Echtzeiterkennung von Gesten zur Interaktion mittels Consumer-Hardware

Dipl.-Inf. Torsten Bierz
Prof.-Dr. Achim Ebert
International Research and Training Group
Technische Universität Kaiserslautern
Postfach 3049, 67653 Kaiserslautern
Tel.: +49 (0) 631 / 205 – 3501 | - 3502
Fax: +49 (0) 631 / 205 - 3270
bierz@informatik.uni-kl.de
ebert@informatik.uni-kl.de

Prof.-Dr. Jörg Meyer
University of California, Irvine
644E Engineering Tower - Irvine, CA 92697-2625, USA
Tel.: +1 949 824 - 9321
Fax: +1 949 824 - 3203
jmeyer@uci.edu

Zusammenfassung

Seit den letzten Jahren rückt die virtuelle Realität und die damit verbundenen virtuellen Welten immer mehr in das Blickfeld der Öffentlichkeit und der wissenschaftlichen Arbeiten. Die inzwischen mannigfaltigen immersiven Visualisierungssysteme und deren Interaktionsmöglichkeiten stellen sowohl für ungeübte Benutzer als auch für erfahrene Anwender eine große Hürde dar. Gerade die Mensch-Maschine Interaktion (HCI) erfordert einfache, intuitive und leicht zu erlernende Eingabe- und Interaktionsroutinen.

Eine Lösung dieses Problems bilden so genannte „Tracking Systeme“, die dem Benutzer eine möglichst intuitive und freie Navigation oder Interaktion ermöglichen. Unglücklicherweise sind gerade solche Systeme meist sehr teuer und erfordern häufig zusätzliche Hilfsmittel wie Marker oder Datenhandschuhe. Diese zusätzlichen Geräte müssen meist erst vor deren Verwendung kalibriert werden. Des Weiteren können sie die persönliche Freiheit des Benutzers beeinträchtigen. Um diese Problematik zu vermeiden und dem Anwender eine einfache, intuitive und robuste Möglichkeit zur Interaktion zu bieten, greifen immer mehr Forscher und

Industrieunternehmen auf teure optische Tracking Systeme zurück. Zu Präsentationszwecken oder zur Mitnahme auf Konferenzen sind diese Systeme allerdings leider meist nur bedingt portabel. Daher besteht eine Bestrebung darin, möglichst günstige und universell kompatible Hardware zu diesem Zweck zu verwenden. Hierzu bieten sich meist schon einfache Consumer Hardware wie Webcams oder digitale Kameras an, die bedingt durch die ihre Schnittstellen (FireWire oder USB) universell einsetzbar sind und zudem, da sie für den Heimbedarf produziert werden auch kostengünstig sind.

Für die Interaktion selbst sollten wie bereits erwähnt keine aufwendigen und unnötigen Hilfsmittel verwendet werden, um den Benutzer nicht von vorne herein zu verunsichern oder zu verwirren. Gerade dafür eignet sich die Gestenerkennung mittels optischen Erkennungssystemen. Diese Systeme liefern eine bestimmte Anzahl Bilder pro Sekunde; bei low-cost oder Konsumerkameras sind dies zwischen 25 und 30 Bilder pro Sekunde. Für die meisten Interaktionen ist diese Anzahl der Bilder vollkommen ausreichend.

Um allerdings eine effiziente und somit echtzeitfähige Interaktion zu ermöglichen, muss die Zeit zwischen aufgenommenem Bild und resultierender Geste oder Pose so gering wie möglich gehalten werden. Um dies zu gewährleisten, können heutzutage viele Berechnungen, die mit jedem einzelnen Bild oder Bildpunkt erfolgen müssen, parallelisiert werden. Dazu werden die Berechnungsschritte auf dem Prozessor der Grafikkarte anstelle des herkömmlichen Prozessors verlagert, um durch deren Parallelisierungsmechanismen eine schnelle und effiziente Berechnung der Ergebnisse zu ermöglichen.

Aufgrund dieser Erkenntnisse wurde ein System entwickelt, welches die oben aufgeführten Punkte aufgreift, und in ein echtzeitfähiges System zur Interaktion mittels markerlosem Tracking vereint. Um die Echtzeitfähigkeit zu gewährleisten, werden hierzu die Berechnungsmethoden auf der Grafikkarte implementiert.

Schlüsselwörter

Tracking, markerless tracking, Interaction, GPU, Computer Vision.

1 Stand der Forschung

In der heutigen Zeit sind Maschinen aus dem täglichen Produktionsvorgängen nicht mehr wegzudenken. Die Arbeitsplätze sind meist Desktoprechner und verfügen über ein bis zwei Displays. Die Interaktion erfolgt dort über Eingabegeräte wie die Tastatur, die Mouse oder Joysticks. Für Präsentationen oder Demonstrationen von Produkten und Forschungsergebnissen oder zur Diskussion von Problemen mit mehreren Personen werden allerdings größere Displaysysteme wie PowerWalls, Tiled Displays oder CAVEs [CSD92] benötigt, um die Informationen für alle ersichtlich und in angemessener Weise präsentieren zu können. Die herkömmlichen Eingabegeräte wie Tastatur, Mouse oder Joystick sind dort oft ungeeignet, da sie einen festen Untergrund für die Eingabe benötigen, was wiederum in der Einschränkung des Aktionsraums resultiert. Gerade bei großen hochauflösenden Displaysystemen liegt ein weiteres Problem in der begrenzten Auflösung der Mouse, d.h. dass der Benutzer die Mouse einige Male neu ansetzen muss, um somit z.B. eine Bewegung vom einem Ende des Displaysystems zum gegenüber liegenden zu realisieren. Ein weiterer bei Tiled Displays auftretender Effekt ist der Mouse Ether [BCH04]. Dieser entsteht durch die Rahmen zweier anliegender Displays, in dem sich die Mouse anders verhält beziehungsweise in einer gewissen Art und Weise springt. Gerade bei diagonalen Mouse Bewegungen ist dies sehr deutlich zu erkennen [BCH04].

Aus dem Grunde der einfacheren Navigation und Interaktion vor solchen Displaysystemen werden nach Möglichkeiten für eine einfachere und vor allem Dingen intuitivere Steuerung und Interaktion gesucht. Multiple Eingabegeräte wie zum Beispiel Stäbe (so genannte Wands [ABC04], [CaB03]), Laser Pointer ([ON01], [ATK05]), die Cubic Mouse ([FrP00]), Dragonfly ([SR03]) oder SOAP ([BSW07]) konkurrieren gegen verschiedene Arten von Datenhandschuhe [KaT99], mobilen Haptik-Lösungen wie das Immersion GraspPack™ [Imm07] oder den HapticMaster ([LFR02], [FCS07]). Viele dieser Geräte sind von der Handhabung her unbequem oder rufen durch ihr Gewicht über einen längeren Nutzungszeitraum einen Ermüdungs- oder Erschlaffungseffekt hervor. Zudem schränken einige bedingt durch ihren stationären Einsatz oder durch mitgeführte Kabel die Bewegungen des Benutzers und den Interaktionsfreiraum ein.

Daher ist es sinnvoll sowie erstrebenswert, die Interaktion möglichst einfach und unabhängig von Geräten zu gestalten. Aus diesem Grunde kommen immer häufiger Tracking Systeme und im speziellen optische Tracking Systeme zum Einsatz, die dazu dienen, die Position des Benutzers zu erkennen und diese Informationen in angemessener Form zu verarbeiten. Dabei muss zwischen Systemen mit und ohne zusätzliche Markierungen, den so genannten Markern unterschieden werden. Die Marker erleichtern zwar das Auffinden von Personen oder Objekten, müssen

aber zum Beispiel an bestimmten Stellen befestigt werden, was wiederum vor dem Einsatz des Systems ein „Aufkleben“ der Marker und des weiteren meist eine Kalibrierung erfordert. Dieser Nachteil wird bei Systemen ohne Marker vermieden. Beispiele für solche Systeme sind die Verwendung der Blickrichtung des Benutzers zur Navigation und Fokussierung. Dabei erzeugt ein längerer Focus eine Änderung des Fokus oder der Darstellung der Szene [BDD03]. Diese Erkennung der Augen beziehungsweise der Blickrichtung und die damit verbundenen Interaktion wird als „Gaze-Tracking“ bezeichnet. Einer der Nachteile oder Gefahren dieser Technik ergibt sich bedingt durch die Natur des Menschen; er fokussiert meist nicht ausschließlich das Objekt oder die Region, die ihn interessiert (Region of Interest (ROI)), sondern er schweift auch öfters von seinem „Ziel“ ab, was somit zu unschönen und meist unerwünschten Nebeneffekten führen kann [Zha03].

Viele Tracking Verfahren, die auf Marker verzichten, müssen die Informationen bezüglich der Personen oder der sich bewegenden Objekten aus verschiedenen Hilfstechniken oder Methoden selbst generieren. Eine der grundlegenden ist die Verwendung eines einfarbigen Hintergrundes, um so im späteren Bild die Person oder das Objekt erkennen und separieren zu können. Als ganz typisches Beispiel dient die in der Filmindustrie oder in CGI’s (Computer generated images) eingesetzte Technik des „Blue Screens“ [Wik07]. Eine weitere Methode ist die Trennung von Vor- und Hintergrund. Dazu wird das Hintergrundbild aufgenommen und vom dem aufgenommen Bild subtrahiert [MZK01]. Daraus entsteht dann als Resultat ein Bild mit der sich darauf befindenden Person oder des Objektes. Problematisch ist dabei ein sich verändernder Hintergrund, zum Beispiel durch Beleuchtung oder durch bewegte Objekte wie Sträucher oder Bäume im Wind, was in einem fehlerhaften Ergebnisbild resultiert. Andere Verfahren verwenden Differenzbilder von mehreren Folgebildern, um somit die Region mit den Bewegungen und die Personen zu erkennen. Bewegt sich allerdings ein Objekt über längere Zeit nicht, wird es nicht mehr erkannt. Um dies zu verhindern werden dazu Filter wie zum Beispiel Kalman eingesetzt. In dem Fall des implementierten Systems kann auf diese Segmentierung des Hintergrundes verzichtet werden, da die Separation über die Erkennung von Haut realisiert wird.

Viele Forscher und Wissenschaftler gehen inzwischen auch dazu über handelsübliche Webcams anstelle von teuren optischen Tracking Systemen zu verwenden. Diese liefern meist Farbbilder in Auflösungen von 640x480 Pixel und arbeiten mit 25 Bildern pro Sekunde. Es fällt somit in sehr kurzer Zeit eine große Menge von Daten an, die verarbeitet werden müssen. Dies ist gerade dann problematisch, wenn die Interaktion echtzeitfähig soll. Das bedeutet, dass zum Beispiel zwischen einer erkannten Geste und der resultierenden Aktion keine große Zeitspanne liegen darf, da sonst der Benutzer zum einen nicht weiß, ob seine Geste erkannt worden ist und zum anderen aufgrund der Verzögerung keine wirklich gute Kontrolle

über seine Aktionen oder Interaktionen mehr verfügt. Um die Interaktion echtzeitfähig zu halten, bietet sich die parallele Verarbeitung auf der GPU an.

Die folgenden Kapitel beschreiben den Aufbau und die Details zu dem entwickelten System. Kapitel 2 erklärt dazu die notwendigen Berechnungsschritte für die Erkennung der Gesten aus dem Bild der Videokamera. Das dritte Kapitel beschreibt Implementierungsdetails. Anschließend werden danach die Ergebnisse diskutiert. Abschließend folgt der Ausblick auf zukünftige Entwicklungen und Arbeiten.

2 Markerlose Interaktion

Dieses Kapitel befasst sich mit den verschiedenen Schritten von der Aufnahme des Bildes bis zur Erkennung der Geste. Im Einzelnen werden dazu die Schritte des Preprocessings, der Erkennung der Haut und der Kontur sowie das Clustern mit der anschließenden Gestenerkennung erläutert.

2.1 Preprocessing

Vorab muss zwischen den Begriffen der Geste und er Pose differenziert werden. Als Pose wird die erkannte Hand in einem Bild bezeichnet. Eine Geste setzt sich aus mehreren aufeinander folgenden Posen zusammen.

In einem ersten Schritt werden verschiedene Posen aus der Datenbank ausgelesen und daraus hierarchische Momente aufgebaut. Bevor diese erläutert werden können, muss zuerst der Begriff der Momente geklärt werden.

Momente in der Bildverarbeitung oder Computer Vision sind in einem bestimmten Verhältnis gewichtete Mittelwerte über die Pixel Farbwerte. Sie werden verwendet, um Bilder oder Teilbereiche zu beschreiben; in diesem Fall handelt es sich um eine Möglichkeit nach einer Segmentierung die Gesten beschreiben zu können. Um eine gute Aussage für den Vergleich zwischen der aktuellen im Bild ausgeführten Geste und den Gesten in der Datenbank treffen zu können, müssen die zu berechneten Bildmomente translations-, skalierungs- und rotationsinvariant sein. Die Berechnung erfolgt mittels des Berechnungsschemas von Flusser [Flu00]. Die Formeln zur Berechnung sind:

$$\psi_1 = \mu_{20} + \mu_{02}$$
$$\psi_2 = (\mu_{30} + \mu_{12})^2 + (\mu_{21} + \mu_{03})^2$$
$$\psi_3 = (\mu_{20} - \mu_{02})((\mu_{30} + \mu_{12})^2 - (\mu_{21} + \mu_{03})^2) + 4\mu_{11}(\mu_{30} + \mu_{12})(\mu_{21} + \mu_{03})$$
$$\psi_4 = \mu_{11}((\mu_{30} + \mu_{12})^2 - (\mu_{03} + \mu_{21})^2) - (\mu_{20} - \mu_{02})(\mu_{30} + \mu_{12})(\mu_{03} + \mu_{21})$$

$$\psi_5 = (\mu_{30} - 3\mu_{12})(\mu_{30} + \mu_{12})((\mu_{30} + \mu_{12})^2 - 3(\mu_{03} + \mu_{21})^2) + (3\mu_{21} - \mu_{03}) * (\mu_{21} + \mu_{03})(3(\mu_{30} + \mu_{12})^2 - (\mu_{03} + \mu_{21})^2)$$

$$\psi_6 = (3\mu_{21} - \mu_{03})(\mu_{30} + \mu_{12})((\mu_{30} + \mu_{12})^2 - 3(\mu_{21} + \mu_{03})^2) - ((\mu_{30} - 3\mu_{12}) * (\mu_{21} + \mu_{03})(3(\mu_{30} + \mu_{12})^2 - (\mu_{21} + \mu_{03})^2)$$

wobei

$$\mu_{pq} = \sum_x \sum_y (x - x_c)^p (y - y_c)^q f(x, y).$$

μ_{pq} sind hierbei die Zentralmomente und (x_c, y_c) der Centroid, der sich aus $(x_c, y_c) = \left(\frac{M_{10}}{M_{00}}, \frac{M_{01}}{M_{00}} \right)$ berechnet. *f(x,y)* ergibt sich aus dem Wert des Pixels an der Position *(x,y)*. Die M_{ij} sind wie folgt definiert:

$$M_{ij} = \sum_x \sum_y x^i y^j f(x, y)$$

Im Falle der Posen in der Datenbank kann dies wie bereits erwähnt in effizienter Weise bereits im Preprocessing erfolgen und muss nicht mehr unnötig während der Laufzeit neu berechnet werden. Dadurch müssen die Berechnungen nur einmal stattfinden und die Ergebnisse liegen direkt zum Vergleich vor.

Um eine höhere Erkennungsrate zu erhalten, werden die Bilder aus der Posendatenbank hierarchisch skaliert, um somit bessere, vom Benutzer frei wählbare hierarchische Tiefe für den Vergleich zu erhalten (Abbildung 1). Die hierarchischen Momente sind also nichts weiter als die Momente berechnet aus den Posen in unterschiedlichen Auflösungen.

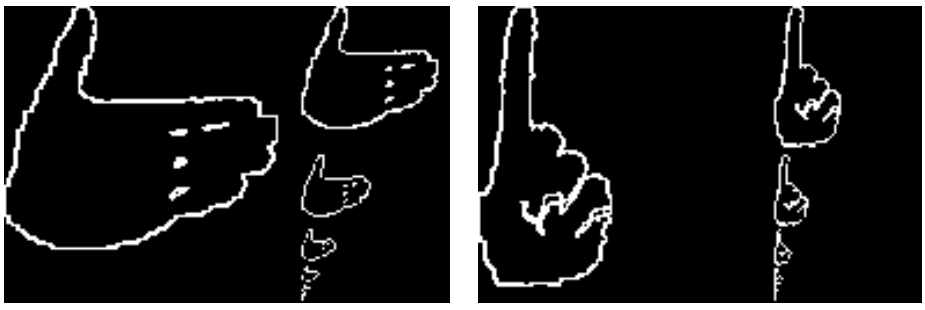

Abbildung 1: Hierarchische Darstellung der Skalierung zweier Posen

2.2 Erkennung der Haut

Für die Erkennung oder Detektion von Haut existieren verschiedene Möglichkeiten. Sie reichen über Pixel basierte Verfahren mittels RGB, normalisierten RGB Farbwerten, HSV und HSI Farbräumen, YCrCb, TSL Farbraum, über wahrscheinlichkeitsorientierten Klassifikationen basierend auf Testdatensätzen, neuronale Netzwerke bis hin zu kompakten generalisierten und parametrisierten Hautmodellen aus Trainingsdaten. Ein guter Überblick über die verschiedenen Möglichkeiten ist zu finden unter [KMB07] and [MSP03].

Zur Erkennung und Klassifikation von Haut wurde das Verfahren von Gomez und Moralez [GM02] verwendet. Dabei werden zur Ermittelung der Hautfarbe normalisierte RGB Farben betrachtet, da diese keine weitere Umrechnung in andere Farbmodelle benötigen und durch die Normalisierung die ambiente Beleuchtung reduziert wird. Die Entscheidung, ob ein Pixel als Haut oder nicht Haut erkannt wird, ergibt sich aus den folgenden Bedingungen:

$$\begin{aligned} \frac{r}{g} &> 1.185 \quad \textit{and} \\ \frac{r*b}{(r+g+b)^2} &> 0.107 \quad \textit{and} \\ \frac{r*g}{(r+g+b)^2} &> 0.112 \end{aligned}$$

Wenn diese drei Bedingungen erfüllt sind, wird der entsprechende Bildpunkt als „Haut“ klassifiziert. Falls sie nicht erfüllt sind, wird das entsprechende Pixel als „nicht Haut“ klassifiziert. Diese Klassifikation wird später hinaus auf der GPU verwendet, um ein Binärbild des aktuellen Bildes aus der Kamera zu erzeugen.

2.3 Entfernung von Rauschen und Kantenextraktion

Ein Problem bei Low-Cost Kamera Systemen bildet die Störanfälligkeit durch Bildrauschen. Eine Möglichkeit das aufgenommene Bild zu verbessern, ist das Weichzeichnen mittels eines Gausfilters. Allerdings gehen bedingt durch den Aufbau des Filters Informationen verloren. Daher wurde anstelle des Verminderns des Rauschens mittels eines solchen Filters, das so genannten Opening (siehe Abbildung 2) verwendet, was eine Kombination aus Erosion und Dillatation darstellt angewendet [Jae02]. Da nach der Klassifikation der Haut nur noch ein Binärbild vorliegt, fallen zum Beispiel isolierte Pixel mit dem Wert „1“ (rot umrandetes Pixel in Abbildung 2) weg.

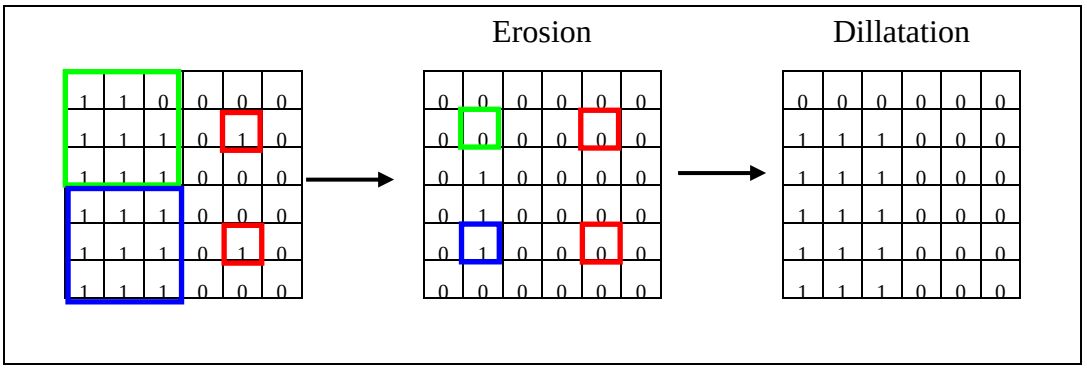

Abbildung 2: Opening mittels einer 3x3 Filtermaske bei einem Binärbild

Nachdem diese Operation stattgefunden hat, müssen für den späteren Vergleich der Posen das Bild auf die Kanten reduziert werden. Zu dieser Extraktion existieren verschiedene Möglichkeiten, wie zum Beispiel gradientenbasierte Methoden, Laplace, Sobel oder Canny [Jae02]. Da das Bild zu dem Zeitpunkt als Binärbild vorliegt, wurde ein anderer Weg dazu gewählt: Mittels eines Hit-Miss-Operators wird das Bild ausgedünnt. Der Operator prüft dabei die 4er Nachbarschaft eines gesetzten Pixels („1“) auf das Auftreten von „0“. Sobald die erste „0“ gefunden ist, bricht der Operator ab. Wird keine „0“ gefunden, erhält das aktuelle Pixel den Wert „0“ (siehe Beispiel (b) in Abbildung 3).

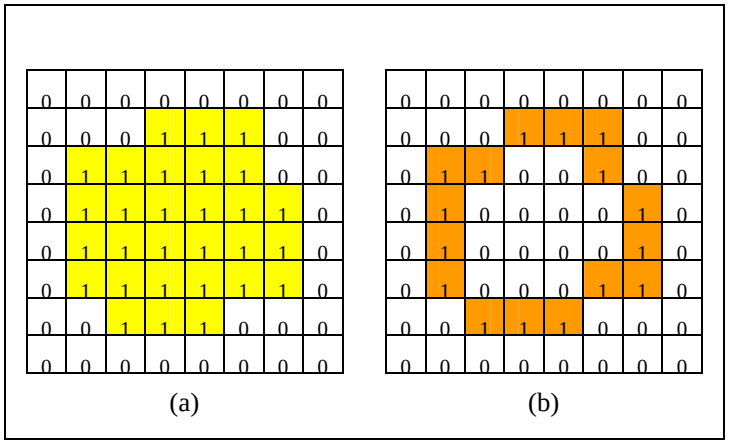

Abbildung 3: Ausdünnen mittels eines Hit-Miss Operators (a) Ausgangsbild, (b) Anwendung des Hit-Miss Operators

2.4 Clustering

Die Zuordnung in Cluster erfolgt dann im weiteren Schritt durch Region Growing. Eine Beschreibung des Region Growings ist zu finden unter [Ver91]. Dabei wird das Bild in der Weite und Höhe bis zum Auffinden eines nicht selektierten Kan-

tenpixels durchlaufen. Dieses Pixel wird nun als Startpixel für den Algorithmus markiert und beginnt rekursiv die Nachbarpixel zu durchlaufen, um somit die komplette Kontur erhalten zu können. Dabei wird die 8er Nachbarschaft betrachtet, d.h. alle umrandeten Pixel. Er terminiert, wenn die Liste der zu besuchenden Pixel leer ist. Aus diesem Algorithmus resultiert nun die Bounding Box für die zu erkennende Pose, die als Grundlage für die Momente in der nachfolgenden Gestenerkennung verwendet wird.

2.5 Erkennung der Posen

Die nun bekannten Cluster können nun für die Berechnung der hierarchischen Momente verwendet werden. Diese werden dann im nächsten Schritt mit denen im Preprocessing Schritt bereits berechneten Werte für die Momente innerhalb einer Epsilonumgebung verglichen. Liegen die Werte des zu untersuchenden Bildausschnittes in diesem Bereich, wird diese Region als die bekannte Pose betrachtet und dementsprechend verarbeitet. Abbildung 4 und Abbildung 5 zeigen links das Originalbild und rechts das verarbeitet und analysierte Bild mit der erkannten Pose. Dabei wird ein Objekt zum Beispiel als „selektiert“ betrachtet, wenn die Pose innerhalb von drei aufeinander folgenden Bildern zu erkennen war. Dies soll als Sicherung vor vermeintlichen Benutzerfehlern durch zu langsame Interaktion dienen.

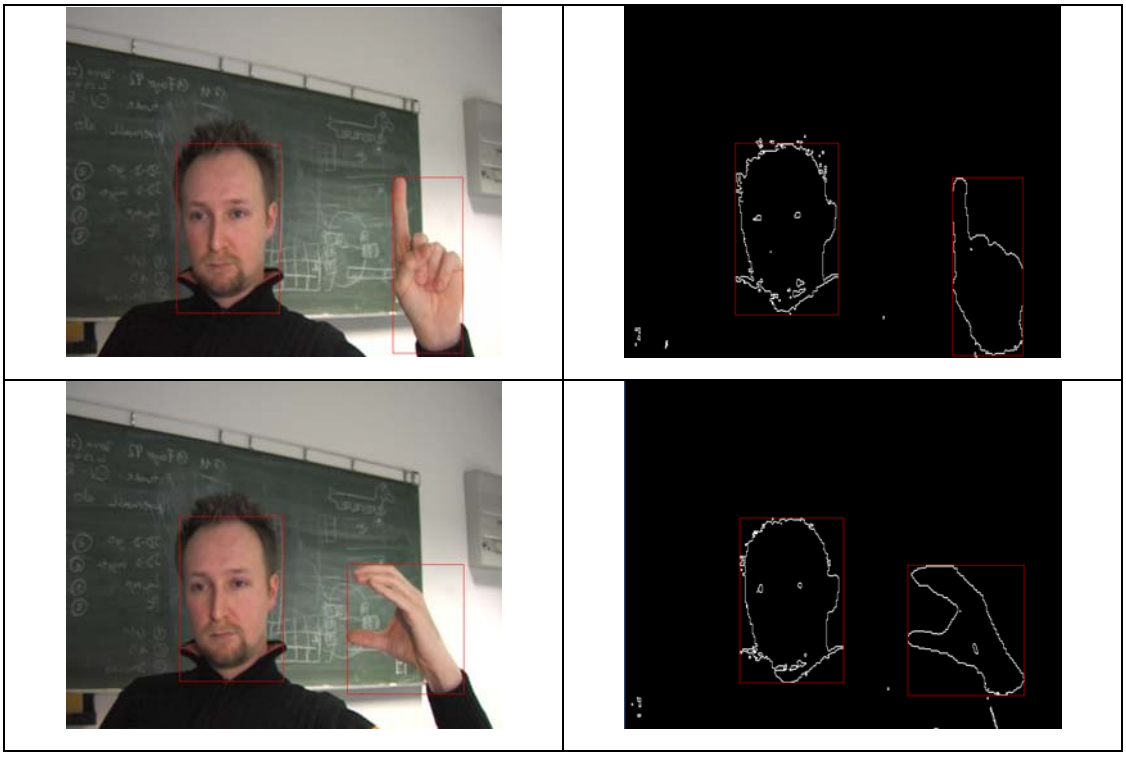

Abbildung 4: Beispielbilder mit einer Pose zum Zeigen (obere beiden Bilder) und Greifen (untere beiden Bilder)

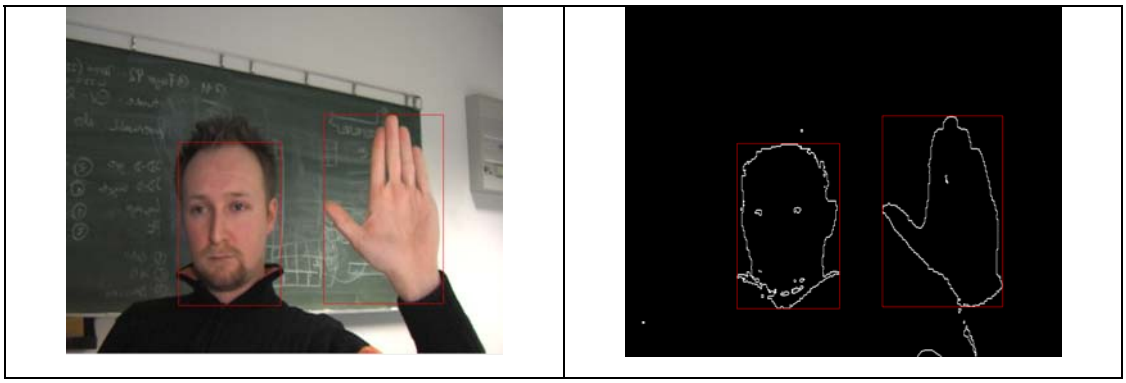

Abbildung 5: Beispiel für die implementierte Stopp Pose (untere Bilder)

3 Implementierung

3.1 Hardware / Software

Die verwendetet Kamera ist ein handelsüblicher Camcorder der Marke Sony mit einer Auflösung von 720x576 Bildpunkten und 25 Bildern pro Sekunde. Um den Datendurchsatz zu gewährleisten, wurde die Kamera über die FireWire Schnittstelle angesprochen.

Die Rechner für das System ist ein Pentium mit 3 Gigahertz und zwei Gigabyte RAM. Als Grafikkarte kommt eine Nvidia 6800 GT zum Einsatz. Aufgrund von Performancevorteilen, der effizienteren Speicherverwaltung und der Ansteuerung für die Kamera wurde Linux anstelle von Windows verwendet. Als Programmiersprache wurde C++ mit den CG Toolkit Nvidia [Nvi07] verwendet.

3.2 Verwendung der GPU zur effizienteren Berechnung

Die Geschwindigkeit und somit die Echtzeitfähigkeit spielt im Hinblick auf die Interaktion eine sehr große Rolle. Da sich viele der Algorithmen und Filter parallelisieren lassen, kann die Rechenleistung beziehungsweise die Möglichkeit der parallelen Verarbeitung der Grafikkarte effizient eingesetzt werden. Gerade die Algorithmen und Verfahren zur Bildverarbeitung können meist einfach realisiert und für die Anwendung auf der Grafikkarte umgeschrieben werden. Der Einsatz des Grafikprozessors für solche Berechnungen wird als „General Purpose Computation on Graphics Processing Unit“ kurz GPGPU ([FuM04], [FM04]) bezeichnet. Um diesen Vorteil auszunutzen, werden die Bilder von der Kamera als Texturen auf die Grafikkarte geladen und dort verarbeitet. Nach diesem Schritt werden die

Algorithmen beziehungsweise die Filter hintereinander angewendet. Das Ergebnis wird am Ende der Berechnung von der Grafikkarte wieder ausgelesen.

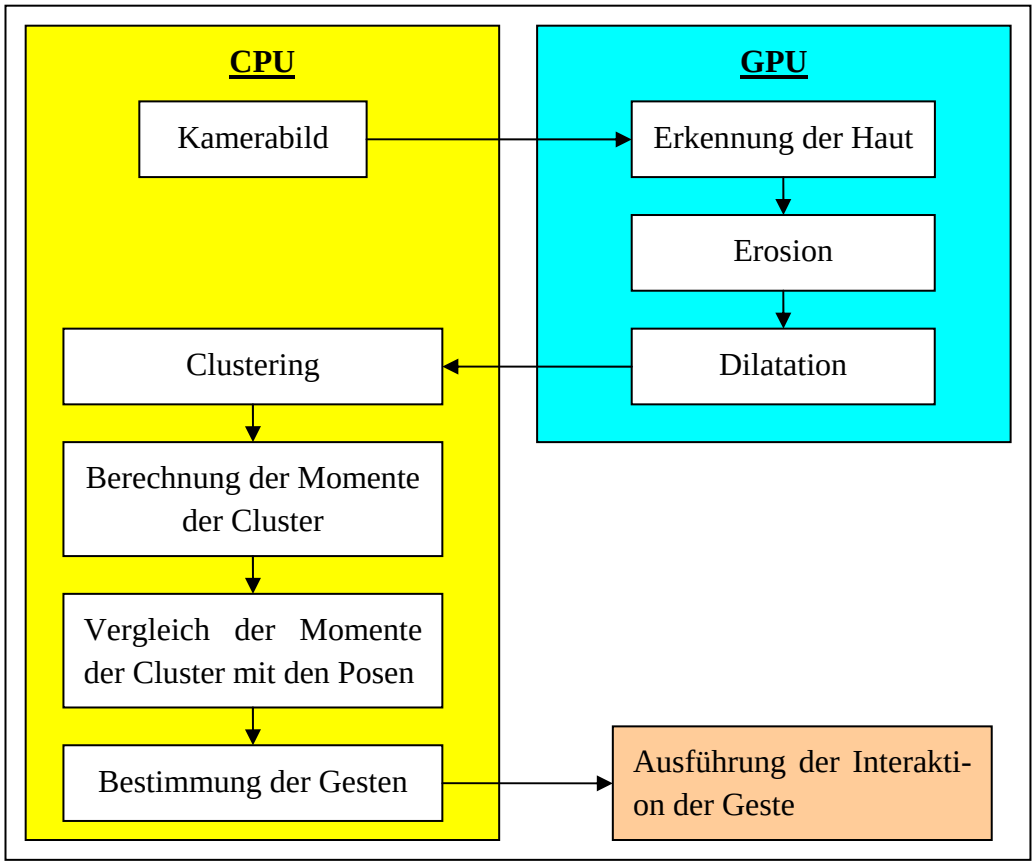

Abbildung 6: Schematische Darstellung der Verarbeitung

Bei der Implementierung wurden alle Schritte bis zum Clustering als Filter auf der Grafikkarte implementiert. Das Clustering mittels Region Growing kann dort leider nicht effizient realisiert werden. Daher wurde vor diesem Schritt das bearbeitete Bild aus dem Speicher der Grafikkarte ausgelesen und dann die Regionen klassifiziert. Die weitere Berechnung der Momente findet ebenfalls auf der CPU statt, da diese Berechnung sich zum einen nur schlecht parallelisieren lässt und zum anderen wieder auf die GPU übertragen und nach der Berechnung wieder ausgelesen werden müsste, was nicht zu einem Performancegewinn führen würde. Die Ergebnisse der Filterung auf der Grafikkarte wurden in den Alpha Kanal der Textur geschrieben, um die Werte des bestehenden Bildes nicht zu verändern und nachher noch verwenden zu können. Vor dem Auslesen des Bildes aus dem Grafikspeicher kann noch ein weiterer Filter aktiviert werden, der es ermöglicht die Aufnahmen in Farbe oder in Schwarzweiß zu erhalten. An der Berechnung ändert dieses nichts, da für das Region Growing nur die Alpha Werte der Textur aus dem

Speicher ausgelesen werden. Ein Überblick über die Verarbeitungsschritte liefert Abbildung 6.

4 Ergebnisse und Bewertung

Die Zeit zwischen zwei aufeinander folgende Bilder beträgt bei einer Bildrate von 25 Bildern pro Sekunde 0,04 Sekunden. Innerhalb dieser Zeit müssen dementsprechend die Berechnungen erfolgen. Dies gelingt aufgrund der Verwendung der GPU für die Berechnungsschritte ohne Probleme. Somit kann die Echtzeitfähigkeit gewährleistet werden.

Die Erkennung der Gesten wurde bei verschiedenen Lichtverhältnissen ausgetestet. Die Ergebnisse waren bedingt durch die Verwendung der normalisierten RGB gut. Allerdings traten Probleme bei starkem Sonneneinfall auf, für die im Weiteren noch eine geeignete Lösung gefunden werden muss.

Durch die Position und die Orientierung der Gesten lassen sich verschiedene Interaktionen realisieren. Diese Informationen können zum einen über das Netzwerk zu bestehenden Systemen übertragen oder direkt in bestehende Systeme integriert werden. Diese wurden zum Beispiel zum Selektieren von Objekten in der bestehenden Grafik Engine des HapTEK System [BDH05] integriert.

5 Ausblick

Einer der geplanten Erweiterungen soll die Miteinbeziehung der Blickrichtung zum Anpassen des Fokus des Benutzers, um somit eine Ausrichtung der virtuellen Szene gerade für dreidimensionale Ausgabemedien wie PowerWalls zu vereinfachen. Da der Kopf prinzipiell ja schon erkannt und segmentiert ist, sollte dieser Schritt im Weiteren kein großes Problem darstellen.

Momentan wird nur die erste erkannte Geste für die Interaktion verwendet. Dementsprechend soll das System für die Benutzung von zweihändigen Gesten erweitert werden, um eine größere Spanne von Möglichkeiten zur Interaktion zu erhalten.

Ein weiterer Schritt ist die Interaktion mit mehreren Personen. Dafür müssen geeignete Mechanismen integriert werden, wodurch sozusagen dem System signalisiert wird, dass eine andere Person jetzt die agierende Person ist und die Steuerung übernimmt. Vorzustellen ist dies zum Beispiel durch eine Stopp Geste, die sobald sie auftritt, nach einer zweiten Geste im Bild sucht, die als Startgeste interpretiert wird, zum Beispiel durch ein „Victory“ Zeichen des anderen Benutzers.

Literatur

[ABC04] ABACI, T., DE BONDELI, R., CIGER, J., CLAVIEN, M., EROL, F., GUTIERREZ, M., NOVERRAZ, S., RENAULT, O., VEXO, F., THALMANN, D.: Magic Wand and Enigma of the Sphinx , Computers & Graphics, Volume 28, Issue 4, August 2004, pages 477 - 484.

[ATK05] AHLBORN, B. A., THOMPSON, D., KREYLOS, O., HAMANN, B., AND STAADT, O. G.: A practical system for laser pointer interaction on large displays. In Proceedings of the ACM Symposium on Virtual Reality Software and Technology (Monterey, CA, USA, November 07 - 09, 2005). VRST '05. ACM Press, New York, NY, 106-109, 2005.

[BCH04] BAUDISCH, P., CUTRELL, E., HINCKLEY, K., AND GRUEN, R.: Mouse Ether: Accelerating the Acquisition of Targets Across Multi-Monitor Displays. In CHI 2004 Extended Abstracts (short paper), Vienna Austria, April 2004, pp. 1379 - 1382.

[BDD03] BAUDISCH, P., DECARLO, D., DUCHOWSKI, A., AND GEISLER, B.: Focusing on the Essential: Considering Attention in Display Design. CACM 46(3), pp. 60--66.

[BDH05] BIERZ, T., DANNENMANN, P., HERGENROTHER, K., BERTRAM, M., BARTHEL, H., SCHEUERMANN, G., AND HAGEN, H.: Getting in Touch with a Cognitive Character. In Proceedings of the First Joint Eurohaptics Conference and Symposium on Haptic interfaces For Virtual Environment and Teleoperator Systems - Volume 00 (March 18 - 20, 2005). WHC. IEEE Computer Society, Washington, DC, 440-445, 2005.

[BSW07] BAUDISCH, P., SINCLAIR, M, AND WILSON, A.: Soap: A Pointing and Gaming Device for the Living Room and Anywhere Else. In SIGGRAPH 2007 (Emerging Technologies demo paper), San Diego, CA, August 5-9, 2007.

[CaB03] CAO, X., BALAKRISHNAN, R.: VisionWand: Interaction techniques for large displays using a passive wand tracked in 3D. ACM CHI Letters, 5(2). Proceedings of UIST 2003, ACM Symposium on User Interface Software & Technology. p. 193-202 (2003).

[CSD92] CRUZ-NEIRA, C., SANDIN, D.J., DEFANTI, T.A., KENYON, R.V., HART, J.C.: The Cave - Audio Visual Experience Automatic Virtual Environment. Commun. ACM 35(6): 64-72 (1992).

[CVF02] CAVENS, D., VOGT, F., FELS, S. S., MEITNER, M.: Interacting with the Big Screen: Pointers to Ponder. Proceedings ACM Conference on Computer Human Interaction (CHI2002). Pages 678-679. Apr. 2002.

[FCS07] MOOG FCS B.V.**:** The HapticMaster, http://www.fcs-cs.com/robotics/products/hapticmaster

[Flu00] FLUSSER, J.: "On the Independence of Rotation Moment Invariants," Pattern Recognition, vol. 33, pp. 1405-1410, 2000

[FM04] FUNG, J., MANN, S.: "Computer Vision Signal Processing on Graphics Processing Units", Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2004), Montreal, Quebec, Canada, May 17--21, 2004.

[FrP00] FRÖHLICH, B. AND PLATE, J.: The cubic mouse: a new device for three-dimensional input. In Proceedings of the SIGCHI Conference on Human Factors

in Computing Systems (The Hague, The Netherlands, April 01 - 06, 2000). CHI '00.

[FuM04] FUNG, J., MANN, S.: "Using Multiple Graphics Cards as a General Purpose Parallel Computer : Applications to Computer Vision", Proceedings of the 17th International Conference on Pattern Recognition (ICPR2004) , Cambridge, United Kingdom, August 23-26, 2004, volume 1, pages 805-808.

[GM02] GOMEZ G., MORALES E.: " Automatic feature construction and a simple rule induction algorithm for skin detection" Proc. of the ICML Workshop on Machine Learning in Computer Vision, 31-38. 2002

[Jae02] JÄHNE, B.: Digitale Bildverarbeitung. Springer-Verlag, Heidelberg, fünfte Edition, 2002.

[Imm07] IMMERSION: CyberGrasp™ Exoskeleton & GraspPack™ backpack http://www.immersion.com/3d/products/cyber_grasp.php

[KaT99] KALLMANN, M., THALMANN, D.: Direct 3D Interaction with Smart Objects, Proceedings of ACM Virtual Reality Software and Technology (VRST), December, 1999, London.

[KMB07] KAKUMANU, P., MAKROGIANNIS, S., AND BOURBAKIS, N.: A survey of skin-color modeling and detection methods. Pattern Recogn. 40, 3 (Mar. 2007), 1106-1122.

[LFR02] VAN DER LINDE, R. Q., LAMMERTSE, P., FREDERIKSEN, E. AND RUITER. B. The HapticMaster, a new high-performance haptic interface. In Proc. EuroHaptics, Edinburgh, 2002, pp. 1--5.

[MZK01] MCIVOR, A., ZANG, Q., AND KLETTE, R.: The Background Subtraction Problem for Video Surveillance Systems. In Proceedings of the international Workshop on Robot Vision (February 16 - 18, 2001). R. Klette, S. Peleg, and G. Sommer, Eds. Lecture Notes In Computer Science, vol. 1998. Springer-Verlag, London, 176-183. , 2001.

[MSP03] MARTINKAUPPI, B., SORIANO, M., PIETIKÄINEN, M,: Detection of skin-color under changing illumination: a comparative study, 12th Interenational Conference on Image Analysis and Processing, 2003.

[Nvi07] NVIDIA Cg Toolkit, http://developer.nvidia.com/page/cg_main.html, 2007.

[ON01] OLSEN, D. R., NIELSEN, T.: Laser pointer interaction. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (Seattle, Washington, United States). CHI '01. ACM Press, New York, NY, 17-22, 2001.

[SR03] STEFANI, O., RAUSCHENBACH, J.: 3D input devices and interaction concepts for optical tracking in immersive environments. In Proceedings of the Workshop on Virtual Environments 2003 (Zurich, Switzerland, May 22 - 23, 2003). EGVE '03, vol. 39. ACM Press, New York, NY, 317-318, 2003.

[Ver91] VERNON, D.: Machine Vision: Automated Visual Inspection and Robot Vision 260 pages, Prentice Hall, September 1991.

[VWF03] VOGT, F., WONG, J., FELS, S. S., CAVENS, D.: Tracking Multiple Laser Pointers for Large Screen Interaction. Extended Abstracts of ACM UIST 2003. Pages 95-96. 2003.

[Wik07] Wikipedia: Bluescreentechnik, http://de.wikipedia.org/wiki/Bluescreen-Technik, 2007.

[Zha03] ZHAI., S.: What's in the eyes for attentive input. Communications of the ACM, 46(3), 34-39, 2003.

Autoren

Dipl.-Inf. Torsten Bierz, Jahrgang 1978, studierte Informatik mit Fachrichtung Computergrafik und Visualisierung an der Technischen Universität Kaiserslautern. Nach seinem Abschluss arbeitete er von Mai 2004 bis Dezember 2004 am Projekt „Entwicklung und Aufbau eines haptischen Mensch-Maschine Interface als neuartiges Werkzeug zum erweiterten Einsatz von Simulation bei Entwicklungs- und Produktionsvorgängen“ in der Arbeitsgruppe von Prof.-Dr. Hans Hagen. Seit 2005 ist Herr Torsten Bierz Stipendiat im Internationalen Graduiertenkolleg der Technischen Universität Kaiserslautern (International Research and Training Group for Visualization of Large and Unstructured Data Sets Applications in Geospatial Planning, Modeling, and Engineering) und in der Arbeitsgruppe Visualisierung von Prof.-Dr. Achim Ebert tätig. Seine Schwerpunkte beinhalten die Themen der immersiven Visualisierungssysteme, Interaktion mit haptischen Geräten, optische Tracking Systeme sowie Beschleunigung mittels GPU basierten Methoden.

Prof.-Dr. Achim Ebert ist seit 2005 Junior Professor an der Technischen Universität Kaiserslautern. Des Weiteren leitet Dr. Ebert die Forschungsgruppe zur Intelligenten Visualisierung und Simulation (IVS) am Forschungsinstitut für künstliche Intelligenz (DFKI) in Kaiserslautern. Er promovierte 2004 an der Technischen Universität Kaiserslautern unter Prof. Dr. Hans Hagen. Prof. Dr. Eberts Forschungsschwerpunkte sind Virtual und Mixed Reality, Mensch-Maschine Interaktion (HCI), Informationsvisualisierung, mobile Visualisierung und Visualisierung im Kontext der künstlichen Intelligenz. Daraus resultieren zahlreiche Anwendungsgebiete wie Dokumenten Management, Visualisierung von Suchanfragen, Visualisierung von Schmutzwasserkreisläufen und Kanalnetzen, virtuelle Kleidungssimulation, Prozessvisualisierung, etc. Er leitete und beteiligte sich im Programmkommitte von verschiedenen Konferenzen. Seit 2005 ist Prof. Dr. Ebert Co-Chair der IASTED Visualitation and Image Processing (VIIP) Konferenz.

Prof.-Dr. Jörg Meyer ist am “Department of Electrical Engineering & Computer Science” und im “Department of Biomedical Engineering in the Henry Samueli School of Engineering” an der University of California, Irvine, USA seit 2002 beschäftigt. Er ist des Weiteren beteiligt am „Center of GRAVITY“ (Graphics, Visualization and Imaging Technology) im kalifornischen Institut für Telekommunikations- und Informationstechnologien (Calit2). Prof. Dr. Meyer habilitierte an der Technischen Universität Kaiserslautern 1999. Danach arbeitete er bis 2000 als wissenschaftlicher Mitarbeiter (PostDoc) und Dozent im Informatik Fachbereich an der University of California, Davis, USA. Anschließend erhielt er eine Assistenzprofessur im Fachbereich Informatik und Maschinenbau an der Mississippi State University. Prof. Dr. Meyer beschäftigt sich mit der Visualisierung von großen Datensätzen, Visualisierung von biomedizinischen Daten, Bildverarbeitung, interaktive Rendering Methoden und virtuellen Umgebungen und Realitäten (VR).