Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Fakultät Design, Medien, Information

Kunst- und Mediencampus Hamburg

Department Medientechnik

Studiengang Media Systems

Audio-Video-Programmierung

Projekt:

Rhythm-Machine

„Entwicklung einer virtuellen Loop-Station mit Gestensteuerung“

Projektteilnehmer: **Marko Vukadinovic, Sebastian Bohn, Johannes Bagge, Florian Langhorst**

Projektleiter: **Prof. Dr. Andreas Plaß**

Inhaltsverzeichnis

1 Projektbeschreibung

Das Projekt beschäftigt sich mit der Erstellung und Funktion eines gestengesteuertem Pattern-Sequencer. Mit einem Pattern-Sequencer hat man die Möglichkeit einfache Sounds zu komponieren und einem sich wiederholendem Muster (loop) abzuspielen. Die Steuerung erfolgt über die Gesten eines vorher kalibrierten Objekts, dass über die Webcam sichtbar ist.
Die Entwicklung basiert auf der Programmiersprache C++, der Bildverarbeitungsbibliothek OpenCV, der Audiobibliothek SFML und der Entwicklungsumgebung Qt Creator und fand unter Linux statt. Für die weitere Ausarbeitung ist eine Kompatibilität für Windows und iOS basierte Systeme geplant.
Die Entwicklung der Anwendung wurde in zwei Gruppen aufgeteilt.

Teilgruppe A: GUI
Florian Langhorst, Johannes Bagge

Teilgruppe B: Programmierung
Marko Vukadinovic, Sebastian Bohn

Teilgruppe A: GUI

2 Entwicklung der grafischen Benutzeroberfläche

2.1 Entwurf

Für eine Steuerung einer grafischen Oberfläche mittels Gesten ist es wichtig, dass interaktive Objekte ausreichenden Abstand zueinander haben, um den Auswahlvorgang zu präzisieren. Sounds sollen in horizontalen Spuren gespeichert werden, welche mit Positionen zum zeitlichen Verlauf bestückt sind. Über Positionen innerhalb der Spuren wird gewählt zu welcher Zeiteinheit der Sound abgespielt wird und wiederholt sich nachdem diese durchlaufen wurde. Die Kalibrierung des Steuerungsobjekts erfolgt über RGB-Slider, die den Hoch- und Tiefpassfilter steuern. Die Annäherung an das gefilterte Objekt kann man in einem zusätzlichen Frame sehen, sowie dessen Position in einem Textfeld.

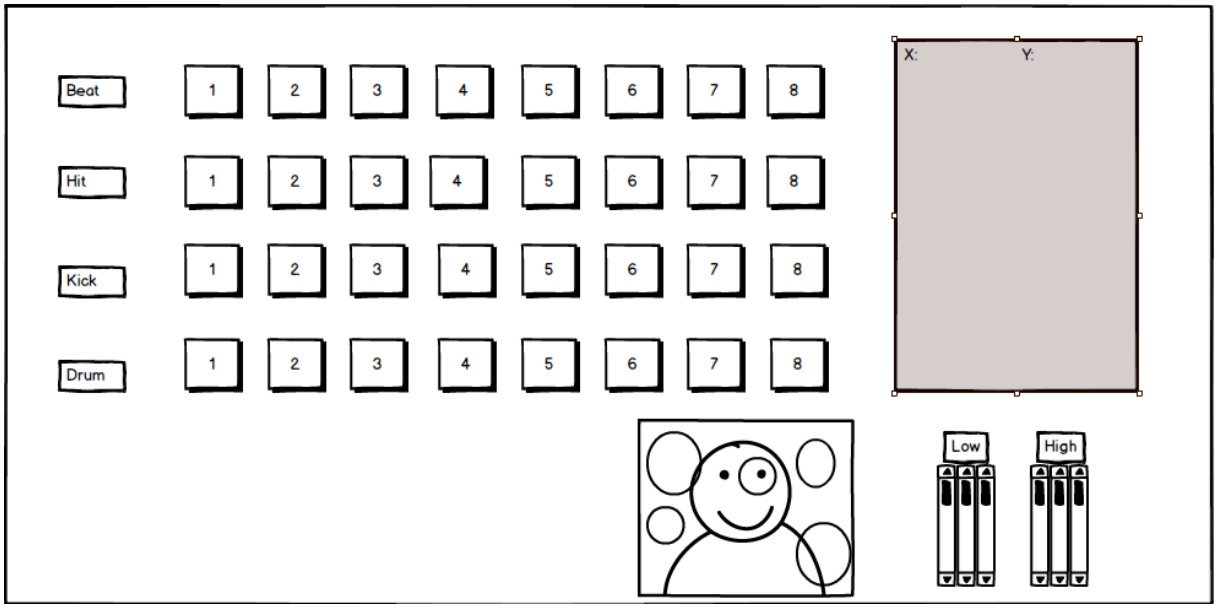

Abbildung1: Entwurf des GUI als Mockup

2.2 Umsetzung

Um eine demonstrative Variation der Kompositionen mit übersichtlicher, grafischer Oberfläche zu gewährleisten, kann mit vier verschiedenen Spuren mit je acht Zeitverlauf-Buttons gearbeitet werden. In demselben Frame wird das normale Webcam Bild gezeigt. In zwei weiteren Frames sind Kopien des Videobilds im HSV-Farbraum und als Binärbild zu sehen. Der Processed-Frame zeigt durch binäre Darstellung an, welcher Bereich des aktuellen Bildes getrackt wird und als aktuelle Cursorposition übergeben wird. Der getrackte Bereich wird im Processed-Frame in weiß dargestellt und zeigt die Annäherung die sich durch die darunter befindlichen Slider und Nummernboxen einstellen lässt. Der Frame im HSV Farbraum hilft, den Farbwert des zu trackenden Objekts leichter zu filtern und zu erkennen. Durch jeweils drei Regler lassen sich hier die Hoch- und Tiefpassfilter auf die RGB-Kanäle des Bildes einstellen. Die eingestellten Werte werden dann in den HSV-Farbraum konvertiert. So werden die Farbwerte, die nicht dem gewählten Objekt entsprechen herausgefiltert, die Position des Cursors wird kalibriert und einem Textfeld als X und Y Koordinaten ausgegeben.

Die vier Tonspuren sind farblich getrennt und jeweils mit einem Sample im WAV-Format belegt.

Bewegt sich der Cursor über eines der farblichen Felder, wird dieses durch die Verringerung der Farbtransparenz gezeigt. Verweilt man auf einem der Felder für zwei Sekunden, wird das Feld aktiviert oder deaktiviert. Bei aktiviertem Feld wird ein Sound an entsprechendem Zeitpunkt abgespielt und im Takt geloopt.

Um die Wiedergabe zu pausieren, kann der Button “Pause Audio” betätigt werden oder mit einer Wisch-Geste nach rechts aus dem Bild. Um sie wieder zu starten, kann selber Button betätigt werden oder eine Wisch-Geste nach links aus dem Bild erfolgen.

Das Video-Capturing kann durch den Button “Pause Capturing” unterbrochen und wieder gestartet werden. Im unterbrochenen Zustand wird die Kontrolle des Cursors wieder an die Hardware übergeben.

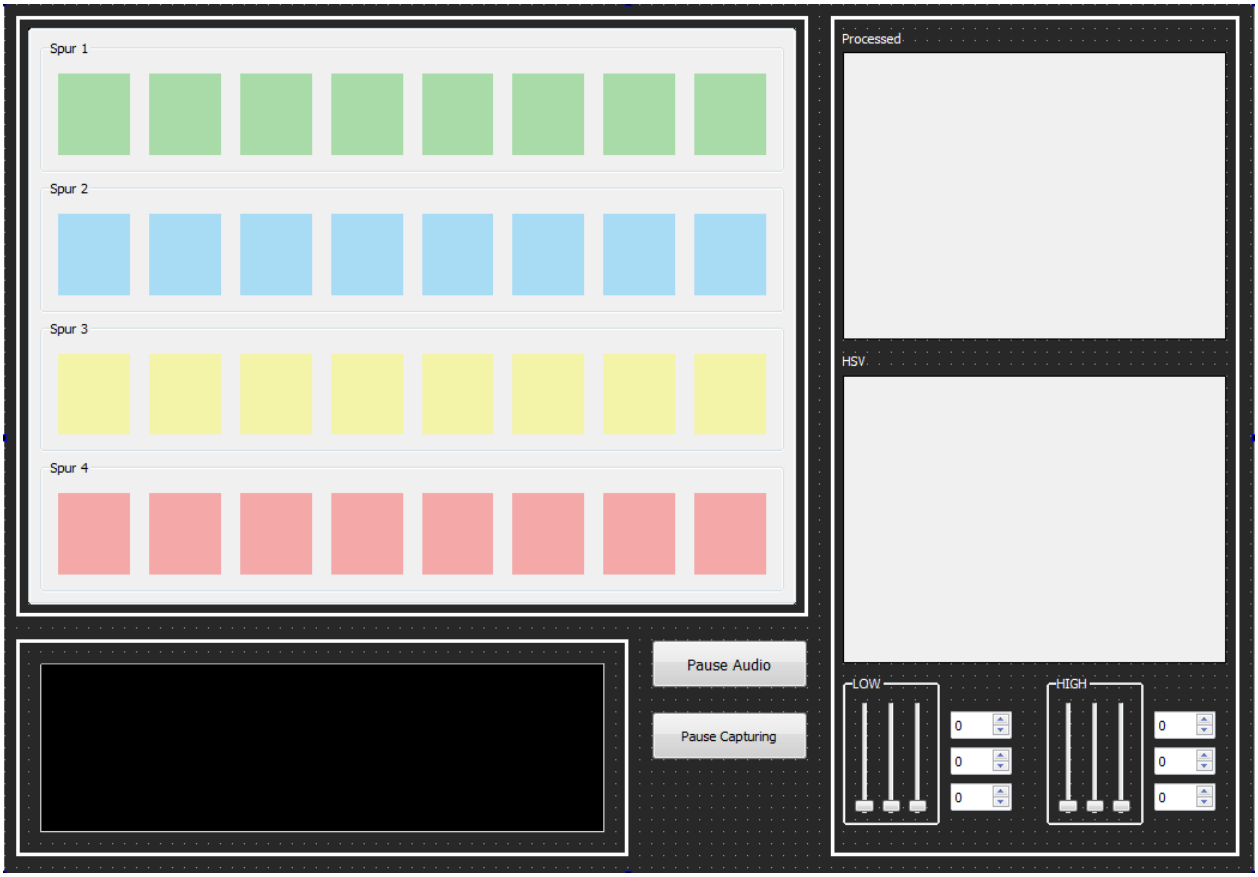

Abbildung 2: Umsetzung des Entwurfs im Qt Creator

Teilgruppe B: Programmierung

3 Entwicklungsumgebungen und Bibliotheken

3.1 Qt

Qt ist eine Multi-Plattform Anwendung und Framework. Der Name Qt enspringt dem damaligem Unternehmen Quasar Technologies. In dieser kann mit C++ und QML (ähnlich CSS und Javascript) entwickelt werden. Adaptionen anderer Programmiersprachen sind aber mittlerweile ebenfalls erhältlich. In der nicht-kommerziellen Entwicklung ist Qt frei verwendbar. Das Tool Qt Creator bietet zusätzlich die Möglichkeiten einer grafischen Oberfläche mit Baukasten. Qt bietet ebenfalls eine umfangreiche Klassenbibliothek. Downloads, Installationsanleitungen und Tutorials sind auf der Homepage zu finden.
http://qt-project.org/

3.2 OpenCV

OpenCV (CV = Computer Vision) ist eine Open-Source Programmbibliothek, die Algorithmen zur Bildverarbeitung bereitstellt. Diese bietet Möglichkeiten in der Filterbearbeitung, Objekt(wieder)erkennung, maschinelles Sehen und vieles mehr. Unterstützt werden dabei alle gängigen Systeme und die Programmiersprachen C, C++, Python und Java. Um OpenCV in der Programmierung nutzen zu können, muss eine Version für das richtige System geladen werden. Je nach System muss OpenCV entsprechend eingerichtet werden. Downloads, Installationsanleitungen und Tutorials sind auf der Homepage zu finden.
http://opencv.org/

3.3 SFML

SFML (Simple and Fast Multimedia Library) ist ein multi-Plattform, multi-Programmiersprachen, Open-Source Framework, dass Module für die Entwicklung in den Sparten System, Window, Graphics, Audio und Network bietet. In diesem Projekt wurde die Audiolibrary von SFML verwendet. Downloads, Installationsanleitungen und Tutorials sind auf der Homepage zu finden.
http://www.sfml-dev.org/

4 Klassen der Anwendung

4.1 Main

Die Klasse Main ist die reguläre Ausführungsklasse, in welcher die Objekte der Widgetklassen erzeugt und dargestellt werden. Diese wird automatisch erzeugt und muss nur bearbeitet werden, wenn die Anwendung mehrere Widgets gebraucht.

4.2 Dialog

Die Dialogklasse enthält alle wichtigen Methoden, Variablen und Objekte für die Darstellung und Funktion der Anwendung. Diese Klasse verhält sich wie folgt:

Konstruktor

- Initialisierung der Widgets, der Objekte, der Timer, des Capturing und dem User Interface mit der Methode initialize()

Destruktor

- Löschen der Widgets, der Objekte, der Timer und dem User Interface

processFrameAndUpdateGUI()

- Einlesen des Webcam-Bildes und die Verarbeitung der gelieferten RGB-Bilddaten
- Kopien des Original-Bildes im HSV Farbraum und binärer Darstellung
- Deklaration des zu trackendem Bildbereichs mit Positionsrückgabe
- Neupositionierung des Cursors mit Grenzen innerhalb der Anwendung
- Wischfunktionen für Start/Pause der Audiodateien
- Skalierungen der Bildcontainer

on_btnPauseResume_clicked()

- Start/Pause der Audiodateien durch ein Buttonevent

on_btnPRCapt_clicked()

- Start/Pause des Camera-Capturing durch ein Buttonevent

processAudio()

- Abspielen einer Sounddatei bei aktivem Button

rescale(Mat &mat)

- Skalierungsverhalten der Bildcontainer

on_*RGB*_valueChanged(int value)

- Änderungen der RGB-Low-Pass und RGB-High-Pass Werte

pairUiButtonstoArray()

- Initialisierung und speichern der User Interface Buttons in Arrays
- Jedes Array beinhaltet die Buttons einer Spur

timerStart()

- Initialisierung der Audio- und Videotimer

initSlider()

- Initialisierung der Slider

initialize()

- Initialisiert die grafische Benutzeroberfläche, das Capturing und die vorangeschriebenen Initialisierungs-Methoden

4.3 AudioEngine

Die Klasse AudioEngine implementiert die SFML-Bibliothek und liefert Methoden zum Einbinden, abspielen und loopen von Sounddateien. Unter Qt selbst bestehen ebenfalls Klassen für die Verarbeitung von Sounddateien. Diese lieferten jedoch keine zufriedenstellenden Ergebnisse, wenn mehrere Sounds gleichzeitig wiedergegeben werden sollen.

4.4 ModQPushButton

ModQPushButton ist eine an die Bedürfnisse der Anwendung modifizierte QPushButton-Klasse.
In dieser werden Methoden für die Bereichserkennung, die Verzögerung und Änderung des Status des Buttons geliefert. Erzeugte Objekte können in der grafischen Bearbeitung des Qt Creators, einfach mit dieser Klasse ersetzt (promoted) werden.

5 Erweiterungsmöglichkeiten

Da das Programm bisher ausschließlich auf einem Linux-System entwickelt wurde, sollte dies im nächsten Schritt Systemübergreifend erweitert werden.
Durch die vorgegebene Anzahl von Spuren und Samples steht dem Benutzer momentan nur eine geringe Auswahl an Kombinationen zur Verfügung.
Daher soll es zukünftig möglich sein, der Benutzeroberfläche weitere Spuren hinzuzufügen und während der Laufzeit Audiodateien durch neue zu ersetzen.
Zudem soll es dem Benutzer ermöglicht werden das zusammengestellte Musikstück aufzunehmen und speichern zu können. Dies könnte wie bisher über einen Button und eine zugehörige Wisch-Geste realisiert werden.
Um die Benutzeroberfläche schlanker zu gestalten wurde ebenfalls angedacht, den Konfigurationsbereich als separaten Menüpunkt auszugliedern.

6 Quellenverzeichnis

http://qt-project.org/
http://opencv.org/
http://www.sfml-dev.org/