Dokumentation zur Tapete augenblicken

Judith Moosburner
in Zusammenarbeit mit Juliane Mielitz, Produktdesign

Projekt TAPETE
Bauhaus Universität Weimar
Gemeinschaftsprojekt der Lehrstühle für Interaction Design, Geschichte und Theorie der Kunst und Systeme der Virtuellen Realität.
Fakultäten Gestaltung und Medien, SS 2005

Betreuung:
Prof. Dr. Bernd Fröhlich, Dipl.-Inf. Jan Hochstrate
Prof. Wolfgang Sattler, Prof. Dr. Karl Schawelka, Dipl.-Ing. Stefan Kraus

Inhalt:

1. Einführung in das Projekt

1.1 Was ist Tapete?

Was ist Wand? Was ist Tapete? - dies waren die grundlegenden Fragen zu Beginn des Projektes.
Zahlreiche Vorträge zur kunsthistorischen Entwicklung der Wandgestaltung von Prof. Schawelka und ein Ausflug ins Tapetenwerk A.S.Creation dienten dazu, einen Einblick in das Thema zu erhalten.



A.S.Creation, das größte Tapetenwerk Deutschlands
- ein Einblick in Herstellungsprozeß und Vertrieb einer Tapete

Eine einwöchige Exkursion nach Vicenza/Venedig veranschaulichte Praxis und Praktiken der Wanddekoration von der Antike bis heute - was hat sich bewährt, was wurde verändert.

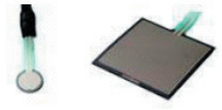


Struktur durch Wiederholung
Das Ornament als zugrunde liegendes Element, aus welchem heraus ein Rappurt entsteht

1.2 Welche technischen Möglichkeiten gibt es? - Grundlagenforschung im Labor

Das Ergebnis des Projektes sollte eine Tapete sein, die dem aktuellen Zeitgeist entspricht, eine interaktive Wand und weg vom starren Rapport, weg von der bloßen Betrachter - Wand Position.
Dazu wurde zuerst geforscht, welche technischen Möglichkeiten zur interaktiven Gestaltung vorhanden und geeeignet sind:

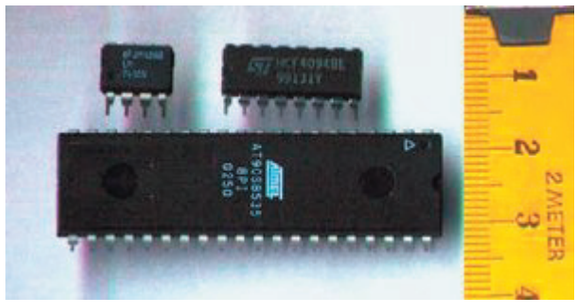
- Elektrotechnik: Einführung in Elektronik, Schaltungen und deren zugrunde liegende Elemente
- Sensorik: Einsatzweise und Funktion von Sensoren



Drucksensor

Bewegungssensor

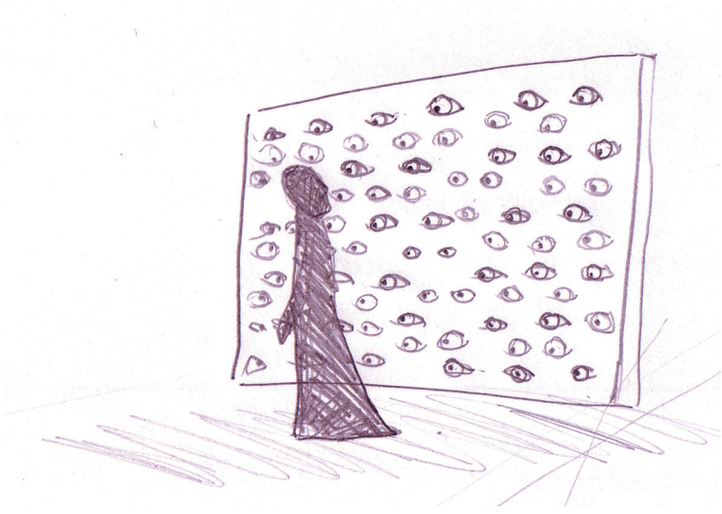
- Mikrocontroller: Kleinstcomputer auf einem Chip
  sie können durch programmierte Logik die Aufgabe analoger, diskreter oder digitaler Schaltungen übernehmen
  sie beeinhalten digiale und analoge Ein- und Ausgänge, AD- Wandler, Flash- und EEPROM- Speicher sowie Schnittstellen für USB, serielle oder parallele Ports und können somit mit anderen Computern kommunizieren, Sensordaten verarbeiten usw.
  sie sind klein, günstig und stromsparend
  programmierbar in Assemblersprachen, und sind endlich oft wiederprogrammierbar



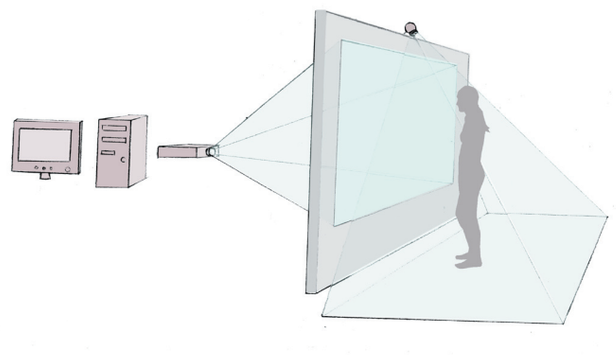
Mikrokontroller unterschiedlicher Größen und Leistungsfähigkeit

1.3 Die Tapete augenblicken - Idee und Umsetzung

Idee:
Umkehrung der Situation Wand und Betrachter - die Wand besteht aus Augen, welche nun den Besucher betrachten und jeder seiner Bewegungen folgen



Bestandteile der Installation sind:
- Projektor, Leinwand: die Augen werden von hinten auf die Wand projiziert - so steht der Betrachter nicht im Lichtstrahl
- Webcam: diese befindet sich oberhalb der Projektionswand und filmt den unmittelbaren Bereich davor, die Daten werden an den Rechner weitergeleitet
- ein leistungsfähiger Rechner, idealerweise mit 2 Grafikkarten ausgestattet
- das Programm MaxMSP inklusive Grafikaufsatz Jitter, erhältlich unter http://www.cycling74.com hier wird der Datenstrom aus der Webcam weiter verarbeitet, die Augen werden angesteuert, und die Ausgabe zum Projektor weitergeleitet
(Das Programm ist leider auch kostenpflichtig, man kann sich jedoch zum Testen eine 30-Tage-Trial Version herunterladen, die Fakultät Gestaltung der BUW besitzt außerdem fünf dauerhafte Lizenzen)



2. MaxMSP Jitter

MaxMSP ist ein Programm, das ürsprünglich zur Soundbearbeitung entwickelt wurde und stellt hier die zentrale Steuereinheit dar. Ein- und Ausgabegeräte können angesteuert und geregelt werden.
Später kam der Grafikaufsatz Jitter hinzu und ermöglichte die Darstellung grafischer Elemente, vom zweidimensionalen Bild über Film bis hin zu dreidimensionalen Objekten, wie hier verwendet.
Das Programm wurde auch laufend weiterentwickelt, ist mittlerweile relativ komplex und lässt sich deshalb auch nicht in wenigen Zeilen erklären. Es gibt aber eine gute Dokumentation zu Max wie auch zu Jitter inklusive Tutorials, die einen Einstieg in das Arbeiten sehr erleichtern.

Begleitend hierzu befinden sich einige Patcher im Ordner DEMO zum Ausprobieren.
Zu Beginn sollten in Max alle Pfade zu den Verzeichnissen, die man verwendet, angegeben werden:
unter *Options -> File Preferences* da sonst externe Objekte, Texturen usw. nicht gefunden werden
Öffnet man dann einen Patcher, so ist der normalerweise im gesperrten Modus, d.h. er kann nur ausgeführt und nicht verändert werden. In den Editiermodus gelangt man entweder durch Drücken von strg + e oder durch strg + linke mouse.

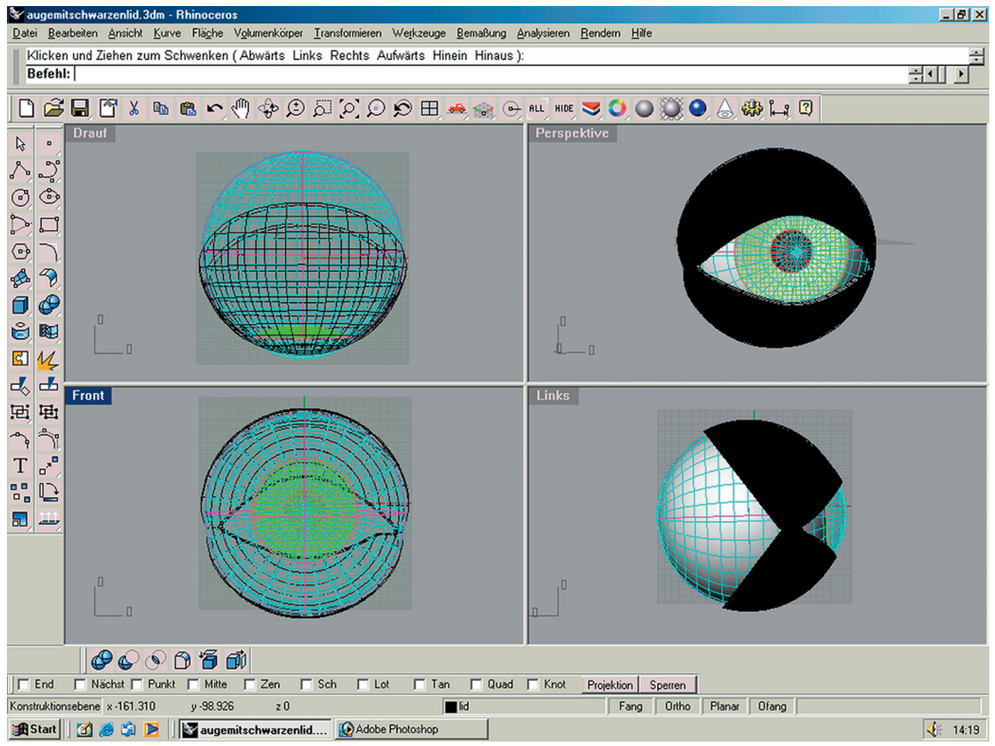
Jetzt kann man den Patcher verändern: zu den wichtigsten Elementen hier zählen die

- *message box*, zur Übermittlung von Nachrichten, Botschaften oder Variablenwerten, und die
- *object box* oder auch *patcher*, hier werden Funktionen aufgerufen oder *subpatcher* mit einer weiteren Menge an Funtkionen erstellt

2.1 Ein neues Objekt erstellen, laden, steuern

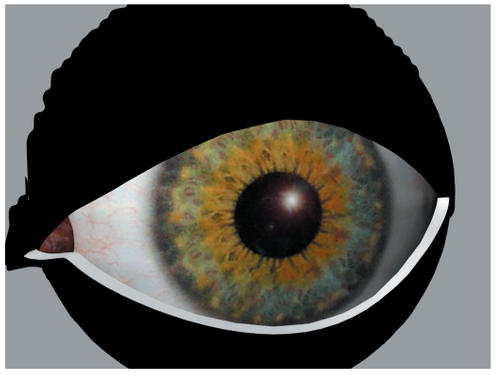
2.1.1 Das Objekt erstellen

Jitter kann mit Objekten anderer Grafik- bzw. 3D- Programme arbeiten, insbesondere Files des Programms Rhino einlesen.
Möchte man also eigene Objekte entwerfen und unabhängig von den wenigen in Jitter vorgegebenen Formen sein, so kann man diese in Rhino modellieren und exportieren.
Für das Auge wurden insgesamt vier Objekte angelegt: der Augapfel,
Ober- und Unterlid und ein Teil Haut für den kleinen Bereich oberhalb der Tränendrüse. Jitter betrachtet ein Objekt immer als ganzes, deshalb die Aufteilung in einzelne Bestandteile, denn diese müssen unterschiedlich beweglich und voneinander unabhängig sein.
Hierbei ist darauf zu achten, dass die Dateien vom Typ .obj sind und man außerdem die zugehörige .mtl - Datei mit liefert, eine Datei mit Materialdefinitionen, die benötigt wird auch wenn man keine speziellen Materialien oder Texturen festgelegt hat.



2.1.2 Textur vorbereiten

Für Texturen empfiehlt es sich ebenfalls, sie erst separat zu erstellen und sie sozusagen zurechtzuschneidern, bevor sie in Jitter geladen werden.
Für das Auge wurde das Foto eines echten Auges als Textur verwendet, für die Lider ein schlichtes schwarzes Bild und ein hautfarbener Ausschnitt für den vierten Teil.



2.1.3 Laden in Jitter

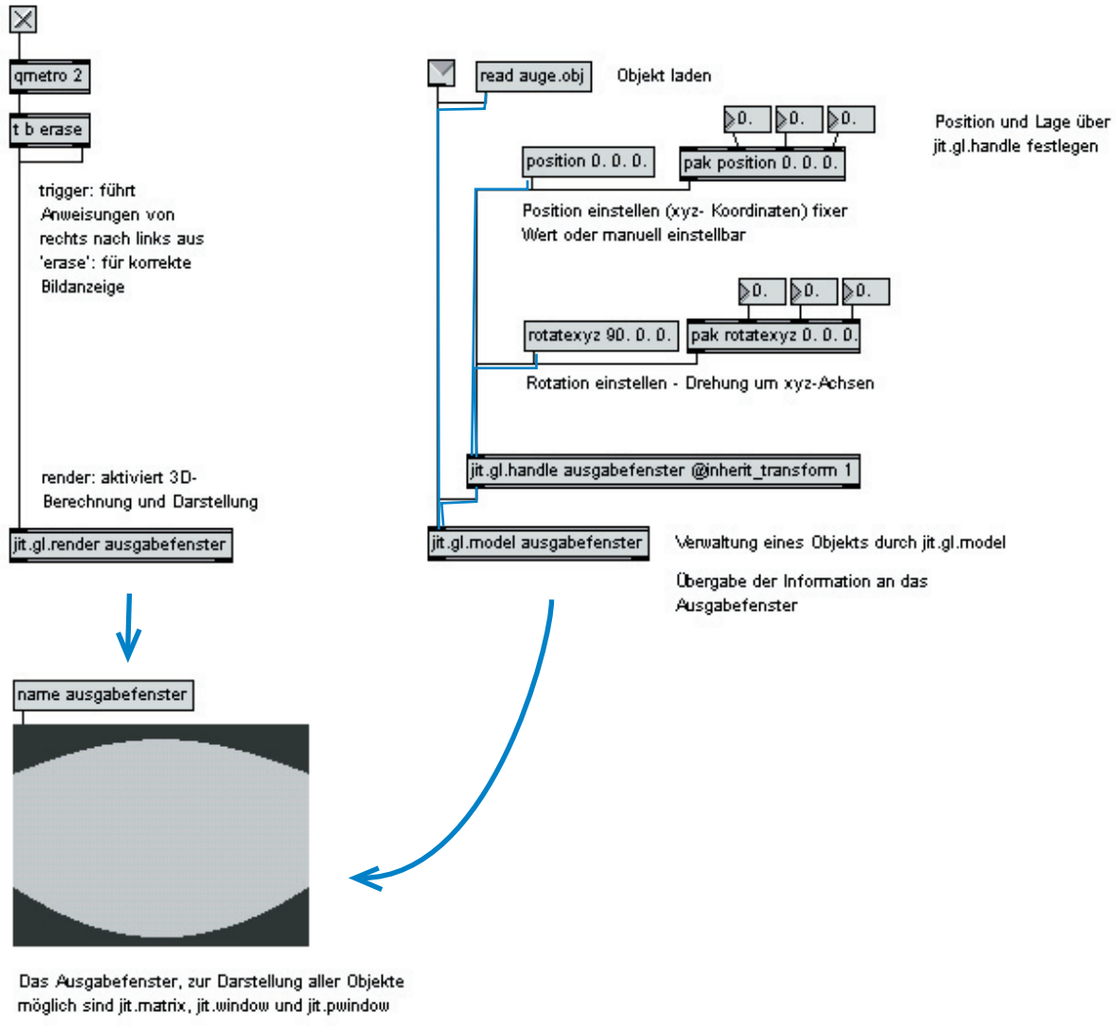
Damit Jitter die Objekte laden kann, ohne dass man jedes Mal aufs neue den kompletten Verzeichnispfad angeben muss, speichert man die Objekte am besten an einem dauerhaften Ort und macht dafür einen Eintrag unter Options -> File Preferences.

Das Laden, z.B. des Augapfels, erfolgt durch den Befehl *read auge.obj*
Dieser Aufruf wird weitergeleitet an *jit.gl.model*, die Funktion zur Bearbeitung selbsterstellter Objekte. Es kann genau ein Objekt verwalten und gibt diese Information direkt an die 3D- Ausgabe weiter.

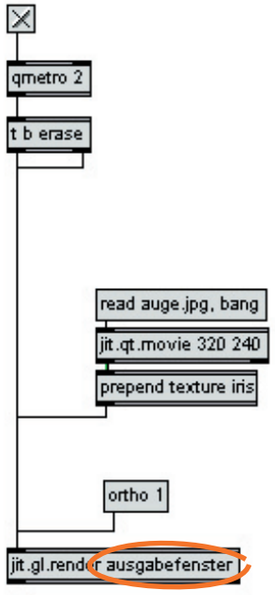
2.1.4 Steuerung, Ausgabe

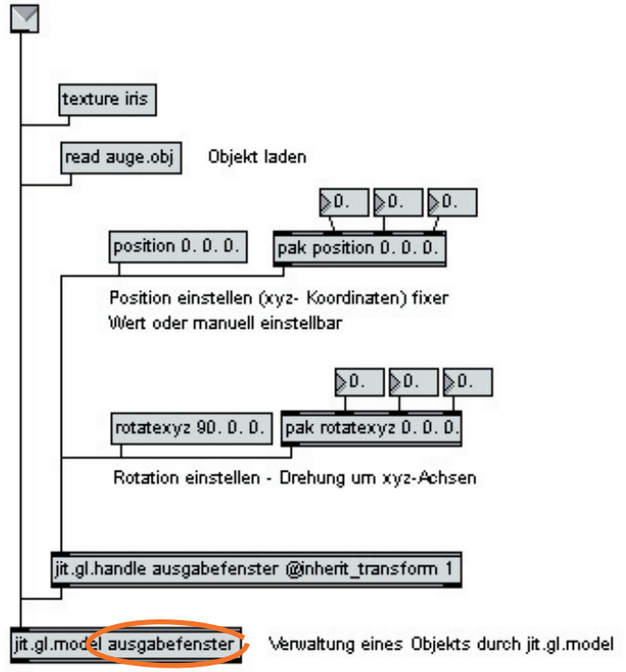
Die Ansteuerung des Auges läuft über den Patcher *jit.gl.handle*, insbesondere durch die Befehle *position* und *rotation*.
Die folgenden Zahlenwerte sind die xyz - Koordinaten bzw. betreffen den Winkel der jeweiligen Achse, um die rotiert wird.
Gibt man die Werte in einer *message box* an, so sind sie statisch und nicht veränderbar. Durch eine *pak*- Instruktion können die Werte einzeln und manuell verändert werden, dabei gilt auch immer der zuletzt angegebene Wert.
All diese Informationen werden wiederum durch *jit.gl.model* an die graphische Ausgabe weitergegeben.

→ *objektLaden1.pat*



Um ein Bild zu erhalten, muss die Funktion *jit.gl.render* aktiviert sein, hier werden alle relevanten Daten berechnet und der Bildaufbau gesteuert. Durch ein *toggle* kann das Renderin an- und ausgeschaltet werden, dem nachgeschaltet ist ein *metro* - Objekt, welches die Taktung des Bildaufbaus steuert.
(hier: metro 2 - 2 heißt, ein delay von 2 ms bzw Neuberechnung alle 2ms)
Das Bild muss ständig neu berechnet werden, da es sich ja um eine Darstellung in Echtzeit handelt.
Die endgültige Darstellung erfolgt in einem Fenster - ein *jit.window* (wie rechts abgebildet), ein *jit.pwindow* (ein eigenständiges Fenster) oder in einer *jit.matrix*.
Der Name des Fensters ist dabei Schlüsselwort und Ziel für alle Teilfunktionen.





2.1.5 Textur

Um eine Textur auf das Objekt zu mappen, muss das zugehörige Bild zuerst als eine solche initialisiert werden.
Diese Aufgabe übernimmt der - rechts gekennzeichnete - Block
read name_des_bildes -> jit.qt.movie -> prepend texture name_der_textur
Die Textur wurde also einmal festgelegt und kann ab diesem Zeitpunkt unter dem ihr zugewiesenen Namen aufgerufen werden. Im fertigen Patcher verweisen z.B. alle Augäpfel auf die Textur *iris*.

Das Ausgabefenster, zur Darstellung aller Objekte möglich sind jit.matrix, jit.window und jit.pwindow

→ *objektLaden2.pat*

2.1.6 Zusammenstellung eines Auges

Die vollständige Zusammensetzung eines Auges aus Oberlid, Unterlid, Augapfel und Hautteil erfolgt nun in einem eigenen Patcher.
Die Lider erhalten eine schwarze Textur, alle Texturen werden wiederum in einem separaten Patcher initialisiert, aber im Patcher des Auges aufgerufen.
Da Jitter jedes Objekt per default auf den Punkt 0. 0. 0. schiebt, ist auf die richtige Positionierung der einzelnen Elemente zu achten.

→ *ZusammenstellungAuge.pat*

2.1.7 Zusammenfassung der Augen zu Paaren: ein eigenständiger Patcher

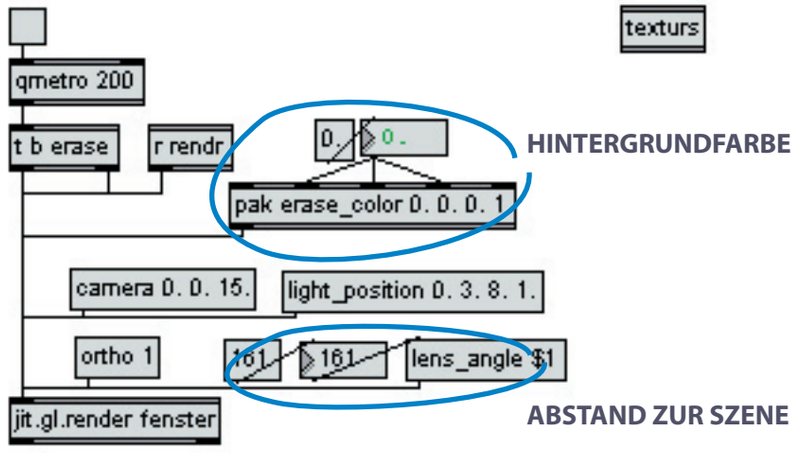
Da es sich bei der Tapete immer um Augenpaare handelt, liegen auch immer zwei Augen in einem Patcher. Sie erhalten zusätzlich noch die Möglichkeit, variable Positionen einzunehmen (wie in *objektLaden1.pat* gezeigt) und können somit angesteuert werden.
Im Hauptpatcher werden alle diese Augenpaare durch den bloßen Aufruf geladen und initialisiert. Hier werden außerdem weitere Umgebungseinstellungen festgelegt, wie etwa die Position und Entfernung der Kamera von der Szene, Beleuchtung etc.

Jedes Augenpaar hat zwei '*recieve*' - Elemente, hier werden Signale des Trackings an die Steuerung der Augen übergeben.

r null erhält Information darüber, ob sich jemand im Bereich der Kamera befindet - wenn nicht, so werden die Augen in einen Ruhemodus geschaltet.

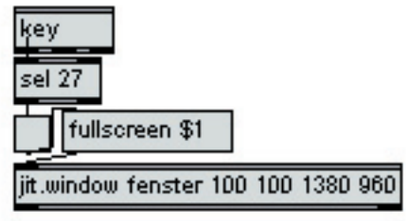
r lr übergibt die Position eines Betrachters an das Auge - der Rotationswinkel des Augapfels ändert sich entsprechend

→ Auszug aus Hauptpatcher *Tapete*

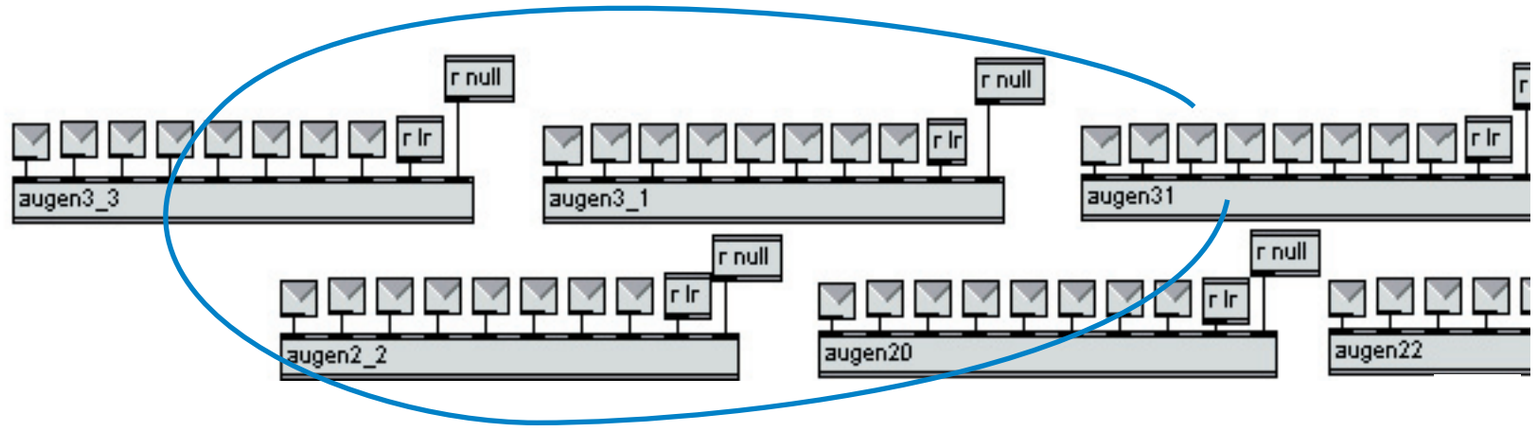


HINTERGRUNDFARBE

ABSTAND ZUR SZENE



VOLLBILDMODUS



AUGENPAARE, GELADEN

2.2 Tracking

2.2.1 Anschließen der webcam und Öffnen des Streams

Um in Jitter mit einer Kamera arbeiten zu können, reicht es eine Webcam mit USB- oder Firewire- Anschluß zu haben. Es funktionieren natürlich auch andere digitale Kameras die einen Datenstrom erzeugen können.

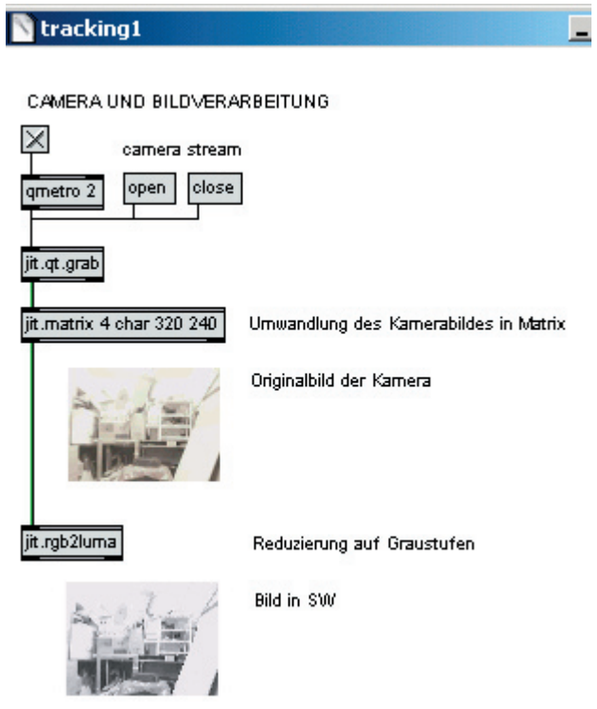
Um den Stream verarbeiten zu können, müssen sowohl Quicktime wie auch ein Programm namens WinVDIG installiert sein.

(erhältlich unter http://www.apple.com/quicktime/download/win.html und http://www.vdig.com/WinVDIG/)

Ist das alles ok, dann kann die Funktion *jit.qt.grab* den Stream einlesen, die vorgeschalteten message boxes *open* und *close* dienen zur Steuerung. Das Bild der Kamera kann sofort in eine Matrix umgewandelt und somit ausgegeben werden. Die Matrix besteht aus vier Kanälen, für die RGB- Werte und einen alpha- Kanal.

Die Funktion *jit.rgb2luma* reduziert die Farbwerte auf Graustufen, da für das Tracking eine Dreiteilung der Farben zu aufwändig wäre und schlecht zu verarbeiten ist.

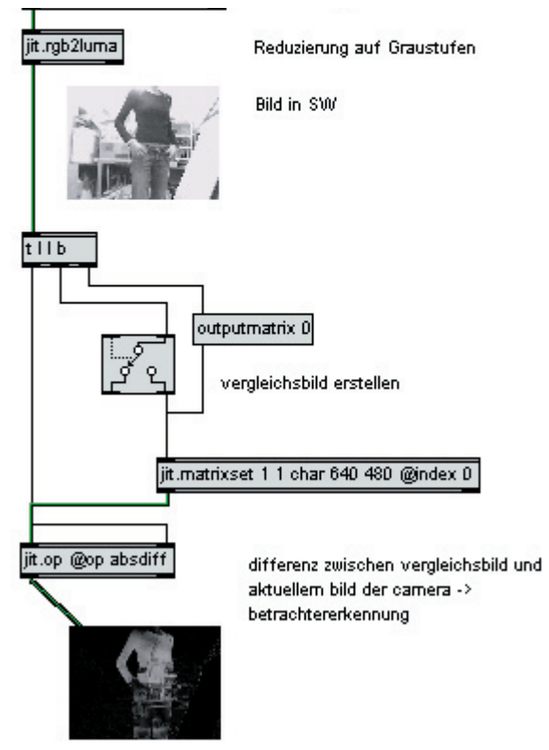
→ *tracking1.pat*



2.2.2 Personenerkennung

Das Erkennen eines Besuchers erfolgt durch eine einfache Bilddifferenz: mit einem anfangs erstellten Referenzbild werden alle folgenden Bilder verglichen und die Differenz daraus berechnet - übrig bleibt der Unterschied zwischen den Bildern, ein Betrachter oder mehrere. Wichtig für diesen Prozess ist ein switch, ein *Ggate* zum Erstellen des Vergleichsbildes und der Operator *jit.op @op absdiff* - die Bilder werden pixelweise voneinander subtrahiert und davon der absolute Wert gebildet.

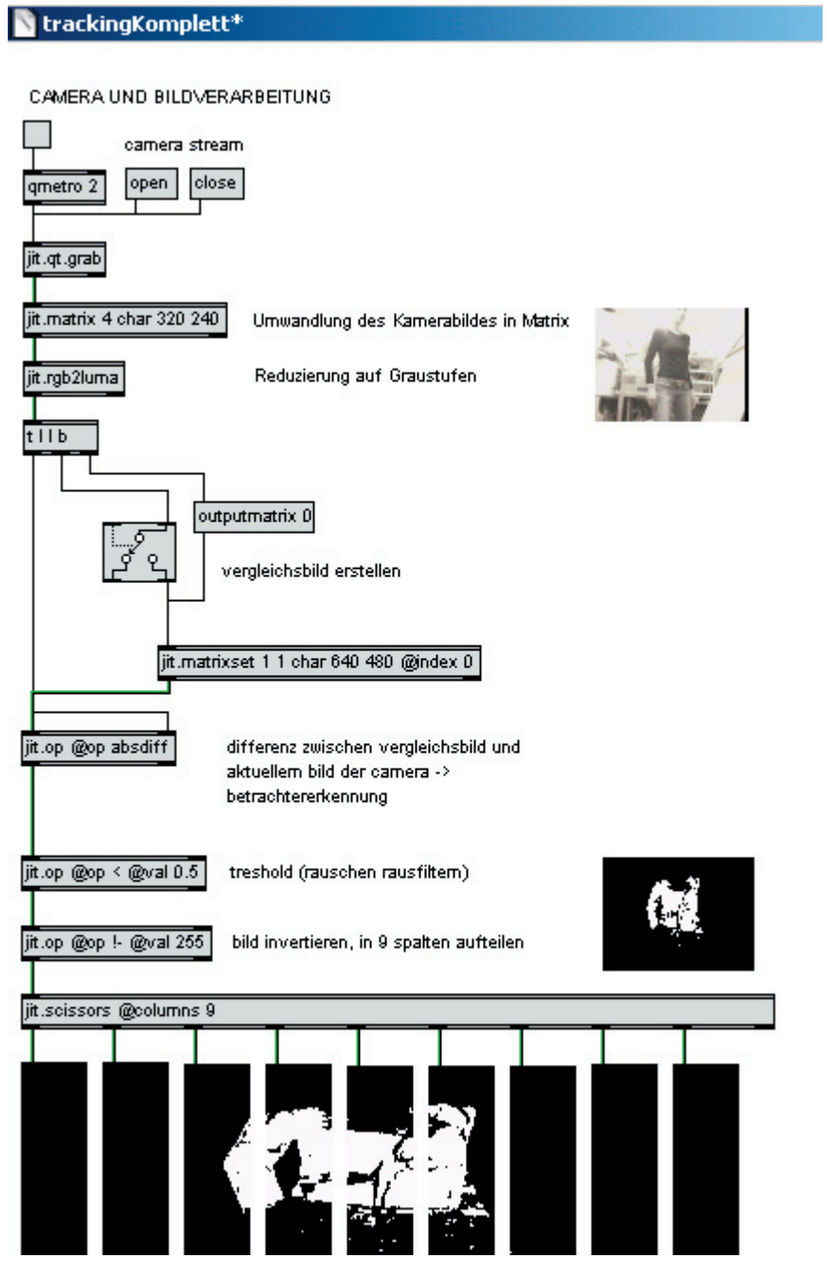
→ *tracking2.pat*



2.2.3 Das vollständige Tracking

Die erstellte Bilddifferenz enthält meist noch einen großen Anteil an Rauschen.
Dieser Teil wird herausgefiltert indem man einen Schwellwert angibt - *jit.op @op < @val 0.5* - der Wert kann zwischen 0 und 1 liegen und sollte idealerweise variierbar sein, da sich z.B. durch andere Lichtverhältnisse andere Grenzen ergeben.
Als nächstes wird die Matrix invertiert, damit sie besser ausgewertet werden kann.
Außerdem wird sie in neun Spalten aufgeteilt, so kann pro Spalte nach Ereignissen abgefragt werden. Diese Auswertung übernimmt die Funktion *cv.jit.mass*, sie berechnet den Anteil an weißen Pixeln in der Matrix - deshalb auch die Invertierung.
(*cv.jit.mass* ist ein externes Modul der Gruppe cv und muß zusätzlich installiert werden)
Ab einer bestimmten einstellbaren Menge an weißen Pixeln gibt die Spalte ein *bang* aus - das wird an alle Augen gesendet, plus der Information in welcher Spalte etwas passiert ist.
Die Augen verändern ihren Rotationswinkel entsprechend dem letzten *bang*.

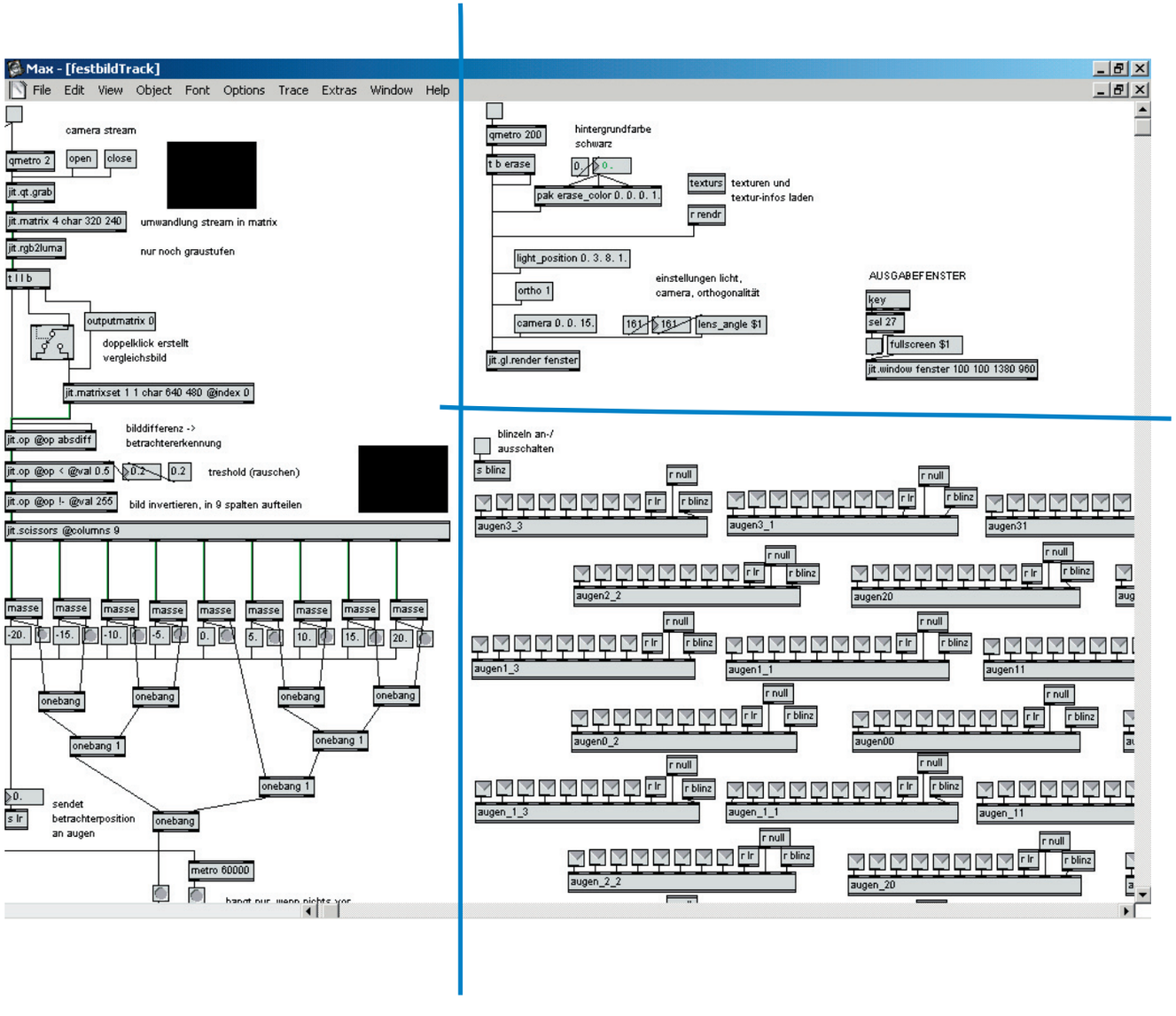
→ *trackingKomplett.pat*



2.3 Der gesamte Patcher

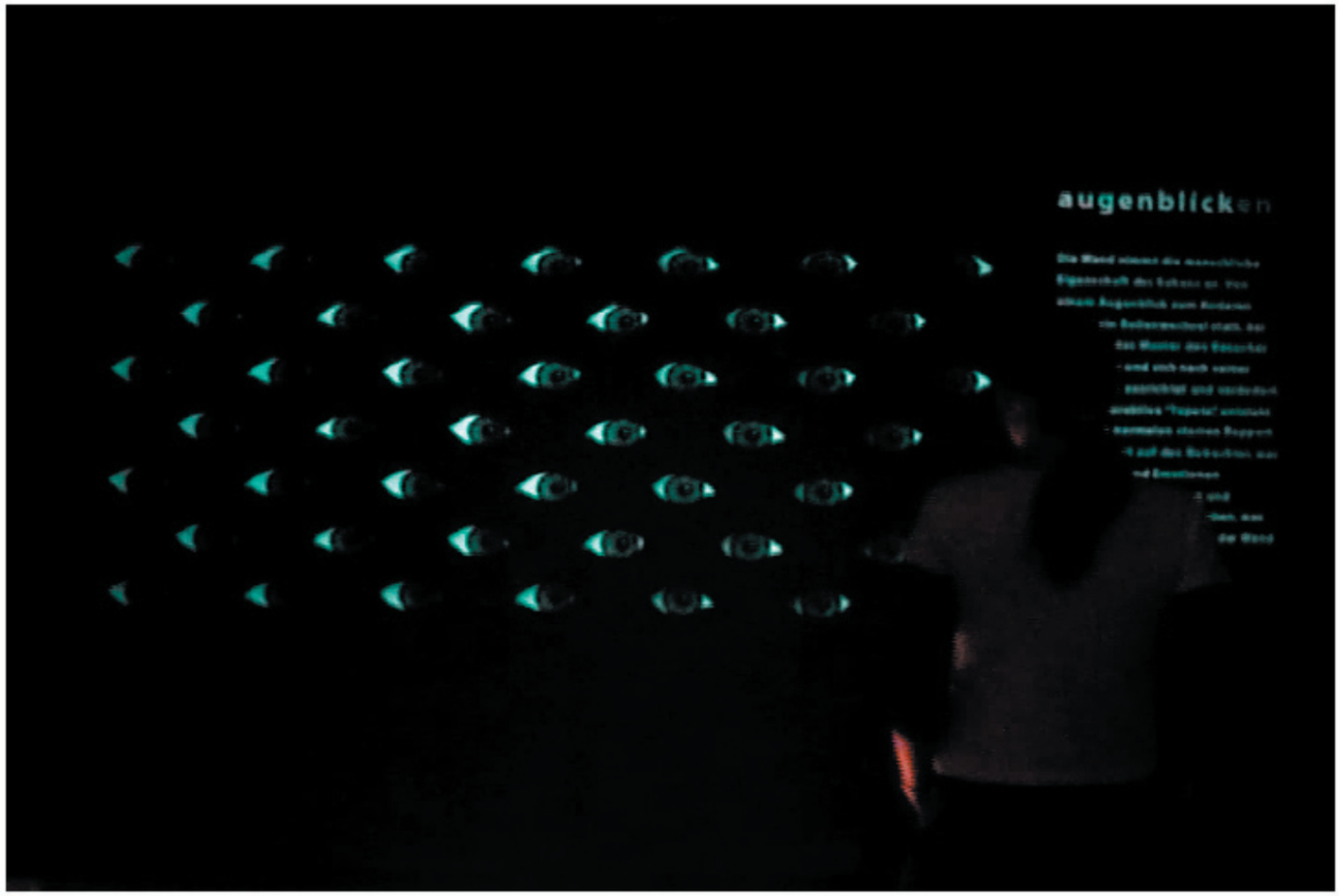
Der Hauptpatcher beinhaltet schließlich alle wichtigen Elemente:
Links befindet sich der Trackingbereich, alle relevanten Informationen am Ende der Berechnungen werden per send - recieve an die Augenpatcher übergeben. Diese sind entsprechend der Ausgabe angeordnet, ein Patcher enthält ein Augenpaar, diese sind zeilenweise nebeneinander und nach oben und unten versetzt angeordnet. Es entsteht ein Rapport.
Die Einstellungen für das Rendern sind im Bereich rechts oben festgelegt, ebenso wie die Definitionen für das Ausgabefenster. Die Taste esc aktiviert den Vollbildmodus. Die Informationen der Objekte werden in den Augenpatchern festgelegt und von dort aus direkt an den Renderbereich übergeben.

Alle Patcher, Objekte, Bilder etc. liegen im Verzeichnis 'final'.
Hauptpatcher: → *Tapete*



3. Die fertige Tapete

Die Tapete in Aktion: findet keine Bewegung vor der Kamera statt, so gucken die Augen wahllos und durcheinander im Raum herum. Betritt ein Besucher den Bereich vor der Projektion, so fixieren ihn die Augen von einem Augenblick auf den nächsten und folgen seinen Bewegungen.



Fühlt der Ruhige sich belästigt, ist der Extrovertierte über so viel Aufmerksamkeit entzückt, der Politische denkt an einen Überwachungsstaat, ein Anderer dass diese Tapete wohl nicht für das Schlafzimmer geeignet wäre und der Nächste findet Spaß daran und hüpft vor der Wand hin und her um zu überprüfen, ob die Augen ihm folgen.