01 Jan 1997

Keyboardless Visual Programming Using Voice, Handwriting, and Gesture

Jennifer Leopold
Missouri University of Science and Technology, leopoldj@mst.edu

A. Ambler

Recommended Citation

J. Leopold and A. Ambler, "Keyboardless Visual Programming Using Voice, Handwriting, and Gesture," *Proceedings of the IEEE Symposium on Visual Languages, 1997*, Institute of Electrical and Electronics Engineers (IEEE), Jan 1997.
The definitive version is available at https://doi.org/10.1109/VL.1997.626555

Keyboardless Visual Programming Using Voice, Handwriting, and Gesture

Jennifer L. Leopold and Allen L. Ambler
Department of Electrical Engineering and Computer Science
The University of Kansas
Lawrence, KS 66045

Abstract

Visual programming languages have facilitated the application development process, improving our ability to express programs, as well as our ability to view, edit, and interact with them. Yet even in visual programming environments, productivity is often restricted by the primary input sources: the mouse and the keyboard. As an alternative, we investigate a program development interface which responds to the most natural human communication technologies: voice, handwriting, and gesture. Speech- and pen-based systems have yet to find broad acceptance in everyday life because they are insufficiently advantageous to overcome problems with reliability. However, we believe that a visual programming environment with a multimodal user interface properly constrained so as not to exceed the limits of the current technology has the potential to increase programming productivity for not only those people who are manually or visually impaired, but for the general population as well. In this paper we report on such a system.

1. Introduction

Visual programming languages have greatly facilitated the application development process, improving our ability to express programs, as well as our ability to view, edit, and interact with them. Yet even in visual programming environments, the productivity of developers and end-users is severely restricted by the primary (and often sole) input sources: the mouse and the keyboard. As an alternative to these traditional human-computer interface tools, we investigate a visual programming user interface which responds to the most natural human communication technologies: voice, handwriting, and gesture.

To contrast multimodal and traditional WIMP (Windows, Icons, Menus, and Pointers) user interfaces, consider the following problem in Formulate, a form-based visual programming language [1, 2]. Suppose that you have the form in Figure 1 and you want the display value of the object tagged "red" to be the sum of the display values of the objects tagged "blue" and "green", and the constant 117.

In the non-multimodal, WIMP user interface for Formulate, this could be accomplished with the following procedure:

(1) Click on the object tagged "red" to select it.
(2) Click on the icon for the display value attribute.
(3) Type "(+" in the expression window edit-item.
(4) Hold down the option key and click on the object tagged "blue" to reference it in the expression. (The display value will be the default referenced property.)
(5) Hold down the option key and click on the object tagged "green" to reference it in the expression. (The display value will be the default referenced property.)
(6) Type "117)" in the expression window edit-item.
(7) Hit the enter key or click on the done button in the expression window to submit the expression for evaluation.

The process of alternately dragging the mouse between windows and typing on the keyboard is typical of spreadsheets, word processors, and most other computer application interfaces. It requires the user to move one hand back and forth from the mouse to the keyboard. While we, the professionals in the field, have grown accustomed to the combination of mouse movements and typing in our human-computer interactions, for some people it is tedious and distracting; an impediment to productivity. This is particularly true for those people who are manually or visually disabled or who have never learned to type. Furthermore, it can lead to repetitive stress injuries.

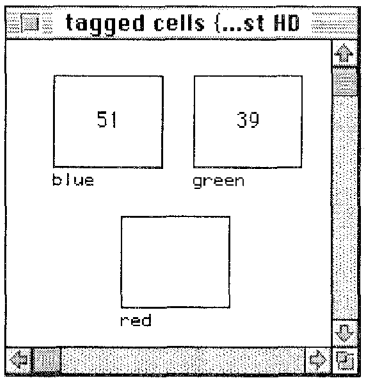

Figure 1. A form containing three tagged objects.

In the multimodal user-interface for Formulate which utilizes a pen-based computer and a lightweight headset microphone, this same task could be achieved by doing the following:

(1) <u>Tap</u> once with the pen on the object tagged "red" <u>or say</u> "select object red" to select it.
(2) <u>Tap</u> once with the pen on the display value attribute <u>or say</u> "select display value."
(3) <u>Say</u> "add reference object blue, reference object green, one hundred seventeen." <u>Or</u>, at the point where you reference an object, <u>tap twice</u> on that object, and resume the spoken input for the remainder of the expression. Any part of the expression (except for the object references) could also be handwritten with the pen in the expression window.
Note that the commas in the expression text above are included simply for readability. They do not have to be spoken. Also note that the display value will be the default referenced property for both of the referenced objects.
(4) <u>Say</u> "done" <u>or tap</u> once on the done button in the expression window to submit the expression for evaluation.

In the multimodal procedure, the entire task can be performed using spoken commands or the dialog can be augmented by handwriting and gestures such as tapping with a pen. There is no need to manipulate more than one input device, nor is the user required to visually follow the path of the cursor on the screen while navigating the input device to an object, which is the case with a mouse. Furthermore, there is no need to have a special skill (e.g., typing) in order to efficiently use the system. The only skills that are required are those typically acquired by at least age twelve: the ability to speak and write.

In this paper, we will discuss how the domains of discourse can be bounded in the user interface for the visual programming language Formulate in order to make effective use of voice, pen, and handwriting as sources of input, dispelling the assertions that multimodal interfaces are unreliable and have no advantage over WIMP interfaces, and bringing us closer to the realization of public programming environments.

2. Background

2.1. Related Work

Much work has been done on multimodal user interface design. Assistive robots and virtual environments have been the focus of [3, 4, 5, 6]. Software tools such as GIVEN (Gesture-based Interactions in Virtual Environments) [7] and ARCHIE [8] have been developed in an attempt to provide application-independent multimodal interface capabilities. The results of those projects in addition to studies such as those with DIVERSE [9], the various MIAMI (Multimodal Integration for Advanced Multimedia Interfaces) projects [10], and Put-That-There [11] have contributed considerably to the knowledgebase of multimodal interaction techniques and paradigms. The majority of these systems have concentrated their efforts on voice recognition and/or gestures (from sources other than a pen), and have not included handwriting.

There has also been work in the area of multimodal interface agents. Agentsheets [12] is a visually oriented authoring environment featuring a construction paradigm consisting of a large number of autonomous, communicating agents organized in a grid. These agents can use different communication modalities such as animation, sound, and speech. This is a much more restricted use of multimodal input than that which is required in the Formulate multimodal user interface.

Application-dependent multimodal interfaces have been implemented for educational, database, file management, and word processing/dictation applications [13, 14]. Most all of these systems recognize spoken natural language commands over a restricted vocabulary. However, only a very few systems, such as ICPdraw [15], a drawing application, and JEANIE [16], an appointment scheduling program, have attempted to combine both voice and pen input modes, recognizing gestures and handwriting as well as spoken language as input. To date, there have been no recognized attempts to develop a pen/voice multimodal user interface for a visual programming language.

2.2. Obstacles to Using Voice, Handwriting, and Gestures

Speech- and pen-based systems have yet to find broad acceptance in everyday life. The two most common arguments against using such input media are: (1) there is no advantage to using pen and/or voice instead of the keyboard and mouse, and (2) reliability of handwriting and voice recognition today is not high enough. In the past, some multimodal user interfaces have simply tried to mimic the actions of the mouse and keyboard instead of trying to improve on the effectiveness of those input modalities. For example, using a pen to tap on the letters in a (soft) keyboard display provides no advantage over typing the letters on an actual keyboard. The reliability issue is a valid complaint as well. When reliability is poor, efficiency suffers as users have to spend more time correcting their input.

But perhaps the real reason why many of the early endeavors in multimodal user interface design have failed is because they have either tried to do too much or too little. Natural language recognition capabilities today are far from those of Hal the computer in *2001: A Space Odyssey*. A user interface which attempts to recognize and act on unrestricted natural language input is unlikely to succeed due to the context-sensitivity of the English language and the unrealistic expectations of the application's users. Interface designers and users must also accept the limitations of current handwriting recognition technology. Personal computers with pen tablets often require that the user first "train" it on their own personal handwriting style. Certain gestures such as for deleting a character must be learned and practiced. For better recognition, the user may be required to print instead of using cursive writing. With both handwriting and voice recognition, the user and interface designer must be willing to make some concessions if they are to enjoy the advantages that these input media can potentially offer.

On the other hand, multimodal user interface designers have sometimes done too little in expecting a single input medium to be sufficient for a particular task. For example, controlling information entry is more difficult using voice input alone. But a pen can supplement and direct a voice recognition interface to show where the spoken input should go [17] such as in the command "put that there." Voice can also be used to disambiguate a pen gesture. For example, tapping once on a Formulate object can potentially have many interpretations, but can be clarified by first saying "reference", "resize", "move", or "select." Used in combination, speech, gesture, and handwriting provide a more direct link between a user's intention and the external expression of that intention [18]. If properly constrained, we believe that they can prove to be reliable and much more effective modes of input than the keyboard and mouse.

2.3 Formulate

Formulate is a form-based visual language in which programs are expressed as systems of equations. Programming proceeds by constructing forms, attaching objects, and specifying equations by which these objects obtain their values. An equation can be a constant or it can contain references to other objects composed with functions like, +, *, etc., as well as user-defined functions (themselves forms). Formulate has structured objects, arrays, lists, and tables, as well as event-handling objects, buttons, text entry objects, and selection objects. Development and execution modes are provided for building, and then executing applications.

3. Objectives

Our objective in developing a multimodal user interface for Formulate is to further improve the ability to view, edit, and interact with visually-oriented programs using voice and pen instead of the mouse and keyboard. A user interface that allows users to "program" an application (in some sense of the word) has more extensive requirements than a user interface that simply allows use of an application. Specifically, the Formulate multimodal user interface requires that the user be able to perform the following tasks using only voice and/or pen:

(1) Point at objects as well as objects embedded within other objects (e.g., an object within a form, a region within an array, etc.) for the purposes of (i) selection and (ii) reference within an expression.
(2) Move and resize objects using direct manipulation.
(3) Resize and split regions in a structured object using direct manipulation.
(4) Enter text into any of the following: (i) the expression window edit item, (ii) a text entry object during execution mode, (iii) a prompt in a dialog, and (iv) a single element within a structured object.
(5) Make selections from menubars.
(6) Make selections from (pop-up menu) selection objects in execution mode.

There is a great deal of context-sensitivity in the interactions that can be performed in this interface. When you "point" in a form window, what exactly are you pointing at and for what purpose? When you perform a dragging gesture, are you trying to reposition an object, resize it, or manipulate another object contained within it? In the non-multimodal user interface for Formulate, these actions are disambiguated by requiring a combination of keystrokes (shift, control, etc.) in conjunction with the mouse manipulation. The multimodal interface must use only voice and/or gestures to accomplish this.

Studies with marking menus [19] and unistroke alphabets [20] have found pen gestures to be more easily learned and more readily used when the gestures are short and easy to draw. Therefore, the gestures for the Formulate user interface must be defined to be simple and reasonably representative of the action to be taken. For most Formulate interactions, tapping and dragging gestures should be sufficient.

Speech recognition tends to be less reliable when a spoken utterance contains disfluencies and self-repairs, including repetitions, false starts, and filled pauses. A study by [21] found that the rate of speech disfluency increases as the length of a spoken utterance increases. Furthermore, the more the syntax is constrained, the fewer words the recognition software has to choose from, thereby helping to increase recognition speed and accuracy. To help ensure reliable and efficient performance from the voice recognition software, the spoken command language for the Formulate interface must be concise, and the vocabulary constrained based on the context in which the spoken command is issued. In particular, these guidelines should be followed: (1) there should be one consistent way of saying each command, (2) the variation of the vocabulary should be limited (e.g., allow either gray or grayish, but not both), and (3) short, monosyllabic words should be avoided in commands since they correspond to relatively short acoustic strings that are often swallowed in continuous speech, particularly when they are grouped together [22].

The multimodal user interface we developed for Formulate allows continuous speech, speaker-independent voice recognition via a lightweight headset microphone worn by the user, and handwriting and gestures through a personal computer-sized tablet and pen. In designing this interface, we consciously decided to avoid input media that require wearing any bulky apparatus, including various head gear and gloves. We also chose to avoid as immature technologies environments that utilize cameras to track eye or hand movements. As nearly as possible we wanted a solution that utilizes the typical implements of the workspace environment, affords the conventional office freedoms to move about, and emulates existing operational metaphors.

4. Implementation of Multimodal Interface Activities in Formulate

As demonstrated in the example given in the Introduction, the user can perform many tasks in Formulate using a choice of input modes. There is evidence that different people have different cognitive styles, and individual preferences may play a role in the selection of one input mode over another [23]. Therefore, when practical, both pen and voice alternatives should be available for interface activities. For some types of tasks, however, one single input mode can be far more effective than others. For example, as we shall see in the section on Object Manipulation, resizing a Formulate object is more easily performed using a pen than by trying to issue spoken directives. Still other interactions are best accomplished using a combination of input modalities. All of these issues had to be taken into account in designing the multimodal user interface for Formulate.

The three primary interactions in Formulate are typical of most human-computer interfaces: navigation, object manipulation, and data and formula input. The first two activities are "meta"-interactions in that they are communications with the system, while the last is input passed through the system that becomes part of the derived program.

4.1 Navigation

As a form-based visual programming language, the Formulate user interface requires direct manipulation of graphic objects in multiple (form) windows. In general, navigation in the Formulate workspace can be performed equally effectively using either voice or pen gestures. Navigation by voice is made possible by tagging a Formulate object with a name which is then displayed below the object's display. As soon as an object is assigned a tag (which must be distinct from other tags in use), that tag is introduced into the voice recognition vocabulary in the context of object identification. Objects can also be pointed at using pen gestures. Various gestures have been defined based on the task that the user wishes to perform (see the discussions on Object Manipulation and Data/Formula Input below). Form windows can be activated by selecting them by title (in effect, their "tag") from a "Window" menu via voice or pen, tapping once on the desired window with the pen, or by saying "activate window *form name*" where *form name* is the title of the form window.

At this time, no attempt has been made to integrate into the Formulate semantics the recognition of spoken deictic expressions such as "this", "that", and "there" for the purpose of navigation. We believe that the unique identification of objects by the use of tags and the specification of positions by pointing with a pen are sufficient and avoid ambiguous or erroneous references.

Formulate pull-down menu selections can be made using voice or pen. However, when menu selections are spoken, the user must be familiar with the available menu options as he/she will not see the visual display of the menu. For this reason, the pen may be the preferred input medium for novice users particularly when navigating through hierarchical (or "walking") menus so that the path of successive menu options is clear. But, for experienced users, voice can provide a much more accelerated selection of a menu item, particularly in a hierarchical menu. Voice also eliminates the motor control problem commonly associated with hierarchical menus [23].

4.2 Object Manipulation

Formulate objects can be created by dragging the pen in a downward motion inside a form window thus establishing the object's position and defining its size by virtue of the upper left and lower right corners of the resulting rectangular image. The user can also create an object by saying "create *objectType*" where *objectType* is the type of object to be made, followed by tapping the pen once in the location for the upper left hand corner of the new object. With the spoken command, the object will be created using a default value for size. The pen gesture is probably the preferred method of creating a new object since it allows the user to specify the size as well as the position in a single interaction. An existing object can be resized by positioning the pen over one of the object's four corner handles then dragging the pen in any direction. We have yet to determine a spoken command that would perform this action as efficiently as the pen. However, the user also has the option of changing the value of an object's width or height indirectly (via handwriting or voice) in the expression window. Creating a new object or resizing an existing object using pen gestures is not significantly different from the actions that would be performed if using a mouse. However, the pen allows direct control on the screen surface whereas the mouse requires indirect control on a surface that is some distance from the focal point of the action.

In the non-multimodal user interface, an object can be repositioned by clicking on the object with the mouse, then holding down the mouse button, dragging the mouse to a new location, and releasing the mouse button. In the multimodal interface, an object can be moved by first selecting the object (using pen or voice), and then using the pen to "drag" the object to the new location. Both the pen and mouse actions mimic the act of physically picking up an object and moving it. However, there is an important distinction between these two input media in that the pen allows absolute positioning whereas mouse navigation necessitates relative positioning, a far more complicated manual task. In an attempt to provide an even more efficient alternative for completing this operation, following the selection of the object to be moved, the user can say "move" then tap the pen once in the new location for the upper left hand corner of the object. As with the width and height attributes, the user still has the option of modifying the values for an object's upper left x- and y-coordinates (via handwriting or voice) in the expression window.

Formulate structured objects can be divided into logical parts, called regions. Regions are used both to define the object's construction and to describe its use in subsequent computations [1, 2]. When a structured object is created, it initially has only a single region which contains all the elements of the object. A new region is formed by dragging a boundary line to effectively split an existing region. Regions can also be resized by grabbing boundary lines and dragging them appropriately [24]. In the non-multimodal Formulate user interface, region selection and manipulation requires a combination of key strokes in conjunction with mouse movements in order to perform the desired actions (splitting, resizing, etc.) unambiguously.

Regions can be assigned tags in the multimodal Formulate user interface. To avoid confusion with the display of the tag that may have been assigned for the structured object containing the region and to avoid a cluttered display in an object with several regions, some of which may be surrounded on all sides by other regions, each region tag is displayed within the rectangular area of the region instead of the region's element values when the menu option "Show Region Tags" has been selected. The element values will be redisplayed when the menu option "Hide Region Tags" is chosen. Figure 2 shows a structured object with tag "my array" and five regions tagged "left", "middle", "top", "right", and "bottom."

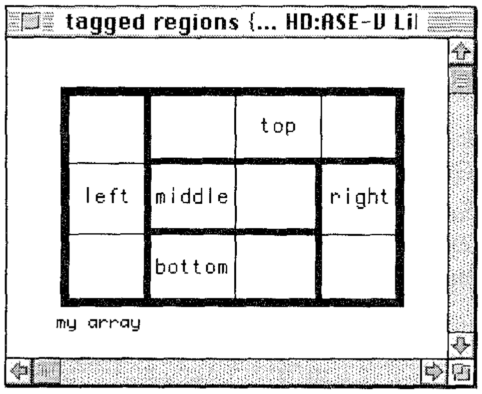

Figure 2. A structured object with five tagged regions.

Because of the nature of region display, their manipulation is best accomplished using voice to constrain the actions of the pen. A region can be selected by saying "select" and the region's tag name or by making a counterclockwise circular gesture with the pen anywhere within the desired region's boundary lines. Simply tapping on the region with the pen is inadequate since it is unclear whether the region or the underlying structured object is to be selected. A selected region can be resized by saying the words "resize", "increase" (or "decrease"), "by", an integer number of columns or rows, "on", and "top", "bottom", "left", or "right." If the user prefers to use the pen to specify the extent of the resizing, he/she can say "resize", tap once on the region boundary line to be moved, and then tap once on the row or column line to which the selected region line is to be moved. A selected (bounded) region can be split by saying the words "split", "down" or "left", "by", and an integer number of rows or columns. If the region to be split is unbounded, the spoken command would just be "split" followed by "down" or "left" since unbounded regions are always split in half. Again, if the user prefers to use the pen to split a bounded region, he/she can use the same procedure as for resizing with a pen, but say "split" instead of "resize" at the beginning. To split an unbounded region using the pen, the user can say "split" and tap once on one of the region boundary lines. Tapping on the top or bottom (left or right) region line will split the region in half vertically (horizontally). Unbounded regions cannot be resized in Formulate.

4.3. Data and Formula Input

Formulate provides both development and execution modes for building, and then executing applications. In development mode, expressions can be specified for any of the numerous attributes of an object. As much as possible, we have given the user the ability to employ whatever combination of the three input modes he/she wants to use in order to construct an expression. However, to help ensure efficiency and reliability of the input media, we did choose to impose some restrictions on this task. For example, only pen gestures are defined for performing text selection for the purpose of editing. This is due to the complexity of trying to describe a section of text using natural language. Once a section of text has been selected, it can be deleted, copied, etc. using either pen gestures or spoken commands. Due to restrictions on the maximum size of the dictionary for voice recognition and the decreased reliability of recognition as the dictionary size increases, the vocabulary recognizable in the context of expression window input is limited to special directives such as "select" and "reference", function names, object tags, numbers, and all of the alphanumeric characters found on a keyboard. If the user finds this inadequate or unpreferable for any part of the expression text, the pen can be used to handwrite entire words, individual characters, or symbols.

Formulate expressions must be given in prefix form. In order to enable the user to use voice input for function calls within an expression and to minimize the amount of text that needs to be specified, commonly used functions such as "+", "*", etc. have been assigned a corresponding spoken form ("add", "multiply", etc.) which translates to a left parenthesis, the operator, and a right parenthesis at the end of the expression string. (It should be noted that neither the multimodal nor the traditional user interface attempts to syntactically check the user's expression prior to its submission for evaluation.) When the user defines his/her own Formulate function, the function name, if pronounceable, will be automatically added to the voice recognition vocabulary in the context of a function name. It can then be used in the same manner as "add", "multiply", etc. If the function name is not made up of words that can be identified by the voice recognition software, the user will be asked if he/she would like to assign a different name for the function so that it can be recognized during spoken input.

In the Formulate execution mode, the user can interact with text entry, selection, and button objects. The user can select the object with which to interact in the same manner as objects are selected in development mode. Button objects are activated when they are selected. A selected text entry object will accept handwriting as well as spoken input. However, due to restrictions on the size of the voice recognition dictionary, the number of words recognizable in this context has been limited (e.g., numbers, words such as "dollars" and "cents", etc.). This is also true for the entry of data into an element of a structured object in development mode as well as for text entry in dialogs that may be displayed to prompt the user for information during a Formulate session.

A selection object allows the user to make a choice from a pop-up menu of user-defined entries during execution mode. For example, there might be a form containing a selection object for gender with entries "male" and "female." When defining the list of menu items for the selection object in development mode, the user will be asked whether he/she wants to add those words (if pronounceable) to the voice recognition vocabulary in the context of activation of this object in execution mode. In most applications, selections are made up of words that are commonly used in the English language. Thus, the majority of the time, the user will have the choice of making a selection using either pen or voice.

5. Current Implementation Status and Future Work

The multimodal user interface for Formulate discussed in this paper is currently implemented on an IBM ThinkPad personal computer using the Windows for Pen handwriting recognition capabilities and the Speech Systems Incorporated Phonetic Engine 500 continuous speech, speaker-independent voice recognition software. The Phonetic Engine software allows the developer to specify a context-free grammar to define the expressions that can be recognized in an application and provides function calls to parse spoken input. The Formulate voice recognition module is able to constrain the spoken commands by beginning the parse of spoken input from different nonterminals in the grammar based on the context in which the command is issued. A dictionary of object tags and function names is dynamically maintained for every Formulate form.

To date, our limited experience using the Formulate multimodal user interface has been very positive. The voice and handwriting recognition have been adequately reliable and the design of the interface has not proven to be too restrictive. There has been no tendency to want to revert to using the keyboard or mouse, a result that is not surprising since experimental studies by [25] and [26] showed that users of graphical interfaces with requirements similar to those of the Formulate interface preferred pen/gesture interfaces over those utilizing a mouse/palette design.

In the near future, we intend to further evaluate the usability of the Formulate multimodal user interface with empirical studies and to investigate the effectiveness of other forms of pen-based input such as an interactive laser-pen on our existing wall screen which is 3456 x 870 pixels. We also plan to test the multimodal interface in a collaborative environment whereby several users can interact within the same Formulate session, each with his/her own pen and microphone.

6. Discussion and Conclusions

Immature technology has held back pen and voice input technologies in the past, but increases in portable computing power and major advances in software and hardware technology will now allow them to be combined effectively [17]. It is our contention that user interfaces that allow users to "program" an application (in some sense of the word) are harder to develop than user interfaces that simply allow use of an application. Consequently, their development has lagged behind that of other multimodal user interface research. We believe that the Formulate environment is representative of public programming environments and indeed many other user interaction environments, and, as such, the results of our work will advance the general knowledge of how to design programming environments for the general public.

7. References

[1] Gerhard Viehstaedt and Allen Ambler, "Visual Representation and Manipulation of Matrices", in *Journal of Visual Languages and Computing*, Volume 3, 1992, pp. 273-298.

[2] Guijun Wang and Allen Ambler, "Solving Display-Based Problems", in *Proceedings of IEEE 12th Symposium on Visual Languages*, 1996, pp. 122-129.

[3] M. Beitler, Z. Kazi, M. Salganicoff, R. Foulds, S. Chen, and D. Chester, "Multimodal User Supervised Interface and Intelligent Control (MUSIIC)", in *Proceedings of AAAI 1995 Fall Symposium on Embodied Language and Action*, Cambridge, MA, 1995, pp. 5-11.

[4] K. Thorisson, "Computational Characteristics of Multimodal Dialog", in *Proceedings of AAAI Fall Symposium on Embodied Language and Action*, Massachusetts Institute of Technology, November 1995, pp. 102-108.

[5] Scott Fisher, "Virtual Interface Environments", in *The Art of Human-Computer Interface Design*, (Brenda Laurel, ed.), Addison-Wesley, Reading, MA, 1990, pp. 423-438.

[6] R. Pausch and R. Williams, "Tailor: Creating Custom User Interfaces Based on Gesture", in *Proceedings of the ACM SIGGRAPH Symposium on User Interface Software and Technology*, Snowbird, UT, 1990, pp. 123-134.

[7] K. Bohm, W. Hubner, and K. Vaananen, "GIVEN: Gesture Driven Interactions in Virtual Environments - A Toolkit Approach to 3D Interactions", in *Proceedings of Interfaces to Real and Virtual Worlds Conference*, Montpellier, France, March 1992.

[8] W. Smart, J. Flensholt, C. Gomard, A. Cairns, I. Ricketts, and E. Holnagel, "ARCHIE: Plan Recognition in a Human-Computer Interface", in *Proceedings of the 12th UK Planning/Scheduling SIG Workshop*, Cambridge, England, May 1993, pp. 6-7.

[9] J. Karlgren, I. Bretan, N. Frost, and L. Jonsson, "Interaction Models Reference and Interactivity in Speech Interfaces to Virtual Environments", in *Proceedings of the 10th Nordic Conference on Computational Linguistics*, May 1995.

[10] K. Hartung, S. Munch, L. Schomaker, T. Guiard-Marigny, B. Le Goff, R. MacLaverty, J. Nijmans, A. Camurri, I. Defee, and C. Benoit, *A Report of the ESPIRIT Project 8579 MIAMI*, WP 4, March 1996

[11] R. Bolt, "Put-That-There: Voice & Gesture at the Graphics Interface", in *Computer Graphics*, 14, 3, 1980, pp. 262-270.

[12] Alexander Repenning and Tamara Sumner "Agentsheets: A Medium for Creating Domain-Oriented Visual Languages", in *Computer*, March 1995, pp. 17-25.

[13] Victor Zue, "Towards Systems That Understand Spoken Language", in *IEEE Expert*, February 1994, pp. 51-59.

[14] Alexander Hauptmann, Kai-Fu Lee, and Alexander Rudnicky, "Survey of Current Speech Technology", in *Communications of the ACM*, March 1994, pp. 52-57.

[15] Jean Caelen, "Multimodal Human-Computer Interface", in *Fundamentals of Speech Synthesis and Speech Recognition*, (Eric Keller, ed.), John Wiley & Sons, Chichester, England, 1994, pp. 339-373.

[16] Minh Vo and Cindy Wood, "Building an Application Framework for Speech and Pen Input Integration in Multimodal Learning Interfaces", in *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, May 1996.

[17] Hewitt Crane, "Pen and Voice Unite", in *BYTE*, October 1993, pp. 98-102.

[18] Alexander Rudnicky and Alexander Hauptmann, "Multimodal Interaction in Speech Systems", in Multimedia Interface Design, (Meera Blattner and Roger Dannenberg, eds.), Addison-Wesley, New York, 1992, pp. 147-171.

[19] Gordon Kurtenbach and William Buxton, "User Learning and Performance with Marking Menus", in *Human Factors in Computing Systems CHI '94 Proceedings*, 1994, pp. 258-264.

[20] D. Goldberg and C. Richardson, "Touch-Typing With a Stylus", in *InterCHI'93 Conference Proceedings*, 1993, pp. 80-87.

[21] Sharon Oviatt, "Interface Techniques for Minimizing Disfluent Input to Spoken Language Systems", in *Human Factors in Computing Systems CHI '94 Proceedings*, 1994, pp. 205-210.

[22] Speech Systems, Inc., Phonetic Engine 500 Speech Recognition System Syntax Development Guide, Speech Systems, Inc., Boulder, CO, 1994, pp. 18-23.

[23] Ben Shneiderman, Designing the User Interface (Second Edition): Strategies for Effective Human-Computer Interaction, Addison-Wesley, New York, 1992.

[24] Jennifer Leopold and Allen Ambler, "A User Interface for the Visualization and Manipulation of Arrays", in *Proceedings of IEEE 12th Symposium on Visual Languages*, 1996, pp. 54-55.

[25] A. Apte and T. Kimura, "A Comparison Study of the Pen and the Mouse in Editing Graphic Diagrams", in *Proceedings of the 1993 IEEE Symposium on Visual Languages*, 1993, pp. 352-357.

[26] W. Citrin, "Requirements for Graphical Front Ends for Visual Languages", in *Proceedings of the 1993 IEEE Symposium on Visual Languages*, 1993, pp. 142-150.