A Tangible Game as Distributable Platform for Psychophysical Examination

Matthias Rath[1] and Sascha Bienert[2]

[1] Technische Universität Berlin, Deutsche Telekom Laboratories,
Ernst-Reuter-Platz 7, 10587 Berlin
matthias.rath@tu-berlin.de
[2] Technische Universität Berlin, Institut für Sprache und Kommunikation, 10623 Berlin
sascha_bienert@gmx.de

Abstract. Through the use of built-in accelerometers a game-software for recent generation MacBooks allows control of a scenario of virtual moving objects by tilting the computer. Together with integrated visual and continuous auditory feedback from models based on physical principles the software forms a possible platform for online collection of psychophysical data.

1 Background and Motivation

The last decades have seen an increasing number of studies concerned with auditory perception of physical–ecological information (such as material [4][11] or size [1]). Most such works deal with discrete sets of information perceived in discrete auditory events, while the roll of *continuous* sonic feedback in as well continuous gestural human interaction with physical artefacts — despite its prominence in the "real" everyday world — has received less attention so far.

One example for the latter case is given by work conducted at an experimental tangible audio-visual interface, "*Ballancer*" [8], with central focus on continuous, seamless auditory perception of information of *velocity* of a virtual rolling ball. In short, in experiments at the *Ballancer* test subjects' performance in a target reaching task has improved under the presence of different types of continuous auditory feedback [9]. An important element of these studies and interface is a sound synthesis algorithm based on physical description of scenarios of contacting solid objects [6]. By using this sound model, apart from velocity information several other physical/ecological attributes may be expressed through sound feedback, such as the weight and diameter of the virtual ball, the roughness (or in general: structure) of the contacting surfaces, the resonance behavior of the supporting plane, which is mainly influenced by its material, and the hardness/elasticity of the rolling object (again a material property).

Tendentially such physical properties are most often and reliably perceived through the tactile channel (by touching the involved objects) but the sound emitted at contact such as rolling may also be an important cue (compare also e.g. [5]). In particular, the possiblity to support limited, or compensate for missing, tactile information through sound — in the line of thinking of *sensory substitution* — may be attractive for practical applications where possibilities of providing tactile feedback are restricted

M.E. Altinsoy, U. Jekosch, and S. Brewster (Eds.): HAID 2009, LNCS 5763, pp. 155–164, 2009.

(e.g. for reasons of technology or display space). It is therefore one current motivation of the authors to exploit the metaphor and techniques behind the *Ballancer* — balanced rolling objects with continuous reactive auditory sound feedback — for examinations of auditory perception of object and surface properties.

Another central idea behind the implementation described in this contribution consists in gaining exeprimental data by means of a freely distributed game/demo-application software: the authors' works at the previous *Ballancer* interface showed that acquisition of human control movement data of sufficient amount to allow for statistically significant conclusions is a tedious task when relying on a laboratory environment. The wide distribution of sensor hardware (such accelerometers) in many devices today might here offer an attractive alternative. In the following we describe the main elements of a game/demo-application-like software which will form the basis for future perceptual studies. A first version of the application may be downloaded by now. It is the authors' intention to discuss the possible potential of such an easily distributable experimental game-like platform, share ideas and facilitate possibilties of collaboration.

2 Interface, Metaphor and Program Structure

As described, the intention behind the software presented in this contribution is to supply an easily distributable platform for psychophysical experiments, however appearing with a possibly "game-like" character. A general challenge was therefore to combine tangible access, a widespread underlying hardware platform, reactive continuous auditory feedback, precise synchronization of in- and output in different sensory channels, computational costs, and in particular the possibly conflicting aspects of experimental focus vs. fun of use.

For gestural user input with possibly widespread tangible devices a convenient decision is to rely on the use of accelerometers, as such sensors are today found in many portable devices such as mobile phones (e.g. Apple iPhone, Nokia N95), the Nintendo Wii game controllers and notebook computers. The presence of accelerometers in portable computers generally serves the aim to protect harddrives rather than purposes of gestural input so that no device-independant APIs for their use as input mode are available. For the last generation Apple *MacBooks* however, data from the built-in 3-d accelerometers may be accessed in a relatively convenient way, across the different models of these series (e.g. by using the open source "Unimotion" library). This was one reason for us to choose this platform for our first implementation, along with the facts of up-to-date computing power and graphical display facilities as well as a good quality sound output and relatively low-latency (as compared to much of other standard PC hardware) driver environment. Also, the implementation under Mac OS X should facilitate porting of the code to the iPhone platform, as far as computational power of this device will be sufficient for our realtime physics-based sound generation algorithms. (Work in this direction has recently been launched.)

2.1 Game Concept

As to our game, continuous gestural control interaction and sound feedback is the main focus. The accelerometer gives us the opportunity to determine the inclination of the

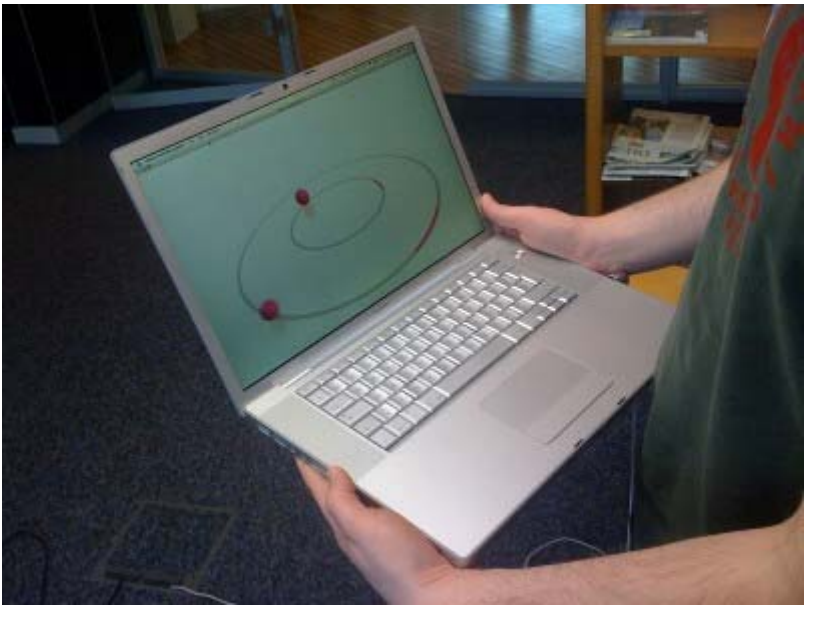

Fig. 1. The *orbits* game being played on a *MacBook*

MacBook. In this manner it can be used as an input device for steering tasks in human-machine interaction. One apparent scenario is to route an object on a plane or on a track. Indeed, there are quite a few games where the user has to balance an object (mostly a ball) in a maze-like virtual world[1].

As already mentioned we want to establish a platform that facilitates psychomechanical tests. In particular our application is geared towards sound feedback and gestural control. We thus chose the scenario of objects tied to circular tracks moving under the effects of gravity and friction. Tilting the tangible device causes a period of oscillation around the equilibrium, until due to friction the objects eventually come to rest. Each object can be steered to any place on the circle by performing adequate balancing movements. The player can also make the object rotate in a circular manner on the track.

Thus sound feedback is the most important feature of our application and explorations of the influence of diverse types of acoustic feedback on user experience and human performance in control tasks is one central interest. The sound feedback is computed based on physical considerations and put out as continuous realtime feedback of the objects movement. In the current configuration we provide a number of differing audio feedback basing upon two different algorithms: a *rolling sound model* and a *sliding sound model*. The sound feedback can be varied depending on parameters like size or mass of the object and resonance behaviour or surface profile of the underground.

In our first realization the scenario consists of two separate tracks with two independently moving objects. Both circular tracks contain target areas, but the challenge does

[1] For example common marble games like “Labyrinth” (iPhone), “Super Monkey Ball” (iPhone, Wii), “Kororinpa” (Wii).

Fig. 2. Screenshots of two visual elements of the *orbits* game

not consist in finding them since they are clearly visible and finding a path to reach these areas is trivial; please compare figure 2.

Using two tracks not only supplies possibilities to create different tasks to accomplish for the user (as will be described below) but also enables us to present two different types of continuous auditory feedback at the same time.

For the visual appearance of the moving objects in our game we chose as first elements the very simple cases shown in figure 2: a ball which is rolling on a circle and a cylinder which is sliding on a circular rail. The object that we mainly use is the ball (primarily because of our focus on rolling sound). Some other sounds (e.g. the sliding sound) may seem unsuitable for the sonification of a ball's movement, a supposition that can be explored by means of the application. In this regard the sliding cylinders can give an impression of the interdependency of auditive and visual perception (for example by altering the objects but not the sound model or vice versa). Besides their function to visually maintain (or contradict) auditory feedback the cylinders are also useful to underline a stronger degree of friction.

As game logic and complexity was not our focus, we kept the game as simple as possible in this regards, disclaiming usual methods of making games challenging: time limits, an increasing number of tasks and goals to achieve... The principle task always remains the same (under varying conditions): the player has to steer both objects (e.g. two balls) into their target areas by tilting the MacBook. The task is fulfilled once both balls are in their target areas at the same time. When the task is fulfilled the target areas change their position and the game continues.

There are different strategies to reach the goal. Besides carefully balancing both balls one obvious strategy is to let them roll more or less randomly in circles until they by chance are in their target areas at the same time. By decreasing the size of the target areas gradually (e.g. every time a task is fulfilled) the second strategy becomes less effective and attentive careful balancing is necessary. On the other hand letting the balls circulate at the track demands a movement pattern which is interesting as well. Periodically tilting the MacBook in a rotatory manner and thus sustaining the velocity of the ball is a steering movement that clearly differs from a balancing task. It is here important to mention that objects react in different ways to inclinations due to different friction parameters whereby, with some skill, it is possible to let only one ball circulate and keep the other one fixed (more or less) at the same place. In contrast to balancing the ball, letting it rotate is a dynamical, periodical process. The player may therefore rely more on the dynamical information that he perceives aurally than on what can be seen at the screen. In this respect the "resonance-like" behaviour of the balls movement

at periodic circular acceleration plays a decisive role. Of course the player can visually perceive in which way the ball reacts to a rotary tilting movement, but such kind of continuous, dynamical information is one strength of the auditive perceptual channel (compare e.g. [9]).

We finally thought about a way to force the player to perform balancing and "rotation control" simultaneously and designed the following scenario as part of the game: the size of the target area of one of two moving objects (on parallel circular tracks) is determined by the velocity of the second object. While the position of the target is constant its size increases with the velocity of the object moving on the parallel track. Under these conditions, in order to fulfill the task the ball that affects its neighbour's target size *must* move because a velocity of zero corresponds to a target size of zero. Letting said object rotate is now not the only way to reach the task but clearly (depending of course on the exact parameters of the coupling) the most obvious and the player is more or less forced to apply the following strategy:

- The faster the circulating ball moves the bigger becomes the target area of the other ball.
- If the ball with the varying target size reached it's goal the player has to keep it there.
- He would keep it there just for a moment, because the circulating ball frequently passes through it's goal.

In this scenario a rotary inclination of the device is a typical gesture the player performs to play the game — a circumstance that certainly supports our current intention (and work) of porting the application to a smaller, truly handheld device.

Altogether we have so far realized a number of instruments that may be combined: a set of sound models, two visually different objects, friction parameters to make them also "feel" different and targets that can vary in position and size (also velocity-dependent target-size). Of course the point is not to arbitrarily combine those building blocks which are all related to each other. A higher setting of the friction parameter will rather suit a sliding sound and the cylinder visualisation, just as the ball visualisation naturally fits the rolling sound. Under the aspect of creating a preferably convincing game one would try to set a combination that is as realistic as possible whereas for psychoacoustical tests it is also interesting to chose unintuitive (or even contradicting) combinations.

3 Technical Realisation / Example

We designed the game as a standalone application that should run on all laptops of the MacBook/MacBookPro series. The native application programming interface (API) for dealing with sound in Apple's Mac OS X operating system is Core Audio [3]. The audio processing of our program is implemented as a callback architecture with Core Audio[2], which means that Core Audio instructs our program to render audio output and write it into a buffer that can be referenced by an assigned memory address. It is important

[2] Here we use Core Audio's "Audio Unit Framework", a plug-in architecture that can be used for effects and virtual instruments.

to understand that we are not telling Core Audio to play back precasted audio samples, instead of that we are continuously computing frames of audio output ourselves and write them into an audio buffer.

Graphical rendering is done with OpenGL, a cross-platform API for computer graphics. For scheduling of in- and graphical output cross-platform free library "GLUT" is used. GLUT also supports callback driven event processing, which means that we also have a callback function for video processing (besides the function for audio).

3.1 Scheduling

When the application is started it registeres itself as a client for audio (Core Audio) and video processing (GLUT) and detects the motion sensor. All physicial simulation is computed with audio rate.

Regarding the physics there are basically two tasks to perform: calculate the movement of the object along the track and synthesise the sound. The rolling sound model is predicated on physical considerations of the interaction between ball and surface. Besides the macroscopic movement of the ball that can be seen there is a microscopic behaviour that can only be heard, both must be computed. One approach (actually the most common, in combination with sample-based audio) would be to use a physics engine for the calculation of the ball's macroscopic movement with a defined temporal resolution and synthesise audio feedback on the basis of the result. In this case the quality of the feedback would strongly depend on the update rate of the used physics engine. The opposite approach would be to unify macroscopic and microscopic movement in one model to obtain both as the output of the same algorithm, which would make movement and sound much more coherent but on the other hand is also much more difficult to achieve. In our simulation we chose a method which is a compromise of these two solutions. We make use of two separate models but both are always calculated in parallel within the same cycle of the audio processing, which means that we update the position of the ball with audio rate and instantaneously calculate the next sample frame of audio feedback in dependence of the position and velocity parallel to the track.

Structure and scheduling are as follows: all physical objects and algorithms of the current active simulation are contained in a data structure that we call "scene". In fact we have a number of different scenes with different objects, sounds, etc., but only one scene is active at a time (the others are stored in memory, ready to be activated). Only the currently active scene is updated, which means that the physical behaviour of the objects and the sound output is computed (with audio rate). These updates are performed within the audio callback function. The video callback function induces the rendering of the scene by accessing relevant data computed in the audio loop.

The buffer size of our audio callback routine is fixed at 1024 samples, which means that the audio callback function is called with about $43Hz$ (at an audio rate of $44100Hz$). Each time the audio callback function is executed the accelerometer data is read *once*, *1024* updates of the scene are performed and the buffer is filled with samples. We have no direct control about the exact moment when the audio callback function is called, we only know that it happens in average 43 times per second. By relying on the callback cycle of the audio driver a certain jitter is unavoidable which we however accept since

an average update rate of $43Hz$ is quite high for accelerometer data used for balancing control.

As already mentioned the video rendering is also executed by means of a callback function, for which we chose a fixed rate of $50Hz$. Since values relevant for graphical output (object position...) are computed in the audio callback cycle the same remarks apply here as for updates of accelerometer data.

3.2 Sound Feedback

In our game sound feedback is not just a fancy supplement but one of our central interests. We seek to explore the effects of the sound on the player. Does it enhance his performance? Does it affect his attitude, his perception of the game? Recent experiments with the *Ballancer* interface [8] showed that the perception of velocity plays a key role in this context, it can improve a player's performance in a control task [9].

As already mentioned our game is based on steering objects on plains affected by gravity. Hence the scenario of a rolling ball is the starting point of our considerations about *sound models* for the game, about their qualities and their implementation. The "rolling sound model" considers the profile and the vibration of the surface as well as the vertical interaction of the surface and the rolling ball. It is our informal supposition that the detailed nature of this vertical interaction plays a strong role in the auditory perception of rolling: it appears that an important factor for sounds to be classified by human listeners as "rolling" is the mixture of short periods of microscopic bouncing and longer periods of contact between the rolling object and the surface. The "sliding sound model" only considers the vibration of the surface and the sweeping of the object across the surface profile. In accordance with our informal supposition on the characteristics of contact sound the audible result resembles the sound of a sliding object.

In our scenario two objects are interacting with each other: the ball (resp. the cylinder) and the surface. There are different approaches to describe the behaviour of vibrating and interacting objects; we chose the modal approach (compare [7]) for a model of the inner resonance behaviour of the vibrating surface. We did that on the assumption that the surface's vibration plays the dominant part in the sound that we hear when an object is rolling or sliding across it while the vibration of the solid rolling object itself (e.g. a marble) can hardly be heard directly. The modal description of the surface (in the following "modal object") is the source where our sound output comes from: the velocity of the surface is written into the audio buffer.

If an impulsive force is applied to the modal object is starts vibrating and becomes silent again after some time due to inner friction. As the ball is bouncing it applies a force to the surface. A rolling ball or a sliding object is continuously impacting the surface because natural surfaces are not perfectly smooth. There are different approaches to model that contact. Regarding the profile as a (force) input signal to the modal object is one way, discussed in the next subsection (sliding sound). A more accurate approach is to include physical and geometrical considerations: the rolling sound model.

3.2.1 Sliding Sound

This sound model is based upon assumptions about the impact force that the sliding object applies to the surface: the impact force has a noise-like character and is bandlimited

in the frequency domain. In other words an incompressible object perfectly strobes the surface's noise-like outline. Vertical interaction between object and surface is not considered, as well as the geometry of the object (e.g. the ball's shape).

Hence we utilise the profile of the surface as (force) input signal to the modal object (in a way similar to some previously used algorithms, compare e.g. [10]). We use bandpass filtered white noise as signal. The centre frequency of the filter is proportional to the (horizontal) velocity of the ball, the bandwidth is kept constant; the sound model can be regarded as a modal resonator.

3.2.2 Rolling Sound

The rolling sound model affords a high degree of realism, it is the natural acoustic complement of what can be seen on the screen when the balls are rolling at their tracks. The rolling sound model embraces the vibrations of the surface, the surface's profile and the vertical contact interaction of the vibrating surface and the ball. Details of the algorithm are described in dedicated articles [6] [7], in the following we try to give an idea of the main points.

The basic principle of the rolling sound has already been introduced in [6] and implemented as a patch for Pure Data[3]. Recently an improved contact model has been presented [7]. In the present work we implement the improved algorithm of [6] based upon the contact model of [7] implemented in C++. In the following we give a short summary about the ideas and methods.

Despite some interesting psychoacoustic studies (e.g. [2]) the question of how to exactly describe the acoustic features responsible for a sound event to be perceived as "rolling" is still not perfectly answered. This observation supports a physics-based approach in the synthesis of rolling sounds. We look at "rolling" as a process of sustained bouncing, eventually microscopic bouncing. The second aspect is that we allow the ball to penetrate the surface, thus a contact is not a change of state at a discrete moment in time. Contacts are continuing processes that have to be modelled. To sum up, the interaction can be both continuous and sporadic: there are moments of contact, in particular continuous contact, and moments when surface and ball are not in contact.

The modelling of the contact is one key building block of the rolling sound model. [7] introduces a method of energy-stable modelling of continuous or repeated contact in dynamical situations, a method that is also applied here. The basic idea is to use a modal description for the period of contact too. If there is no contact the state of the surface is described by a modal object, the ball's state is stored separately. In case of contact surface and ball are regarded as one object, which is described by means of modal parameters as well. The model of rolling includes frequent "switching" between both configurations.

Besides physical considerations (improving the contact model) there are also geometric aspects to consider. Although we constrain that the contact only takes place at one contact point the ball of course has a geometric attribute that should not be disregarded: its radius. We simplify the profile of the surface to a white noise distribution of profile points (with constant horizontal distance, but scattered vertical position). One

[3] A graphical programming language for the creation of interactive computer music and multimedia works.

can imagine that the ball cannot touch every point since it is too big to fill each gap. Therefore the profile needs to be filtered. Instead of bandpass filtering (as in the next section) we have to filter the "profile signal" based on geometrical considerations. The exact procedure is explained in [6] and for now called "rolling filter". To sum up, the result is a profile signal consisting of arcs of circles which is the path that a ball would follow if it was perfectly succeeding the surface's profile.

In the sliding sound model the profile signal is used directly as input force for the modal object. Since the rolling sound model contains the modelling of contacts (and resulting forces) based on physical considerations the profile is regarded as a position-dependent (and thus timevariant) input parameter to the contact model. The contact model described above involves a continuous computation of the distance between ball and surface. At this point the profile of the surface is incorporated as a vertical distance offset. Besides several "white noise variants" as profiles for surfaces the rolling filter technique also enables more complex profiles like saw tooth surfaces[4] that enable interesting possibilities to imitate natural surfaces.

4 Summary and Future Work

We have described a recently implemented game on the MacBook platform intended for use as easily distributable tool for collection of psychophysical data. The software uses the MacBook's built-in accelerometer for gestural control of a balancing scenario. It offers precise synchronous visual and auditory feedback based on models of different degrees of complexity of physical contact. A first demo-version of the game platform — fully functional in its main components — is ready for free download and installation on recent generation MacBooks.

Of course this contribution is to be understood as a report on the development and potential of the presented software as basis for future online data acquisition. Concrete experiments at the platform are currently being planned. To this end, mechanisms of automated online data collection are currently being incorporated into a next version of the software and concrete experimental game concepts are being developed. Also, the plan of porting the environment to the iPhone, a platform that might further enhance possibilities of distribution and user interaction is being explored more closely.

References

1. Carello, C., Anderson, K.L., Kunkler-Peck, A.J.: Perception of object length by sound. Psychological Science 9(3), 211–214 (1998)
2. Houben, M., Kohlrausch, A., Hermes, D.: Auditory cues determining the perception of size and speed of rolling balls. In: ICAD 2001, Espoo, Finland, pp. 105–110 (2001)
3. Apple Inc. (2009), `http://developer.apple.com/documentation/musicaudio`
4. Klatzky, R.L., Pai, D.K., Krotkov, E.P.: Perception of material from contact sounds. Presence: Teleoperators and Virtual Environment 9(4), 399–410 (2000)
5. Lederman, S.J.: Auditory texture perception. Perception 8, 93–103 (1979)

[4] Combined with above-mentioned white noise "micro structure"...

6. Rath, M.: An expressive real-time sound model of rolling. In: Proceedings of the 6th International Conference on Digital Audio Effects(DAFx 2003), London, United Kingdom (September 2003)
7. Rath, M.: Energy-stable modelling of contacting modal objects with piece-wise linear interaction force. In: Proceedings of the 11th International Conference on Digital Audio Effects (DAFx 2008), Espoo, Finland (September 2008)
8. Rath, M., Rocchesso, D.: Informative sonic feedback for continuous human–machine interaction — controlling a sound model of a rolling ball. IEEE Multimedia Special on Interactive Sonification 12(2), 60–69 (2005)
9. Rath, M., Schleicher, R.: On the relevance of auditory feedback for quality of control in a balancing task. Acta Acustica United With Acustica 94(1), 12–20 (2008)
10. van den Doel, K., Kry, P.G., Pai, D.K.: Foleyautomatic: Physically-based sound effects for interactive simulation and animation. In: Proc. ACM Siggraph 2001, Los Angeles (August 2001)
11. Wildes, R.P., Richards, W.A.: Recovering material properties from sound. Natural Computation, 356–363 (1988)