Hand-drawn symbol recognition in immersive virtual reality using deep extreme learning machines

Hubert Cecotti, Cyrus Boumedine, and Michael Callaghan

Faculty of Computing and Engineering, Ulster University,
Magee Campus, Northland Road
Derry∼Londonderry, Northern Ireland BT48 7JL, UK

Abstract. Typical character and symbol recognition systems are based on images that are drawn on paper or on a tablet with actual physical contact between the pen and the surface. In this study, we investigate the recognition of symbols that are written while the user is immersed inside a room scale virtual reality experience using a consumer grade head-mounted display and related peripherals. A novel educational simulation was developed consisting of a virtual classroom with whiteboard where users can draw symbols. A database of 30 classes of hand-drawn symbols created from test subjects using this environment is presented. The performance of the symbol recognition system was evaluated with deep extreme learning machine classifiers, with accuracy rates of 94.88% with a single classifier and 95.95% with a multiple classifier approach. Further analysis of the results obtained support the conclusion that there are a number of challenges and difficulties related to drawing in this type of environment given the unique constraints and limitations imposed by virtual reality and in particular the lack of physical contact and haptic feedback between the controller and virtual space. Addressing the issues raised for these types of interfaces opens new challenges for both human-computer interaction and symbol recognition. Finally, the approach proposed in this paper creates a new avenue of research in the field of document analysis and recognition by exploring how texts and symbols can be analyzed and automatically recognized in virtual scenes of this type.

Keywords: Virtual Reality, Symbol Recognition, Extreme Learning Machine, Multiple Classifier Systems

1 Introduction

Virtual reality (VR) extends and augments computer-generated 3D environments providing users with a means to enter, interact with and become immersed in alternate worlds [11]. Access to low cost, efficient, fully immersive and usable consumer grade VR systems with head mounted displays (HMD) and high quality graphics has become a reality for VR enthusiasts with the release

of the Oculus Rift, HTC Vive and similar devices. In addition, the use of 360^o motion tracked handheld wireless controllers with accompanying base stations allows the user to directly create, interact with and manipulate elements that are present in the virtual world with a high level of granularity and accuracy. In this context, we have proposed and created an educational simulation consisting of a virtual classroom with whiteboard where users can draw symbols on the screen. The symbols are automatically recognized and identified by the system.

The recognition of symbols is a central pillar of graphical image analysis and recognition systems identifying graphical entities that are present in technical documents (e.g., architectural diagrams, engineering drawings, technical maps) where the main tasks include image segmentation, layout understanding and graphics recognition [9, 28]. Graphics recognition has a long history of intensive research in the pattern recognition and document analysis community, and it is considered as a core part of graphical document image analysis and recognition systems. The recognition of single handwritten character and symbols is an old field of research in image processing and pattern recognition [21, 22, 13, 25, 20]. While accuracy remains below 100% for some problems, advances in machine learning and features extraction methods have created systems that reach almost human performance in some visual tasks. However, there are still some challenges remaining, i.e., documents with noisy characters or symbols, and particular scripts [6], which cannot be recognized with commercial optical character recognition technologies [3]. Symbol recognition is typically used in industries with a rich heritage of hand-drawn documents, and where there is a need to extract and understand the content and the logical structure of documents. Symbol recognition is therefore a key aspect of automatic image analysis systems and processes using a number of different approaches (e.g., statistical, structural and syntactic [10]). While in today's society documents are usually created on computers and there is a decline in the use of handwritten and hand-drawn content there are some situations where it is still easier, more convenient and intuitive for the user, to rapidly draw or sketch a symbol instead of searching for it using a menu driven approach [4].

In this paper, we present an application of symbol recognition using symbols created in and captured from a fully immersive virtual reality environment. In [27], a 3D sketch recognition framework for interaction within non-immersive virtual environments was presented, which allowed the user to draw symbols that would trigger subsequent commands or actions. Our approach takes advantage of new consumer-level virtual reality devices (e.g., HTC Vive) to create a room scale virtual reality experience that includes a white board that can be drawn on using the handheld Vive controller represented as a pen in the virtual scene. In contrast to other symbol recognition applications where the variation across examples is largely due to the quality of the input document (e.g., noise in the image) hand-drawn symbols of this type offer similar challenges as are found in handwritten character recognition applications. In addition, the shapes of the symbols created are usually more complex than the shapes of single characters or digits necessitating the use of advanced techniques such as Bayesian Networks

and Hidden Markov Models for processing [31, 32]. In order to recognize the hand-drawn symbols captured from the virtual reality environment, we propose to use deep extreme learning machines as this type of approach has shown a high level performance and accuracy in state-of-the-art single handwritten digit recognition in different scripts [5].

The remaining parts of the paper are organized as follows. Section 2 describes the Extreme Learning Machine framework and the classifiers used. Section 3 describes the virtual environment and the database of hand-drawn symbols created. The classification results are given in Section 4 and finally discussed in Section 5.

2 Methods

2.1 Random-vector functional links

Random Vector Functional-Link (RVFL) and Extreme Learning Machine (ELM) networks are a particular type of artificial feedforward neural networks that contain a single hidden layer [14, 15, 35]. This type of network can be used for both classification and regression [2, 24, 30]. It corresponds to a linear combination of non-linear representations of the input data (e.g., using a sigmoid function). The main characteristic of this system is the way in which the parameters (i.e., the weights) are assigned. The input weights and biases are set randomly and do not change over time. While this step is simple and it may seem inefficient due to the lack of training, RVFLs have the capability of universal approximation if the dimension of the input representation is large enough [17]. The parameters of a RVFL network can be obtained with linear regression methods using only matrix inversions and multiplications. It is worth noting that these operations are particularly well adapted for distributed learning [29]. Finally, RVFL networks can be estimated with a lead-square approach for learning the weights in the last layer.

Let us first consider a regression problem with one-dimensional scalar outputs $y \in \mathbb{R}$. A Functional Link Artificial Neural Network (FLANN) with a single output neuron is defined as a weighted sum of B non-linear transformations of the input $\mathbf{x}$ [24]:

$$f(\mathbf{x}) = \sum_{m=1}^{B} \beta_m h_m(\mathbf{x}; \mathbf{w}_m) = \beta^T h(\mathbf{x}; \mathbf{w}_1, \ldots, \mathbf{w}_B) \tag{1}$$

where the mth transformation is obtained with the parameters $\mathbf{w}_m$, and $\mathbf{x} \in \mathbb{R}^d$. Each functional link h_m maps the input data to a real number. The non-linearity is obtained with the sigmoid function, such as the multilayer perceptron:

$$h_m(\mathbf{x}; \mathbf{w}_m; b) = \frac{1}{1 + \exp^{\sigma}} \tag{2}$$

where $\sigma = -\mathbf{w}^T\mathbf{x} + b$. The set of parameters $\mathbf{w}_m$, $1 \leq m \leq B$, is chosen before the learning process and without any prior assumptions about the data. Moreover, the parameters are set randomly, in relation to a predefined probability

distribution [30]. After the estimation of the set of parameters, the weights $\boldsymbol{\beta}$ must be estimated. We consider a dataset $\mathcal{X}_{Train} = \{(\mathbf{x}_i, y_i)\}$ of N couples that contain an example $\mathbf{x}_i$ and the expected output y_i, $1 \leq i \leq N$. We denote by $\mathbf{H}$ the matrix containing the B representations of the N examples.

$$H = \begin{pmatrix} h_1(\mathbf{x}_1) & \dots & h_B(\mathbf{x}_1) \\ \vdots & \ddots & \vdots \\ h_1(\mathbf{x}_N) & \dots & h_B(\mathbf{x}_N) \end{pmatrix} \tag{3}$$

where each function h_m includes the corresponding set of parameters $\mathbf{w}_m$. The estimation of $\boldsymbol{\beta} = [\beta_1, \dots, \beta_B]^T$ can be obtained through a regularized least-square problem:

$$\boldsymbol{\beta} = \underset{\beta \in \mathbb{R}^B}{\arg\min} \frac{1}{2} \|\mathbf{H}\boldsymbol{\beta} - \mathbf{Y}\|_2^2 + \frac{\lambda}{2} \|\boldsymbol{\beta}\|_2^2 \tag{4}$$

where the vector $\mathbf{Y} = [y_1, \dots, y_N]^T$ is the ground truth of $\mathcal{X}_{Train}$. As the problem is convex, an estimation of $\hat{\boldsymbol{\beta}}$ can be obtained by:

$$\hat{\boldsymbol{\beta}} = \left(\mathbf{H}^T\mathbf{H} + \lambda\mathbf{I}\right)^{-1} \mathbf{H}^T\mathbf{Y} \tag{5}$$

where I is the identity matrix of size $B \times B$. For multiclass classification with M classes, $M \geq 2$, the ground truth $\mathbf{Y}$ is a matrix of size $N \times M$. $\mathbf{Y}(i,j) = 1$ if $\mathbf{x}_i$ belongs to the class j, $1 \leq j \leq M$, $\mathbf{Y}(i,j) = 0$ otherwise.

(6)

2.2 Multi-layer RVFL/ELM

Different variations of RVFL and ELM networks have been successfully used in a range of diverse but popular classification problems [16], and have been inspired by other techniques [8, 7]. Furthermore, ELM can be extended for deep learning architectures (ML-ELM) [18, 33]. The learning approach performs layer-by-layer unsupervised learning by using ELM auto-encoder (ELM-AE), which represents features based on singular values. With an ELM-AE model, the output $\mathbf{Y}$ is similar to the input $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_N]$. The decoder, i.e., the function that maps the input representation $h_1(x), \dots, h_B(x)$ of the input x to itself, corresponds to the parameters $\hat{\boldsymbol{\beta}}$ that are estimated. In addition, a key function for setting the value of the weights is to constraint the set of random weights in each layer to be orthogonal [18]. To create the coder afterward, $\hat{\boldsymbol{\beta}}^T$ is used to map x to the representation that was obtained. Then, ML-ELM stacks on top of ELM-AE to create a multilayer neural network similar to other deep network architectures. ML-ELM is a greedy approach, and it doesn't require fine-tuning after estimating the weights of the last layer. In a RVFL network with a single hidden layer, we denote by $\hat{\boldsymbol{\beta}}^{\mathbf{1}}$ the estimation of the weights for the first hidden layer. In the same way, we denote $\hat{\boldsymbol{\beta}}^{\boldsymbol{l}}$ as the estimation of the weights for the layer l, $1 \leq l \leq L$,

for ML-ELM of L hidden-layers. An ELM-AE is set for each layer l, and the extracted weights of an ELM-AE l are used subsequently to generate the inputs of the ELM-AE at the next layer $l+1$. An example of deep ELM with two hidden layers is presented in Fig. 1

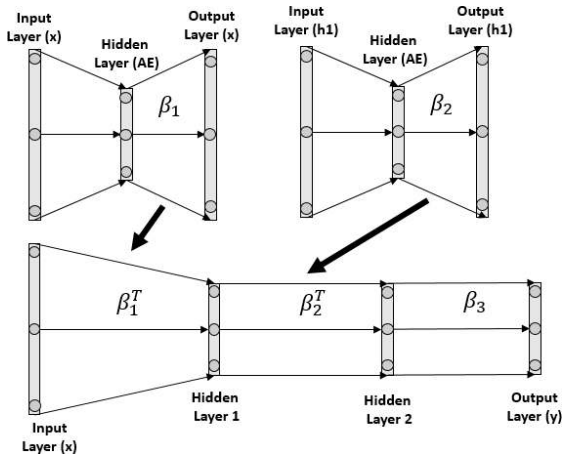

Fig. 1. A deep RVFL/ELM with two hidden layers.

As the generation of the RVFL/ELM classifiers is based on random weights, we assume it is possible to enhance the performance of the overall decision by combining the decisions of different classifiers [26]. We therefore propose to evaluate the performance by combining the decision scores resulting from 10 runs. Three functions are evaluated: the mean (the average score from each run is used before selecting the maximum), the maximum rules (the maximum score in each class and each run), and majority voting (the decision is based on the class that is chosen in most of the runs).

2.3 Performance evaluation

In the subsequent sections, we evaluate two types of architectures. The first type includes a single hidden layer (the number of neurons is set to: 100, 500, 1000, 10000). The second type includes two hidden layers, the number of neurons in the first hidden layer is set to: 400, 600, or 800, and the number of neurons in the second hidden layer is set to: 6000, 8000, 10000. For each architecture, 10 runs are evaluated.

3 System overview

3.1 VR scene

The room scale virtual scene containing the white board is depicted in Fig.2. The virtual environment was developed with the Unity game engine on an Alienware X51 desktop (I5-6600k, 16gb ram, Nvidia GTX 970 graphics card). The HTC Vive and accompanying base stations facilitate highly accurate 360 motion tracking where virtual representations of the physical handheld wireless controllers, used to navigate and interact with the virtual environment, are visible to the user inside the virtual scene. The HTC Vive wireless handheld controllers offer six degrees of freedom, and are tracked using the Lighthouse technology [1]. Each controller features several customizable buttons that can be tailored to a range of functions required in the application (e.g., a button to validate an image and draw a new image). A trackpad on the face of the controller is also available which is accessible with the users thumb and can be used to erase the content of the white screen.

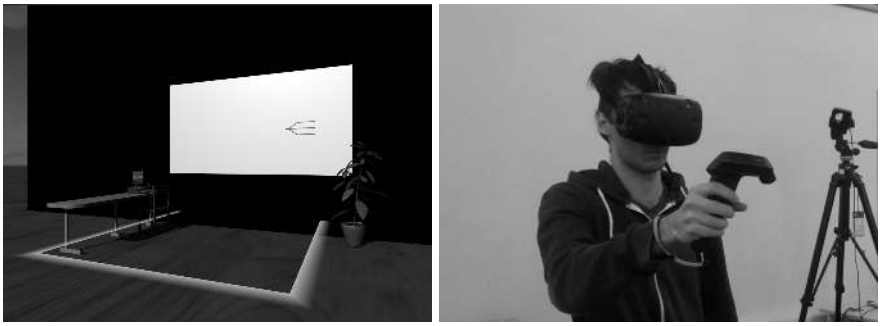

Fig. 2. Room scale virtual environment with the virtual white board (**left**: the virtual scene created (the lines on the ground represent the boundaries area of the simulation where the user can move safely using the Vive Chaperone system), **right**: a user wearing the HTC Vive headset and drawing in the virtual scene.

3.2 Controller choice

In order to write or draw on a virtual screen in a virtual environment without any physical contact between the controller and the virtual surface we used two main approaches. In the first approach (C1), the points that are drawn on the virtual screen correspond to the intersection between the virtual white board and a line coming from the orientation of the controller. In the second approach (C2), the points that are drawn on the screen are based on the intersection between the plane containing the virtual white board and a tangent to this plane passing by the position of the controller, i.e., parallel to the plane of the ground. In both

cases, the representation of the pen in the virtual environment does not need to touch the virtual screen i.e., it is the equivalent of drawing with a pointer. The two approaches were tested before the creation of the database. From this it was found that better results were obtained using first approach, which is based on the orientation of the controller. However, the first approach proved more difficult for the user and the level of difficulty increased depending on the distance between the user and the screen as a slight change in orientation of the controller will have a greater impact on the position of what is drawn on the white board. To illustrate the impact of the two different approaches used on the writing style, different examples of the word "Information" have been acquired from two participants, and at different distances (0.3 m, 0.6 m, 1 m, and 1.5 m). Fig. 3 depicts the large variability that occurs when a word is written based on controller orientation. The largest variations occur at a distance of 1.5 m where the curves of the letters have many deformations.

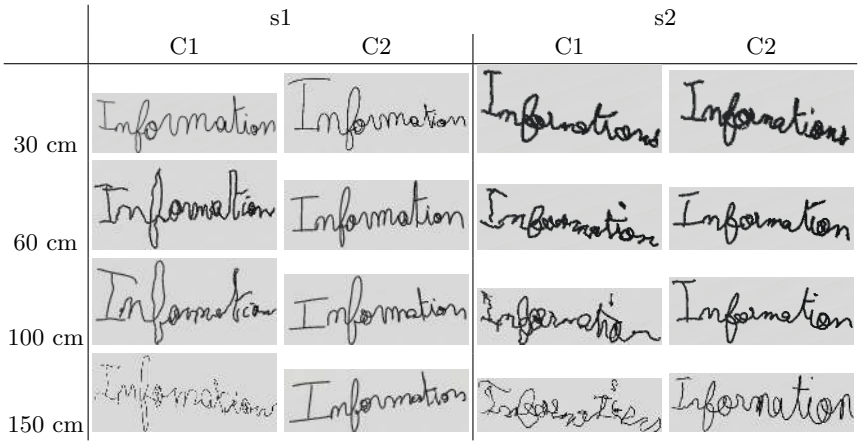

Fig. 3. Effect of the control mode on the style.

3.3 Image acquisition

Data was recorded from 18 able-bodied participants (age=22.8 ± 3.5, 3 females). Each participant had to write a series of symbols taken from a subset of 30 symbols that were used during the International Symbol Recognition Contest GREC'2011 [1] [12, 34]. The distance between users and the screen varied between participants. Each user had to reproduce a version of the symbol that was depicted on the right hand side of the screen. Each model of symbol was presented with eight different orientations (from 0 to 2π, with steps of $\pi/4$). An

[1] TC-10: http://iapr-tc10.univ-lr.fr/

example of the scene in use is given in Fig. 4. There were no time constraints imposed on participants when drawing the symbols. The total number of acquired images is 3059, with 170 ± 112 examples per person (min=23, max=354).

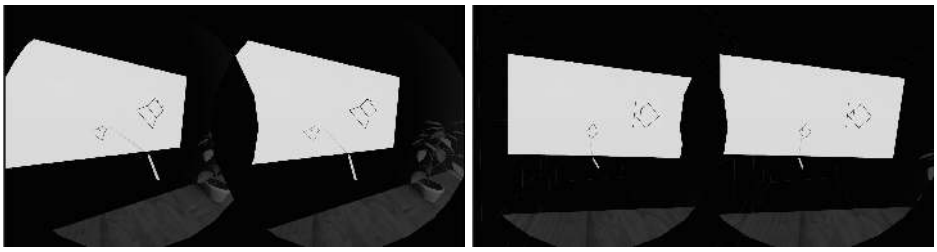

Fig. 4. Experimental paradigm in the virtual scene. The user has to draw the symbol on the left side of the white board following the template or model displayed on the right.

Each acquired image is in effect a screenshot (1024×768 pixels) of the virtual white screen in the virtual world and as a result the image background contains a number of artefacts related to the lighting effects used in the scene. Each image was pre-processed using the following approach: the image is binarized using the Otsu method to separate the background from the drawing [23]. The background of the image is then set to white while the drawing remains in gray scale. The image is then cropped and centered using its center of gravity and the white border around the image removed. The image is resized to 60×60, and a border of 2 white pixels was added around the image to provide a total image size of 64×64. Images drawn based on a particular orientation of the model used were normalized in relation to the orientation of the model in order to keep all the images with the same orientation. In the following section, we only consider symbols from the database with a minimum of 96 examples and limit the number of examples to 96 for each symbol. A representative image of each class is depicted in Fig. 5. For classification purposes, a 8-fold cross validation procedure is performed where for each class 84 examples are used for training and 12 examples are used for the test. In addition, we also consider an extended database where each image is rotated by $\{-15, -10, -5, 5, 10, 15\}$ degrees increasing the total number of images by a factor of 7. We denote this by using DB_0 and DB_1 for the original and the extended databases respectively.

4 Results

In the first instance, we are evaluating the extent to which it is possible to retrieve images that are similar to an image given as an input or template. To do this, we are using the inner-distance approach that takes into consideration complicated shapes and part structures such as handwritten symbols [19]. The method is

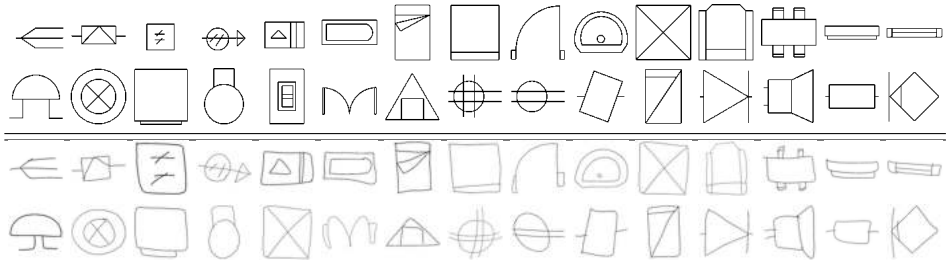

Fig. 5. Representation of the 30 symbols (rows 1 and 2: models, rows 3 and 4: representative examples of hand-drawn symbols acquired in virtual reality).

evaluated using DB_0 by estimating the bull's eye test where the number of relevant images is summed and divided by the maximum number of relevant images. The best performance is achieved with 1 image at 66% (see Fig. 6).

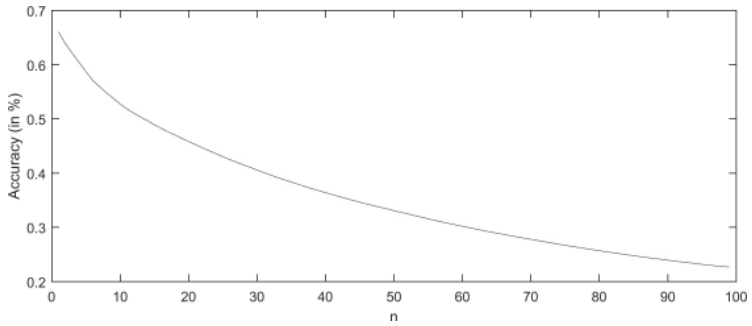

Fig. 6. Bull's eye test with the inner-distance on DB_0, in relation to n selected images.

The results corresponding to image classifications are given in Tables 1, 2, and 3. For each classifier architecture the number of functional links is given for each layer. As the analysis was performed with a 8-fold cross validation and 10 random initializations of the ELM classifiers, all the results correspond to the average accuracy (in %) across 80 runs. The best performance, by using a single classifier, for both DB_0 and DB_1 was obtained with architectures using two hidden layers. For DB_0, the best accuracy, 91.43% was obtained with the most complex architecture (800,10000). It is worth mentioning that the standard deviation across runs is about 0.01%. For DB_1, the highest level of accuracy was reached using the architecture (600,10000) with a performance of 94.88%. When the decisions of 10 classifiers are combined, the performance reaches 93.06% and 95.95% for DB_0 and DB_1, respectively, by using the "mean" rule. The performance using the "max" combination rule increases but the impact of this combined approach on the accuracy is lower than with the "mean" rule. The results

between the "mean" rule and majority voting are relatively similar. These results suggest that the different random initializations of the ELM classifiers can be exploited to create multi-classifier systems that improve the overall accuracy. By considering results from the top 5 best answers in each example, the accuracy reaches 99%. The accuracy corresponding to the top 5 shows the accuracy when the correct model is detected within the 5 top answers. In these cases, this performance can be used with the controller where the user can directly select one of the best answers with a button click. Finally, the evaluation was performed on a desktop with an Intel®Core^{TM}i7-3770 running Matlab 2015b with 16GB of memory, which resulted in a training time of about 20 s for DB_0 and about one minute for DB_1. The results indicate that the addition of hidden layers in the ELM classifier, the inclusion of the rotated images into the database, and the combination of decisions from several classifiers improve the overall performance.

Table 1. Accuracy (in %) with ELM using a single hidden layer.

	DB_0					DB_1				
	Acc.	Acc. (top5)	Mean	Max	Maj	Acc.	Acc. (top5)	Mean	Max	Maj
(100)	66.50	90.50	76.67	72.33	74.44	64.30	90.51	75.53	70.10	73.35
(500)	75.08	92.52	82.43	79.90	80.69	75.60	93.36	80.84	79.07	80.75
(1000)	71.14	89.92	81.74	78.99	80.83	76.48	93.23	81.54	79.64	81.05
(5000)	70.08	88.43	75.76	73.33	74.79	73.57	91.03	81.07	78.72	80.06
(10000)	73.00	89.81	74.51	73.40	73.61	59.18	83.48	81.88	73.27	78.73

Table 2. Performance with ELM using two hidden layers with DB_0 (time in second, accuracy in %).

	$Time_{train}$	$Time_{test}$	Acc.	Acc. (top5)	Mean	Max	Maj
(400,6000)	7.55	0.08	90.49	97.96	93.06	91.94	92.71
(400,8000)	7.84	0.09	90.72	98.04	92.50	91.46	92.15
(400,10000)	9.17	0.11	90.96	97.94	92.43	92.01	92.36
(600,6000)	13.04	0.10	90.76	98.15	92.99	91.74	92.85
(600,8000)	13.17	0.12	91.16	98.13	92.71	91.88	92.78
(600,10000)	14.73	0.14	91.28	98.24	92.57	92.08	92.29
(800,6000)	19.21	0.11	90.85	98.06	92.43	91.94	92.64
(800,8000)	20.63	0.13	91.23	98.32	92.29	92.08	92.22
(800,10000)	22.40	0.16	91.43	98.28	92.64	91.67	92.36

5 Discussion and conclusion

The ability to draw with a high level of granularity and accuracy in a fully immersive virtual environment opens new possibilities for a large number of

Table 3. Performance with ELM using two hidden layers with DB_1 (time in second, accuracy in %).

	Time_{train}	Time_{test}	Acc.	Acc. (top5)	Mean	Max	Maj
(400,6000)	27.37	0.50	94.77	98.99	95.47	95.12	95.59
(400,8000)	35.14	0.70	94.87	99.03	95.46	95.18	95.48
(400,10000)	37.56	0.79	94.83	99.00	95.53	95.33	95.58
(600,6000)	42.19	0.57	94.57	98.85	95.68	95.13	95.46
(600,8000)	48.90	0.76	94.83	98.92	95.66	95.03	95.72
(600,10000)	50.34	0.86	94.88	98.95	95.75	95.36	95.68
(800,6000)	61.91	0.64	94.26	98.69	95.66	95.01	95.66
(800,8000)	71.80	0.95	94.43	98.71	95.95	95.35	95.79
(800,10000)	78.02	1.11	94.41	98.67	95.93	95.17	95.95

application areas ranging from entertainment (e.g., video games) to professional (e.g., education) and clinical. The types of elements that can be drawn in virtual reality depend on the layout and composition of the virtual environment, the relationships between the devices in the real world and limitations on what can be rendered in the virtual world. The major challenges of using these types of drawing capabilities effectively are related to both the areas of human-computer interaction and pattern recognition i.e., shapes can be drawn in both 2D as in this study, but also in 3D. However, although virtual reality offers key advantages for managing and interacting with technical documents in this way, there are challenges related to motion sickness in virtual reality, particularly where the user is immersed for long sessions (none of the 18 participants reported motion sickness). In this paper, we have presented a system based in virtual reality that allows a user to draw on a white screen to create and acquire symbols, which are automatically recognized and categorised. We have shown that it is possible to reliably detect and categorise drawings using a deep extreme learning machine approach taking inputs at pixel level with low number of images per class. By further manipulations (rotating images) to the image database, we greatly increased the sample size and performance of the system with scope for additional performance increases by using more geometric deformations.

The current study uses a virtual white board metaphor to represent a typical class room scenario. The accurate detection of the symbols created on the virtual whiteboard could be used to quickly sketch technical documents and to automate the process of producing documents with a logical structure (i.e., with recognized elements). This type of system could be used in an educational setting where the student or lecturer could quickly draw technical documents or designs using symbols that can be automatically recognized and replaced by their actual models, which are easier to read. The current system only allows the recognition of single symbols however it is possible to draw symbols in sequence and validate each symbol with a button press. In this scenario, the user can do this quickly without the need to search through a long list of symbols. Further research will extend the system to include the online recognition of a range of multi-oriented symbols within the virtual reality environment.

References

1. HTC Vive - Lighthouse tracking technology. https://github.com/ValveSoftware/openvr http://doc-ok.org/?p=1478 (2016)
2. Alhamdoosh, M., Wang, D.: Fast decorrelated neural network ensembles with random weights. Information Sciences 264, 104–117 (2014)
3. Arica, N., Yarman-Vural, F.T.: An overview of character recognition focused on off-line handwriting. IEEE Trans. Systems, Man, and Cybernetics, Part C: Applications and Reviews 31(2), 216–233 (May 2001)
4. Assenmacher, I., Hentschel, B., Cheng, N., Kuhlen, T., Christian, B.: Interactive data annotation in virtual environments. In: Proc. of the Eurographics Symposium on Virtual Environments. pp. 119–126. ACM SIGGRAPH (2006)
5. Cecotti, H.: Deep random vector functional link network for handwritten character recognition. In: Proc. of the Int. Joint Conf. Neural Networks (IJCNN). pp. 1–6 (2016)
6. Cecotti, H.: Hierarchical k-nearest neighbor with GPUs and a high performance cluster: Application to handwritten character recognition. International Journal of Pattern Recognition and Artificial Intelligence 31(2), 1–24 (2017)
7. Chen, C.L.P.: A rapid supervised learning neural network for function interpolation and approximation. IEEE Trans on Neural Networks 7(5), 1220–1230 (1996)
8. Chen, C.L.P., Wan, J.Z.: A rapid learning and dynamic stepwise updating algorithm for flat neural networks and the application to time-series prediction. IEEE Trans on Systems, Man, and Cybernetics, Part B 29(1), 62–72 (1999)
9. Chhabra, A.K.: Graphic symbol recognition: An overview. In: LNCS. vol. 1389, pp. 68–79 (1998)
10. Cordella, L.P., Vento, M.: Symbol recognition in documents: a collection of techniques? IJDAR 3(2), 73–88 (2000)
11. Craig, Alan B.and Sherman, W.R.: Understanding Virtual Reality: Interface, Application, and Design. Morgan Kaufmann (2002)
12. Dosch, P., Valveny, E.: Report on the second symbol recognition contest. In: GREC LNCS. vol. 3926, pp. 381–397 (2005)
13. Ghosh, D., Dube, T., Shivaprasad, A.: Script recognition: A review. IEEE Trans. Pattern Analysis and Machine Intelligence 32(12), 2142–2161 (Dec 2010)
14. Huang, G.B., Saratchandran, P., Sundararajan, N.: A generalized growing and pruning RBF (GGAP-RBF) neural network for function approximation. IEEE Trans. on Neural Networks 16(1), 57–67 (2005)
15. Huang, G.B., Siew, C.K.: Extreme learning machine with randomly assigned rbf kernels. Int. J of Information Technology 11(1), 16–24 (2005)
16. Huang, G.B., Zhu, Q.Y., Siew, C.K.: Extreme learning machine: A new learning scheme of feedforward neural networks. In: Proc. of IEEE Int. Joint Conf. on Neural Networks. vol. 2, pp. 985–990 (2004)
17. Igelnik, B., Pao, Y.H.: Stochastic choice of basis functions in adaptive function approximation and the functional-link net. IEEE Trans on Neural Networks 6(6), 1320–1329 (1995)
18. Kasun, L.L.C., Zhou, H., Huang, G.B., M., V.C.: Representational learning with extreme learning machine for big data. IEEE Intelligent Systems 28(6), 31–34 (Dec 2013)
19. Ling, H., Jacobs, D.W.: Shape classification using the inner-distance. IEEE Trans on PAMI 29(2), 286–299 (2007)

20. Liu, C., Nakashima, K., Sako, H., Fujisawa, H.: Handwritten digit recognition: Benchmarking of state-of-the-art techniques. Pattern Recognition 36(10), 2271–2285 (Oct 2003)
21. Nagy, G.: At the frontiers of OCR. Proc. of IEEE 80, 1093–1100 (July 1992)
22. Nagy, N.: Twenty years of document image analysis in PAMI. IEEE Trans. Pattern Analysis and Machine Intelligence 22(1), 38–62 (2000)
23. Otsu, N.: A threshold selection method from gray-level histograms. IEEE Trans. Sys. Man. Cyber. 9(1), 62–66 (1979)
24. Pao, Y.H., Takefuji, Y.: Functional-link net computing: theory, system architecture, and functionalities. Computer 25(5), 76–79 (1992)
25. Plamondon, R., Srihari, N.S.: On-line and off-line handwritten recognition: a comprehensive survey. IEEE Trans. Pattern Analysis and Machine Intelligence 22(1), 63–84 (Jan 2000)
26. Rahman, A., Fairhurst, M.: Multiple classifier decision combination strategies for character recognition: A review. Int. J. of Document Analysis and Recognition (IJDAR) 5(4), 166–194 (2003)
27. Rausch, D., Assenmacher, I., Kuhlen, T.: 3d sketch recognition for interaction in virtual environments. In: Workshop in Virtual Reality Interactions and Physical Simulation (VRIPHYS). The Eurographics Association (2010)
28. Santosh, K.C., Wendling, L.: Graphical Symbol Recognition. John Wiley & Sons, Inc. (2015)
29. Scardapane, S., Wang, D., Panella, M., Uncini, A.: Distributed learning for random vector functional-link networks. Information Sciences 301, 271–284 (2015)
30. Schmidt, W.F., Kraaijveld, M.A., Duin, R.P.W.: Feedforward neural networks with random weights. In: Proc. of 11th IAPR Int. Conf. on Pattern Recog., Conf. B: Pattern Recognition Methodology and Systems. vol. 2, pp. 1–4 (1992)
31. Sezgin, T.M., Davis, R.: HMM-based efficient sketch recognition. In: Intelligent User Interfaces. pp. 281–283 (2005)
32. Sezgin, T.M., Davis, R.: Sketch interpretation using multiscale models of temporal patterns. IEEE Computer Graphics and Applications 27(1), 28–37 (2007)
33. Tang, J., Deng, C., Huang, G.B.: Extreme learning machine for multilayer perceptron. IEEE Trans. Neural Networks and Learning Systems 27(4), 809–21 (2016)
34. Valveny, E., Delalandre, M., Raveaux, R., Lamiroy, B.: Report on the symbol recognition and spotting contest. In: LNCS. vol. 7423, pp. 198–207 (2011)
35. Wang, L.P., Wan, C.R.: Comments on the extreme learning machine. IEEE Trans on Neural Networks 19(8), 1494–1495 (2008)