Recontextualizing the Multi-touch Surface

Patrick McGlynn†, Victor Lazzarini†, Gordon Delap†, Xiaoyu Chen‡
† Sound & Digital Music Technology Group, {patrick.j.mcglynn, victor.lazzarini, gordon.delap}@nuim.ie
‡ Department of Computer Science, {xiao.y.chen}@nuim.ie
National University of Ireland, Maynooth, Ireland.

ABSTRACT
This paper contends that the development of expressive performance interfaces using multi-touch technology has been hindered by an over-reliance upon GUI paradigms. Despite offering rich and robust data output and multiple ways to interpret it, approaches towards using multi-touch technology in digital musical instrument design have been markedly conservative, showing a strong tendency towards modeling existing hardware. This not only negates many of the benefits of multi-touch technology but also creates specific difficulties in the context of live music performance. A case study of two other interface types that have seen considerable musical use – the XY pad and button grid – illustrates the manner in which the implicit characteristics of a device determine the conditions under which it will favorably perform. Accordingly, this paper proposes an alternative approach to multi-touch which emphasizes the implicit strengths of the technology and establishes a philosophy of design around them. Finally, we introduce two toolkits currently being used to assess the validity of this approach.

Keywords
Multi-touch, controllers, mapping, gesture, GUIs, physical interfaces, perceptual & cognitive issues

1. INTRODUCTION
This paper discusses design issues for digital musical instruments (DMIs) [14] which utilize multi-touch (MT) technology. The focus is firmly upon experimental and/or innovative instrument designs which engage with the users' sense of tacit knowledge [15] and facilitate spontaneity and improvisation.

There are three main sections – 'Surface-based Interfaces' describes in detail the data generated by two popular types of controller and how it influences their use in DMI design. The MT interface is then discussed in the same context and a summary of notable uses is provided. 'Designing Multi-touch Interfaces' describes the often-restrictive use of graphic user interfaces (GUIs) in MT systems and suggests an alternative approach with an emphasis on gestural, as opposed to visual, interaction. Finally, 'Research tools' provides a brief introduction to two projects which are being developed in order to explore the proposed design space.

A brief note on terminology – controllers are often referred to as *'n-dimensional'* to signify how many independent streams of data they produce. Within the context of this paper, this terminology is misleading – both due to the spatial connotations of concepts like 2D / 3D and the loosely-coupled mapping strategies previously proposed by the author [13]. A more precise system is required to describe the capabilities of a device. Therefore, the term *degrees of freedom* (DOF) will be used to describe how many data streams can be independently controlled and the term *resolution* will refer to the range of data or number of states available. For example, a switch is a low resolution (0-1) controller with one DOF, whereas a MIDI fader is a higher-resolution (0-127) controller with one DOF. Further detail on this terminology can be found in [10].

Issues related to mapping strategies, while certainly a goal of the work being described, are beyond the scope of this paper. They are referred-to solely in order to illustrate typical applications for the controllers being discussed and suggest ways in which their data may be used.

2. SURFACE-BASED INTERFACES
This section consists of a review of various surface-based interfaces when used as musical controllers. The surfaces in question are simple XY pads, button arrays (also known as 'grids') and multi-touch surfaces. The grouping of these devices under the heading 'surface-based interfaces' is not to suggest some kind of abstract category, but rather to emphasize their shared physical characteristics – all are basically flat sensor devices which respond to human finger-touches, albeit in different ways.

2.1 Historical roots
There is a rich history of analog synthesizers designed to respond to touch – the *Ondes Martenot* (1928), *Trautonium* (1929), *Fingerboard Theremin* and *Keyboard Theremin* (1932) and *Electro-Theremin* or *Tannerin* (1958) all used precise finger movements as their primary means of control and laid the foundation for more contemporary devices such as the MIDI ribbon controller [2]. Pen-based interfaces such as *UPIC* (conceived by Xenakis and implemented by Centre d'Etudes de Mathématique et Automatique Musicales (CEMAMu) in Paris) also inspired computer musicians to begin working with tablets. The "quantitative merits" of the tablet as a musical controller have been well-established, practically as well as theoretically, by research carried out at CNMAT [23, 24].

2.2 XY pads
The XY pad is a control surface which offers 2 DOF via its horizontal and vertical axes. Resolutions vary, but are typically high enough to accommodate continuous parameter control. The XY pad can be seen as combining the functionality of two faders into a single interface, as it offers simultaneous and independent control of two streams of data (although this comparison highlights some interesting differences, as discussed below).

The *Korg Kaoss Pad*[1] range brought mainstream attention to the use of XY pads for a variety of musical tasks with a

[1] http://korg.com/products.aspx?ct=4

NIME'12, May 21-23, 2012, University of Michigan, Ann Arbor.

selection of high-profile users such as Beardyman, Johnny Greenwood and Matt Bellamy. The *KP*[2] range use the surface to control various live signal processing patches while the spin-off *Kaossilator* series are designed for pattern recording/playback using a selection of onboard synthesis patches. The manual for the *Kaoscillator Pro*[3] gives a comprehensive list of the mapping schemes employed and is indicative of the typical function of these devices within a performance setup.

The continuous nature of the output means that this kind of device lends itself well to glissandi and sweeping effects. Typical mapping schemes establish a one-to-one connection between each axes and a pair of parameters – cutoff/resonance of a filter, for example, or pitch/ loudness of a synthesizer. Some interesting observations upon the combinations of parameters are discussed in [20]. It has been suggested that any two parameters mapped in this way (i.e. controlled by a single point of contact from the user) have a high degree of integration [6] and should ideally influence closely-related elements of the sound. Of the examples given above, the cutoff/resonance combination is preferable as it deals exclusively with the behavior of the filter and allows users to associate a particular space on the surface with a certain type of sound or effect. Pitch/loudness are not so closely-coupled, as they deal with perceptually-separate aspects of the sound, and it has been observed that users may find this kind of mapping less intuitive [20].

As mentioned above, it is worth noting a number of differences not made explicit in the 'pair of faders' analogy. While the potential for simultaneous and independent manipulation of a pair of data streams is theoretically identical in both cases, there are three major differences between an XY pad and two faders: (1) the capacity for 'teleportation' – it is possible for a user of an XY pad to break contact with the surface and jump to a higher/lower position, (2) an XY pad does not provide any feedback (unless it is combined with a visual display) – a fader provides both tactile and visual feedback indicative of its current state, and (3) an XY pad can be manipulated with a single fingertip, whereas certain manipulations with the faders are difficult without the use of multiple fingers or hands.

XY pads are typically allocated an ancillary role in a performance system – playing a similar role to pitch-bend/modulation wheels or controlling effects – while primary tasks such as note selection or event triggering are left to devices such as keyboards or samplers.

2.3 Grid-based interfaces

A style of interface that has seen comparatively more musical experimentation is the grid-based layout popularized by devices such as the *Tenori-On*[4], *Monome*[5] and *Novation Launchpad*[6]. While generally represented by an array of separate buttons, the device is essentially a discretized version of the XY pad – replacing a high-resolution 2DOF controller with an array of low-resolution (binary) 1DOF controllers. The grid-interface can therefore be described as an array of switches.

Given the relative lack of precision that this description seems to imply, one could be forgiven for assuming that the usage scenarios are comparatively less-musical and flexible compared to those of the continuous XY pad. However, the opposite is true – grid-based interfaces have been employed in a vast array of musical tasks including sample-triggering[7], multi-effects processing[8], FM synthesis[9], step sequencer-control[10], visualization[11] and animation[12].

There are a number of reasons why this is the case. Firstly, the physical nature of an array of buttons provides a kind of tactile feedback which an XY pad cannot replicate. The importance of a tactile relationship between performer and instrument is well-acknowledged [17]. With an array of buttons, it is possible to discern the location of your fingers without relying upon visual feedback or actually triggering a reaction from the device. Secondly, many button-array controllers (including those listed above) light up individual buttons in order to indicate their individual status or to form a collective abstract shape. This capacity for unambiguous, immediate visual feedback is significant, as it allows the user to maintain a relationship with any number of abstract variables or multiple layers of functionality once the corresponding symbolism has been established and committed to memory. Accordingly, this added channel of communication with the user encourages more complex multimodal systems. Finally, it should be mentioned that the visual appeal of the lights themselves can be a motivation for employing these devices in a live context, even as works of art in themselves[13].

Together these factors give an impression of the increased potential of the button array as part of a robust live performance system. What appear to be trivial additions (buttons and lights) are actually partly-responsible for the variety of creative DMI designs that employ button arrays.

2.4 Multi-touch surfaces

This section discusses approaches to musical performance using multi-touch surfaces within three categories – covering hardware, academic and mobile application development, respectively.

Commercial hardware for MT music performance began with the *JazzMutant Lemur*[14] - a high-resolution touchscreen with a flexible and powerful interface editor. The Lemur arguably set the standard for MT music control – the direct influence of its approach, from the futuristic visual style to its use of Open Sound Control (OSC)[15], can be seen across a broad range of projects today.

While the Lemur was a generalized controller, recent trends in MT music interfaces tend to be designed with more specific tasks in mind such as mixing (*Line 6 Stagescape*[16], *KS-1974*[17], *Mackie DL1608*[18]), synthesizer performance (*Haaken Continuum*[19], *Soundplane*[20] and *Misa Kitara Era*[21]) and portable composition (*KDJ-One*[22]).

---

[2] http://www.korg.com/uploads/Support/KP3_OM_EFG1_633659261667720000.pdf
[3] http://www.korg.com/uploads/Support/KAOSSILATOR_PRO_OM_EFG1_634067737513300000.pdf
[4] http://www.global.yamaha.com/tenori-on/
[5] http://monome.org/
[6] http://www.ableton.com/launchpad
[7] http://youtu.be/CYv5jqHMe5c
[8] http://youtu.be/umsO-KLjRX8
[9] http://docs.monome.org/doku.php?id=app:straw
[10] http://stretta.blogspot.com/2011/05/plane-m-vi-cv.html
[11] http://vimeo.com/29517018
[12] http://vimeo.com/30976072
[13] http://vimeo.com/1338613
[14] http://www.jazzmutant.com/lemur_overview.php
[15] http://opensoundcontrol.org/
[16] http://line6.com/stagescape-m20d/
[17] http://www.smithsonmartin.com/kontrol-surface-ks-1974/
[18] http://www.mackie.com/products/dl1608/
[19] http://www.hakenaudio.com/Continuum/
[20] http://madronalabs.com/hardware
[21] http://www.misadigital.com/
[22] http://www.kdj-one.com/

Academic research into multi-touch music performance is widespread and diverse. As such, a comprehensive account is beyond the scope of this paper but some notable examples are referenced in [12]. Projects such as the *Reactable*[23], *Linnstrument*[24] and David Wessel's *SLABS*[25] also provide interesting and progressive examples of contemporary work. One particularly useful online presence is maintained by the Natural User Interface Group – both their forum[26] and free book *Multi-Touch Technologies* are invaluable sources of up-to-date information and advice [21].

Mobile applications are understandably a popular way to package and distribute MT music software. There is a vast selection of musical 'toys' on the iOS App Store which demonstrate an extremely-limited range of possibilities and are accordingly of little interest to musicians. There have been a number of attempts at 'serious' instruments – most of which are designed to resemble an existing piece of hardware (*Yamaha TNR-i*[27], *Korg iElectribe*[28]), though exceptions do exist (*TC11*[29], *Mugician*[30]).

Some of the more flexible musical tools available on mobile devices are dedicated 'controller' applications. These perform tasks only at the input stage of the DMI architecture and produce no sound. Instead, the users' interactions with onscreen widgets prompt the device to send data wirelessly to a computer via protocols such as MIDI, OSC and TUIO [7]. The host computer can then use this data to control synthesis or signal processing.

While a number of applications are specifically designed to complement existing hardware or software (*DL1608 Master Fader*[18], *V-Control Pro*[31], *Omni TR*[32]) the majority of controller applications allow the user to customize the layout of the screen in some respect – for example, to accommodate alternative keyboard layouts (*Musix* [11], *ExpressionPad*[33]). Most applications consist of a widget-based GUI - in this case the screen forms a canvas which can be populated by a selection of pre-designed faders, buttons, dials and touchpads (*Control*[34], *mrmr*[35], *TouchOSC*[36], *Lemur*[37]). This approach to musical performance using MT technology is by far the most popular due to its relative ease-of-use and familiar metaphors.

3. DESIGNING MULTI-TOUCH INTERFACES

3.1 Rethinking the GUI

As outlined above, the most popular way to design multi-touch user-interfaces is via a toolkit of widgets that provide typical GUI-like elements such as windows and menus. For musical interfaces, these toolkits usually contain a selection of hardware-inspired widgets such as faders, dials, drum pads, etc. While MT interfaces often resemble typical GUIs, there are vastly different design issues that need to be considered. These issues are well-established and have been under investigation for many years (see [18] for a comprehensive introduction and the work of Bill Buxton[38] for more-detailed analysis). So why do we persist in our use of a design strategy that is not ideally-suited to the device itself?

The explanation can be illustrated with a comparison to music controllers in general. A well-established criticism of MIDI interfaces has been their over-reliance upon the piano-keyboard metaphor, which by its nature cannot accommodate many of the features unique to synthetic sound (freedom from discreet pitch-structures, continuous control over timbre, etc.). There are many practical reasons, however, why the keyboard interface dominates – the most prevalent being that they allow pianists to utilize their existing skills to control new hardware/software and thus represent less of a financial risk to the manufacturers.

For the same reason, it makes perfect sense for designers of new DMIs to adhere to familiar GUI/WIMP (Windows, Icons, Menus, Pointers) paradigms. These design clichés allow us to exploit several decades-worth of embedded cultural and technological knowledge in our interfaces and there are abundant resources which enable us to do so. However, in much the same way as the piano keyboard was not designed to accommodate continuous pitch changes or gradual manipulation of timbre, the GUI was not designed with MT input or live music performance in mind.

The GUI paradigm has been optimized for use with a keyboard and mouse combination – it is therefore misguided to adopt this style of interaction on MT surfaces without any modification [24]. There are arguably some benefits to using a MT GUI in performance – the inability of a mouse to manipulate more than one onscreen object simultaneously is a limitation that the MT surface does indeed surmount. However, there is a vast array of negative repercussions – for example, over-reliance upon visual feedback, tendency for users' hands to obscure the screen (and hence, the only source of feedback) and the rigorous precision demanded by most MT GUIs make them a less-than-ideal solution for live musical performance.

Widget-based GUIs by their very nature encourage one-to-one mapping and tight-coupling at the procedural stage of DMI design – both restrictive approaches that lead to systems bound by 'the instrumental paradigm' [5, 9]. This kind of design approach imposes a cognitive load on the user which can impair their level of engagement with the performance, especially when other musicians are involved. It has been acknowledged that the emergence of social affordances during music-making can be seriously compromised by tightly-coupled DMIs [8].

This is not to suggest that robust and innovative GUI-based DMIs cannot be designed for MT surfaces. Rather it is being proposed that we should investigate, with equal vigor, the possibility of creating new interaction paradigms that best exploit the unique properties of the MT surface as a performance interface.

3.2 Beyond the GUI

As illustrated in previous publications, interactions with MT surfaces generate extremely rich data [12]. There are many resources which can help DMI designers to access this data – Reactivision[39], TUIO, CCV[40] and the NUI Group all provide a

[23] http://www.reactable.com/
[24] http://www.rogerlinndesign.com/preview-linnstrument.html
[25] http://cnmat.berkeley.edu/user/david_wessel/blog
[26] http://nuigroup.com/forums/
[27] http://uk.yamaha.com/products/musical-instruments/entertainment/tenori-on/tnr-i/
[28] http://www.korg.com/ielectribe
[29] http://www.bitshapesoftware.com/instruments/tc-11/
[30] http://rrr00bb.blogspot.com/2010/08/mugician-heiroglyphics.html
[31] http://www.neyrinck.com/en/products/v-control-pro
[32] http://www.spectrasonics.net/products/omni_tr.php
[33] http://expressionpad.com/
[34] http://charlie-roberts.com/Control/
[35] http://mrmr.noisepages.com/
[36] http://hexler.net/software/touchosc
[37] http://liine.net/en/products/lemur/
[38] http://www.billbuxton.com/papers.html#anchor1442822
[39] http://reactivision.sourceforge.net/
[40] http://ccv.nuigroup.com/

variety of tools for accessing raw touch data and generating higher-level information such as speed of travel, point history, etc. A number of interesting projects have sought to utilize this data for musical performance, such as the geometrically-driven DMIs of Kevin Schlei[29] [19] and the examples cited in [12]. However, the vast majority of applications fail to make use of this data in any meaningful way.

One possible reason is the volatility of geometrically-derived data. Some of the examples mentioned use algorithms that calculate, for example, 'angle to previous point' or 'distance to first touch'. There is a danger in mapping this kind of data to any kind of prominent synthesis parameter as it is highly-dependent upon the order of touch initialization upon the surface – two perceptually-identical gestures can quite easily result in the establishment of totally different point-relationships.

Another reason is the difficulty of implementing high-level 'gestural' response systems. Anyone intending to design a gesture-based MT DMI must have (at least) a competent grasp of the hardware and protocol being used, coordinate geometry and intermediate programming concepts such as event handling, control flow and multi-threading. This overhead is a significant deterrent to any musician, composer or performer who wants to explore MT interaction. There are many solutions which offer high-level gesture support, but none specifically-designed for musicians.

Figure 1 is a purely illustrative graph which places some popular approaches to MT music control on a two-dimensional continuum. The different systems are situated according to the programming expertise required (vertical axis) and how closed-off they are (horizontal). Naturally, these systems all function very well in certain contexts – the purpose of this diagram is to suggest how these approaches relate to one another and also to establish a point at which there may be a deficit of resources.

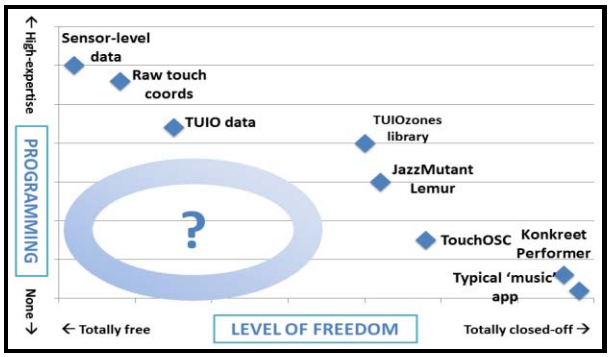

Figure 1. Comparison of MT development options

The area in the lower left of the diagram has been identified as an ideal space to aim for when developing tools for DMI design. An approach that could be placed within this area would allow more freedom to experiment, with less specialist requirements and prescriptive boundaries influencing the design process.

The ability to engage in 'reflective practice' is indispensable to the digital musician [4] – therefore, a fluid transition from evaluation to implementation (and indeed all stages of the DMI design cycle) is vital [1, 16]. Tools which allow rapid and transparent development ensure that the designer can concentrate upon the critical aspects of mapping and user experience.

4. RESEARCH TOOLS

This section provides a brief introduction to two complementary projects which are under development. Both pieces of software are designed with non-graphical interaction techniques in mind – treating the MT surface as a sensitive data-gathering device rather than a canvas for widget-based interactions. The collective goal of these projects is to develop research tools which will enable future studies into MT interface design for music performance.

4.1 SurfacePlayer

SurfacePlayer is being developed within *Processing*[41]– a development environment which uses easily-readable syntax and tends to be popular among artists working with sound and visuals. The program provides a selection of gesture-handling functions which expand-upon the current TUIO library[42]. The functions generate useful high-level information in response to user gestures which can then be routed to external synthesis software.

The objective of SurfacePlayer 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 set of high-level tools will allow designers to concentrate their attention on more musically-critical aspects of the interface, such as mapping and feedback, and encourage more experimentation with MT music performance.

More information on SurfacePlayer can be found in [12] and on the author's website[43].

4.2 OSCar

OSCar is an iOS application for realtime music control which incorporates the built-in gesture-recognition algorithms of the Cocoa Touch UI framework[44].

4.2.1 Implementation

OSCar works by analyzing raw touch information on the mobile device itself. The resulting high-level data is sent via a wireless connection in an easily-readable OSC format which describes the properties of commonly-used MT gestures (parameterized multi-tap, swipe, pinch, pan), orientation sensors (accelerometer values) and other abstract properties (number of touches, quantized location). Different gesture recognizers may be toggled on and off to accommodate easy testing and mapping of data to musical parameters. The corresponding preferences menu is external to the application itself in order to prevent accidental changes during live performance.

The gesture-recognizers all operate independently and simultaneously – this approach, coupled with the simple message formatting, allows rapid experimentation with combinations of different gestures and mappings. This loosely-coupled set of behaviors is analogous to the selection of widgets in an application such as *TouchOSC*.

The visual aspect of OSCar is minimal and serves only to provide basic cues for the user. There are no widgets or menus - the priority is to provide rich, modeless feedback [3] in a non-prescriptive application.

[41] http://processing.org
[42] http://www.tuio.org/?processing
[43] http://aColossalEmptySpace.com
[44] http://developer.apple.com/technologies/ios/cocoa-touch.html

4.2.2 Preliminary results

The examples which will be made available online demonstrate some simple performance scenarios[43]. One predicted benefit of the non-prescriptive application design is the potential for discovery of new gesture combinations. Some examples of these combinations can be seen in the videos - the 'running' swipes are used to control a beat-slicing effect and the 'pinch' gesture controls the cutoff of a LP filter. These are movements that, independent of any graphical reinforcement, resemble real-world physical manipulations and could easily be used in the design of user-defined gestures (a comprehensive summary of related issues can be found in [22]).

Both pieces of software are already being employed to investigate MT mapping strategies – OSCar has been successfully used to perform music from genres as diverse as drum & bass and drone-based ambient. The research possibilities are vast – outside of the aforementioned musical applications, OSCar represents an interface development system that permits users to rapidly prototype and evaluate different combinations of gestures. The ability to quickly and clearly define behaviors in response to complex touch events can be used to elucidate the mapping stage of DMI design without the significant workload usually required to employ MT interfaces in this way.

A more detailed discussion of musical implementations, mapping schemes and performance methodology will be the subject of future papers.

5. CONCLUSION

A preliminary investigation into touch-based music interfaces has shown that, in general, successful design approaches have been strongly-linked to the implicit capabilities of the devices themselves. This study has also illustrated how this kind of engagement with multi-touch devices has been somewhat hindered by the persistence of the GUI paradigm. We have therefore proposed a different approach which emphasizes the implicit characteristics of the MT surface and establishes a philosophy of design centered upon its strengths. We have introduced two tools – SurfacePlayer and OSCar – which will be used to explore these possibilities in future research, with an enhanced focus upon the user experience and musical process itself.

6. ACKNOWLEDGMENTS

This work is funded by the John & Pat Hume Scholarship at NUI Maynooth, Ireland.

7. REFERENCES

[1] Barbosa, J., Calegario, F., Magalhães, F., Cabral, G., Teichrieb, V. and Ramalho, G., Towards an evaluation methodology for digital musical instruments considering performer's view: a case study. In *Proceedings of the 13th Brazilian Symposium on Computer Music (SBCM'11)*, Vitória, ES. Brazil, 2011.

[2] Chadabe, J. *The Past and Promise of Electronic Music*. Prentice Hall, New Jersey, 1997.

[3] Cooper, A. and Reimann, R. *About Face 2.0: The Essentials of Interaction Design*. Wiley, 2003.

[4] Hugill, A. *The digital musician*. Routledge, 2008.

[5] Hunt, A. *Radical User Interfaces for Real-time Musical Control*. PhD Thesis, Department of Electronics, University of York, 1999.

[6] Jacob, R.J.K., Sibert, L.E., McFarlane, D.C. and M. Preston Mullen, J. Integrality and separability of input devices. *ACM Trans. Comput.-Hum. Interact.*, *1* (1). 3-26. 1994.

[7] Kaltenbrunner, M., Bovermann, T., Bencina, R. and Costanza, E., TUIO - A Protocol for Table-Top Tangible User Interfaces. In *Proceedings of the 6th International Workshop on Gesture in Human-Computer Interaction and Simulation (GW'05)*, Vannes, France, 2005.

[8] Keller, D., Barreto, D.L., Queiroz, M. and Pimenta, M., Anchoring in Ubiquitous Musical Activities. In *Proceedings of the International Computer Music Conference 2010*, New Orleans, USA, 2010, 319-326.

[9] Keller, D., Flores, L.V., Pimenta, M.S., Capasso, A. and Tinajero, P. Convergent Trends Toward Ubiquitous Music. *Journal of New Music Research*, *40* (3), 2011. 265-276.

[10] MacKenzie, S. Motor Behaviour Models for Human-Computer Interaction. In Carroll, J.M. ed. *HCI Models, Theories, and Frameworks: Towards a Multidisciplinary Science*, Morgan Kaufmann, San Francisco, 2003, 27-54.

[11] Maupin, S., Gerhard, D. and Park, B. Isomorphic Tessellations for Musical Keyboards. *Sound and Music Computing 2011*, Padua, Italy, 2011.

[12] McGlynn, P., Analyzing Multi-touch Data for Expressive Musical Performance. In *Proceedings of the 13th Brazilian Symposium on Computer Music (SBCM'11)*, Vitóia, ES. Brazil, 2011.

[13] McGlynn, P., Towards more effective Mapping Strategies for Real-Time Musical Control. In *Proceedings of the 9th Annual Linux Audio Conference (LAC'111)*, Maynooth, Ireland, 2011, 93-98.

[14] Miranda, E.R. and Wanderley, M. *New Digital Musical Instruments: Control and Interaction Beyond the Keyboard*. A-R Editions Inc., Wisconsin, 2006.

[15] Norman, D. A., *The Design of Future Things*. Basic Books, New York, 2007.

[16] O'Modhrain, S. A Framework for the Evaluation of Digital Musical Instruments. *Computer Music Journal*, *35* (1). 28-42.

[17] O'Modhrain, S. *Playing by Feel: Incorporating Haptic Feedback into Computer-Based musical Instruments*, Ph.D. Thesis, Stanford University, 2000.

[18] Saffer, D. *Designing Gestural Interfaces: Touchscreens and Interactive Devices*. O'Reilly Media, Sebastopol, CA, 2009.

[19] Schlei, K. 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 (NIME'10),* Sydney, Australia. p. 136-139.

[20] Sinclair, S. A guitar-inspired touch pad controller. http://www.music.mcgill.ca/~sinclair/touchpad-guitar_sinclair.pdf

[21] Teiche, A., Rai, A., Yanc, C., Moore, C., Solms, D., Cetin, G., Riggio, J., Ramseyer, N., D'Intino, P., Muller, L., et al. *Multi-touch technologies*. NUI Group, 2009.

[22] Wobbrock, J.O., Morris, M.R. and Wilson, A.D. User-defined gestures for surface computing. In *Proceedings of the 27th international conference on Human factors in computing systems*, ACM, Boston, MA, USA, 2009, 1083-1092.

[23] Zbyszynski, M. An Elementary Method for Tablet. In *Proceedings of the 2008 international conference on New interfaces for musical expression*, Genova, Italy, 2008, 245-248.

[24] Zbyszynski, M., Wright, M., Momeni, A. and Cullen, D. Ten years of tablet musical interfaces at CNMAT. In *Proceedings of the 7th international conference on New interfaces for musical expression*, ACM, New York, New York, 2007, 100-105