Rub Synth : A Study of Implementing Intentional Physical Difficulty Into Touch Screen Music Controllers

Ozan Sarıer
Center for Advanced Research In Music
Istanbul Technical University
ITU Macka Campus, 34367
Istanbul, Turkey
sariero@itu.edu.tr

ABSTRACT

In the recent years many touch screen interfaces have been designed and used for musical control. When compared with their physical counterparts, current control paradigms employed in touch screen musical interfaces do not require the same level of physical labor and this negatively affects the user experience in terms of expressivity, engagement and enjoyment. This lack of physicality can be remedied by using interaction elements which are designed for the exertion of the user. Employing intentionally difficult and inefficient interaction design can enhance the user experience by allowing greater bodily expression, kinesthetic feedback, more apparent skill acquisition, and performer satisfaction.

Rub Synth is a touch screen musical instrument with an exertion interface. It was made for creating and testing exertion strategies that are possible by only using 2d touch coordinates as input and evaluating the outcomes of implementing intentional difficulty.

This paper discusses the strategies that can be employed to model effort on touch screens, the benefits of having physical difficulty, Rub Synth's interaction design, and user experience results of using such an interface.

Keywords

NIME, human computer interaction, interface design, touch screen, multitouch, mobile musical instrument, proprioception, kinesthesia, muscular tension, intentional difficulty, exertion interfaces.

1. INTRODUCTION

Performing with physical instruments require varying amounts of physical effort and most instruments accepts and responds to great magnitudes of forces. On the other hand, most of touch screen only instruments designed up to date neither require the same level of physical effort nor have the sensory capabilities to directly capture them.

Although this lack of necessity for effort is considered favorable in terms of efficiency from the viewpoint of classical HCI, it is unfavorable for musical instrument design as the presence of physical effort has an important role in many aspects of musical interaction [4].

Without auxiliary sensors, the capability of a touch screen is too limited for directly requiring and sensing the extremes of effort a performer can produce. Intentional physical difficulty can be implemented within a touch screen interface by going the opposite way of interaction efficiency and requiring effort over time,.

Game developers have been employing methods like this in mini-exertion interfaces in video games for a long time. A common example is requiring the player to continuously tap a button or move a joystick/pointer above a certain pace while an action is being performed (usually linked with in-game physical effort).

Effort is an essential part of music performance and it is closely tied with expression [6]. Tanaka states that, by varying the exerted effort of gestures that are essential to sound production, expressivity can be shaped in an articularly fashion. This exertion of effort is a key element for audience perception of a performance [9]. Audiences are accustomed to observing physical effort in performances, which they eventually link with commitment to the music and emotional intensity [4]. Implementing intentional difficulty into a touch screen music instrument can greatly enhance expressivity and visual communication with the audience.

Studies related to physical activity in video games show that body movement, meaningful physical actions, physical challenge and exertion has significant positive effects on fun, enjoyment, pleasure and engagement of video gaming [1] [7] [8][5]. The same positive effects can be expected for music performance with non-trivial physical difficulty. The satisfaction in music performance partly comes from the difficulty involved just like video games. A suitable level of difficulty is beneficial for enjoyment [4]. Physical activity is also pleasurable on its own when it doesn't cross over to the unareobic region of exertion. It's been observed that muscular activity can arouse pleasure and brief episodes of pleasure are experienced whenever a significant exertion ends.[2]. This might explain the enjoyment that is experienced after performing a physically difficult act on a performance.

Another role of effort on satisfaction is the investment factor. With increased physical difficulty and invested energy, perceived value of a musical act also increases. For example, sounding the very high registers of a wind instrument are commonly valued higher than the sounding of a relaxed register's sound by both performers and audience. Completing a task with adhered value can be expected to evoke greater satisfaction.

Varying the physical difficulty amongst various features of a virtual instrument can lead to the possibility of mastery. Though both the beginners and the experienced can play such an instrument, some of the instrumental features would require a seasoned player's skills.

Lastly, kinesthetic feedback increases proportional to the muscular exertion. Mechanical, chemical and temperature "signals" are created in proportion to the muscular activity, which creates a peripheral sensation descriptive of the act performed [2]. This feedback is beneficial towards creating virtual instruments that have distinctive "feels".

2. DESIGN PATTERNS FOR IMPLEMENTING INTENTIONAL DIFFICULTY

The amount of the range of effort that can be captured in an instant by a touchscreen is extremely limited when no auxiliary sensing such as force sensors, accelerometers and alike is present. As the amount of work that can be sensed by a touch screen in an instant is quite fixed, the main strategy towards creating a touch screen exertion interface involves requiring input over time. The simplest example to this is requiring a swipe or double tap instead of a tap. The physical work done by swiping or double tapping is greater than just tapping once. By varying the velocity, count and frequency demands of a gesture, different levels of physical difficulty can be achieved.

2.1 Excitation Gestures

The gestures listed below are for activation of a simple action or a complex model that will be discussed in the section 2.2 .

2.1.1 Consecutive tapping

Instead of a single tap, a series of taps can be demanded from the performer. The frequency of the tapping is the dominant factor determining the difficulty

2.1.2 Continuous tapping

Instead of a single tap or a series of them, a continuous stream of tapping can be required

2.1.3 Swipes, Consecutive Swipes and Continuous Swipes

Swipes can be adjusted to a variety of physical demand. The velocity, distance and the geometry of the swipe determine the difficulty. Multi-finger swipe like gestures like pinching or squeezing can also be adjusted in difficulty.

For increased difficulty, consecutive swipes can be demanded. When the following swipe is in opposite direction of the one before, the inertia of the hand comes into effect and adds to the difficulty created.

2.1.4 Rubbing

Rubbing can be considered a sub-class of swipes, as it consists of an endless swipe. The user can be expected to do a rubbing gesture between two poles, a circular rubbing gesture or one with a random path. The inertia of the hand and the friction between the finger and the screen contribute to the physical difficulty. Again, path, distance, speed and frequency can be adjusted for varying levels of difficulty.

2.2 Leaking Value Pool Model

To further the energy expenditure, the excitation gestures above can be connected to leaking value pool models. The name of this model comes from the analogy of filling a container with liquid. While the excitation gestures fill in the pool and raise the value, a predetermined leak empties the pool thus reducing the control value. With this model, the user has to continuously spend energy to keep a value where he/she wants. By adjusting the filling/leaking rate, the difficulty can be adjusted.

To add more levels of difficulty, leak rate can go up proportional to the value held in the pool.

2.3 Diminishing Input Model

This model is similar to the leaking value pool. In addition to employing a leak mechanism, this model employs a mechanism in which the input gets scaled to a lower value over time. After the initial excitation, the user's inputs become gradually more ineffective as the time passes thus making sustaining values even harder.

2.4 Gating of the Input

In order to impose a starting condition, users input can be gated before going into either of the methods. For example, the users input can be ignored if it is under a certain velocity or frequency. After the initial condition is met, the gating can be dismissed. This can be used to simulate functions that are harder to excite.

2.5 Using Multi-Stroke Connections Instead Of On-Screen Buttons:

With this method, instead of simply touching a button, the user has to drag his finger into various boxes to trigger an action. The boxes can be placed along the opposing edges of the screen, both making them easy to find and making the swipe distance longer.

If the hand user's hand travels more than 5 centimeters and the user is in a hurry to complete the action, the whole arm (including the fingers, wrist, upper arm and shoulder) is used. Compared to activating a simple button with a direct touch, this method requires tremendous activity. It is also easier to memorize as a motor action and it has higher repeatability when compared to pinpointing a small button amongst others. In addition, target boxes can be arranged in such a way to suggest a langue. For example going through sample1, reverb, and bypass boxes turns the reverb effect on the sample number 1 in to a bypassed state. Although this implementation doesn't offer any expressive benefits, it is still beneficial in terms of general pace, satisfaction, audience communication and kinesthesia.

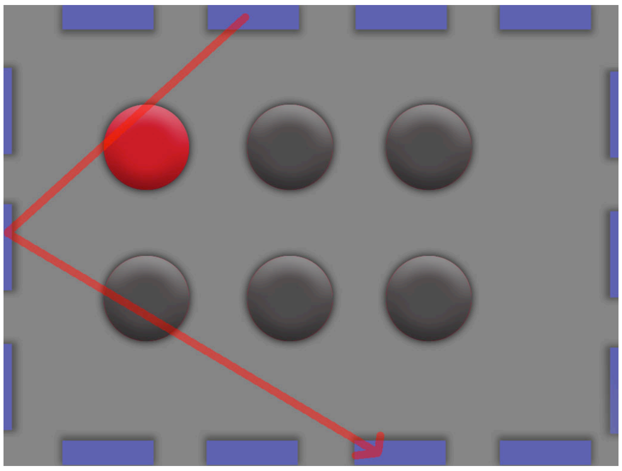
Figure 1. Rub Samples sound selection a1-b2-c3.

3. RUB SYNTH

Rub Synth is a touch screen musical instrument made for the iOS operating system. It is made for demonstrating the usage of methods described in the section 2 of this paper and evaluating the benefits discussed in section 1.

Rub Synth uses Apple's UIKit for display, custom gesture recognizers derived from Apple's classes for touch handlin, and PDLib for the synthesis and implementing depleting value models[3].

3.1 Implementation and Functionality:

Rub Synth employs a design which allows/encourages users to "rub" the on-screen keys instead of just tapping on them for activation. When a user rubs one of the keys, the corresponding keys slowly start to change color, whilst the corresponding sound slowly fades in. Rubbing excitation gesture is tied to a leaking value pool model. While the user tries to rub and activate a sound, this system tries to fade the sound out.

In order to sustain a sound, the user has to continuously spend physical effort. The amount of difficulty in activating and sustaining a sound is directly proportional to the strength of the deactivation system. The difficulty can be set individually for all the keys. The difficulty variance of playing different registers of real world instruments can be mimicked with this ability. In the alpha version of Rub Synth, the second octave requires significantly more effort than the first. The difficulty is also varied through the second octave. In the second octave, the difficulty is proportional to the pitch in question.

The user can vary the vigorousness of the rubbing gesture to change the attack of the sound. The attack portion of the rubbing difficulty can also be set individually for each key.

In the current implementation of Rub Synth, a user can either tap on a key to initiate a preset attack, then continue to rub the key for further crescendo and sustain or initiate a sound by rubbing and deciding on the envelope of the attack themselves. By regulating the strength of the rubbing, users can sustain the sounds, make the sounds stronger/louder and move between different steps of dynamic markings.

The rubbing gesture is allowed to exceed the limits of the graphical are of a key. The user only has to land a finger on the correct key once, after that he/she is free to exceed the limits of the graphical area. As long as the finger doesn't go off the screen, the rubbing gesture is continuously tracked –even the whole screen can be vigorously rubbed. Multiple keys can be rubbed at the same time even in different directions.

Another implemented feature is the detection of the direction of the rubbing gesture. Rub Synth's framework can differentiate between diagonal, lateral or vertical rubbing. Although it is possible to differentiate between all three, whilst performance it is hard to consciously perform rubbing on three different directions. It is very easy to accidentally perform a diagonal rubbing gesture while trying a vertical or a horizontal one. In this current implementation of the Rub Synth, horizontal rubbing is linked to controlling the envelope of the sounds while the vertical gesture is reserved for linking to other synthesis parameters.

An additive synthesis engine is responsible for the synthesis. The balance of the partials is affected by the excitation of the keys. Upper partials increase in volume more than the lower partials with rubbing. This allows the user to modify tone color simultaneously while controlling the envelope.

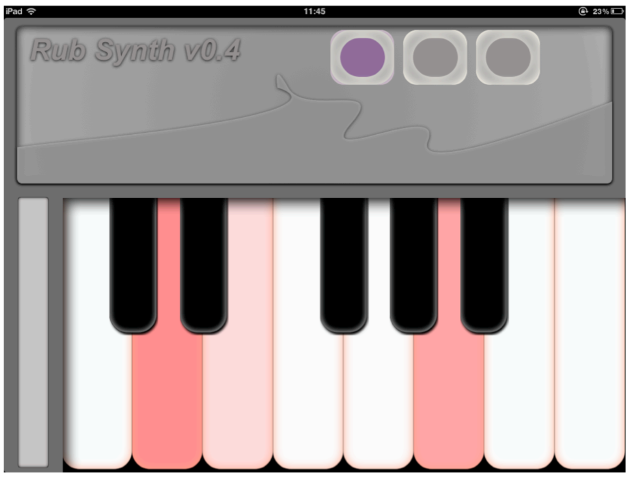

Figure 2. Rub Keys interface showing color display

4. CONCLUSION AND FUTURE WORK

4.1 Preliminary testing:

The alpha version of the Rub Synth was evaluated informally by a small test group of 14 people at the time of writing. The test group included both people with musical backgrounds (8 people) and non-musicians (6 people). Each participant was asked to use two versions of Rub Keys for 10 minutes after a brief demonstration (Some of the testing was done in groups. In that case the time was shared amongst the testers.). While one of the versions is identical to the one described in this paper, the other version didn't employ the rubbing gesture and employed an effortless method to set the sustain instead (swiping up on a key set the sustain, the length of the swipe determined the level of the tone). Afterwards, participants are asked to compare the two versions in terms of physical difficulty, fun while performing, expressiveness, ability to channel emotion and engagement.

All of the testers reported in favor of the version with the rubbing gesture. They reported that they found it more physically demanding, more pleasurable, more expressive and easier to engage. People with musical background and non-musicians reacted differently to some of the implemented difficulty elements(further discussed in section 4.2). Although the test group is too small to come to a complete conclusion, the positive results encourage us to further our research.

4.2 Future direction:

The preliminary tests show that the suggested interaction models were successful in creating artificial physical difficulty. In the near future, Rub Synth will enter the beta stage and will be put into a formal test with a larger test group.

During the preliminary tests, people with musical background specifically reported that the exertion had an affect on expressiveness. With the large test group, we would like come up with qualitative and quantitative findings to support this.

We also observed that the musician test group enjoyed the technical challenges caused by the poor coding at the time of early testing. One of those challenges was that the rubbing has to be done nearly perfect vertically, which added a steering difficulty. Another challenge was that exciting a key slowly without a predetermined attack required landing a finger on a key in motion. This mini-gesture has a learning curve steeper than the rest of the gestures. While the musician group enjoyed the challenge of the mini-gesture and the increasing difficulty in the second octave, the non-musicians found it a nuisance. One of our future directions will be researching the difference on reception of difficulty between non-musicians and established performers. Understanding what kind of physical difficulties increase the fun factor for non-performers, and what kind does not can be important when designing instruments with a high adoption rate in mind. Another correlation we want to look for is between touchscreen instruments' survivability and the amount of physical difficulty they have. We suspect that implementing the right amount of intentional difficulty into touch screen music instruments can be beneficial in terms of survivability.

Another future research topic of ours is the bounds of the "aerobic" region of effort on a touchscreen. We would like to gather data on what amount of effort falls into the aerobic region and what amount falls into the not pleasurable and excessive region. With the alpha version of the RubKeys, we found out that how the touch screen device is held directly affects the difficulty of implemented physical demand of various gestures. For example, users are capable of performing faster rubbing gestures while the device rests on a surface. When the device is held, the maximum rubbing frequency decreases. In the future, a system, which tracks the devices position and modifies the difficulty according to the usage, can be developed.

Lastly, we are hoping to make the custom gesture recognizer libraries available online when RubSynth is release ready. We wish to see many new instruments that implement intentional difficulty in the future.

4.3 Inspiration and expansion of the listed design patterns onto other interfaces:

While the design patterns discussed above are thought with software interfaces drawn on touch screens in mind, they can be used with many other software or hardware interfaces. In fact, the main inspiration for this paper's topic came from an experimental thermosensitive controller called the "Rock Bottom" [10].

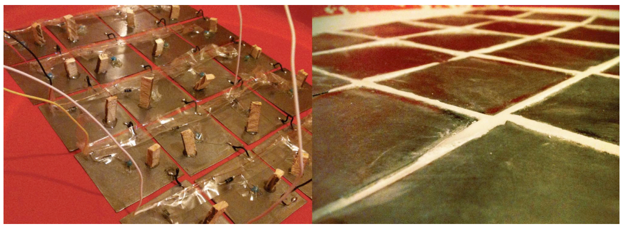
Figure 3. The Rock Bottom. Sensor grid before casting (left), top surface (right)

The Rock Bottom consists of a 5x5 grid of metal plates with resistive temperature sensors which is embedded in a block of plaster cast. The sensors are both slow and insensitive, so the users have to rub on the metal pads and warm them up to interact. When the user interaction ends, the pads naturally cool down, thus require input back again. Pads are painted with thermochromic paint and change color upon warming up for visual feedback. Although the intended interaction model was the rubbing action, tests showed that laying one's hand on a pad was faster at heating it than the rubbing action. Still, the test users spent extensive efforts to keep multiple pads warmed. As the sensing limits were greater than the warmth that can be generated by warming the pad with a stationary hand, the users resorted to improvised methods such as blowing hot air onto the pads. The Rock Bottom is similar to Rub Synth in terms of interaction methods, value holding (although it is totally analog), exhaustion ability and its inability to detect force. The two interfaces are also similar in terms of the muscle work and tension created when the users use a rubbing gesture.

The discussed patterns can also be made to work with existing hardware controllers. Possible candidates range from a simple device with just a momentary button to more complex ones such as a gamepad, joystick and many others. For example users can be asked to quickly alternate between two poles of a joystick or any other 2d control. Combined with the value holding and depletion strategies discussed in the section 2, this would have a result similar to the rubbing gesture.

The patterns discussed can be put to use even without a physical controller or a visual software interface present. One example would be employing the "Leaking Value Pool" and the "Diminishing Input" strategies in a skeleton tracking based performance. Like requiring rubbing or hastily tapping, certain alternating or repeating body movements can be asked from the performer, and coupled with the strategies above to further tune the difficulty and exhaustion demands.

Lastly, additional difficulty can be implemented on already existing augmented instruments. One example to such instruments is the TouchKeys, which is a keyboard with capacitive touch sensing capabilities [11]. The interaction models that are in the RubSynth can be implemented on TouchKeys to further augment the already interesting instrument.

4.4 Conclusion:

Suggested interaction models were successful in creating artificial physical difficulty. Using effort as an input has strong ties with expressiveness, emotion and enjoyment. Novel modalities of expression and interaction can be employed with the interaction methods described in this paper.

5. REFERENCES

[1] Bianchi-Berthouze, N., Kim, W. W., & Patel, D. (2007). Does body movement engage you more in digital game play? And Why?. In *Affective Computing and Intelligent Interaction* (pp. 102-113). Springer Berlin Heidelberg.

[2] Cabanac, M. (2006). Sensory pleasure optimizes muscular work. *Clinical and investigative medicine*, *29*(2), 110.

[3] LibPD – http://github.com/libpd/libpd

[4] McDermott, J., Gifford, T., Bouwer, A., & Wagy, M. (2013). Should Music Interaction Be Easy?. In *Music and Human-Computer Interaction* (pp. 29-47). Springer London.

[5] Mueller, F., Agamanolis, S., & Picard, R. (2003, April). Exertion interfaces: sports over a distance for social bonding and fun. In *Proceedings of the SIGCHI conference on Human factors in computing systems* (pp. 561-568). ACM.

[6] Ryan, J. (1991). Some remarks on musical instrument design at STEIM. *Contemporary music review*, *6*(1), 3-17.

[7] Scollan, R. (2007). Designing a Pleasurable Interface: Emotion in Human-Computer Interaction. *Interaction Design and Information Architecture, University of Baltimore: MD*.

[8] Shinkle, E. (2008). Video games, emotion and the six senses. *Media, culture, and society*, *30*(6), 907.

[9] Tanaka, A. (2000). Musical performance practice on sensor-based instruments. *Trends in Gestural Control of Music*, *13*, 389-405.

[10] Sarıer, O. (2011) The Rock Bottom - www.ozansarier.com/rockbottom.html

[11] McPherson, A. (2012). TouchKeys: Capacitive multi-touch sensing on a physical keyboard. In *Proc. NIME.*