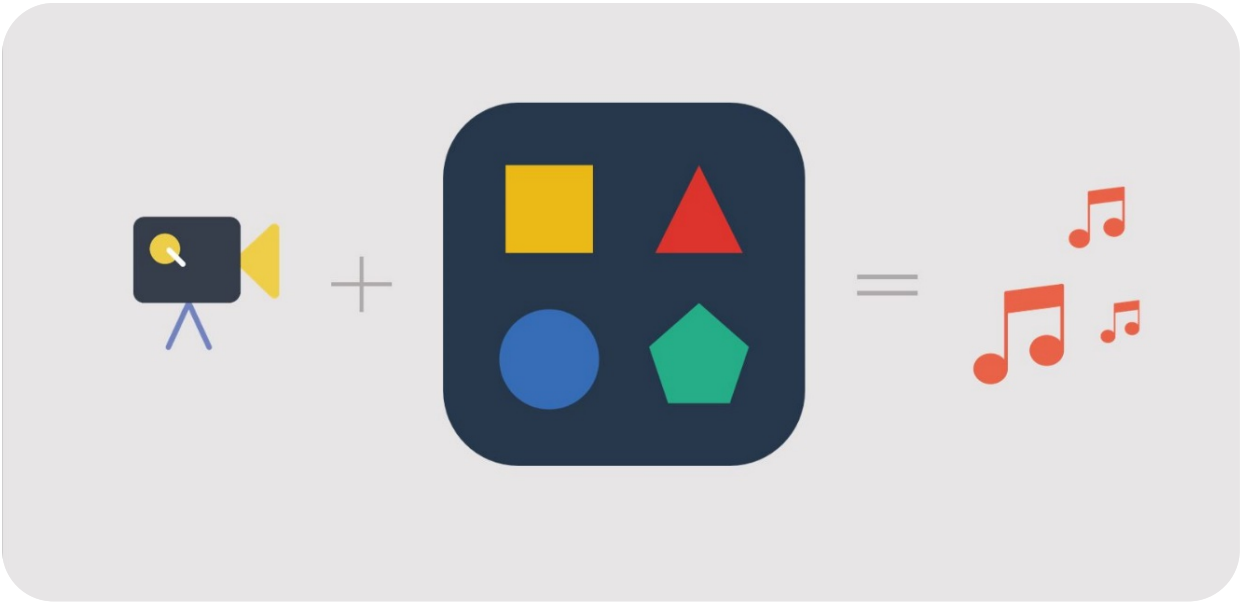


Projektkonzept

Formenklang

für den Kurs Audio-Video-Programmierung
bei Prof. Dr. Andreas Plaß & Jakob Sudau
von Nadine Krietenbrink (2322554) und Caroline Wolf (2320493)

Einleitung

Für das Wahlpflichtfach Audio-Video-Programmierung für die Bachelor-Studiengänge Media Systems und Medientechnik des Departments Medientechnik der HAW Hamburg ist als Prüfungsleistung ein Projekt im Umfang von 78 Stunden pro Person abzugeben.

Die Lehrveranstaltung ist in die Bereiche Audio und Video aufgeteilt. Inhalt des Video-Teils ist die Videoverarbeitung in der Programmiersprache C++ mit OpenCV und der Software QT. Der Audio-Teil befasst sich mit der Web-Audio-API und Programmierung in JavaScript sowie Gestaltung von Web-Oberflächen mit HTML und CSS.

Ziel der Lehrveranstaltung sind der Einstieg in die Programmiersprache C++ sowie das Verständnis und die Anwendung von Algorithmen der Video- und Audiobearbeitung.

Das Projekt kann sich sowohl nur mit Audio- oder Video-Programmierung als auch mit einer Kombination aus beiden Bereichen befassen. Im Rahmen einer kleinen Ausstellung im Gebäude der HAW sollen die Projekte der Studierenden vorgestellt werden.

Unser Projekt „Formenklang“ ist Teil dieser Ausstellung. Es befasst sich sowohl mit Videoverarbeitung mit OpenCV als auch der Audiobearbeitung mit Hilfe der Web-Audio-API. Wir haben uns entschieden beide Bereiche zu nutzen, um eine spannende Verbindung zwischen Video und Audio herstellen zu können. Es soll das Interesse des Nutzers wecken mit Formen und Farben Sound zu erzeugen.

Projektziel

Ziel des Projektes ist, dass der Benutzer mit Hilfe von Formen und Farben eine Audio-Sequenz kreiert.

Der Benutzer kann mit den ausgedruckten Formen ein Muster in einem bestimmten Bereich legen. Dieser Bereich wird von einer Kamera aufgenommen und durch eine x- und y-Achse definiert.
Die x-Achse des Bereichs stellt die Zeit dar. Die abgelegten Formen werden entsprechend ihrer Position von links nach rechts abgespielt. Die Länge einer Audio-Sequenz wird darüber definiert und von uns festgelegt.
Die y-Achse dient als Lautstärkeregler. Platziert man die Formen im unteren Bereich, werden die jeweiligen Sounds der Formen leiser abgespielt. Platziert man diese im oberen Bereich, werden die Töne entsprechend lauter.

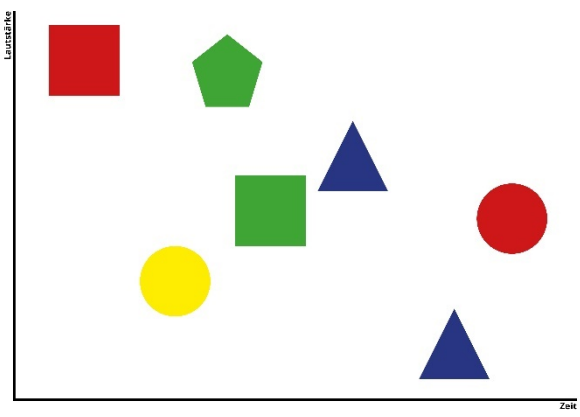


Abbildung 1: Koordinatensystem mit Formen

Die einzelnen Formen symbolisieren Instrumente einer Band. Das heißt zum Beispiel, dass bei einem Dreieck ein Sound von einer Gitarre zu hören sein wird. Zusätzlich werden die Formen in verschiedenen Farben dargestellt. Die Farben definieren die unterschiedlichen Sounds eines Instrumentes, womit man mehr Varianz in seine Audio-Sequenz bringen kann.

Hat der Benutzer seine Formen abgelegt, kann er das Tracken der Kamera durch einen Start-Button aktivieren. Der Text des Buttons ändert sich und dient anschließend als Stopp-Button. Ab dann werden die Daten an den Webbrowser geschickt und die Audio-Sequenz wird in Dauerschleife abgespielt. Die einzelnen Sound-Elemente werden im Browser visuell dargestellt. Wenn ein Sound

abgespielt wird, erscheint die Form und die Farbe in einem Bereich des Browsers, pulsiert und verblasst anschließend wieder. Zusätzlich kann die Audio-Sequenz mittels Schiebereglern optimiert werden.

Während des Trackings kann der Benutzer die Position der Formen ändern, Formen entfernen und weitere Formen hinzufügen. Durch das Tracking werden die neuen Daten an den Browser geschickt und die Audio-Sequenz entsprechend angepasst.

Hat der Benutzer eine Audio-Sequenz erstellt, die ihm gefällt, kann er diese herunterladen.

Anforderungsanalyse

Funktionale Anforderungen

C++-Programm
Die Software …
- muss eine Verbindung mit der Kamera aufbauen.
- muss das Kamerabild auf der Programmoberfläche darstellen.
- muss Formen innerhalb des Kamerabildes erkennen.
- muss Farben erkennen und diese den Formen zuordnen können.
- muss die Position der einzelnen Formen berechnen.
- muss für jede mögliche Farb-Form-Kombination einen individuellen Wert berechnen.
- muss die x-Koordinate und die y-Koordinate jeweils in einen 1 Byte Wert umrechnen.
- muss aus allen drei Informationen einen individuellen 3-Byte-Datensatz zusammensetzen.
- muss MIDI-Datensätze erstellen können.
- muss den Datensatz als MIDI-Datensatz übertragen.
- startet erst die Übertragen, wenn der Play-Button geklickt wurde.
- ändert den Text des Buttons nach einem Klick.
- stoppt die Übertragung, wenn der Play-Button erneut geklickt wurde.

Webbrowser
Die Software …
- muss die Sound-Dateien laden und speichern.
- zeigt einen Platzhaltertext an, solange keine Übertragung von MIDI-Daten stattfindet.
- muss MIDI-Datensätze empfangen können.
- muss die Datensätze analysieren können, d.h. den Werten einen jeweiligen Sound mit einer entsprechenden Lautstärke zuordnen.
- muss Audios in einem bestimmten Zeitablauf im Browser abspielen.
- muss zu jedem Sound Formen im Browser darstellen können.
- muss den Formen eine Animation zuweisen.
- muss Schieberegler darstellen und bei einem Input die Regler auswerten.
- Stellt einen Download-Button dar, der bei Betätigung die aktuelle Audio-Sequenz herunterlädt.

Nichtfunktionale Anforderungen

- **Wartbarkeit**:
  Die Software soll änderungsfähig sein. Das heißt, dass die Implementierung von Korrekturen, Verbesserungen und Anpassungen ohne großen Aufwand möglich ist.

- **Erweiterbarkeit**:
  Die Software kann nach Abschluss des Projektes durch Erweiterungen modifiziert werden, um eventuellen neuen Anforderungen gerecht zu werden. Die Erweiterbarkeit erfolgt dabei durch Hinzufügen von Klassen, nicht aber durch Änderung bereits bestehender Klassen.
- **Zuverlässigkeit**:
  Das Programm funktioniert fehlerfrei über einen Video-Aufnahme-Zeitraum von 15 Minuten. Es können bis zu 20 Formen gleichzeitig gelegt sein, ohne dass Probleme bei der Speicherkapazität auftreten.
- **Benutzbarkeit**:
  Die Bedienung der Software wird vom Benutzer innerhalb von 5 Minuten verstanden und erlernt.
- **Schnelligkeit**:
  Die Übertragung der MIDI-Datensätze soll innerhalb von einer Sekunde stattgefunden haben.

Technische Rahmenbedingungen

Die Umsetzung findet auf Rechnern mit Windows-Betriebssystemen statt. Das Betriebssystem ist im späteren Ausstellungsraum vorhanden und bietet sich daher an.

Für die Video-Aufnahme wird eine Webcam an den Computer angeschlossen. Die Auflösung der Webcam muss so hoch sein, dass Farben und geometrische Formen genau voneinander zu unterscheiden sind.

Die zu verwendende Software ist durch die Vorlesungsinhalte bereits vorgegeben.
Für den Videoteil verwenden wir das Programm QT mit der Programmiersprache C++ und der Bibliothek OpenCV für Bildbearbeitung. Als Alternative zu C++ bietet sich Python an, welches ebenfalls mit OpenCV verwendbar wäre. Für C++ haben wir uns entschieden, um unsere Kenntnisse in dieser, für uns neuen Programmiersprache, zu verbessern.

QT ist eine Programm-Bibliothek zur plattformübergreifenden Entwicklung von Programmen sowie grafischer Benutzeroberflächen. OpenCV (CV = Computer Vision) ist eine Programmbibliothek mit Algorithmen für die Bildverarbeitung und maschinelles Sehen. Sie unterstützt die Programmiersprache C++ und lässt sich leicht in QT integrieren.

Für den Audioteil benutzen wir die Web-Audio-API mit JavaScript, HTML und CSS.
Die Web-Audio-API ermöglicht es in Webbrowsern mit JavaScript Audio-Signale zu erzeugen und Audio-Bearbeitungen durchzuführen. Durch HTML und CSS werden Regler für die Bearbeitung dieser Signale im Browser zur Verfügung gestellt.

Für die Datenübertragung zwischen QT und dem Web-Browser wird MIDI (Musical Instrument Digital Interface) verwendet. MIDI ist ein Industriestandard für den Austausch musikalischer Steuerinformationen zwischen elektronischen Instrumenten. Er beinhaltet unter anderem das Kommunikationsprotokoll für die zu übermittelnden Daten zwischen Instrument und Computer, welches wir nutzen werden.

Um MIDI an QT anzubinden, verwenden wir die open-source-Bibliothek Drumstick und den virtuellen MIDI-Treiber LoopBe. Drumstick stellt Input- und Output-Klassen für MIDI in QT zur Verfügung. LoopBe nutzen wir, um einen MIDI-Ausgang von QT mit dem MIDI-Eingang der Web-Audi-API im Browser zu verbinden. Die Weiterleitung geschieht in Echtzeit.

Die Versionskontrolle erfolgt mittels Git über GitHub.

Technisches Konzept

Die in der Vorlesung bereits verwendeten Klassen werden für dieses Projekt genutzt. Dazu gehören die VideoPlayer-Klassen, die ColorKeyer-Klassen und die MIDI-Klassen.

Um die Formen der einzelnen Kamerabilder zu erkennen, wird das Bild in ein binäres Bild umgewandelt. Mithilfe der Funktion „cvFindContours“ werden alle Konturen erkannt und gespeichert. Die Funktion „cvApproxPoly“ erkennt Formen und speichert Vertices für jede Kontur ein. Durch die Anzahl der Vertices können die Formen bestimmt werden. Ein Dreieck besteht zum Beispiel aus drei Vertices, ein Quadrat aus vier. Wichtig hierbei ist es, nur Formen zu nehmen mit einer ungleichen Anzahl von Vertices.

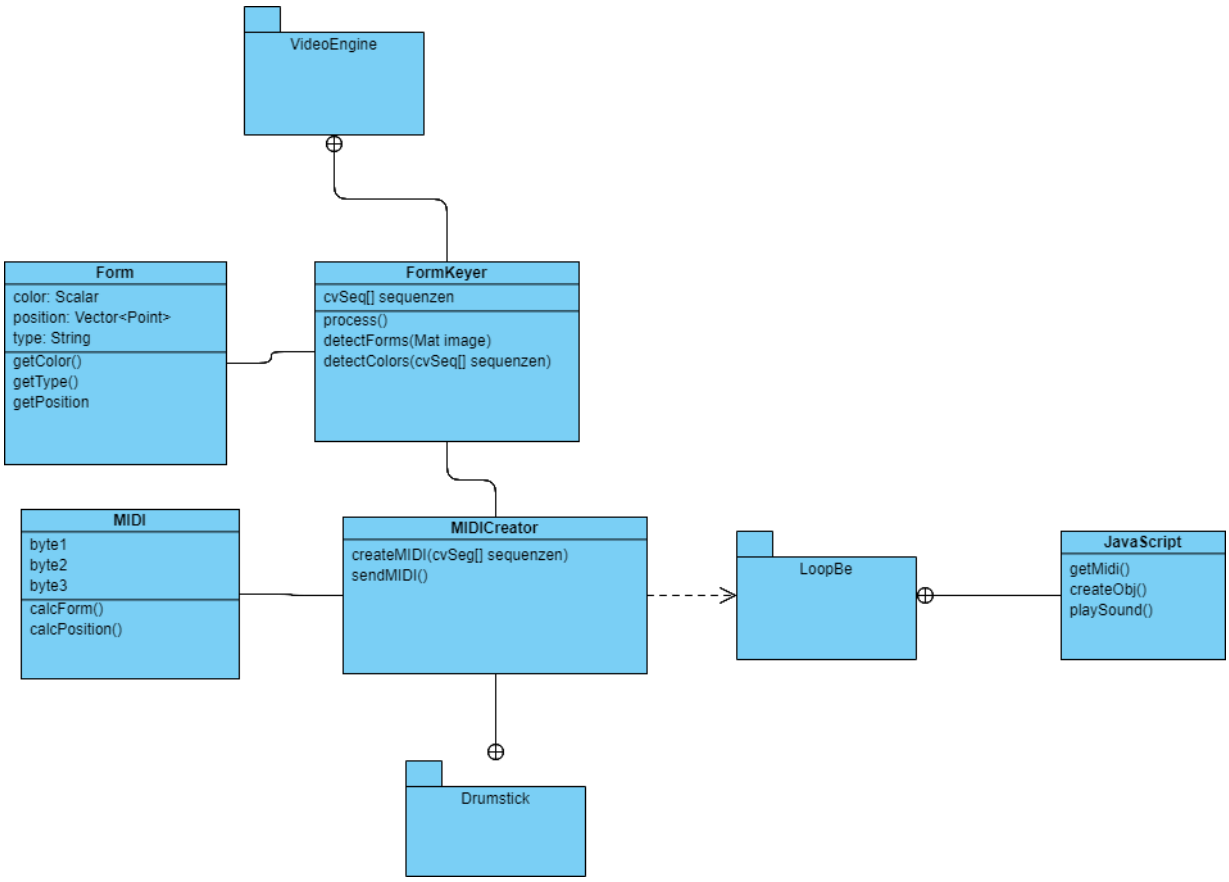
Die Farbe der Formen wird durch die ColorKeyer-Klasse erkannt.

Um die Position der Formen herauszubekommen, wird pro gefundene Kontur ein Mittelwert der Pixel berechnet. Diese Daten sollen die Koordinaten repräsentieren.

Hat das Programm die Formen, die Farben und die Position erkannt, wird aus diesen Daten für jede Form ein MIDI-Datensatz erstellt. Das erste Byte wird für die Form und die Farbe verwendet. Der Wert wird entsprechend der folgenden Tabelle berechnet:

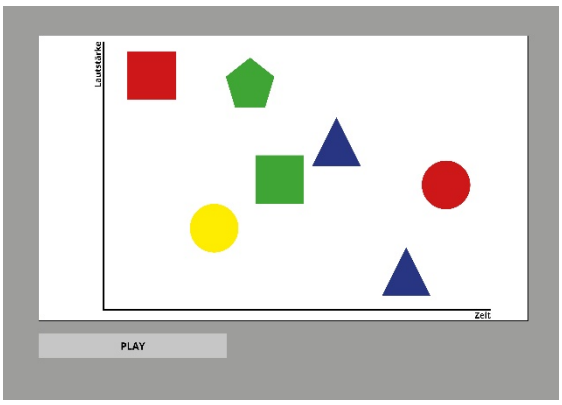
	Wert der Form	**Wer der Form + 1 für Rot**	**Wer der Form + 2 für Blau**	**Wer der Form + 3 für Grün**
Kreis	10	11	12	13
Dreieck	20	21	22	23
Quadrat	30	31	32	33
Pentagramm	40	41	42	43
Beispiel: blaues Quadrat = 30 + 2 = 32				

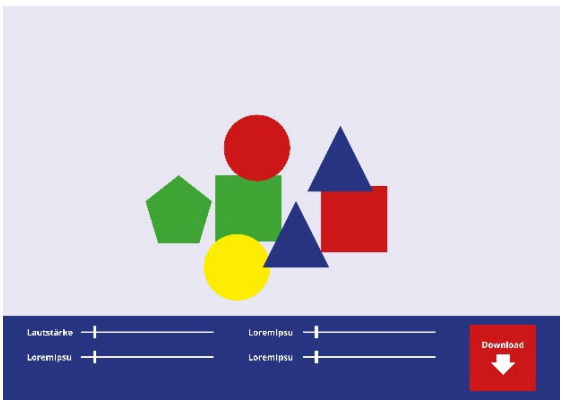
Das zweite und das dritte Byte wird mit den x- und y-Koordinaten der Form belegt. Dies muss entsprechend auf 255 skaliert werden, damit die Zahl in das Byte hineinpasst.
Die MIDI-Daten werden gesendet, gehen über LoopBe und werden von dem Browser als MIDI-Daten erkannt. Das Script muss nun diese Daten erkennen und entsprechend wieder umwandeln.
Das erste Byte wird entsprechend einer Sound-Datei zugeordnet. Die Koordinaten werden als Lautstärke und zum Timing genutzt.



Bedienkonzept

Das Endprodukt wird aus zwei Oberflächen bestehen. Einmal das C++ Programm und einmal die Browser-Ansicht.
Die Oberfläche des C++ Programmes stellt das Kamerabild dar und einen Button. Durch diesen Button lässt sich das Tracking starten und stoppen.
Die Oberfläche des Browsers wird ähnlich simpel gehalten. Die Animation der Sound-Elemente wird flächendeckend dargestellt. Im unteren Bereich des Browsers ist ein Bereich definiert, in dem die Slider und Buttons zur Optimierung des Sounds dargestellt werden. Dieser Bereich nimmt ca. ¼ der Fenster-Höhe ein. Eingestellt werden können Lautstärke, Nachhall (Reverb) und Frequenz.





Zeitplan

Geschätzter Aufwand

Aufgabe	Zeit (in Personenstunden)
Ideenentwicklung	5
Konzept schreiben	10
Arbeitsplatz einrichten mit entsprechender Software	8
Einarbeitung in C++ und JavaScript mit Hilfe von Tutorials	20
Programmierung C++	45
Programmierung JavaScript	25
Layout Design für Web-Browser & QT Oberfläche	15
Herstellung von Lege-Material für Video	2
Qualitätssicherung + Testing	8
Projektmanagement	25
Projektbericht	10
Gesamt	173

Meilensteine

	Oktober		November				Dezember			
	15. - 21.10	22. - 28.10	29.10 - 4.11	5. - 11.11	12. - 18.11	19. - 25.11	26.11 - 2.12	3. - 9.12	10. - 16.12	17. - 23.12
	3. Woche	**4. Woche**	**5. Woche**	**6. Woche**	**7. Woche**	**8. Woche**	**9. Woche**	**10. Woche**	**11. Woche**	**12. Woche**
Ideenfindung										
Planung/Konzept			Präse Konzept							
Tools einrichten										
Design										
Programmierung									Puffer	Abgabe 18.12
Testphase										
Abschlussreflexion										
Projektbericht										

Ideenfindung
Brainstorming und Ideenfindung soll bis 21.10.2018 abgeschlossen und in der Gruppe besprochen worden sein.

Planung/Konzept
Das Konzept soll am 30.10.2018 präsentiert werden und muss bis 29.10.2018 fertig erstellt sein.

Tools einrichten
Die nötigen Softwareinstallationen von QT und Implementierung der Libraries sowie Einrichtung der Versions-Kontrolle mittels Git soll bis 4.11.2018 beendet sein.

Design
Das Design für die QT-Oberfläche und das Layout für den Web-Browser soll bis 4.11 fertig sein. Außerdem sollen die Bastelarbeiten für das Legen der geometrischen Elemente abgeschlossen sein.

Programmierung
Die Programmierung der Video-Software und der Webseite wird sich teils überschneiden. Zuerst soll die grundlegende Funktionalität in C++ stehen, damit MIDI-Daten übermittelt werden können. Dies soll bis zum 18.11.2018 erfolgen. Erst dann startet die Entwicklung der Webseite.
Für den Fall der Verzögerung des Projektes steht ein Puffer von einer Woche zur Verfügung, der Notfalls für die Programmierung verwendet werden kann.

Testphase
Um Fehler zu finden und zu beheben gibt es am Ende der Programmierphase eine Testphase bis 9.12.2018.

Präsentation & Puffer
Am 18.12.2018 werden die Projekte im Rahmen der Vorlesung präsentiert.

Abschlussreflexion & Projektbericht
Nach der Präsentation wird das Team sich zusammensetzen, um über den Projektverlauf zu sprechen. Hier werden sowohl negative als auch positive Aspekte diskutiert und es wird ein Verbesserungsvorschlag für zukünftige Projekte ausgearbeitet. Anschließend wird der Projektbericht geschrieben.

Teamplanung

Teammitglieder sind Nadine Krietenbrink und Caroline Wolf.

Da das Team nur aus zwei Beteiligten besteht, gibt es keinen konkreten Projektleiter. Entscheidungen werden zusammen und nach gemeinsamer Absprache getroffen.

Frau Krietenbrink übernimmt die Programmierung des C++-Teils. Frau Wolf wird die Webseite umsetzen und das Material vorbereiten.

Alle weiteren Aufgaben werden kurzfristig verteilt oder gemeinsam bearbeitet.