A VOCALISATION-BASED DRAWING INTERFACE FOR DISABLED CHILDREN

Edward Burke[a,b], Annraoi de Paor[a,b], Gary McDarby[c]

[a] *National University of Ireland, Dublin, Dept. of Electronic and Electrical Engineering, Belfield, Dublin 4, Ireland.*
[b] *National Rehabilitation Hospital, Rochestown Ave., Dún Laoghaire, Co. Dublin, Ireland.*

Summary In our work with disabled children at Ireland's National Rehabilitation Hospital, a problem we have experienced in the facilitation of art activities is that traditional art materials and standard computer drawing programs sometimes prove inaccessible. In this paper, an original system, called "PaintMyVoice" is presented which facilitates the creation of two or three-dimensional images using a variety of novel input modalities. In particular, vocalisations can be used to create original images of a variety of objects, including trees, flowers and landscape elements. Additional input to the system can optionally be provided via mouse, keyboard, switch interface or digital camera depending on the abilities of the user. Here, the program's user interface is described, with an emphasis on accessibility features. The signal processing techniques used to measure various vocal characteristics including intensity, pitch and other spectral characteristics are outlined. The means of translation from vocalisation to visual representation is also explained for each type of object discussed. This technology facilitates artistic expression by all children, but especially those with severe physical and/or intellectual disabilities. Furthermore, in certain cases, it may be used to provide motivation in therapeutic vocal exercises. Finally, the results of initial user trials are presented.

1. INTRODUCTION

In this paper, the design of a computer program called Paint My Voice is described. This experimental software is designed to facilitate artistic expression by people, especially children, with disabilities. It is intended to explore the use of multimodal and highly accessible interfaces in the creation of visual art. By suitable adaptation, most computing tasks can be carried out under the control of single switch, often using a scanning paradigm [1]. Furthermore, several programs that facilitate drawing using external switches and other special input devices are commercially available. Many such programs sacrifice flexibility in the interest of broad accessibility, allowing the user to perform relatively simple "drawing" functions, such as revealing a pre-drawn image bit by bit. By contrast, the Paint My Voice program features a number of drawing tools, both conventional and unconventional, which are highly configurable. The input mode used with each of the program's drawing tools can be uniquely tailored to an individual user's needs. Furthermore, a special emphasis is placed on the use of sound as an input modality for computer drawing programs. As a means of human-to-computer communication in rehabilitation, the use of sound is well established, both in speech recognition systems [2] and in input systems based on non-verbal sounds [3]. Computer-based drawing is an application particularly well suited to non-verbal audio input.

2. PROGRAM DESCRIPTION

The Paint My Voice program facilitates drawing for users with disabilities through its unique input mode customisation feature. It caters for a variety of input devices including mouse, keyboard, external switches and non-verbal audio input. A number of different drawing tools are provided, such as paintbrush, line, rectangle, ellipse, etc. The way in which each of these drawing tools is controlled by the input devices used may be configured individually, allowing the program to be tailored to the needs and abilities of an individual user. The support of an external switch interface enables the program to be used with an enormous range of pre-existing switch devices, such as chin switches, suck-and-blow switches, etc.

The program features three components, each with a distinct mode of operation – a *canvas editor*, a *tool input editor* and an *object generator*. The canvas editor is the main drawing interface. In this mode, the user selects one tool at a time from those shown in the toolbar. Each tool provides functions to add, modify or remove elements from the picture. Paint My Voice is a vector-based tool, rather than a raster-based one [4]. An array of picture element objects is maintained by the application. In the program's object model, each element in this array is an instance of one of a number of classes derived from a generic picture element base class. Furthermore, each of the application's tools is an instance of a specific tool class derived from a generic tool class. The application maintains an array of tool objects from which the user selects one at a time.

Each tool is controlled via an interface exported by it to the application. Associated with each tool is a user-defined *tool input map* that determines how messages from the each of the input devices used are mapped onto the interface of that tool. These tool input maps are created using the application's tool input editor.

The application's third component is an object generator. This is used for generating two and three-dimensional object models and images from vocalisations and other sounds. The objects created are exported as two-dimensional raster images for incorporation into drawings in the canvas editor.

3. TOOL INPUT CONFIGURATION

The tool input editor facilitates the customisation of user control of each of the tools made available by the program. In this mode, as in the canvas editor, the array of tools available to the user is displayed in the toolbar. However, unlike the canvas editor, the main area of the window is used to display a block diagram, graphically representing the relationship between input devices and the drawing interface exported by the currently selected tool.

Each input map is comprised of blocks of three distinct types – input blocks, intermediate blocks and output blocks (including a block for the tool in question). An input map may be thought of as a signal flow diagram where the input blocks are signal sources, the output blocks are signal sinks and the intermediate blocks are used to perform signal transformations. Blocks are added to the graph one at a time by the user and positioned using a click and drag interface. Each block features a number of input and output *pins*. Input pins are drawn on the left side of a block while output pins are drawn on the right. Connections are formed between blocks by clicking on the output pin of one block and dragging to the input pin of another block (or vice versa). An output pin may be connected to more than one input pin, but an input pin cannot be connected to more than one output pin. Each pin is of one of three possible types – *event*, *Boolean* or *floating-point*, denoted in the tool input editor by the labels "(!)", "(b)" and "(f)" respectively. The user may only connect an output pin of one block to an input pin of another block if both are of the same type.

An input block typically encapsulates an input device. Blocks are included for the mouse, the keyboard and an external switch interface. Audio input functionality is also encapsulated within a number of additional input block types. An input block has no input pins; rather the signals emanating from its output pins are generated by user interaction with the input device represented by that block. For example, the mouse input block has four output pins – one Boolean output representing the state of each of the left and right buttons (with value 1 when pressed, 0 otherwise), and one floating-point output for each of the X and Y coordinates of the mouse pointer. Another type of input block is the constant block which has one output pin, of either the floating-point or Boolean type, which can be assigned a constant value in the input map editor.

Most input maps contain only one output block, representing the drawing interface exposed by the tool in question. The exposed interface is accessed by means of a number of input pins displayed on the block. In terms of the block diagram, output blocks represent signal sinks and consequently have no output pins. For instance, the output block for the paintbrush tool, features 9 input pins (as listed in Table 1).

Tab. 1. All drawing operations of the paintbrush tool are accessed through these nine input pins.

Paintbrush Tool Output Block Pins		
Pin	**Name**	**Type**
1	X coordinate	Floating-point
2	Y coordinate	Floating-point
3	Painting	Boolean
4	Speed of movement	Floating-point
5	Moving	Boolean
6	Speed of rotation	Floating-point
7	Rotating	Boolean
8	Angle value	Floating-point
9	Turn through angle	Event

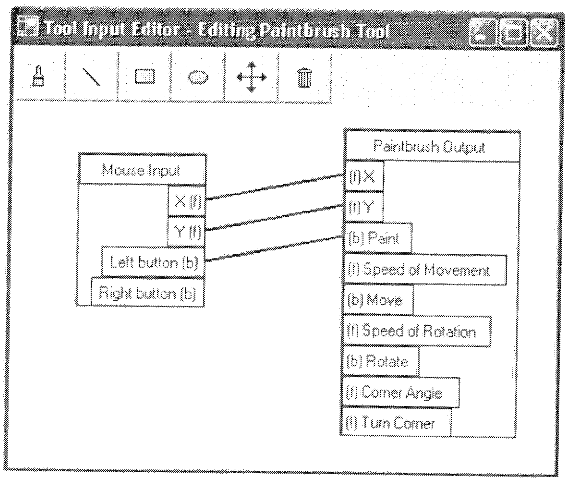

Fig. 1. A simple input map for controlling the paintbrush tool with the mouse.

A tool's input map typically contains a number of intermediate blocks that are used to transform the signals emanating from the input blocks into a form suitable for connection to the input pins of the tool's output block.

A very simple example of an input map for the paintbrush tool is that shown in Fig. 1, in which the X and Y coordinate outputs of the mouse input block are connected to the X and Y coordinate inputs of the paintbrush tool output block and the Boolean left button output is connected to the painting input. With the paintbrush tool's input map configured thus, the user draws by clicking and dragging on the canvas with the mouse. As long as the left mouse button is held down, moving the mouse will paint on the canvas.

An alternative input map for the paintbrush tool is shown in Fig. 2. This map is designed to facilitate drawing with two external switches

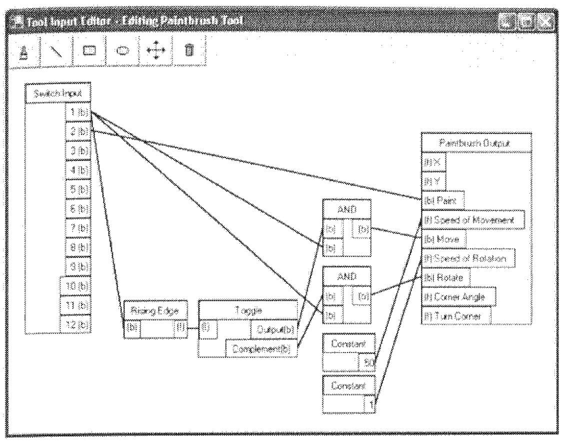

Fig. 2. An alternative input map for the paintbrush tool. In this case, two external switches are used to control the tool.

attached to a peripheral switch interface. One switch (Switch 1) is used to control movement of the paintbrush coordinates. The other switch (Switch 2) is used to turn painting "on" and "off". Only when painting is "on" does the brush mark the canvas as it moves. Switch 1 controls orientation and movement of the paintbrush alternately. First the user presses and holds switch 1 while the paintbrush direction arrow rotates, until the desired direction of movement is reached. Next, the user presses and holds Switch 1 again while the paintbrush moves in the selected direction, until the desired position is reached. If the Painting input is "on" during this process, a brush stroke will be drawn between the starting and ending points. This example illustrates the use of tool input configuration in the facilitation of drawing using special input devices.

The Paint My Voice program provides several audio input blocks for inclusion in tool input maps. Each of these blocks outputs a value determined by some property of a user-generated audio signal recorded through a microphone. Single channel audio data are recorded at a sampling frequency of 20.05kHz, with 16-bit resolution. Each audio input block calculates a different value from the recorded data. Blocks are included for calculation of audio pitch, intensity and zero-crossing rate (ZCR). Each of these blocks has a single floating-point output pin. Furthermore, blocks are included for the recognition of certain sounds, including the phonemes /o/ (as in "hoed") and /s/ as in ("see") and whistling. Each of these blocks has a single Boolean output pin.

The method of pitch estimation is based on the identification of series of harmonics in the short-term power spectrum of the audio signal, calculated by multiplying each sample in a 1024-point Discrete Fourier Transform (DFT) of the signal by its complex conjugate. All power-spectrum samples lower than a threshold value are zeroed. The estimated pitch is the maximum frequency for which more than a pre-defined fraction of the sum of all remaining samples in the power spectrum is contained in peaks located at integer multiples of that frequency.

Sound intensity is estimated simply by rectifying and summing the audio data in a 1024-sample window. The ZCR is the number of times that the audio signal changes sign in a 1024-sample window. The recognition of phonemes is based on the distribution of zero-crossings in the audio data.

Each audio input block used in a given input map is an instance of a distinct class. However, each such class is derived from a common base class. Audio capture and buffering is handled by static members of the base class, allowing each chunk of audio data captured to be shared between whatever audio input blocks are in use. Furthermore, where two or more audio input blocks have a computationally expensive operation, such as a DFT for example, in common, that computation is implemented in the static members of a common class [5] derived from the audio input block base class, and from which the specific classes are derived. This means that if no blocks requiring the operation in question to be carried out are included in the current input map, it will not be done. Moreover, if one or more blocks requiring the calculation to be performed are included, it will be carried out once and only once on each chunk of audio data.

4. OBJECT GENERATION

For certain users, in particular those with an intellectual disability, or with a profound physical disability, even relatively straightforward modes of input may prove inaccessible. The Paint My Voice program provides an additional novel means of drawing for such users – translation of individual sounds or utterances directly into visual forms of several different types. Simply by making sounds, the user can create unique picture elements, including trees, flowers and landscape forms. Once generated, an object can be exported as a 2-dimensional raster image to the canvas editor, where the user can position it as desired.

The rules for generation of objects from utterances rely on the calculation of several speech signal characteristics, in particular intensity, pitch, timbre, and the distinction between voiced and unvoiced speech segments [6].

Each tree object consists of a solid region of foliage, on top of a trunk. The rules governing the creation of a tree object from an utterance may be summarized as follows:

- Tree size is determined by the total signal energy in the utterance.

- The ratio of the heights of the trunk and foliage portions of the tree is a function of the ratio of the sum of the lengths of unvoiced segments of the utterance to that of voiced segments. The more of the utterance is voiced, the shorter the trunk in relation to the height of the foliage.
- The irregularity of the surface of the foliage is determined by the average spectral spread in voiced portions of the utterance.
- The variation in colour across the surface of the foliage is determined by variations in pitch during the voiced segments of the utterance.

A similar set of rules governs the generation of flower objects, except that flower head diameter takes the place of foliage height, stalk length that of trunk height and so on.

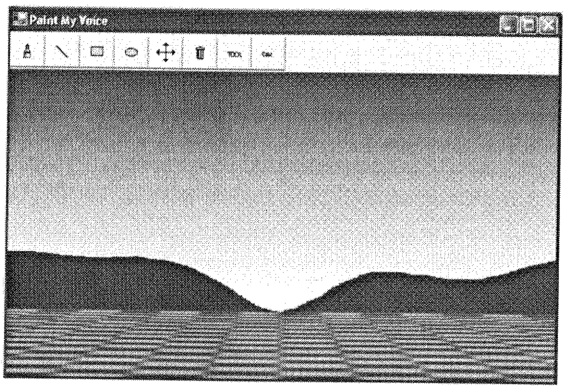

Fig. 3. The canvas editor displaying a background created by the object generator from user vocalisations.

Finally, a background landscape consisting of three elements, ground texture, hill-line on the horizon and sky colour can be created and exported to the canvas editor to serve as a backdrop for the user's drawing (see Fig. 3). Each of the three elements is generated in turn. The rules governing the creation of each may be summarised as follows:

- The hill-line is generated simply by creating a filled region spanning the full width of the canvas and bounded on the upper side by a plot of the waveform of a voiced segment of the input signal.
- The sky's colour is determined by the pitch of a voiced segment of the input signal.
- The ground texture is a green checkerboard pattern, the spatial frequency of which is determined by pitch. The gradient between dark and light regions is determined by the harmonic content in the input signal.

Once generated, the landscape can be exported to the canvas editor where it serves as a backdrop to the user's drawing.

5. USER TRIALS

Informal user trials were carried out with a small number of able-bodied subjects to get subjective feedback on the usability and appeal of the program. Subjects were asked to draw a simple figure with the paintbrush tool using only audio input. The feedback was encouraging, but this interface can be frustrating to a user who is able to use a mouse instead. All users responded very positively to the tree, object generation tools, reporting that the experience was both novel and rewarding. Trials are planned with disabled children in the school at Ireland's National Rehabilitation Hospital.

6. CONCLUSION

The Paint My Voice program facilitates creative expression, in the form of drawings, by children for whom traditional drawing materials are inaccessible. While initial testing suggests that the program shows much promise, streamlining of the interface is required before the program is considered suitable for everyday use. The program's object generation features have generated the most positive feedback. Development of the program is ongoing.

Acknowledgement

This work is generously supported by the Higher Education Authority of Ireland.

REFERENCES

[1] R.C. Simpson, H.H. Koester, "Adaptive one-switch row-column scanning," IEEE Trans. on Rehabilitation Eng., Vol. 7, No. 4, pp. 464-473, Dec. 1999.
[2] M.C. Su, M.T. Chung, "Voice-controlled human-computer interface for the disabled," IEE Computing and Control Engineering Journal, Vol. 12, Issue 5, pp. 225-230, Oct. 2001.
[3] E. Burke, Y. Nolan, A. de Paor, "An investigation into non-verbal sound-based modes of human-to-computer communication with rehabilitation applications," in Adjunct Proceedings HCI International 2003, Crete, June 22-27 2003, pp. 241-242.
[4] C.M. Eastman, "Vector versus raster: a functional comparison of drawing technologies," Computer Graphics and Applications, IEEE, Vol. 10, Issue 5, pp. 68-80, Sept. 1990.
[5] B. Stroustrup, "The C++ Programming Language," Addison-Wesley, Mar. 2002, pp. 228-229.
[6] J. Deller, J.G. Proakis, J.H.L. Hansen, "Discrete-time processing of speech signals," MacMillan USA, 1993.