A VOCALISATION-BASED DRAWING INTERFACE FOR DISABLED CHILDREN

Edward Burke\textsuperscript{ab}, Anamrao de Paor\textsuperscript{a}, Gary McFarby\textsuperscript{a}

\textsuperscript{a} National University of Ireland, Dublin, Dept. of Electronic and Electrical Engineering, Belfield, Dublin 4, Ireland.
\textsuperscript{b} National Rehabilitation Hospital, Rochestown Ave., Dun 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 “Paint My Voice” 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 a 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, sack-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 it 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 “♀”, “♂” and “O” 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 audio-related input blocks. 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 1 indicating 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 signals which consequently have no output pins. For instance, the output block for the paintbrush tool, features 9 input pins (as listed in Table 1).

<table>
<thead>
<tr>
<th>Pin</th>
<th>Name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>X coordinate</td>
<td>Floating-point</td>
</tr>
<tr>
<td>2</td>
<td>Y coordinate</td>
<td>Floating-point</td>
</tr>
<tr>
<td>3</td>
<td>Pressure</td>
<td>Event</td>
</tr>
<tr>
<td>4</td>
<td>Speed of movement</td>
<td>Floating-point</td>
</tr>
<tr>
<td>5</td>
<td>Moving</td>
<td>Boolean</td>
</tr>
<tr>
<td>6</td>
<td>Speed of rotation</td>
<td>Floating-point</td>
</tr>
<tr>
<td>7</td>
<td>Rotating</td>
<td>Boolean</td>
</tr>
<tr>
<td>8</td>
<td>Angle value</td>
<td>Floating-point</td>
</tr>
<tr>
<td>9</td>
<td>Turn through angle</td>
<td>Event</td>
</tr>
</tbody>
</table>

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

The Paint My Voice program provides several audio input blocks for inclusion in tool input maps. Each of these blocks outputs a voltage 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 20.056 kHz, 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 /s/ (as in “tooled”) and /z/ (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 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 audio 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.
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, 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 "E", "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 audio input 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 1 for pressed, 0 for unpressed) and one floating-point output for each of the X and Y coordinates of the mouse pointer. Another type of input block is a 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).

<table>
<thead>
<tr>
<th>Pin</th>
<th>Name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>X coordinate</td>
<td>Floating-point</td>
</tr>
<tr>
<td>2</td>
<td>Y coordinate</td>
<td>Floating-point</td>
</tr>
<tr>
<td>3</td>
<td>Pressure</td>
<td>Floatingle-point</td>
</tr>
<tr>
<td>4</td>
<td>Speed of movement</td>
<td>Floating-point</td>
</tr>
<tr>
<td>5</td>
<td>Moving</td>
<td>Boolean</td>
</tr>
<tr>
<td>6</td>
<td>Speed of rotation</td>
<td>Floating-point</td>
</tr>
<tr>
<td>7</td>
<td>Rotating</td>
<td>Boolean</td>
</tr>
<tr>
<td>8</td>
<td>Angle value</td>
<td>Floating-point</td>
</tr>
<tr>
<td>9</td>
<td>Turn through angle</td>
<td>Event</td>
</tr>
</tbody>
</table>

4. OBJECT GENERATION

For certain users, in particular those with an intellectual disability, or with a profound physical disability, even relatively straightforward inputs of audio 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. 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.

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.

- **Sound intensity** is estimated simply by integrating 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 base class, and whose 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.

A vocalisation-based drawing interface for disabled children...

[Image for Fig. 1: A simple input map for controlling the paintbrush 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 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. When the paintbrush tool's input map configuration is set, 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...
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


A SYSTEM FOR MONITORING PRESSURES AND SPINAL CURVATURE IN SPINALY INJURED PEOPLE IMMOBILISED ON A SPINAL RAFT

Ruairi de Fréin, Eoin Flinn, Annmarie de Paor, Edward Burke

National University of Ireland, Dublin, Dept. of Electronic and Electrical Engineering, Belfield, Dublin 4, Ireland
National Rehabilitation Hospital, Rochestown Ave., Don Lauphaire, Co. Dublin, Ireland

Summary

We present a system designed to study the pressure at various ‘hot spots’ on the back of the body and the deformation of the spine experienced by a patient when strapped to a spinal board, and the potential alleviation of both by the addition of an inflatable ‘spinal raft’ (or other similar device). In measuring pressure we devised a system of air-filled sacks interfaced with a PC. Each sack, placed under a particular key point on the body, is inflated until it faces just begin to separate and thus permits the pressure to be measured. The pressure reading is then captured and displayed by the computer. Seeking a non-invasive method of measuring the curvature of the vertebral column, we discovered that we could use a magnetometer to measure the vertical displacement of a magnetically-tagged vertebrae from a fixed horizontal plane. The results of our study will be employed by an independent body to determine the merits or demerits of the spinal raft.

1. INTRODUCTION

When a person is suspected to have sustained a spinal injury, emergency paramedics typically strap the patient down tightly on a rigid board, known as a spinal board. The purpose of this board is to immobilise the spine in transit to a hospital, lest any movement should cause further damage to the spinal cord. In certain cases, particularly when a spinal injury occurs in a remote location, a patient may remain strapped to the board for several hours before being seen by an appropriate specialist. As well as being uncomfortable for the patient, prolonged immobilisation on a spinal board can have serious long-term medical consequences, including the formation of pressure sores and flattening of the spine.

Also known as decubitus ulcers, pressure sores occur when pressure is maintained over a number of hours on an area of skin, preventing adequate circulation of blood leading to the death of skin cells in and around the point of contact [1,2]. Certain points on the back of a patient strapped to a spinal board are particularly at risk, including the back of the head, the shoulder blades and the buttocks.

In cases where pressure sores do develop, the first signs are typically already visible when the patient is released from the spinal board. The skin at the points in question at first appears red and tender and within a few days rotsways, leaving a large open wound. In addition to the discomfort associated with these lesions while they remain open, long-term problems persist even after healing. For example, the formation of a pressure sore on the back of the head may result in a permanent and unsightly bald patch.

For a patient strapped to a spinal board, the primary factors affecting the likelihood of occurrence of pressure sores are the magnitude of the pressure at each of the danger zones, the health of the patient (particularly skin condition and blood circulation) and the length of time spent on the board without repositioning [3,4].

The second problem associated with the use of spinal boards is flattening of the spine [5]. To minimise the chances of further damage occurring while the patient is on the spinal board, the spine should ideally be in as near as possible to neutral position. This is defined as the normal anatomical position of the head and torso that one assumes when standing looking straight ahead. Unfortunately, the spinal position of a patient strapped to a spinal board is often far from ideal [6]. In fact, even if the patient turns out not to have suffered a spinal injury, unnatural positioning of the spine on the spinal board can cause musculoskeletal damage [7].

![Fig. 1. The inflatable back raft on top of the rigid spinal board. Spread out to either side of the board are the velcro straps used to secure the patient.](image)

It is hoped that the use of a back raft might alleviate the problems discussed above. This inflatable device is placed between the patient and the spinal board. The raft’s cushioning effect is designed to redistribute the forces between the patient and spinal board, reducing the magnitude of the pressure at the points of likely pressure sore...