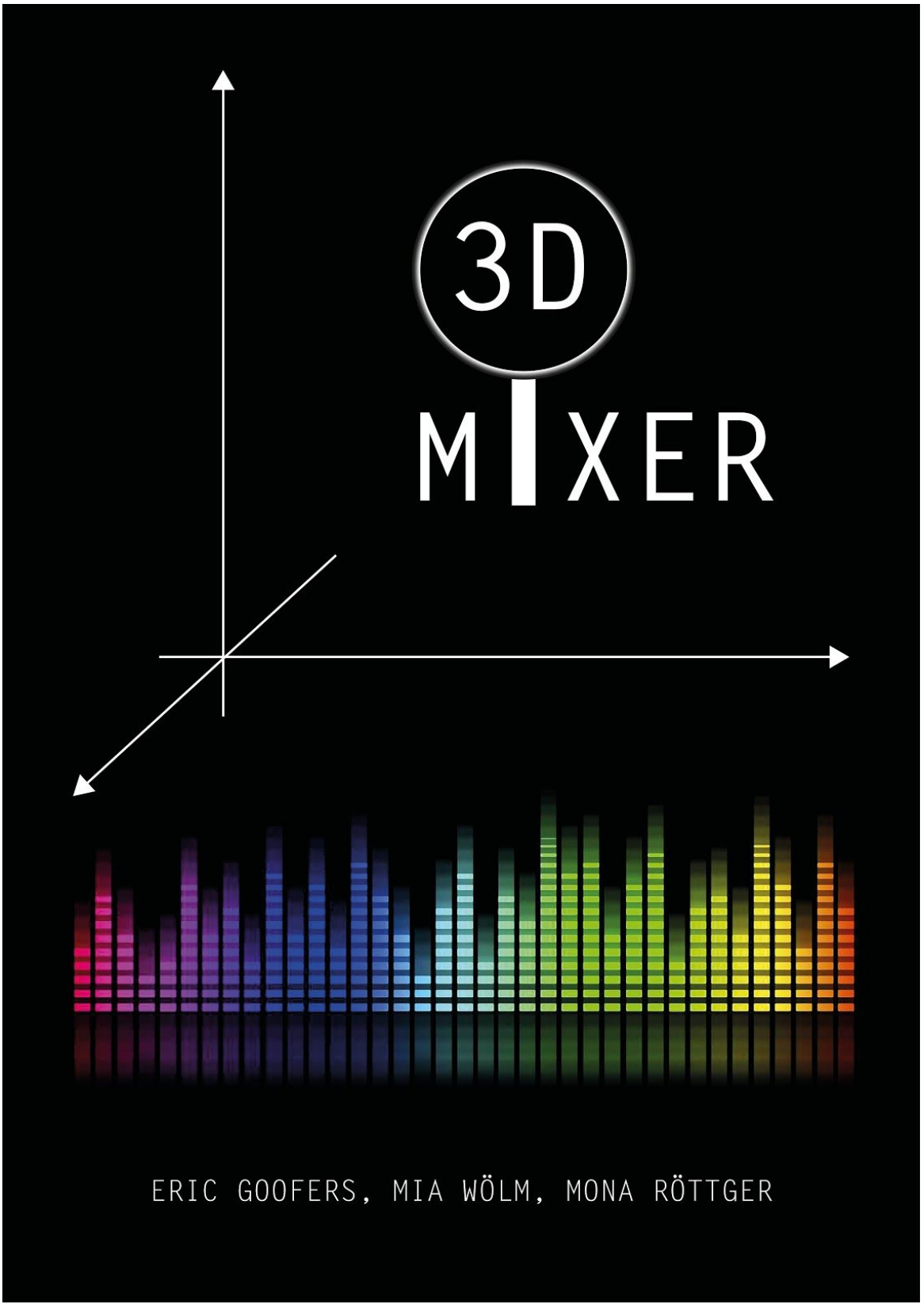
Technische Dokumentation “3D-Mixer”

HAW Hamburg | Media Systems | AVPRG | Plaß / Sudau



Über das Projekt

Mit dem 3D-Mixer können bis zu vier User gleichzeitig dynamisch synthetisch erzeugte Instrumente in einem simplen Song steuern. Die User können frei auswählen, welches Instrument sie im Song steuern möchten und können dann den passenden Knob frei im Raum bewegen.
So entsteht gemeinschaftlich eine Live-Performance, die sich in den Bewegungen der User ausdrückt und die Musik fühlbar und erlebbar macht.

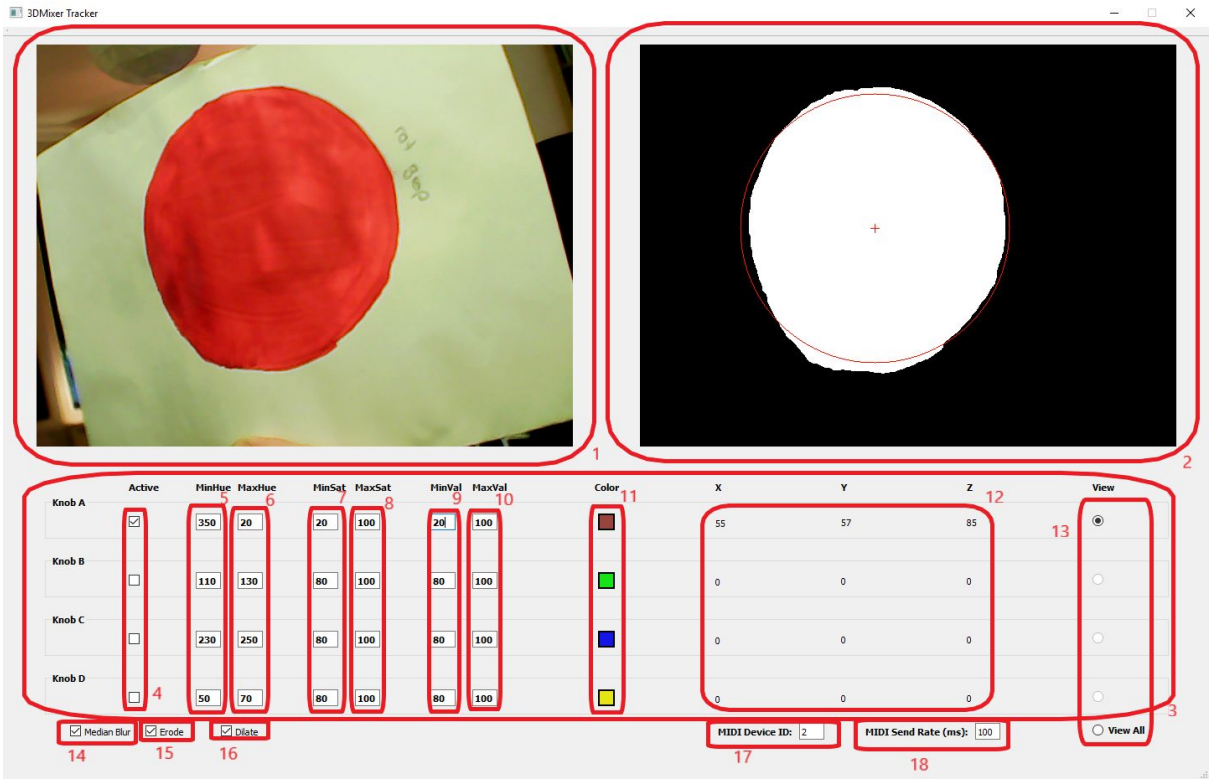
Aufbau

- Der Interpreter erkennt beim Start automatisch die Default-Systemkamera und beginnt die Tracking-Daten auf dem eingestellten MIDI-Kanal zu senden
- Der Synthesizer wird im Webbrowser Chrome geöffnet und empfängt automatisch die MIDI-Daten
- Der User wählt einen aus drei Songs aus und weist seinem Knob ein Instrument zu
- Instrumente werden per Javascript synthetisiert, die Parameter der Klangsynthese werden auf Basis der Midi-Daten aus dem Interpreter verändert
- Bis zu vier verschiedenfarbige Knobs können getrackt werden, wodurch jedes der vier auswählbaren Instrumente in einem Song abgedeckt werden kann
- Theoretisch könnte jeder der vier Knobs von einer anderen Person gehalten werden, wodurch bis zu vier Personen gemeinsam eine Live-Performance kreieren können

Interpreter

Der Interpreter ist in C++ mit dem QT Creator und mithilfe der opencv-Bibliothek realisiert. Über ein GUI kann der User bis zu vier Knobs gleichzeitig tracken lassen, jeweils individuell die Parameter für das Tracking einstellen und aus weiteren Optionen für Tracking und Output auswählen.
Die durchs Tracking berechneten Positionsdaten werden für jeden Knob einzeln ausgegeben. Außerdem kann sich der Benutzer die Maskierung und Koordinatenberechnung von entweder einem einzelnen oder allen Knobs zugleich grafisch anzeigen lassen.
Das Programm wurde auf Leistung optimiert, um das Intervall der Positionsdaten-Updates so klein wie möglich zu halten.

GUI

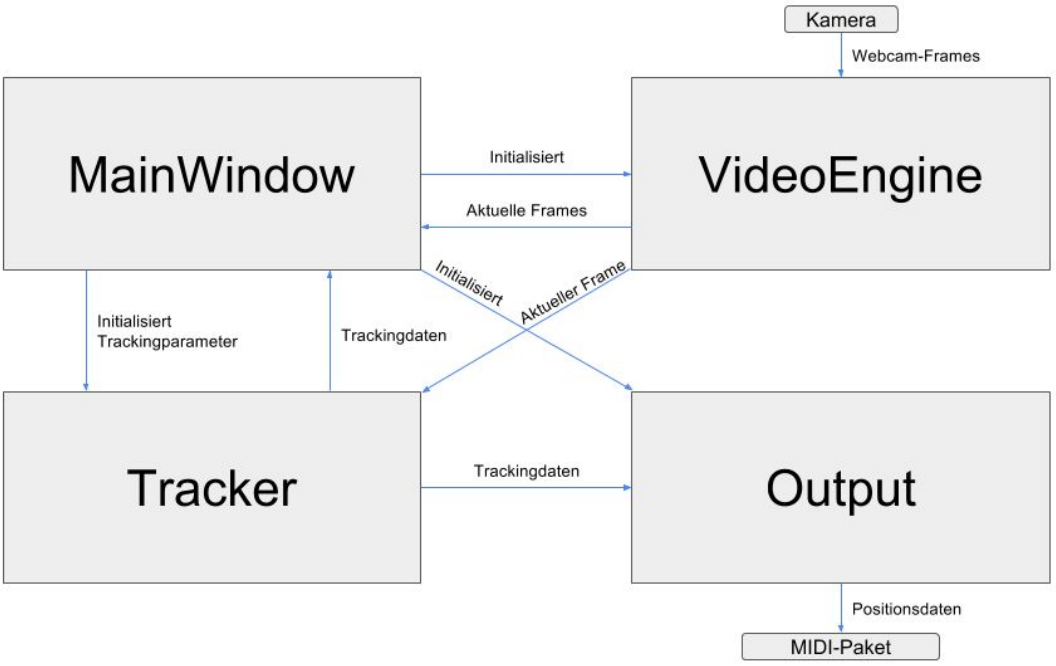


Legende:

1. Das von der Kamera aufgenommene Rohbild
2. Das Binärbild des momentan ausgewählten Knobs mit grafischem Tracking-Feedback
3. Die vier Knobs mit jeweils einstellbaren Parametern und Trackingdaten
4. Soll dieser Knob getrackt werden? *(nicht benutzte Knobs sollten deaktiviert werden, um die Leistung des Interpreters zu erhöhen)*
5. Die untere Schwelle des *Hue*-Werts des Knobs *(Wertebereich: 0-359)*

6. Die obere Schwelle des *Hue*-Werts des Knobs *(Wertebereich: 0-359)*
7. Die untere Schwelle des *Saturation*-Werts des Knobs *(Wertebereich: 0-100)*
8. Die obere Schwelle des *Saturation*-Werts des Knobs *(Wertebereich: 0-100)*
9. Die untere Schwelle des *Value*-Werts des Knobs *(Wertebereich: 0-100)*
10. Die obere Schwelle des *Value*-Werts des Knobs *(Wertebereich: 0-100)*
    (*Für das Umrechnen von RGB zu HSV -Farbwerten bietet sich das nützliche Tool Colorizer an.)*
11. Die aktuell gewählte Farbe des Knobs *(ergibt sich aus den Durchschnittswerten von Hue, Saturation und Value)*
12. Die zuletzt erkannten Koordinaten der Knobs *(Wertebereich 0-127, diese Werte werden per MIDI verschickt)*
13. Welches Binärbild angezeigt werden soll. Ist ein Knob deaktiviert, wird kein Binärbild von ihm erstellt, weshalb die Anzeigeoption dann nicht verfügbar ist.
    Mit Auswahl von“ViewAll” werden die Positionsdaten aller aktiven Knobs in deren aktueller Farbe grafisch dargestellt
14. Sollen die Binärbilder zum Verbessern des Trackings mit einem *Median Blur* versehen werden?
    (Deaktivierung verbessert Leistung erheblich, senkt die Qualität des Trackings bei schlechten Lichtverhältnissen jedoch auch deutlich)
15. Sollen die Binärbilder zum Verbessern des Trackings erodiert werden?
16. Sollen die Binärbilder zum Verbessern des Trackings anschließend nochmal geweitet werden?
17. Der MIDI-Kanal, auf dem die Trackingdaten gesendet werden sollen
    (kann nicht größer als die Anzahl der verfügbaren MIDI-Kanäle sein)
18. Das Intervall, in dem die aktuellen Trackingdaten per MIDI gesendet werden
    (in Millisekunden, Wertebereich: 10-9999)

Programmaufbau und Klasseninteraktionen



Im Wesentlichen besteht der Interpreter aus vier Klassen:

MainWindow

Verwaltet das GUI, initialisiert alle anderen Klassen. Gibt die Tracking-Parameter an den *Tracker* weiter.

VideoEngine

Zuständig für das Auslesen der Webcam-Frames. Leitet die Frames an *MainWindow* und *Tracker* weiter.

Tracker

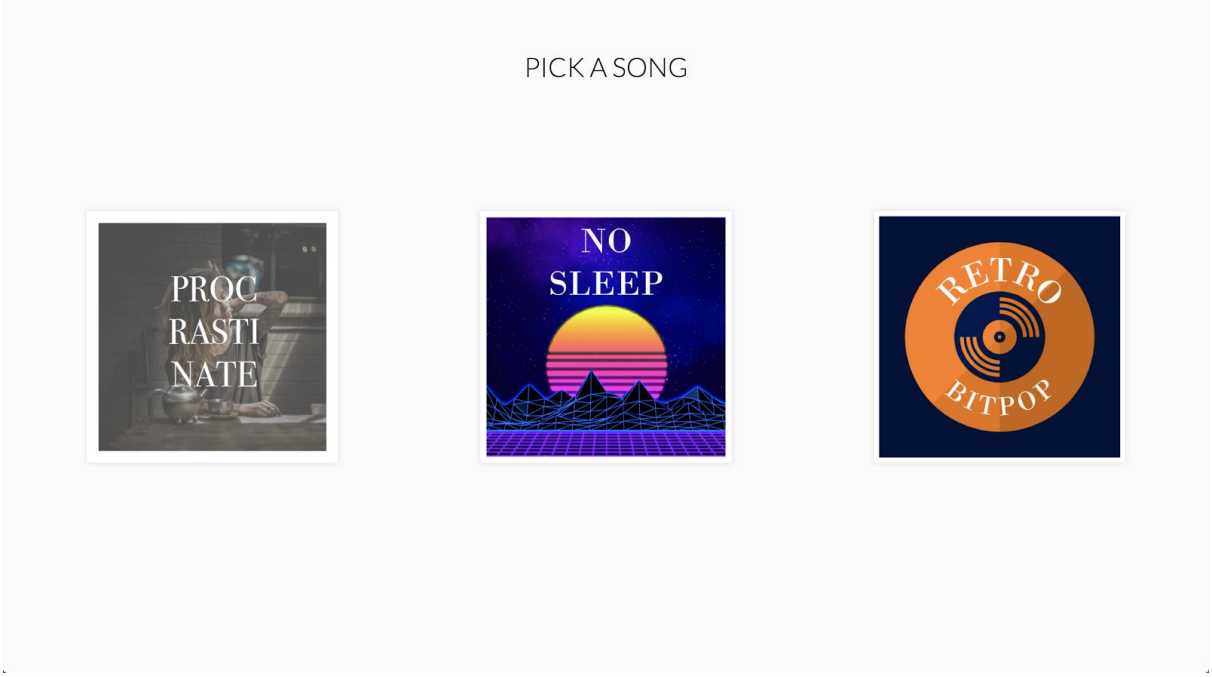
Implementiert den zentralen Algorithmus für das Tracking der Knobs. Berechnet die Positionsdaten auf der x- y- und z-Achse der Knobs und speichert diese zwischen, wo sie jederzeit von *MainWindow* und *Output* abgerufen werden können.

Output

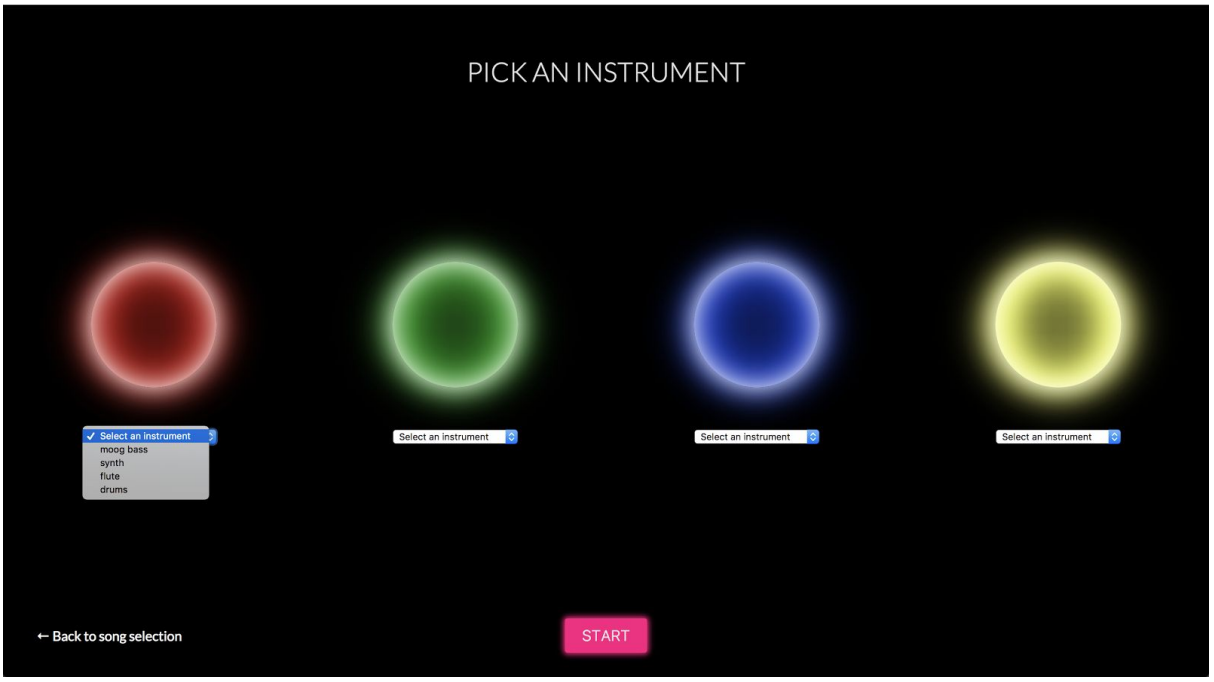
Verantwortlich für das Senden der Positionsdaten des Trackers über einen MIDI-Kanal. Der MIDI-Kanal und das Sendeintervall können vom User über das GUI festgelegt werden.

Synthesizer

Nach dem öffnen des Synthesizers kann der User zwischen drei Songs auswählen. Wenn er mit der Maus über ein Song Cover hovert, kann er sich den Song anhören.



Die Songs haben wir komponiert und als ein MIDI-File exportiert, in dem alle Instrumentenspuren definiert sind. Entscheidet sich der User für einen Song, wird das passende MIDI-File geladen. Um die Informationen, wie z.B. wann welcher Ton gespielt wird, herauszulesen, benutzen wir einen Javascript-MIDI-Converter. Dieser dekodiert MIDI-Files in ein JSON-Objekt.
Aus diesem JSON-Objekt lesen wir alle Instrumente in dem Song aus und erstellen dann für jeden Knob eine Selectbox mit den jeweiligen Instrumenten als Optionen. Der User kann nun auswählen, welches Instrument er mit welchem Knob steuern möchte.



Für jedes Instrument wird ein JS-Instrument-Objekt erstellt und die Informationen aus der MIDI, wann welcher Ton gespielt wird, mit übergeben. Wird der Song nun gestartet, wird für jeden Knob, der einem Instrument zugewiesen wurde, ein solches Instrument initialisiert und ein Scheduler gestartet. Dieser Scheduler plant die jeweiligen Töne der Instrumente schon ein paar Millisekunden, bevor sie abgespielt werden, damit das Timing in dem Song stimmt. Für jeden Ton eines Instrumentes wird ein Oszillator initialisiert. Hat ein Instrument mehrere Voices, dann wird für jede Voice ein Oszillator erstellt. Jeder Oszillator wird mit einer GainNode verknüpft. Diese steuert die Lautstärke des Instrumentes. Die GainNode wird dann mit zwei BiquadFilterNodes verknüpft. Diese steuern den Lowpassfilter.
Danach wird jeder Oszillator mit der vom Scheduler übergeben Zeit gestartet, und wieder gestoppt, wenn der Ton zu Ende ist.
Die Drums (Kick, Hihat und Snare) sind einfach WAV-Dateien, die lediglich mit einer GainNode verknüpft sind. Sie stellen die einzigen Sounds dar, die nicht dynamisch synthetisiert werden.

Wird jetzt ein Knob vom User bewegt, werden MIDI-Daten, mit dem jeweiligen KnobIndex und den dazugehörigen Positionsdaten versehen, vom *Interpreter* an den *Synthesizer* gesendet, der entsprechendes Paket den richtigen WebAudio-Nodes zuordnet und so die Klangsynthese beeinflusst.

Der Synthesizer wurde von vornherein mit Erweiterbarkeit aufgebaut, dass heißt weiter Songs können einfach hinzugefügt werden. Der Song muss als MIDI-Datei vorliegen und es muss eine JSON-Datei erstellt werden, die Informationen über die jeweiligen Instrumente, wie z.B. Voices, Volume, Lowpassfilter etc. enthält. Dann kann beides einfach vom Synthesizer geladen und die Instrumente initialisiert werden.

Abweichungen vom Originalkonzept

Anders als im Originalkonzept vorgesehen, besteht der obere Teil der Knobs aus Styropor und nicht aus Holz. Damit der Interpreter die farbigen Knobs gut tracken kann, müssen sie einen bestimmten Durchmesser aufweisen. Holzkugeln dieser Größe hätten das Budget gesprengt und erwiesen sich dazu als überraschend schwer und unhandlich. Die Styropor-Lösung ist deutlich leichter, trotzdem liegt der Knob durch seinen Holzgriff gut in der Hand. Die verwendeten Farben besaßen eine gute Deckkraft, waren leicht aufzubringen und wurden in den Testphasen gut von der Kamera erkannt. Es stellte sich allerdings heraus, das einige Farben stärker glänzten als erwartet, was den Knob für schwierige Lichtverhältnisse deutlich anfälliger machte.
Bei der Präsentation im Tonlabor hatte der Interpreter aufgrund der komplizierten Lichtsituation leider einige Schwierigkeiten, die gewählten Farben herauszufiltern.

Auch waren für die finale Version weitere Instrumente geplant, für die jedoch am Ende nicht mehr Zeit war. Glücklicherweise ließen sich alle drei Songs ohne größere Probleme mit denselben Instrumenten realisieren.

Team

2218011 - Eric Goofers
2220740 - Mona Röttger
2221086 - Mia Wölm

- *Hamburg, 23. Januar 2017*