Soundsense
a haptic sound generation instrument

Oana-Iuliana Popescu
Bauhaus-Universität Weimar
oana-iuliana.popescu@uni-weimar.de

Moritz Loos
Bauhaus-Universität Weimar
moritz.loos@uni-weimar.de

Supervisor: Dr. Günther Schatter
Bauhaus-Univesität Weimar
guenther.schatter@uni-weimar.de

ABSTRACT
Playing a music instrument has a large number of benefits such as increasing creativity and concentration skills, as well as improving memory skills. Whereas learning to play an instrument, especially a classical one, takes years of practice, digital instruments take less time and might offer the same results. Even though the musician and the physical instrument create a relationship over time, the same can be achieved with the help of digital instruments, by enabling the user to interact with it as it would with a real instrument.
Soundsense is a new electronic music instrument, designed for children, as well as adults. The goal of this haptic instrument is to enable the creation of electronic music pieces. Its use gives the same feeling as a real musical instrument, but with reduced learning time needed. Soundsense makes possible the generation of real time sound with the help of an inertia measurement unit attached to a Raspberry Pi device. By capturing quick movements, similar to using drum sticks, and recognizing their position on the X and Y axis, different musical tones are generated.

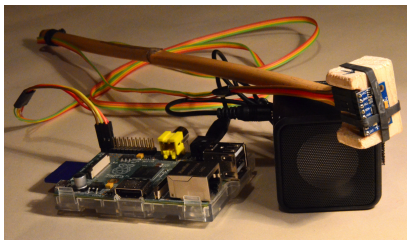

Figure 1: The stick of the Soundsense system.

Categories and Subject Descriptors
H.5.2 [**Information Interfaces and Presentation**]: Auditory (non-speech) feedback, Haptic I/O, Interaction styles

Keywords
haptic; Raspberry Pi; sound generation

1. INTRODUCTION
Soundsense is an electronic music instrument with a shape that resembles a drum stick. The generation of the sound is done with the help of a sensor which tracks the movement of the stick in air. The instrument is capable of recognizing slow and fast movements and map them to sounds which are then played in real time.
The motivation behind this instrument is giving users the possibility of playing a digital instrument and still interact with it as if it were a physical music instrument, but with a lower complexity. Therefore, children who are not old enough for learning to play a classical instrument, as well as adults who do not have time to invest in learning a real music instrument, but wish to be artistic, have the possibility to play this instrument and fulfill their creative needs.
Digital music instruments have been previously created, the "Radio Baton" and "The Digital Baton" being two of the most relevant examples. Whereas these instruments are especially designed for performers, which already have musical experience, we designed our instrument for users with no or little previous knowledge of music or music theory.
The initial design concept of this instrument was the creation of a percussion instrument that imitates a real drum kit. This would recreate, to a certain extent, the feel of a real drum stick. Our main focus was creating an instrument that entertains and brings joy and relaxation when used.
Due to technical limitations, the current implementation of the Soundsense device differs from the initial design. The final prototype is an electronic music instrument which plays tones depending on the position of the stick in air and the speed of the movement. We present in the following chapters an accurate description of the initial concept, current state and the imposed limitations that led us to the state in which the instrument currently is.

2. RELATED WORK
2.1 Léon Theremin
The theremin
The era of electronic music instruments started with the creation of the theremin in 1920 by Léon Theremin [16]. The instrument consists of two antennas which need to be controlled by a performer. The relative position of the hands is recognized with the help of two metal antennas. With the help of hand movements, the performer controls fre-

quency with one hand and amplitude with the other. The signals from the theremin are then send to a loudspeaker and played.
This instrument paved the way for newer electronic music instruments and for computer music.

The history of computer performance of music started approximately 60 years ago, with Max Mathews as a pioneer of this technological breakthrough. Computer music was very appealing because of the broad possibilities and lack of physical limitations which physical musical instruments have. This led to further research and creation of works that put the basis of sound synthesis, digital instruments and the development of this technology.
After creating the first computer music piece in 1957, called "Hal's Tune" [6], Max Mathews created in 1987 the "Radio baton", a musical instrument for controlling certain aspects of musical pieces such as tempo in real time[14].

2.2 Max Mathews "Radio Baton"

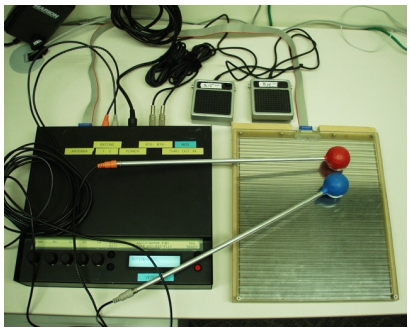

Figure 2: Radio Baton

The "Radio Baton" (shown in figure 2) is a controller which enables the users to control musical pieces being played with the help of the "Conductor Program".
The musical instrument works as a MIDI controller, which controls the sound production. This instrument is composed of two drumsticks and a drum surface on which these drumsticks can be moved. The two drumsticks emit slightly different signals. The signal is then intercepted by the surface and then further processed into X, Y and Z coordinates. With the help of this tool, the sound can be modified according to the preference of the artist using it. [15] This tool proved to be attractive for experimental music artists, but also for soloists who found the possibility of modifying the accompaniment according to their needs and wishes useful. [3]

2.3 Keane and Gross "The MIDI Baton"

In 1989, David Keane and Peter Gross revealed their work on the "MIDI Baton" [8]. This tool is designed as a MIDI controller to be used by conductors to coordinate live performers and alter MIDI-compatible systems. The goal of this instrument was allowing the conductor to use the same gestures used for conducting live performers, but at the same time altering the output sound.
The system is composed of the baton itself, which has to be connected to a live synchronization device, and a foot switch. The baton is started (or restarted) by a single foot pedal. The baton contains an electrical sensor which sends signals when the direction is altered. This mechanism consists of a brass tube with a metal weight suspended longitudinally. When the tube is moved, the weight is moved with it and triggers an electrical contact when it hits the wall of the tube. The signal is processed after to remove noise caused by oscillations of the weight and then sent to the analogue input of the synchronisation unit, which then reads the pulse and adjusts the tempo accordingly.

2.4 Marrin and Paradiso "The Digital Baton"

"The Digital Baton" [1] is one of the most relevant works for us. Built in 1996, this complex system in the form of a baton is used for creating electronic music.[2]

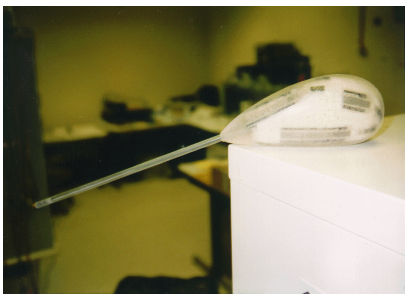

Figure 3: The Digital Baton. [1]

With the help of this device, the composer can control not only the overall sound, but also the details of particular sounds. This is done with the help of multiple sensors built in the device, each triggering different actions: mapping sounds of classical instruments to the tracked data, changing the tonality or adjusting speed of the sequence.
An infrared LED is positioned at the tip of the baton and tracked by a camera which only recognizes the 20 kHz signal of the LED (in order to filter other lights). The camera contains a photodiode which then processes the signal and transforms it into X and Y coordinates. At the base of the baton, three orthogonal accelerometers measure the acceleration. Furthermore, the baton contains a pressure sensor at the base, such that hand and finger pressure is also recognized.
This digital musical instrument allows performers to compose complex musical pieces by just moving the baton in the air.

3. DESIGN CONCEPT

To provide the affordance of a physical instrument, we designed our tool in the shape of a drum stick. Our wish was to create an intuitive tool which would be quickly understood by possible users. Reducing the learning time would reduce frustration among the users. Since one of our goals was also presenting a tool that would entertain and create joy among the users, reducing frustration was necessary. To appeal to

the intuition of the users, the shape resembles a drum stick because it is activated by beating it in the air.

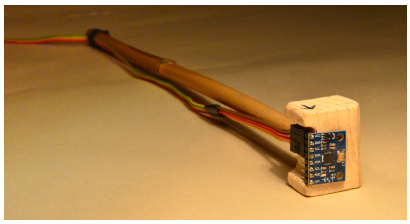

Figure 4: The Soundsense system.

In the initial concept, the system was designed so that the user would generate drum sounds by using the stick. The user would have two regions in air, one standing for the drum and one for the cymbal of the drum kit. Depending on the angle at which the user would beat the stick, the position in the defined region and the intensity of the beat, the user would generate a sound corresponding to drumming with a certain intensity a certain region of the drum kit. The following figure presents these virtual regions and their mapping to the drum kit surface.

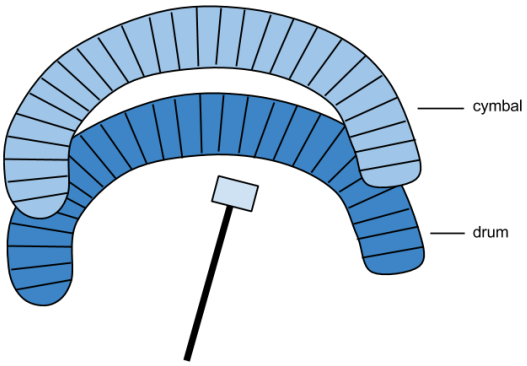

Figure 5: Initial design with two different beat areas and their mapping to the drum kit.

Therefore, the user could learn how to use the instrument quickly, just by experimenting. The tool would be easy to use because only one part of the drum kit would be activated, making it impossible to play more than one sound with one stick at a time. Users with a higher interest in the tool could learn to use two sticks at a time and actually use the instrument the same way as they would use a drum kit. Unfortunately, technical and time limitations did not allow us to implement our initial design concept. We have further tailored the design to the possibilities, but keeping in mind the needs of our potential user groups: children and adults with little or no musical knowledge.
The design went through two iterations until its current state. First, we left out one of the regions and restricted ourselves to only one. Moreover, we decided the sound would be an electronic music sound instead of drum sounds.
In the first iteration, we mapped the rotation on the X axis to the frequency values of the sound and the rotation on the Y axis to the amplitude of the sound. On the X axis, the frequency increased from left to right, with the entire region representing one octave. The amplitude increased with the vertical position; moving upwards increased the amplitude. The sound consisted of a single damped wave that lasted 0.5 seconds. Nevertheless, the first iteration proved to be problematic because the sensor data, even though filtered, did not give reliable results. Therefore, the generated sound did not enable the user to actually create a musical piece. In a sense, our first iteration was a theremin which recognized the position of the instrument itself instead of being activated by hand gestures.

To solve the problem of unreliable sensor data, we modified the concept further. The current implementation of our prototype reminds of a virtual xylophone. To compensate the wrong sensor data we separated the virtual region from left to the right in 6 parts, each standing for a different tone. Every subregion is associated with the sound of the hexachordon (C D E F G A), as depicted in Figure 6. Depending on where the user hits the virtual surface, the associated tone is played. From our point of view, this approach makes it easier to understand and play this instrument. The tone length is mapped to the speed of the motion: a really slow motion implies a longer sound, of 2 seconds, while a fast motion implies a shorter tone, of 0.8 seconds.

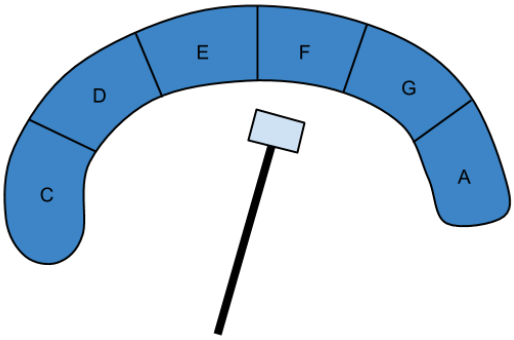

Figure 6: The different sound regions.

4. IMPLEMENTATION

The entire system structure and connection between hardware and software is depicted in Figure 7. The following subsections of this chapter describe in detail the implementation of the first, but also the second (current) iteration.

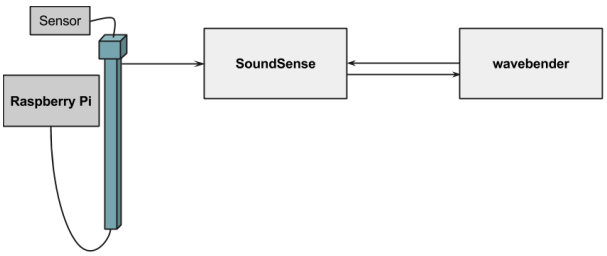

Figure 7: The system structure.

4.1 Hardware

The hardware in Soundsense consists of an inertia measurement unit, as pictured in Figure 8, that is connected to a Raspberry Pi. The inertial measurement unit (IMU) is a commonly used chip that combines a 3-axis gyroscope and a 3-axis accelerometer and uses a standard I2C bus for data transmission[10]. In total, the MPU6050 sensor has 6 degrees of freedom (DOF).

This sensor is attached to a Raspberry Pi device model B

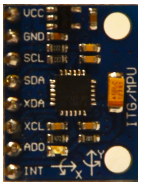

Figure 8: The sensor used: MPU6050.

of the first generation. The operating system installed on the device is Raspbian, date 05-05-2015.

4.2 Software

The program needed for reading the sensor data and processing the data is written in Python, version 2.7. and is called "SoundSense". The sound synthesis is made using the open source Python framework "wavebender" created by Zach Denton.

4.3 Data extraction and filtering

Our MPU6050 sensor builds the heart of our system because it provides the relevant data needed for sound synthesis.

To access the raw data from the sensor, the sensor is connected over the GPIO pins to the Raspberry Pi device. The raw values are read from the I2C device. Every functionality (gyroscope and accelerometer) of the sensor has a number of registers that can be accessed by a special number and hold the raw data in 16-bit Two's complement format [5].

Once the raw data is read, it has to be rescaled and converted to fit our needs.

According to the datasheet of the MPU6050 [12], the raw data of the accelerometer has to be divided by 16384 to get a multiple of the earth's gravity ($g = 9.8\frac{m}{s}$), while the gyroscope raw data has to be divided by 131 to get the angular velocity in seconds [10].

It is important to know that the provided component of gravity of the accelerometer that acts on the X axis is a sine wave while that one acting on the Y axis is a cosine. Therefore, the tilt angle around the X axis (α_{accel}) and around the Y axis (β_{accel}) of the accelerometer can be computed using all three outputs, as described in the following formulas [9]:

$$\alpha_{accel} = \arctan \frac{X}{\sqrt{Y^2 + Z^2}} \tag{1}$$

$$\beta_{accel} = \arctan \frac{Y}{\sqrt{X^2 + Z^2}} \tag{2}$$

The angular computation for the values measured by the gyroscope is different, since the measured data for the gyroscope is the alteration rate in orientation angle. Therefore, the angular velocity has to be converted to absolute angles. In order to compute the current orientation, the sensor is initialized with a known position (e.g. from the accelerometer). Afterwards, the angular velocity (ω) around all three axes at specific measuring intervals (Δt) is measured (sample rate). The result of the multiplication of both values results the change in angles (Formula 3).

$$\Delta A_{gyro} = \omega \cdot \Delta t \tag{3}$$

The final orientation angle is composed of the original angle and the sum of the changes.

$$A_{gyro} = last\ A_{gyro} + \Delta A_{gyro} \tag{4}$$

Unfortunately both the accelerometer and gyroscope module have disadvantages. The accelerometer provides accurate data only if gravity is the only force acting on the sensor. Due to the user moving and rotating the device, other forces are applied to the sensor as well, resulting in noise on the short term. Nevertheless, the sensor is still accurate on the long term.

The opposite is the case for the gyroscope. In order to find orientation, small computed intervals are integrated. Repeatedly adding up increments of ΔA_{gyro} results in small systematic errors being magnified over time. This the cause of gyroscopic drift and the reason why the gyroscope data is inaccurate over a long timescale.

To combine the advantages of both accelerometer and gyroscope, the complementary filter is used.

4.3.1 Complementary Filter

The complementary filter was chosen because it is computationally very cheap. In order to avoid latency, our goal was to avoid expensive filters which could cause slower processing of the data and lead to a lower response time. The complementary filter is a combination of low and high pass filters and is described by following figure and corresponding formula (5).

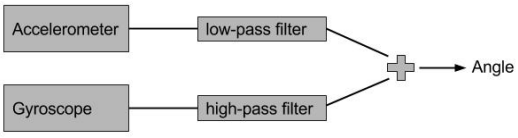

Figure 9: The complementary filter.

$$A_{filtered} = \alpha * A_{gyro} + (1 - \alpha) * A_{accel} \tag{5}$$

$$with \quad \alpha = \frac{\tau}{\tau + \Delta t}$$

The role of the low-pass filter is to filter the short-term fluctuations of the accelerometer. This is achieved through forcing the changes to build up step by step in subsequent times through the program loop (see $(1 - \alpha) * A_{accel}$). The

high-pass filter achieves the opposite and allows only short-duration signals from the gyroscope to pass through (see $\alpha * A_{gyro}$).
The time constant τ is the relative duration of the signal on which the filter will act. For the low-pass filter all signals shorter than the time constant are filtered out, while the rest will pass through. The opposite is true for high-pass filter [7]. Therefore the time constant is set to a value greater than the timescale of typical accelerometer noise. We used a time constant of 1 and have the sample rate of the Raspberry Pi Δt of about 0.03 seconds, giving a value of $\alpha = 0.97$.

The following graph (figure 10) shows the rotation on the Y axis of the accelerometer (coded with the colour red), the gyroscope (coded with the colour green) and our filtered angle (coded with the colour blue). First a 90 degree rotation in the positive direction is done, followed by a 90 degree rotation in the negative direction, which results in returning to the initial position. This clarifies the problems described above. The accelerometer is noisy on the short-term but returns to the initial position, while the gyroscope is accurate over a short timescale but drifts over a longer time.

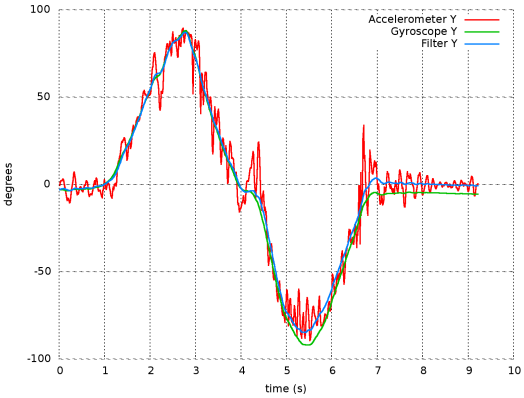

Figure 10: Comparison graph: accelerometer is coded red, gyroscope coded green and complementary filter coded blue.

4.4 Sound Generation

The "wavebender" library provides us with the possibility of creating different types of waves, such as sine, square or damped waves or composing multiple waves to create sound. The data of the sensor is mapped to different values of the frequency and amplitude of created waves. In the current state of the instrument, multiple damped waves are composed in order to generate a short sound for every beat of the stick. One reason for using this type of waves instead of the sound designed initially is that the synthesis of the drum sounds proved to be very difficult. One reason behind it could be the limitation that the "wavebender" framework imposes. Another reason could be the difficulty of the drum sound itself and recreating the differences that come from hitting different areas of the surface of the drum or cymbal. Unfortunately, our knowledge in sound synthesis also restricted us from enhancing the "wavebender" framework for the creation of the sound originally designed. The implementation of both iterations are explained in detail in the following subsections.

4.4.1 Transformation of Sensor Data

In the first implementation of the prototype, the filtered sensor data was mapped to values for the amplitude and frequency of sine waves. The data from the sensor was received as degrees: on the X axis, -90 degrees represent left and +90 degrees represent right; on the Y axis, -90 degrees represent downwards, while +90 degrees represent upwards. The data on the X axis (from left to right) is mapped to frequency values from 400 to 800 Hz (one octave): maximum left stands for the lowest frequency, 400 Hz, while maximum right stands for the highest frequency, 800 Hz. The amplitude values are given by the rotation on the Y axis (upwards or downwards): maximum downwards stands for the lowest amplitude, of 0.01, while maximum upwards stands for the highest amplitude, of 0.1.

Nevertheless, the accuracy of the data, even after applying the filter, was lower than expected. Because of the fast hit movements and the other forces applied to the sensor besides gravity, the data was not accurate enough for reproducing the same sound twice. Therefore, it would often happen that hitting more than once in the same region in air would result in different sounds.

In approximately 30 percent of the cases the correct sound would be played. For example, hitting right up would give a loud tone with a high frequency. In 70 percent of the cases, the result would differ (for example, a soft tone with a high frequency).

To increase the chances of success, six predefined tones were implemented in the second iteration. The tones can be activated by hitting in different subregions of the hit area, as Figure 6 represents. The reduction of the number of sounds increases from theoretically infinity to only 6. To further increase the system stability, mapping on the X axis of the amplitude was removed.

The precomputed tones correspond to a hexachordon (C D E F G A). The low frequency tone "C" is played while hitting the leftmost subregion and the high frequency "A" is played while hitting the rightmost subregion. The other tones are distributed between these two virtual subareas. The lowest frequency used is the one of the tone C, with a value of 180 Hz. The highest frequency, the one of the tone A, is 330 Hz. The amplitude for all tones of the hexachordon are set to 1.

4.4.2 Sound Synthesis

The reason behind choosing damped waves for our prototype is the fact that the amplitude of the oscillation of these waves decreases with time until null. Therefore, the sound slowly fades instead of abruptly ending. This creates the illusion that the sounds melt into each other, creating a musical piece that is more pleasing to the ear.

In the first iteration, the final sound was composed of seven

waves, each one having different amplitudes and frequencies, calculated depending on the frequency and amplitude value mapped from the movement position. Each generated sound had a length of 0.4 seconds and was sampled at 44.1 kHz.

In the second iteration, each sound (C D E F G A) is composed of only one damped wave with a frequency between 180 and 330 Hz and an amplitude of 1. The sound frequency and amplitudes are predefined and the tone does not depend on the movement anymore. The movement then defines which predefined tone has to be played. Also the length of the sound depends on the movement, as described previously. The length of the sound varies between short tones of 0.8 seconds, which are mapped to quick movements and longer tones of 2 seconds for slower movements. The sound is sampled at 44.1 kHz.

5. EVALUATION AND DISCUSSION

Although the current implementation of our prototype differs from the initial design concept, the most important functionality of the tool is working as expected: reading the sensor data and generation of sound based on the data.

The sensor and filter currently used offers reduced accuracy. This often leads to the user hitting the same region, but not getting the expected audio feedback. Therefore, the second iteration of the implementation was necessary in order to reduce the inefficiency of the system. We consider adding the 6 subareas for the 6 different tones of the hexachordon an improvement in the reliability of the instrument. The current implementation has a success rate of approximately 80 percent. A method to further increase the success rate of the system would be implementing a new filter for the data, such as the 1€-filter [11], Kalman-Filter [13] or to use the intern digital motion processor (DMP) of the sensor for data fusion. Another possibility to improve the sensor data is to add a second sensor and combine the values from both of them in order to get more precise positions.

Unfortunately, the presented filtering methods are expensive and require more computation time. A trade-off between accuracy and speed had to be met, because not giving immediate feedback to the user would defy the purpose of the tool itself.
Furthermore, because of the limitation of the sound synthesis framework used, playing drum sounds is not possible at the moment. A solution for that would be using another framework. Our knowledge in this field is strongly limited, therefore the planned time for the project was not sufficient for including this functionality. Nevertheless, we find that the current design of the sound is a good compromise which does not alter the initial designed functionality much.
Another way of improving the Soundsense system would be to create a better sound experience for the user. Unfortunately, the "wavebender" framework does not allow melting the tones into each other in order to create a continuous melody. A solution would be to add this functionality to the framework or to find a different Python library for sound synthesis.
Due to time limitations, a planned user test of the tool was not possible, although ideas for a future user study could center on discovering whether our assumptions about the ease of learning and ease of use are true. Therefore, the user study should focus on bringing all the test persons on the same level of knowledge with regard to the tool and then testing whether they intuitively discover how it should be used. Furthermore, the previous musical knowledge of the test persons should be filtered, as it might strongly influence the final results of the tests. The tutorial for the tool should also be short enough in order to avoid the test persons being distracted.

Another possible research question for the user study would be to see which of the iterations of the prototype is more appropriate for children and which one for adults. The results of testing the instrument with children would be more accurate if they would be split into groups of ages in order to observe the differences between these groups. Tasks with different difficulty levels according to age could be given in order to assess whether the tool fulfills its goals. To investigate whether the users understood how to use the tool and whether they found it as entertaining as we imagined it, we propose using the "User Experience Questionnaire" as a means of analysis. This questionnaire benchmarks our tool with respect to other tools on six scales: attractiveness, perspicuity, efficiency, dependability, stimulation and novelty [4]. This user study would be appropriate either after implementing the missing functionality of the drum sound or for the current state of the tool and the first iteration of the prototype. We assume that dependability will not score very high due to the unreliable sensor data, but that attractiveness, perspicuity, stimulation and efficiency would have a score of good or very good. Even though the tool is not novel, for those who do not have much experience in the field of music would see the tool as novel, since it is especially designed for them.

6. CONCLUSION

We present here a musical instrument designed for entertainment, but which also offers the possibility to be extended to an actual digital music instrument. The affordance of this instrument should enable the user to learn how to use it fast and without many problems. Even though technical limitations have restricted us from creating a sound feedback that is similar to the drum kit, the current state of the device resembles a virtual xylophone and allows the user to create musical pieces.

7. ACKNOWLEDGMENTS

We would like to thank our project supervisor, Prof. Dr. Günther Schatter, for presenting us with this theme and pointing us to previous works, on which we based our prototype.
Furthermore, we would like to thank Zach Denton for allowing us to use and modify his program.
We also thank our colleagues, Janek Bevendorff and Hagen Hiller, for the Raspberry Pi devices they provided us with.

8. REFERENCES

[1] The digital baton. http://web.media.mit.edu/~joep/SpectrumWeb/captions/Baton.html, July 2015.

[2] The digital baton. http://web.media.mit.edu/~joep/TTT.BO/baton.html, July 2015.

[3] Max mathews' radio baton. https://ccrma.stanford.edu/radiobaton/, July 2015.

[4] User experience questionnaire. http://www.ueq-online.org/, July 2015.

[5] A. Birkett. Reading data from the mpu-6050 on the raspberry pi. http://blog.bitify.co.uk/2013/11/reading-data-from-mpu-6050-on-raspberry.html, July 2015.

[6] C. Chafe. A short history of digital sound synthesis by composers in the U.S.A. 1999. https://ccrma.stanford.edu/~cc/pub/pdf/histSynUSA.pdf.

[7] S. Colton. The balance filter. http://web.mit.edu/~jinstone/Public/filter.pdf, 2007.

[8] P. G. David Keane. The MIDI baton. 1989. http://quod.lib.umich.edu/cgi/p/pod/dod-idx/midi-baton.pdf?c=icmc;idno=bbp2372.1989.037.

[9] C. J. Fisher. Using an accelerometer for inclination sensing. http://www.analog.com/media/en/technical-documentation/application-notes/AN-1057.pdf, July 2015.

[10] GeekMomProjects. Gyroscopes and accelerometers on a chip. http://www.geekmomprojects.com/gyroscopes-and-accelerometers-on-a-chip/, July 2015.

[11] D. V. GÃľry Casiez, Nicolas Roussel. 1e filter: A simple speed-based low-pass filter for noisy input in interactive systems. http://cristal.univ-lille.fr/~casiez/publications/CHI2012-casiez.pdf, May 2012.

[12] I. Inc. Mpu-6000 and mpu-6050 - register map and descriptions - revision 4.0. https://www.olimex.com/Products/Modules/Sensors/MOD-MPU6050/resources/RM-MPU-60xxA_rev_4.pdf, July 2015.

[13] S. J. Julier and J. K. Uhlmann. New extension of the kalman filter to nonlinear systems, 1997.

[14] Wikipedia. Max mathews. https://de.wikipedia.org/wiki/Max_Mathews, July 2015.

[15] Wikipedia. Radiodrum. https://en.wikipedia.org/wiki/Radiodrum, July 2015.

[16] Wikipedia. Theremin. https://en.wikipedia.org/wiki/Theremin, July 2015.