Smartsound: a framework for multi-user sound interaction

John Eacott
Unviversity of Westminster
London
john@informal.org
www.informal.org

Abstract

While designing interactive sound experiences a need has emerged for a flexible user interraction and controlling system which can be used across a variety of applications. An example of this type of interraction is currently being developed by Ambigence for the Intelligent Street project in which users control and interract with a sound environment by sending simple, one word commands as text messages from their mobile phone. There may me many users sending commands within a short time frame and the system must be able to respond to each one in a way that allows the user to perceive the effect of their input. Although the idea of sending commands is simple, in order to interpret the command within a complex and fluid field of interractivity we need to develop a general method of mapping these simple commands within a rich parameter space. Smartsound has been conceived with this type of usage in mind.

Overview and concept

The wave of 'interactive sound' pieces and ideas over the last 10 years has served to fill the middle ground between poles of conventional musical experiences. At one pole, live performance offers unique, transient experiences of in which there are significant and more subtle differences from one performance to the next. These experiences are complex and highly non-linear, in the sense that every aspect of the experience can be influenced by every other variable factor. The interrelationship between performers is the most obvious example of feedback mechanisms at play, while the participation and interaction of the audience is just as important - enthusiastic applause or the shout of a heckler can profoundly affect the shape of the experience as it unfolds. The experience class of which live performance is an instance is volatile and unstable and our experiences of live events varies dramatically because of this instability. The quality of the experience varies greatly as a result of many factors whilst our choice of live experiences is rather more limited (particularly if you consider physical location - i.e. whether you live in a city or rural area) because of the availability of events. It is not possible to hear 'Busted' live every evening but our appreciation of them my be enhanced by this rarification!

At the other pole, the almost universal acceptance of linear, fixed, recorded media offers a very different experience. Users can realise pre-rendered experiences and repeat them at will. This experience class is highly consistent and its availabilty is much greater. There is little unpredictability or volatility.

The very general notion of interactive sound takes many forms, including - and certainly not limited to - CDRoms, soundtoys, web based experiences, computer games, virtual sound environments, sound installations and a huge range of new instruments and performance tools. The richness of this field is a product of the size and complexity of the forms and paradigms of these approaches.

Whilst diversity is the foundation of evolution, there is also value in some degree of uniformity. Through various projects I have been involved with I have begun to notice a need for a general framework for interacting with sound. On its most fundamental level this framework proposes something that bridges between the two poles of musical experience described above. As a sysem it must be able to deal with musical events which exist within the following parameter spaces:

- Unique experiences / availabilty
- Transience / consistency
- Non-linear feedback mechanisms / controlabilty
- Instability / stabilty
- A sense of 'event' / repeatability

The notion of 'Smartsound has evolved specifically from the need to create a multi-user control system for 'Intelligent Street'[1]. The framework also extends directly from the work of d'Inverno and Eacott to develop the iHiFi (intelligent Hi Fi)[2]

User interraction

User interraction is provided through simple command systems which allow for inputs by multiple users. The mobile phone based SMS systen used for Intelligent Street has the advantage of being available to many of our target users. It also implies an extension of the use of mobile phones from communication devices to remote controllers for a variety of potential applications. multi-user input. Smartsound is designed specifically for multi-user input. In the case of the SMS based imput used for Intelligent Street the maximum possible number of inputs is vast - something like 2,000,000 per second has been quoted from although real-life figures will be substantially lower, probably between 0 and 20 per minute.

Functionality

For its users, Smartsound should be easily understood by its users and behave in a simple, intuitive way. Our starting point in terms of user interraction are the controls on a hifi - volume and the abilty to change content (by inserting a CD or choosing tracks on a CD). These two controls allow us to shape our experience albeit through somewhat stark choices. The volume control offers a continuum while selecting a track (or a CD) is a dicrete choice. What if we like the soothing flow of a Bill Evans jazz standard but would prefer more contemporary sounds or musical styles? It would be desirable perhaps to be able to adjust the style of music, or its energy in the same way along a continuous axis like we adjust the volume?

In order to achieve this we need to build a range of parameters into a system and find general rules for how those parameters are interpreted, by a wide range of musical styles and forms. The music itself is rendered through real-time algorithmic / generative processes. Despite recent huge leaps in the possibilities of real-time sound generation there will inevitably be compromises here - it took hundreds of years for the design and performance technique of say a violin to evolve and it may be extremely hard at this stage to replicate the subtlety and expression (and behaviours) of a single violinst let alone a violin section. Other forms of music, particularly synthesized and beat based forms such as dance music, offer a more realistic goal at this time.

In the case of'Intelligent Street' which is a multi-location interactive sound installation in which users create and interract with their sound environment by sending SMS messages from their mobile phone, it is envisaged that users transmit simple terms like 'energise' or 'mellow' and possibly specific style commands such as 'techno' or 'funk'.

Programme design

A Smartsound implementation requires the definition of 2 types of data:

- User controls
- Processing parameters

User controls

Our choice of control parameters is governed by the primary requirement that they be ubiquitous, i.e. they should (aspire to) be independent of musical style or specific musical knowledge of the listener. A parameter such as 'Energy' for example is farily widely understood and would fulfil this criteria wheras music specific terms such as scale, key, time signature etc would not. These user controls would be defined specifically for each Smartsound application, i.e. a particular sound installation or product and the choice and naming of controls would be dependent on the type, purpose and intended users of the piece. For Intelligent Street for example we are considering terms such as:

Energise / mellow

Urbanism / non urban

Current / retro

move / listen

as well as specific style choices such as 'funk' 'techno' etc. or even named pieces such as 'kevin's tune' which are incorporated into the system.

It should be clearly noted that the choice of user controls suggested here are simply an example. These user controls can be changed and added to very easily. A control system can be established which has few or many user controls.

Each user control is linked with a translation vector which converts the user command into an appropriate and perceptible sound response. So that if you send the command 'mellow' the music may reduce in tempo, reduce the density of instrumentation and select softer instrument timbres.

Processing parameters

While User controls can be changed and varied easily, the processing controls which deal directly with the generation of sound are more 'hard wired' into the system and their selection and design must be considered more carefully.

Although there is no set limit to the number of processing parameters used, it is practical to use a faily limited number at this stage. Parameters may be thought of as axis lines which combine together to define a multi-dimensional parameter space. To provide an easier way of conceptualising the system I have chosen to pair related parameter axes together into ‘2 dimensional ‘parameter planes’

In this example I have used 3 parameter planes (of 2 dimensions each) making a total parameter space of 6 dimensions.

The 3 parameter planes used in this example are:

Tempo / feel

Style

Sound / timbre

and are illustrated below.

There are many other parameter planes that could be added, obvious examples could be: time signature / meter, density of instrumentation / dynamics etc.

‘The Tempo and feel’ plane can be visualised as in figure 1. A position to the lower left would indicate a slow swung feel and the upper right a fast ‘straight’ feel. Any position within a plane can be represented as a pair of co-ordinates.

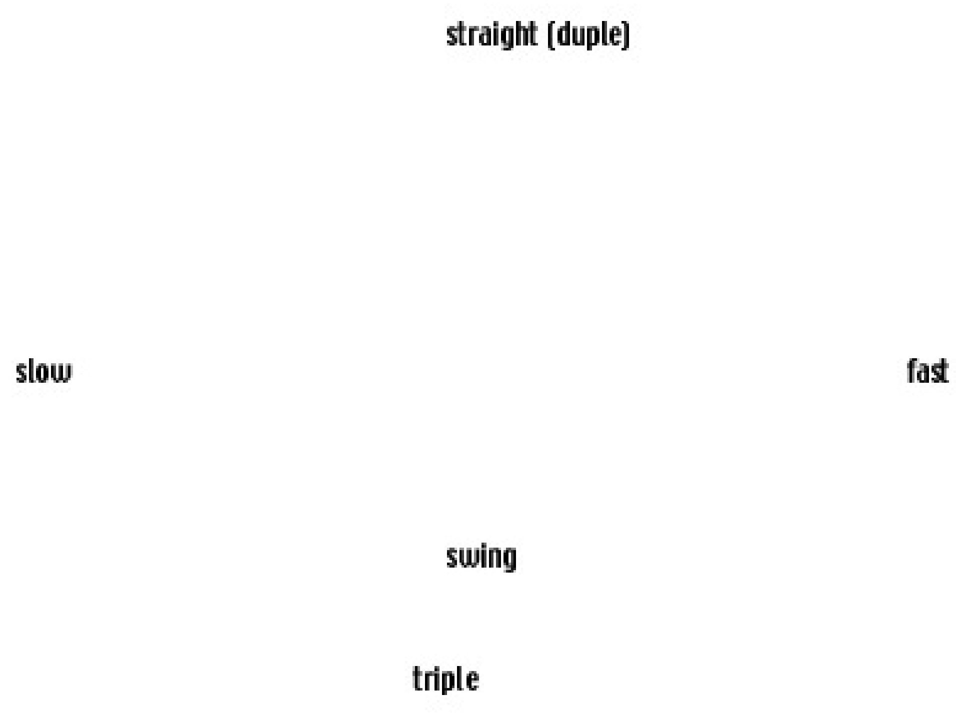

fig 1. Tempo / feel plane

While not generally regarded as a continuum it may be useful to regard ‘Style’ in that way in order to offer a flexible and controllable parameter. As with Tempo / feel, Style can be mapped on a 2 dimensional plane with a number of identified styles positioned on it.

The choice and construction of such a mapping is hugely subjective and to achieve it with any degree of thoroughness would require much more space than we have here. Any choices of basic axes are bound to be contentious but for illustrative purposes lets use the following using axes to represent ‘classical - popular’ and ‘melodic - rhythmic’.

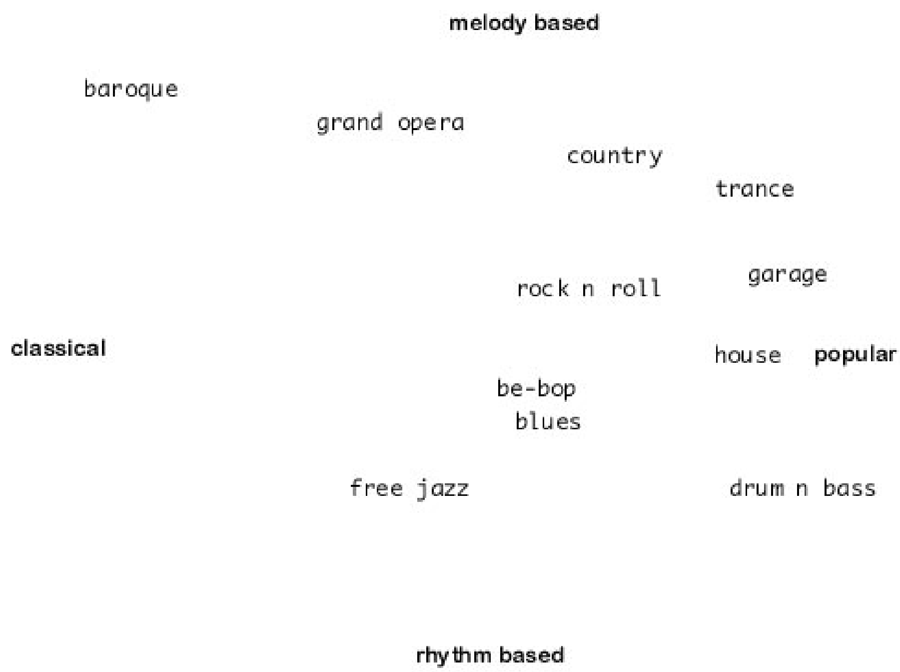

fig 2. Style mapping

Styles may be positioned on the style mapping plane according to their relationship with the axis parameters. Once positioned their location provides a reference coordinate.

It is concievable that further style dimensions (i.e. more than 2) may be useful although I shall not examine the implications of that here.

Our last chosen parameter ‘sounds / timbre’ could be represented as in figure 3.:

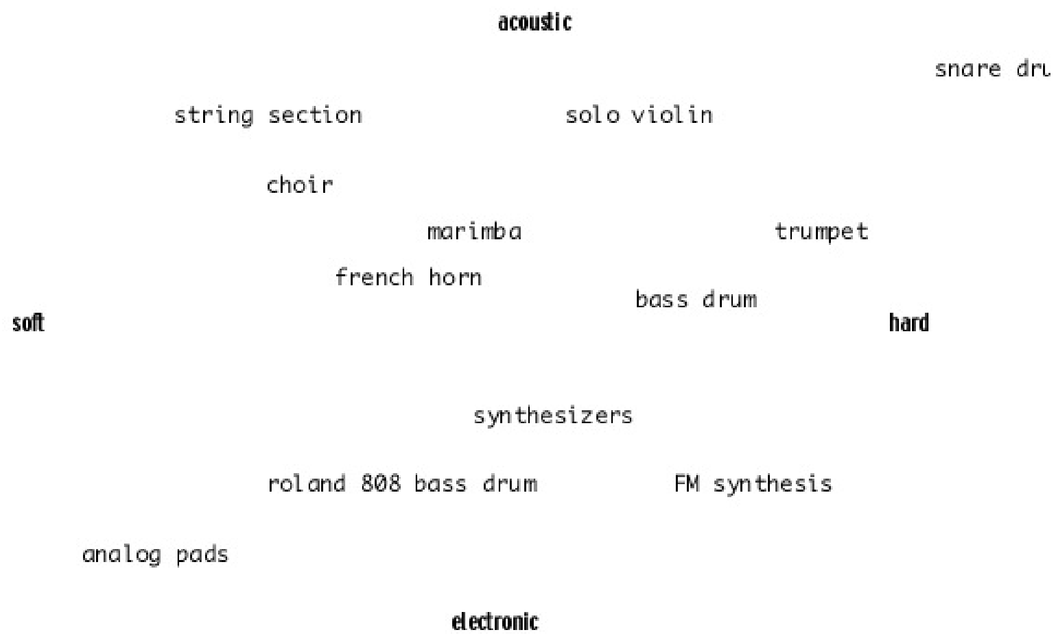

fig 3. Sound mapping

Multi-dimensional model

Through the combination of, in this case 3 planes we have a 6 dimensional model of the sound parameter space. Our users are able to define and move to any point in that space through their use of control commands.

Interpretation of parameters

The choice and design of algorithms is the critical issue at the heart of an interactive / generative sound process. In 'Smartsound' we have a limited range of parameters of which some - tempo and feel are simple arithmetic values whereas 'style' and'sound' are design critical.

Style in itself is immensely complex and cannot be bounded by strict rules about either its creation or interpretation. Any system dealing with style will contain approximations, assumptions and simplifications. My intention here is to illustrate how, for a given set of assumptions, it is possible to make a continuum that bridges between styles. The authenticity of the styles themselves is not to be considered here. It will be further shown how changes of style will normally have implications for other parameter planes, such as tempo / feel and sound. I should note here the work that David Cope has undertaken in this field[3], and his notion of style 'signatures' underpins the ideas developed here.

To show the type of design considerations necessary to implement choices in this parameter plane I shall demonstrate with a set of algorithms dealing with rhythmic aspects of style, specifically drum beats.

Firstly I identify motific elements - examples used here are only one bar - better examples would use longer phrases of say 4 or 8 bars.

Funk, (Bill Withers; Who Is He and What Is He To You[4])

tempo = 90

swing = 0.5

bd = [1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0]

sd = [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0]

ch = [1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0]

Techno (Leftism: Song of Life)[5]

bd = [1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0]

sd = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

ch = [0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0]

tempo = 112

swing = 0.5

R n B (Craig David: 7 Days)[6]

bd = [1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1]

sd = [0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0]

ch = [0, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 0, 1, 0, 1, 0]

oh = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

tempo = 96

swing = 0.56

Garage (Fill Me In)[7]

bd = [1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0 , 0 , 0, 0, 0]

sd = [0, 0, 0, 0, 1, 0, 0 , 0, 0, 0, 0, 0, 1, 0, 0, 0]

ch = [0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 1, 0]

tempo = 130

swing = 0.6

Continuous moving between discrete parameters.

As the current position moves in the gaps between specified locations of stles for example we have to find a way of 'tweening' or creating hybrid styles which fuse elements of nearby styles depending on their proximity.

There are several ways this could be accomplished. The solution I present here is possibly the simplest.

We take the distance (within the style parameter plane) from the current position to the nearest style positions. The algorithm could specify whether all styles or just say, the nearest 2 or 3 are considered. For this example let us take the nearest 3 styles. The data array of each style is read and divided by its distance so that the nearer you are to a style the greater its values are magnified. Then all of the values of each style array are added together and the result is normalised i.e factored up or down to lie within the range of a single array . Where decimals or fractions exist and cannot be interpreted we must derive an appropriate way or rounding up or down. The results of such a ‘style tweening’ are shown in appendix 1.

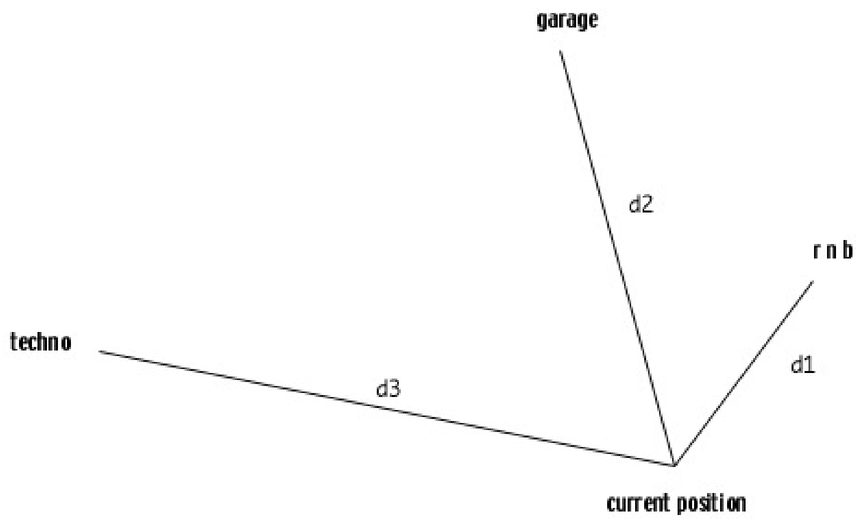

fig. 4 style ‘tweening’

For illustrative purposes lets consider the tempo and bassdrum (bd) values and use the following values for the distances of each style from our current position:

d1 = 2

d2 = 4

d3 = 5

d4 = 1.75

Tempo = 96 / 2 (rnb tempo ÷ d1)

+ 130 / 4 (garage tempo ÷ d2)

+ 112 / 5 (techno tempo ÷ d3)

= 102.9

rounded up becomes 103bpm.

Bassdrum patterns:

[1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1] / 2

+ [1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0 , 0 , 0, 0, 0] / 4

+ [1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0] / 5

= [0.95, 0, 0, 0, 0.2, 0, 0.25, 0.75, 0.7, 0, 0, 0, 0.2, 0, 0.2, 0.5]

rounded (with a cut off of 0.6) and normalised becomes:

= [1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0]

We arrive at a new tweened bassdrum pattern (while stongly retaining characteristics of the garage and rnb patterns) at a tweened tempo of 103.

Clearly the tweened styles may or may not be musically interesting styles of their own. This will depend substantially on the specific choices of available styles, their assumed locations in the parameter space and their relationship with other paramers such as sound and energy etc. and - of course - the taste(s) of the users! I imagine that the results will be varied, a multidimensional 'curates egg' in which there may be pockets of interest and other more barren regions!

Current position

The musical output of the system is determined by its 'current status' which is actually a reference position on each of the parameter planes. In our example using 3 planes, each requiring 2 values, a current status could be represented as an 6 value array thus:

[10, 99, 02, 08, 94, 82]

User control interpretation

Because it is envisaged that user controls can be modified and changed often within the lifetime of a peice wheras the processing control parameters are 'hard wired' into the system, it is neccessary to have an interpreter process, to convert user input to processing control. Although the design of the interpretation may take considerable thought, its implementation is extremely simple and takes the form of a vector, with a value representing a shift within each dimension or processing control. So an interpretation of the command ' energise' may take the form of the vector [10, 0, 10, -10, 10, -10] (i.e increase the tempo, move towards 'popular' and 'rhythm-based' styles and harden and make more 'electronic' the sounds)

Relative and absolute controls

Wheras normal control vectors are a relative position commands, i.e. the position moves in relation to the existing current position, it is envisaged that there may be the need to offer 'absolute commands' which take the current position to an exact, pre defined location within the parameter space. This could be useful if you wish users to be able to access a previously tested state - such as a specific style. In that case rather than a vector the command takes the form of a 'goto' and provides an abolute position within the parameter space.

Conclusion

The framework presented here is intended to be a simple and robust system for handling multi-user input within a wide range of sound installations and interactive sound experiences. We plan to use this system in the implementation of 'Intelligent Street' and possibly other pieces and projects. Through the expected lifespan of those pieces we hope to have a good opportunity for its testing and evaluation.

---

[1] Ambigence.com "Intelligent Street" Ambigence 2003. www.ambigence.com

[2] M. D'Inverno and J. Eacott; "On Embedded Intelligen Ambient Music" in J. Eacott (Ed) Cybersonica Proceedings 2002

[3] D. Cope; "Virtual Music" MIT Press, Cambridge, Mass. 2001

[4] B. Withers and S. McKenney "Who is He and What is He to You" Polygram Records, New York 1973

[5] Leftism; "Leftism - Song of Life" Sony Musici 1995

[6] Craig David; "7 Days" Wildstar Records 2000

[7] Craid David; "Fill Me In"Wildstar Records 2000