VizDraw: A Platform to Convert Online Hand-Drawn Graphics into Computer Graphics

A.K. Mishra, J.A. Eichel, P.W. Fieguth and D.A. Clausi

University of Waterloo, Vision and Image Processing Group
akmishra, jaeichel, pfieguth, dclausi@uwaterloo.ca

Abstract. With the adoption of tablet-based data entry devices, there is considerable interest in methods for converting hand-drawn sketches of flow charts, graphs and block diagram into accurate machine interpretations, a conversion process with many applications in engineering, presentations, and simulations. However, the recognition of hand-drawn graphics is a great challenge due to the visual similarity of many system components. This is complicated due to the significant differences in drawing styles between users.
The proposed method, VizDraw, establishes an architecture that utilizes a number of pattern recognition tools to convert hand-drawn diagrams into computer graphics by segmenting the original diagram into individual components. This method generates hypothesis graphs for each component, evaluates the hypotheses using forward and backward dynamic programming, and finally utilizes a rule-based floor planning routine for component and symbol placement. VizDraw is invariant to scaling, rotation, translation and style of drawing. The preliminary results show how VizDraw is used for engineering drawings, simulation, and incorporation into computer aided design (CAD) models.
Index Terms: **Online hand-drawn diagram recognition, hypothesis generation and evaluation, stroke-based recognition**

1 Introduction

Computer-represented flow charts, block diagrams, circuit diagrams, and graphs are typically entered using one of a variety of utilities, such as MapleSim, MATLAB Simulink, Microsoft Visio, Microsft Word and CorelDRAW. Such representations are convenient due to ease of editing and integration into other documents. Furthermore, the computer representation provides a model of the underlying diagram permitting simulation of electrical, mechanical and thermal systems [1]. Unfortunately, the user spends significant time learning how to draw and insert symbols, causing the diagram creation process to be cumbersome and unintuitive [2, 3].

The existing field of document analysis [4–6] is relevant for automated processing of hand-drawn graphics. Graphics recognition applications include the conversion of hand-drawn flow charts, block diagrams and graphs into machine interpretations and printed graphics using on-line recognition of curve based graphic symbols [3, 7].

The online graphic recognition is required to address three issues for successful pattern recognition. First, to resolve ambiguous classifications, the classification of the entire block diagram must be decomposed into subtasks[8], such as the classification of primitive components. For example a component might consist of horizontal or vertical edge. Second, the classes representing diagram components must be properly defined in a representative model [5, 9]. For example the representative model might group primitives into the form of an op-amp or rectangular signal block. Third, the recognition engine must be properly designed to efficiently classify the representation models [10].

Typically, humans identify diagram component symbols from visual features [11, 12]. For example, a straight and curved line can be distinguished by observing the total absolute change in tangent angle. Similarly, a triangle and rectangle can be distinguished by determining the number of edge segments. Complex shapes are usually represented using relational graph models [2, 8]. For example, an op-amp, from an electronic circuit, can consist of a triangular object, multiple resistors, one voltage source, and one ground; the spatial distribution of these symbols can be built into a graphical relation. Consequently, in this manner, graphical models are used to represent diagram components [2, 3, 5]. Further, several graph matching methods are available to compare an instance of the symbol with that of the model; the graphical symbols are matched with an electrical model of an op-amp [2, 7, 8, 13].

Although, existing architectures capable of recognizing diverse engineering symbols do not exist in the current literature, there are several methods for recognizing isolated hand-drawn graphic symbols. These include hidden Markov models [14–18], Bayesian networks [9], neural networks [19], and wavelet networks [20, 21].

In this paper, VizDraw is proposed to convert online hand-drawn text and symbols into machine interpretations and printed graphics. For text recognition, VizDraw uses the existing Microsoft handwriting recognition engine. For online recognition of graphics symbols, however, a more general system is required. First, VizDraw categorizes the symbols into four classes: flow-chart, mechanical, electrical and thermal; these symbol categories provide context to help the recognition engine resolve otherwise ambiguous symbols while reducing the amount of manual user input. Next, VizDraw decomposes the hand-drawn input into diagram components, forming a component graph using relational graphical models at an abstract level. Using the component graph, VizDraw evaluates the likelihood of each symbol by implementing forward and backward dynamic programming. Finally, VizDraw replaces the hand-drawn symbols with a set of computer generated aesthetically pleasing symbols, using a set of user defined preferences that govern the overall look and feel. VizDraw uses advanced pattern recognition and statistical machine learning concepts to reduce dimensionality and to fuse prior and observation probabilities.

The contribution of the paper are:

- Class decomposition: The large number of symbols for all engineering domains increase classification difficultly to the similarity of components across domains. Instead, based on context, the graphic symbols are classified into four mutually exclusive classes: 1) flow chart, 2) electrical, 3) mechanical and 4) thermal symbols using a technique known as confuser analysis.
- Reduction of feature dimensionality by decomposing symbols into component classes: The symbols are decomposed into set of components using relational graph model.

- Hypothesis generation and evaluation: The input graphics are represented as an hypothesis graph of the components. Then a forward and backward dynamic programming algorithm is used to evaluate the hypothesis efficiently.
- Symbol placement: A floor planning route places the recognized symbols in appropriate place by employing center alignment approach.
- Architecture: VizDraw governs the interaction of the aforementioned techniques to convert hand-drawn graphics into computer graphics.

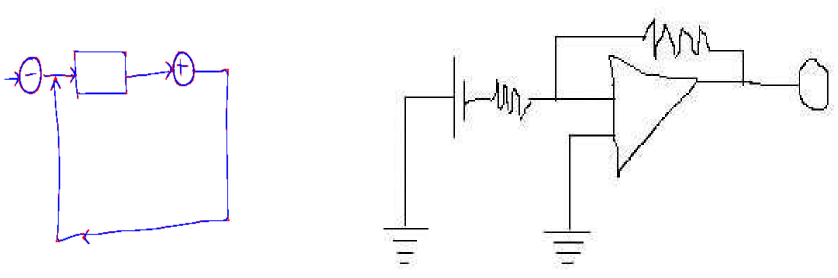
The paper is organized as follows: section 2 describes the proposed method and section 4 discusses real world applications.

2 Proposed Approach

Using the following method, VizDraw is designed to convert hand-drawn graphics into machine interpretations and printed graphics.

2.1 Preprocessing and Interface Design

The user provides data using a hand-held tablet and an electronic pen. The system collects information regarding the pen coordinates and pressure to create Connected Components (CC). Due to the discrete nature of electric pens, the raw pen coordinates contain quantization and human input error, causing the input to resemble a zig-zag (figure 1 red color). To improve smoothness of the pen coordinates, VizDraw approximates the pen-coordinates using a B-spline interpolator.



(a)Control flow diagram (b)An electronic circuit

Fig. 1. Online hand-drawn examples for VizDraw. (a) The flow diagram, in red, is represented by the raw pen coordinates, while blue represents the preprocessed curve using a B-spline interpolator. (b) The user has the ability to input hand-drawn sketches into a previously processed document, which produces a combination of hand-drawn and recognized components of an op-amp.

Searching the entire hand-drawn diagram from left to right and top to bottom, the Connected Components are then identified and enumerated. To provide better recognition results, the user specifies whether VizDraw operates in text recognition mode or graphics recognition mode. Further, the graphic conversion mode is divided into four groups: 1) flow chart, 2) electrical, 3) mechanical and 4) thermal symbols. The system waits for the user to finish drawing, pausing for a short period after each data input to ensure that the user has completed the input. VizDraw initiates the recognition process and displays the most probable symbol, based on context and historical user feedback. The selection of the results is detailed in the section 2.2. If an incomplete symbol is drawn, the system will present the best match and will continue to iteratively refine the matches if the user decides to modify the hand-drawn diagram. Presented with the results, the user can either accept the most likely result or manually select an alternative, which can be used as training data to refine future recognition results. The correct component is then inserted and enumerated as part of the document.

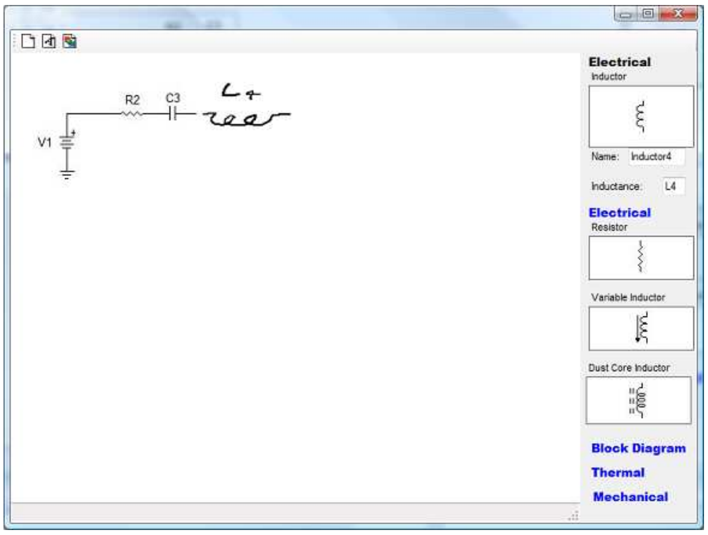


Fig. 2. The above screen shot represents the intended user interface. The figure illustrates the element currently being drawn next to a previously recognized system. The proposed method is in the research stage and is currently implemented in MATLAB, with the intent to integrate the existing code into a stand-alone application using the interface shown above.

2.2 Hypothesis Generation and Evaluation

To determine which component the user is drawing, VizDraw implements a segmentation and segregation algorithm. The algorithm, applied to the ordered Connected Components, generates and evaluates the symbol hypotheses. First, the ordered Connected

Components are segmented into the following multiple stroke sub-components: vertical (y), horizontal (x), and diagonal (xd and yd). Each component is defined to be the segment between a minima and maxima or vise versa.

Let the the function $y = f(x)$ represent the x and y co-ordinates of the Connected Components. Using the following method, VizDraw extracts the maxima and minima of the function $f(x)$ in the vertical, horizontal and diagonal direction. First, the tangent angle of $f(x)$ is defined as:

$$\frac{dy}{dx} = \tan(\theta) \tag{1}$$

The maxima and minima in the vertical (y), horizontal (x), diagonal (xd and yd) are obtained by setting $\theta = \pm 90, \pm 0, \pm 45, \pm 135$. Then, the $x-$, $y-$, $xd-$ and $yd-$ components are formed as a combination of three conjugative extrema or, equivalently, two strokes. VizDraw represents the symbols from a set of features, derived from a set of measurements. VizDraw captures three types of measurements: 1) upstroke (minima-maxima-minima), 2) downstroke (maxima-minima-maxima) and singleton. From these measurements, the following features are extracted: the curvature at smooth regions and corner regions, ratio of the strokes, and total change in tangent angle. When recognizing rectangles, squares, triangles, circles or ellipses, VizDraw uses the features illustrated in figure 3.

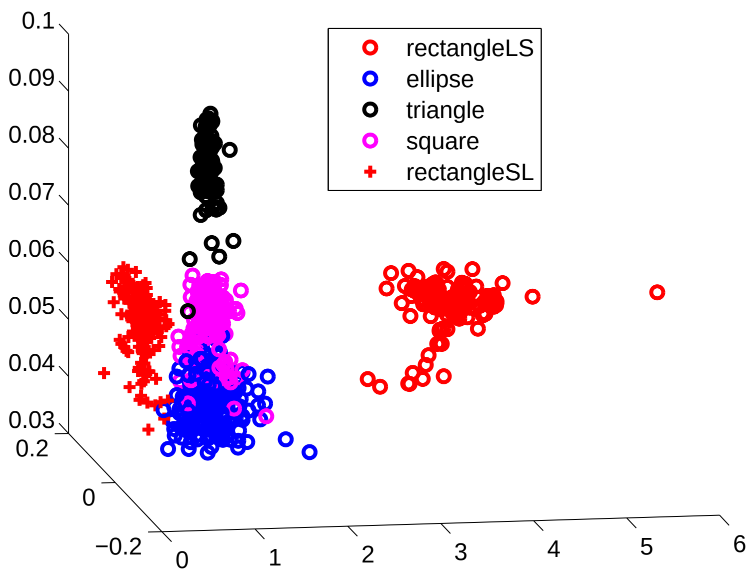


Fig. 3. The left and right figures show an exemplar cluster for five component classes. x-axis, y-axis and z-axis represent the ratio of the arclength for two strokes, the curvature at corner region and the curvature at smooth region.

Figure 3 illustrates an exemplar cluster for symbol classes, where the left and right panel shows the obtained clusters using the ratio of arclength of two strokes, the curvature at corner region and the curvature at smooth region.

In general, the hypothesis graph of the segmented components is generated using a hidden Markov model (HMM). Consider $C_{i,i=\{1,2\cdots u\}}$ and $S_{i,i=\{1,2\cdots n\}}$ that represent the component and symbol models for flow chart database, where S is a collection of components, C. Suppose,

$$L_i \in [C_1, C_2 \cdots C_u]\,, \text{ and, } L_i \in [S_1, S_2 \cdots S_n]\,, \text{ for } i \in [1, k] \tag{2}$$

where L_i is the i^{th} components of the hand-drawn sketch. Given the observation probability of the components ($p(L_i)$), prior probability of the components and symbols ($p(C_i)$ and $P(S_i)$), and the transition probability ($p(L_{i-1}, L_i)$) from L_{i-1} to L_i. The method attempts to find the model component sequence that maximizes the following joint probability:

$$p\left(L_1, L_2 \cdots L_k \middle| \left(C_1, C_2 \cdots C_u\right), \left(S_1, S_2 \cdots S_n\right)\right) \tag{3}$$

To reduce the exhaustive set of $(u + n)q$ possible solutions, sub-optimal methods, such as dynamic programming, can be implemented. The optimization problem is solved using a forward and backward dynamic programming approach. In the forward step of the dynamic programming, VizDraw evaluates the ten most probable path sequences by evaluating:

$$p\left(L_1, L_2 \cdots L_k | C_1, C_2 \cdots C_u\right) \tag{4}$$

In the backward step, the symbols corresponding to each path sequence are generated using the joint probability for each component given the corresponding symbols; the joint probability is obtained in the forward path. Mathematically, the probability is expressed as:

$$p\left(L_1, L_2 \cdots L_k | S_1 = S_{1f}, S_2 = S_{2f} \cdots S_n = S_{nf}\right) \tag{5}$$

where $S_{1f}, S_{2f} \cdots S_{2f}$ are the optimal symbols obtained in the forward path. The symbols corresponding to the most likely path in the backward step are considered to be the optimal symbol sequence. Placing of recognized symbols in appropriate place is a complex data association problem. Instead of using a complex data association approach, VizDraw follows a simple heuristic floor planning routine that aligned the center of the symbols which are in line vertically or horizontally. Floor planning facilitates symbol placement to improve the drawing's aesthetics. The following rules are implemented for layout: the symbols are placed such that the amount of empty space within the bounding box of the diagram should be minimum, the symbols in different rows and columns should aligned, and the drawing should look symmetric. The enumeration of the symbols is preserved during the process. The right panel of figure 5 shows a machine printed drawing of figure 2(a) after the appropriate floor planning algorithm is applied.

3 Experimental results

The performance of VizDraw is evaluated for isolated flow chart symbols and for hand-drawn graphics containing multiple components. Since the focus of this paper is on

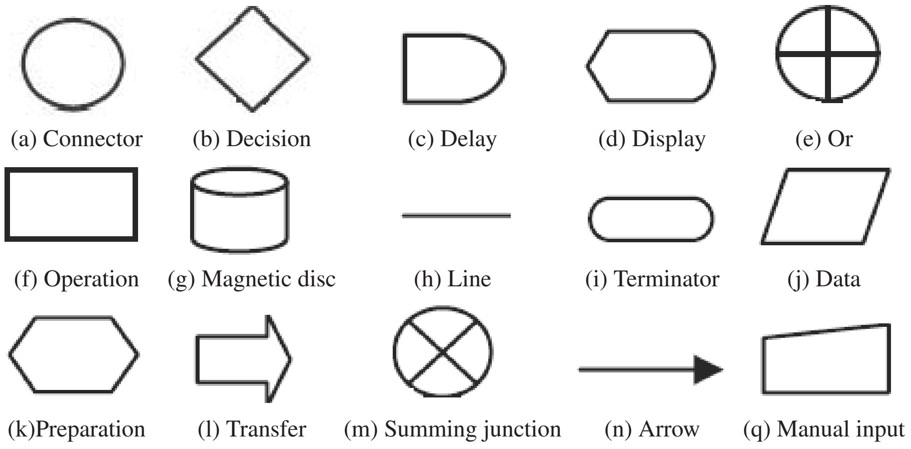


Fig. 4. Flow chart symbol sets. A comprehensive test on these fifteen flow chart symbol sets is conducted to evaluate the performance of the VizDraw.

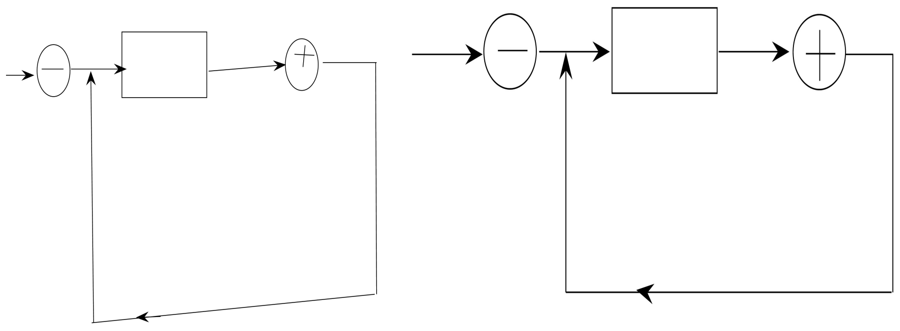


Fig. 5. Recognition performance of VizDraw. The left panel illustrates the recognized symbols of figure 1 without proper placement, whereas the right panel uses a center alignment algorithm to improve aesthetics.

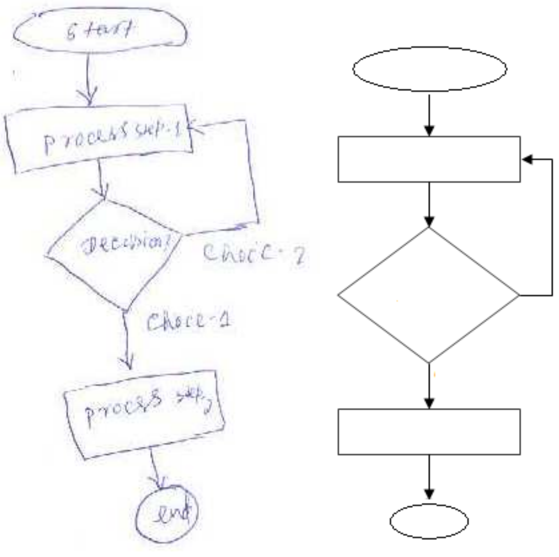


Fig. 6. Recognition performance of VizDraw. The left panel illustrates the hand-drawn flowchart while the right panel shows the machine printed graphics. The text is processed using Microsoft character recognizer and is not included as direct output of the symbol recognition engine, shown in the right panel.

user interactive conversion of hand-drawn graphics into computer graphics, extensive testing on isolated symbols has not yet been conducted. However, the performance of VizDraw on a limited data set has been evaluated. The set consists of 3000 mutually exclusive flow chart symbols that correspond to 15 classes of flow chart symbols (the classes are shown in figure 4). Initially a user was asked to draw a flow chart symbol

Table 1. Average percentage classification accuracy of proposed method compared to other two methods across five flow chart symbols.

Symbol	VizDraw	SVM-HMM hybrid approach [17]	Traditional HMM approach
operation	98.9	97.2	96.4
decision	99.2	96.3	95.2
I/O	98.7	96.3	94.8
connector	97.5	92.1	90.3
termination	99.0	93.2	91.4

for each of the 15 classes. Then, random noise and affine distortions were applied to these 15 flow chart symbol classes to generate a total of 3000 symbols. The 3000 flow chart symbols were divided into training and testing sets of 1000 and 2000 symbols, respectively, For the testing data set, the average classification accuracy of VizDraw is found to be 98.7%, while the classification accuracy of SimuSketch [15] is found to be 94% when these symbols were used. Further the performance of VizDraw is compared with SVM-HMM hybrid approach [17] and traditional HMM approach on a limited data set consisting of five flowchart symbols. The performance comparison of VizDraw with SVM-HMM hybrid [17] and traditional HMM is demonstrated in table 1.

Some preliminary testing are performed on other three symbol sets, but a comprehensive testing on these symbols will be carried out in future. Nevertheless, the preliminary overall performance of VizDraw in converting hand drawn graphics into computer graphics is demonstrated in figures 5 and 6.

4 Discussion and conclusion

By using context information and relational graph models, VizDraw can provide an architecture capable of converting hand-drawings into computer graphics. These digital diagrams and system representations can then be used in existing simulation software, allowing the user to naturally input system diagrams without having to manually search through libraries of stock components. VizDraw can also be expanded to accommodate digital white boards. The online system can improve the aesthetics during live presentations and, based on the recognized components, can interface with third-party software that might present the transfer function of individual or grouped components. This method can be used in engineering drawing to convert hand-drawn sketch into CAD model for storage and future retrieval. VizDraw can provide the ease of use lacking in many existing applications.

The general architecture of VizDraw can also be used for evaluation of other pattern recognition techniques. Using a hierarchial recognition system, VizDraw, can iteratively classify complicated components by decomposing them into groups of simple primitives. The primitives can be defined parametrically or non-parametrically depending on which definition is best to identify a particular class.

Challenges in developing VizDraw to robustly convert hand-drawn graphics into computer graphics without user interface are: 1) the drawing variations from user to user; 2) the large number of classes and similarity between symbols, humans cannot distinguish some symbols without context; and 3) the fusion of context, prior knowledge, and observation information.

Future research, should allow VizDraw to be less user dependent and should incorporate the ability to convert both online (tablet based) and offline (scanner based) hand-drawn graphics into computer graphics into the existing functionality of VizDraw.

5 Acknowledgement

This research has been sponsored by the Natural Sciences and Engineering Research Council (NSERC) of Canada through individual Discovery Grants as well as GEOIDE (GEOmatics for Informed Decisions) which is a Network of Centres of Excellence under NSERC.

References

1. Groen, F., Sanderson, A., Schlag, J.: Symbol recognition in electrical diagrams using probabilistic graph matching. Pattern Recognition Letter **3** (1985) 343–350
2. Lladoós, J., Martí, E., Villanueva, J.J.: Symbol recognition by error-tolerant subgraph matching between region adjacency graphs. IEEE Transaction on Pattern Analysis and Machine Intelligence **23**(10) (2001) 1137–1143
3. Huang, T.: Mathematical models of graphics. **12**(2) (February 1980) 127–135
4. Fletcher, L., Kasturi, R.: A robust algorithm for text string separation from mixed text/graphics images. IEEE Transaction on Pattern Analysis and Machine Intelligence **10**(6) (November 1988) 910–918
5. Chan, K., Yeung, D.: Mathematical expression recognition: A survey. **3**(1) (2000) 3–15
6. Lin, X., Shimotsuji, S., Minoh, M., Sakai, T.: Efficient diagram understanding with characteristic pattern detection. Computer Vision Graphics and Image Understanding **30**(1) (April 1985) 84–106
7. Hse, H., Newton, A.: Sketched symbol recognition using zernike moments. (2004) I: 367–370
8. Llados, J., Sanchez, G.: Symbol recognition by error-tolerant subgraph matching between region adjacency graphs. IEEE International Conference on Image Processing **2**(10) (2003) II – 49–52
9. Kasturi, R., Bow, S., El Masri, W., Shah, J., Gattiker, J., Mokate, U.: A system for interpretation of line drawings. IEEE Transaction on Pattern Analysis and Machine Intelligence **12**(10) (October 1990) 978–992
10. Minoh, M., Munetsugu, T., Ikeda, K.: Extraction and classification of graphical symbol candidates based on perceptual organization. (1992) II:234–237
11. Bottoni, P., Cugini, U., Mussio, P., Papetti, C., Protti, M.: A system for form-feature-based interpretation of technical drawings. **8**(5) (1995) 326–335
12. Murase, H., Wakahara, T.: Online hand-sketched figure recognition. Pattern Recognition **19**(2) (1986) 147–160
13. Llados, J., Sanchez, G.: Symbol recognition using graphs. (2003) II: 49–52
14. Huang, B., Kechadi, M.: An hmm-snn method for online handwriting symbol recognition. (2006) II: 897–905
15. Kara, L.B., Stahovich, T.F.: Hierarchical parsing and recognition of hand-sketched diagrams. In: SIGGRAPH '07: ACM SIGGRAPH 2007 courses, New York, NY, USA, ACM (2007) 17
16. S, T.M., Davis, R.: Hmm-based efficient sketch recognition. In: IUI '05: Proceedings of the 10th international conference on Intelligent user interfaces, New York, NY, USA, ACM (2005) 281–283
17. Yuan, Z., Pan, H., Zhang, L.: A novel pen-based flowchart recognition system for programming teaching. (2009) 55–64
18. Taxt, T., Olafsdottir, J., Daehlen, M.: Recognition of handwritten symbols. Pattern Recognition **23**(11) (1990) 1155–1166
19. Seda, P., Yasar, B.: Wavelet networks for nonlinear system modeling. Neural Computing and Applications **16**(4-5) (May 2000)
20. Pradhan, A., Routray, A., Behera, A.: Power quality disturbance classification employing modular wavelet network. Power Engineering Society General Meeting, 2006. IEEE (June 2006) 2006
21. P., S., B., Y.: Gradient-based polyhedral segmentation for range images. PRL **24**(12) (August 2003) 2069–2077