Projektkonzept „Inverses Oztaskop“

Einleitung

„Haben Sie eine Vorstellung wie schön gute Musik aussieht und wie hässlich schlechte Musik ist?“ (Oztafan Kolibril, „Rumo und die Wunder im Dunkeln“ von Walter Moers)

Diesen schönen Satz wollen wir umdrehen und uns fragen: „Machen schöne Filme auch gute Musik?“. Um dies herauszufinden möchten wir ein Programm erstellen, welches Videoeingaben auswertet um eine Tonspur zu erzeugen, die auf Farb- und Helligkeitswerten aus dem Video basiert.

Dieses Projekt findet im Rahmen der Lehrveranstaltung Audio-Video-Programmierung statt. Im Laufe dieser Veranstaltung sollen wir einen Einblick in die Bildauswertung, Audioprogrammierung und maschinelles Lernen bekommen. Es war keine Vorgabe alle diese Themen im Projekt abzudecken, aber mit unserer Idee lassen sie sich gut miteinander kombinieren. Am Ende des Projektzeitraums würden wir gern eine Website präsentieren, auf der unsere Software läuft.

Wir haben unser Projekt „Inverses Oztaskop“ genannt. Dieser Name entstand in Anlehnung an das Oztaskop von Oztafan Kolibril aus Walter Moers‘ Buch „Rumo und die Wunder im Dunkeln“. Das Oztasktop ist ein Gerät, mit dem man unter anderem Musik sichtbar machen kann und da wir genau das Gegenteil machen wollen kam der Zusatz „invers“ dazu.

Projektziel

Das fertige Programm soll es Nutzern ermöglichen Videoclips in einer Maske hochzuladen. Die hochgeladenen Videos durchlaufen dann einen Berechnungsprozess und wenn dieser abgeschlossen ist, kann man sein Video mit neu generierter Audiospur auf der Website ansehen. Die generierte Audiospur entsteht aus den Bildinformationen des Inputvideos und soll dementsprechend dazu möglichst gut passen.

Bei der Abschlusspräsentation wollen wir ein Video live auswerten lassen und einige vorbereitete Beispiele zeigen um einen Eindruck zu bekommen, wie die Software auf unterschiedliche Eingaben reagiert.

Als Bonus wäre noch eine Visualisierung der generierten Audiospur denkbar.

Anforderungsanalyse

Das Programm soll Eingabevideos bezüglich der Farb- und Helligkeitswerte auswerten. Konkret geht es um das Unterteilen des Videos in Abschnitte. Diese Abschnitte werden in einem gewissen Intervall abgefragt um Informationen weiterzugeben.

Mit diesen Informationen sollen dann Berechnungen durchgeführt werden, die dafür sorgen, dass ein Modell mithilfe maschinellen Lernens Töne erzeugt. Zusammen sollen diese Töne dann eine Audiospur ergeben, die einen Musikcharakter hat. Es soll vermieden werden, dass der Zuhörer den Eindruck hat zufällige Töne zu hören. Deshalb ist es wichtig, dass abrupte Farbwechsel, ausgelöst durch harte Schnitte im Inputvideo, so verarbeitet werden, dass der Schnitt deutlich wird ohne „abgehackt“ zu klingen. Es soll eine gewisse Harmonie herrschen.

Zum Schluss soll das Eingabevideo mit der Audiospur synchronisiert werden und anschließend dem Nutzer auf der Website präsentiert werden.

Technische Rahmenbedingungen

Da wir beide mit unterschiedlichen Betriebssystemen arbeiten, versuchen wir unser Projekt möglich kompatibel mit MacOS und Windows zu halten.

Zur Bildauswertung werden wir die Bibliothek OpenCV verwenden. Diese wird von der Lehrveranstaltung vorgegeben, allerdings werden wir zum Zugriff Python statt C++ verwenden, da ein Teammitglied in dieser Sprache schon recht vertraut ist und schon erste Erfahrungen im Umgang mit OpenCV gesammelt hat.

Für den Audioteil werden wir auf die, aus der Vorlesung vorgegebene, Web Audio API zurückgreifen. Diese ist weit verbreitet und wir konnten in den ersten Lehrveranstaltungen schon einen guten Überblick über die Funktionen und Möglichkeiten erlangen.

Den Bereich des maschinellen Lernens wollen wir mit der JavaScript API Magenta.js abdecken. Diese Bibliothek wird uns in der Vorlesung noch nähergebracht. Nach kurzer Internetrecherche scheint dies die beste Wahl für unsere Anforderungen zu sein.

Die Voraussetzungen um das Endprodukt ausführen zu können, sind ein Rechner mit Webbrowser und Web Audio API Support.

Technisches Konzept

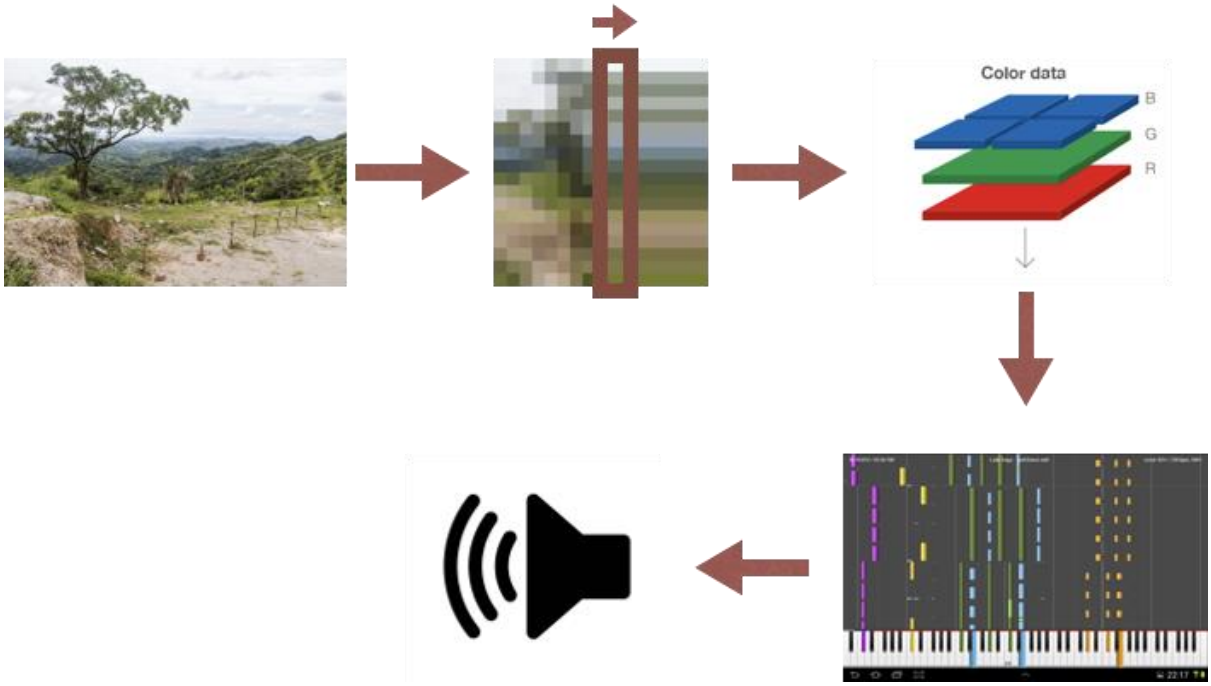

Abbildung 1: Programmablauf

Ein Nutzer lädt über eine Eingabemaske einen Videoclip auf die Website hoch. Dann teilen wir mittels Python und OpenCV die Frames des Inputvideos in Abschnitte auf. Über die Größe der Abschnitte bzw. die Auflösung sind wir uns noch nicht ganz klar. Aus diesen einzelnen Abschnitten der verschiedenen Frames ziehen wir dann vier Kanäle: Rot, Grün, Blau und Helligkeit. Anhand dieser werden dann für jeden Abschnitt die drei Werte: min, max und mean errechnet.

Dann muss man die so gewonnenen Daten aufbereiten, damit sie als Parameter für das Modell funktionieren, mit dem mittels maschinellen Lernen die Tonspur erzeugt wird. Man hat also einen Haufen an Farbdaten, die mit einer Transferfunktion in MIDI-Noten überführt werden sollen. Wie

dies genau aussehen könnte ist uns noch nicht ganz klar und wir hoffen in der nächsten Lehrveranstaltung konkretere Vorstellungen diesbezüglich zu bekommen. Außerdem haben wir einen Termin mit Herrn Sudau vereinbart um Hilfe zu erhalten, wie die Verbindung genau aussehen könnte. Im Moment stellen wir uns vor, das Modell entsprechend der mit Farbwerten verknüpften Emotionen zu trainieren. So wird beispielsweise ein freundlicher, ruhiger Song mit der Farbe Rot assoziiert. Es ist auch vorstellbar, dass die drei Farbkanäle einzelne Instrumente darstellen, die dann über die Helligkeitswerte variiert werden.

Die Farben könnten auch einen Einfluss auf die Verarbeitung in der Web Audio API haben. So könnten verschiedene Effekte und Filter mit Farben verknüpft werden um noch mehr Dynamik und Varianz in die Tonspur zu bringen.

Zum Schluss wird die erzeugte Tonspur entweder mit dem Videoclip synchronisiert abgespielt, oder die Tonspur wird dem Clip hinzugefügt und das Resultat dann wiedergegeben.

Bedienkonzept

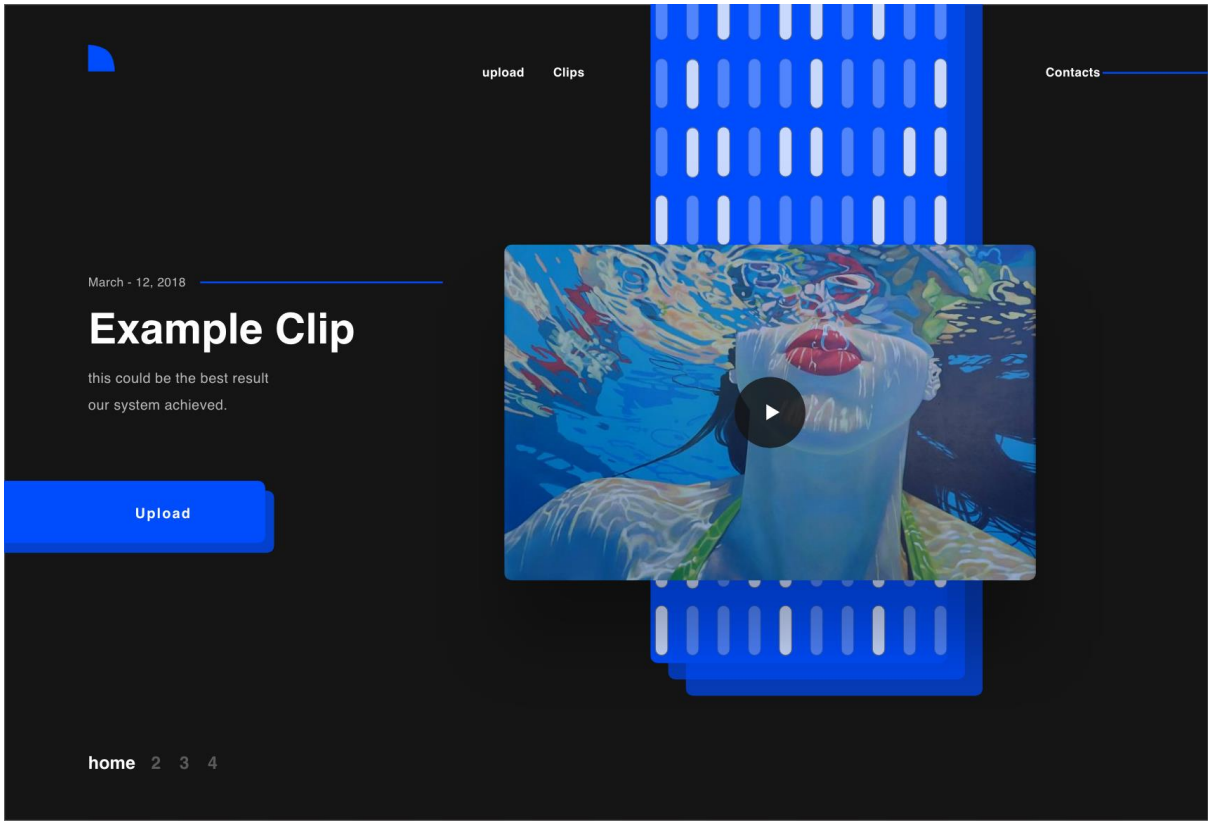

Abbildung 2: UI-Mockup (https://www.behance.net/gallery/63176771/Sierra-Free-Personal-Portfolio-Template)

Es wird eine Upload-Maske geben, über die der Nutzer sein Video über das Internet einstellen kann und einen Button um die Eingabe zu bestätigen. Weitere Einstellungen sind im Interface nicht vorgesehen.

Nachdem die Eingabe bestätigt wurde, beginnt das Programm mit der Auswertung und Berechnung. Der Nutzer soll hierüber informiert werden, beispielsweise mit einem Ladebalken oder einer Prozentanzeige.

Wenn die Berechnung abgeschlossen ist, kann man das Video mit der neuen Audiospur durch einen Klick darauf starten und auch wieder stoppen. Eventuell wird es um das Video herum eine Visualisierung des Tons geben.

Zeitplan

Der Zeitraum bis zur Projektvorstellung beträgt ungefähr sieben Wochen. Alle zwei Wochen wollen wir einen Meilenstein schaffen. Der erste Meilenstein wird die Bildverarbeitung und das Teilen in Abschnitte beeinhalten, konkreter gesagt: die Gewinnung der gewünschten Daten in der gewünschten Frequenz und im gewünschten Format. Parallel dazu läuft die Einarbeitung in maschinelles Lernen und das Anwenden auf den Musikkontext. Der zweite Meilenstein wird die Transferfunktion, also das Überführen der Daten aus dem Video auf die Musikerzeugung durch Magenta.js. Da dies voraussichtlich Probleme machen wird, definieren wir den zweiten Meilenstein so, dass es uns reicht eine grobe Funktionalität zu haben. Der dritte Meilenstein ist dann das feine Ausarbeiten der Überführung von Farbe zu Musik. Um diesen Punkt abhaken zu können, sollte schon ein Musikcharakter und eine gewisse Harmonie erkennbar sein. Zu diesem Meilenstein gehört auch das Ausgestalten des User Interface. Die letzte verbleibende Woche planen wir als Puffer und Gelegenheit zum generellen Aufräumen und Vervollständigen.

Teamplanung

Die ursprüngliche Idee zu diesem Projekt kommt von Christoph und deshalb ist er unser Projektleiter. Er wird hauptsächlich den Videoteil und grafische Aufgaben, wie das Gestalten des User Interface, übernehmen. Trotzdem wird er auch an der Überführung der gewonnenen Informationen in Musik mithelfen.

Das zweite Teammitglied ist Tim, der sich fast ausschließlich mit dem Audioteil und dem maschinellen Lernen beschäftigen wird.

Aufgabe	Christoph	Tim	Personenstunden
Bildverarbeitung	x		15
Schnittstelle Python JavaScript	x	x	3
HTML Dummy		x	2
Einarbeitung maschinelles Lernen	x	x	10
Transferfunktion	x	x	30
Verfeinern der Transferfunktion		x	25
User Interface	x		15
Aufräumen	x	x	10

Tim Fock (2270398)
Christoph Mührke (2278652)