Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Ausarbeitung Projekt - SoSe 2009

Arne Bernin

Räumliche Segmentierung mit Differenzbildern

Fakultät Technik und Informatik
Department Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Inhaltsverzeichnis

1 Einführung

Interaktion von Mensch und Maschine über die visuelle Erkennung von Gesten ist kein neues Thema. Weder für Filme aus Hollywood (Minority Report) , noch im Bereich der Informatik (siehe [Beale u. Edwards (1990)], [Davis u. Shah (1994)]). Ein großes Problem war bisher die Verfügbarkeit und vor allem der Preis von Hardware entsprechender Leistungsfähigkeit, zudem wurden oft Systeme mit am Körper zu tragender Gerätschaften, wie passive Marker, eingesetzt ([artrack (2009)]). Der von Microsoft angekündigte Controller (“Projekt Natal”, siehe [E3 (2009)]) geht hier eine Stufe weiter, er verwendet nur den Körper des Benutzers als “Eingabegerät”.

Abbildung 1: Quelle: [Clement (2009)]

Die Beliebtheit von Systemen wie der Nintendo Wii ([Nintendo (2006)]) mit ihrer Bewegungssteuerung zeigt, dass die Bereitschaft, andere Formen der Eingabe zu akzeptieren, bei Benutzern generell vorhanden ist. Zumindest im Bereich der Spielsteuerung.

Dabei ist die Gestenerkennung sicherlich nur ein Baustein einer umfassenden, multimodalen Interaktion mit der umgebenden, “intelligenten” Umgebung. Da die angekündigten Kamerasysteme (Microsoft, Sony) zur Bewegungs- und Gestenerkennung derzeit eben nur angekündigt sind, bleibt also die Frage, wie man dies mit verfügbarer Technik und möglichst geringem (finanziellen) Aufwand erreicht.

1.1 Zielsetzung im Kontext der Masterarbeit

Endgültiges Ziel ist die Entwicklung eines Systems, das sowohl (grobe) Gesten erkennt, als auch eine Schnittstelle für die Positionsbestimmung von Personen im Raum bietet, die von anderen Anwendungen genutzt werden kann, beispielsweise zur Ausrichtung von Richtmikrofonen für eine Sprachsteuerung. Den gesamten Bereich einer Wohnung ständig visuell nach Gesten abzusuchen erfordert einen immensen Aufwand. Besser ist es, schon vorher eine räumliche Segmentierung in (sich bewegende) Personen im Vordergrund, und den (meist) uninteressanten Hintergrund durchzuführen. Mit diesen Resultaten kann der Bereich stark eingegrenzt werden, der für eine weitere Betrachtung relevant ist.

1.2 Zielsetzung im Masterprojekt

Ziel dieses Masterprojektes ist die Erprobung von Techniken zur visuellen Segmentierung im räumlichen Kontext. Dies soll, wenn möglich, mit Standard-Hardware erfolgen, also mit Standard Rechner(n) und Kameras, die einen Netzwerk- oder USB-basierten Strom von Bildern liefern.

Dieses System soll ohne künstliche Marker oder zu tragende Hardware, wie beispielsweise Handschuhe, Reflektoren für Infrarot oder (Leucht-) Dioden auskommen.

2 Hardware

2.1 Kamerasysteme

Es gibt eine Vielzahl unterschiedlicher Kameras für PC-Systeme. Drei davon, wurden einer Evaluierung für das Projekt unterzogen. Die gewollte räumliche Einordnung verlangt den Einsatz mehrerer (mindestens zweier) Kameras um Entfernung und die Lage im Raum bestimmen zu können.

Playstation 3 Eye

Diese Kamera wird für die Spielekonsole Playstation 3 von Sony hergestellt. Sie wird zudem gerne im Rahmen von Projektionstischen eingesetzt ([MultitouchTable (2009)]) und bietet sowohl hohe Frameraten (bis zu 125 Bilder in der Sekunde) als auch einen niedrigen Preis.

Abbildung 2: Playstation 3 Eye, Quelle: [ps3eye (2009)]

Zu Testzwecken erfolgte der Umbau einer Kamera auf das Infrarotspektrum. Im Netz verfügbare Anleitungen (beispielsweise [Peau-Productions (2009)]) empfehlen dazu das Entfernen des verbauten Infrarot-Sperrfilters durch rohe Gewalteinwirkung. Nach Durchführung dieser Maßnahme zeigte sich jedoch, dass die Kamera durch das Fehlen des Filters ihre Fähigkeit verliert, per Autofokus die Bildschärfe zu garantieren. Abhilfe brachte in diesem Falle nur das Ersetzen des gesamten Objektives.

Ein Versuch mit einem Normallicht-Sperrfilter (als einfache Ausführung aus dem Innenleben einer Diskette hergestellt) brachte die Erkenntnis, dass dadurch zwar das Licht des sichtbaren Spektrums weitgehend ausgeschlossen wurde. Jedoch war die Helligkeit von (menschlichen) Körpern, die mit einem Infrarotscheinwerfer angestrahlt wurden, nicht ausreichend, um diese zu erfassen. Lediglich der Einsatz von direkt leuchtenden Infrarot-Dioden führte zu verwertbaren Ergebnissen, scheidet aber aufgrund der Anforderungen, keine Marker zu verwenden, aus.

Die Konsequenz aus diesen Versuchen war, die angedachte Verwendung von selbstgebauten Kameras im Infarotbereich zu verwerfen.

Aviosys CS-9070 IP Cam

Hintergrund dieses Testes war die mögliche Ausstattung des Labors im Living Place Hamburg mit diesen Kameras und die Klärung der Frage, ob eine zusätzliche Installation weiterer Kameras nötig ist. Diese Kamera der Firma Aviosys liefert einen Mpeg4-kodierten Bildstrom, der mit Hilfe des RTSP-Protokolls übertragen wird. Eine direkte Integration und Verwendung als Quelle ist via OpenCV mangels Vorhandensein einer Schnittstelle für Netzwerkströme im Framework nicht möglich. Über den Umweg von AVLD ([Avld (2008)]), einem virtuellen Videotreiber für Linux, war die Einbindung jedoch erfolgreich. Somit wurden die von der Kamera gelieferten 30 Frames pro Sekunde bei einer Auflösung von 1280x720 Pixeln (720p) erreicht. Dabei zeigte sich jedoch, dass der verwendete Doppelkern-Rechner vom verdreifachten Datenvolumen pro Kamera überfordert wurde. Sollten diese Kameras zum Einsatz kommen, ist ein (softwareseitiger) Filter zum Verringern der Auflösung vorzusehen oder der Einsatz mehrerer Rechner zur Auswertung der Kamerabilder.

Üblicherweise wird diese Kamera einen Datenstrom in voller Auslösung an ein zentrales System übermitteln. Die Weiterverteilung für unterschiedliche Zwecke (wie Usability-Tests, Gestenerkennung) erfolgt von dort.

Abbildung 3: Aviosys CS-9070 , Quelle: [cs9070 (2009)]

Logitech Quickcam 9000 Pro

Diese Kamera ist eine Standard-Webcam mit einer hohen Auflösung von 1600x1200 Pixeln (2MP). Die Maximale Framerate liegt bei 30 Frames pro Sekunde. In den praktischen Versuchen schwankte sie jedoch dynamisch im Wertebereich zwischen 10-30 Bilder in der Sekunde. Ob dies an der Kamera selber oder am Linux-Treiber lag wurde nicht offensichtlich. Aufgrund der hohen Auflösung und der damit verursachten Last des Systems bei der Bildverarbeitung wurde die Kamera nur zu Kompatibilitätstests herangezogen.

Abbildung 4: Logitech Quickcam Pro 9000 , Quelle: [quickcam (2009)]

2.2 Beurteilung

Die Playstation Eye ist für das Einsatzgebiet geeignet. Die möglichen 125 beziehungsweise 60 Frames in der Sekunde sind für eine reine Bewegungserkennung übertrieben, diese lässt sich auch mit 30 oder gar nur 15 Bildern erreichen. Für eine spätere Weiterverarbeitung der Bilder für die Gestenerkennung ist allerdings eine hohe Framerate eine wichtige Voraussetzung. Hier bietet die Playstation Eye genügend Reserven.

Ein mögliches Problem bei der Playstation Eye ist die relativ gering Auflösung. Dies kommt zwar der benötigten Rechenleistung in der Verarbeitung zu Gute, führt aber zu Problemen bei der Erkennung von Bewegungen über Entfernungen jenseits von circa fünf Metern, verursacht durch die geringe Anzahl sich ändernder Bildpunkte. Für die Tests spielte dies aber keine Rolle, ob es im Kontext des Living Place ein Problem ist, wird sich zeigen.

3 Software

3.1 Anforderungen

Die zu entwickelnde Software soll günstig im Bezug auf benötigte Resourcen und finanzielle Kosten (Opensource statt kommerzieller Bibliotheken) sein. Zudem ist eine Verarbeitung (möglichst) in Echtzeit anzustreben, das System soll sich selbst-adaptierend an veränderte Lichtverhältnisse anpassen. Wie schon Eingangs erwähnt ist das Ziel die Erfassung von Bewegungen im Raum und die Abbildung auf einen zweidimensionalen Raumplan.

3.2 Verfahren

Differenzbildung

Die Bewegungserkennung erfolgt mit Hilfe der fortlaufenden, dynamischen Differenz. Diese Technik beruht auf der Subtraktion (der Grauwerte) einander nachfolgender Bilder. Dazu wird das Kamerabild vorher in Graustufen umgewandelt. Die Differenzbildung kam schon in einem früheren Gemeinschaftsprojekt an der HAW zum Einsatz([Pressburger (2009b)], [Pressburger (2009a)]).

Bildklärung

Der nachfolgende Schritt ist die Elimierung des Grundrauschens des Kamerabildes mit Hilfe eines Schwellwert-Algorithmus. Der verwendete Schwellwert ist variabel und zur Laufzeit konfigurierbar. Als weiterer Schritt wird der Connected Components Algorithmus (siehe Bild 5, [Hopcroft u. Tarjan (1973)]) verwendet, um kleinere Störungen zu entfernen.

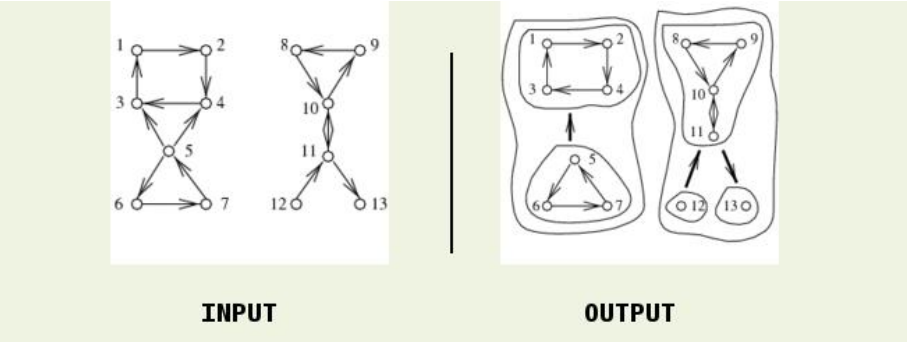

Abbildung 5: ConnectedComponents , Quelle: [connComp (2008)]

Überschneidungen

Der Connected-Components-Algorithmus liefert eine Liste von Flächen, die als Zusammenhängend erkannt wurden. Diese Flächen werden jeweils umgeben von einem einschließenden Rechteck. Die jeweiligen Mittelpunkte dieser Flächen werden auf ein eindimensionales Koordinatensystem (in diesem Falle die X-Achse) abgebildet.

Mit Hilfe dieser Koordinaten wird eine Projektion von der Kamera ausgehend auf eine planare Oberfläche durchgeführt, um die Koordinaten der Objekte an den Schnittpunkten zu bestimmen (siehe Abbildung 6). Zur Bestimmung der Schnittpunkte wird der intersection Algorithmus ([Cawsey (2006)]) eingesetzt.

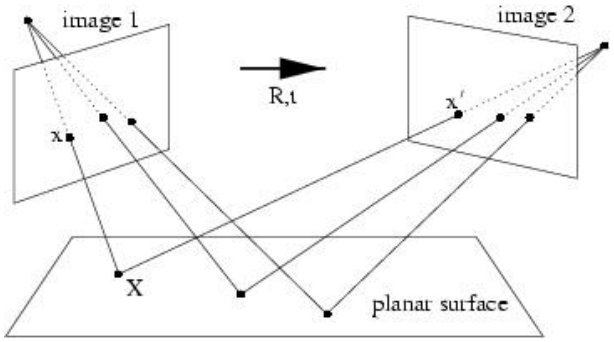

Abbildung 6: 2D-Triangulation , Quelle: [Hartley u. Zisserman (2000)]

Beurteilung

Die verwendeten Verfahren sind im Großen und Ganzen gut geeignet, die Segmentierung durchzuführen. Insbesondere verfügt die Bewegungserkennung mit Differenzbildung über eine schnelle Anpassung an (massive) Helligkeitsveränderungen. Außerdem verbraucht sie wenig Rechenzeit.

Es zeigte sich jedoch, dass sie anfällig gegenüber Reflektionen und Schatten ist, da die durch diese Effekte ausgelösten Bildveränderungen nicht von direkten Bewegungen differenziert werden können. Auch flackerndes Licht störte die Erkennung massiv.

Eine Abhilfe bietet hier beispielsweise das Einbeziehen von Farbinformationen (Hauterkennung), um zumindest Schatten auszuschließen. Ebenso ist in statischer Konfiguration das Definieren von räumlichen Bereichen, die von der Verarbeitung im System ausgenommen werden, möglich.

3.3 Programm

Das im Rahmen des Projektes in der Programmiersprache C entwickelte Programm “diffcam” zeigt die Ergebnisse der Verarbeitung in fünf verschiedenen Fenstern. Die Originalkamerabilder mit (roter) Hervorhebung der erkannten Bewegungsbereiche, sowie die Schwarz-Weiß Bilder nach erfolgter Verarbeitung (siehe Bild 7). Das letzte Fenster dient zur Anzeige der räumlichen Einordnung der erkannten Bewegung auf einem zweidimensionalen Feld (Abbildung 8).

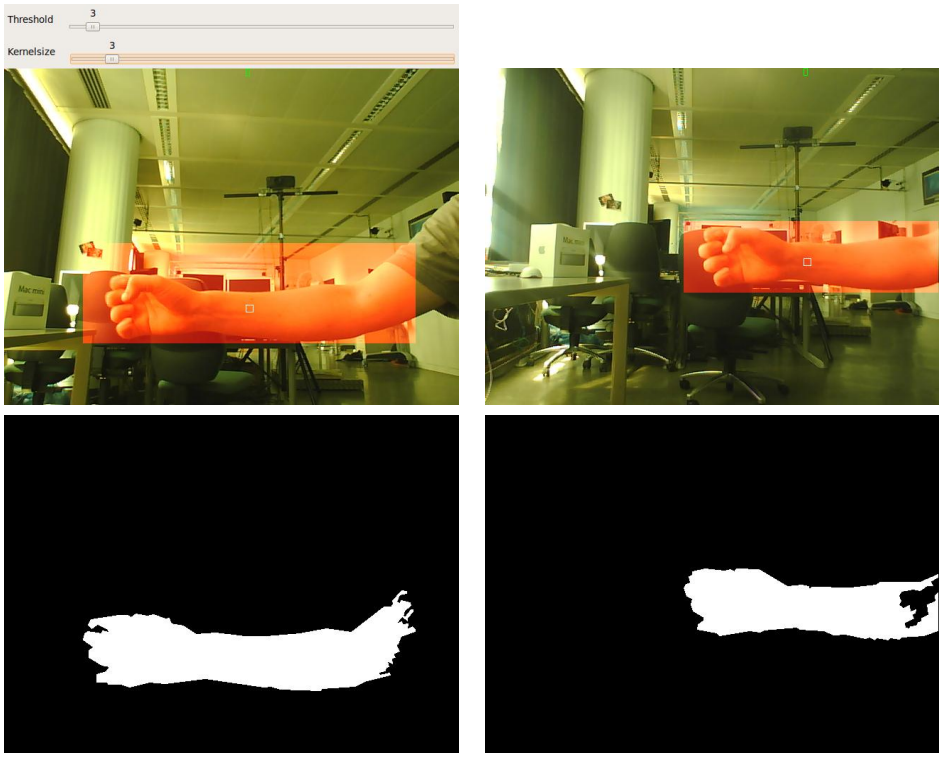

Abbildung 7: diffcam: Kamerabilder im Original und nach Bildklärung

Kalibrierung

Die Kalibrierung der räumlichen Anzeige erfolgt von Hand über das Positionieren der Kameras auf dem Übersichtsfenster. Dabei wird aufgrund der Position am Bildrand (oben, unten, rechts, links) auf die Ausrichtung des Blickwinkels der Kamera geschlossen. Die besten Resultate erzielte dabei eine Anordnung auf einer Sichtlinie, Analog zu einer Stereokamera.

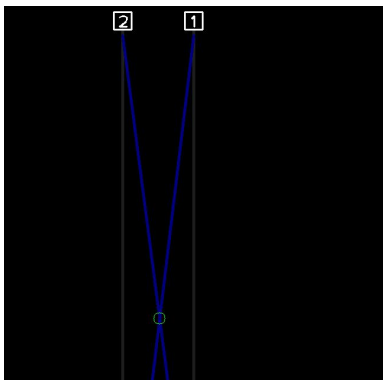

Abbildung 8: diffcam: Positionsbestimmung

Schwellwert und Kernelgröße für den Connected-Components-Algorithmus erfolgen über Regler, die besten Resultate ergaben sich bei einem Schwellwert von drei und einer Kernelgröße von drei.

Performance

Die verwendeten Algorithmen haben sich als ausreichend performant erwiesen, die CPU Last blieb auf dem verwendeten System bei zwei Kameras und einer Auflösung von 640x480 bei 30 Bildern in der Sekunde unter maximalen 80 Prozent der Auslastung eines CPU-Kernes. Weitere Untersuchungen zur Skalierung mit mehr als zwei Kameras auf einem System mit mehreren Kernen sind zur weiteren Beurteilung der Skalierung sinnvoll, ebenso wie die mögliche Verteilung auf mehrere Rechner. Die erstellte Applikation ist als Proof-of-Concept zu sehen und verfügt deshalb sicherlich über weitergehendes Optimierungspotential im Bezug auf die Verarbeitungsgeschwindigkeit.

3.4 Framework

OpenCV

OpenCV ([OpenCV (2009)]) ist eine ursprünglich von der Firma Intel entwickelte Bibliothek zur Umsetzung zahlreicher (ca. 500) relevanter Algorithmen aus dem Bereich Bildverarbeitung. Sie ist unter der BSD Lizenz sowohl für Windows, Linux sowie Mac OS X erhältlich.Als umfangreiche Dokumentation zu OpenCV steht ein Tutorial als Buch [Bradski u. Kaehler (2008)] zur Verfügung. Im Bereich der Bildverarbeitung mit OpenSource-Software bildet OpenCV den quasi-Standard. Es stehen Schnittstellen zu verschiedenen Sprachen (C, C++, Python) zur

Verfügung, die Implementation der Bibliothek erfolgte in der Sprache C. Ein besonderes Augenmerk bei der Entwicklung lag auf der Geschwindigkeit, dies zeigte sich auch im praktischen Einsatz. Zum Einsatz kamen die Versionen 1.0 sowie 1.1pre.

GpuCV

GpuCV [GpuCV (2009)] ist eine Umsetzung von Teilen des OpenCV Frameworks auf die GPU von Grafikkarten. Ziel ist eine Optimierung der Performance durch die speziell auf Bildverarbeitung ausgelegten Grafikprozessoren. Ein Test von GpuCV scheiterte zunächst am schwierigen Übersetzungsvorgang der Bibliothek auf dem verwendeten Linuxsystem.

Nach erfolgreicher Übersetzung scheiterte ein erster Testlauf an internen Initialisierungsfehlern der Grafikkarten GPU. Aufgrund dessen und dem zu erwartenden weiteren Arbeitsaufwand wurde von weiteren Versuchen erst einmal abgesehen. Die weitere Entwicklung dieses Projektes ist unklar, wäre allerdings interessant für einen Geschwindigkeitsvergleich mit OpenCV.

3.5 Kamera Treiber

Der Linux Kernel liefert ab Version 2.6.29 einen Treiber für die Playstation Eye mit, der jedoch nur über eingeschränkte Funktionalität verfügt. So ist kein Wechsel der Aufnahme-Auflösung , so wie der Framerate möglich. Aus diesem Grunde wurde für das Projekt ein modifizierter Treiber verwendet ([kubel (2009)]). Die Testläufe dieses Treibers lieferten die vorausgesagten 125 Frames pro Sekunde ebenso wie unterschiedliche Auflösungen.

3.6 Bildfehler

Im Verlauf der Verwendung des OpenCV-Frameworks in der aktuell stabilen Version (1.0) zeigte sich, dass wiederholt Bildfehler auftraten. Diese entstanden offensichtlich beim Auslesen des Bildes aus dem USB-Kamera-Treiber, bei der Verwendung eines anderen Programms zum Anzeigen des Bildstromes (luvcview) traten sie nicht auf. Die Bildfehler, in Form von Rauschen am oberen Bildrand, führten zu falsch-positivem Erkennen vermeintlicher Bewegungen. Ebenso war auffällig, dass die Häufigkeit der Fehler mit der Last des Systems und der eingestellten Bildfrequenz korrespondierte. Da die Fehler auch mit der Logitech QuickCam auftraten, war ein Fehler im Kamera-Treiber unwahrscheinlich. Ein Update der OpenCV-Bibliothek auf die Entwicklerversion (1.1.0 pre) führten zum Verschwinden des Fehlers.

3.7 Ausblick

Die folgenden Erweiterungen erscheinen aus meiner Sicht sinnvoll:

- **Erweiterung auf mehr (n) Kameras:** Die derzeitige Software erlaubt nur die Benutzung von zwei Kameras gleichzeitig. Durch eine zusätzliche Abstraktionschicht zwischen der Bewegungserkennung pro Kamera und der räumlichen Auswertung wird das System deutlich skalierbarer, da nur die ermittelten Koordinaten weitergegeben werden müssen. Die räumliche Auswertung kann so auf einer anderen Maschine erfolgen, die Kameras ebenso auf mehrere Rechner verteilt werden.
- **Automatische Kalibrierung:** Statt Positionierung von Hand auf dem Übersichtsfenster, die Verwendung eines Systems das die Kameraposition automatisch erkennt.
- **Statische Ausblendung bestimmter Bereiche:** Um Reflektionen an Spiegeln, Fenstern und Bildschirmen auszublenden.
- **Evaluierung weiterer Verfahren zum Ausschluss von Falsch-Positiven:** Beispielsweise die Nutzung von Hauterkennung über die (Haut-)Farbe.

4 Fazit

Das entwickelte System erfüllt grundsätzlich die angestrebten Ziele, es bietet sowohl eine Erkennung von Bewegung, als auch deren räumliche Einordnung und Visualisierung. Dabei hat sich sowohl die Auswahl der verwendeten Frameworks OpenCV, als auch der Playstation Eye Kamera als richtig erwiesen.

Es ist jedoch nicht perfekt. Gerade die Anfälligkeit des vorgestellten Systems gegenüber optische Effekten wie Reflektion und Schattenwurf sollte durch weitere Techniken verringert werden.

Auf diesem Projekt aufbauend bietet sich eine gute Grundlage, um Regionen, in denen eine Bewegung erkannt wurde, nach dort vorhandenen Gesten abzusuchen. Ob dieses quasi als Zusatzfunktion des bestehenden Kameraaufbaus möglich ist oder den Einsatz zusätzlicher Kameras mit spezieller Optik (Zoom, größere Auflösung) erfordert, ist noch zu evaluieren. Dies hängt nicht zuletzt von der Größe des zu betrachtenden Raumes ab.

Abbildungsverzeichnis

Literatur

[artrack 2009] *Advanced Realtime Tracking GmbH*. Verifiziert am 16.6.2009. http://ar-tracking.euVerizifizert:05.01.2008. Version: 2009

[Avld 2008] *Another Video Loopback Device*. Verifiziert am 25.8.2009. http://allonlinux.free.fr/Projets/AVLD/. Version: 2008

[Beale u. Edwards 1990] BEALE, R. ; EDWARDS, A.D.N.: Gestures and neural networks in human-computer interaction. In: *Neural Nets in Human-Computer Interaction, IEE Colloquium on*, 1990, S. 5/1–5/4

[Bradski u. Kaehler 2008] BRADSKI, Gary ; KAEHLER, Adrian: *Learning OpenCV*. O'Reilly, 2008

[Cawsey 2006] CAWSEY, Alison: *Line Intersection*. Verifiziert am 18.8.2009. http://www.macs.hw.ac.uk/~alison/ds98/node114.html. Version: 2006

[Clement 2009] CLEMENT: *Minority Report*. Verifiziert am 18.8.2009. http://historyofeconomics.files.wordpress.com/2008/12/minority-report.jpg. Version: 2009

[connComp 2008] *Connected Components*. Verifiziert am 18.6.2009. http://www.cs.sunysb.edu/~algorith/files/dfs-bfs.shtml. Version: 2008

[cs9070 2009] *Aviosys CS-9070*. Verifiziert am 18.8.2009. http://www.aviosys.com/ip_k9070_cs.html. Version: 2009

[Davis u. Shah 1994] DAVIS, J. ; SHAH, M.: Visual gesture recognition. In: *Vision, Image and Signal Processing, IEE Proceedings* - 141 (1994), Apr, Nr. 2, S. 101–106. – ISSN 1350–245X

[E3 2009] *Project Natal: Microsoft und die Revolution im Wohnzimmer*. Verifiziert am 16.6.2009. http://www.golem.de/0906/67480.html. Version: 2009

[GpuCV 2009] *GpuCV: GPU-accelerated Computer Vision*. Verifiziert am 18.6.2009. https://picoforge.int-evry.fr/cgi-bin/twiki/view/Gpucv/Web/. Version: 2009

[Hartley u. Zisserman 2000] HARTLEY, R. I. ; ZISSERMAN, A.: *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521623049, 2000

[Hopcroft u. Tarjan 1973] HOPCROFT, John ; TARJAN, Robert: Algorithm 447: efficient algorithms for graph manipulation. In: *Commun. ACM* 16 (1973), Nr. 6, S. 372–378. http://dx.doi.org/http://doi.acm.org/10.1145/362248.362272. – DOI http://doi.acm.org/10.1145/362248.362272. – ISSN 0001–0782

[kubel 2009] KUBEL, Vw: *PS3 Eye on Linux*. Verifiziert am 18.8.2009. http://wiki.nuigroup.com/PS3_Eye_on_Linux. Version: 2009

[MultitouchTable 2009] *Mulit-touch Blog*. Verifiziert am 18.8.2009. http://userfolio.com/multitouch/. Version: 2009

[Nintendo 2006] NINTENDO: *Nintendo Wii*. Verifiziert am 18.8.2009. http://wii.com. Version: 2006

[OpenCV 2009] *Welcome - OpenCV Wiki*. Verifiziert am 18.6.2009. http://opencv.willowgarage.com/wiki/. Version: 2009

[PeauProductions 2009] PEAUPRODUCTIONS: *Video Tutorial - PS3 Eye Camera: Removing IR Blocking Filter, Installing Visible Blocking Filter (BandPass), and IR Light Tests*. Verifiziert am 28.8.2009. http://nuigroup.com/forums/viewthread/4189/. Version: 2009

[Pressburger 2009a] PRESSBURGER, Julia: *Digital Art Design*. Verifiziert am 18.6.2009. http://users.informatik.haw-hamburg.de/~ubicomp/projekte/master08-09-proj/pressburger/bericht.pdf. Version: 2009

[Pressburger 2009b] PRESSBURGER, Julia: *Interaktive Kunst*. Verifiziert am 18.6.2009. http://users.informatik.haw-hamburg.de/~ubicomp/projekte/master08-09/pressburger/bericht.pdf. Version: 2009

[ps3eye 2009] *Playstation 3 Eye*. Verifiziert am 18.8.2009. http://www.beefjack.com/blog/news/ps3/pseye-facial-recognition/. Version: 2009

[quickcam 2009] *Logicam Quickcam Pro*. Verifiziert am 18.8.2009. http://macbitz.files.wordpress.com/2009/06/logitech-quickcam.jpg. Version: 2009