

# Sketch-based User Interface for Creative Tasks

Xiaogang Xu<sup>a</sup>, Wenyin Liu<sup>b</sup>, Xiangyu Jin<sup>a</sup> and Zhengxing Sun<sup>a</sup>

<sup>a</sup>State Key Lab for Novel Software Technology Nanjing University,  
Nanjing 210095, PR China

<sup>b</sup>Department of Computer Science, City University of Hong Kong,  
Hong Kong, PR China

[xuxg@graphics.nju.edu.cn](mailto:xuxg@graphics.nju.edu.cn), [csluwy@cityu.edu.hk](mailto:csluwy@cityu.edu.hk)

## ABSTRACT

Designers often need to write down their improvisatory ideas. In that process, they concentrate on their creative ideas instead of symbols they use to deliver their thoughts. However, most of the current graphics constructing tools require users to input graphic components using mouse/keyboard with lots of toolbar buttons or menu items for selection. This inconvenient, formal, and unnatural user interface is not suitable for creative tasks, especially not applicable on handheld devices with small screens and no keyboards. Sketch-based User Interface offers a good solution to this problem. This article puts forward a sketch-based user interface for schematic/conceptual design tasks. Our work focuses on the three merits: *humanistic*, *intelligent*, and *individualized*. The three merits are combined harmoniously in the prototype system—Smart Sketchpad, which is designed for schematic/conceptual design tasks. More than 300 pre-defined graphic objects (which can be easily extended and modified) are installed in the system. Both the traditional menu/toolbar-button-based user interface and the sketch-based user interface are integrated in this system, for the users' convenience to compare the usability between them. As a result, most users unanimously prefer the sketch-based user interface.

**Keywords:** Sketch-based User Interface, On-line Graphics Recognition, Schematic/Conceptual Design, User Adaptation

## 1. INTRODUCTION

### 1.1. A Revolution is Undertaking

Computer equipments are more easily available today. People expect computers to assist in human-oriented tasks, such as drawing, writing, designing, and so on. This is causing some user-interface (UI) researchers to reconsider the traditional reliance on approaches that are more machine-oriented and to look at ways to support humanistic properties, such as ambiguity, creativity, and informal communication. The idea is to bend computers to people's way of interacting, not the other way around (Landay 2001). Hence, a new revolution is undertaking. The central idea of this revolution is to make computers more *humanistic*, more *intelligent*, and more *individualized*. To make a computer humanistic is the ultimate goal of human-computer interface design, which makes the computers easily to be used, and makes them more adaptable to human-preferred communication ways. This revolution, including classical issues of information science, is impelled by newly arising inter-discipline research areas, such as human-computer interaction (HCI) (Dix 1993), which covers various of topics ranging from computer science to human science. The word "intelligent" involves the supporting techniques for such humanistic interface, such as Artificial Intelligence. The word "individualized" revealed that future computers are

user-specific. This requires the computers to automatically adapt to the user's preference.

### 1.2. Why sketch-based UI for Creative Tasks?

Most current computer-assisted design tools, including Microsoft Office, PhotoDraw, Visio, and many CAD systems, have casted away traditional paper/pen-based design interface, but bend users to computers' way of interacting. These systems ask users to input graphic components by selecting lots of toolbar buttons or menu items. This is not a harmonious combination of traditional design approaches and modern electronic equipments for the following reasons.

#### (i). It is an inconvenient user interface.

Users frequently find that it is inconvenient to draw their intended shapes via too many mouse clicks. Moreover, if there are many predefined objects/shapes, it is hard to remember where to select the intended one. For instance, in Microsoft Visio, there are nearly 10,000 composite shapes, which are also known as master objects, listed in many stencils for users to select.

#### (ii). It is an unnatural user interface.

To implement some creative tasks, such as schematic designs, users prefer to rapidly deliver their design ideas. However, the menu/mouse way requires the users to spend more time on operation than their

thoughts and hence frequently interrupts their thoughts and inhibits their creativity.

**(iii). It is unsuitable for small screen devices.**

It is unrealistic to do such design works on small screen devices, such as Personal Digital Assistant (PDA), which solely depend on pen-based user-interface and have no keyboard, for there are no enough room to accommodate so many toolbar buttons or menu items.

More and more situations require people to write down their bursting ideas omnivorously. This requires techniques that support a more convenient and natural user interface to do such creative works on mobile devices. The key to the success is to provide the design tools with the ability to recognize graphical elements common to a particular domain while the designer draws them *by sketching*.

Using sketches to deliver information is not a new idea. In fact, communication through sketches precedes the invention of writing by more than 30 centuries (Harley 1987). Sketching with a pen is a mode of informal, perceptual interaction that has been shown to be especially valuable for creative design tasks (Gross 1996). We believe, as other authors (Gross 1996, Landay 1995), paper and pencils have still to be designers' preferred choice to quickly sketch bursting ideas. For designers, the ability of sketching objects with uncertain types, sizes, shapes, and positions rapidly is important to the creative process. This uncertainty, or ambiguity, encourages the designer to explore more ideas without being burdened by concern for inappropriate details such as colors, fonts, and precise alignment. Leaving a sketch uninterrupted, or at least in its rough state, is key to preserving this fluidity (Hearst 1998). In this early phase, ambiguity also improves communication, both with collaborators and with the target audiences of the designed artifact. For example, a reader examining a sketch design will be inclined to focus on the important issues at this early stage, such as the overall structure and flow of the interaction, while not being distracted by the details of the look (Rettig 1994, Wong 1992). Interactive sketching-rectification user interface is referred as *Sketch-based User Interface* in this article, which exploits the convenience of pen-based sketching and the accuracy of digital document.

### **1.3 What Kind of Sketch-based UI Are We Interested In?**

There are varieties of topics concerning sketch-based user interface study. Considering the way to input sketch, we have on-line and off-line sketching. Considering the

application level of sketching, we have three different levels. In the following, we will present them in details.

#### **1.3.1. On-line vs. off-line**

There are two ways to input sketch, on-line and off-line. The on-line case deals with a spatio-temporal representation of the input, whereas the off-line case involves analysis of the spatio-luminance of an image. On-line sketching is an interactive process, whereas the off-line is a one-step process, that is, non-interactive process. The off-line sketch is inputted as an image, and analyzed to extract user drawn components. The on-line sketch is drawn by a pen directly on a screen or tablet. Isochronously, the system will give dynamic interpretations for the user sketch and formalize it, thus the user can adjust the result accordingly.

From the point of view of pattern recognition, on-line sketch recognition is different from off-line sketch recognition. First, off-line sketch is usually drawn in a more formal situation. However, on-line sketch is usually directly drawn without rules or other drawing assistant tools, its strokes are much freer. Second, for off-line case, entities are difficult to be separated from each other, for no information other than raster can be utilized. While for on-line case, strokes, which belong to one entity, can be easily separated from others for usually they are consecutive and adjacent. Finally, for on-line systems, the pen's moving velocity can be extracted, which can be a supplement to stroke segmentation and shape recognition. But there are no means to extract velocity information for off-line systems. These characteristics discussed above make the on-line recognition process quite different from the off-line recognition process.

#### **1.3.2. Three application levels**

There are three different application levels for sketch-based user interface (UI), according to the inputting patterns and stroke orders.

The first level is applications, such as handwriting recognition. Characters for a certain culture are fixed and pre-defined and how to input them is well known. The stroke sequence of characters is also fixed to some extent. Domain specific diagram constructing tools are the second level applications. These applications require users to have a common knowledge background. But there is no requirement for the fixed stroke inputting order. Users can sketch the fixed graphic pattern using his/her preferred way, with free order. This makes the graphic recognition much more difficult than handwriting recognition (Plamondon 2000). The third level is more general than the first two kinds of application. It is a more difficult one (Landy 2001). Different user has different comprehension for a same concept. Even if they share the

same comprehension, the stroke orders vary absolutely. Sketch-based image retrieval is a case in point. For a concept like “lion on grassland”, different users have different sketches. Even the same user may change his/her sketch from time to time.

### 1.3.3. Our research focus

From the previous discussion, in order to acquire an interactive, general, and informal sketch-based user interface, we focus our research topic to be on-line graphics recognition, and constrain our application for conceptual/schematic design, which has fixed pattern and free stroke order.

## 2. THREE DESIGNING PRICIPLES: HUMANISTIC, INTELLIGENT, AND INDIVIDULIZED

### 2.1. Make the UI Humanistic?

Since the traditional menu/toolbar button-based graphics-inputting user interface is clumsy, unnatural, and inefficient, the problem for us is, with the support of on-line graphics recognition techniques, how to design an efficient, humanistic and natural sketch-based graphics inputting interface for conceptual/schematic design tasks.

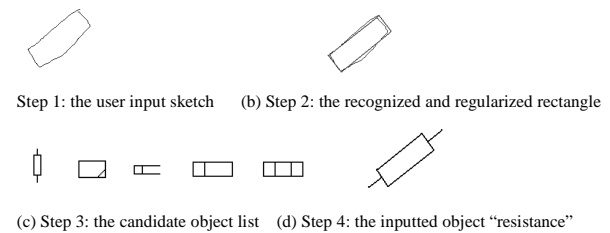
In traditional graphics-inputting process, the input of graphic object patterns and their parameters are separated. If we can combine the pattern inputting and the following parameter adjustment into one harmonious process, the interaction time will be reduced and the user will feel a much smoother interface.

When drawing a pre-defined graphic object, the user tends to divide the graphic objects into several primitive shapes. And he/she tends to input one primitive shape in a single-stroke or in several consecutive strokes. In our graphics recognition approach, we first discover latent primitive shapes among user strokes, then recognize and regularize this primitive shape, and finally show the regularized drawing on the screen immediately. Users can then adjust the recognition and regularization results by the suggestion of the system. This immediate feedback strategy makes the user interaction smoother and humanistic. And it has an extra advantage to reduce inner-stroke and inter-stroke noises, which are generated due to non-proficiency or un-professionalism of the user.

After being recognized and regularized, primitive shapes, which belong to one graphic object, are grouped together. Then they are segmented and combined to form an object skeleton. Partial structural similarities are calculated between the user drawn graphic object and the candidate ones in the database. The graphic objects that are the most similar to what the user has drawn are

suggested to the user in a ranked list for the user to choose and confirm.

Figure 2.1 describes such an interactive process of inputting a graphic object named “*resistance*” by sketch. In order to avoid too much intrusive suggestions of composite shapes of low similarity, only those candidates whose similarities are above a threshold will be displayed for suggestions. Obviously, this user scenario is much more reasonable and easier to be used than the traditional one, especially on small screen devices.

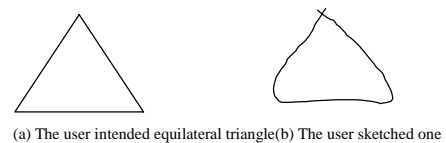


**Figure 2.1** The Proposed User Scenario for Graphics Inputting

One advantage of interactive sketching is the ability to quickly edit them (Marti 1998). Some editing gestures, such as deleting, redoing, undoing, grouping, etc., should also be provided.

### 2.2. How to Make the UI Intelligent?

Sketch is inherently imprecise, ill-formed and ambiguous. Due to non-proficiency or un-professionalism, the user drawn sketch may be quite different from the user’s original intention. Take Figure 2.2 as an example, the original intention of a user is to input an equilateral triangle, cf. Figure 2.2(a). But the one drawn by the user is quite different, cf. Figure 2.2(b).



**Figure 2.2** The User Intended Object might be Different from The Sketched One

Although the designer may not have difficulty to browse his/her own sketches later, other audiences may misunderstand the designer’s purpose. Hence, it is very important to recognize and regularize the user sketch to be a generally acceptable one. The graphics recognition task not only includes the task of recognition of the user sketch, but also includes the restoration of the original intention. User intention prediction is critical to sketch-based user interface.

The on-line graphics recognition is divided into two stages, which are referred to as Primitive Shape Classification and Regularization and Composite Graphic Object Recognition, respectively. We use four sub-processes, including preprocessing, shape classification, shape fitting, and shape regularization to recognize user's original graphic component. The whole process is described in (Sun 2003).

After primitive shape recognition, we get regularized stroke segments or objects, which are regarded as parts of a composite object. How to make computer intelligently predict the composite object from these stroke segments/objects is most important. This prediction should have the following three characteristics.

**(i). Partial**

In order to predict the user intended object in an early stage, it is even better to recognize the Source Object in an incomplete form. Hence, the similarity calculation for prediction is asymmetric between the Source and Candidate objects.

**(ii). Structural**

Currently, most shape similarity assessment strategies only take the contour of the object into consideration, while the inner structure of the object is discarded. However, inner structure usually plays an important role for human to identify symbols. Moreover, we propose that a reasonable shape similarity assessment strategy should not only satisfy the three traditional invariabilities (Shifting Invariability, Rotation Invariability, and Mirroring Invariability) but also should satisfy *Structural Invariability*.

**(iii). Stroke-number and stroke-order free**

However, different users may have different opinions in decomposing a composite object into primitive shapes. Even the same user may change his ideas from time to time. The ambiguity makes it even difficult for us to compare shapes stroke-to-stroke. We suppose the prediction to be invariant to stroke-number and stroke-orders.

We obtain spatial relations between graphic primitives in pairs, thus to convert the representation of Composite Graphic Objects into *Spatial Relation Graphs* (SRGs). We also propose *Conditioned Partial Permutation* to cut down the computational cost of computing the isomorphism of two SRGs.

### 2.3. How to Make the UI Individualized?

User adaptation is a classical problem in user interface study. Many pattern recognition problems are user

specific, since users' handwritings, drawing styles, and accents are different from user to user. As on-line graphics recognition is concerned, the input sketches for the same shape may be quite different from user to user and even from time to time. A system with high general performance may still yield poor results for a specific user. There are two approaches to solve such problems, rule based approaches or statistic machine learning based approaches.

For rule-based approaches, some parameters are available to be adjusted through user feedback. Rule-based relevance feedback is widely used in the research area of information retrieval and content-based image retrieval (CBIR). Unfortunately, although user feedback can adapt the system to a new user, the system may lost its universality to other users. Rule based approaches are not powerful enough to reflect very complex relations due to their intrinsic limitations such as scalability and adaptability. Moreover, intuitive feedback strategy usually results in ad hoc situations, which gains "personalized" performance on the expense of the loss of its universality. Hence, more complex, completed, statistical approaches such as Neural Network (NN) (Yaeger 1996) and Support Vector Machine (SVM) (Vapnik 1995) are required.

For statistic machine learning based approaches, to adapt a user's preference means to retrain the recognizer/classifier with training samples obtained for this particular user. Since the newly introduced sample set is usually much smaller than the previously accumulated one and retraining using all these samples is not economic. In this case, incremental learning, which mainly involves the newly obtained samples in retraining and avoids full usage of all previous samples, is more preferable.

We constructed multi-class incremental classifiers based on one-against-one structure and one-against-all structure employing two SVM-based incremental learning algorithms, Syed et al.'s (Syed 1999) and Xiao et al.'s (Xiao 2000), respectively, for our shape classification problems. As a result of performance evaluation, both incremental learning algorithms can save much of the retraining time, while little classification precision is lost. Unlike rule-based approach, both algorithms' precision curve will not conflict among different users. Finally, one-against-one structure is more adaptive to the multi-class classification environment than one-against-all structure and is therefore employed in the Smart Sketchpad system.

### 3. THE SMART SKETCHPAD & ITS UI EVALUATION

#### 3.1. The Smart Sketchpad System

Using Microsoft Visual C++ 6.0, we have developed a prototype system “Smart Sketchpad” (Liu 2001), which is running on Microsoft Windows 2000 with a Wacom LCD digitizing tablet. This system has integrated two user interfaces. One is traditional toolbar button-based user interface, and the other is sketch-based user interface. These two interfaces can be easily switched to each other. The components of the system and their functions are depicted as Figure 3.1.

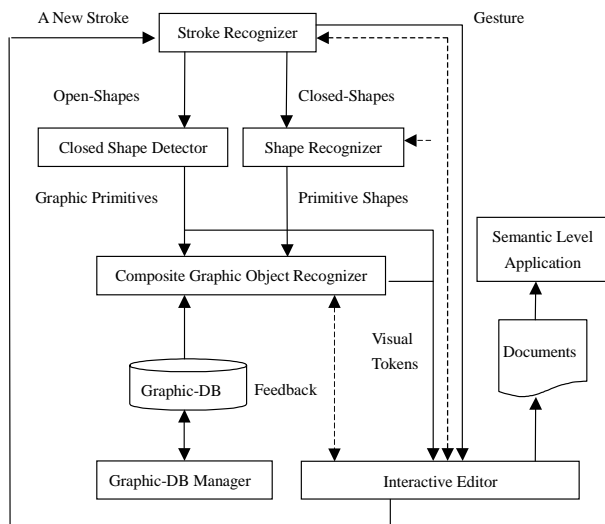


Figure 3.1 System Architecture of Smart Sketchpad

#### 3.2. The Sketch-based User Interface of Smart Sketchpad

In the user interface of Smart Sketchpad (cf. Figure 3.2), users can draw sketches in the sketching area. Each user stroke, combined with its neighboring strokes if necessary, will be automatically classified and then regularized to the most similar primitive shape. The corresponding message will be shown in the Smart Toolbox, including a precision bar showing a confidence of this judgment and a candidate shape which is the next similar to the sketched one. The primitive shape type that the system can recognize includes triangle, quadrangle, pentagon, hexagon, and ellipse.

In order to avoid too much intrusive suggestion of composite shapes of low similarity, only the first six objects whose similarities are above a threshold are displayed in the smart toolbox for suggestions. Only

those candidate objects which pass the spatial relation check can be kept for further comparison, thus to relieve the computational burden. The whole progress in which the user inputs graphic object named “diode” is illustrated in Table 3.1.

Table 3.1 The Process of Inputting a Graphic Object “Diode”

Input Steps	The input sketch	Similarity	Rank	Candidate objects in the smart toolbox arranged according to similarities in descending order
Step 1		0.41	19	< T / Y < A
Step 2		0.52	8	T A U <
Step 3		0.79	3	Y A A A
Step 4		0.84	1	A A A A A

When the intended object appears in the smart toolbox. The user can click the object to replace these strokes by the intended object with proper size, direction, and position. Some editing gestures, such as Delete, Redo, Undo, are also recognized.

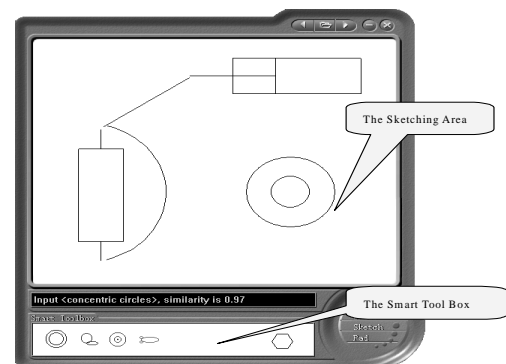


Figure 3.2 The User Interface of Smart Sketchpad

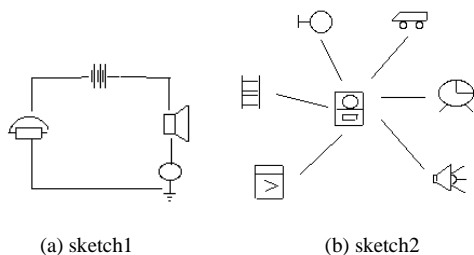
However, for the imperfectness of the similarity calculation strategy, during the drawing processes, irrelevant objects may obtain higher ranks in the object list. In order to avoid such situation, relevance feedback is employed to improve both input efficiency and accuracy. Because each user has his/her drawing style/preference, a specific user tends to input the same object through a list of components of fixed constitution and fixed input time sequence. Each time when the user drags an object from the smart toolbox, the currently grouped input primitive shape list will be saved as a “word” index for this intended object with a probability.

E.g., a list rectangle-triangle may be used for index of “envelope” and “arrow box”. The probability can be easily obtained from the relative times when the intended object is selected for this “word”. Hence, for an inputted source object and a candidate object in the database, we can not only acquire their shape similarity by partial structural similarity calculation strategy, but can also acquire the probability (initially, all possibilities are set to 0) that the candidate object is just the intend one. We linearly combine them with different weights to obtain a new metric for similarity comparison.

### 3.3. User Interface Evaluation

In this experiment, we evaluate the usability of the two integrated user interfaces of Smart Sketchpad, traditional toolbar based approach and our proposed approach with on-line graphics recognition. In order to simulate a real PDA application, we restrict the drawing area of the Smart Sketchpad to be  $480 \times 340$  pixels and the area of Smart Toolbox to be  $520 \times 45$  pixels.

In order to avoid subjectivity of single experimentalist, we ask 10 different subjects in this UI evaluation, including 8 graduate students and 2 undergraduate students. All these subjects are with average knowledge of using computers. We asked them to draw the following two sketches (cf. Figure 3.3) with the two different UI and then asked them to evaluate the two user interfaces.



**Figure 3.3** Sketches for The User Interface Evaluation

For the first feeling, all of the ten users believe sketch-based UI is easier to be used than traditional toolbar-button-based one. They all find it inconvenient to find graphic objects among so many stencils for traditional toolbar button-based UI. Users complain that sometimes they may “miss” the intended object from the stencils. They also admit that on-line graphic recognition not only provides high efficiency, but also provide continuous ideas without being interrupted by concerns for precise alignment and other things. In the experiment of drawing the two given sketches, the sketch-based UI is

faster than the traditional one. Averagely, the sketch-based UI is 22.4% and 42.9% more efficient than the traditional toolbar button-based UI for sketch1 and sketch2, respectively. This shows that when the graph becomes more complex, the sketch-based UI is more preferred. The detail information is shown in Table 3.2. Moreover, for the evaluators’ convenience, we show 12 objects each time for the traditional UI. If the candidate area is very small and only can accommodate 6 objects, the difference between these two UI will be enlarged.

**Table 3.2** Drawing Time for Different Sketches Using Different UIs

#User ID	Drawing Time for Sketch 1 (s)		Drawing Time for Sketch 2 (s)	
	Sketch-based	Traditional	Sketch-based	Traditional
1	104	125	157	190
2	93	99	151	288
3	69	98	156	294
4	59	81	122	156
5	64	178	135	191
6	63	100	85	231
7	61	120	92	203
8	72	91	119	195
9	70	70	156	252
10	78	110	125	201

Admittedly, the on-line graphics recognition result of Smart Sketchpad is still not satisfying. On one hand, the composite graphic object recognition approach is based on graphic primitives. This requires a precise segmentation of strokes to graphic primitives. However, the stroke segmentation strategy we employed in the system is quite naïve and performs poorly. We plan to solve this problem in future works. On the other hand, each edge’s direction has severe influence on the recognition results. This makes our approach very sensitive to the direction of the edges. Another reason is that the users are not familiar with the pen/tablet device. We find that drawing with a pen/tablet device is not like drawing on a real sheet of paper, for users cannot see anything on the tablet while he/she is drawing. Users need to look at the screen of the computer while drawing by their hands in our experimental environment. This uneasy job requires harmonious coordination of the user’s hand and eyes. A new user does need some training to acquire such skill. Nevertheless, we are sure this problem can be overcome when the user directly draws on the screen, as is similar to most handled devices.

## 4. SUMMARY

Current human-computer interface design has been converting from computer-oriented to human-oriented, and becoming more humanistic, intelligent, and individualized. Here, the word “humanistic” represents our ultimate goal of human-computer interface design.

“Intelligent” is actually a supporting technique, such as artificial intelligence, behind this goal. “Individualized” means the future human-computer interface should be user-specific.

As we have discussed above, traditional computer interfaces are too formal and precise for creative tasks, such as conceptual/schematic design. For these brainstorming tasks, people often sketch their thoughts informally on paper and whiteboards. After sketches are done on paper they must be transferred to the computer for further processing. Why not use a computer for the sketching process itself? As we have illustrated in Section 1, there is an inherent contradiction in today’s sketching programs—users select graphical primitives from a palette, resulting in clean, precise-looking diagrams despite the fact that only a sketch was intended.

This article has analyzed the above problems, and proposed a sketch-based user interface for conceptual/schematic design tasks. A prototype system, Smart Sketchpad, is also implemented for user interface evaluation. Some relevant techniques, such as on-line graphics recognition and user adaptation, are further studied for this purpose.

### ACKNOWLEDGMENTS

Part of this work is supported by National Natural Science Foundation of China (69903006).

### REFERENCES

1. [Dix 1993] Dix AJ et al. (eds.), "Human-Computer Interaction", Prentice Hall, 1993.
2. [Gross 1996] Gross MD, "The Electronic Cocktail Napkin - a Computational Environments for Working with Design Diagrams", Design Studies, Vol. 17, pp. 53-69, 1996.
3. [Harley 1987] Harlay JB and Woodward D, "The History of Cartography", Vol.1, Univ. of Chicago Press, 1987.
4. [Hearst 1998] Hearst MA, "Sketching Intelligent Systems", IEEE Intelligent Systems, Vol.13, No.3, pp.10-19, 1998.
5. [Landay 1995] Landay J and Myers B, "Interactive Sketching for the Early Stages of User Interface", Human Factors in Comp. Syst. Conf., ACM Press, pp. 43-50, 1995.
6. [Landay 2001] Landay JA and Myers BA, "Sketching Interfaces: Toward more human interface design", IEEE Computer, Vol. 34, No.3, pp. 56-64, 2001.
7. [Liu 2001] Liu W, Qian W, Xiao R, and Jin X, "Smart Sketchpad-An On-line Graphics Recognition System", Proc. of ICDAR2001, Seattle, 2001, pp. 1050-1054.
8. [Marti 1998] Marti A. Hearst, "Sketching intelligent systems", IEEE Intelligent Systems, Vol.13, NO.3,pp.10-19,1998
9. [Plamondon 2000] Plamondon R and Srihari NS, "On-line and Off-line Handwriting Recognition: A comprehensive Survey", IEEE Transaction on Pattern Analysis and Machine Intelligence, Vol.22, No.1, 2000.
10. [Rettig 1994] Rettig M, "Prototyping for Tiny Fingers," CACM, Vol. 34, No. 4, pp. 21-27, 1994.
11. [Sun 2003] Sun J et al., "A method of Fast On-line Graphics Recognition and Regularization", to appear in *Computer Science*, No.2, 2003
12. [Syed 1999] Syed N, Liu H, and Sung KK, "Incremental Learning with Support Vector Machines", Proc. Workshop on Support Vector Machines at the International Joint Conference on Artificial Intelligence (IJCAI-99), Stockholm, Sweden, 1999.
13. [Vapnik 1995] Vapnik V, "The Nature of Statistical Learning Theory", Springer-Verlag, 1995.
14. [Wong 1992] Wong YY, "Rough and Ready Prototypes: Lessons from Graphic Design," Posters and Short Talks: Proc. Human Factors in Computing Systems, ACM Press, New York, pp. 83-84, 1992.
15. [Xiao 2000] Xiao R, Wang J, and Zhang F, "An Approach to Incremental SVM learning algorism", Proc. the 12th International Conference on Tools with Artificial Intelligence, pp.268-273, 2000.
16. [Yaeger 1996] Yaeger L, "Neural Networks Provide Robust Character Recognition for Newton PDAs", IEEE Expert - Intelligent Systems and Their Applications, Vol.11, No.4, pp.10-11, 1996.