

COMBINING DJ SCRATCHING, TANGIBLE INTERFACES AND A PHYSICS-BASED MODEL OF FRICTION SOUNDS

Kjetil F. Hansen

Royal Institute of
Technology
Speech, Music and
Hearing

Marcos Alonso

Universitat Pompeu
Fabra
Music Technology
Group

Smilen Dimitrov

Aalborg University
Copenhagen
Department of
Medialogy

ABSTRACT

This paper reports on two Short-Term Scientific Missions (STSM) in the ConGAS European Cost Action. Several sound models and (musical instrument) interfaces were combined to study how DJ gestures of scratching can be applied to new situations. In one experiment, the gestures were used to control a physics-based model of friction sounds, for instance to simulate the sound of a bowed violin string. In the other experiment, DJ gestures from the program Skiproof were adapted to the Reactable framework, allowing users to perform the complicated DJ gestures with ease. This paper describes each model and the adaption and implementation of the models.

1. INTRODUCTION

Modern DJs often need to acquire great instrument control to accommodate the problems of playing an unpredictable and in many ways unreliable instrument such as the turntable. Further, it has been a long-time tendency among hip-hop DJs to perfect their skills and perform virtuoso playing styles, like scratching and beat juggling.

We wanted to test how the very defined gestures of DJs can be applied to other applications and areas. To achieve this, existing models of scratching were combined with two other systems: the reactable and a physics-based model for friction sounds.

It is our hope that the merging of the different software and hardware environments will allow users to explore whether control derived from one type of a physical interaction can be applicable to a different kind of instrument. In these cases the intent is to be able to use the control derived from hand motions during turntable scratching, and apply them to parameters of a physical model of friction sounds, for instance to produce bowed-string-like sounds, and to reuse the same gestures in a new, intuitive interface.

Within the frames of the European Cost Action 287¹ two STSM² research visits to KTH from UPF and AUK³ were arranged, with the aim to combine scratch models

with other systems. This paper will briefly describe the results from those visits, which are reported in [1] and [4].

2. METHODOLOGY

The method was quite similar for both STSMs. The first tasks were to make the systems described below compatible and communicate, and then use control data from one model to drive the other system. After the implementations, simple experiments and explorations of the new possibilities were performed.

2.1. Systems and models

Three systems were included in the projects plans of the research visits:

- *Skiproof*, a DJ application and virtual turntable developed by Hansen and others [5] at KTH
- The *Reactable*, a musical instrument and interface developed by Jordá and others [6, 7, 8] at UPF
- The *friction model*, developed by Stefania Serafin and others [9, 10, 11] at AUK

The following sections give brief introductions to each model or system.

2.1.1. *Skiproof and scratch models*

Skiproof, a patch written for Pure Data (Pd)⁴, is both a virtual turntable and mixer, and an application for exploring the musical language of DJs. The main purpose is to “scratch” sound files using gesture controllers of different kinds. Its most important feature is to allow execution and control of the scratch technique models, and to do this with customizable interfaces and controllers. The collection of scratching techniques are based on analysis of recordings by several professional DJs. Each of these techniques consists of a record movement with the corresponding crossfader movement.

¹ ConGAS (Gesture Controlled Audio Systems), www.cost287.org

² Short-Term Scientific Mission

³ The Royal Institute of Technology (KTH), Universitat Pompeu Fabra (UPF), Aalborg University Copenhagen (AUK)

⁴ <http://puredata.info>

The graphical user interface, made with GrIPD⁵, serves as a collection of controllers a DJ normally would expect to find in addition to novel ones. All the elements in the GUI and the model can be accessed by any controller. Until now, Skipproof has been played by a number of devices, including computer peripherals and tablets, MIDI devices, Max Mathews' *Radio-Baton* [3] and various sensors. The system is shown in Figure 1.

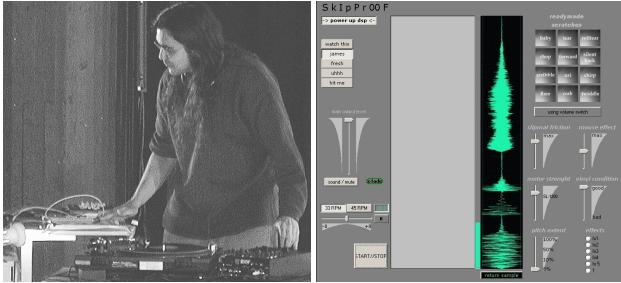


Figure 1. The left image shows a DJ in a live performance with Skipproof using a variety of sensors and interfaces such as Radio-Baton and light sensors. Right is a screenshot of the GUI in SkippooF. The performer has access to the controls normally found on a turntable and audio mixer with some additional controllers.

2.1.2. The Reactable

The reactable is a collaborative electronic music instrument with a tabletop tangible multi-touch interface. Several performers can control the instrument simultaneously by moving and rotating physical objects on a round table surface. Each reactable object represents a modular synthesizer component with a dedicated function for the generation, modification or control of sound. A simple set of rules automatically connects and disconnects these objects according to their type and affinity and proximity with the other neighbors.

The hardware is based on a translucent, round multi-touch surface. A camera situated beneath the table, continuously analyzes the surface, tracking the player's finger tips and the nature, position and orientation of physical objects on its surface. A projector, also from underneath the table, draws dynamic animations on its surface, providing a visual feedback of the state, the activity and the main characteristics of the sounds produced by the audio synthesizer.

The reactable intends to be an intuitive instrument or interface for the performer. While easy to begin playing with, it aims to be sonically challenging and interesting and allow development of instrumental skills for making advanced performances.[6]

2.1.3. The friction model

The topic of friction modeling is often addressed in the fields of computer graphics, haptics, and sound modeling.

⁵ <http://crca.ucsd.edu/~jsarlo/gripd/>

The main objective is the simulation and rendering of sliding contact of rigid bodies. Sound synthesis from physics-based models of friction have many applications.[2]

The aim of this model is to create a physically based friction synthesizer able to reproduce the behavior of rubbed surfaces. The friction model combines waveguide synthesis with recent research results on friction interaction modeling where a modal resonator is excited by a sophisticated elasto-plastic friction model.

The model exhibits a physically consistent behavior. For example, when simulating a bowed string, the force versus velocity characteristics has a behavior that has been observed in experiments on real instruments.[9]

A merge between scratching and the sound model would potentially allow a user to experiment in producing physically based friction sounds through re-mapping of movements derived from DJ gestures.

2.2. Implementation

Although the method that was envisaged for both projects was basically the same, the implementation was done differently in the end. While the reactable and Skipproof merged mostly on a conceptual level, the friction model and Skipproof were communicating more directly. For the second project, we also prepared for control of the models using other gesture interfaces.

2.2.1. Software and controllers

All three authors had various programming background, and the three systems are programmed within different frameworks. Skipproof is developed in Pd and GrIPD, reactable has an open structure that allows for multiple programming paradigms, including Pd and Supercollider⁶, while the violin friction models was implemented in Max/MSP⁷. Common to all environments is that they offer fast and convenient methods for building prototypes and applications.

Gestural control of the systems was one of the main areas of interest. For the reactable project, the limitations were already set by the existing concept, and the task was to 'play' scratch models by using the physical objects. For the friction model project, it was the other way around, using DJ gestures to control a sound model. Additionally, we used a Wacom Intuos⁸ tablet for testing purposes.

2.2.2. Skipproof and reactable

We wanted to focus the work on finding the best way to control the DJ scratch model using the reactable interface and how to incorporate it into the reactable modular paradigm.

In order to develop and test the system, a small and portable version of the reactable was prepared, see Figure 2. The system was a small table with a transparent

⁶ <http://superCollider.sourceforge.net>

⁷ <http://www.cycling74.com>

⁸ <http://www.wacom-europe.com/int/products/intuos/>

surface and internal illumination, a webcam at the bottom facing up and a reduced set of tangibles marked with fiducial markers. We decided to re-implement the KTH model using the reactable audio engine instead of adapting it.

We started with the implementation of the speed controller. This object simulates the way a scratch DJ moves the record on the turntable forward and backward. Simple inertia function were added to the loop player to simulate the way a record spins on a turntable. Different envelopes to the speed controller were implemented, but instead of using prerecorded gestures like in Skipp proof, algorithmic models were used, and this way the player can create smooth transitions between the different gestures.

For the amplitude controller we decided to use a set of predefined patterns. Based on analyzed performances, a list of common patterns of fader hand movements were created. Some random variations to the movement amplitude were added to make it sound more natural.

In all, three objects were used to ‘scratch’ with; the sound sample object (A), the record movement object (B) and the fader movement object (C), see Figure 2.

Finally, the temporal synchronization between the speed and the amplitude controllers was implemented according to the conceptual framework of reactable.



Figure 2. The reactable prototype version for scratch implementation in the left picture. The right image is a screenshot showing the visualization projected onto the table from underneath. The top, square object is the loop player that contains a sound file. The bottom left, circular object is the amplitude controller. The bottom right object is the record speed control. Interaction between the objects is visualized with lines and waveform representations between the object bodies in real-time. The circles around each object convey information on the state of the object.

2.2.3. Skipp proof and friction model

The Skipp proof program uses the playback speed and fader volume of a turntable (as recordings or modeled parameters), in order to render a scratching sound from an arbitrary sample. These two parameters could be readily applied to the frequency and amplitude of a friction model, however it would also be interesting to see whether there are any additional playable combinations—like applying these control parameters to velocity or force, which are available as control parameters of the model.

As such, much of the work carried out was in solving the programmatic issues in being able to map the desired control signals to the desired audio rendering engine - and providing an initial user interface for it. The development went beyond the issues of connecting the models, and into providing a generic interface, where four different types of signal inputs (recorded, modeled, or sensor input) could be patched to two different audio rendering engines (friction model or Skipp proof).

For each of the possible mappings between control input and audio output, a patching matrix exist to allow the user to choose which input parameter should be mapped to which output parameter.

One of the primary technical problems in developing the interface between Skipp proof and the friction model, was to port from Max/MSP to PD. The first attempt was to use the Flex⁹ library, which would generate a cross-platform model object. This approach proved too time consuming due to difficulties in compiling Flex objects, so the model was manually ported.

A Wacom tablet was added as a way to safely test possible sensor interaction. In addition, the Wacom can also be seen as an exciting interaction device for both friction and scratching, especially considering the multiple real-time parameters that could be used for musical interaction; x and y position, pressure, orientation and tilt. To obtain an interface to the Wacom tablet in PD, the wintablet¹⁰ object was used, which originally does not support orientation and tilt; however, some experimental efforts were made in modifying the wintablet sourcecode, to allow using even these parameters.

3. RESULTS

Both research visits turned out rewarding and successful. The reactable objects worked very well in tests, and the friction model could be driven by DJ gestures. Nonetheless, this non-formal evaluation was only done during the course of the projects by the three authors and colleagues.

3.1. The reactable

The resulting system is divided into three main components; each one of those is controlled with a tangible object on the table. The system mimics a real scratching DJ scenario where one object represents the record on the turntable (loop player), another one represents the hand the DJ uses to move the record (speed controller) and the last one represents the hand the DJ uses to control the amplitude (amplitude controller).

To our ears—none of us being DJs—the result sounded quite convincing and the instrument was really addictive. The new approach to scratching accommodates users without DJ experience while also catering for the users with knowledge of scratching techniques.

⁹ <http://grrrr.org/ext/flex/>

¹⁰ <http://crca.ucsd.edu/jarlo/pd/>

3.2. The friction model

The resulting system represents a merge of the two models, and foremost offers novel ways to control the audio engine of the friction model by means of DJ gestures. Additionally, the systems allows for cross-mapping and interaction using either sound engine and various control devices.

Due to the lack of experience in usage and fine-tuning of the friction model, initial attempts at getting a realistic violin sound more or less failed. Due to the limited time span of the study visit, it was quickly decided that most of the efforts should be put in the technical establishment of an interface between the friction model and Skipp proof; whereas the model was set to produce the simplest sound (audibly close to a sinusoid). Thus, it was easier to determine the problems surrounding the programmatic aspect of the interfacing, whereas work on achieving greater (more realistic) audio quality from the friction model was left for a later stage. As such, at this stage of development, the developers aimed for simply hearing a scratching gesture effect, produced by the friction model, and dictated by the Skipp proof side of the interface.

3.3. Performance

For both projects, a refinement of the algorithms and implementation is necessary for a really usable result.

4. DISCUSSION

Even though the systems are fully functional, they need to be tested by actual DJs to verify the decisions we made during the development and to adjust both the new algorithmic scratch technique models and the control mappings. Once the system is finished, how to eventually incorporate it into the current reactable modular synthesizer paradigm and friction model model must be decided.

The method used in the development during the STSMs, was simply to try and assemble a working prototype—‘working’ in the sense that a user would be able to experience the different audio rendering engines and interaction interfaces, and be able to recognize when control signals were applied. The criterion for whether the prototype was ‘working’ was left solely at the judgement of the developers

For the further development of scratch models, the opportunity of studying how DJ gestures can be applied to novel conditions is very welcome.

5. ACKNOWLEDGEMENTS

The authors would like to thank everyone involved in the development of the systems and models. We greatly appreciate all the valuable help, comments and feedback from Stefania Serafin, Roberto Bresin and the three anonymous reviewers. The STSMs were made possible by ConGAS, European cost action 287.

6. REFERENCES

- [1] Marcos Alonso. Scientific Report from ConGAS Short Term Scientific Mission (STSM) to Stockholm. Technical report, ConGAS Cost action 287, <http://www.cost287.org/documentation/stsms>, October 2006.
- [2] Federico Avanzini, Davide Rocchesso, Alberto Belussi, Alessandro Dal Pal, and Agostino Dovier. Designing an urban scale auditory alert system. *IEEE Computer*, 37(8):73–79, 2004.
- [3] Richard Boulanger and Max Mathews. The 1997 Mathews Radio-Baton & Improvisation Modes. In *Proceedings of the International Computer Music Conference*, pages 395–398, Thessaloniki, Greece, 1997.
- [4] Smilen Dimitrov. Scientific Report from ConGAS Short Term Scientific Mission (STSM) to Stockholm. Technical report, ConGAS Cost action 287, <http://www.cost287.org/documentation/stsms>, March 2007.
- [5] Kjetil Falkenberg Hansen. The Basics of Scratching. *Journal of New Music Research*, 31(4):357–365, 2002.
- [6] Sergi Jordá, Günter Geiger, Marcos Alonso, and Martin Kaltenbrunner. The reactable: Exploring the synergy between live music performance and tabletop tangible interfaces. In *Proceedings of the first international conference on "Tangible and Embedded Interaction"*, Baton Rouge, Louisiana, 2007.
- [7] Sergi Jordá, Martin Kaltenbrunner, Günter Geiger, and Ross Bencina. The reacTable*. In *Proceedings of the International Computer Music Conference*, Barcelona, Spain, 2005.
- [8] Martin Kaltenbrunner, Sergi Jordá, Günter Geiger, and Marcos Alonso. The reactable*: A collaborative musical instrument. In *Proceedings of the Workshop on "Tangible Interaction in Collaborative Environments" (TICE), at the 15th International IEEE Workshops on Enabling Technologies*, 2006.
- [9] Stefania Serafin, Federico Avanzini, and Davide Rocchesso. Bowed string simulation using an elasto-plastic friction model. In *Proc. Stockholm Music Acoustics Conference*, pages 95–98, 2003.
- [10] Stefania Serafin and Diana Young. Bowed string physical model validation through use of a bow controller and examination of bow strokes. In *Proc. Stockholm Music Acoustics Conference*, 2003.
- [11] Stefania Serafin and Diana Young. Toward a generalized friction controller: from the bowed string to unusual musical instruments. In *Proceedings of the New Interfaces For Musical Expression Conference*, pages 108–111, 2004.