Mapping Strategies for Musical Performance

Andy Hunt
Department of Electronics
University of York (U.K.)
adh@ohm.york.ac.uk

Ross Kirk
Department of Electronics
University of York (U.K.)
ross@ohm.york.ac.uk

Overview

This article examines various strategies for mapping human gestures onto synthesis parameters for live performance. It describes some experimental work which compares three types of interface mapping for a real-time musical control task. The outcomes of the experiment suggest that certain forms of continuous multiparametric mapping are beneficial to successful computer-based musical instruments.

Introduction

Human beings have been making music with musical instruments for thousands of years. It is therefore perhaps surprising that many computer-based musical tasks are undertaken via interfaces which have only been developed over the last few decades (and often designed for office tasks). The richness of physical control required for performance with traditional acoustic musical instruments takes time to learn. In many computer interfaces this is often replaced by sets of *choices* that the user must continuously think about [Hunt, Kirk 1999].

This article begins by explaining the difference between two modes of thought - *analytical* and *holistic*. It then defines an explorative mode of interaction, termed *Performance Mode*, which is more suitable to real-time control than the many "choice-based" approaches. An experiment is described which compares three interfaces, using different mapping strategies, for a real-time musical performance task. The interface designs are explained, and the results of the experiments given along with the qualitative comments from the test subjects. Finally the implications for Performance Mode are considered for the designers of real-time musical interfaces[1].

1. Modes of thought for Musical Performance

It was mentioned above that many computer interfaces are choice-based. These choices may be in the form of menus, icons, lists, or even some types of gestural interpretation. All these interfaces depend on the user selecting from a set of options provided contextually by the computer. In other words the computer is in charge of the dialogue, and the operator is restricted to the current choices on offer. This forces users into a mode of thinking where they have to continuously read (or visually scan) the options on offer, then select the most appropriate response, then wait for the computer to move into the next set of choices. The user is constantly having to analyse the computer's options. This mode of "analytical" thinking is contrasted with a more creative mode of "holistic" thinking in the following section.

1.1. Cognitive Modes: Analytical / Holistic

We can say that humans are thinking in ‘analytical mode’ when their attention is directed towards the breaking down, or decoding, of information which they are perceiving. Analytical thinking is often directed towards a particular goal, for example trying to find a numerical solution to a problem. Parameters are studied one at a time, in sequential, logical order.

[1] This article is based on a summary of Andy Hunt’s DPhil thesis on interface design [Hunt, 1999] and therefore more details and background discussion can be found there. This article also expands on a short paper published at the EuroMicro conference, Milan 1999.

Reprint from : *Trends in Gestural Control of Music*, M.M. Wanderley and M. Battier, eds.

The ‘holistic’ mode is more difficult to define. One reason for this is that it usually exists beyond language, and as such language is a difficult tool for expressing it. This mode of thinking is likely to involve looking at the perceived object or subject as a whole. Individual details are less important than the overall effect. Many interrelated ‘information streams’ can be perceived at once and their relationships become apparent. The person thinking holistically can often be doing something else at the same time.

As an example, people can listen to music in either of these cognitive modes. In analytical mode, they may be following a score, or at least breaking down the perceived sound into identifiable instruments and patterns. Their purpose in this case is to draw conclusions, make comparisons, and produce data or information for future use. Usually, this data can be verbalised, for example "The cadence at bar 512 uses the chord of A minor for the first time in this section".

The same piece of music can be listened to in "holistic" mode. In this mode of thinking the listeners will perceive the overall effect of the music, maybe not even being aware of following individual instrumental lines. The effect on the listener is often an emotional one and can generate an almost subconscious desire to move with the music. There is no ‘goal’ or ‘information’ here, but some would argue that this emotional response was the whole point of music.

A similar cognitive dichotomy is well known in the field of acoustics where the phrase ‘holistic listening’ (sometimes called ‘synthetic listening’) is regarded as the normal everyday way of hearing a set of related harmonic components as a single coherent sound. With training it is possible to listen analytically and to ‘pick out’ individual harmonics which were hitherto fused into a whole sound.

Another example of the two cognitive modes in action might be to consider the different ways in which people can analyse sales figures. One way involves focusing on the individual figures, studying them in a logical order, applying mathematical calculations and producing a set of prediction figures as an output. Another way involves quickly scanning several graphs and making very general and high-level predictions about future trends based on graphical trajectories.

This article suggests that these two distinct styles of thinking can be equally applied to the interaction of humans with computers, and that computer interfaces are unhealthily dominated by ‘analytical’ interaction.

1.2. Interactive Control Interfaces

In stark contrast to the commonly accepted choice-based nature of many computer interfaces are the control interfaces for musical instruments and vehicles, where the human operator is totally in charge of the action. Many parameters are controlled simultaneously and the human operator has an overall view of what the system is doing. Feedback is gained not by on-screen prompts, but by experiencing the moment-by-moment effect of each action with the whole body.

The interaction of a player with a musical instrument could be summarised quite appropriately by considering that the player directly manipulates a complex musical object, thus exploring the sonic and tactile environment which in turn provides continuous feedback. Similar analogies could be made about a person driving a car.

The attributes of an instrumental real-time control system seem to be:

- There is no fixed ordering to the human-computer dialogue.
- The human takes control of the situation. The computer is reactive.
- There is no single permitted set of options (e.g. choices from a menu) but rather a series of continuous controls.
- There is an instant response to the user's movements.
- Similar movements produce similar results.
- The overall control of the system (under the direction of the human operator) is the main goal, rather than the ordered transfer of information.
- The control mechanism is a physical and multi-parametric device which must be learnt by the user until the actions become automatic.
- Further practice develops increased control intimacy and thus competence of operation.
- The human operator, once familiar with the system, is free to perform other cognitive activities whilst operating the system (e.g. talking while driving a car).

1.3. Characteristics of Performance Mode

The above attributes of a real-time control system form the beginnings of the definition of 'Performance Mode', a term which has been coined for this study. One of the main characteristics of such a mode of operation is that it allows humans to *explore* an environment in a continuous manner, rather than to 'perform a series of unit tasks'.

Explorative operation means that the user discovers how to control a device by exploring different input control positions and combinations, thus gaining an immediate response from the system. The user may appear to be 'playing around' with the control, but they are actually discovering hidden relationships between parameters within the system. Users feel that there is always something more that they could do; a movement they could learn or improve, an increase in subtlety, which would give them better control over the system. This is the experience of a typical acoustic instrumental musician; the instrument stays constant whilst the focus is on the improvement of the human player.

In addition to the list of attributes outlined in the above section, we will add the following suggested characteristics which enable performance mode to occur in real-time musical systems:

- The Performance Mode is the user's first point of contact with the instrument. Other modes (e.g. editing) are provided as secondary options.
- The primary feedback is sonic, tactile & kinaesthetic. Visual feedback is also provided, but advanced users make less use of this.

Cadoz [2000] describes instrumental gestures as having three functions: *ergotic* (controlling and manipulating the environment), *epistemic* (sensing or perceiving the environment) and *semiotic* (giving coded information to the environment). The epistemic effect is the feedback mentioned above, which comes primarily from the body moving the input device. Musical instruments also make sound which gives a second level of feedback to the user. The visual sense can also be used to provide feedback to the user, but too often this predominates at the expense of the sonic, tactile and kinaesthetic feedback.

It is proposed that artistic systems should, where possible, provide an explorative mode of operation. This could be the entire system interface, or just a small part.

2. The Design of Computer Instruments

Many devices are referred to as computer *instruments* but are not designed for live performance, rather for non-real-time editing operations. The increasing processing speed of computing technology means that real-time operation is now possible. However, many designers are simply translating the editing paradigm onto a faster processor in order to get 'live performance'. We propose here that 'fast analytical editing', whilst being a laudable goal in its own right, is completely different from creative performance. A human player requires an instrument that can be continuously controlled in a subtle manner without having to enter into traditional human-computer dialogue.

It should be noted that in a good system many of the facets of live performance *could* be present at the editing stage. Editing and reconfiguration generally involve a great degree of analytical thought, so it is appropriate that menu-options and iconic tools be present to facilitate such actions. However, the explorative operation outlined above should be applied to creative editing just as much as to live performance.

For example, should the task of 'creating a new sound' be entirely a case for navigating menu options and entering parameter values? Would it not be better to allow the user to have access to the same type of explorative direct manipulation techniques as outlined above? In this way, the act of editing becomes an active physical process, utilising instant feedback from gestures. Cognitive analysis and parameter adjustment can be used at any point *that the user feels is appropriate*, rather than being the only option for changing the sound.

3. Multiparametric Interfaces and Mapping Strategies

In section 1.2 we stated that in order for a human to explore a device in real-time we would require that:

The control mechanism is a physical and multiparametric device which must be learnt by the user until the actions become automatic.

and that:

There is no exclusive "set of options" (e.g. choices from a menu) but rather a set of continuous controls.

In other words for a device to permit Performance Mode to occur it needs to allow the user continuous control of several parameters at the same time.

Such multiparametric interfaces are rare in the computing world, but are abundant in the world of mechanical devices such as musical instruments and vehicles. Two particular concepts are now discussed which we believe are the key to the design and development of richer interfaces for computing systems.

- Multiple parameters should be coupled together.
- The system should utilise the human operator's energy.

3.1. Parameter Mapping in conventional interactive systems

Consider a violin and ask the question "where is the volume control?". There is no single control, rather a *combination of inputs* such as bow-speed, bow pressure, choice of string and even finger position. This is an example of a 'many-to-one' mapping, where several inputs are needed to control one parameter (see figure 1). Rovan et al [1997] refer to this as *Convergent Mapping*.

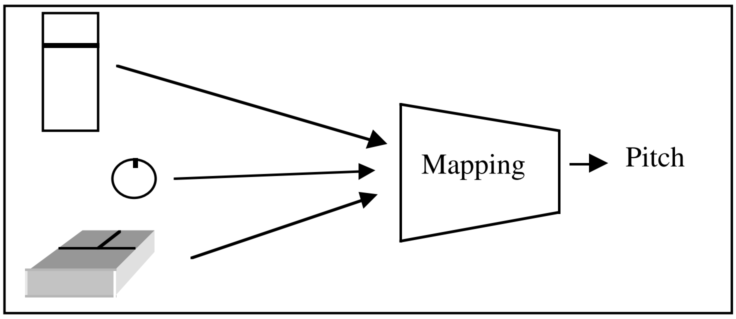

Fig. 1. Convergent Mapping; Many controls operate one parameter.

Again, considering the violin, ask the question "which sonic parameter does the bow control?" It actually influences *many* aspects of the sound such as volume, timbre, articulation and (to some extent) pitch. This is therefore an example of a 'one-to-many' mapping (see figure 2). Rovan et al call this *Divergent Mapping*.

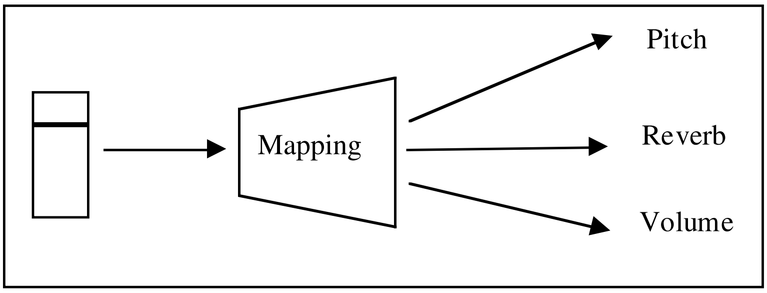

Fig. 2. Divergent Mapping: One control operates many parameters.

Human operators *expect* to encounter complex mappings, and yet so often engineers provide nothing but 'one-to-one' correspondences (for example a set of sliders, each controlling a synthesis parameter).

3.2. Use of Energy in conventional interactive systems

In many real-time devices (for example a violin, a bicycle, a clarinet or a drum-kit) the human operator has to *inject energy* or 'excite' the system before it will operate, and must continue to supply energy to keep it going. Then, the energy is *steered* through the system or *damped* (dissipated) in order to achieve the task such as playing a note or climbing up a hill.

These two operations (inject/excite & steering/damping) are often carried out by different conscious body controls (e.g. bowing with one arm and fingering notes with another, pushing bicycle pedals with the legs and steering with the arms, blowing a clarinet and using the fingers to key the notes). Even in motorised systems (the car being the most common example) the concept of injecting energy with one limb and steering with another holds true. A motor actually generates the energy, but its injection and damping is controlled by the driver.

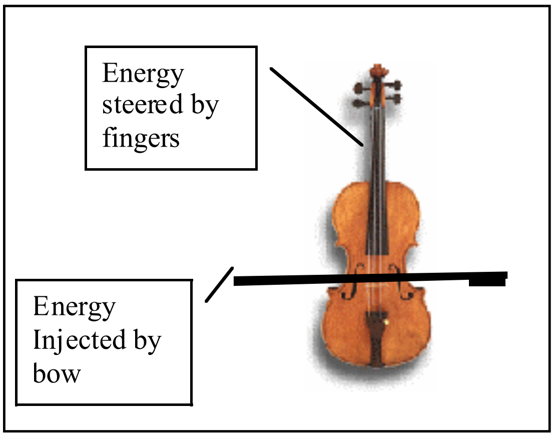

Fig. 3. Human energy input and control.

3.3. Further discussion of Mapping Strategies

The incoming parameters can be mapped into a variety of ways onto the synthesis variables. Two inputs can be *summed* or *averaged*. In this way both forms of input have an effect on the resultant sound parameter. In the violin example the pitch is controlled by a combination of the effects of finger position (on the string) and bow pressure. The finger position is the predominant effect, so we can also consider the *weighting* of each input parameter. The equation for the violin pitch might look like this:

Pitch = ((large weighting) x finger position) + ((small weighting) x bow pressure)

Note how this is a convergent mapping, but there are also *simultaneous* divergent mappings taking place. The bow pressure (which contributes to the pitch) also has an effect on the volume and the timbre of the sound, each of which is affected by a range of other inputs. We can therefore see that the resulting mapping of input parameters to sound parameters in a traditional acoustic instrument resembles a 'web' of interconnections.

Rovan et al [1997] describe this process in more detail for an acoustic clarinet. They introduce a new form of gestural interaction known as *biasing*. This is more than just a summing function; it describes the situation where one parameter needs to be activated to a certain level before another one can even have an effect. The example cited is of a clarinet's embouchure which needs to be set to an appropriate level before the 'air-flow' parameter has any effect [Editors' note: see the article by Cadoz and Wanderley in this volume].

The total effect of all these convergent and divergent mappings, with various weightings and biasing, is to make a traditional acoustic instrument into a highly non-linear device. Such a device will necessarily take a substantial time to learn, but will give the user (and the listener) a rich and rewarding experience. Many computer interfaces concentrate on simple one-to-one mappings and, though easier to learn, can give an impoverished experience to the player and listener. In our experimental work here we have attempted to define a relatively simple interface that exhibits some of these complex mapping strategies.

3.4. Design of Multiparametric interfaces

The characteristics identified in this article for allowing Performance Mode are thus:

- Continuous control of many parameters in real time.
- More than one conscious body control (or limb) is used.
- Parameters are coupled together.
- User's energy is required as a system input.

With a multiparametric interface each physical device (for example a slider) controls a variety of parameters. In other words there is no longer a one-to-one mapping between a control device and a system parameter. There will be more than one control device for the user to control simultaneously (for example, several sliders and a mouse). The parameters are *grouped* under each control device such that each control device has a distinct characteristic (e.g. the standard pedals in a car). To illustrate this imagine a design where moving the mouse to the left decreases the pitch *and* softens the timbre, whilst moving the mouse pointer upwards increases the volume *and* decreases the reverb level. In this example a circular mouse movement controls pitch, timbre, volume and reverb level, all at the same time.

The way that the parameters are grouped will affect which areas of the system parameter space can be covered. In our imaginary example it is impossible for the user to have a very loud sound with a high reverb level.

We now describe a set of user interface experiments designed to explore how human users react to having grouped parameters which steer them away from an analytical one-to-one control/parameter mapping to a more holistic performance exploration of the parameter space.

4. Comparison of User Interfaces for Real-time Control

A series of tests has been carried out at the University of York, UK, in order to study the effectiveness of different interfaces when used for a real-time musical control task. The data that was gathered was used to compare how a group of human test subjects performed in the exercise. Particular emphasis was given to comparing the results from different interfaces *over a period of time*. In other words the aim of the tests was to gather a set of data which measures how people respond to a range of interfaces and shows how that performance varies over time.

At least one of the interfaces chosen for the task needed to represent the commonly accepted way of controlling a system, so this used a mouse to select individual parameters and alter them. At least one of the other interfaces needed to be a more radical design which allowed the user to control multiple parameters at the same time in an explorative (holistic) manner.

4.1. Choice of Interfaces for comparison

A range of different interfaces was originally envisaged for the tests. The intention was to allow the user to perform a real-time task on those interfaces which are commonplace in computer music and to compare these with a new multiparametric interface operating on the principles outlined in the above section.

The following three interfaces were chosen for the study:

- A set of on-screen sliders controlled by a mouse.
- A set of physical sliders moved by the user's fingers.
- A multiparametric interface which uses parameter coupling and the user's energy.

They represent a series of stages – from the most commonly accepted through to the most radical.

In order for the tests to be compared in a fair manner it was important to define what parameters were being controlled and to have exactly the same parameters in each of the interfaces. All the sounds were made up of the following *four* parameters:

- Pitch
- Volume
- Timbre
- Panning

Each of the chosen interfaces is now described.

4.2. Mouse Interface

This interface consists of four sliders on a computer screen, one for each of the sonic parameters that can change. During the test, the player uses the mouse to move the sliders (see figure 4).

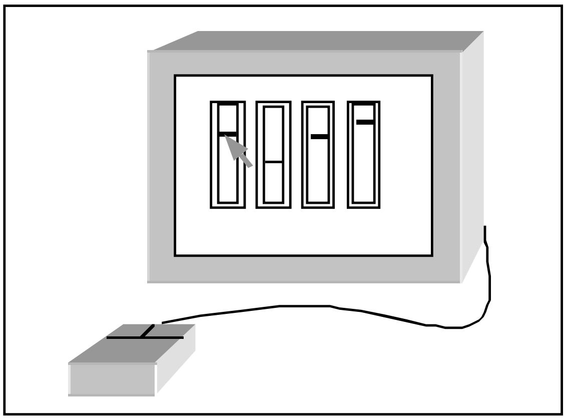

Fig. 4. The Mouse & Sliders Interface.

Each slider control can be 'dragged' to produce a trajectory or 'clicked' to produce a step change. The mouse can even be moved left and right *across* the bank of sliders and, if the button is held down, each slider will "snap" into position as the mouse is passed over it.

Some initial pre-trial studies of this interface showed that the only way it could be made viable was to 'pre-set' the starting positions of each slider control to correspond with the starting values of the sound in question. At least this way, the user had some chance of reproducing the sounds. Otherwise, too much time was spent trying to set each of the sliders into an appropriate starting position.

This interface could be deemed to be similar to the ubiquitous WIMP (Windows, Icons, Menus, Pointers) interface paradigm, but actually it is not a 'true' WIMP interface as it has no menus. The user does not have to *search* for the parameters, but simply move the mouse to the appropriate sliders. An interface with menus would actually slow down the whole process so as to make interactive continuous control impossible. This interface does not allow the user to change more than one parameter at a time, so it is not a multiparametric interface. The following section describes an interface which *does* allow the user to operate more than one of the parameters simultaneously, whilst remaining conceptually similar.

4.3. Sliders Interface

This interface uses four of the sliders on a Roland SC-155 sound module (see Figures 5 and 6). This was configured to send out MIDI information and thus control the sound algorithms on the University of York's *MIDAS* computer system (see section 5).

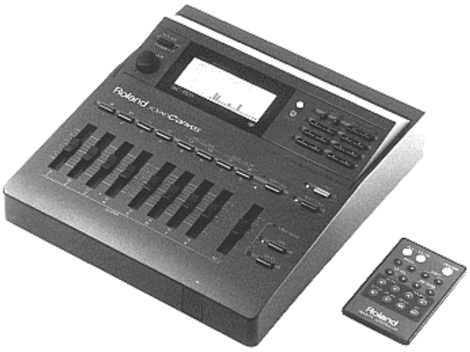

Fig. 5. The Roland SC-155 Sound Module.

Each slider controls a single sound parameter (i.e. a one-to-one mapping). The user can move each of the sliders independently and can thus simultaneously control all four sound parameters.

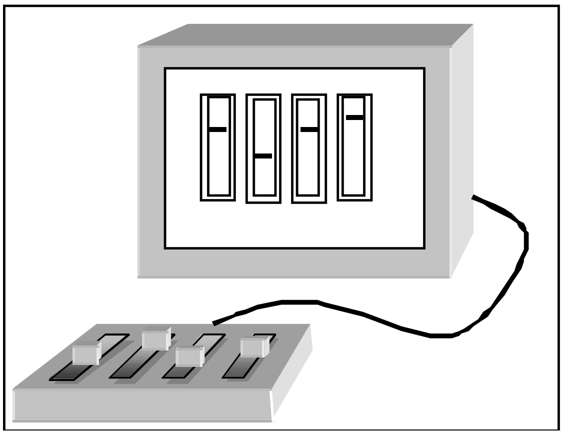

Fig. 6. The Physical Sliders Interface.

The slider positions are also shown on the screen, but the user does not *need* to look at the screen in order to use this interface.

Note that this sliders-based interface fulfils the first two requirements of a multiparametric interface outlined in section 3.3 (i.e. "many parameters" and "more than one body control"), but not the final two ("coupled parameters" and "use of energy"). Of course the user expends a small amount of energy in moving the sliders, but it is only the slider *position* that determines the state of the parameters. We therefore need to design an interface which fulfils all of the requirements in order to establish the effect of parameter coupling and energy use. The following section describes such an interface.

4.4. Multiparametric Interface

This interface uses the same hardware as interfaces 4.2 and 4.3 (the mouse and physical sliders on a sound module), but it uses them in two radically different ways. Firstly the system expects the user to expend some physical *energy* to continuously activate the system. Secondly, there is only one direct one-to-one correspondence (mapping) between a physical control and an internal sound parameter (for panning). All other mappings are complex. The multiparametric interface used in the study is shown in figure 7.

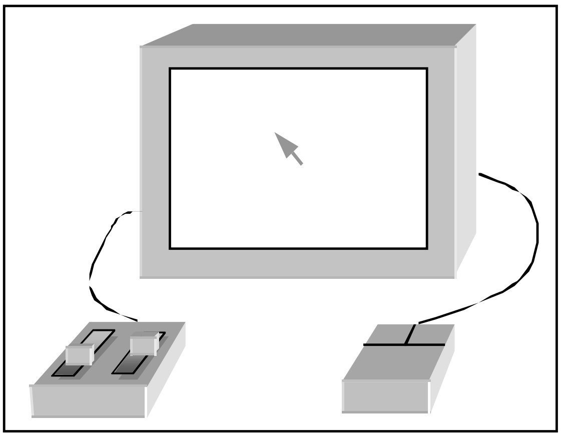

Fig. 7. The Multiparametric Interface.

The user finds that the computer screen is blank (in contrast to the two previous interfaces where the screen shows a representation of four sliders). Sound is only made when the mouse is *moved*. The sound's volume is proportional to the *speed* of mouse movement. This ensures that the user's physical energy is needed for any sound to be made, and that the amount of energy has an effect on the quality of the sound.

The volume, pitch, timbre and panning are controlled by *combinations* of the mouse position and the position of two sliders, as shown here and in figure 8:

- Volume = speed of mouse + mouse button pressed + average position of two sliders.
- Pitch = vertical position of the mouse + speed of movement of slider no. 2.
- Timbre = Horizontal position of the mouse + difference in the two slider positions.
- Panning = Position of slider no. 1.

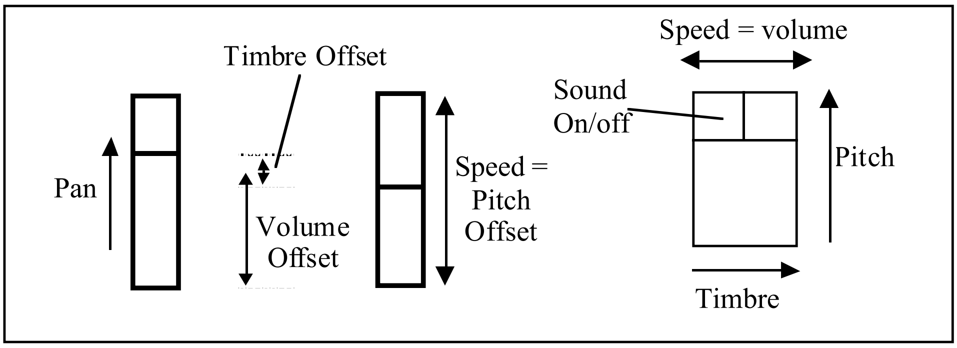

Fig. 8. Multiparametric mappings.

This ensures that there are several many-to-one mappings. Simultaneously there are various one-to-many mappings (e.g. slider 1 affects volume, timbre and panning). Two limbs are used, as the player has to use two hands – one on the mouse, one on the sliders.

There is no 'obvious' mapping of hand position to sound produced. The user must experiment. During the tests users tended to be somewhat baffled at first, because they could not find 'the volume control'. After a while (and a few verbal hints, such as 'try wobbling the mouse left and right, and listen to what happens) then they gradually developed a 'feel' for the interface. After more time most people finally began to think in terms of gestures and shapes, a holistic control, rather than a one-to-one analytical approach.

5. Implementation on the MIDAS system

The University of York UK's MIDAS system, running on a Silicon Graphics Indy machine, was used to construct the audio algorithms, the user interfaces and the data monitoring systems. For some of the interfaces, an external piece of hardware was used. The hardware was linked to the MIDAS system via a MIDI connection.

This section describes the MIDAS system, and points the reader to further information. It then outlines how the user interfaces were constructed, before giving details on how the system was configured for each of the tasks for creating sound, gathering data and analysing results.

5.1. Description of MIDAS

MIDAS [Kirk, Hunt 1996] is an acronym for the Musical Instrument Digital Array Signal-processor. It was conceived by Ross Kirk in the early 1990s as a test-bed for real-time performance control of audio signal processing algorithms. It has since been under development by Andy Hunt, John Tuffen and Kingsley Ash at the University of York. MIDAS allows users to manipulate a "tool-kit" of audio-visual algorithms for constructing interactive systems. It also provides a means of distributing these algorithms over a network of connected computers of different types, so as to maximise the available processing power.

MIDAS is based around the concept of the *Unit Generator Process (UGP)*. A UGP is a piece of code which handles an individual task, such as drawing a rectangle, establishing the current position of the mouse, or generating a sound output. The concept will be familiar to computer musicians in the form of audio units, such as oscillators and filters in programs such as *Csound*, but MIDAS UGPs are designed to run in real-time on distributed processors. UGPs can be connected together into *networks* which together perform a more complex audio-visual processing task. MIDAS is therefore capable of being configured into performing any possible synthesis method. Figure 9 is a graphical representation of a MIDAS network running on several different computing platforms.

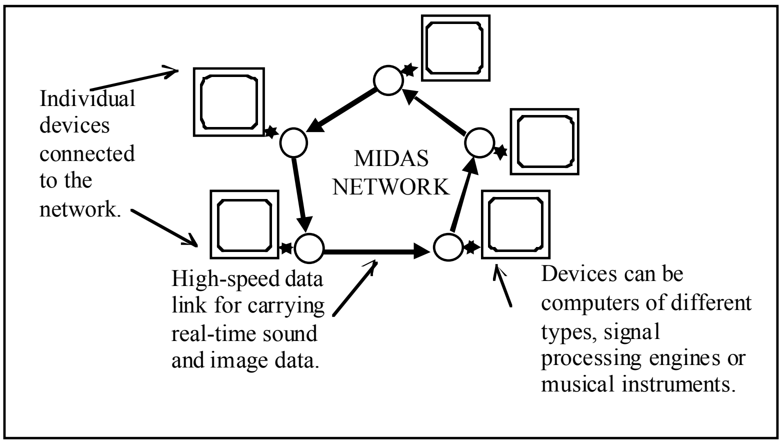
Individual devices connected to the network.
MIDAS NETWORK
High-speed data link for carrying real-time sound and image data.
Devices can be computers of different types, signal processing engines or musical instruments.

Fig. 9. The MIDAS system.

The entire network of UGPs is run from top to bottom within a single sample period. This ensures that data flows from the "upper" UGPs to the lower ones and can be updated at a maximum rate equivalent to the audio sampling period. Each UGP is buffered from the one above and the one below by a "data holder" (a variable). Therefore MIDAS does not need the concept of a separate "control rate" and "sample rate" since any lower rate data is just read repeatedly by a UGP which works at the audio sample rate.

The MIDAS web-site [Hunt 1997] provides access to the MIDAS code, and full details of the UGPs that are available. It divides them into a series of categories for easy reference and these are shown here:

Category	Example UGPs
• Audio	("oscillator", "stereo output")
• Data	("integer variable", "array")
• Graphical & User Interface	("rectangle", "slider")
• Maths & Logical	("adder", "inverter")
• MIDI Handling	("extract note", "filter channel")
• Musical Structure	("sequencer", "draw score")
• Signal Manipulation & Flow Control	("rate-of-change", "conditional sub-routine")

The benefits of producing the system are manifold. Not only do we now have a cross-platform architecture for audio-visual interaction, but it is easily open to development. It has allowed practical exercises in multimedia, MIDI systems and audio synthesis & control to be run on the Masters course at York. If anyone creates and publishes a UGP it can be employed by any other user in any network.

For this study it was possible to use the some of the UGPs developed by others and to write new ones to help create the entire test environment. The following sections describe how MIDAS was used in different ways to implement the user interface study.

5.2. Network for Sound Generation

The first role of the MIDAS system in this study is to generate sound that is continuously controlled by the four parameters *pitch, volume, timbre and panning*. Figure 10 shows the network of UGPs which carries out this task.

The sound source is kept extremely simple. It consists of a square-wave oscillator which has the two controls Frequency (*pitch* input) and Amplitude (*volume* input). The harmonically rich square-wave is passed through a low-pass filter whose cut-off frequency is controlled by the *timbre* input. Finally the filtered signal is panned in stereo according to the *panning* input.

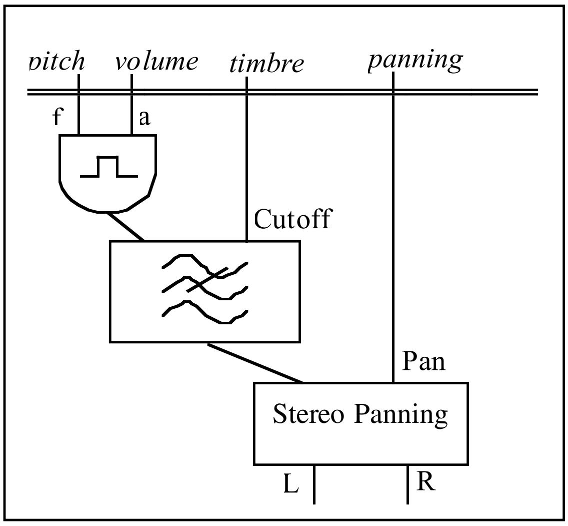

Fig. 10. The MIDAS sound generation network.

When MIDAS is required to play a sound, data values are fed into the inputs at the appropriate time. When the computer is meant to play a sound for the user to listen to, these data values are read from a set of data files by a C program. This program loads the data values onto each of the four sound parameter inputs then instructs the MIDAS network to run.

5.3. Network for Data Gathering

The same sound synthesis network is used when the human test subject performs on the interface. Data comes in from the interface, is stored for future reference and is also passed on to the appropriate input of the synthesis network. The storage is done by a set of four "data-logger" UGPs. As the data on each input changes, the new value is stored in a text file along with the current time (measured in audio sample ticks) for later analysis.

Figure 11 shows how the sound synthesis network is controlled from the changes in the user interface.

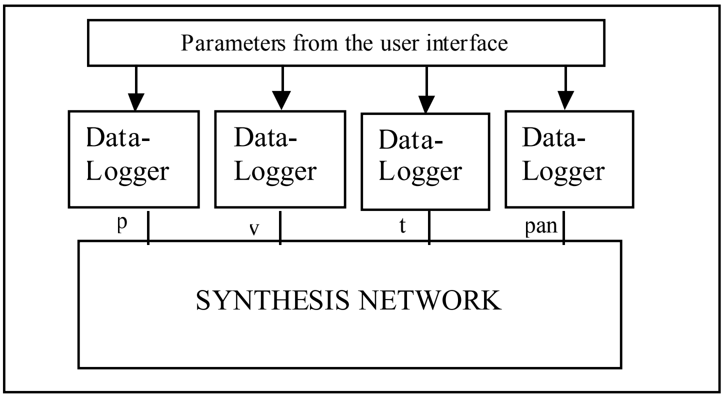

Fig. 11. The data gathering network.

5.4. Networks for the User Interfaces

The interfaces themselves are made up from real physical devices which feed data into the computer. The data is processed by a set of UGPs specific to each interface before being fed into the data gathering and sound processing networks (as shown above in figure 11).

5.4.1. The Mouse Interface

This consists of the standard Silicon Graphics mouse which is used to control four on-screen sliders (see section 4.2). These graphical sliders are in fact MIDAS slider UGPs. They allow the user to adjust the current value (between upper and lower limits) by moving the mouse pointer over the slider area when the button is held down. This means that the slider bar can be "clicked" into position, or "dragged" up and down. The current value of the slider is sent out of the UGP as a data value into the data-logger UGPs and on to the sound synthesis engine (see figure 12).

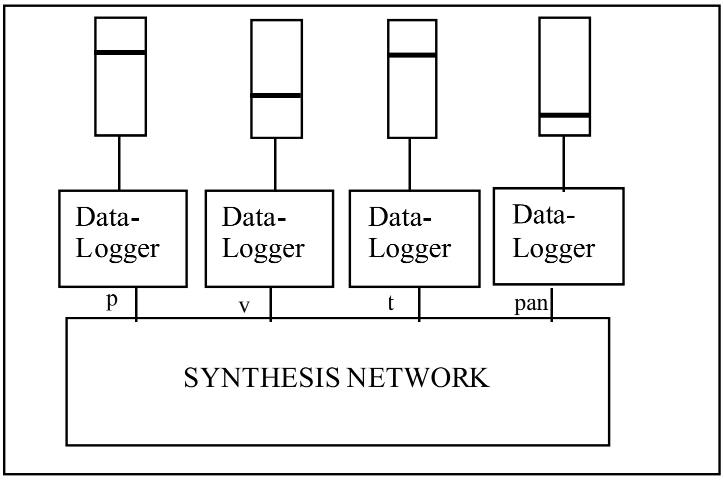

Fig. 12. The "mouse" interface network.

5.4.2. The Sliders Interface

This is an extension of the above network which allows the on-screen sliders to be controlled from a bank of physical sliders on an external MIDI device (see section 4.3). This is made possible because the "slider" UGP can be controlled not only by the mouse pointer, but by a data value injected at one of its inputs. Figure 13 shows the UGP network which creates this interface.

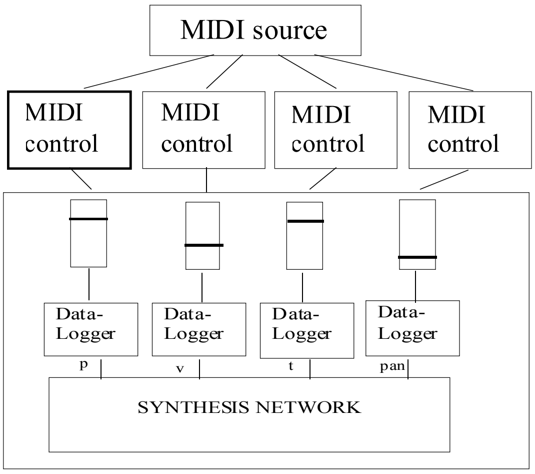

Fig. 13. The "sliders" interface network.

The Roland Sound Canvas SC-155 has a built-in bank of physical sliders which can be configured to send out MIDI "volume control" messages. Each slider sends out a message on a different MIDI channel so they are distinguishable when sent down a single MIDI cable. These volume messages are picked up by specially written "MIDI control extract" UGPs and are fed into the slider UGPs. Thus any movement on a physical slider causes a corresponding slider on the screen to move to the same position. In addition the result of moving that slider is stored in a data file and causes a change of sound from the synthesis engine.

5.4.3. The Multiparametric Interface

As described in section 4.4 this interface consists of two physical devices - the Silicon Graphics mouse and two sliders of the SC-155 module. The module communicates with MIDAS via a MIDI cable as with the "sliders" interface, outlined above. The mouse is read by the MIDAS "window_manager" which makes available the current mouse position and button state to any UGP that requires it.

Where this interface really differs from the other two is in the *processing* of the inputs before they reach the sound engine (see figure 14).

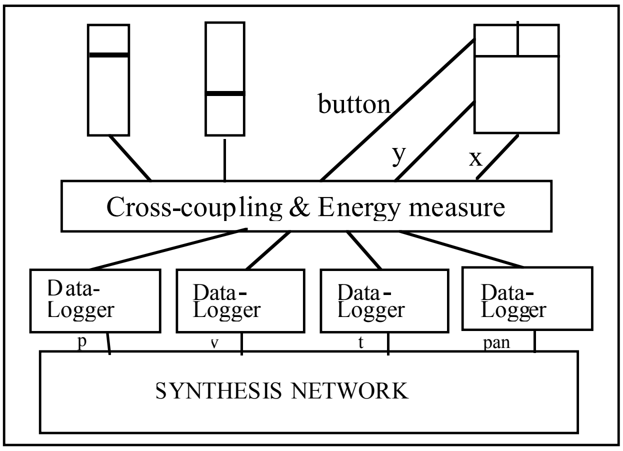

Fig. 14. The "multiparametric" interface layout.

There are no "slider" UGPs and so there is no visual representation on the screen. All the user sees is a blank window containing the mouse pointer. A network of UGPs is used to cross-couple the user's inputs into the four sonic parameters required by the sound engine. This is the "mapping" outlined in section 4.4. The user inputs are gathered from the mouse (x_position, y_position and left_button_state) and from the MIDI port (volume continuous controllers on channels 1 and 2). Figure 15 shows the set of interconnections that implement the mapping function. Two special UGPs were developed for this purpose.

The first is the "speed" UGP which outputs the *rate of change* of the input variable. This is central to getting a measure of the user's energy. In the network below it is mainly used to transform the speed of mouse movement back and forth in the x dimension into a volume control.

The second is the "range" UGP which scales any input range into any desired output range. This is required, for example, when implementing the "volume" input from a combination of slider position and speed of movement in the x direction.

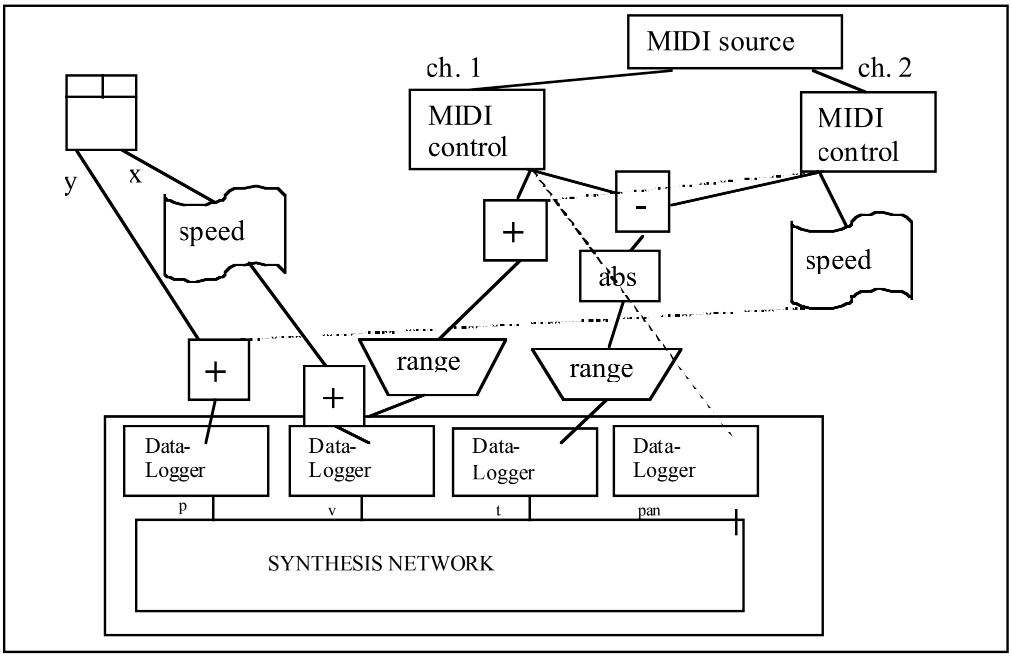

Fig. 15. The "multiparametric" mapping network.

5.5. Network for Analysis

MIDAS was also used in the analysis of the test data. An identical synthesis network was used to play the original tests and then the files recorded from the user. In this way a human marker was able to compare each test with the "ideal" sound played by the computer. Section 6.3 describes in more detail how the tests were analysed.

6. Cross-sectional User Interface tests

The purpose of the cross-sectional tests was to compare the performance of several users on the three interfaces. Each user was asked to spend approximately 15 minutes on each interface, making up a session of 45 minutes duration. Within each 15-minute period they undertook to listen to and reproduce 24 sound examples. These sound examples were the same for all three interfaces. Users came back after several days to repeat the entire process and then again a few days after that. Each human subject therefore experienced (24 sounds x 3 interfaces x 3 sessions) = 216 tests altogether.

There are many variables in these tests, but every example is a comparison over four parameters of what the computer plays and what the human subject manages to perform. The results have been studied to compare how different human subjects respond to each of the interfaces and how their performances varied with time over the three sessions. We will also see how the results are affected by the test complexity, for example the number of parameters that are altered simultaneously.

6.1. Musical Test Examples

Each test set consists of 24 sound examples each of which lasts between 2 and 4 seconds. They vary in complexity and can be notionally divided into three groups (although the user is not made aware of these groupings).

Group A (tests 1-8) contains sounds which have simple stepwise changes in one parameter only while the other parameters are held constant (see figure 16).

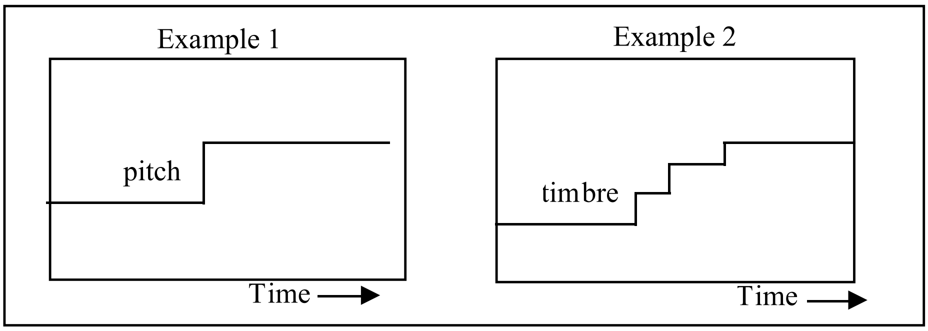

Fig. 16. Group A sounds: stepwise uni-parameter changes.

For example the pitch may step up a tone after 1 second while timbre, volume and panning do not alter (see Example 1 in figure 16). Towards the end of this group, there may be several step changes in a single parameter (see Example 2).

Group B (tests 9-16) introduces "trajectories" for a single sound parameter. These are perceived by the user as more of a continuous change. Towards the end of this group more than one parameter changes during the test, but *not at the same time*. Figure 17 shows examples from either end of this group.

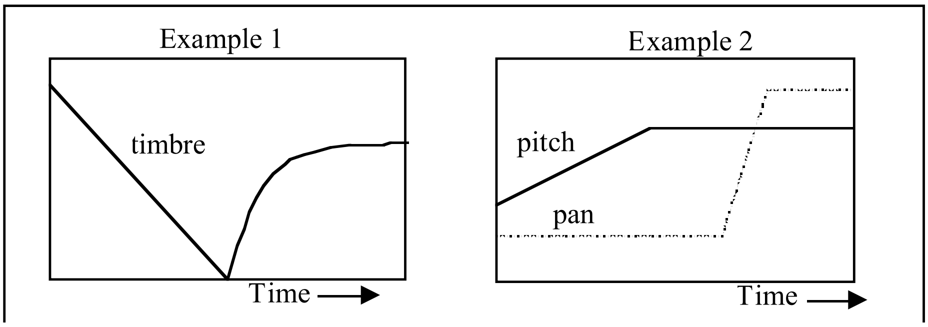

Fig. 17. Group B sounds: continuous non-simultaneous changes.

Group C (tests 17-24) consists of sounds where more than one parameter is changing at once (see figure 18). Towards the start of this group just two parameters change together, whereas for the last few all four parameters are moving with their own trajectories.

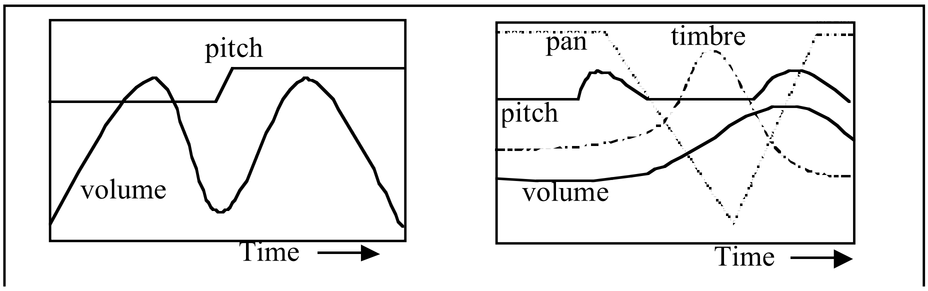

Fig. 18. Group C sounds: continuous simultaneous changes.

Comments have been made that when listening to some of these sounds they take on a specific character e.g. "a wolf-whistle effect that gets brighter while spinning from left to right". This tendency to mentally "fuse" several concurrent sonic events may have some bearing on the later analysis.

The tests were created in a variety of ways. Some were produced on the interfaces themselves. Others were typed in as breakpoint files. Another set were individually constructed from a combination of those two methods. The key point is that all the tests are the same for *every user* and for *every interface* and so can be used to compare individual differences in performance.

6.2. Description of the Test Environment

The test subjects sat in front of a Silicon Graphics computer terminal (see figure 19). The standard QWERTY keyboard was moved to one side, but the "spacebar" was still accessible by the subject's left hand. In front of them was the particular interface being tested. They wore a pair of headphones that allowed them to hear the tests, but also to hear the instructor's voice if they needed any help.

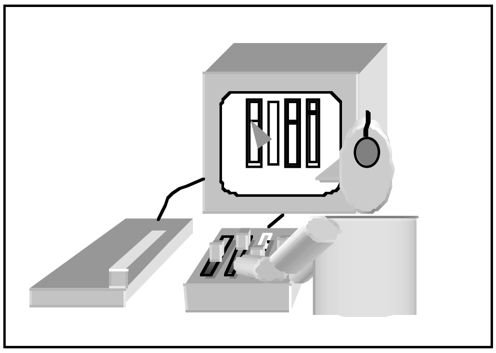

Fig. 19. The test environment.

Each test batch (i.e. 24 sounds on one interface) was completely automated. The MIDAS program co-ordinated playback of each sound and the gathering of the user's performance data in response to that sound. It achieved this by giving the user recorded instructions such as:

"Next test. Press the spacebar to hear the sound"

<user presses spacebar>

In other words the test subject steps through each stage of the test by tapping the spacebar (on the QWERTY keyboard). This ensures that the user is in control of the overall timing of the procedure. The instructions get shorter for the subsequent tests so that they take up less time as the subject gets used to the procedure.

6.3. Methods of Analysis

Every test result was stored on the computer and later given a score by a human marker.

The marker's job was to listen to the original sound, then to the performed sound (as many times as necessary) and give a mark out of 10 for each of three categories:

- Timing Accuracy (how do the timings of the sonic events match the original?)
- Parameter Accuracy (how near to the original are the parameter values?)
- Trajectory Accuracy (how well do the parameters move compared to the original?)

The final outcome for each test is a percentage score (where 100% means that the sound has been apparently perfectly reproduced). Human marking of audio material is normally considered essential for music exams, competitions, recitals at college, and feedback in music lessons etc. A single human judge (the author) marked the entire range of (216 per subject x 16 subjects) = 3456 tests. This took a large investment of time, but it meant that a consistent marking scheme was established – with no problems of equating marks between separate judges. A moderator was hired to randomly select several test sets and to mark them. The results of this process showed that there was about 85% correlation between the marks given by the original judge and those given by the moderator.

6.4. Taped Interviews

Every subject was interviewed after each set of tests on each interface. They were asked how they felt about their performance and the interface. Towards the end of the session they were asked to sum up how they had done overall and to compare the different interfaces. All of these interviews were recorded on audio cassette and later transcribed. They are summarised in section 8.

7. Results of the Cross-Sectional tests

A selection of graphs is presented to summarise the overall findings of the study. Particular emphasis is given to the description of how the tests of different complexity are handled on each of the three interfaces.

7.1. Results across all subjects and tests

The graph shown in figure 20 is a summary of the entire cross-sectional study. The horizontal axis shows time in terms of session number. The points on the graph represent the average score for *all subjects* and for *all sound examples* on a particular interface. The points on the graph are joined by lines for visual clarity, although it should be made clear that there is no "continuity of information" between sessions.

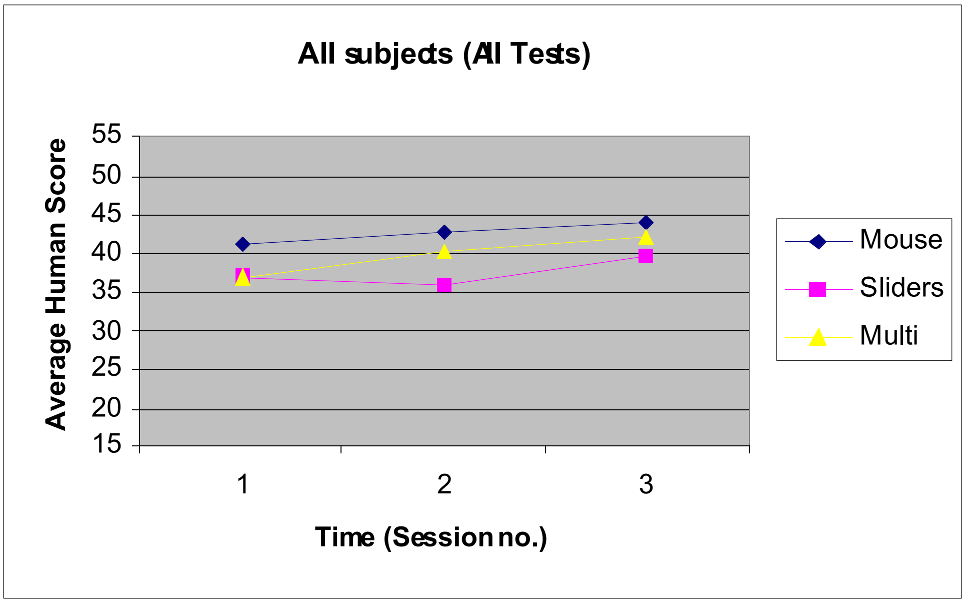

Fig. 20. Summary of the cross-sectional tests.

This graph appears to show that each interface overall has an upward trend over the three sessions. In other words the subjects are learning how to do the tests and how to use the interfaces. The rates of learning appear to be different for each interface. The mouse interface appears to give the best overall scores, but it is being "caught up" by the multiparametric interface. We could speculate about whether it would match or even overtake the mouse given a few more sessions. The sliders interface gives the lowest overall marks! By the third session it seems to make a recovery but is still overall the least successful interface.

7.2. Results for Tests of Different Complexity

In the above summary, we have averaged together the scores of *all* the tests, from the simplest through to the most difficult. One of the aspects to be investigated was the success of the different interfaces on the various levels of test complexity.

In section 6.1 the tests were described as fitting into three groups:

- Group A (simple step changes in one parameter).
- Group B (continuous changes, some with more than one parameter but not simultaneously).
- Group C (simultaneous parameter changes).

Figures 21 to 23 show the average results across all test subjects for tests in each of these groups.

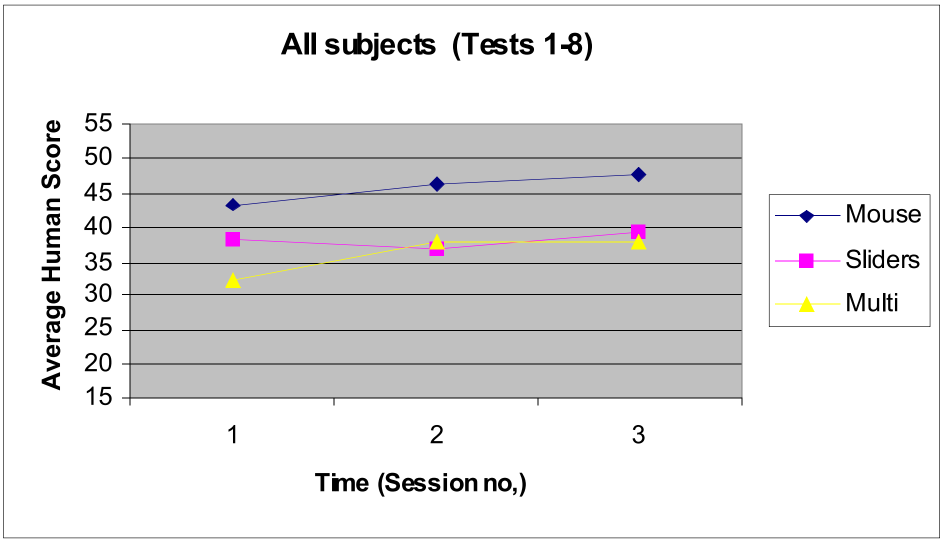

Fig. 21. Summary of the Group A tests.

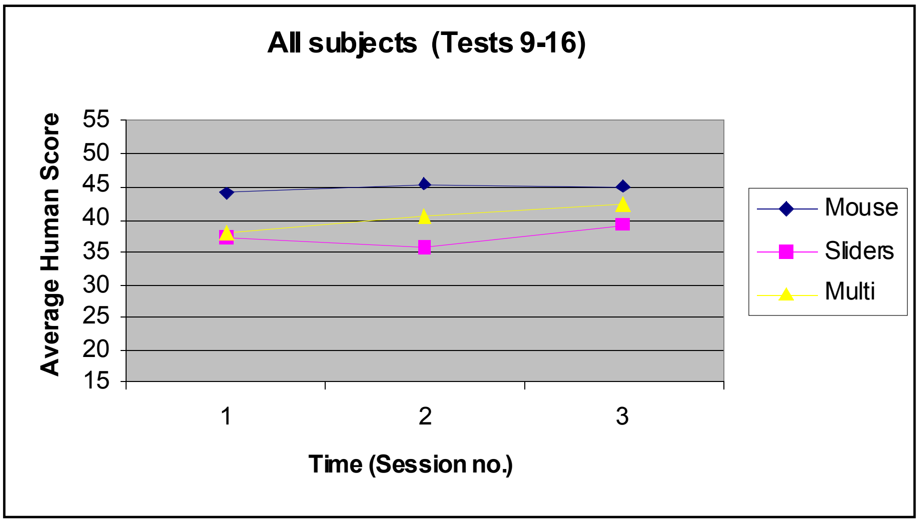

Fig. 22. Summary of the Group B tests.

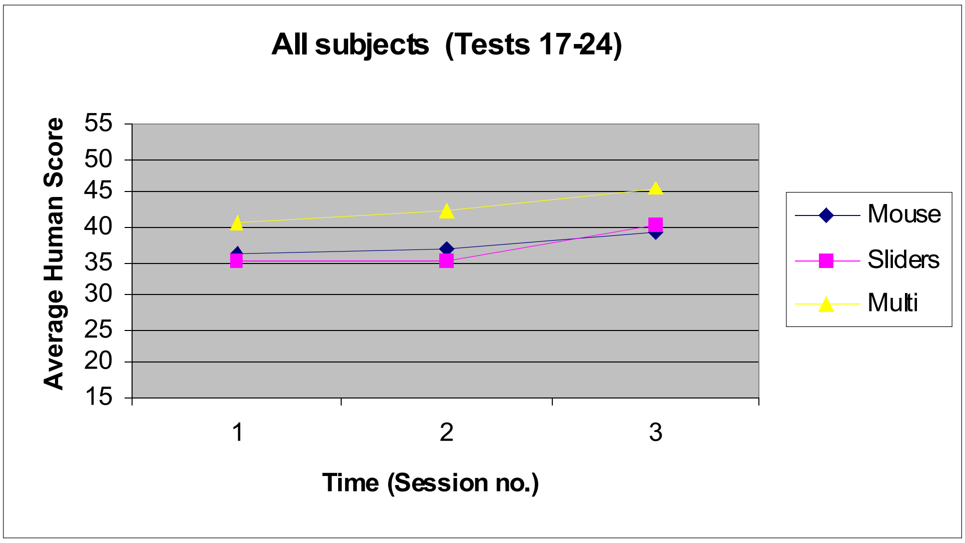

Fig. 23. Summary of the Group C tests.

A comparison of these graphs reveals a very different story to the overview shown in figure 20. The mouse interface is the clear winner for Group A and Group B tests. However, as soon as there is more than one parameter changing simultaneously (Group C) the multiparametric interface become the best interface. A comparison of graphs for Groups A and B shows that the multiparametric interface is better than the sliders for continuous changes (Group B), but they are similar in performance for the simple step changes (Group A).

In summary these graphs show that the mouse interface is best for simple step changes, but that only the multiparametric interface allows a good average performance when more than one parameter changes simultaneously. In fact the best scores encountered for the multiparametric interface are on the most difficult tests. Perhaps a "complex" interface is needed for a complex test! This is discussed in more detail in section 10.3.

8 Analysis of Taped Interviews

The taped interviews with the test subjects have been analysed for consistent comments given by different people. The headings for the following eight sections represent the essence of these comments, and they are each supported by some example quotes from the interviews.

8.1. The mouse is easier to start with, but has obvious limitations

"Being restricted helps you to concentrate" (Adrian, session 1)

"It's good that it "prompts" you with where the sliders start" (Annette, session 2)

"It's easiest to use, but I've no idea how I get simultaneous moves" (Andy, session 1).

"I feel comfortable with the instrument's limitations" (Steve, session 3).

"It's easy for the single parameter changes" (David, session 3).

"I like the way it helps you to start off, but it definitely has much less control than the other interfaces" (Elena, session 3).

"This was fine at first" (Gregor, session 1), *"I've got as far as I can get with this"* (Gregor, session 3).

"You can learn this one fast, but there's not much depth to it" (Jamie, session 3).

8.2. Some users preferred the sliders interface on certain sessions

"I would choose this one (multiparametric)" (Andy, session 2). *"I'd probably choose this (sliders) as my favourite"* (Andy, session 3).

"This seemed easier today. I do feel I've improved" (Ross, session 3).

"I'd definitely prefer this interface! This is my favourite." (Ian, sessions 1 & 3)

"I like this one best" (Brenda, session 2), *"I'm comfortable with this , but I prefer the (multiparametric) one today"* (Brenda, session 3).

8.3. These same users tended not to like the multiparametric volume

"Volume is difficult. I'd prefer it on a slider" (Andy, session 2)

"I couldn't get used to wobbling the mouse. This is really alien!" (Ian, session 1)

"The wobbling really put me off" (Brenda, session 1)

8.4. Sliders interface causes a mental block/confusion

"I had to think 'what does the pitch do?", "what does the timbre do?" (Adrian, session 2)

"I don't like these! I'm having to concentrate on one or two things; I can't do all four" (Louise, session 2)

"I HATE this one! While it's counting in I'm having to think where each slider needs to be. Then I forget the sound." (Annette, session 2).

"I do find this difficult! Breaking it down into four distinct parameters" (Mark, session 2)

"I cannot get on with this at all! I'm trying to think of too many things at once; there's too many things for my mind here. The interface forces you to decompose the sound and I don't think our minds are made for decomposing stuff like that" (Tony, session 3).

"I find I have to change each of the parameters sequentially, even though they can be physically controlled together" (Steve, session 2).

"It sound silly, but I had a panic halfway through about which slider controls what" (Steve, session 3)

"I just can't be bothered with this one. I keep looking at the sliders and thinking 'what???' " (Gregor, session 2).

"I don't like this. There's too much coming at you" (Paul, sessions 1& 3).

8.5. Multiparametric allows spatial thinking

"This feels multi-dimensional, gestural. I sometimes found myself thinking of a shape" (Adrian, session 1)

"The 'hard' sound sounded like shapes in the screen. It felt more natural and I could think in gestures. You can almost feel the performance space" (Mark, session 1)

"I'm not thinking of timbre as a 'parameter' like I do with the sliders - but rather that it's 'sharper' over this side and 'duller' over this side" (Mark, session 2).

"I could concentrate on the performance without worrying about the actual mechanics of it" (Steve, session 1).

"I could warm to this. I like the gesture. I found myself on a couple of those sounds thinking diagonally! I didn't do that on the other interfaces. Vibrato is circular! You start to think gesturally." (David, session 1).

"I felt like I was 'thinking' too much about this. It's like my fingers knew where to go but the way I was thinking got in the way" (Brenda, session 1)

8.6. Multiparametric interface elicits subconscious control

"You can use your unconscious to play it after a while. You can forget about the interface so that you can concentrate on the sound" (Jamie, session 3).

"This feels free - which is good - but in a way like I'm not completely in control" (Louise, session 1)

"There's less mental load. All the controls are in one, so it's easier to position" (Annette, session 2)

"Felt easy to use, but felt more out of control" (Andy, session 1).

"It became more like driving a car - in that you've got physical actions that you can sort of get on with, leaving you free to think of other things" (Andy, session 2).

"This is a lot better [than the other interfaces] *even though I felt out of control"* (Steve, session 1)

"I felt like I knew where to move it. I didn't have to think. My brain was free enough to be distracted by the workmen [external noises which Gregor didn't notice when doing the other interfaces]" (Gregor, session 1).

8.7. Multiparametric interface is fun

"This is really good fun! Even when you're not doing so well!" (Tony, session 2).

"One movement controlling several things is more fun. It's not like a task - it's like playing an instrument" (Gregor, session 1).

"You're not so worried about getting it right. You can just say "ah well" if it's a bit off and then adjust it. It's less technical" (Mark, session 2)

8.8. Multiparametric interface has more long-term potential

"This is intuitively easier. I appear to have more degrees of freedom. It's more closely mirroring the way the brain is working" (Tony, sessions 2 & 3).

"This interface has possibilities. I'd choose this over the long term" (Steve, session 3).

"Yeah - I like this the best; definitely! It's a lot freer - more flowing" (Adrian, session 3)

"This is more of an instrument than a device. I'd like to use this not for tests but just to play around on" (Louise, session 3).

"I'd like to play this freehand - not in a test setting" (Annette, session 3).

"It's the easiest of the three. It has the musical 'edge'. I'm controlling one thing (mouse) to get more than one dimension, whereas with the sliders I'm moving 3 objects if I need 3 dimensions" (David, session 3).

"You have more freedom. Your mind isn't restricted in a visual way. It's a blank screen." (Elena, session 3).

"This is the best interface. You've got more freedom, and there's so much potential if I could just get the hang of it" (Paul, session 3).

To sum up, the majority of users enjoyed playing the multiparametric interface and thought (quite enthusiastically) that it had the best long-term potential. This was because it caused the least mental load and allowed spatial thinking. In contrast the sliders caused mental overload and forced too much "decomposition" (analysis) of the sound. The mouse interface was popular for the simple tasks but everyone realised its limitations. There were a small group of people who much preferred the sliders, and they tended to be the group that commented on not liking the "wobbling" volume control of the multiparametric interface.

9. Longitudinal User-Interface Tests

The above results are very interesting and provide the basis for an interesting discussion about the nature of the three interfaces and how people react to them when partaking in tests of varying complexity. However we predict that, over time, those interfaces based on **non**-analytical modes of operation will yield a greater accuracy and flexibility of control. Based on observations of performing acoustic musicians and car drivers we would not expect the results to show themselves **instantly**. Instead, we would expect a general improvement over time, but a lot of practice would be needed. It is predicted that at some point (in a user's practice schedule) the non-analytical styles will yield more accurate results than the directly analytical paradigms. It is not necessarily within the first few sessions that this will happen.

Therefore it was decided to run a second study, using fewer test subjects, but over a longer period of time. These "longitudinal tests" would provide a more detailed time axis on which to compare the performance on the three interfaces. The following sections describe the format of the tests and present the results in a graphical format.

9.1. Overview of Longitudinal tests

The tests were designed to plot the progress of three subjects over ten sessions. The test environment was exactly the same as for the cross-sectional tests and the same three interfaces were compared. Nine test sounds were used in each set so that sessions lasted about 15 minutes in total for the three interfaces. The results were plotted on three dimensional graphs so that the effect of test complexity and practice time can be seen on the scores for each interface.

9.2. General form of the 3D plots

Figure 24 shows the generic form of all the subsequent three-dimensional plots.

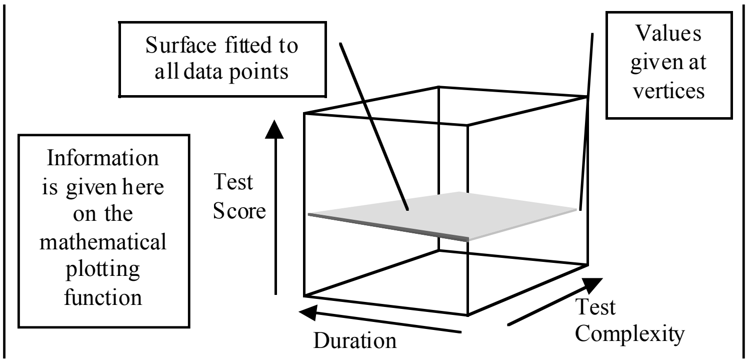
Surface fitted to all data points
Values given at vertices
Information is given here on the mathematical plotting function
Test Score
Duration
Test Complexity

Fig. 24. Structure of the 3D graphs.

The test scores are plotted on the vertical axis, with a ‘perfect’ score of 100% being at the top of the cube. The axis labelled *Test Complexity* is simply the test number (1-9), as the tests are numbered numerically in increasing order of parameter complexity. The labelled *Duration* is the session number and thus represents increasing user contact time with the interface.

Once the data points have been plotted, the MATLAB code fits the points to a curved surface using a Quadratic Surface function. The values for this function are given on the left of the graph. The factor labelled ‘Squared Residuals’ is a measure of how far the data points lie outside the surface. In other words it is a measure of statistical ‘fit’, with a value of 0 meaning that every point lies precisely on the surface.

Finally the data values at the corners of the surface are printed. This is useful for numerically comparing more than one graph in order to see how the surface differs, for example, from one interface to another.

The 3D graphs are now presented - one for each interface.

9.3. Results for each interface

The results across all the human test subjects are encapsulated into the following three graphs. The data from the three subjects was averaged before plotting each graph.

Figure 25 shows the first of the 3D plots. This graph encapsulates *all* the data collected for the *mouse* interface during the longitudinal tests.

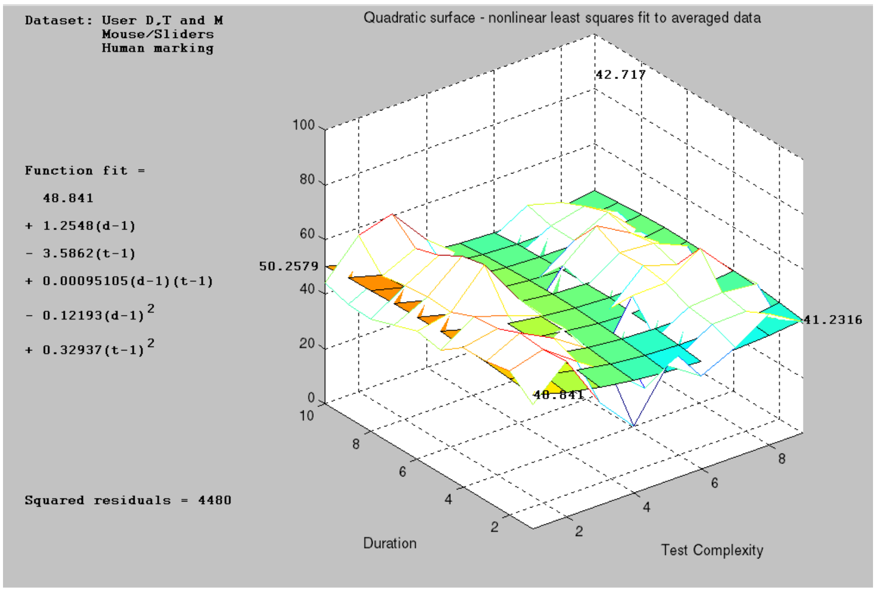

Fig. 25. ‘Mouse’ interface data from longitudinal tests.

This graph shows that the mouse does indeed give a reasonably ‘flat’ response over all the tests. There are signs of a very small improvement over time (e.g. an average of 48.8 in session 1, rising to 50.2 after ten sessions). Note the upward bend in the plane that indicates that the best scores are for the simplest few tests.

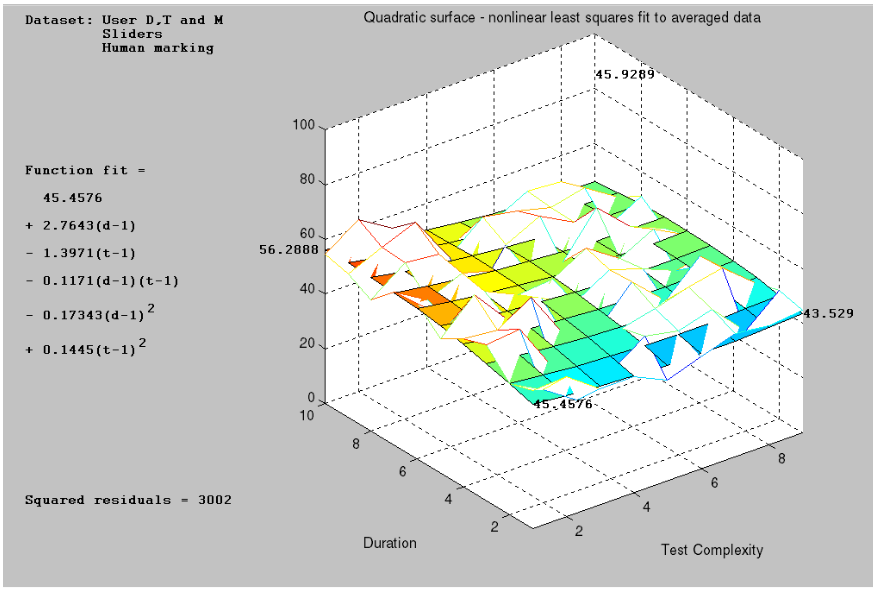

Fig. 26. ‘Sliders’ interface data from longitudinal tests.

Figure 26 shows a similar plot but for the sliders interface. The angle of the surface allows some immediate comparisons to be made with the mouse interface plot above.

- For the simplest tests the initial scores are lower than the mouse.
- There is a rapid learning curve for the simpler tests which means that at the final session the score is much higher than the mouse.
- The sliders perform slightly better than the mouse for the more complex tests.
- The learning curve is only slight for the more complex tests.

This indicates that the sliders feel a bit more difficult than the mouse to start with, but a significant amount of learning can take place for the simpler tests.

Figure 27 shows the same type of plot for the multiparametric interface.

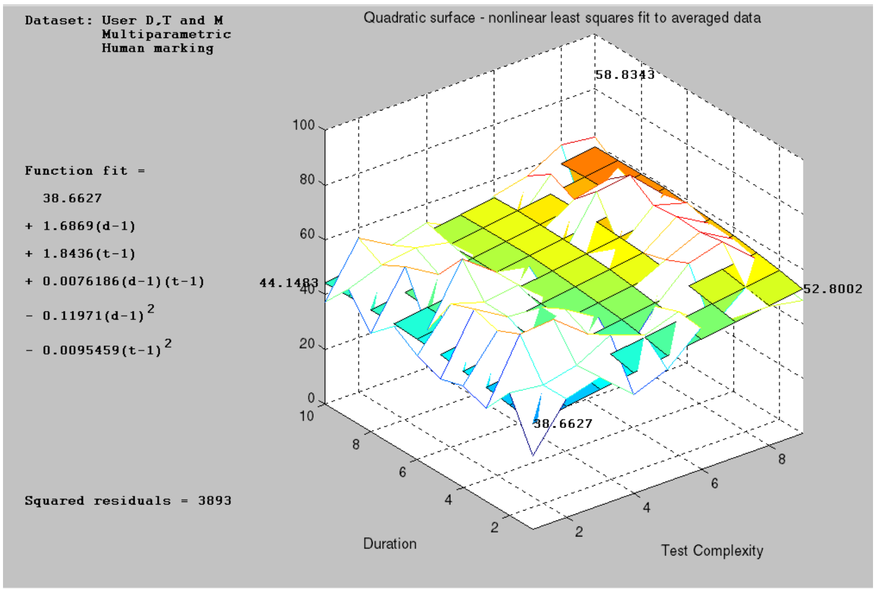

Fig. 27. 'Multiparametric' interface data from longitudinal tests.

This shows a dramatically different picture. The angle of the surface shows clearly that something very different occurred with the multiparametric interface. The following points of comparison with the previous two graphs are noted:

- For the simplest test the scores are always lower than those for the mouse or sliders, but they improve over time.
- The scores **get better** for more complex tests and are much higher than the other two interfaces.
- There is a good improvement over time across **all** test complexities.

The upward tilt of the plane towards the far-right corner is the most notable feature of the graph. It demonstrates that on average, the multiparametric interface performs better on the complex tests and yet allows a general all-round improvement on tests of all complexities. Neither of the other interfaces had this characteristic. The limitation of this interface appears to be that the simplest tests are difficult to achieve and especially so on the first few sessions.

10. Major conclusions

The following five major conclusions have been drawn from the test results and the user comments:

1. Real-time control *can* be enhanced by the multiparametric interface
2. Mappings which are *not* one-to-one are more engaging for users
3. Complex tasks may need *complex* interfaces
4. The "mouse interface" is good for simple tests and for little practice time
5. Some people prefer to think in terms of separate parameters

The following five sections correspond to these conclusions. They expand each conclusion with reference to the test results and the user interview analysis.

10.1. Real-time control can be enhanced by the multiparametric interface

The following two results are perhaps the most significant of the entire experiment:

- For tests where more than one parameter changes simultaneously the multiparametric interface gives the best overall results.
- The multiparametric interface nearly always allows improvement over time (independent of test complexity).

Taken together they show that the multiparametric interface is the only interface of the three that consistently permits improvement by the user, and that for the harder tests it is the only one which allows good results. These findings are best expressed graphically and can be seen in Figures 25 to 27.

Clearly real-time control can be enhanced by the multiparametric interface.

10.2. Mappings which are not one-to-one are more engaging for users

The following three comments summarised from the users indicate that there was something about the multiparametric interface which allowed spatial thinking that was entertaining and engaging.

- The multiparametric interface allowed people to think gesturally, or to mentally rehearse sounds as shapes.
- The majority of users felt that the multiparametric interface had the most long-term potential. Several people commented that they would quite like to continue to use it outside the context of the tests!
- Several users reported that the multiparametric interface was *fun*.

In contrast the sliders interface often elicited the opposite response:

- The majority of people found the sliders interface confusing, frustrating or at odds with their way of thinking. This was often focused on the requirement to break down the sound into separate parameters.

Since both the sliders and multiparametric interfaces allowed the user to have continuous control over all four sound parameters, we can conclude that the above differences can be accounted for by the parameter *mapping*. In other words:

Non one-to-one mapping strategies can be more engaging to users than one-to-one mappings, leading to a relative improvement in performance over time.

10.3. Complex tasks may need complex interfaces

The following result is really rather bizarre:

- The performance of the multiparametric interface nearly always *increases* with test complexity.

It is almost counter-intuitive to conclude that results could get better for harder tasks, but that it clearly what is shown in figure 27. This feature is only demonstrated consistently for the multiparametric interface. Various comments from the users confirmed that this was indeed what was happening. It is worth being reminded that only one of the sound tests out of the nine (test no. 9) was actually created on the multiparametric interface. This shows that the interface allows people to cope with tests of arbitrary complexity, not just with gestures that work well with a mouse.

Perhaps we should ask what we mean by a "difficult" or "complex" test? Maybe we call them ‘hard’ because they **are** hard on traditional interfaces. However most users were able to use the multi-dimensional nature of the multiparametric interface to formulate spatial and motor strategies for navigating the complex data-space. They therefore found it easier to cope with the more highly dimensional sounds. In contrast the "simple" sounds, made up of a step change in one parameter, are very difficult to achieve on the multiparametric interface. This is because the network of interconnected mappings makes it difficult to isolate a single change in one parameter without affecting any of the others.

This is reminiscent of the work done by Jacob et al where the authors note that:

"Current input device taxonomies and other frameworks typically emphasise the mechanical structure of input devices. We suggest that selecting an appropriate input device for an interactive task requires looking beyond the physical structure of devices to the deeper perceptual structure of the task, the device, and the interrelationship between the perceptual structure of the task and the control properties of the device". [Jacob, 1994]

They describe simultaneous tasks as being *integral* (perceived as one by the user) or *separable* (perceived as independent). Their experiment shows that interfaces which couple together integral tasks perform better than those which keep them separate. This could perhaps provide a reason why the ‘complex’ tasks are carried out best on the multiparametric interface (which clearly couples several parameters together).

Our research hypothesis is that performance improves when the perceptual structure of the task matches the control structure of the device". [ibid.]

This is confirmed by the user comments that the multiparametric interface allowed them to think in shapes or gestures. In other words the pitch/timbre/volume space is perceived as being *integral* and thus needs an interface to match.

If a general conclusion can be drawn from this result it is this:

Interfaces should be designed to suit the user's perception of the data-set that is to be controlled.

10.4. The "mouse interface" is good for simple tests and for little practice time

The effectiveness of the mouse interface can be summarised by this result from the tests:

- For most people the mouse interface gives the best results for the simplest tests, and the multiparametric the worst.

and this statement from the user comments:

- The mouse is the easiest interface to use at first, but is clearly limited to controlling one parameter at a time.

These tests have yielded a series of very positive conclusions about the multiparametric interface. The drawback is that it is the worst interface for the simplest tests (at least initially). It is here that the mouse interface scores best. In other words:

If people only tried the simple tasks for a short time they would conclude that the mouse interface was clearly the best.

Many tasks are indeed simple and do not require any significant depth of user learning. The emphasis on easy-to-use systems is quite relevant for public systems such as cash-point and parking ticket machines. In these cases the user only has occasional and "shallow" contact. Interface styles such as menus act as a reminder to the user about what (limited) options are available. Where a more regular and more involved level of contact is required, such as with office machines, industrial control systems and musical processing devices, the commonly accepted interface styles may be lacking.

It should be noted that all the test subjects used a mouse for office-based tasks on a daily basis, so they were already practised at its physical manipulation, particularly for point-and-click tasks. It also needs to be stressed that there are a number of concessions made with the mouse interface used in these tests that give it an advantage over more traditional mouse interfaces. There are no menus. The on-screen sliders are set up into their starting positions automatically. Real-time operation is allowed. There is no off-line programming of numbers or lists. The sliders on the screen can be directly manipulated with the mouse. Even with all these advantages the multiparametric interface wins with increasing practice time and test complexity.

10.5. Some people prefer to think in terms of separate parameters

There seems to be a small group of people who prefer to work in *analytical* mode rather than the holistic, performance mode. One of the user comment summaries stated:

- A small proportion of people (about one quarter) favour the sliders interface.

Could it be that certain people are highly analytical and do better on an interface which reflects the breaking down of the task into individually manipulable components? Could the following argument be true?:

Most people who are employed to design user interfaces are highly analytical. They read the HCI literature which is highly analytical. They produce interfaces which suit highly analytical people. They represent only a small proportion of the population. Their interfaces are used by the population at large, the majority of which think in a very different way, and therefore find the interfaces difficult to use.

Of course this is highly speculative and it is not within the scope of this work to try to prove or disprove that argument, but it could possibly account for the analytical nature of common computer interfaces.

11. Further Applications

The above experiments show some very good results for the interface with complex mapping strategies. In the following sections we propose some areas of research that may benefit from the application of such techniques. In each case the we would expect a custom interface to be designed to take account of the particular control requirements for each parameter set.

11.1. Musical Synthesis and Editing

New instruments should have multiparametric and non one-to-one mapping control if they are to engage musicians to develop a high level of control intimacy.

We suggest that there are many electronic music functions, such as phase-vocoding, time-stretching, and granular synthesis (that are typically controlled by text files) and standard synthesis algorithms that could be manipulated in real-time using a multiparametric interface. Rovan et al explain it like this:

Additive synthesis, for instance, has the power to virtually synthesise any sound, but is limited by the difficulty encountered in simultaneously controlling hundreds of time-varying control parameters; it is not immediately obvious how the outputs of a gestural controller should be mapped to the frequencies, amplitudes and phases of sinusoidal partials. [Rovan, 1997]

11.2. Studio Mixing

A typical mixing desk consists of many sliders and dials, and is rather like the "sliders" interface on a grand scale. Whilst it would be folly to suggest that such a time-honoured interface would be replaced, it could perhaps be enhanced. The Mixing-Desk style, as it stands, is an excellent way of having individual control of many parameters and of distributing these parameter controls in space.

Imagine a new mode of operation where a multiparametric interface (perhaps using a pair of data-gloves as a control device) is used to control many of the mixing attributes simultaneously. It may be possible to have hand gestures which continuously control hundreds of mixing desk parameters in real-time. An example of this might be that the closing of one hand controls the High-EQ (low-pass filters) on sound channels 2-20, whilst the position of one hand in relation to the other determines the relative volume of all the strings and the brass. Thus whole sections of instruments can be mixed and moulded with hand gestures, independent of their layout on the main desk.

This way of working is much closer to the task of ‘conducting’, which is also a time-honoured way of controlling the balance between groups of instrumentalists.

11.3. Artistic packages

It is proposed that performance modes may be equally beneficial to visually-based artistic or multimedia packages. Wherever there is a complex parameter set that warrants exploration by the user, then a multiparametric interface may improve the performance. Too often the artistic process is sequentialised and broken up into individual commands *because of the structure of the interface*.

12. Summary

The experimental work into the comparison of three different interfaces (mouse, sliders and multiparametric) has revealed five major conclusions:

1. Real-time control *can* be enhanced by the multiparametric interface
2. Mappings that are not one-to-one are more engaging for users
3. Complex tasks may need *complex* interfaces
4. The "mouse interface" is good for simple tests and for little practice time
5. Some people prefer to think in terms of separate parameters

It is clear that time is needed to learn an interface, and that some interfaces may require a longer practice time in order to achieve good results. This is perhaps hardly surprising if we consider the amount of time needed to learn to drive a car or play a musical instrument. If it took several months or years to master a computer interface it would typically be rejected as non-viable. In many cases there is an assumption that a user interface should only take a few minutes to master. This would rule out every musical instrument and vehicle driving system that had ever been invented.

So therefore perhaps our preconceptions about computer interfaces are wrong. Possibly for some tasks we should *not* expect an ‘easy’ interface which takes minimal learning. For many real-time interactive interfaces we need control over many parameters in a continuous fashion, with a complex mapping of input controls to internal system parameters.

Maybe we need a substantial amount of time to learn a complex control interface in order to give us confident real-time control over complex systems.

Acknowledgement

Many thanks to John Szymanski (University of York, UK) for the MATLAB programming which rendered the 3D plots of the data generated in this experiment.

References

Cadoz, C., and M. Wanderley. 2000. "Gesture – music", in this volume.

Jacob, R., L. Sibert, D. McFarlane and M. Mullen. 1994. *"Integrality and Separability of Input Devices"*, *ACM Transactions on Human-Computer Interaction* 1(1): 3-26.

Hunt, A. 1999. "Radical User Interfaces for Real-time Musical Control", DPhil thesis, University of York, UK, October.

———. 1997. MIDAS Web site http://www-users.york.ac.uk/~adh2/midas/midas.html.

———, and R. Kirk. 1999. "Radical User interfaces for real-time control", Milan : Proc. EuroMicro, September.

Kirk, R. and A. Hunt. 1996. "MIDAS-MILAN : an Open Distributed Processing System for Audio Signal Processing", *Journal of the Audio Engineering Society*, 44 (3): 119-129.

———, and A. Hunt. 1999. *Digital Sound Processing for Music and Multimedia*. Oxford: Butterworth-Heinemann.

Rovan, J., M. Wanderley, S. Dubnov, S. and P. Depalle. 1997. "Instrumental Gestural Mapping Strategies as Expressivity Determinants in Computer Music Performance." *Kansei, The Technology of Emotion*. *Proceedings of the AIMI International Workshop*, A. Camurri, ed. Genoa: Associazione di Informatica Musicale Italiana, October 3-4, 1997, pp. 68-73. Published on the Internet at http://www.ircam.fr/equipes/analyse-synthese/wanderle/Gestes/Externe/Mapp/kansei_final.html.