DT2300 Sound in Interaction Lab 2 & 3

In the two laboratory sessions, you will tackle three levels of increasing complexity. In the fist laboratory session you should finish level 1 and level 2. In the second session, you have to finish level 3. Each level contains several tasks for you to choose from. You finish a task by presenting your finished results to the lab session supervisor. If you haven't finished level 1 and 2 when the first laboratory session has ended, you have to finish them by yourself before the next session so that they can be presented at the start of the second lab session.

Read the instructions for each task carefully and ask questions if something is unclear. Each task has a list of suggested objects that will help you solve the task. Look at the help files for the suggested objects and make sure you understand how they work. Make sure that your solution fulfils all the requirements in the instructions before you present your work.

- The lab is equipped with computers, but if you prefer it you can use your own laptop.
- To log in to the lab computers, use the user name "stud101Lab" and the password "stud101Lab".
- Create a folder and save all your files to one place. When the lab session is done, back up your files and then delete them from the lab computer.
- Save often and save to different files, maintaining a history of your work.
- If you open files in pd by double clicking them in Explorer a new instance of pd will be created which might lead to confusing results. Use the file-menu in pd to avoid this.
- The integrated help in pd contains enough information for you to be able to solve all the tasks below. Right click on an object and select "Help" to get more info on that object

Level 1: Getting Started

Some simple exercises that cover the basic pd objects that you need to be familiar with to solve the tasks in level 1 & 2. You must finish all tasks before advancing to the next level.

Bang by comparison

Create a variable number "A" (e.g. a slider or a number-box). Compare that number to another number "B". The comparison should output 1 when "A" is bigger then "B" and 0 otherwise. Bang once when the comparison changes from 0 to 1. Important objects to explore are VSlider, Number, Bang, [<], [select], [change].

Combine comparisons

Create a variable number "A" (e.g. a slider or a number-box). Compare that number to two other numbers "B" and "C". The comparison should output 1 when "B" < "A" < "C" and 0 otherwise. Bang once when the comparison changes from 0 to 1. Important new objects to explore are [∗].

Beep

Create a toggle button that controls the amplitude of a sinus oscillator. When the toggle is active, the sound from the oscillator should be heard. When the toggle is inactive, there should be no sound. The oscillator should have a frequency of 900 Hz. Important new objects to explore are Toggle, [∗~], [cycle~].

Beep by comparisons

Combine a comparison like the one in task 2 ("B" < "A" < "C") with the beep from task 3 so that the sound is heard only when the comparison is true.

Map to a scale

Create a sawtooth oscillator and two variable numbers. Use the variable numbers to control the amplitude and pitch of the oscillator. Limit the variable numbers so that the amplitude control can vary between 0 and 1, while the

pitch control varies between the C note at 130.813 Hz (MIDI note nr 48) and the C note at 523.251 Hz (MIDI note nr 72). In addition, the pitch control should change in steps of 2 semitones, creating a whole note scale. Important new objects to explore are [int], [mtof], [phasor~].

Level 2: Musical Instruments

Create real time musical interaction by mapping the data from the MoCap system to synthesised sounds in pure data. Use the provided patch for a reference on how to receive data from the MoCap system in pd. Select two tasks and solve them to move on to the next level.

Theremin

Create a Theremin-like instrument by controlling a sound using the position of your trackable object. Map the x, y, and z position of the object to the amplitude, pitch and timbre of the sound, respectively. Make sure that the possible values for the amplitude and pitch are reasonable and create an interaction where you can control the sound reliably enough to play a simple melody. Smooth the incoming MoCap data to reduce uncontrolled variations in the sound. Create the sound by generating a sawtooth wave and modify the timbre with a low pass filter. Important objects to explore are [mavg], [lop~].

Drum set

Create a virtual drum set by defining a set of 3 squares in the MoCap space. Detect when your object hits a square and play a sound. A square is considered hit when the object passes through it in one direction (but not the other). The sound should be played only once when the object hits the square. Connect the different squares to different sounds. One sound should consist of band pass filtered noise, emulating a snare drum. Another sound, emulating a hi-hat sound, should consist of three sawtooth waves that are multiplied with each other and sent through a high pass filter. Choose whatever method you like to produce the third sound. The sounds should be perceived as equally loud. Important objects to explore are [noise~], [bp~], [hip~], [line~].

Velocity sensitive piano

Create a virtual piano by defining a keyboard rectangle in the MoCap space. Detect when your object hits the rectangle and play a piano sound. The rectangle is considered hit when the object passes through it in one direction (but not the other). The sound should be played only once when the object hits the rectangle. Depending on where the object passes through the rectangle, the resulting sound should have a different pitch, just like how each note on the piano has a different pitch. Make sure that the pitches correspond to the notes available on a piano. The sound should be generated by playing back a sample of a piano note. A sound file containing a piano note will be provided in the lab files. To achieve the difference in pitch, the playback-speed should be varied. The amplitude of the sound should correspond to the speed of the trackable object when it hits the keyboard rectangle. Important objects to explore are [tabread4], [array], [line].

Your own idea

If you have an idea of your own for a musical instrument, talk with the lab session supervisor and make sure that it is practically possible and of suitable complexity.

Level 3: Games

Create playful interaction using interactive real time audio production. Solve at least one task to complete the lab session.

Head, Shoulders, Knees and Toes

Create a periodic rhythm and play a sound on each beat. Measure the height of a player's head, shoulder, knees and toes. Create a game where the player is to move the trackable object so that it is equal in height to the measured heights, on time with the beat, in the a predefined order. Play a sound every time a correct height is measured and another sound if an incorrect height is measured. Reset the game if the player gets it wrong, i.e., an incorrect measurement is detected at the time of the beat.

Seek and Smash game

Choose a random point in the capture space. Guide the player to the point using interactive sound. You can use whatever sounds you like. You should at least map the distance to the point to the sound, but if you also use the direction to the point it will be more effective. When the player moves the trackable object close enough to the point you should play a sound and move the point to a new random location in the capture space.

Your own idea

If you have an idea of your own for a game, talk with the lab session supervisor and make sure that it is practically possible and of suitable complexity.