Pattern Recognition and Beautification for a Pen Based Interface

Luc JULIA and Claudie FAURE

ENST dpt SIG - CNRS URA 820
46, rue Barrault - 75634 PARIS Cedex 13 - FRANCE
Tel.: (+33-1) 45.81.77.84 - Fax: (+33-1) 45.88.79.35

julia@sig.enst.fr cfaure@sig.enst.fr

Abstract

This paper presents the algorithms for recognition and beautification which are used in incremental graphic design applications. These applications propose multimodal interfaces integrating handwriting, gesture, and speech on a pen-computer. User and computer collaborate to perform the task of incrementally designing a drawing. Processing and data representation take into account the variable quality of handwritten data, the man-machine interaction context and the cooperation between the user and the interpretation system. Both recognition algorithms may be used in combination in order to increase the speed and the set of recognized figures. Local recognition is followed by the beautification of the global structure in order to detect alignments and logical structures. The beautification enables the user to display a clean version of the original draft. The applications which we developed are used to recognize tables, gestures, geometrical figures or diagram networks.

1: Introduction

Through pen-computers, a new field of application has arisen in the human-machine interaction domain. Pen-computers allow users to recreate the paper-pen situation where they are able to quickly express visual ideas. However, to successfully attain cooperation between a user and his machine, the interpretation of on-line data has to be accurate, involving high quality pattern recognition processes, fast visual feedback and some knowledge on the domain of the application.

The work presented here is part of a project concerning the use of multimodal interfaces for incremental design of graphical documents. Incremental design means that the user can easily modify the drawing by manipulating or adding on objects onto the screen. A pen-computer is augmented by a speech recognition device in order to take advantage of the natural tendency to combine gesture and speech.

The focus of this paper is on pattern recognition and on the interpretation of pen input. The methods described here are tested on the recognition of tables, geometrical figures and gestural commands. They are easily extensible to other types of drawings. The algorithms are designed to work fast, as is required by the interactive applications in which they are used. The interpretation methods are responsible for the layout of the whole drawing by correctly positioning each graphical object (equalization of columns in a table or object alignment in a network diagram).

Two prototype systems have been developed which allow for an incremental design of graphic documents using a pen and voice interface: TAPAGE, a table editor and DERAPAGE, a diagram editor. The methods used for both systems are identical, demonstrating the re-usability and generality of the approach.

2: Pattern recognition in a Man-Machine interaction context

For the applications described in this paper, we use a pen-based PC augmented with voice input from a speech recognition board on a second PC. Corresponding to the hardware we selected, available input media are Pen and Microphone, and the modalities they support are drawing, gesture and speech. Commands may be given either by speech or by pen. The pen allows gestural command and selection of menu options. When dealing with different modalities, there are various ways of combining them and interpreting their combinations. As indicated by various studies [3, 5], the parallel and combined use of both pen and voice helps an application interface to be more convivial and natural to the user. These results encouraged us to strive towards building a real synergistic multimodal interface based on an original agent architecture [2].

0-8186-7128-9/95 $4.00 © 1995 IEEE

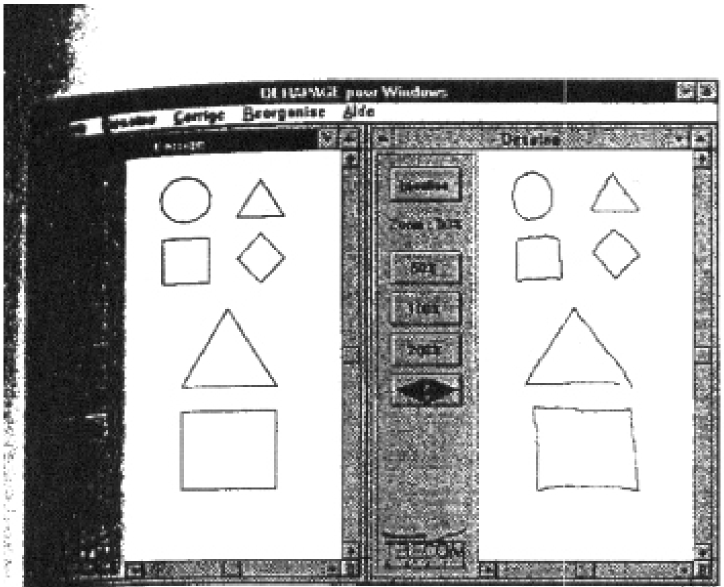

Figure 1: The two main windows of DERAPAGE

Ambiguities may arise when data might be categorized either as command or graphic input (for instance: selecting an object by encircling it vs. drawing a new circle object). Contextual interpretation is necessary to reduce the ambiguity, but the useful context itself is difficult to handle and to define. In our pen-based interfaces, ambiguities are mainly solved by the user who informs the computer of the nature of the stroke category by choosing the correct window (Figure 1): on the right is the drawing window and on the left, the correction window.

To understand the problems that we faced in designing a diagram editor, we give the following scenario of an interactive design of a graphical document: after the user has sketched a draft on the sensitive area of the pen-computer, an accurate version of the drawing is immediately displayed in the correction window. It may not match the user's intentions. These differences result from errors in automatic interpretation, or from changes in the user's mental model, which are strong characteristics of incremental design. The user is then allowed to correct the displayed drawing and the new corrected version appears.

Graphic document design in a man-machine cooperation context means that several requirements have to be fulfilled. The first requirement is to provide a recognition algorithm which does not depend on the production style. In our systems, the optimal values of the thresholds involved in the graphic interpretation are set up according to the quality of the produced drawing.

The second requirement is certainly the most important in a man-machine interaction: the swiftness of the feedback. During the design and modification of a drawing, interpretation must be almost immediate. Therefore, we added a requirement to our systems: the returned results should satisfy the user in less than one second after initial production; we optimized data structures and balanced the trade-off between accuracy and recognition speed accordingly.

It should be kept in mind that the user is in front of the machine and that the incremental design allows him to interact with the displayed drawing in order to compensate for any deficiencies of the machine.

3: Delayed and early recognition

We developed two kinds of methods for the recognition of on-line pen entries: delayed and early recognitions. These algorithms are used in various domains: tables, gestural commands or geometrical figures. They are presented through several examples in order to show that it is easy to re-use such modular algorithms and to make them complementary. Moreover, their efficiency is underlined: what is processed by early recognition does not need to be re-recognized during the delayed recognition.

3.1: Delayed recognition

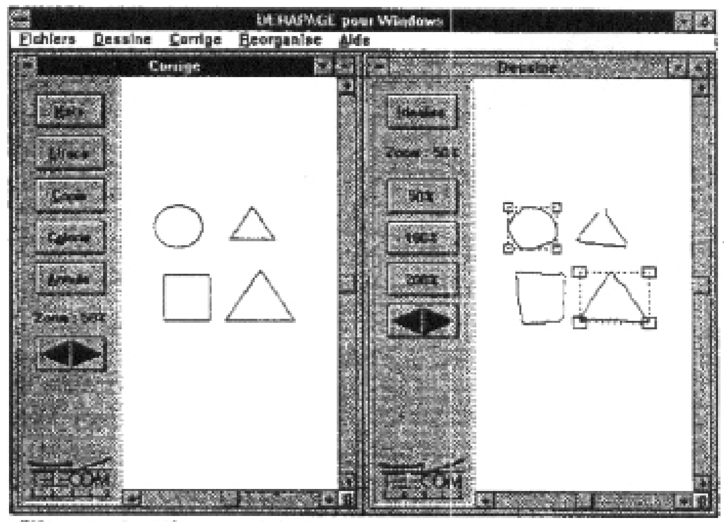

Figure 2: The multistrokes objects are managed by the delayed recognition

The delayed recognition algorithm was written for TAPAGE in order to recognize horizontal and vertical lines in tables [4]. The recognition of tilted lines has been added later as a generalization of the method. The ink data (see Figure 2, right window) are processed in two main steps. The first performs segmentation by splitting strokes into segments with a recursive dichotomy: each stroke is divided into two new strokes until a stroke is classified as a segment. The second step merges the contiguous segments which have approximately the same direction. Each resulting segment then becomes a graphical object. Around each segment, an "attracting" zone is defined using the numerical coordinates of the rectangle surrounding the original stroke. Figure 3 shows that the attracting zone may have different widths according to the quality of the draft strokes.

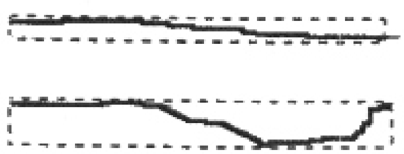

Figure 3: Attracting zones around a segment

The widths of these zones define local threshold values for attracting free end-points and reconstruct most of the junctions between segments by building two reference tables. The values in those tables are the average of closest coordinates according to the attracting zone. The references (RefX and RefY) are used to represent segments: each segment is a graphic object with four pointers corresponding to the coordinates of its extremities (Figure 4). To prepare further processing, the references are sorted in decreasing order.

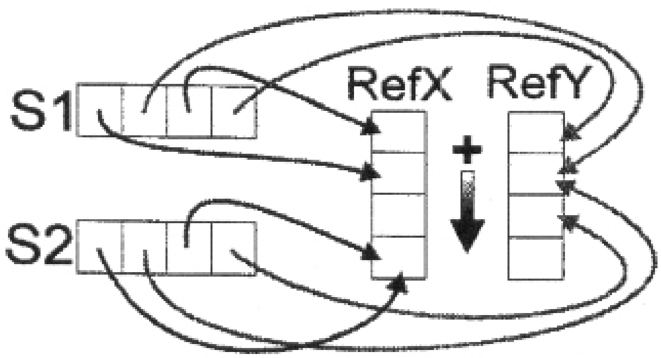

Figure 4: Structure for the segments S1 and S2

At this stage of the algorithm, the variability of data is handled by the reference extraction process. The displayed drawing is neat, but not interpreted. In the case of geometrical figures, it is possible to display an accurate version of the figures before the system identifies them as rectangles or diamonds. To find out the identities of the objects in the drawing, model matching is necessary. The model is a structural description of an object (organization of segments) which corresponds to a geometrical definition. This analytical method performs the recognition of each object in an attempt to reconstruct them as a combination of segments.

For instance, in DERAPAGE, three kinds of figures are considered: rectangles (and squares), triangles and diamonds. To recognize a triangle, the algorithm scans the list of segments to find the ideal combination: 3 segments attached by 3 vertices. A similar description is given for each figure. It is easy to add other kinds of figures (trapezoid, parallelogram,...) through appropriate definition.

In addition, in order to synchronize with early recognition and to prepare for structure recognition, the rectangle surrounding each object is computed and added on to an object list.

3.2: Early recognition

This algorithm is called "early recognition" because, in DERAPAGE, it is used to process some strokes before involving delayed recognition, which is a more complex task. It is also used, with additional capabilities, as a gesture recognizer during the correction task. This algorithm uses the global characteristics of a drawing.

The aim of this algorithm is to recognize a set of figures produced by the user. One figure is a series of strokes, where each stroke is a series of dots. Each stroke is processed separately and is defined by a set of descriptors: the number of changes in direction and the values of the main directions, its surrounding rectangle and what we call its sensitive areas. The main directions are computed according to a Freeman code on 8 directions. The sensitive area is defined by rectangles on each vertex of the surrounding rectangle which are proportional to this surrounding rectangle (see Figure 6). After this computation, a model matching is performed. Each model is represented by a stroke description or by a series of stroke descriptions.

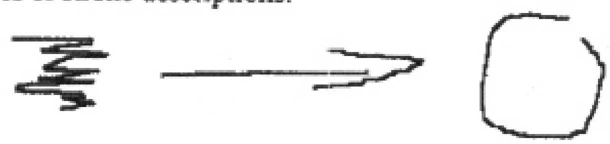

Figure 5: An erasure, an arrow and a surrounding circle

Figure 5 shows a set of gestures, useful for the correction of drawings; they correspond to the actions "erase", "move" and "select". The model of the erasure is a single stroke description with at least 3 direction changes. The model of an arrow is a list of two descriptions: the first stroke has one main direction, the second has two main directions. In this case, the surrounding rectangles are both used to determine which are the objects involved in this gesture (which are the objects to move) and to determine the direction of this arrow by the relative position of the surrounding rectangles.

The use of this algorithm in several pen interfaces with different sets of gestures led us to implement it in

DERAPAGE as an early recognizer for a set of geometrical figures. Any object that this algorithm can recognize does not have to be recognized again by the delayed recognition; DERAPAGE allows a kind of task sharing.

In order to easily discriminate curves that will be sent to the early recognizer, we decided to choose one-stroke closed curves. A closed curve is a series of dots where at least one dot of the last quarter of the series is "near" the first dot of the series. The main steps for processing this stroke are:

○ Computation of the surrounding rectangle (*surrounder*).

○ Computation of the four sensitive areas (*SA*) on the vertex of the *surrounder*, proportional to the *surrounder* (Figure 6).

○ Determination of the number of segments by direction quantization on 8 values.

○ Classification and decision.

For the figures it has to recognize, the algorithm has the following descriptions:

- Rectangle: 4 vectors, at least 2 *SA* occupied
- Diamond: 4 vectors, less than 2 *SA* occupied
- Triangle: 3 vectors, no constraint on *SA*
- Ellipse: at least 4 vectors, less than 2 *SA* occupied

If a new figure, for instance a parallelogram, has to be recognized by the system, new descriptions can be given: in order to correctly discriminate the rectangle from the parallelogram, the rectangle's description sharpens.

- Rectangle: 4 vectors, at least 3 *SA* occupied
- Parallelogram: 4 vectors, 2 *SA* occupied

In the same way, if the trapezoid is added, the description of the parallelogram has to be more precise:

- Parallelogram: 4 vectors, 2 opposite *SA* occupied
- Trapezoid: 4 vectors, 2 successive *SA* occupied

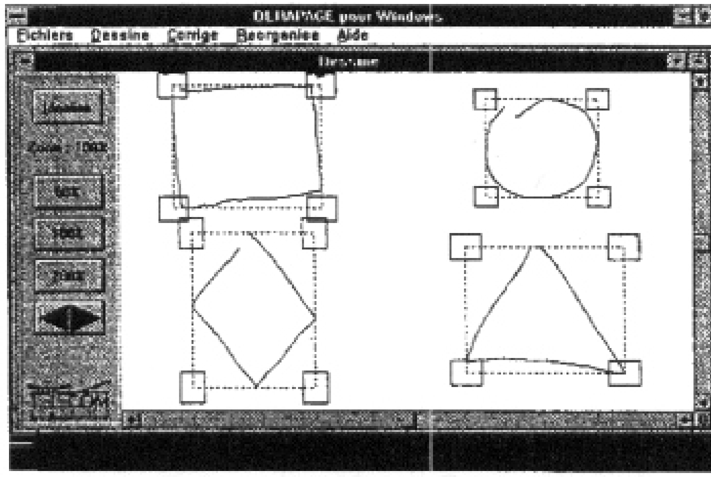

Figure 6: Samples of closed curves, their *surrounders* and *SA*.

At this stage, squares are classified as rectangles and circles as ellipses. The final decision of their specificity is taken during the structure recognition. Polygonal figures can be recognized either by the early or by the delayed recognizer depending on the number of strokes. Circles and ellipses can only be recognized by the early recognizer: it is assumed that they are always one stroke figures.

If a figure is recognized, each stroke is removed from the list of strokes which are sent to the delayed recognition. A list of *surrounders* is maintained with their main characteristics and the figure they surround. It is useful for the structure recognition step.

In the case of network diagrams, after the computation of each algorithm, it is possible to have some remaining free segments which were not recognized by the model matching. Figure 7 shows an example:

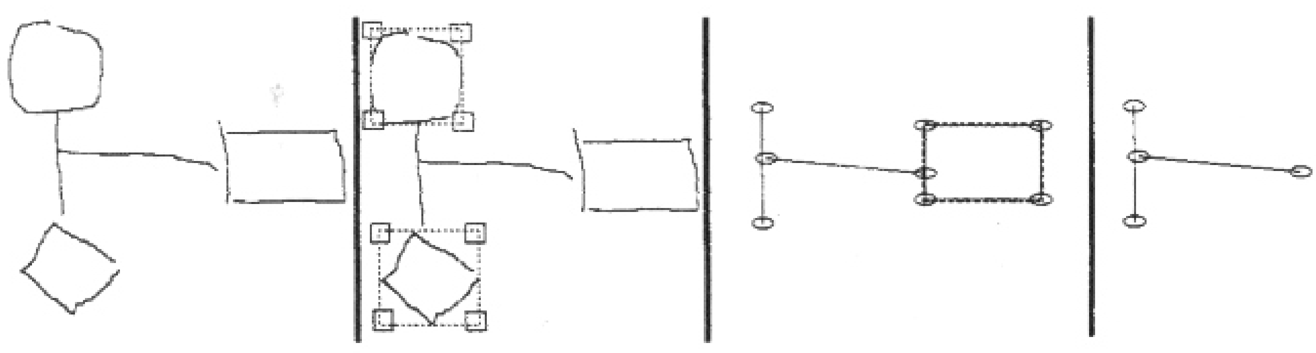

Figure 7: The process to find free segments

The circle and the diamond are recognized by the early recognition algorithm as closed curves. The strokes involved in this process are withdrawn from the original list. Then the delayed recognizer finds a rectangle using model matching. The last two segments will be considered as links of the diagram network.

4: Structure recognition

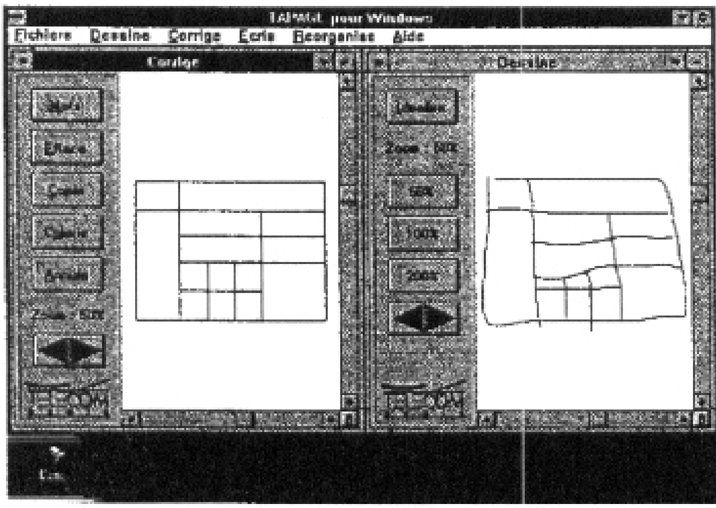

Figure 8: Table recognition in TAPAGE

The method developed in TAPAGE [4] was aimed at recursively finding the columns or the lines which have to be equaled (Figure 8).

This technique is extended in DERAPAGE in order to find the global structure of the drawing. As an example, we consider the drawing of Figure 1. Figure 9 illustrates each step of the method. After the pattern recognition, the maintained list of *surrounders* representing objects is accessible. In the first step, a new reference table is built, detecting alignments between edges of each *surrounders*, in order to describe each *surrounder* with these new references (step 2).

The third part of the algorithm is a recursive comparison between *surrounders*. Each time that at least two neighboring *surrounders* are similar, a new *surrounder*, enclosing them, is constructed and the comparison continues until it is impossible to construct a new *surrounder*. The assumption is that all objects in a *surrounder* have the same size according to the recursion level. The last step of this structure recognition is the reconstruction of each object with its new dimensions.

As shown on left in Figure 10, multiple alignments may be found at the same recursion level with this method. One solution is to decide that objects are involved in only one *surrounder* (for instance the first one found) but some significant alignments may get lost in this operation. Another solution is to assign a constraint value to each *surrounder* corresponding to the number of surrounded objects.

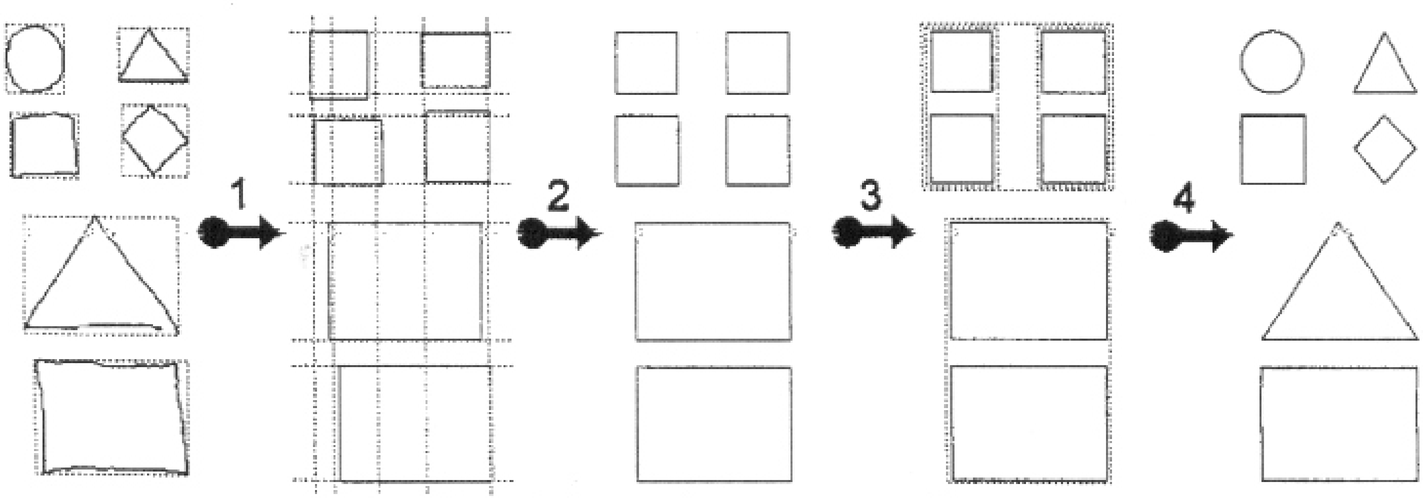

Figure 9: Steps of structure recognition

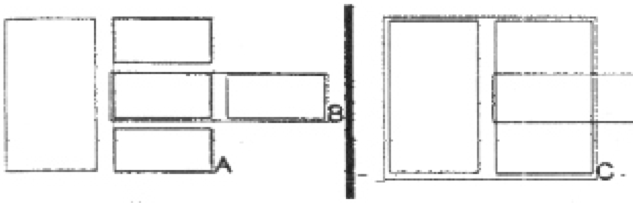

Figure 10: The recursive process

In the example above, the *surrounders* A and B are linked in order to take into account of the two alignments even if the second recursion level (Figure 9, on right) changes the size of A because of the construction of C. After that, objects in B have the same sizes as objects in A, because A has a bigger constraint value than B. Another constraint is propagated during the computation, the regularity of a figure: if a *surrounder* is almost a square, the resulting object will be regular, i.e. an ellipse will be a perfect circle. If free segments were found in the previous algorithms, they are placed in the middle of the nearest side of a *surrounder*.

5: Conclusions and perspectives

We described two recognition methods that we used in a Human-Machine interaction context. Each method is fast and works on several kinds of figures. Their combination increases the speed and the set of recognized figures by exploiting their modularity. It is possible to expand the set of figures by finding out the discriminant structural descriptions of models.

The applications we presented are pattern and spatial layout beautifiers. The analysis of the global structure of a drawing combined with local recognition is an original way of processing that we will investigate further in a new release of DERAPAGE.

Data management on a pen-computer without reserving specific areas for commands and drawing are also of interest. We have already implemented another prototype which resolves ambiguities by opposing in a parallel fashion our drawing recognizer to an on-line hand-writing recognizer.

It is difficult to give a quantitative analysis of the results for an incremental method where the user can intervene at different steps of the design. Specific interfaces are built in order to test the performance of the recognizers as well as the way the users perform the gestures. The global evaluation of the gestural interface must combine the performance of the recognizers and other factors such as the use of personal gestures which differ from the models, the parallax which shifts the gesture from the expected location. A first result obtained on 269 erasing gestures performed by 9 novice users leads to 63 ineffective gestures. Surprisingly, the rather poor performance (74% of effective gestures) did not discourage the users who never chose the menu when they had the choice between gesture and menu for erasing.

In an other test, where the focus was more on the interaction than on the quality of the recognition, 8 novice users and an experienced one copied a set of 7 tables using the pen interface. They obtained satisfactory results for all tables. A total of 56 tables were produced. 39 were produced without using the correction for adjusting the interpreted feedback. In four cases, the users gave up on the session when the adjustment by correction lasted too long and started again. If the algorithm allows an instantaneous display of drawings after each user's operation (adding, deleting or moving an object), the automatic beautification sometimes does not match with the user's intention and is followed by other corrections. There was no significant differences between novices and the experienced user in the production duration. However, we will carry out more user studies to examine this issue.

In the future, pen-computers may be considered as a complementary device in a complex computer environment offering functionalities that are not available with conventional computers. The pen-computer as a front-end is proposed by [1] for visual languages. The integration of a pen-computer with existing systems, such as 3D image synthesis or multi-media databases, opens new perspectives for the application domains of pen-based interaction.

REFERENCES

[1] CITRIN W.V. (1993). Requirements for Graphical Front Ends for Visual Languages. IEEE Symposium on Visual Languages, pp. 142-150.

[2] FAURE C. & JULIA L. (1994). An Agent-Based Architecture for a Multimodal Interface. AAAI-IM4S: Stanford (USA), pp. 82-86.

[3] HAUPTMANN A.G. & McAVINNEY P. (1993). Gestures with speech for graphic manipulation. International Journal of Man-Machine Studies, 38, pp. 231-249.

[4] JULIA L. & FAURE C. (1994). L'interpretation de tableaux dans une situation d'interaction utilisateur-machine. CNED'94: Rouen, pp. 215-222.

[5] OVIATT S.L. (1992). Pen/Voice: Complementary multimodal communication. Speech Tech'92: New York, pp. 238-241.