



U N I V E R S I T Ä T
K O B L E N Z · L A N D A U

Fachbereich 4: Informatik

Emotion Video

Studienarbeit

im Studiengang Computervisualistik

vorgelegt von
Patrick Ohler

Betreuer: Prof. Dr. Stefan Müller
Institut für Computervisualistik/Arbeitsgruppe Computergrafik

Koblenz, im Juni 2006

Institut für Computervisualistik
AG Computergraphik
Prof. Dr. Stefan Müller
Postfach 20 16 02
56 016 Koblenz
Tel.: 0261-287-2727
Fax: 0261-287-2735
E-Mail: stefanm@uni-koblenz.de



UNIVERSITÄT
KOBLENZ · LANDAU

Fachbereich 4: Informatik

Aufgabenstellung für die Studienarbeit

Patrick Ohler

(Matr.-Nr. 202120739)

Thema: Emotion Video

Die Welt ist vielfältig und komplex. Wie gerne würden wir Szenen aus unserem Leben festhalten oder einmal in die Haut eines anderen schlüpfen; dem Profi mal über die Schultern schauen oder anderen die eigene Welt zeigen. Was wir Erlebnis nennen, ist eine Ansammlung von äußeren Einflüssen, die in der Gesamtheit die Situation beschreiben. Die Herausforderung ist, so viele dieser Sinneswahrnehmungen wie möglich digital aufzuzeichnen und wiederzugeben.

Ziel dieser Studienarbeit ist die Konzeption und prototypische Umsetzung eines erweiterbaren Gesamtsystems zur Aufzeichnung und Wiedergabe von Wahrnehmungen. Diese sollen dadurch transportiert und konserviert werden. Bei der Implementierung möchte ich mich auf visuelle und auditive Reize konzentrieren. Zentrale Punkte hierbei sind die Definition eines Speicherformats, der Entwurf eines Hardware-Systems, sowie die Analyse möglicher Nutzungs-Szenarien.

Schwerpunkte dieser Arbeit sind:

1. Einarbeitung in Stereoskopie und Videoformate
2. Konzeption und Entwurf eines Aufzeichnungssystems inkl. Hardware
3. Implementierung der Software-Komponenten
4. Demonstration von Ergebnissen
5. Dokumentation und Auswertung

Koblenz, den 01.12.2005

- Prof. Dr. Stefan Müller-

Vorwort

Motivation

Gefühle durch ein Medium übertragen, das klingt unglaublich und doch hat es einen großen Reiz. Was könnten wir alles machen? Wir könnten dabei sein, wenn ein Astronaut in den Weltraum fliegt oder einen Sonnenaufgang in der Südsee erleben ohne dort zu sein. Man könnte die Gefühle eines Gewinners ebenso weitergeben, wie die einer Person, die Angst hat. Aber auch andere Aspekte, wie die Arbeit aus Sicht eines Profis zu betrachten oder etwas Neues kennenzulernen wäre möglich. Den eigenen Tag, die letzte Woche oder sogar weit zurückliegende Ereignisse aus dem eigenen Leben noch einmal erleben, das alles macht den Anreiz an diesem Thema aus.

Aufbau

Der Aufbau dieser Arbeit soll die Entwicklung von der Entstehung von Emotionen beim Menschen über die technischen Möglichkeiten zur Aufzeichnung von Sinneswahrnehmungen bis zum Gesamtkonzept mit prototypischer Umsetzung zeigen. Zunächst soll geklärt werden, was nötig ist um Emotionen „einzufangen“ und zu konservieren und welche Möglichkeiten sich für die Wiedergabe dieser anbieten. Zentral soll ein Konzept sein, welches die momentanen technischen Möglichkeiten berücksichtigt, aber auch darüber hinaus aufzeigt, was wichtig und nötig wäre um dem Ziel möglichst nahe zu kommen, Emotionen zu übertragen. Dieses Konzept soll im Anschluss prototypisch umgesetzt werden um es so zu evaluieren. Dabei soll eine erweiterbare Plattform zunächst zur Aufzeichnung visueller und auditiver Reize entwickelt werden.

Inhaltsverzeichnis

Vorwort	3
Motivation	3
Aufbau	3
1 Einarbeitung	6
1.1 Stereoskopie	6
1.2 Stereo Violation	7
2 Konzept	10
2.1 Was sind Emotionen?	10
2.2 Die Sinne	10
2.2.1 Gesichtssinn	11
2.2.2 Gehörsinn	11
2.2.3 Tastsinn	12
2.2.4 Geruchssinn	12
2.2.5 Geschmackssinn	13
2.3 Weitere Informationen	13
2.4 Möglichkeiten zur technische Umsetzung	15
2.5 Benutzerszenarien	18
3 Implementierung	20
3.1 Aufbau der Kameras	20
3.2 Programmiersprache und Oberfläche	20
3.3 Anbindung der Kameras	22
3.4 Bild und Video speichern	23
3.5 Aufbau und Ablauf der Anwendung	24
4 Auswertung	27
5 Fazit	28

Abbildungsverzeichnis

1.1	Beispiel für ein Stereobild	6
1.2	Toe-In Methode	7
1.3	Stereobild, ein Halbbild ist unscharf	8
2.1	Datenhandschuh CyberGlove der Firma Immersion	12
2.2	Interaktionshandschuh CyberForce mit Force-Feedback	13
2.3	Logitech Quickcam Fusion	15
2.4	Bilder einzeln und kombiniert	17
2.5	Beide virtuelle Kameras zusammengesetzt als Stereokamera	17
2.6	Ausschnitt aus dem Stereovideo	18
3.1	Befestigungsschiene zur festen Montage der Kameras	20
3.2	Code der getRawFrame(int*)-Funktion	24
3.3	Klassenstruktur des Programms	25
3.4	Die Oberfläche des Programms	26
4.1	Lorgnette aus transparentem Kunststoff	27

1 Einarbeitung

In einer ersten Phase soll eine Einarbeitung in die Thematik, die benötigten Komponenten und in Stereoskopie stattfinden. Weiterhin werden Vorüberlegungen zum Konzept angestellt und grundsätzliche Dinge auf ihre Machbarkeit hin überprüft. Natürlich ist bereits an dieser Stelle eine grobe Idee des Systems vorhanden, jedoch fehlt eine Gesamtstruktur und eine detaillierte Überlegung, die erst in der Konzeptphase, im zweiten Kapitel dieser Arbeit, entstehen sollen.

1.1 Stereoskopie



Abbildung 1.1: Beispiel für ein Stereobild, Quelle: <http://www.fotocommunity.de/pc/pc/cat/5585/display/5266213>

Zentrale Frage in Bezug auf die Stereoskopie ist, wodurch ein Stereoeindruck zustande kommt und auch, welche Probleme auftreten können. Zunächst einmal ist es für das dreidimensionale Sehen notwendig, für jedes Auge ein eigenes Bild aufzuzeichnen und wiederzugeben. Die kleinen perspektivischen Unterschiede in den Bildern, die durch die Parallaxe, also dem Abstand der Augen zueinander, entstehen, sind die Tiefeninformationen für unser Gehirn. Möchte man also Abbildung 1.1 als 3d-Bild betrachten, muss man mit dem linken Auge das linke Bild fixieren und mit dem rechten Auge, das rechte Bild. Die angewandte Technik wird auch Parallelblick genannt, da nicht wie normalerweise die Augen *einen gemeinsamen* Punkt anvisieren, sondern die Augen möglichst parallel auf zwei verschiedene Bil-

der gerichtet werden müssen. Eine stereoskopische Aufnahme bezeichnet man auch als *Stereobild*. Sie besteht aus zwei *Halbbildern*, eines für jedes Auge. [Kuh99]

Zur Aufnahme und beim Betrachten von Stereobildern sollten einige Regeln beachtet werden. So sollte die Aufnahmebasis, also der Abstand der Kameralinsen, dem durchschnittlichen Augenabstand von 65mm entsprechen, damit beim Aufnehmen und Betrachten die gleichen Sehwinkeldifferenzen gelten. Es muss unbedingt verhindert werden, dass korrespondierende Punkte in den beiden Bildern ein Höhenunterschied haben. Da unsere Augen auf einer Ebene liegen, kann unser Gehirn Bilder mit einer horizontalen Verschiebung nicht verarbeiten; es kommt also kein Stereoeindruck zustande. Werden die beiden Bilder mit unterschiedlichen Objektiven aufgenommen, kann es auch innerhalb eines Halbbildes zu Höhenverzerrungen kommen, die im korrespondierenden Halbbild nicht auftreten.

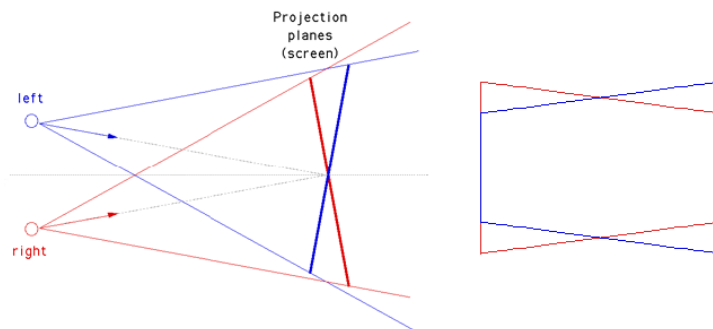


Abbildung 1.2: Toe-In Methode, links: in der Draufsicht, rechts: Perspektive aus Sicht des Betrachter, Quelle linkes Bild [Mül06]

Horizontale Verschiebungen können auch bei der Toe-In Methode mit symmetrischen Kamerafrusta entstehen. Beim Toe-In kreuzen sich die Mittelachsen der Kamerafrusta in der Bildebene. Da eine Kamera mit einer regelmäßig geschliffenen Linse immer ein symmetrisches Frustum aufspannt, kommt es an den Rändern, bedingt durch die perspektivische Verzerrung, zu horizontalen Verschiebungen (siehe Abbildung 1.2, rechts). [Mül06]

Interessant ist, dass unserem Gehirn ein scharfes Halbbild genügt, um ein klares Bild zu erhalten. In der Beispielabbildung 1.3 ist das rechte Bild mit einem Weichzeichner gefiltert. Betrachtet man nun das Stereobild mit der Parallelblicktechnik, sieht man ein scharfes und vor allem 3-dimensionales Bild. Es werden also aus dem rechten Bild lediglich die Tiefeninformationen verwendet.

1.2 Stereo Violation

Eine Verletzung des Stereosehens tritt dann auf, wenn dem Betrachter Bilder präsentiert werden, die das menschliche Gehirn nicht als 3-dimensionales akzeptiert. Ein



Abbildung 1.3: Stereobild, ein Halbbild ist unscharf, Quelle: <http://www.stereoscopy.com/gallery/index.html>

Problem, das wegen des geringeren Öffnungswinkels der Kamera im Gegensatz zum menschlichen Auge entsteht, ist das Auftreten sich widersprechender *Depth-Cues*¹. Das bedeutet, dass Dinge, die sich am linken oder rechten Rand der Szene aufhalten und nahe bei der Kamera sind, nur von einer Kamera erfasst werden. Beim Betrachten sieht also nur das eine Auge einen Gegenstand, den das andere Auge in der Realität auch sehen würde. Dieser Konflikt führt dazu, dass der Stereoeindruck zusammenbricht. Dieser Effekt tritt nur an den seitlichen Rändern auf, da sich die oberen und unteren Ränder auf der gleichen Höhe befinden. [Mül06]

Ein weiteres Problem, das mit der Auflösung der Halbbilder in Zusammenhang steht ist das *Tiefen-Aliasing*. Es bezeichnet den Effekt, dass Punkte, die auf einer gemeinsamen Ebene liegen, unterschiedliche Parallaxen haben, bzw. dass Punkte, die in verschiedener Tiefe liegen, die gleiche Parallaxe haben. Durch Erhöhen der Auflösung wird dieser Effekt gemindert, da die Rasterisierung feiner ist und somit die Anzahl der Bildpunkte einer Parallaxe ansteigt. Der maximal auftretende Fehler wird damit reduziert. [Mül06]

Unschärfe ist ein Phänomen, das sehr stark bei Kameras auftritt. In der Fotografie macht man sich die Schärfentiefe, also den Bereich in dem die Objekte scharf abgebildet werden, zunutze, um die Szene zu gestalten und einzuteilen. Es wird somit deutlich, was im Vordergrund steht und was nur dekorativen Charakter hat. Tritt dieser Effekt jedoch bei beiden Halbbildern eines Stereobildes auf, führt dies zu einer starken Irritation des Betrachters. Das Gehirn ist darauf trainiert, im 3-dimensionalen Raum Objekte zu fokussieren, die dann auf der Netzhaut scharf abgebildet werden. Dinge die weiter weg oder näher am Betrachter sind, werden unscharf. Fokussiert man nun ein weiter entfernt liegendes Objekt, so erscheint

¹Tiefenhinweise, beispielsweise fluchtende Linien oder Verdeckung von Objekten geben Hinweise auf die Entfernung derselben zum Auge des Betrachters

dieses scharf. Ist die Unschärfe jedoch mit beiden Halbbilder aufgezeichnet, so können wir das Objekt nicht mehr richtig fokussieren. Es sollten also Unschärfen in Stereobildern vermieden werden. [Kuh99]

2 Konzept

In diesem Kapitel soll es vor allem um die Entwicklung eines Konzeptes gehen. Dabei ist es unerlässlich zunächst zu betrachten, wie Gefühle entstehen. Nach einem kurzen Blick auf die Sinne des Menschen sollen technische Möglichkeiten aufgezeigt werden, wie diese digitalisiert werden können. Am Ende des Kapitels folgt das Aufstellen von Benutzerszenarien. Obwohl sie der Anlass zu dieser Arbeit sind, stehen sie am Ende des Konzeptes, da erst die Vielfalt der Möglichkeiten die Dimension der Szenarien aufzeigt.

2.1 Was sind Emotionen?

Emotionen sind kurz gesagt die Summe aus Sinneswahrnehmungen und persönlicher Erfahrung. Von unseren Sinnesorganen werden dabei Impulse an unser Gehirn weitergegeben die im Folgenden bewertet werden. [PGZ04] Dabei werden die Sinne verschieden stark wahrgenommen. Stehen wir im absoluten Dunkeln, konzentriert sich die Wahrnehmung auf den Hörsinn, den Tastsinn und den Geruchssinn; befinden wir uns hingegen am Steuer eines Fahrzeugs, ist vor allem das Sehen gefragt. Einige dieser Gewichtungen sind angeboren oder werden vom Gehirn automatisch vollzogen, wenn beispielsweise ein Sinnesorgan keine Informationen liefern kann, wie das Beispiel im Dunkeln deutlich macht.

Ein Teil der Gewichtungen ist jedoch von der Person abhängig und so empfinden Personen gleiche Situationen auf unterschiedliche Weise. Die Faktoren der persönlichen Bewertung sind vielfältig und in diesem Rahmen nicht umfassend zu beschreiben. Sie können beeinflusst sein vom Kulturkreis, der Erziehung der Eltern, Gelerntem oder bereits Erlebtem. Eine Person aus einer Wüstengegend hat mit großer Sicherheit eine andere Beziehung zu Wasser als ein Küstenbewohner. Ein extrem negatives Erlebnis kann eine Person nachhaltig negativ konditionieren und Ängste in einer vergleichbaren Situation hervorrufen, die andere Personen nicht haben. [Gol02]

Um nun die Sinneswahrnehmungen einer Person aufzuzeichnen, ist es zweckmäßig, Aufzeichnungsgeräte in der Nähe der Sinnesorgane zu platzieren um gleiche Bedingungen zu haben. So zeichnet eine am Kopf befestigte Kamera ein ähnliches Bild auf, das auch die Augen an das Gehirn weitergeben.

2.2 Die Sinne

Der menschliche Körper besitzt fünf klassische Sinnesorgane, die im Folgenden einzeln genannt und zu denen gleichzeitig Möglichkeiten zur digitalen Aufzeichnung aufgezeigt werden sollen. Darüber hinaus kennt man in der modernen menschlichen Physiologie vier weitere Sinne: *Thermozeption* (Temperatursinn), *Nozizeption*

(Schmerzempfindung), *Gleichgewichtssinn* und *Propriozeption* (Körperempfindung oder Tiefensensibilität).

Die fünf Sinne lassen sich weiterhin in zwei Gruppen unterteilen: Fernsinne (Sehen und Hören) und Nahsinne (Fühlen, Riechen, Schmecken), wobei die erste Gruppe enorm wichtige Informationsquellen für Menschen sind. Weiterhin besitzen die Organe unterschiedliche Aufnahmekapazitäten und damit indirekt auch unterschiedliche Prioritäten.

2.2.1 Gesichtssinn

Die Augen sind bei den meisten Menschen der wichtigste Sinn zur Aufnahme von Informationen. Sie dienen auch der Orientierung im Raum. Treffen wir eine Person, sammeln wir zunächst Informationen über diese indem wir sie anschauen. Möglicherweise sind die Augen auch deshalb sehr gut durch verschiedenste Geräte nachempfunden. Dies beginnt mit dem einfachen Fotoapparat und geht bis hin zu high-tech Videokameras. Unsere Augen haben eine Aufnahmekapazität von ungefähr 1 Million Bit pro Sekunde, was ca. 125 KB/s entspricht und für jeden durchschnittlichen PC handhabbar wäre. Für die Aufzeichnung könnte man jede Kamera verwenden, die über einen Anschluss zu einem Laptop verfügt. Es gibt in diesem Bereich DV-Kameras, die meist über Firewire Anschluß finden oder auch Webcams, die im Consumer-Bereich zum Telefonieren mit Video im Internet verwendet werden. Sie verfügen über einen USB²-Anschluss, der sich gerade im mobilen Bereich als Standard-Schnittstelle etabliert hat. Die Verwendung von Webcams hat darüber hinaus den Vorteil, dass sie klein, leicht, günstig und gut verfügbar sind.

2.2.2 Gehörsinn

Der Gehörsinn bildet den zweiten Fernsinn des Menschen. Er ermöglicht es, Geräusche, Laute und Töne wahrzunehmen und die Herkunft, also die Richtung und die ungefähre Entfernung, zu bestimmen. Blinden Menschen ist es sogar möglich durch Schallreflektion, ähnlich wie bei Fledermäusen oder dem Sonar, ein grobes Bild der Umgebung im Gehirn zu konstruieren. [son06] Eine technische Entsprechung zum menschlichen Ohr ist das Mikrofon. Es ist mit einer Membran ausgerüstet, die ein Pendant zum Trommelfell darstellt. Die Signale werden analog erfasst und erst im Computer mittels eines Analog/Digital-Wandlers auf der Soundkarte in einen digitalen Strom umgewandelt. Die Bedeutung des Sinns für unsere Wahrnehmung wird deutlich, wenn man betrachtet, wie viele Medien für dieses Sinnesorgan existieren. Für die digitale Aufzeichnung im Computer kommen Kompressionsverfahren zum Einsatz, die die zu speichernde Datenmenge stark reduzieren, ohne das es zu hörbaren Qualitätsverlusten kommt.

²Universal Serial Bus

2.2.3 Tastsinn



Abbildung 2.1: Datenhandschuh CyberGlove der Firma Immersion zur Dateneingabe, Quelle: <http://www.immersion.com/3d>, April 2006

Den Tastsinn bezeichnet man auch als haptische Wahrnehmung, die sich in die Bereiche taktile und kinästhetische Wahrnehmung untergliedert. Erstere bezeichnet die Oberflächensensibilität, also das Empfinden von Druck, Berührung und Vibration auf der Haut, im Gegensatz zur Tiefensensibilität, dem Empfinden von Bewegungen und der Bewegungsrichtung. Allgemein benötigt der Sinn eine direkte Verbindung zur Quelle, weshalb man ihn auch zu der Gruppe der Nahsinne zählt. Eine Aufzeichnungsmöglichkeit existiert in Form eines Datenhandschuhs sowohl für die kinästhetische, im Fall des Handschuhs beschränkt auf die Bewegung der Finger, als auch für taktile Wahrnehmung, als Tastsensoren an den Fingerspitzen. Auch die Bewegungen anderer Gelenke lassen sich mittels Biegesensoren aufzeichnen. Die Wiedergabe der Aufzeichnungen erfordert schon im Fall der Hand größere Apparaturen, die Kräfte beispielsweise mittels Motoren und Seilzügen auf die einzelnen Gelenke übertragen. Der technische Aufwand für ein solches Gerät ist dementsprechend hoch, wie auch in Abbildung 2.2 zu sehen ist.

2.2.4 Geruchssinn

Der Geruchssinn wird allgemein beim Menschen als weniger wichtig erachtet. Die Funktion der Riechorgane ist sehr komplex und wurde bisher nicht vollständig technisch nachgebildet. Es gibt mehrere Erfindungen mit dem Namen *elektronische Nase*³, die aber bisher nur auf spezielle Anwendungen ausgelegt sind. Sie können Muster wiedererkennen, wie zum Beispiel den Reifegrad oder die Qualität von industriell erzeugten Lebensmitteln überprüfen, jedoch können sie noch keine komplexen Gerüche in ihre Teile zerlegen und erkennen. Das Wiedergeben von Gerüchen wurde im Zusammenhang mit Filmen untersucht und getestet. Da Duft sich jedoch nicht aus einer geringen Anzahl von Grunddüften zusammenmischen lässt, wie beispielsweise eine Farbe aus den drei Grundfarben Rot, Grün, Blau, ist der Umgang mit

³<http://www.wissenschaft.de/wissen/hintergrund/231551.html>, Mai 2006



Abbildung 2.2: Interaktionshandschuh CyberForce mit Force-Feedback der Firma Immersion, Quelle: <http://www.immersion.com/3d>

Gerüchen noch in einem sehr frühen Stadium. Aufzeichnung und Wiedergabe ist daher noch nicht realisierbar.

2.2.5 Geschmackssinn

Der Geschmackssinn, dessen Rezeptoren sich auf der Zunge befinden, hat eine enge Verwandtschaft zum Geruchssinn. Auch dieser wird in der zivilisierten Welt nicht mehr als überlebenswichtig eingestuft, obwohl sein Fehlen eine deutliche Minderung der Lebensqualität bedeutet. Die im Abschnitt 2.2.4 vorgestellte *elektronische Nase* funktioniert, da sie auf der Identifikation von Molekülen basiert, auch in Flüssigkeiten. Es kann somit Zucker oder Salz nur ermittelt werden, wenn diese in einer neutralen Flüssigkeit gelöst werden. In diese Flüssigkeit kann nun die elektronische Nase eingetaucht werden, um die Funktion einer Zunge nachzubilden. Auch hier gelten die gleichen Einschränkungen wie in 2.2.4 - es ist weder Aufzeichnung noch Wiedergabe zur Zeit realisierbar.

2.3 Weitere Informationen

Je nach Benutzerszenario (siehe Abschnitt 2.5 auf Seite 18) kann es auch sinnvoll sein, weitere Informationen zu verarbeiten und aufzuzeichnen, um sie dem Betrachter später zur Verfügung zu stellen. Um festzustellen, wo sich die Person befindet, können im Freien GPS⁴-Daten mit abgespeichert werden. Später kön-

⁴Global Positioning System, ein satellitengestütztes System zur weltweiten Positionsbestimmung

nen diese Positionsdaten dann beispielsweise in eine Landschaftskarte eingeblendet werden. In Gebäuden steht die Standortermittlung mit dem satellitenbasierten GPS-System nicht zur Verfügung, weshalb hier andere Verfahren wie elektromagnetisches Tracking, WLAN-Tracking, Markertracking oder auch markerloses Tracking zum Einsatz kommen können.

Auch die Position zum Beispiel der Hände kann interessant sein, wenn diese sich außerhalb des Kamerabildes befinden. Die von unserem Körper automatisch durchgeführte Propriozeption, also das Wissen unseres Gehirns um die Position und Stellung unserer Gliedmaßen, kann so auch für den Betrachter erfahrbar gemacht werden. Technisch könnte mittels Biegesensoren an den Gelenken, die Position der Hand zurückberechnet werden.

Möchte man digital festhalten, welche Kräfte zum Zeitpunkt der Aufnahme auf den Körper wirken, könnte man dies mittels Trägheitssensoren messen. Damit lassen sich auch Änderungen der Bewegungsrichtung oder Beschleunigungskräfte bestimmen.

Oft kann es für den Rezipienten eines *Emotion Video* wichtig sein zu wissen, wohin der Blick der Person geht. Sind die Kameras am Kopf befestigt, so weiß man lediglich, in welche Richtung der Kopf gedreht ist, nicht aber, was angeschaut wird. Hier könnte Eye-Tracking die Bewegungen des Augapfels verfolgen um die Stelle im Video bei Bedarf markieren zu können. Es kann dabei je nach Anwendungsfall nötig sein, nur die Fixationen oder Sakkaden oder auch beides zu tracken.

Zu Archivierungszwecken kann es sinnvoll sein, in regelmäßigen Abständen Zeitstempel aufzuzeichnen. Informationen lassen sich so möglicherweise leichter sortieren und überblicken.

Interessant könnten auch, je nach Anwendungsfall, Informationen über Körperfunktionen wie beispielsweise Puls sein. Ängste, Freude oder körperliche Anstrengung werden so für den Rezipienten erfahrbarer. Diese Daten müssen jedoch visuell eingeblendet oder, im Fall vom Puls, hörbar gemacht werden, so wie es auch in Filmen zur Dramatisierung eingesetzt wird.

Um die oben genannten zusätzlichen Informationen speichern zu können, bestehen grundsätzlich mehrere Möglichkeiten. Für jede Informationsart kann eine spezielle Datei mit eigenem Format angelegt werden. Diese Methode hat den Vorteil, dass die Daten getrennt voneinander behandelt werden und so spezielle Kompressionen eingesetzt werden können. Nachteilig ist jedoch das Synchronisieren der Daten beim Abspielen. Speichert man hingegen alle anfallenden Datenarten in einem Dateicontainer, kehren sich Vor- und Nachteile um. Ein zusätzlicher Vorteil ist, dass die Handhabung einer Datei für Benutzer transparenter ist, als für eine Aufnahme gleich mehrere Dateien zu erhalten. Angestrebt wird also das Speichern einer Datei unter Berücksichtigung integrierbarer Kompressionen.

2.4 Möglichkeiten zur technische Umsetzung



Abbildung 2.3: Logitech Quickcam Fusion, Quelle: <http://www.logitech.de>

Bei der technischen Umsetzung musste zunächst die Entscheidung getroffen werden, welche Hardware zum Einsatz kommen sollte, sowohl für die Aufnahme als auch zum Abspielen der Aufzeichnungen. Die Kameras sollten dabei folgende Anforderungen erfüllen: hohe Auflösung, klein, USB oder Firewire, preiswert, verfügbar. Nach einiger Recherche fiel die Entscheidung zugunsten der Quickcam Fusion (siehe Abbildung 2.3) der Firma Logitech⁵. Mit einem 1,3 MP Sensor und einer Auflösung im Video Modus von 640x480 Pixeln, bei gleichzeitig guter Belichtung durch die patentierte RightLight™-Technologie, hatten diese Kameras die besten Werte und sind dabei noch relativ handlich. Von Logitech gibt es ein spezielles Quickcam SDK⁶, was eine gute Software-Anbindung an die Kamera-Hardware verspricht und besonders den Zugriff auf besondere Einstellungen ermöglicht.

Um Ton aufzuzeichnen besitzen die vorgestellten Kameras bereits ein brauchbares Mikrofon, das über den gleichen USB-Anschluß verfügbar ist. Weitere Geräte zur Sound-Aufzeichnung wurden deshalb zunächst nicht in Betracht gezogen.

Damit ein Stereobild aufgezeichnet werden kann, müssen beide Kameras fest montiert sein, d.h. weder Abstand noch Winkel darf sich während der Aufnahme ändern, denn eine Veränderung würde den Verlust des 3d-Eindrucks beim Abspielen bedeuten. Es kann jedoch sinnvoll sein den Kameraabstand (sprich: die Stereobasis) je nach Anwendungsgebiet zu ändern, weshalb sie auf eine Schiene montiert werden sollten, auf der sie leicht verschoben und arretiert werden können. Auch der Neigungswinkel und der Winkel der Kameras zueinander sollte variabel bleiben. Zur Kalibrierung ist eine Softwareunterstützung wünschenswert, die ein Testmuster einblendet, das von beiden Kameras zur Deckung gebracht werden muss.

Die Suche nach geeigneter Software gestaltete sich ein wenig schwieriger. We-

⁵<http://www.logitech.de>, Mai 2006

⁶Software Development Kit

gen der besseren Unterstützung der Kameras wurde das Betriebssystem Microsoft Windows XP als Plattform ausgewählt, denn nur dafür existieren Treiber des Herstellers. Zunächst wurde als Programmiersprache C# ausgewählt. Als Anbindung an die beiden Kameras existierte beim Kauf der Hardware ein Quickcam SDK des Herstellers, welches jedoch *nicht* kompatibel zur neuen Quickcam-Serie war, was aber zunächst verschwiegen und erst nach mehreren gescheiterten Versuchen von einem Hotline-Mitarbeiter bestätigt wurde. Das Entwickler-Internetportal für Quickcams von Logitech wurde wenige Tage nach dem Anruf geschlossen und die Frage nach einem neuen SDK verneint. Um die Kameras dennoch nutzen zu können, sollte nun das plattformübergreifende OpenSource Projekt PortVideo⁷, was unter Windows die Library DsVideoLib⁸ verwendet, benutzt werden.

Zum Speichern der Daten sollte ein existierendes Container-Format verwendet werden, das die Möglichkeit bietet, unterschiedliche Multimedia-Daten abzuspeichern. Als wünschenswert hat sich dabei das Matroska-Format⁹ gezeigt, das sich jedoch in einer noch frühen Entwicklungsphase befindet. Es bildet ein Container für Video und verschiedene Audiospuren, sowie Untertitel oder sonstige Datenspuren. Es kann dabei selbst festgelegt werden, welcher Komprimierung die einzelnen Spuren unterzogen werden. Die momentan jedoch geringen Möglichkeiten zur Nutzung in eigenen Programmen und fehlende Dokumentation ließen dieses Format ausscheiden. Das am weitesten verbreitete Container-Format ist das von Microsoft entwickelte AVI-Format. Es ist zwar nicht so flexibel wie das zuvor vorgestellte, allerdings ist es gut dokumentiert und es gibt einige Libraries, die den Zugriff auf dieses Dateiformat ermöglichen.

Zur Kompression der Videodaten sollte die freie MPEG4 Kodierung XviD¹⁰ genutzt werden. Sie ermöglicht eine besonders gute Qualität bei starker Komprimierung der Daten. Doch wie soll das Video von zwei Kameras gespeichert werden? Eine Möglichkeit ist, für jeden Videostrom der aus der Kamera kommt eine eigene Datei anzulegen. Dies hätte jedoch den Nachteil, dass sowohl beim Abspeichern, als auch beim Abspielen darauf geachtet werden muss, dass beide Dateien synchron geschrieben bzw. gelesen werden. Eine weitere Möglichkeit wäre, ein eigenes Format für Stereovideo zu entwickeln, das mithilfe von Keyframes die Synchronisation regelt. In einem speziellen Format könnte außerdem berücksichtigt werden, dass sich Stereobilder wenig voneinander unterscheiden (siehe Bild 3 in Abbildung 2.4). Anstatt also die Bilder separiert voneinander zu betrachten, könnte man dieses Wissen bei der Kompression nutzen. Wie beispielsweise bei der MPEG-Kodierung mehrere aufeinander folgende Frames betrachtet werden und nur die Änderungen von einem Frame zum nächsten gespeichert werden, könnte man nur die Unterschiede

⁷<http://www.iaa.upf.es/mtg/reactable/?portvideo>, April 2006

⁸<http://www.ims.tuwien.ac.at/~thomas/dsvideolib.php>, April 2006

⁹<http://www.matroska.org>, April 2006

¹⁰<http://www.xvid.org/>, April 2006

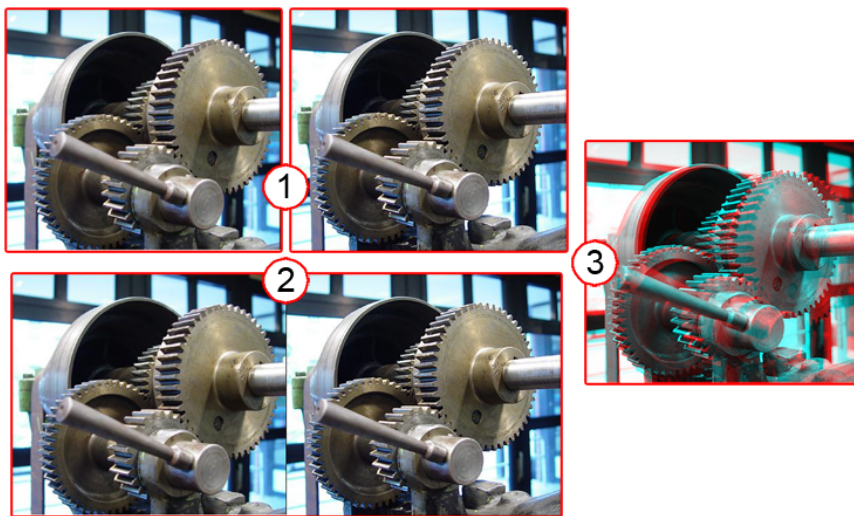


Abbildung 2.4: 1. ein Bild von jeder Kamera, 2. zusammengesetztes Frame für Video, 3. Parallaxe im Anaglyphenbild sichtbar

von rechtem zu linkem Kamerabild abspeichern, was eine Reduzierung der Speicherplatzanforderungen bedeuten würde. Zu diesem Ansatz existieren zum jetzigen Zeitpunkt jedoch noch keine fertigen Implementierungen und die Entwicklung einer solchen soll nicht Gegenstand dieser Arbeit sein.

Die einfachste Lösung ist es daher, zwei Bilder, jeweils ein Bild jeder Kamera, zeilenweise zu einem Videobild zusammenzufügen. So erhält man eine Videodatei, die auf der rechten Seite das Bild der rechten Kamera und auf der linken das der linken Kamera enthält. (siehe Bild 2 in Abbildung 2.4)

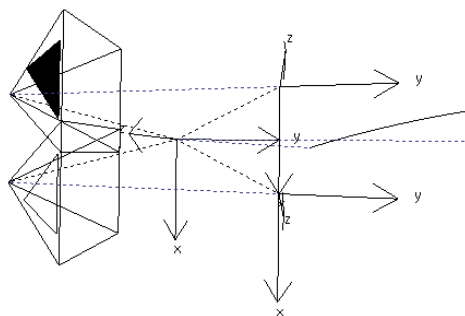


Abbildung 2.5: Beide virtuelle Kameras zusammengesetzt als Stereokamera in Blender3d

Die Machbarkeit dieses Ansatzes wurde bereits in der Einarbeitungsphase auf einer Stereoleinwand erprobt. Dazu wurde eine 3d-Szene mit dem OpenSource 3d-

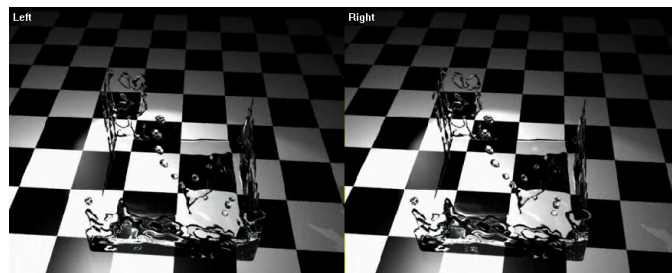


Abbildung 2.6: Ausschnitt aus dem Stereovideo, das mit Blender3d erstellt wurde. Dabei fährt die Stereokamera (siehe Abb. 2.5) um eine Flüssigkeit.

Designer Blender¹¹ erstellt. Eine spezielle Anordnung aus zwei Kameras mit mehreren Abhängigkeiten zu Empty-Objekten erlaubt das Rendern eines Videos für das rechte und eines für das linke Auge. Erstellt wurde eine Kamerafahrt um sich bewegende Flüssigkeit, wie es Abbildung 2.5 zeigt. Daran sollte der Mindestabstand der virtuellen Kamera zum Objekt und der Stereoeindruck vor der Leinwand untersucht werden. Die Hardware der Leinwand wird so konfiguriert, dass die beiden Videobeamer mit einem PC verbunden werden. Anschließend erweitert man den Desktop der Oberfläche auf zwei Monitore. Nun kann ein beliebiger Player für Videodateien verwendet werden. Einzige Voraussetzung ist, dass dieser den Vollbild-Modus unterstützt. Stellt man nun die Anzeige des Videos auf Vollbild, so wird automatisch das Bild für das rechte und das linke Auge jeweils auf einem Beamer angezeigt. Nun muss nur noch dafür gesorgt werden, dass die Zuordnung von Beamer zum Auge des Betrachters korrekt ist.

Da nun das Container-Format festgelegt ist, muss sich das Audioformat an diesem ausrichten. Wäre die Entscheidung zugunsten des Matroska-Formates gefallen, so wäre Vorbis¹² - meist ogg-vorbis genannt - der Kompressions-Codec der Wahl gewesen. Ebenso wie auch XviD bietet dieser für den Audibereich eine hohe Kompression bei dennoch guter Qualität. Er lässt sich jedoch nicht in das gewählte AVI-Format integrieren. Der wohl zu Zeit bekannteste Audio-Codec mp3¹³ kann als Alternative verwendet werden, auch wenn bei diesem die Kompression nicht zu hoch gewählt werden sollte, da sonst Kompressionsartefakte hörbar werden. Eine Datenrate von 128kbit/s sollte hier als Minimum angesetzt werden.

2.5 Benutzerszenarien

Abschließend für dieses Kapitel soll ein Blick auf den Benutzer geworfen werden, denn nun sind verschiedene Aufzeichnungsmöglichkeiten klar und so kann ein ge-

¹¹<http://www.blender.org/>, Mai 2006

¹²<http://www.vorbis.com>, April 2006

¹³<http://www.iis.fraunhofer.de/amm/projects/mp3/>, Mai 2006

naueres Bild der Machbarkeit entstehen. Die einzelnen Szenarien sollen anhand passender Bezeichnungen klassifiziert werden.

Das naheliegendste Szenario kann man als *person-immersion* bezeichnen. Man taucht dabei in die Welt einer anderen Person ein, um möglichst exakt das zu erleben, was diese Person erfährt. Damit lassen sich sowohl Aufnahmen zur reinen Unterhaltung erstellen, wie man sie auf Jahrmärkten meist als Space-Shuttle findet, oder auch Lehrfilme, um sich in Personen hinein zu versetzen. So kennt schon jeder Fahrschüler vor der ersten Praxisstunde das Gefühl am Steuer zu sitzen oder man versetzt sich damit in Personen mit besonderen Fähigkeiten oder Bedeutung. Der Astronaut beim Start oder einem Weltraumspaziergang ist da nur ein Beispiel. Man könnte auch erfahrbar machen, mit welchen Schwierigkeiten Randgruppen unserer Gesellschaft umgehen müssen. Auch könnten damit verschiedene Gefühle direkt erfahren werden, wenn sie nötig sind, zum Beispiel ein Strandspaziergang nach einem anstrengenden Arbeitstag. Das Szenario lässt sich auch noch um Echtzeitfähigkeit erweitern, indem die Anwendung um eine Komponente zur direkten Übertragung der Daten über ein schnurloses Netzwerk erweitert wird.

Ein weiteres Szenario kann gut mit dem Begriff *over-the-shoulder* beschrieben werden. Hier soll der Rezipient möglichst viele Informationen des Akteurs vermittelt bekommen. Es ist hierbei wichtig, das Wissen und/oder Fähigkeiten übermittelt werden. Immersion spielt hier nur eine untergeordnete Rolle. Abhängig vom Einsatz können verschiedene Informationsquellen aufgezeichnet werden. Möchte man dem zukünftigen Werksmechaniker durch ein solches Video sein Aufgabengebiet näher bringen, so erscheint es sinnvoll, neben der visuellen und auditiven Wahrnehmung auch Positionsdaten, Position der Hände und Augenbewegungen aufzuzeichnen, damit diese Informationen dem Lernenden später zur Verfügung stehen. Ein großer Vorteil ist, dass einmal gespeicherte Aufnahmen immer wieder angesehen werden können und so eine Bibliothek von Problemlösungen entstehen kann. Das besondere dieser Aufnahmen ist die Perspektive, die dem Betrachter zur Verfügung gestellt wird. Sollte er später das Gelernte umsetzen, so ist ihm die Perspektive bereits vertraut.

Eine Mischform der beiden genannten ist die *self-immersion*. Dabei ist Rezipient und Akteur die gleiche Person, jedoch zeitversetzt. Man könnte sich abends Ausschnitte aus dem vergangenen Tag anschauen oder letztes Weihnachten mit der Familie erneut erleben. Wie in einem Tagebuch lassen sich so Ereignisse archivieren und erneut betrachten um sie noch einmal zu erleben. Dafür sind auch Zeitraffer oder ähnliche Funktionen denkbar.

3 Implementierung

Nach der Phase der Konzepterstellung folgt nun das exemplarische Implementieren und Evaluieren des Konzepts. Dabei wurde erst eine Konstruktion für die Kameras entworfen, damit diese fest montiert und dennoch justiert werden können. Dieser Aufbau sollte zunächst für Tests mit Standbildern verwendet werden, bevor ein Programm zur digitalen Aufzeichnung von Stereo-Bild und Ton erstellt werden sollte.

3.1 Aufbau der Kameras



Abbildung 3.1: Befestigungsschiene zur festen Montage der Kameras, es sind jedoch Position und Winkel justierbar.

Der an den Kameras bereits befindliche Befestigungsfuß zum Fixieren der Kamera an einem Monitor o.ä. wurde demontiert und beide Geräte nebeneinander auf einer Schiene angebracht, wie Abbildung 3.1 zeigt. Um die Objektive der Kameras im Abstand der Augen befestigen zu können, wurde eine Kamera „über Kopf“ montiert, wie auch in der Abbildung zu sehen ist. Um das Bild dieser Kamera jedoch wieder korrekt zu bekommen, kann man im dazugehörigen Treiber x- und y-Achse spiegeln.

3.2 Programmiersprache und Oberfläche

Zunächst wird betrachtet, welche Voraussetzungen an eine Programmiersprache zu stellen sind. Sie soll sowohl handhabbar, als auch zukunftsorientiert sein und

die Entwicklung eines GUI¹⁴ unterstützen. Zusätzlich sollte die Sprache bereits etabliert und zukunftsorientiert sein, um einen Nutzen für die Zukunft aus der Arbeit ziehen zu können.

Es kommen mehrere Programmiersprachen für das Projekt in Frage. Es soll zunächst *C#* im Detail betrachtet werden, da diese Sprache viele Vorteile mit sich bringt und ein Einarbeiten in Syntax und Funktionen reizvoll erscheinen. Ein Ausschlusskriterium allerdings ist die zur Zeit fehlende Kameraschnittstelle. Alle verfügbaren Libraries, die diese Funktion erfüllen würden, sind inkompatibel zur Programmiersprache und lassen sich somit nicht verwenden.

Als nächstgelegene Alternative wird die Microsoft Implementierung des Common-Language-Infrastructure-Standards für Windows *.net*¹⁵ betrachtet. Da auch hier, wiederum wegen fehlender Libraries im von *.net* geforderten *managed code*, keine Schnittstelle zur Kamera existiert, kann diese Sprache nicht verwendet werden.

Im nächsten Schritt wird die Programmiersprache C++ betrachtet, da hier bereits einige Erfahrung im Umgang und Programmierung vorhanden ist. Die Sprache verhält sich leider wenig tolerant gegenüber Fehlern von Seiten des Programmierers, bei richtigem Umgang bietet sie jedoch eine hohe Geschwindigkeit der Applikation, da der resultierende Code als maschinenlesbar vorliegt und nicht in einem Zwischen-code wie bei den vorher betrachteten. Die Entwicklung eines GUI ist jedoch nicht direkt vorgesehen und so muss auf eins der zahlreichen Frameworks zurückgegriffen werden, die die Erzeugung eines GUI ermöglichen und den Programmierer bei der Entwicklung unterstützen. Ein direkter Zugriff auf das Windows-API wird wegen der unkomfortablen Programmierung und der wenig vielversprechenden Zukunftsaussichten abgelehnt.

Andere - zur Zeit populäre - Möglichkeiten Fenster zu erzeugen und zu verwalten, wären *wxWidgets*¹⁶, *gtk*¹⁷, *Qt*¹⁸ u.v.m., die jeweils mit C++ verwendet werden können und sogar plattformunabhängig sind. Es gibt viele weitere Fenstermanager, jedoch sollen nur die bekanntesten und anerkanntesten Vertreter betrachtet werden. Schon bei der Betrachtung der einfachen *Hello World* Programme fällt auf, dass Qt die klarste Struktur und eine eindeutige Syntax mit sich bringt. Weiterhin lässt sich nicht jedes Framework mit jedem Compiler verwenden. So benötigt *gtk* und *Qt-OpenSource* unbedingt den *Gnu-C-Compiler*¹⁹ (*gcc*), wohingegen *wxWidgets* und *Qt* in der lizenzierten Version auch mit dem *Microsoft Visual C-Compiler* (*MSVC*) zusammenarbeiten. Dies ist deshalb wichtig, da viele Libraries mit dem *MSVC* erstellt wurden und sich somit nicht mit dem *gcc* verwenden lassen. In diesem Fall betrifft dies besonders *DsVideoLib*, auf die im folgenden Abschnitt noch

¹⁴Graphical User Interface

¹⁵<http://www.microsoft.com/net/>, April 2006, gesprochen „dot net“

¹⁶<http://www.wxwidgets.org>, April 2006

¹⁷<http://www.gtk.org>, April 2006

¹⁸<http://www.trolltech.com>, April 2006

¹⁹meist nur *gcc* genannt, <http://gcc.gnu.org>, April 2006

gesondert eingegangen werden soll.

3.3 Anbindung der Kameras

Die unkomplizierteste und beste Anbindung an jede (Multimedia-)Hardware stellen meist die Hersteller des Produktes den Entwicklern in Form von Treibern zur Verfügung. Da diese für ein Produkt zugeschnitten sind, können sie die Möglichkeiten besser und effizienter nutzen, die die Hardware zur Verfügung stellt. Oft wird erst mit der Software des Herstellers die Möglichkeit des Zugriffs auf spezielle Parameter eröffnet. Zum Zeitpunkt der Hardwarebeschaffung stellte auch der Hersteller der Webcams Logitech ein *Software Development Kit* (SDK) zur Verfügung, ohne jedoch kundzugeben, dass dieses nicht mit allen Hardware-Versionen kompatibel ist. Kurze Zeit später wurde der Entwickler-Support für diese Hardware-Sparte des Herstellers komplett eingestellt und somit muss ein neuer Weg gefunden werden, um an die Daten der Kameras zu gelangen.

Eine Möglichkeit ist es, auf das kompatible Gerät per Microsoft DirectShow Schnittstelle zuzugreifen. Allgemein stellt Microsoft unter dem Namen DirectX²⁰ ein SDK zur Verfügung, das die Entwicklung multimedialer Applikationen unterstützt. Es wird besonders in der Programmierung von Computerspielen eingesetzt, da sich der Programmierer nicht mehr um die eingesetzte Hardware speziell kümmern muss. Dies betrifft sowohl Sound-, Grafik-, als auch spezielle Spiele-Hardware wie Joysticks oder Gamepads. Im Fall der Webcam kommt das DirectShow-SDK in Frage, welches kürzlich Teil des von Microsoft zur Verfügung gestellten *Platform SDK*²¹ geworden ist. Ein Blick in Beispielquellcodes zeigt, dass zur Programmierung von DirectShow ein größeres Verständnis des gesamten SDKs benötigt wird, um auf die Daten der Hardware zuzugreifen.

Eine Abhilfe verspricht ein Programmier-Framework namens PortVideo. Es bietet zusätzlich den Vorteil, das es plattformunabhängig ist und für C++ geschrieben wurde. Unter Windows verwendet es die Library DsVideoLib, was soviel heißt wie: *DirectShow Video Library*. Bei genauerer Betrachtung sieht PortVideo zu umfangreich aus, was daraus resultiert, dass alle betriebssystemgebundenen Funktionen gekapselt werden. Da das Projekt jedoch nur auf dem Windows-Betriebssystem lauffähig sein soll, ist DsVideoLib sehr gut geeignet. Nachteile sind das frühe Entwicklungsstadium, die noch gänzlich fehlende Dokumentation und die ungewisse Zukunft des OpenSource-Projekts, welches zur Zeit von einem Mitarbeiter der Universität Wien, Thomas Pintaric, entwickelt wird.

Obwohl jedoch DsVideoLib das Betreiben zweier baugleicher Kameras ausdrücklich unterstützt, ist es aufgrund der Logitech-Kameratreiber nicht gelungen jede Kamera individuell zu identifizieren, denn die abgefragte USB-ID unterscheidet

²⁰<http://www.microsoft.com/directx>, April 2006

²¹<http://www.microsoft.com/msdownload/platformsdk/sdkupdate/>, April 2006

sich lediglich von der ID der USB-Ports, die IDs der Kameras sind jedoch identisch. Dies hat zur Folge, dass die entwickelte Applikation nur mittels eines Workarounds startbar ist. Schließt man beide Kameras an und startet anschließend die Applikation, so werden zwar zwei Kameras gefunden, es wird jedoch versucht, aufgrund der uneindeutigen Zuordnung, zweimal die gleiche Kamera zu initialisieren, was zu Speicherzugriffsfehlern führt. Abhilfe schafft hierbei, die erste Kamera in den niedriger priorisierten USB-Anschluß zu stecken und danach die Anwendung zu starten. Nach Erkennen dieser Kamera kann die Zweite eingesteckt werden. Wenige Sekunden später wird diese vom System erkannt und von der Anwendung als zweite Kamera initialisiert. Von nun an läuft die Applikation bis zum Beenden normal ab - beide Kameras werden getrennt behandelt und angesteuert. Ein Neustart erfordert auch das Wiederholen der zuvor beschriebenen Prozedur.

3.4 Bild und Video speichern

Das Abspeichern von Bildern in verschiedenen Formaten lässt sich recht einfach implementieren. Für die meisten Formate gibt es entweder bereits in einigen Frameworks Speicherfunktionen oder man sucht sich eine passende Library zum jeweiligen Datei- bzw. Bildformat aus. Für das Speichern als raw-Format wurde eine eigene kleine Funktion implementiert, die ein Frame, unverändert vom Kamerapuffer in eine Datei schreibt und diese danach schließt. Funktionen um Bitmap- (Dateiendung *bmp*) oder JPEG-Dateien (Dateiendung *jpg* oder *jpeg*) zu speichern, werden von Qt bereitgestellt.

Zum Speichern eines Videos, dazu noch eines komprimierten, existieren sehr wenig Beispiele oder fertige Bibliotheken, was die Auswahl und damit die Entscheidung stark einschränkt. Eine OpenSource-Bibliothek die, wie im Abschnitt 2.4 auf Seite 15 festgelegt, einen MPEG4 komprimierten Videostrom abspeichert, ist *revelib*²². Sie ist für C++ geschrieben und kompatibel zum *Microsoft Visual Compiler*. Es ist sogar direkt möglich auch Sound zum Video abzuspeichern. Damit erfüllt die Bibliothek alle an sie gestellten Anforderungen.

Um nun einen von *DsVideoLib* gelieferten Frame auf der Oberfläche anzuzeigen und in einer XviD kodierten Datei zu speichern sind mehrere Schritte nötig. Zunächst werden die Daten, die sich in einem *unsigned char array* befinden in eine Qt kompatible Form gebracht. Dazu werden jeweils drei aufeinander folgende Werte zu einem *QRgb*²³ zusammengefügt und anschließend in ein zuvor angelegtes *QImage*²⁴ gespeichert. Dieses kann nun zur Anzeige in einem eigens angelegten *Widget* genutzt werden. Zur Programmierung von Qt wurde das Buch [JB04] benutzt.

Zum Speichern des Frames im Video muss es jedoch in einem anderen Format

²²<http://revel.sourceforge.net>

²³ein Objekt in Qt zum Speichern eines RGB-Wertes

²⁴ein Objekt in Qt, welches ein Bild mit Metainformationen speichert


```

void ECamera::getRawFrame(int* framebuffer)
{
    // Werte für Höhe und Breite sowie Frame auslesen
    int xMax = camera->getWidth();
    int yMax = camera->getHeight();
    int index = 0;
    unsigned char *picture = camera->getFrame();

    // 3 unsigned char Werte in einen Integer-Wert durch Verschiebung umrechnen
    for(int i=0; i<yMax; i++) {
        for(int j=0; j<xMax; j++) {
            framebuffer[index] = ((DWORD) ((BYTE) (picture[3*index+2]) |
            ((WORD) (picture[3*index+1]) << 8)) |
            ((DWORD) (BYTE) (picture[3*index])) << 16));
            index++;
        }
    }
}

```

Abbildung 3.2: Code der `getRawFrame(int*)`-Funktion um das Frame in ein *Integer Array* zu konvertieren

vorliegen, weswegen wieder auf das von *DsVideoLib* gelieferte Array zurückgegriffen werden muss. Nun werden wieder drei *unsigned char* Werte betrachtet und mittels Verschiebung zu einem *Integer* zusammengefasst. Diese Konvertierung wird durchgeführt, bevor jeweils ein Bild jeder Kamera zu einem Frame zusammengefügt werden. Mit dem Befehl *memcpy* werden Speicherbereiche, die eine Zeile eines Frames umfassen, im Reißverschlussverfahren in eine neues Array kopiert. Das bedeutet, dass angefangen mit dem linken Frame, sich eine Zeile aus dem linken Frame und anschließend eine Zeile aus dem rechten abwechseln. Das Array aus *Integern* beschreibt dann wieder einen Frame, der sich aus zwei nebeneinanderliegenden Frames zusammensetzt. Dieser wird an die Library übergeben. Einen Ausschnitt aus dem Code ist in Abbildung 3.2 zu sehen.

3.5 Aufbau und Ablauf der Anwendung

Die Anwendung ist zentral aufgebaut, ausgehend von der Oberfläche. Die *main*-Methode initialisiert zunächst die Geräte über die entsprechenden Klassen und die Benutzeroberfläche. Anschließend wird die Qt-GUI aufgerufen, die die weitere Kontrolle übernimmt. Ab jetzt wartet die Anwendung auf Benutzerinteraktion und reagiert darauf, indem die geforderten Funktionen der Objekte aufgerufen werden. Der modulare Aufbau der Klassen erlaubt eine unkomplizierte Erweiterung des Systems um zusätzliche Funktionen oder Geräte wie sie im Konzept in Kapitel 2 genannt werden.

In Abbildung 3.3 ist die Klassenstruktur der Anwendung zu sehen, jedoch ohne die Klassen des Qt-Frameworks. Die grau dargestellten Klassen sind nicht oder

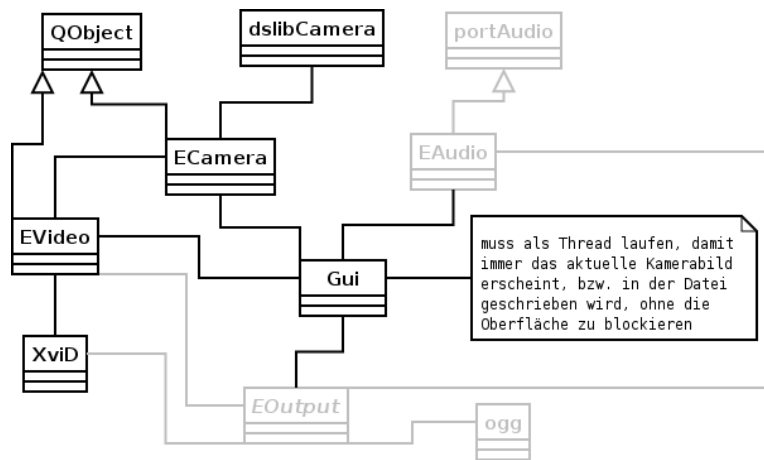


Abbildung 3.3: Klassenstruktur des Programms, alle grauen Bereiche sind nicht fertig implementiert

nicht fertig implementiert, aber bereits im Klassenkonzept vorgesehen.

Startet man nun die Anwendung, sollte bereits eine Webcam angeschlossen sein (dieser Workaround und seine Gründe wurden bereits in Abschnitt 3.3 auf Seite 22 erläutert). Im Abfragefenster finden sich bereits die Default-Einstellungen, die mit einem Klick auf *Ok* bestätigt oder an die Bedürfnisse angepasst werden. Man sollte Änderungen der Einstellungen jedoch stets für beide Kameras vornehmen, damit Bilder mit den gleichen Parametern geliefert werden. Bevor man fortfährt sollte die zweite Kamera mit dem PC verbunden werden. Nun folgt das Einstellungsfenster für die als nächstes gefundene Kamera. Nachdem hier mit den gleichen Einstellungen fortgefahren wird, erscheint die Oberfläche der Applikation (siehe Abbildung 3.4).

Grundsätzlich wurden alle Einstellungen, die eine Kamera betreffen auch auf der entsprechenden Seite der Anwendung platziert. In der unteren Mitte befinden sich die Aufnahmeeinstellungen und ein Button um die Aufnahme zu starten.

Einstellungen die die Aufnahme von Sound betreffen, sollten dann auch der jeweiligen Kamera zugeordnet werden, da die Mikrofone dort integriert sind und eine Aufteilung in rechts und links bei Stereo üblich ist. Sollten darüber hinaus digitale Quellen angebunden werden, sollten die betreffenden Einstellungen in Fenster ausgelagert werden, die sich erst bei Bedarf öffnen. Dies ermöglicht eine gute Übersicht sowie eine einfache und komfortable Bedienung.

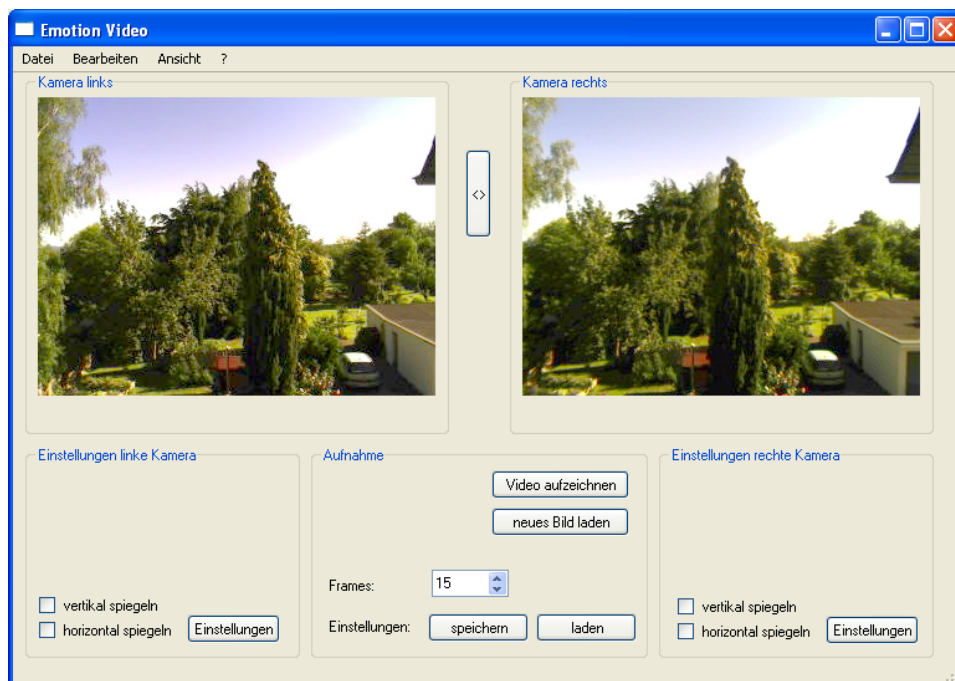


Abbildung 3.4: Die Oberfläche des Programms

4 Auswertung

Wie erste Tests zeigen konnten, bieten die Videos eine Menge Potential. Die Tiefenwirkung der Bilder ist, auch wenn die Auflösung der verwendeten Kameras eher beschränkt ist, gut erkennbar. Um die Videos auch auf einem Monitor betrachten zu können, wurde eine stereoskopische Brille verwendet. Diese Lorgnette, wie in Abbildung 4.1 zu sehen, ist aus transparentem Kunststoff hergestellt und besteht aus zwei Linsen, die den Blick nach außen lenken. So ist es möglich zwei nebeneinander platzierte Bilder als ein Stereobild zu betrachten.



Abbildung 4.1: Lorgnette aus transparentem Kunststoff um zwei nebeneinander liegende Bilder in Deckung zu bringen, Quelle: <http://www.3d-brillen.de>, Mai 2006

Zur Benutzbarkeit des Systems lässt sich sagen, dass eine Minimierung der technischen Geräte unumgänglich für einen praktischen Einsatz ist. Allgemein gilt, je kleiner und leichter, desto besser. Jedoch sollte dabei die Leistungsfähigkeit nicht zu kurz kommen. Im idealen Fall werden sehr kleine Kameras verwendet, die sich in einen Brillenbügel integrieren lassen. Ein Handheld-PC ersetzt den unhandlichen Laptop und findet leicht in einer Hosentasche und nicht mehr im Rucksack Platz. Dadurch lässt sich auch die Anwendung leichter bedienen, da der direkte Zugriff einfacher ist. Alternativ könnte die Anwendung auch schnurlos ferngesteuert werden, zum Beispiel per Bluetooth. Alle Peripheriegeräte sollten kabellos mit dem Anwendungsrechner kommunizieren und dabei eine Mindestlaufzeit von 10 Stunden haben. Nur so könnte das System in den im Abschnitt 2.5 vorgestellten Benutzerszenarien auch produktiv eingesetzt werden.

Am wichtigsten ist es den Nutzen und auch den Mehrwert der Anwendung, im Gegensatz zum konventionellen Video, zu untersuchen. Zwar klingt die Argumentation schlüssig, dass jemand viel lernt, wenn ihm auch viel zur Verfügung steht, jedoch kann es auch zu einer Überforderung oder zu Schwierigkeiten bei der Selektion der Informationen kommen. Auch wenn das System einen Vorteil bringen sollte, so muss untersucht werden, wie groß dieser ist und ob der damit verbundene Aufwand, wie Anschaffung, Entwicklung und Pflege, gerechtfertigt ist. Durch die vielen aufgezeichneten Daten kann es sowohl zu Speicherplatzproblemen, als auch zu einer Überflutung von Datenmaterial kommen. Eine angemessene Archivierung sowie eine Nachbearbeitung ist daher unerlässlich.

5 Fazit

Zum Abschluss lässt sich sagen, dass fast alle gesetzten Ziele erreicht wurden. Die Aufnahme von Sound wurde zwar im Klassenkonzept berücksichtigt, konnte jedoch nicht mehr vollständig implementiert werden. Eine Verbesserung des Systems könnte vor allem durch bessere Unterstützung der Hardware erreicht werden. Wünschenswert sind bessere Kamertreiber, die es erlauben, die Kameras exakt zu identifizieren. Der zweite Schritt wäre eine Aktualisierung des *Quickcam SDK* mit einer Unterstützung der neuen Hardware. Dies würde den Zugriff auf momentan unerreichbare Funktionen der Hardware ermöglichen.

Das entwickelte System bietet viele Möglichkeiten zur Erweiterung und zur Implementierung neuer Ideen. Die zukünftige Weiterentwicklung der Hardware, hin zu kleineren und leistungsfähigeren Geräten, wird die Benutzerfreundlichkeit besonders fördern.

Obwohl viele technische Schwierigkeiten bei der Entwicklung bestanden, welche die eigentliche Arbeit stark behinderten, war die Implementierungsphase dieser Arbeit eine Erfahrungsbereicherung. Es hat sich gezeigt, dass trotz vorhandener Standards, das Einbinden von Multimedia-Hardware schwierig bleibt. Einblicke in interdisziplinäre Bereiche, zum Beispiel in die Wahrnehmungspsychologie, waren ebenso interessant, wie das Erfassen der stereoskopischen Zusammenhänge. Vor allem die Einarbeitung in das Qt-Framework hat einen Nutzen, auch für die Zukunft, gebracht.

Literatur

- [Gol02] GOLDSTEIN, E. BR.: *Wahrnehmungspsychologie*. Spektrum Akademischer Verlag, 2002.
- [JB04] JASMIN BLANCHETTE, MARK SUMMERFIELD: *C++, GUI-Programmierung mit Qt3*. Addison-Wesley, April 2004.
- [Kuh99] KUHN, GERHARD: *Stereofotografie und Raumbildprojektion. Theorie und Praxis, Geräte und Materialien*. vfv-Verlag Gilching, 1999.
- [Mül06] MÜLLER, STEFAN. http://geri.uni-koblenz.de/ws0506/vrарfolien/09_rendering_2.pdf, Januar 2006.
- [PGZ04] PHILIP G. ZIMBARDO, RICHARD J. GERRIG: *Psychologie. Eine Einführung*. Pearson Studium, 16. Auflage, Juni 2004.
- [son06] http://www.daserste.de/wwiewissen/thema_dyn~id,ey5fka28tpdurv6n~cm.asp, Mai 2006.