HAW Hamburg
AVPRG

Prof. Dr. Andreas Plaß
Jakob Sudau

Project - VDrums

Niklas Bartsch	niklas.bartsch@haw-hamburg.de	2216285
Simon Hoyos Cadavid	simon.hoyoscadavid@haw-hamburg.de	2319104

1.Introduction.

It is hard to count the amount of times in a student's life, that he, or she, is confronted by the thought of scratching for ideas for all the different projects they must finish. It's not easy being faced by the emptiness of the white page in front of you, staring back, silently looking. During the class Audio and Video Programming in the Hamburg University of Applied Sciences (HAW Hamburg) we, the students, were challenged by one such idea of making a computer program which integrates either one of the following programming languages:

- JavaScript, utilizing the integrated Audio API.
- C++, making use of OpenCV.
- Both.

The clue that was introduced from this was obviously that a software had to be programed, which either used sound in a browser and applied it to something else – for example the artificial creation of music – or utilized OpenCV to code an application that was able to edit video in real time. In the end the decision was made to go for the programing language that had been learnt already and in which the students were confident in: JavaScript.
With JavaScript in mind the concept started flourishing. Not only should the program take music and do something with it but, taking inspiration in the real time video editing instructed in the assignment, it was chosen for the project to be able to take in real time audio from an electronic drum set and represent it in the browser as a music visualizer of sorts. The visualization of the music that's being played in the drum set would be then made in 3D with another tool that that the students had learnt before, three.js. With this integration of WebGL plus the Web Audio API in browser it was thought by the students that it would be enough to satisfy the requirements of the assignment.
The goal of the program is to be able to immerse the user in another "world" which is not our own. Human beings cannot see sounds, only hear them, therefore it's always an interesting experiment to see what is listened, creating impulses, but generating a different kind of reaction to the one that's to be expected. To generate a better involvement both sound and visuals are going to be played at the same time from the computer. The user won't see anything, until he starts playing the drums, drawing a clear parallel to him not hearing anything. Moreover, this project is not to be taken as an experiment, research or especial contribution to humanity, but the whole purpose is for the user to have fun while using it.
After all the explanation above, one would ask oneself the reason behind having this as the basis of one's project. The answer is to the proposed question is quite easy: One of the students working in this project is a drummer and thought that the idea to show the world a little bit of what's happening in his head while playing was interesting enough for it to become a full-fledged project, after all, sharing some part of the world that one has on his or her mind is the seed of every good idea.

2. Goal.

As stated before, the project has a clear goal of bringing the mind of a drummer to life, but the question that comes to mind is what will the user see, when he is playing the drums. The first observation that must be made clear is what the program will **not** be, which is a simple representation of a drum set made in three.js, it will **not** have discernible drums and cymbals that show the person using it exactly what they are playing. The reason for this is that the mind doesn't work this way, the mind is not a crude recollection of images that clearly show what you see with your eyes; the mind is a mixture of colors, sounds and blurry figures that feed our imagination, which is exactly what it's wanted from the program.
The way of thinking explained above can then be exploited to the advantage of the project, as a result concretizing what's going to be shown to the user. The user is going to start with a completely black screen, not being able to see anything – because nothing has been played, there is nothing to hear, therefore nothing to see – instantiating the sense of blindness, which parallels the notion being deaf.
As soon as the user starts playing the drums, the person will start seeing different basic shapes appear in the respective places, each representing one of the eight different pieces of the instrument[1]. Out of each one of the aforementioned basic shapes, different effects will take place to simulate the same kind of sound, but visually. These so named effects are going to vary depending of what is being hit, because of course a ride, for example, has a different sound than a snare drum. Taking all this into account the following effects will take place in the screen:

- *Cymbals*: All cymbals start with a size of zero. When they are struck, their size will increase depending of the gain of the sound and the color will change accordingly.
  - *Hi Hat*: The Hi Hat will be represented by two cubes, one on top of the other floating with a little bit of distance in the Y axis in between (Aligned in both X and Z axis). Since the Hi Hat has normally two states, so will the visual representation.
    1. *Open*: While the Hi Hat remains opened, the Y distance between both cubes is a couple of dozen pixels long, enough to discern that the cymbal is open. If the cymbal is hit while open it will produce a big number of particles exploding out of the cymbal in a more savage fashion. While this is happening, both cubes will start shaking accordingly, like the vibration of the cymbal.
    2. *Closed*: While closed, the Y distance is zero. If struck, the cubes will generate a more contained output: Particles will sprout out of the contact surface of both cubes, but instead of exploding, they are released in a circle forming outwards and disappearing after a certain time. The particles generated will all move together in the circle across the X and Z axis, but their Y position will change making the form of a sine curve.

[1] Hi Hat, Ride, Crash, Tom (One), Tom (Two), Floor Tom, Bass Drum and Snare Drum.

- *Crash*: The crash represents the wildest and most wrapping sound of the drum set. It represents a change, an explosion of sound. Because of this the visualization of the crash has to represent the same thing. The crash in our representation is a sphere, which as it expands, throws an explosion of particles all over the place, that swiftly fly outwards in all directions, until the sound stops.
- *Ride*: This cymbal usually has three different sounds, but for the sake of this project, only one of them will be used. The ride (unless struck on the border, which won't be used for this project) is a more contained sound than the craziness of the crash and its representation will demonstrate it. As the user hits the ride a tornado of particles splashes out of a circle. The tornado appears as long as the sound of the ride plays. Apart from this, each time the note is casted – or better each time it is struck – a pentagon will appear horizontally from the screen, this figure will be slightly tilted in order for it to be seen.

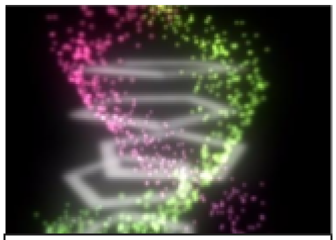
Image 1: Inspiration Ride

- *Drums*: The color of the drums visual representation change accordingly when being played.
  - *Snare*: The snare is represented by a circle engulfed in bars. Vertical lines will be popping from the circle, depending on how the sound and gain is.
  - *Toms (One and Two):* The toms are going to be shown as small and short cylinders, tilted slightly to the sights (resembling their real shapes). When hit their height changes, without moving the base, meaning only "growing" upwards. From the upper base of the cylinder particles might jump out creating arcs.
  - *Floor Tom:* Much like the other toms, the floor tom will be a cylinder, but one bigger, wider and heavier erecting upwards from the floor. This cylinder does not only change its height, but also its radius when its triggered.
  - *Bass:* The bass drum is supposed to represent the base of the rhythm in the drum set. For this reason, the bass will be the floor on which every single piece stand, as a plane in the 3D room. It's not going to start at a size of zero – in contrast to many of the other pieces – but it's going to be the same color of the background. As soon as struck, the plane will become a few pixels bigger in all axis, while changing color and being swiftly elevated another few pixels in the Y axis; after that the plane will start descending, changing color slowly until it comes back to the original parameters of position and color.

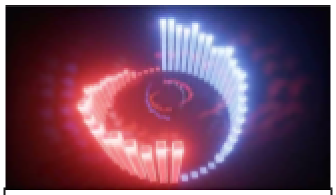
Image 2: Inspiration Snare

3. Requirements analysis.

Before beginning the concise description of the program, it should be taken into account that it will have a clear input from the user, being him playing the drums; with a direct real time output from the browser – the sound and visualization. It is also important to note that at the moment the user starts the program, proceeding to play the drums, the 3D world is already computed by it, meaning the world is ready to react to the user's actions. After it is finished loading a MIDI signal will be sent from the drum set to the browser, which will be then read with help of the JavaScript Audio API. This will in turn change both the world that's going to be displayed, and the respective sound.

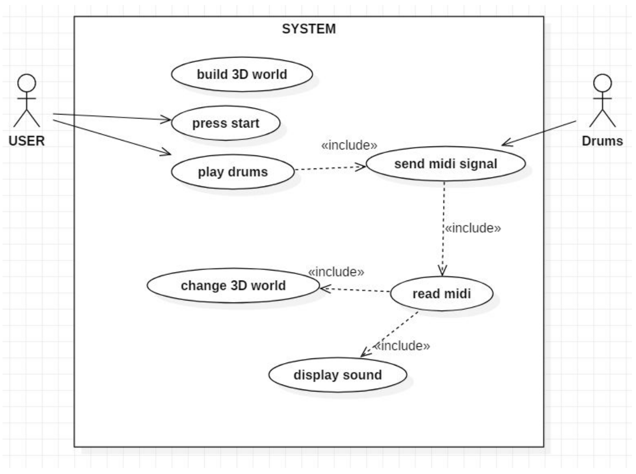

Diagram 1: UML Use Case Diagram

The system requirements for the project are not that complicated. A computer is needed with microsoft windows installed, with any of the most common used browsers (But safari might bring several problems). Concerning three.js it would be better for the computer to have a good processing capacity, so all the meshes can be rendered in a swiftly manner. Preferably the minimal requirements for the system should be around 4GB RAM and a four core processor, with a non-integrated graphics card. This will made all the process faster, which in turn will reduce the latency: this is wanted to be as low as possible, considering one would like to be able to listen and see directly after the pad has been struck.

4. Technical Frameworks.

As stated before, the program will be coded utilizing the JavaScript programming language – with of course implies that it is a web application. Most of the Web applications of today are programmed using a combination of html5, JavaScript and CSS, and this program is no exception. The reason of the choosing of JavaScript is because of the two different programming languages that were made a choice for this specific project, JavaScript was the one the authors of this text were more confident with. Moreover, the authors also wanted to focus their attention in audio programming, which only made the choice more obvious. The best approach to audio editing in web development is the JavaScript integrated Web Audio API, which is going to be used for this project. Having this in mind this API will be exploited to read the MIDI signals from the drum set.

Another important aspect in the matter is the visual representation in the browser. This will be done using the open source library three.js. Three.js applies the canvas element provided by WebGL, expanding on it, making an ease of use and access framework for 3D visualization. It was decided to use three.js because the project was thought to be realized in a digital 3D world, while exploiting the characteristics, like particle systems, that three.js provides. It is not known what technical difficulties the usage of three.js can bring to the project, but the authors are optimistic that it will fulfill the requirements.
Derailing the topic to the hardware usage, the idea is for the program to run in a computer with access to a browser – most for that matter –, an Alesis Nitro Drum set Kit and a USB 2.0 cable (USB A male to USB B male). The Alesis Nitro Kit will be provided by the authors of this text.

5. Technical Concept.

After the explanation of the chosen Frameworks, it is of utmost importance to have a clear view of how the architecture of the project is going to be. First of all, the program will have three main libraries, which are going to be used to develop the concept; these libraries are the MIDI library and the Web Audio API integrated in JavaScript and three.js that's going to be in charge of the 3D world building. Apart from the helping components, the project is going to implement four different main software components, with a hardware component linked to the end. The four main software components are the MIDI controller, which is in charge of receiving MIDI signals, while interpreting them, giving them the respective frequency and velocity; the Web Audio Controller, which will either generate the sounds that are going to be displayed, or just play the sounds (It is important to know that the sounds are either computer generated utilizing oscillators or are just pre recorded sounds to be played. This will depend on time and effort); the 3D World Controller is where three.js will be implemented, and it's the component in charge of all the building and changing of the 3D world; and the Browser Action Controller, which as the name implies will be controlling all the actions the browser has to do (For example here is where the HTML5 document will be contained). In the end the only remaining component is the Drum set, which is a hardware controller that only generates and sends MIDI signals.

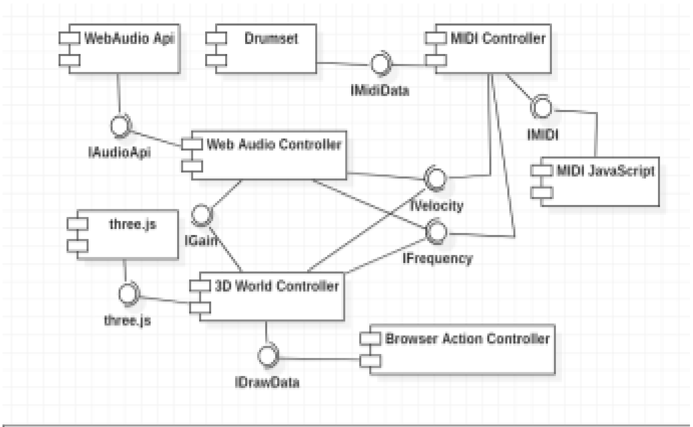

Diagram 2: UML Component Diagram

6. Operational Concept.

We want to keep the operation as simple as possible, because no one wants to get distracted by too many options in a new unknown software. We start with a black browser window, representing deafness. Once the user plugs in his or her electrical drum set into the USB Port of the PC, the browser window will show a notification message the device was recognized. If everything is working as intended the visualization starts with a short tutorial to test each piece of the drum set.
After the tutorial the visualizations of the different drums start instantly as the user plays the drums. Making the parallel between sound and image.
We are going to make the sound with the Web Audio API, utilizing either prerecorded sounds, or the internal oscillator provided by the Web Audio API (This will depend on the timing left and the amount of effort); so that the user is able to choose from different Drum-Sets in the settings. The preferences also include various options to change the volume, the background color and other settings. Nice to have would be distortion effects for the oscillator.

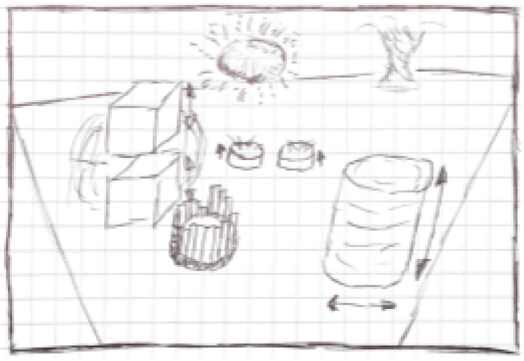
Image 3: Raw first Sketch of main GUI

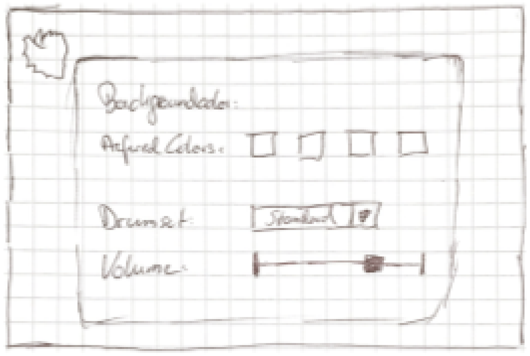

Image 4: Raw first Sketch of Preferences GUI

7. Time Plan.

Nr.	**Task**	**Date**	**Time (h)**	**Who (N - Niklas, S - Simon)**
1	Milestones			
1.01	PC reads MIDI Inputs	23.10.18	2	S
1.02	MIDI Inputs works with THREE.JS	1.11.18	6	N, S
1.03	Prototype with sound (Played Sounds)	1.11.18	4	N, S
1.04	Prototype with simple visualizations and oscillator.	4.11.18	10	N, S
1.05	Visualization - Testing what is possible	4.11.18	10	N, S
1.06	Visualization - High Hat	6.11.18	8	S
1.07	Visualization - Crash	13.11.18	8	N
1.08	Visualization - Ride	13.11.18	8	S
1.09	Visualization - Snare	13.11.18	8	N
1.10	Visualization - Tom1	13.11.18	6	S
1.11	Visualization - Tom2	20.11.18	3	N
1.12	Visualization - Tom Floor	20.11.18	4	S
1.13	Visualization - Base	1.12.18	3	N
1.13	Final Audio adjustments	3.12.18	6	N, S
1.14	Final adjustments	5.12.18	15	N, S
2	Deliveries			
2.1	Project-KickOff	16.10.18	3	N, S
2.2	Concept Preparation	30.10.18	20	N, S
2.3	Prototype Presentation	27.11.18	5	N, S
2.4	Presentation	18.12.18	8	N, S
2.5	Project report	13.01.19	15	N, S
	Total		**142**	**N, S**

8. Team Plan.

Due to both authors have some experience with JavaScript and the required libraries, we will not need too much time to get back into the languages. In the time overview it can be seen that about 142 hours will be needed for the project to be completed, this is mostly due to wanting to be careful with the details (Especially particles). Both have about the same tasks as both are interested in programming the visualization. Niklas is going to take over the project management because he hasn't got any experiences in the past with managing a project and other people have done it in his project teams. Because Simon is the only one with an electronic drum set, it was set to regularly meet at his place for programming and to make direct attempts on the hardware.