to sun light – between 9.00 AM and 13.00 PM. A main problem remains the influence of the ambient illumination over the face recognition process.

As we have mentioned previously, in this paper, the goal was not to obtain higher classification rates (for this objective a more powerful neural network can be used and, additionally, some image preprocessing algorithms, to make face recognition invariant to illumination condition), but to prove the feasibility of the main concepts of the system.

The necessary time for the system to identify a user or an intruder and to act accordingly varies depending on the face position and orientation. The system will continuously make face orientation estimation and will acquire a face image only in frontal case. In a normal situation the system will respond, in the worst case, in approximately 30 seconds. The response time of the system, when a face image was acquired, is less than 200 ms.

Power consumption tests were performed during the normal system operation and reflect the power consumption only of the embedded system at which we have connected Kinect sensor, mouse and keyboard – the software make a normal cycle of image face recognition. As a result the power consumption was of 12.5W up to 14W – these values correspond to a current of 2.5A up 2.8A at a supply voltage of 5V DC. To make a comparison, with an Intel i7 personal computer, the measured standby power consumption and active power consumption was 33.03W and 102.2W respectively (for only one core – the system has 4 cores) [12].

VI. DISCUSSIONS AND CONCLUSIONS

In a classical system growing the complexity and the computational costs of the software compromise the execution efficiency and increase the power consumption. Exploring and identifying a solution for increasing the computational expenses without obtaining too much degradation in overall performance and energy consumption of the system are necessary steps for facial recognition applications in order to make them as popular as personal computers. The Windows Azure cloud technology can represent a solution for this problem. Our project and the obtained result sustain this conclusion.

Using and exploiting Kinect technology we can detect the faces in real time faster, easier and more accurate, Fig. 2, than other similar system [9].

The 3G data link connection is enough; the maximum quantity of exchanged data is around 200 Kbyte in one session.

As a final conclusion, the previously presented face recognition concepts can be used in a large field of application (as a human computer interface), obtaining higher classification rates without any significant computational burden to the embedded system. As a result of the above described concepts implementation, the final system is low-cost and it is able to provide all the functionalities of similar high-end facial recognition system.

ACKNOWLEDGMENT

The authors are grateful to the Microsoft Romania for the donation of a set of 2 Kinect devices.

REFERENCES

[1] D. Xingjing, L. Chunmei, and Y. Yongxia, "Analysis of detection and track on partially occluded face," Int. Forum on Inf. Tech. and App., vol. 3, 2009, pp. 158 – 161

[2] U. Yang, M. Kang, K. A. Toh, and K. Sohn, "An illumination invariant skin-color model for face detection", 4th IEEE Int. Conf. on Biometrics: Theory App. and Syst., 2010, pp. 1-6

[3] H. Chang, A. Haizhou, L. Yuan, and L. Shihong, "High-performance rotation invariant multiview face detection," IEEE Trans. on Patt. Analysis and Mach. Intell., vol. 29 , no. 4, pp 671 – 686, 2007

[4] L. Goldmann, U. J. Monich, and T. Sikora, "Components and their topology for robust face detection in the presence of partial occlusions," IEEE Trans. on Inf. Forens. and Sec., vol. 2, no. 3, part 2, pp. 559 – 569, 2007

[5] W. Yanjiang, and Y. Baozong, "Segmentation method for face detection in complex background," Elect. Lett., vol. 36, no. 3, pp. 213 - 214, 2000

[6] C. N. R. Kumar, and A. Bindu, "An efficient skin illumination compensation model for efficient face detection," 32nd Annual Conf. on IEEE Ind. Elect., France, Paris, 6-10 Nov. 2006, pp. 3444 – 3449

[7] R. Jafri, and H. R. Arabnia, "A survey of face recognition techniques," J. of Inf. Proc. Syst., vol. 5, no. 2, pp. 41-68, June 2009

[8] C. Zhimin, Y. Qi, T. Xiaoou, and S. Jian, "Face recognition with learning-based descriptor," IEEE Conf. on Comp. Vision and Patt. Recogn., 13-18 June 2010, San Francisco CA, pp. 2707 – 2714

[9] "Big brother? Microsoft unveils technology to recognize faces in video," http://nocamels.com/2011/03/big-brother-microsoft-unveils-technologyto-recognize-faces-in-video/, date of access: 23 feb. 2013

[10] P. Marshall, K. Keahey, and T. Freeman, "Elastic site: using clouds to elastically extend site resources," 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing, 17-20 May 2010, Melbourne, Australia, pp. 43 - 52

[11] C. Bishop, Neural Network for Pattern Recognition, New York, NY, USA, Oxford University Press, 1995

[12] Y. C. Wang, B. Donyanavard, and K. T. Cheng, "Energy-aware real-time face recognition system on mobile CPU-GPU platform," Int. Workshop on Comp. Vision on GPU 2010, Crete, Greece, Vol Part II, pp. 411-422