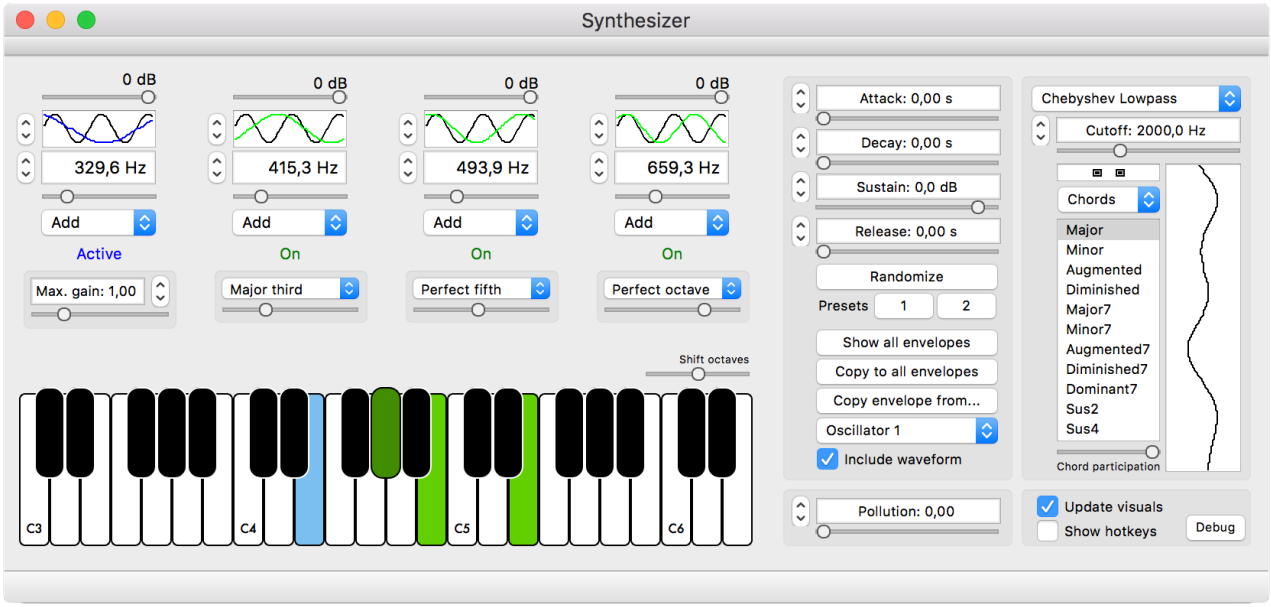


FM-SYNTHESIZER

mit einstellbaren Parametern via Gesichtserkennung

Kurs: AV-Programmierung

Erstellt von: Kevin Detter (2148745), Robert Krockert (2093487), Arne Sibilis (2203514)

28. Januar 2017

AUFGABE

Als Prüfungsleistung für den Kurs AV-Programmierung sollte während des laufenden Semesters ein *Facial Expression Audio Synthesizer* entwickelt werden. Den Abschluss des Kurses bildete eine Messe-ähnliche Ausstellung am 24. Januar 2017, an der sowohl andere Studierende als auch Vertreter verschiedener IT-Firmen (wie z.B. Steinberg oder Cellular) teilnahmen.
Die Basis des von uns zu entwickelnden Programms bildet ein traditioneller FM-Synthesizer, der mit Hilfe eines MIDI-Controllers bedient bzw. gespielt werden kann. Die Effekte und Filter, die den typischen Sound eines Synthesizer maßgeblich ausmachen, werden aber nicht wie herkömmlicherweise durch so genannte Pitch- oder Mod-Wheels (Räder zur Variation der Tonhöhe bzw. zur Effekt/Filter- Modulation) sondern intuitiv über eine Kamera gesteuert, die per Gesichtserkennungs-Algorithmus die Mimik des Benutzers auswertet.

INSTALLATIONSANLEITUNG

Da wir unser Programm unter MacOS entwickelt und es bisher auch nur auf diesem Betriebssystem getestet haben, beschränken wir uns in dieser Dokumentation auf die Installation unter Mac. Das Programm *sollte* jedoch ebenfalls problemlos unter Windows ausführbar sein.
Der Ordner enthält eine ausführbare Datei. Um diese starten zu können, bedarf es allerdings noch der Installation der Qt Entwicklungsumgebung sowie der Open Source Library *OpenCV*.

Installation Qt

A. Herunterladen des kostenlosen Online Installer von der Qt Website unter https://www.qt.io/download-open-source/
B. Ausführen der heruntergeladen Datei und den Anweisungen folgen

Installation OpenCV

A. Installation von Homebrew über die Kommandozeile: /usr/bin/ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)“ => offizielle Seite: http://brew.sh/
B. Installation von OpenCV Version 3.X mittels Homebrew: Terminal => brew install opencv3
C. Einbinden der OpenCV-Bibliothek in QT:
   - Finden des Pfad von pkg-config: which pkg-config
   - Zu QT wechseln, Projekt in der linken Seitenleiste auswählen und bei Build-Umgebung bei Variablen „PATH“ den „WERT“ bearbeiten und den gefundenen Pfad aus Schritt 3.1. hinzufügen.
   - find /usr/local -name „opencv.pc“

- Zu QT wechseln, Projekt in der linken Seitenleiste auswählen und bei Build-Umgebung eine neue Variable namens „PKG_CONFIG_PATH“ anlegen und als Wert den gefunden Pfad aus Schritt 3.3. einfügen.
- Zur .pro-Datei des Projektes wechseln und folgende Zeilen hinzufügen:

```
LIBS += -L/usr/local/lib -L/usr/local/Cellar/
opencv3/3.1.0_4/share/OpenCV/3rdparty/lib/
QT_CONFIG -= no-pkg-config
CONFIG += link_pkgconfig
PKGCONFIG += opencv
```

- bei LIBS ist der Pfad anzugeben, wo sich der „lib“-Ordner der installierten openCV- Bibliothek befindet.

D. Wenn OS X „El Capitan“ in Verwendung ist, dann muss folgender Schritt zusätzlich ausgeführt werden:
   - In dem Reiter „Projekt“ in der linken Seitenleiste auf „Ausführen“ wechseln und bei der „Ausführungsumgebung“ den Wert von der Variablen „DYLD_LIBRARY_PATH“ leeren.

E. Falls die entsprechenden Haar-Cascades nicht schon im Projekt vorhanden sind, dann müssen diese noch entpackt und dem Projekt hinzugefügt werden.
   - Gegebenenfalls muss in der Detect.hpp Klasse bei den Variablen für die Cascade-Names der Pfad angepasst werden - je nach dem wo die Datei abgespeichert wurde.

BEDIENUNGSANLEITUNG

Hardware

Für die Nutzung des Programms ist ausser einer Webcam und einem Computer keinerlei Hardware von Nöten. Falls ein MIDI-Keyboard vorhanden ist, erleichtert das natürlich die Eingabe, allerdings kann der Synthesizer auch über die Computertastatur oder mit einer Maus bedient werden.

Für eine problemlose Gesichtserkennung empfehlen wir neben einer gleichmäßig hellen Beleuchtung einen einfarbigen Hintergrund (z.B. eine weiße Tapete). Da sich unsere Software auf die Augen des Users konzentriert, sollten Brillenträger auf das Tragen ihrer Brille verzichten, da Reflexionen in dieser zu Problemen führen könnten. Darüber hinaus sollte sich jeweils nur ein Benutzer zur Zeit vor der Kamera befinden.

Gesichtserkennung

Durch das Blinzeln der Augen lassen sich folgende Parameter variieren:

- **Random:** Es lassen sich zufällige Hüllkurven erzeugen

- **Chord:** Bei geöffneten Augen wird über dem gespielten Grundton ein Dur-Akkord erzeugt, bei geschlossenen Augen ein Moll-Akkord
- **Volume:** Bei geöffneten Augen erhöht sich die Lautstärke, bei geschlossen verringert sie sich
- **Modulate:** Bei geöffneten Augen wird die Frequenz und der Index des Frequenzmodulators ständig verändert

User Interface

Da es insgesamt sehr viele einstellbare Parameter gibt, haben wir ein Schaubild erstellt, das jede einzelne Funktion des GUI kurz erläutert:

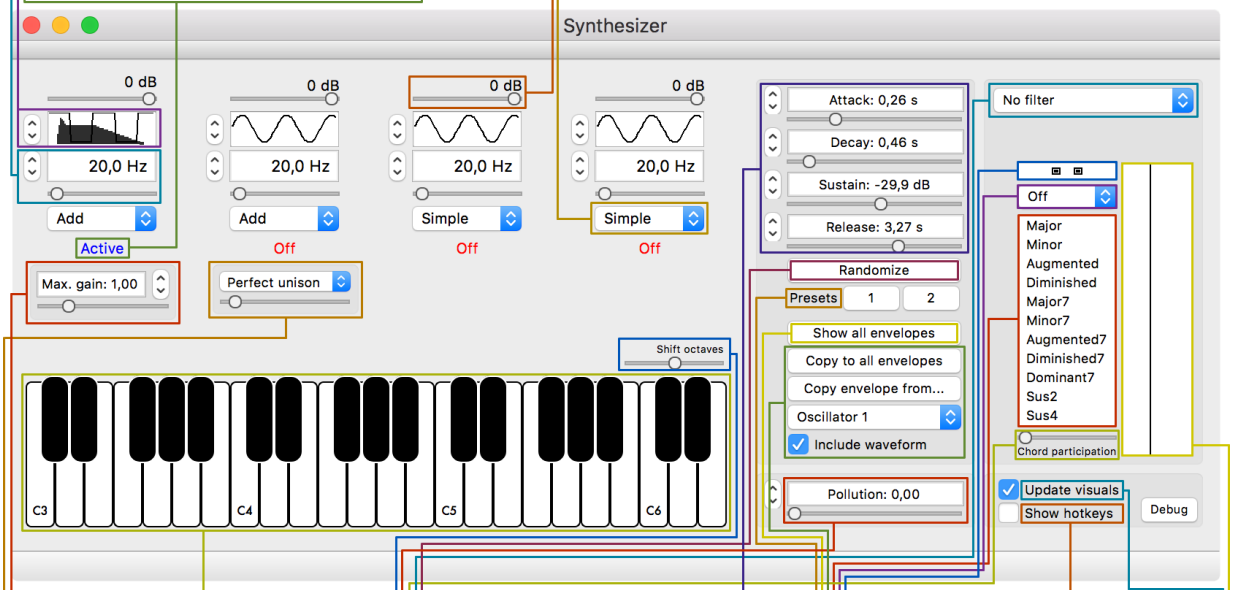
Frequenz
Zeigt die aktuelle Frequenz des Oszillators an. Kann manuell eingestellt werden, um präzise modulierte oder addierte Synthesen zu erzeugen.

Verstärkung
Legt die Verstärkung bzw. Dämpfung der Ausgabe des jeweiligen Oszillators fest.

Wellenform
Bestimmt die Wellenform, die der Oszillator erzeugt.

Typ
Wählt aus, in welcher Funktion der Oszillator eingesetzt werden soll. 'Simple' liefert das unverfälschte Signal des aktiven Oszillators. 'Add' erzeugt das zusammenaddierte und normalisierte Signal aller Oszillatoren dieses Typs und 'Carrier' sowie 'Modulator' ergeben gemeinsam eine FM-Synthese. Bei gleichzeitiger Betätigung mehrerer Tasten agiert ein einzelner Oszillator darüber hinaus wie eine Gruppe aus Oszillatoren des Typs 'Add'.

Oszillator: Status
Zeigt den Status des Oszillators an. Durch Anklicken wird der Oszillator auf aktiv gesetzt.



Klaviatur
Die Tasten können durch Klicken, Betätigung der entsprechenden Hotkeys oder eingehende MIDI-Signale gesteuert werden.

Verstärkungsfaktor / Modulationsindex
Ist eine FM-Synthese oder Frequenzaddition aktiv, erscheint ein Regler unter dem aktiven Oszillator, mit dem entweder der Verstärkungsfaktor (der mit dem Ausgangssignal mulitpliziert wird) oder der Modulationsindex bestimmt werden kann.

Intervall
Ist dem aktiven Oszillator mindestens ein weiterer Oszillator des Typs 'Add' zugeschaltet, kann hier ein festes Intervall gewählt werden, um den Oszillator immer in Abhängigkeit vom Grundton auf eine entsprechende Tonhöhe einzustellen. Alternativ kann diese automatische Schaltung mit der Option 'Off' deaktiviert werden.

Sichtbarer Teil der Klaviatur
Mit diesem Regler kann die virtuelle Klaviatur in Schritten von je einer Oktave in beide Richtungen verschoben werden.

"Verunreinigung"
Wird dieser Regler auf einen Wert über Null gesetzt, werden die Ausgabewerte des gewählten Oszillators durch addierte Zufallswerte geringfügig modifiziert, so dass zum eigentlichen Ton ein Rauschen hinzugefügt wird. Hiermit lassen sich perkussive Effekte erzeugen.

Akkorde: Beteiligung
Mit diesem Regler können schnell und einfach mehrere Oszillatoren auf den Typ 'Add' gestellt werden. Der vorige Typ wird gespeichert, so dass er wiederhergestellt werden kann, wenn der Regler wieder heruntergestellt wird.

Hüllkurve
Mit den vier Reglern für die ADSR-Werte lassen sich beliebige Hüllkurven für den aktiven Oszillator realisieren.

Zufällige Hüllkurve
Erzeugt eine Hüllkurve für den aktiven Oszillator aus zufällig generierten Werten.

Hüllkurven: Voreinstellungen
Ordnet dem aktiven Oszillator eine von zwei fest eingestellten Hüllkurven zu.

Hüllkurven: Alle anzeigen
Blendet die grafischen Repräsentationen der Hüllkurven aller Oszillatoren für einige Sekunden ein (sofern bereits eine Hüllkurve initialisiert wurde).

Hüllkurven: Kopieren
Der Button 'Copy to all envelopes' kopiert die Hüllkurve des aktiven Oszillators auf alle übrigen Oszillatoren und überschreibt dabei deren bisherige Einstellungen. Mit dem Button 'Copy envelope from...' dagegen wird die Hüllkurve des aktiven Oszillators durch jene des im Feld darunter gewählten Oszillators (von links nach rechts: 0 bis 3) ersetzt. Ist der Haken bei 'Include waveform' gesetzt, wird dabei jeweils auch die Art der eingestellten Wellenform kopiert.

Filter
Wählt den Typ des Audiofilters aus, der auf den jeweiligen Oszillator angewandt werden soll. Mit einem eingeblendeten Schieberegel kann die Grenzfrequenz eingestellt werden.

Audiosignal
Zeigt eine grafische Repräsentation des aktuell verarbeiteten Audiosignals, das sich aus den Einzelsignalen der beteiligten Oszillatoren errechnet.

Anzeige von Hotkeys
Zeigt oder versteckt die Hotkeys zur Bedienung der virtuellen Klaviatur auf den jeweiligen Tasten.

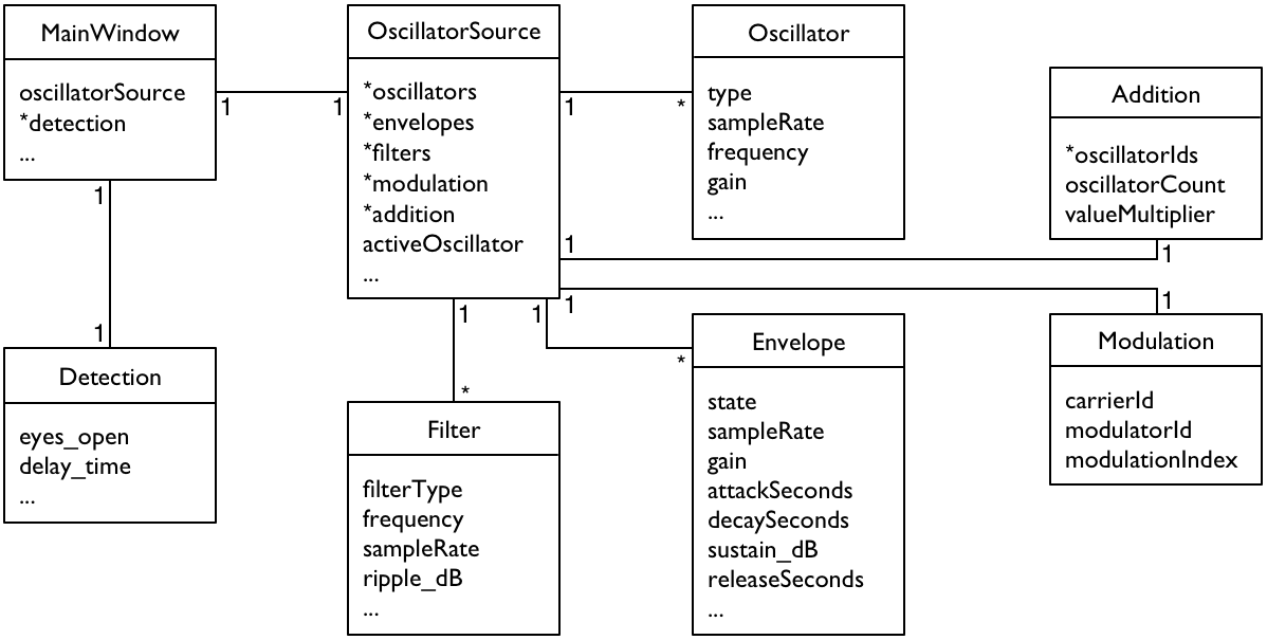
Aktualisierung von Grafiken
Durch Deaktivierung werden Repräsentationen von Wellen und Hüllkurven sowie der Status der Oszillatoren nicht mehr angezeigt. Hierdurch können ggf. Störgeräusche reduziert werden.

Augenerkennung: Kontrollanzeige
Zeigt den Status der Augenerkennung an. Konnte keine Verbindung zur Kamera hergestellt werden, ist dieses Feld durchgestrichen.

Augenerkennung: Steuerungsoption
Legt fest, welche Funktionen des Synthesizers durch die Augen gesteuert werden sollen. 'Random' erzeugt auf dem aktiven Oszillator eine zufällige neue Hüllkurve, wenn ein Blinzeln erkannt wird. 'Chords' erzeugt über dem gespielten Grundton (die zuerst betätigte Taste bestimmt jeweils den Grundton) einen Dur-Akkord, wenn die Augen geöffnet sind und einen Moll-Akkord, wenn sie geschlossen sind. 'Volume' erhöht die Verstärkung des aktiven Oszillators bei geöffneten Augen und verringert sie bei geschlossenen Augen langsam wieder. 'Modulate' erzeugt eine FM-Synthese und variiert bei geöffneten Augen ständig die Modulationsfrequenz und den -index.

Akkorde
Sind mindestens zwei Oszillatoren auf den Typ 'Add' eingestellt, können die angegebenen Arten von Akkorden durch Anklicken automatisch eingestellt werden. Hierzu werden die Intervalle je nach Anzahl der beteiligten Oszillatoren möglichst sinnvoll eingestellt. Kann auch mit den 'Pfeil hoch'- und 'Pfeil runter'-Tasten bedient werden.

SYSTEMARCHITEKTUR



Der Synthesizer wird, ähnlich dem Beispiel aus der Vorlesung, von einem Objekt der Klasse *OscillatorSource* koordiniert. Es kontrolliert eine gewisse Anzahl von *Oscillator*-, *Envelope*- sowie *Filter*-Objekten. Neben den vier Oszillatoren, die im Synthesizer sichtbar sind, wird eine Reihe weiterer Oszillatoren samt Envelopes, im Code als *„oods“* und *„eods“* (*oscillators / envelopes on demand*) bezeichnet, initialisiert. Diese werden verwendet, um Mehrstimmigkeit bei der Betätigung mehrerer Tasten zur gleichen Zeit zu erzeugen (siehe Beschreibung des technischen Teilaspekts).
Außerdem enthält *OscillatorSource* je ein Objekt des Typs *Addition* und *Modulation*, die automatisch modifiziert werden, wenn der Benutzer die Typen der Oszillator entsprechend verändert: Existiert sowohl ein Carrier als auch ein Modulator und ist einer der beiden als aktiver Oszillator registriert, wird die Frequenzmodulationssynthese aktiviert. Sind dagegen mindestens zwei Oszillatoren auf „Add“ gestellt und ist einer davon der aktive Oszillator, werden die Signale zusammenaddiert und normalisiert.
OscillatorSource liefert permanent Samples an das aus der Vorlesung bekannte *AudioPlayer*-Framework. Dabei werden die Samples zunächst vom entsprechenden Oszillator oder einer Kombination mehrerer Oszillatoren generiert und vom dazugehörigen *Envelope*-Objekt verarbeitet. Sind *On-Demand-Oszillatoren* aktiv, werden daraufhin auch deren Signale in der gleichen Weise angefragt und mit dem Ursprungssignal verrechnet. Das auf diese Weise modifizierte Signal wird anschließend gegebenenfalls noch von einem *Filter*-Objekt bearbeitet und letztlich dem *AudioPlayer* übergeben.

Alle Einstellungen, die der Benutzer an der Bedienoberfläche vornimmt, werden von der *MainWindow*-Klasse registriert und an *OscillatorSource* übermittelt. Außerdem fragt *MainWindow* den Status der Augenerkennung, gesteuert über einen Timer, vom *Detection*-Objekt, welches auf einem separaten Thread läuft, ab.
Eingehende MIDI-Signale werden unter Verwendung des *Drumstick*-Frameworks direkt in der *Main*-Klasse verarbeitet und an *MainWindow* weitergeleitet. Auch hierfür wird ein eigener Thread verwendet.

BESCHREIBUNG EINES TECHNISCHEN TEILASPEKTES

Polyphonie

Der entwickelte Synthesizer ist in der Lage, durch Kombination mehrerer Einzelsignale polyphone Klänge zu generieren. Dies wird auf zwei unterschiedliche Weisen ermöglicht: Zum einen können mehrere der vier dargestellten Oszillatoren gleichzeitig aktiv sein. Das bedeutet, dass in diesem Fall bis zu vier Oszillatoren mit möglicherweise unterschiedlichen Wellenformen und Hüllkurven die Einzelsignale liefern. Diese werden im Laufe der Sample-Verarbeitung zunächst addiert, um das Ergebnis anschließend mittels Division durch die Zahl der beteiligten Oszillatoren zu normalisieren, um Verzerrungseffekte zu vermeiden. Letzteres kann allerdings auch umgangen werden, indem der *„Max. gain“*-Regler (siehe Synthesizer-Übersicht) auf einen Wert über 1 verschoben wird.
Die andere Herangehensweise, um Mehrstimmigkeit zu erzeugen, findet Einsatz, wenn am Synthesizer mehrere Tasten gleichzeitig betätigt werden, wie es beim gewöhnlichen Einsatz ja ständig vorkommt. In diesem Fall werden bis zu 30 so genannte *„On-Demand-Oszillatoren“* samt passender Hüllkurven bereit gehalten, die bei Bedarf aktiviert werden können. Wird also zur ersten eine zweite Taste am Synthesizer gedrückt, werden alle Einstellungen, die den aktiven Oszillator betreffen auf den ersten freien *„On-Demand-Oszillator“* kopiert, bevor er aktiviert wird. Der Vorgang kann also für bis zu 30 zusätzlich gedrückte Tasten ausgeführt werden, wobei beim Loslassen einer Taste der entsprechende Oszillator wieder deaktiviert und freigegeben wird.
Auch in diesem Fall muss das errechnete Sample nach dem Addieren wieder normalisiert werden. Im Zuge dieser Berechnung konnte, besonders bei kurzen Attack-Zeiten, das Problem auftreten, dass durch ständiges Zuschalten einzelner Oszillatoren ein hörbares Knacken entsteht. Daher werden neu aktivierte Oszillatoren zunächst mit einer verringerten Gewichtung in das Endsignal eingebracht, so dass während der Berechnung der ersten Samples keine zu starken Sprünge im Signalverlauf verursacht werden.
Auch eine Kombination aus beiden Arten der Polyphonie ist möglich: Ist der Synthesizer etwa so eingestellt, dass über dem Grundton Akkorde generiert werden, so wird der erste gespielte Ton als Grundton interpretiert, während alle weiteren zeitgleich betätigten Tasten keinen Einfluss auf die

eingestellten Intervalle haben. Hinsichtlich der Gewichtung werden die vier visualisierten Oszillatoren im Synthesizer den *„On-Demand-Oszillatoren“* gegenüber bevorzugt: Alle durch gleichzeitiges Betätigen mehrerer Tasten erzeugten Töne teilen sich den Signalanteil des Oszillators, der den Grundton vorgibt.Auswertung

Synthesizer

Das größte Problem waren auftretende Störgeräusche (wie z.B. Knacken bzw. Knistern), die wir anfangs nicht zuordnen konnten. Es hat einige Zeit gedauert, bis wir herausgefunden haben, dass der Timer, den wir zum Aktualisieren der Grafik verwenden, dafür verantwortlich war. Egal wie groß wir das Aktualisierungsintervall gesetzt haben, die Berechnungen sind anscheinend aufwändig genug, um bei der Ausführung die Verarbeitung des Audiosignale zu verzögern und ein Knacken zu verursachen. Die Lösung war dann, die Aktualisierungen der Grafik im Interface deaktivierbar zu machen. Außerdem haben wir, auch um eine andere Art von Knacken, nämlich das beim Zu- und Wegschalten von Oszillatoren, zu verringern, zusätzlich zu den Hüllkurven eine Art Dämpfung der zugeschalteten Signale eingebaut, so dass ein Oszillator bei Aktivierung für eine (sehr kurze) Zeit mit verringerter Gewichtung in des Endsignal reingerechnet wird. Man kann das sehr gut mit einer Fensterung (z.B. Blackman-Harris-Fenster) vergleichen.

Gesichtserkennung

Zuerst hatten wir mit Problemen bei der Einbindung der OpenCV Bibliothek unter MacOS zu kämpfen. Durch die Verwendung von Homebrew und OpenCV via Homebrew funktionierte es dann aber letztendlich recht gut.

Durch die verwendeten Haar-Cascades funktionierte das Erkennen des Gesichtes insgesamt sehr gut. Das Erkennen der Augen bereitete allerdings einige Probleme, sodass wir viel herum probieren mussten, bis wir vernünftige Werte erhielten. Auch die Registrierung der geöffneten Augen, nachdem sie geschlossen waren, stellte uns vor Probleme. Wir mussten hier viel ausprobieren bis die Software zwischen Trackingverlust und Blinzeln unterscheiden konnte.

Zusammenführung

Bei der Zusammenführung vom Synthesizer und der Gesichtserkennung mussten wir die Bildverarbeitung auf einem anderen Thread auslagern, damit sich die Berechnungen nicht gegenseitig blockieren. Das stellte sich als insgesamt nicht so einfach heraus, hatte aber letzten Endes hauptsächlich mit den Eigenheiten von C++ zu tun.

Filter

Beim Programmieren der Filter kam es zu Problemen mit der Mathematik in C++. Zuerst versuchten wir FIR-Filter zu verwenden und die nötige Faltung durch FFT zu realisieren, was allerdings in Echtzeit zu enormen Latenzen führte. Die Suche nach einer geeigneten Mathematik Library, die auch direkt Faltung unterstützt, stellte sich als hoffnungslos heraus. Schlussendlich haben wir aus der „A Collection of Useful C++ Classes for Digital Signal Processing“[1] Bibliothek einen Chebyshev-Filter verwendet.

Fazit

Insgesamt haben wir nicht zu 100% umgesetzt, was wir ursprünglich geplant haben. Vor allem hinsichtlich der Gesichtserkennung, haben wir es nicht geschafft, andere Bereiche des Gesichts (z.B. Mund) zur Steuerung des Synthesizers sinnvoll mit einzubauen. Letzten Endes lag das daran, dass wir den zeitlichen Aufwand dieses Projektes etwas unterschätzt haben. Wahrscheinlich hätte eine Person mehr im Team das Ganze merklich entzerrt und hätte dazu beigetragen, die Zielsetzung zu erreichen.
Hinsichtlich der Usability ist hier natürlich noch viel Luft nach oben. Das Entwickeln eines rundum zufrieden stellenden Produktes würde jedoch den zeitlichen Rahmen dieses Kurses sprengen. Dennoch sind wir überzeugt, dass wir den Vergleich mit den anderen Projekten des Kurses nicht scheuen müssen und ein gutes, funktionierendes Programm erstellt haben, dass durch seine Vielseitigkeit zum Ausprobieren einlädt.

[1] http://www.cplusplus.com/forum/general/37896/