Analyzing Multi-touch Data for Expressive Musical Performance

Patrick McGlynn

Sound and Digital Music Technology Group,
National University of Ireland, Maynooth,
Co. Kildare,
Ireland

patrick.j.mcglynn@nuim.ie

Abstract. *Performance data from multi-touch devices can be interpreted in great detail and used to describe the speed, shape, lifespan, size and position of multiple gestures over time. However, most applications employ a GUI-based interface which, by its nature, only affords significance to the coordinates of a user's fingers.*

This paper introduces the SurfacePlayer project – which aims to provide a comprehensive library of high-level functions for working with multi-touch surfaces in the Processing development environment. By providing convenient access to information which was previously time-consuming to obtain, it is hoped that this project will enable the rapid prototyping and creation of elegant, expressive and intuitive digital musical instruments.

Analisando Dados de Multitouch para Performance Musical Expressiva

Resumo. *Dados de interfaces multitouch podem ser interpretados em grande detalhe e usados para descrever a velocidade, forma, tempo de vida, tamanho e posição de múltiplos gestos durante o tempo. No entanto, a maioria dos aplicativos emprega interfaces gráficas que, por sua natureza, apenas dão significado às coordenados dos dedos do usuário.*

Este artigo apresenta o projeto SurfacePlayer - que irá fornece uma biblioteca completa de funções de alto nível para trabalhar com interfaces multitouch em Processing. espera-se que este projeto permite acesso para informação demorado caso contrário, com o objetivo último facilitar a criação de instrumentos musicais digitais expressivos e elegantes

Introduction

This paper contends that the design of innovative multi-touch interfaces for the expressive control of music has been hindered by a tendency to rely upon traditional GUI-based environments. As a result, these systems seldom make intelligent use of the subtle and complex data that can be derived from multi-touch devices. In order to address this problem, this paper introduces the SurfacePlayer project – dedicated towards developing a Processing library which allows users to manipulate multi-touch data using a selection of high-level functions. It is hoped that this library will encourage a more experimental approach towards multi-touch gestures and facilitate the creation of radically innovative and expressive performance interfaces.

1. Context

This section describes recent trends in the use of multi-touch technology for musical performance and highlights some areas where improvement may be made.

1.1 Multi-touch technology

Much like the widespread adoption of Graphical User Interfaces in the 1980's, the recent ubiquity of the multi-touch (MT) interface has prompted significant changes in the field of Human Computer Interaction (HCI). There has been a massive upsurge of commercial devices which employ MT since Jeff Han demonstrated his work in 2006 [1][2]. The new means of interaction afforded by this technology has been of particular interest to the computer music community – a well-designed MT interface has the potential to combine the tactile elegance of an acoustic instrument with the rich data output typically required for the expressive performance of electronic music.

The discontinued *Jazzmutant Lemur* [13] set a high standard for MT musical control by combining a high-resolution touchscreen with a flexible and powerful interface editor. The Lemur demonstrated the usefulness of many features which are fundamental to similar interfaces today – e.g. multiple touch points, Open Sound Control [14] compatibility, network-based communication and customizable surfaces. Applications such as *TouchOSC* [22] provide similar functionality, albeit minus the advanced scripting and physics capabilities of the *Lemur*, on Apple iOS devices [10] and thus represent a more cost-effective implementation of the 'customizable control surface' idea. A more experimental approach is demonstrated by applications such as *Konkreet Performer* [8] which allow the user to send MIDI or OSC data by manipulating abstract shapes displayed upon the screen.

Alongside these commercial ventures, online communities such as the Natural User Interface Group [9] are dedicated towards maintaining "a collaborative environment for developers that are interested in learning and sharing new HCI methods and concepts" [23]. Open-source frameworks such as the NUI Group's own Community Core Vision [11] and the TUIO protocol [12] allow for the rapid prototyping and development of experimental MT interfaces.

However, despite the fact that “gestural interfaces have a much wider range of actions with which to manipulate a system” [5], the vast majority of MT applications are built around a familiar GUI-style layout. While this traditional format undoubtedly benefits from MT functionality (through the use of gesture-based shortcuts and the ability to simultaneously manipulate several objects at once) over-reliance on interaction paradigms inherited from WIMP (Windows, Icons, Menus, Pointers) interfaces has made it difficult for developers to explore the full potential of MT interaction.

1.2 The multi-touch surface as a music controller

Previous research by the author has highlighted the importance of “adopting a methodical approach towards identifying and classifying the types of data generated by a particular device” [4]. A cursory glance at the capabilities of any MT device which uses the TUIO protocol allows us to infer the information shown in Table 1.

- The location of individual fingers at any given point in time
- Whether or not the surface is being touched
- The number of fingers in contact with the surface
- The distance and angle between any of these points
- The location, area, perimeter and shape of a space defined by these points
- Whether or not a point is static or moving
- The speed at which a point is moving
- The direction in which a point is moving
- The length of time a point has been present on the surface
- The previous movements and average position of a given point…

Table 1. Example performance data available from TUIO devices

This table serves to illustrate the problem with widget-based music software on a MT platform. Such environments solely employ the first point, multiple finger positions, to interact with various onscreen widgets such as buttons, faders, etc. The other types of data outlined above, while they might appear abstract or trivial, can in fact be combined in a wide variety of ways to create rich metaphors and gestural cues. It is plain to see how, in terms of designing software for a role as potentially nuanced as musical performance, the dominant GUI-based approach fails to utilize the available data in an intelligent manner.

Exceptions do exist, however, which fittingly treat the MT surface as a complex and sensitive tool rather than just a novelty controller. Kevin Schlei’s *MDrumSynth* and *MStretchSynth* both rely heavily upon relationship-based analysis for multiple parameter control [6], Balz Rittmeyer’s *Akustisch* recognizes and responds to a selection of expressive gestures using an elegant interpreter [15], and Christian Bannister’s *Subcycle Labs* cleverly analyses the number of touches present on the surface to toggle various DSP effects [16].

These examples, while markedly disparate in functionality and application, share one important characteristic – the use of high-level metaphors derived from basic MT data. The remainder of this paper describes a project which has been undertaken in order to assist musicians and composers in the development of such metaphors.

2. Development

This section outlines the concepts and goals behind SurfacePlayer – a project which aims to produce an open-source library for the design of innovative multi-touch interfaces for musical performance.

2.1 Aims & objectives

One of the main reasons for the relative scarcity of experimental interfaces, such as those mentioned above, is the amount of work required to analyze the data generated by the MT surface. The requisite knowledge of basic networking, control flow, geometry and HCI serves to form a significant barrier for even the most experienced users. While there are plenty of libraries and applications available to obtain raw touch data, there is a lack of support for high-level data which may prove to be more perceptually-relevant in a live performance context.

The objective of the SurfacePlayer project is to provide musicians and composers with a modular set of tools to facilitate the construction of expressive touch-based performance interfaces. It is hoped that this new set of interpretive tools will allow designers to concentrate their attention on more musically-critical aspects of the interface, such as mapping, and encourage more experimentation with MT music performance.

2.2 Implementation

SurfacePlayer is currently being developed as a library for Processing [17] – a development environment which uses easily-readable syntax and tends to be popular among artists working with sound and visuals. The project comprises a selection of algorithms which generate high-level information in response to MT data. This information can be quickly accessed via concise function calls, thus allowing the user to circumvent a considerable amount of programming.

Prior to the work described in this paper, designers were restricted to the use of raw data which describes the coordinates, speed and path history of a point, for example. Hard-coding even simple gestures using this raw data can be a time-consuming and tedious process. The SurfacePlayer library will support a wide variety of commonly-used functions which receive TUIO data and check for certain conditions. When these conditions are met, a gesture is recognized and relevant data related to that gesture can be used within the performance patch.

For example, in order to infer the direction of movement for a given touch, it has previously been necessary to undertake a cumbersome analysis of the path history and the average angle between points (or, alternatively, devise an algorithm which infers the direction based upon the relative speeds of X and Y-axis movement). Similarly, an action as ubiquitous as a ‘multiple-tap’ (where taps made using more than one finger are differentiated) requires an analysis of touch coordinates, birth/death time and the use of multithreading in order to be of any practical use. The complexity of these processes is likely to discourage the widespread use of the often useful information which they can generate.

In response to this issue, the functions being developed for the SurfacePlayer library allow users to access this kind of information using succinct and easily-readable commands such as 'movementDirection()' and 'multiTap()'. Users can thus experiment with different combinations and sequences of cues which were previously difficult to implement.

The functions are all being added to the library separately, allowing for the possibility of user-defined/requested algorithms, and are compatible with existing TUIO implementations for Processing – the user simply needs to import SurfacePlayer and call upon the functions within their own code.

2.3 Example of use

This section describes how SurfacePlayer might be integrated into the architecture of a typical MT performance system. Please note that, at the time of writing, it would be impractical to provide examples which feature code as the syntax is still being finalized. However the library will be made available, along with detailed instructions for use, on the project website [18].

A typical use of SurfacePlayer may be broken into three distinct components – the input layer, interpretation layer, and output layer. These layers are illustrated in Figure 1 and described in the following sections.

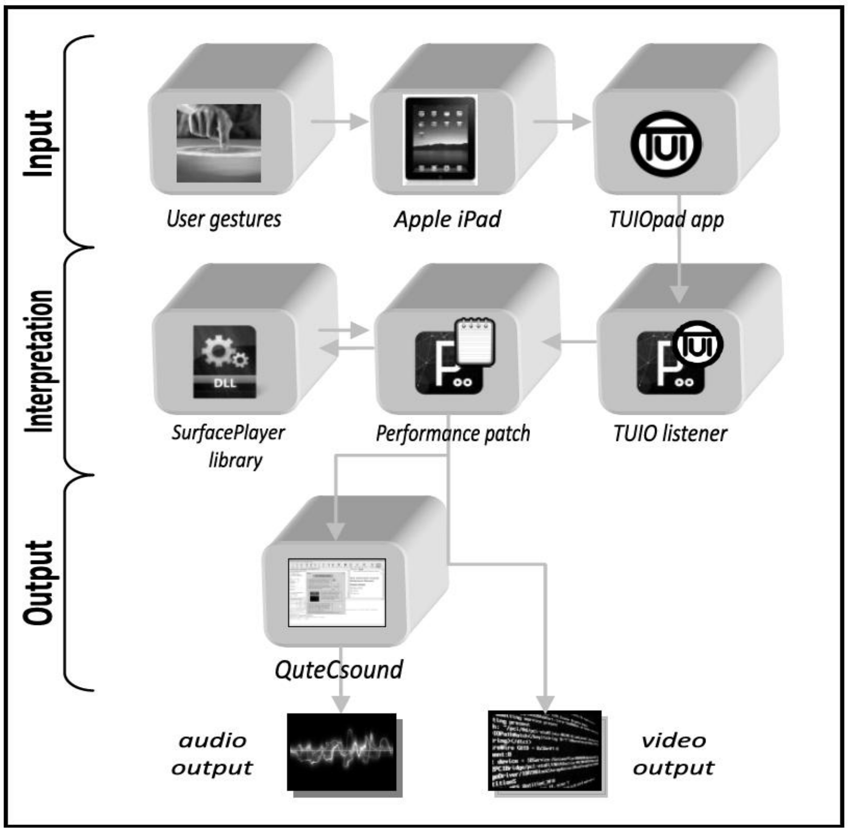

Figure 1. SurfacePlayer architecture

2.3.1 Input layer

This layer consists of any device, or number of devices, capable of generating TUIO data in response to user gestures. In the example above, an iPad running the open-source application TuioPad [19] sends MT data to a computer via a wireless network.

The TUIO protocol was chosen due to its flexibility and active user community. It also renders the system non-hardware-dependant – allowing the algorithms implemented within SurfacePlayer to be used with any device capable of outputting TUIO-formatted cursor data.

2.3.2 Interpretation layer

The composite elements of this layer are implemented within the Processing development environment. The Processing TUIO Client API [20] listens for incoming TUIO events and generates data related to touch positions, such as time tags and coordinate paths. This data is subsequently interpreted by the functions provided by the SurfacePlayer library – which itself is referenced from within the user-created performance patch.

As outlined in section 2.2, the information generated by the library can easily be interpreted in a variety of ways depending on the specific needs of the user.

2.3.3 Output layer

According to the needs of the user, the gestures described by SurfacePlayer's functions can be used to send OSC or MIDI data to other applications. The examples provided are all implemented in the open-source sound synthesis environment QuteCsound [21]. A number of templates will also be developed which can be used to generate simple visual feedback in Processing itself. Projected or displayed on a convenient screen during performance, this feedback can eliminate the need for a performer to look down at the surface itself constantly while playing.

3. Future work

The current version of the SurfacePlayer library provides easy access to some of the most commonly-used MT cues – such as tap and double-tap recognition, multiple-taps supporting up to ten fingers, and directional swipes of varying speeds. It can also be used to determine the surface area, diameter, centroid and perimeter of shapes formed by surface touches. The accompanying examples illustrate how these cues can be combined in complementary ways to create novel and expressive interfaces for the control of music.

In addition to supporting this kind of fundamental component, the project aims to provide functions capable of tracking and interpreting more complex gestures. It is intended that these will eventually allow the user to design and implement their own discreet gestures in a manner similar to the work of Wobbrock, Wilson and Li [7]. The library will continue to be freely-available online and a forum will allow users to request new functions, share code, and show examples of the library in use.

Ultimately, the tools developed for the SurfacePlayer project will be used to illustrate and evaluate new approaches to the mapping problem which are currently being investigated by the author [4]. It has been suggested that, with regard to electronic music performance, complex interaction schemes have the potential to provide a more intuitive and rewarding user experience than simple systems [3]. The SurfacePlayer library will hopefully prove to be a valuable resource for anyone trying to elucidate these issues through experimentation with touch-based interfaces.

Conclusion

This paper has drawn attention to a number of shortcomings with regard to the use of multi-touch technology in musical performance. In particular, the tendency of current applications to rely extensively upon coordinate data has resulted in the exclusion of other potentially-useful data. In response to this problem, the SurfacePlayer library is being developed in order to allow users to engage with this neglected information quickly and easily. It is hoped that this library will encourage a more experimental approach towards multi-touch gestures and facilitate the creation of radically innovative and expressive performance interfaces.

Acknowledgements

The author would like to express his sincere gratitude towards Dr. Victor Lazzarini and Dr. Gordon Delap for their endless enthusiasm, support and inspiration throughout this research. This research project is funded by the John and Pat Hume Scholarship programme at NUI Maynooth, Ireland.

References

[1] Han, J. (2005) “Low-Cost Multi-Touch Sensing through Frustrated Total Internal Reflection”. In *Proceedings of the 18th Annual ACM Symposium on User Interface Software and Technology (UIST05),* Seattle, USA. p. 115-118.

[2] Han, J. (2006) “Multi-Touch Demonstration - TED 2006”. http://www.ted.com/talks/jeff_han_demos_his_breakthrough_touchscreen.html

[3] Hunt, A. (1999) *Radical User Interfaces for Real-time Musical Control.* PhD thesis, University of York.

[4] McGlynn, P. (2011) “Towards more effective mapping strategies for digital musical instruments”. In *Proceedings of the 9th Annual Linux Audio Conference (LAC2011)*, Maynooth, Ireland. p. 93-98

[5] Saffer, Dan. (2009) *Designing Gestural Interfaces: Touchscreens and Interactive Devices.* Sebastopol, CA: O’Reilly Media. p.6

[6] Schlei, K. (2010) “Relationship-Based Instrument Mapping of Multi-Point Data Streams Using a Trackpad Interface”. In *Proceedings of the 2010 Conference on New Interfaces for Musical Expression (NIME2010),* Sydney, Australia. p. 136-139.

[7] Wobbrock, J.O., Wilson, A.D. and Li, Y. (2007) “Gestures without libraries, toolkits or training: A $1 recognizer for user interface prototypes”. In *Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology (UIST07),* Newport, Rhode Island. p. 159-168.

[8] http://konkreetlabs.com/performer/overview/

[9] http://nuigroup.com/log/about/

[10] http://developer.apple.com/technologies/ios/

[11] http://ccv.nuigroup.com/

[12] http://www.tuio.org/

[13] Jazzmutant. Lemur. http://www.jazzmutant.com/lemur_overview.php

[14] http://opensoundcontrol.org/introduction-osc

[15] Rittmeyer, B. Akustisch. http://akustisch.digitalaspekte.ch/

[16] Bannister, C. Subcycle Labs. http://subcycle.org/

[17] Fry, B., and Reas, C. Processing. http://processing.org/

[18] McGlynn, P. SurfacePlayer. http://www.patrickmcg.com/surfaceplayer

[19] https://code.google.com/p/tuiopad/

[20] Kaltenbrunner, Martin. Processing TUIO. http://www.tuio.org/?processing

[21] http://qutecsound.sourceforge.net/

[22] http://hexler.net/software/touchosc

[23] http://nuigroup.com/log/nuigroup_book_1