Der Regensburger Marathon-Cluster

– Informationswissenschaftliche Betrachtung –

Hubert Feyrer <hubert@feyrer.de>

17. Juli 2002

Zusammenfassung

Der vorliegende Text beschreibt das im Sommer 2000 durchgeführte Projekt eines Workstation-Clusters zur Berechnung von Zieleinlaufs-Videos für den Stadtmarathon. Nach Informationen über das Projekts und dessen Durchführung wird auf die informationswissenschaftlichen Aspekte in den Bereichen Video-, Image- und Audio Retrieval eingegangen, speziell auf die Themen Segmentierung der Eingabedaten, Selektion "interessanter" Daten, deren Synthese sowie die Vorbereitung zum Durchsuchen der Ergebnisse. Anschließend wird anhand mehrere Beispiele erläutert, wie dieses Verfahren bei weiteren Projekten eingesetzt werden kann.

Inhaltsverzeichnis

1 Überblick

2001 leisteten die R-KOM und die Fachhochschule Regensburg ihren Beitrag zum Stadtmarathon durch das Erstellen der Zieleinläufe in Bild und Film für jeden einzelnen Läufer im Internet. Ein Cluster aus 45 Rechnern der Fachhochschule Regensburg berechnete dabei die mehr als fünftausend Einzelbilder und Filme. [1]

Das Projekt wurde als Gegenstand der Informatik durchgeführt, Schwerpunkte dabei waren Cluster Computing, Parametrisierung der Lastverteilung auf die einzelnen Knoten, Deployment der Software auf die Clients, die Kommunkation des Cluster Control Rechners mit den einzelnen Knoten sowie die technische Realisierung des Datenaustauschs mittels eines gemeinsamen File-Storage-Systems.

Die inhaltlichen Schritte - das eigentliche Rechenvorhaben - wurden gemäß den Projektvorgaben definiert. Sie sind es, die aus dem Blickwinkel der Informationswissenschaft von Interesse sind, da sie Aspekte aus dem Bereich Video Retrieval bzw. Image Retrieval beinhalten und aus diesem Grund im zweiten Teil näher betrachtet werden sollen.

Eine parallele Darstellung der Rechenschritte des Marathon-Cluster-Projektes einerseits und der abstrakten, informationswissenschaftlichen Schritte andererseits wurde in Erwägung gezogen, der Klarheit halber werden die beiden Aspekte jedoch jeweils getrennt mit all ihren Einzelschritten besprochen – erst eine allgemeine Projektvorstellung, anschließend die abstrakten Schritte mit Fokus auf weitere Anwendungen.

2 Das Marathon-Cluster-Projekt

Bevor näher auf die informationswissenschaftlichen Aspekte des Regensburger Marathon-Cluster-Projektes eingegangen wird soll dieses Projekt zuerst vollständig mit all seinen Schritten vorgestellt werden. Im Kapitel 3 werden dann die einzelnen Schritte nochmals aus dem Blickwinkel der Informationswissenschaften beleuctet.

2.1 Zustandekommen des Projekts

Als Sponsor des Sports Experts Marathons realisierte die R-KOM die Umsetzung der Zieleinläufe ins Internet. Um den Qualitätsansprüchen gerecht zu werden reichten die hausinternen Rechner jedoch nicht aus um in endlicher Zeit Ergebnisse zu liefern. Durch Kontakte zur Fachhochschule konnte hier eine Zusammenarbeit erreicht werden, und der Fachbereich Informatik stelle Rechnerleistung und Support beim Setup zur Verfügung.

An der Fachhochschule Regensburg standen 45 Rechner inklusive Installation und Management der einzelnen Knoten sowie der nötigen Speicherplatz für Daten zur Verfügung, Angehörige des Fachbereichs Informatik/Mathematik half beim Optimieren und Parametrisieren der Unix-basierten Software, die von Mitarbeitern der Firma R-KOM aus frei verfügbaren Komponenten zusammengestellt wurde.

Die Software setzte sich aus Komponenten zum Aufteilen von Video-Streams in Einzelbilder sowie zum Errechnen von Video-Sequenzen aus diesen Einzelbildern, basierend auf der Einlaufzeit der einzelnen Läufer, zusammen. Wichtig war hier eine automatisierte, scriptgesteuerte Abarbeitung für die 5500 durchs Ziel gekommenen Läufer.

2.2 Über den Hardware- und Software-Setup

Für den Regensburger Marathon-Cluster stellte der Fachbereich Informatik der Fachhochschule Regensburg drei Rechnerräume mit insgesamt 57 Rechner zur Verfügung. 15 der Rechner liefen unter Solaris, dies konnte auch nicht geändert werden. Die Hardware der verbleibenden Rechner bestand aus Dell OptiPlex PCs mit unterschiedlich viel Arbeitsspeicher und CPU. In zwei der Räume (U512, U513) waren je 15 Rechner mit PII-400MHz, 64MB RAM und 4GB Platte, im anderen (U521) waren 12 Rechner mit PIII-1GHz, 256MB RAM und 10GB Platte. Diese Rechner konnten beliebig installiert werden, und es wurde NetBSD als Client-Betriebssystem verwendet, da hierfür die gesamte Software zugänglich, das Betriebssystem auf den Cluster-Rechnern problemlos aufzusetzen, und das nötige Know-How im Haus vorhanden war.

Für den Cluster-Setup wurde einer der PII-400 Rechner aufgesetzt und unter NetBSD installiert, anschliessend die nötigen Programme installiert (dumpmpeg, mpeg_encode). [15] [16] [17] Für das gesamte Cluster-Projekt existierte eine einzelne Kennung, deren Home-Verzeichnis vom NFS-Server gemountet wurde, der auch für sämtliche Berechnungen den nötigen temporären Speicherplatz bereitstellte. [14] Als NFS-Server diente hier der Unix-Server des Fachbereichs Informatik/Mathematik, eine Sun Ultra 10 unter Solaris 2.6 mit 300MHz CPU, 1024MB RAM und 120GB Plattenplatz.

Zu Monitoring-Zwecken wurden auf dem Client-Prototyp noch ein SNMP-Agent, der rstat-Service sowie das Programm tload installiert, das auf einer der Text-Consolen die Auslastung des Rechners mit Balken anzeigte. Die diversen Pakete die von dumpmpeg und mpeg_encode benötigt wurden rundeten die Client-Installation ab.

Nachdem alles zur Zufriedenheit lief wurde das Festplatten-Image des fertig konfigurierten Client-Rechners mittels der Harddisk-Image Cloning Software “g4u” auf dem Master Deployment Rechner der FH Regensburg abgelegt. Das 4GB grosse Plattenimage wurde dabei auf ca. 650MB kompromiert.

Der Marathon-Lauf fand am Sonntag Nachmittag statt, die Vorbereitungen des Regensburger Marathon-Clusters begann bereits am Samstag Abend mit dem Setup der Cluster-Rechner.

Dabei wurde das erstelltes Festplatten-Image mit Hilfe von “g4u” jeweils auf einen Rechner in jedem der drei Rechnerräume installiert, und anschließend jeder dieser drei Rechner selbst als Installationserver für die restlichen Rechner des Raums vorbereitet und benutzt. Abbildung 1 illustriert dieses Vorgehen. [1] [10]

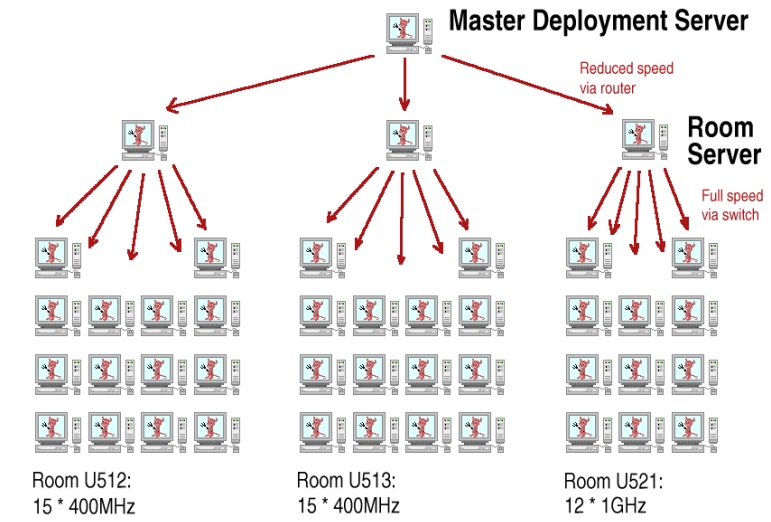

Abbildung 1: Client-Deployment

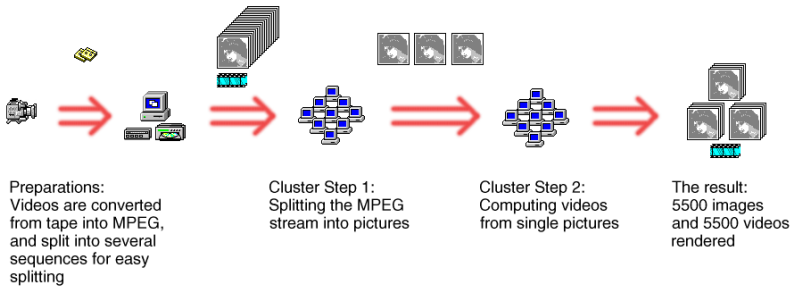

Abbildung 2: Die logischen Schritte beim Marathon-Cluster-Projekt

2.3 Vorarbeiten

Die eigentliche Rechenaufgabe des Regensburger Marathon-Clusters bestand aus zwei Schritten:

1. Aufsplitten der MPEG-Sequenzen in Einzelbilder
2. Zusammenfügen von Einzelbildern zu einem Zieldurchlaufs-Film

Ein Überblick über alle durchgeführten Schritte ist in Abbildung 2 zu sehen. Als Vorbereitung stand am Zieleinlauf zwei Videokameras, die den Zieleinlauf auf Band aufzeichneten. Es wurden zwei Kameras benutzt, die dicht nebeneinander standen und so den gleichen Blickwinkel boten. Die Kameras liefen abwechselnd mit Overlap, so daß beim Bandwechsel keine Bilddaten verloren gingen. Die jeweils ca. 2 Stunden langen Bänder wurden digitalisiert und in kleinen, 11 Minuten langen Sequenzen auf CD gebrannt, die dann für den ersten Rechenschritt auf die einzelnen Rechner verteilt wurden.

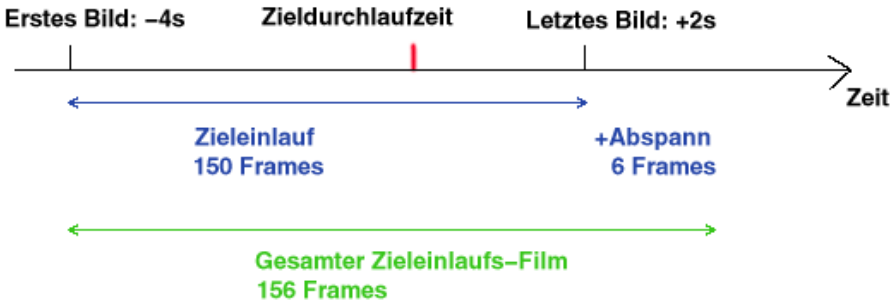

Abbildung 3: Auswahl der Bilder für das Zieleinlaufs-Video

2.4 Erster Rechenschritt

Für das Zerlegen der Sequenzen wurden die 1GHz Maschinen der FH gewählt. Nach dem Einlegen der CD mit einer Sequenz wurde diese 11 Minuten lange Videosequenz mittels “dumpmpeg” in einem ca. 45 Minuten dauernden Rechenprozess in ca. 16.500 Einzelbilder im JPEG Format zerlegt und auf dem zentralen Fileserver für den zweiten Rechenschritt abgelegt. [1]

2.5 Zwischenschritt

Für die weitere Berechnung im zweiten Schritt wird für jede Sequenz die genaue Start- und Endzeit benötigt, und die jeweils zugehörigen Dateinamen des ersten und letzten JPEG-Bildes. Diese Daten wurden für die weitere Verarbeitung in eine MySQL-Datenbank abgelegt. Weiterhin wurde aus diesen Daten die wirkliche Bildrate ermittelt, da z.B. durch Schwankungen der Bildrate in Folge von Wärme das MAZ-Gerät nicht konstant 25 Bilder liefert. Eine leichte Varianz könnte sich hier im Verlauf von 5 Stunden Filmmaterial erheblich aufschaukeln und zu unbrauchbaren Resultaten führen, wenn später die exakten Bilddaten einer bestimmten Zeit benötigt werden.

2.6 Zweiter Rechenschritt

Nachdem im ersten Schritt alle Einzelbilder abgespeichert wurden, werden sie nun wieder zu kleinen Filmen vereint. Ziel ist, daß jeder Läufer anhand seiner Startnummer seinen persönlichen Durchlauf durch’s Ziel ansehen kann, und - vorab berechnet - jeder Läufer ein eigenes Video bekommt. Von den gestarteten knapp 7000 Läufern gingen insgesamt ca. 5500 durch’s Ziel. Es existierten drei Diszipline: Marathon (42km), Halbmarathon (21km) sowie Speedskating (21km), für Herren und Damen wurden getrennte Ergebnislisten mit der Zieldurchlaufzeit geführt. Ausgehend von der Zieldurchlaufzeit wurde mit Hilfe der MySQL-Datenbank das zugehörige Bild ermittelt. Von diesem wurde dann zwei Sekunden zum letzten Bild des Films vorgegangen, und ab diesem dann rückwärts 150 Bilder in ein temporäres Arbeitsverzeichnis kopiert. Die 150 Bilder stehen hier für 6 Sekunden Film bei a 25 Bilder pro Sekunde.

Abbildung 4: Template und Beispiel für Abspann

Abschließend kamen noch sechs Frames mit gleichem Inhalt. In eine vorgefertigten Maske mit den Logos der Sponsoren der Zieleinlaufsvideos wurde der Name des jeweiligen Läufers, seine Startnummer, die Zeit, Plazierung und die Disziplin geschrieben, um diese am Ende des Zieleinlaufsfilms als Abspann anzuzeigen. Abbildung 4 zeigt sowohl die leere Maske als auch ein ausgefülltes Beispiel.

Das Zusammenrechnen des Films aus den einzelnen Bildern wurde mit dem Programm mpeg_-encode gemacht. Durch eine Parameterdatei wurde festgelegt welche Bilder zu einem Film zusammengerechnet werden sollten, und welche Client-Rechner diese Aufgabe übernehmen würden. mpeg_encode startete dann auf jedem Client eine Berechnung für ein paar Bilder, um die Geschwindigkeit der einzelnen Knoten abschätzen zu können. Anschließend wurden dann die verbleibenden Bilder entsprechend auf die einzelnen Knoten verteilt, die die Bilder dann via NFS lasen und die berechneten Teilfilme auch via NFS wieder an den Haupt-Prozess zurücklieferten, der die Teile dann vereinigte und das Ergebnis als eine MPEG-Datei ablegte. All diese Schritte inklusive dem dabei enthaltenen Load Balancing auf die verschiedenen Knoten führte das Programm ohne jegliches Zutun automatisch aus, so daß hier sehr viel manuelle Arbeit (und mögliche Fehler) verhindert wurden.

Die relativ kurzen Filmlänge von 156 Frames förderte die Eigenheit on mpeg_encode zu Tage, dass nicht mehr als ca. 15 Rechner an einem solchen Film rechnen konnten. Aus diesem Grund wurden die Rechner in vier Subcluster aufgeteilt, siehe Abbildung 5. Die Parameterdatei von mpeg_encode lag für jeden der vier Subcluster vor, so daß die Ergebnisliste jeder Disziplin auf jedem Subcluster berechnet werden konnte - je nachdem wo gerade Rechenzeit frei war. Das Job-Scheduling welcher Subcluster welche Ergebnisliste berechnet wurde hier per Hand gemacht.

Neben dem Zieleinlauf-Film wurde zusätzlich noch das Bild des exakten Zieleinlaufs gesichert, und wiederum der Name des Läufers, Zeit, Plazierung und Disziplin eingetragen, siehe Abbildung 6.

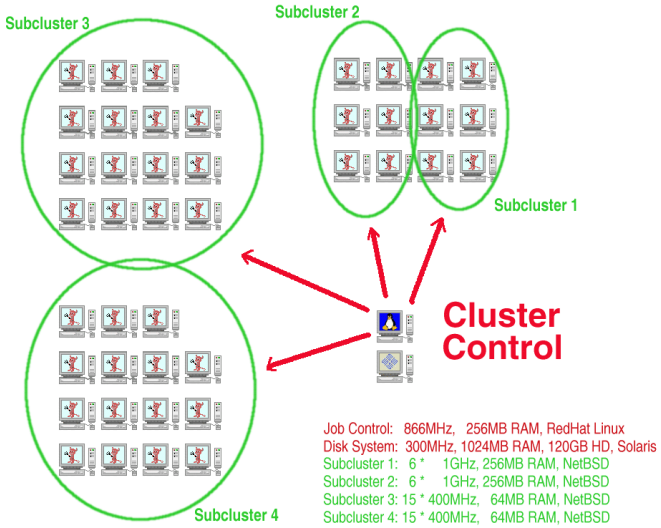

Abbildung 5: Die vorhandenen Rechner wurden in vier Subcluster unterteilt

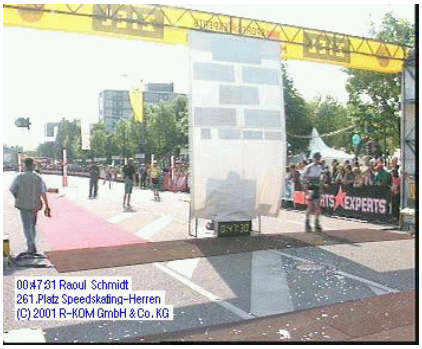

Abbildung 6: Zielfoto

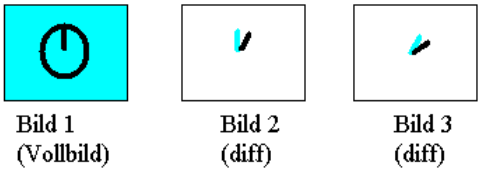

Abbildung 7: Statische und dynamische Bildkomponenten des MPEG-Formats

3 Schritte des Verfahrens

Im Kapitel 2 wurden die einzelnen Schritte beschrieben, die wärend des Marathon-Cluster-Projekts durchgeführt wurden. Bezugnehmend darauf sollen im Folgenden die informationswissenschaftlichen Aspekte des Projektes in Form von Segmentierung, Selektion, Synthese sowie der Indexierung zur Durchsuchung näher betrachtet werden.

Da die hier beschriebenen Schritte auf weitere Anwendungen übertragbar sind sollen die betroffenen Aspekte hier verallgemeinert besprochen werden, Kapitel 4 beschreibt ein paar denkbare Anwendungen.

3.1 Segmentierung

Der erste Rechenschritt des beschriebenen Projektes war die Zerlegung der Eingabedaten, die in Form eines MPEG-Streams vorlagen, in einzelne JPEG Bilder. Die Auswahl der Daten aus dem MPEG Stream, die jeweils zu einem Bild gehören, wurde durch die Tatsache interessant gemacht, daß das MPEG-Format statische Bildinhalte nicht für jedes Bild des Films neu speichert, und so auch später ein Zusammenfügen der statischen und dynamischen Komponenten für jedes Video aus verschiedenen Komponenten erfolgen mußte. Beim MPEG-Format spricht man von P- und B-Frames, die die statischen und dynamischen Bildelemente speichern, und beim Segmentieren kombiniert werden müssen, wie in Abbildung 7 zu sehen ist.

Generell wäre auch eine rein zeitlich basierte Zerlegung der Eingangsdaten bzw. Segmente mit fester Datenlänge denkbar, wenn das Eingabe-Format dies erlaubt. V.a. bei nicht bildhaften Eingabedaten ist dies von Interesse, z.B. wenn Sprache oder Ton verarbeitet werden soll. Dies ist auch für die Weiterverarbeitung bzw. Übertragung in paketorientierten Diensten hilfreich, vgl. Beispiel “Voice over IP”.

3.2 Selektion

Die Auswahl der “interessanten” Daten im beschriebenen Projekt basierte auf der Zeit, zu der der Läufer durch’s Ziel lief. Ausgehend von dieser Zeit wurde das Zieleinlaufs-Foto bestimmt, und davon ausgehend Bilder ausgewählt und weiterverarbeitet.

Anstatt der hier beschriebenen rein zeitbasierten Auswahl wären auch andere Kriterien und Verfahren denkbar, wie sie im Bereich der Content Extraction beim Image-, Video- und

Audio-Retrieval eingesetzt werden. Etwa mittels Verfahren der Szenenerkennung, bei der gewisse Kriterien auf mehrere Bilder angewandt werden, und innerhalb gewisser Grenzen (z.B. “Läufer gut sichtbar im Bild”) Bilder ausgewählt werden, die diesen Kriterien entsprechen. Je nach gerechtgertigtem Aufwand kann auch mit KI-Methoden eine Gesichtserkennung aufgrund Neuronaler Netze bzw. Fuzzy Logik erfolgen. Bei 5500 Läufern würde dies jedoch einiges an Vorarbeit bedeutet - bereitstellen der Fotos, erstellen eines Schemas zur Mustererkennung, etc. - was in der Praxis noch durch Faktoren wie mangelnde Qualität der Bilder oder Zieleinlauf in Gruppen, bei denen einzelne Läufer nicht erkennbar sind verkompliziert wird.

Je nach Aufgabenstellung kann auch für die weitere Verarbeitung ein ganz anderer Satz an Daten gewählt werden. Im Bereich der Augmented Reality sollen beispielsweise Bildteile erkannt, verstärkt oder betont werden, so daß hier sicherlich mit bilderkennenden Methoden zu arbeiten ist. Denkbar wären hier Die Veränderung des Kontrasts, Farbseperation und -segmentierung, Kontur- und Texturerkennung. [13] [19] Anders ist bei der Verarbeitung von Audio-Daten mittels Frequenz- und Spektrums-Analyse wie z.B. der Fourier-Transformation, Hoch- und Tiefpaß-Filtern festzustellen, ob Daten innerhalb bestimmter Grenzen sind. Siehe hierzu die später diskutierten Anwendungen “Entfernung von Kratzern aus Schallplattenaufnahmen” bzw. “Voice over IP”.

3.3 Synthese

Nach dem Aufteilen des ursprünglichen Datenstroms sowie der Auswahl “interessanter” Daten daraus stellt sich die Frage, was mit diesen Daten weiter geschehen soll. Im Fall des Marathon-Projekts wurde aus den ausgewählten Bildern wieder ein (kurzer) MPEG-Film errechnet. Durch die Eigenart des MPEG-Formats, redundante Bildteile zu separieren und nur einmal zu speichern war dazu ein relativ hoher Rechenaufwand nötig, der dann im Projekt auch auf jeweils einen Teilcluster von 15 Rechnern verteilt wurde. Andere Videoformate wie Quicktime, AVI, DivX oder RealVideo wären hier als Alternative denkbar gewesen.

Andere Arten, die ausgewählten Daten weiterzuverarbeiten wären etwa bei Audio-Daten, diese durch verlustbehaftete Codierung wie z.B. MPEG-Layer III (mp3), MiniDisc, WMF, Ogg Vorbis oder RealAudio zu komprimieren, ein unkomprimiertes/uncodiertes Format wie WAV oder CDDA abzuspeichern, oder die Audio-Daten in zeitbasierte Zellen zu unterteilen, um sie über ein Daten- oder Mobilfunk-Netz weiterzuleiten, bevor sie vieder mittels Digital-Analog-Wandlung in Tonsignale umgewantelt werden (vgl. VoIP Beispiel unten).

3.4 Ergebnisse zum Durchsuchen vorbereiten

Neben den Kernthemen des Bereichs Image/Video Retrieval ist die Durchsuchbarmachung und Indizierung von Multimedia-Inhalten ein weiteres Forschungsgebiet der Informationswissenschaften.

Im vorliegenden Projekt wurde die Suche über die insgesamt 5500 erzeugten Zieleinlaufsvideos und -Bilder anhand der Startnummer realisiert, die für jeden Läufer eindeutig vergeben wurde. Über ein Web-Formular konnte die Startnummer eingegeben und ausgewählt werden,

ob das Foto (Einzelbild) des Zieldurchlaufs angezeigt werden sollte, oder ob die zugehörige Video-Sequenz gezeigt werden sollte. Anhand der Startnummer konnte dann direkt auf das zugehörige Video bzw. das Foto zugegriffen werden, da diese jeweils die Startnummer als Dateiname enthielten.

Die Indexierung erfolge hiermit also durch Abspeichern mit der Startnummer als Dateinamen (startnummer.jpg, startnummer.mpg), als Ordnungssystem wird das Dateisystem mit seinen "normalen' namensbasierten Zugriffsmethoden bei der Abfrage verwendet. Mit ihnen kann aus der Zielnummer eindeutig die jeweilige Dateiname ermittelt werden konnte. [13]

Verallgemeinert kann für die Indexierung im vorliegenden Ansatz gesagt werden, daß für den Endanwender die Suche auf ein um Größenordnungen größeres Datenvolumen ausgerichtet ist, und z.B. beim Marathon-Lauf nicht auf ein einzelnes Bild (1/25 Sekunde) sondern einen wesentlich größeren Bereich (± 1 Sekunde) gerichtet ist.

4 Weitere Anwendungen des Verfahrens

Im Folgenden soll untersucht werden, wie das beschriebene Verfahren mit seinen abstrakten Schritten (Segmentierung, Selektion, Synthese) auch auf andere Einsatzgebiete übertragen werden kann, und anhand diverser Beispiele illustriert werden.

4.1 Rad- und Autorennen

Segmentierung:

> Wie beim Marathon-Lauf erfolgt die Zerlegung der Eingangsdaten von Video- und Bilddaten zeitbasiert. Durch die höhere Geschwindigkeit der betrachteten Teilnehmer empfiehlt es sich bei Rad- und Autorennen jedoch, High-Speed-Kameras zu verwenden, die mehr als die üblichen 25 Bilder pro Sekunde liefern. Im Marathon-Projekt war dies bereits bei den Speed-Skatern abzusehen.

Selektion:

> Auch hier kann die Zeit des Zieldurchlaufs (-durchfahrt) als Selektionskriterium verwendent werden. Weitere Möglichkeiten wären Ereignisse wie Zieldurchfahrt, Rundenzeit oder passieren bestimmter Wegpunkte. Aufgrund der höheren Geschwindigkeit empfiehlt sich auch hier eine genauere Zeitmessung als beim Marathon-Lauf. Weiterhin ist zu klären wieviele Bilder ausgewählt werden sollen. Diese Anzahl ergibt sich anhängig von der Auflösung der benutzten Bildquelle und der Zeit, in der das Zielobjekt im Bild ist.

Synthese:

> Diese ist ebenfalls wie beim Marathonlauf abhängig vom gewünschten Zielformat, die ausgewählten Bilddaten werden hier ebenfalls einer Video-Codierung unterzogen.

Durchsuchen:

Wie beim Marathon auch wird wohl der Name des Fahrers als primärer Suchschlüssel dienen, evtl. durch eine Startnummer codiert.

4.2 Werbepausen im Fernsehen entfernen

Segmentierung und Synthese:

Zerlegung in Einzelbilder sowie (Wieder)Zusammenfügen wie beim Marathon-Cluster-Projekt.

Selektion:

Die interessante Frage hier ist, wie man Werbespots erkennt. In der Praxis erprobte Verfahren wie Ausstrahlen eines Signals zu Beginn/Ende der Werbepause, entweder durch die Fernsehanstalten selbst (z.B. VPS) oder durch unabhängige Dienstleister scheitern hier am Widerstand der Werber. [11] [12] Sie zahlen schließlich viel Geld für die Plazierung ihres Produktes, so daß diese Anwendung sicher gegen ihre Interessen spricht.

Welche Auswege bleiben, um dem Werbegeplagten TV-Junkie vor Werbung zu schützen? Wenige.

Verfahren der KI bzw. Bilderkennung sind sehr Aufwendig. Ein weiteres Problem ist, Schnitte zwischen Szenen zu erkennen und zu bewerten, ob diese nun zum Werbeblock gehören oder nicht. Dies ist rein aufgrund der Bildinformation kaum möglich. Eine zusätzliche Auswertung der Audio-Daten (gemeinsame Melodie über alle Szenen? Verbindungsmelodie/Sprache zwischen Szenen?) ist hier denkbar, jedoch auch nicht 100% sicher.

Alternativ wäre eine Erkennung anhand eines (als Video) vorliegenden Spots bzw. gegen eine bestehende Datenbank aus Werbespots denkbar (“Query by Example”, [13]). Die Erkennung könnte hier Bild für Bild geschehen. Nach einigen erkannten Bildern kann dann ein Spot identifiziert werden. Die genaue Anzahl der zu erkennenden Bilder (2, 5, 100, ...) ist dabei anhängig von der Qualität der Bilderkennung.

Offene Fragen, die hier jedoch nicht weiter behandelt werden sollen sind die Pflege bzw. Aktualität der Spot-Datenbank, die Akzeptanz der Werber sowie nicht zuletzt die Frage, was anstatt der Werbung geschaltet werden soll.

Durchsuchen:

Wie bei der Selektion auch stellt sich hier die Frage, nach welchem Kriterium in einem Videostream gesucht werden soll. Bei gezielter Suche nach Videospots kann hier ein Mapping von Produkt/Herstellernamen erfolgen, da die Anwendung jedoch das Ausblenden (Entfernen) von Spots ist, ist hier eine Suche wenig sinnvoll, da ja voraussichtlich alle Spots ausgeblendet werden sollen.

Verwandte Anwendungen wie Kindersicherungen müssen hier genauer unterscheiden.

4.3 Augmented Reality

Segmentierung:

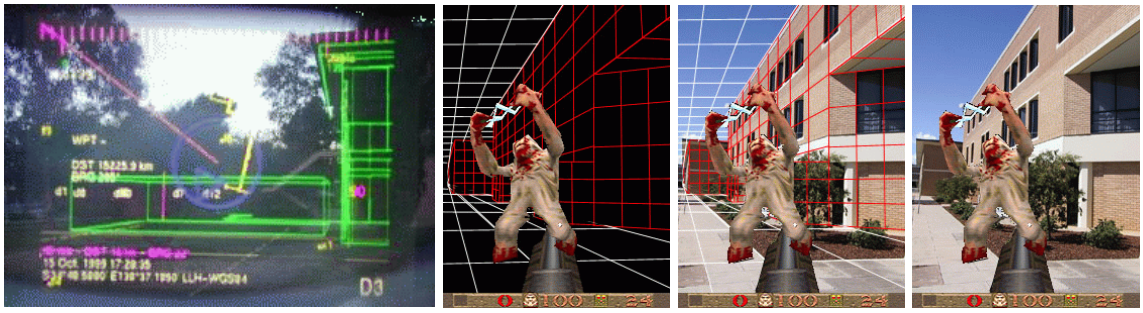

Abbildung 8: Beispiele für Augmented Reality

Zerlegung des Videostroms, z.B. von einer tragbaren Videokamera, wie besprochen in Einzelbilder bzw. direktes Auslesen von Bilddaten mittels Framegrabber-Karte, ohne Umweg über Stream-Format.

Selektion:

Es sollen alle Bilder wiedergegeben werden, jedoch ggf. mit weiteren Informationen versehen. Denkbar sind hier:

- Optische Verbesserung der gesamten Bildqualität, z.B. bei Nebel, Rauch, Dunkelheit (vgl. Restlichtverstärkung)
- Einzeichnen wichtiger Teile / Kanten / Ecken, ggf. basierend auf Konstruktionsdaten von Gebäuden etc. [3]
- Erkennen und Kennzeichnen von Personen, z.B. für Rettungseinsätze

Diese Verbesserungen können rein aufgrund der bildbasierten Daten einer mobilen Kamera erfolgen. Abbildung 8 zeigt ein paar Beispiele hierzu.

Je nach Anwendung und Einsatzgebiet ist es auch möglich, als Eingabedaten nicht (nur) reine Bilddaten, sondern auch Standort-Koordindaten wie z.B. aus GPS-Systemen zu verwenden. Angaben aus Kreiselkompaß-Systemen können Informationen über Blickrichtung und Kopfneigung etc. geben. Gekoppelt mit einem System das über Raum-/Gebäudepläne verfügt ist hier eine bessere Orientierung möglich.

Eine Kombination beider Verfahren erlaubt es, die Navigation in Gebäuden sicher zu gestalten, und zusätzlich “interessante” Bildelemente mit Hilfe mustererkennender Verfahren zu kennzeichnen.

Als Beispielanwendung ist hier ein System für Rettungseinsätze denkbar, das die Orientierung in verrauchten Räumen durch Gebäudepläne unterstützt, jedoch gleichzeitig zu rettende Personen mittels Bilderkennung und anderen Methoden der Content Extraction lokalisiert und kennzeichnet. [2] [13]

Synthese:

Die von der Kamera gelieferten Bilddaten werden durch “interessante” Landmarks überlagert, und z.B. direkt an einen Framebuffer gesandt oder auf einem kleinen Monitor, Head-Up-Display oder Autoshutter-Brille dargestellt. [4] [5] Art und Menge der

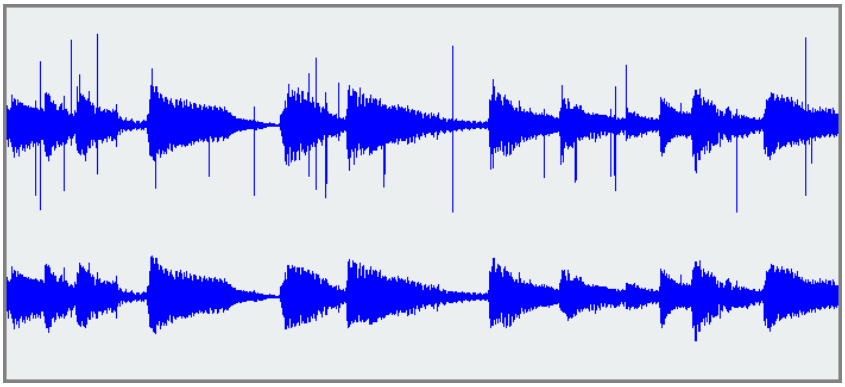

Abbildung 9: Audio-Signal mit (oben) und ohne (unten) Knacksern

ergänzten Informationen sind abhängig vom jeweiligen Einsatzgebiet. Eine Videocodierung ist i.A. nicht nötig, da die Daten sofort angezeigt und nicht aufgezeichnet oder archiviert werden.

Durchsuchen:

Im Rahmen der Selektion können Erkennungskriterien für einzelne Elemente (Raumteile, Personen, ...) hinzugeschaltet bzw. entfernt werden, die dann fortan (nicht mehr) angezeigt werden sollen. Ein Suchen nach einzelnen Bildelementen, z.B. Tracking von einzelnen Personen, Fahrzeugen etc. ist ebenfalls denkbar.

4.4 “Knackser” aus Schallplatten-Aufnahmen entfernen

Die bisherigen Beispiele basierten wie das ursprüngliche Projekt auch auf grafischen Daten. Daß das Schema jedoch auch auf Audio-Daten angewandt werden kann soll in den folgenden beiden Beispielen belegt werden. Im ersten Beispiel soll das entfernen von “Knacksern” illustriert werden, wie sie entstehen wenn Schallplatten digitalisiert werden.

Segmentierung:

Die Segmentierung von Audio-Signalen geschieht üblicherweise in Samples von 16bit Auflösung, wenn CD-Qualität gewünscht ist ergeben sich so 44100 Samples pro Sekunde bei zwei Tonkanälen (Stereo). Abhängig vom Ausgangsformat erfolgt hier etwa eine rein zeitbasierte Segmentierung, bei codierten / komprimierten Daten ist vorher eine Entkomprimierung vorzunehmen, vergleichbar der MPEG-Video-Dekodierung bei Videostreams, siehe “Segmentierung”, 3.1.

Selektion:

“Knackser” zeigen sich durch charakteristische Muster in der Hüllkurve, vergleich Abbildung 9.

Erkannt werden diese Muster durch Fourier Transformation über das Frequenzspektrum.

Andere Störgeräusche wie Rauschen können durch weitere Filterung der Daten entfernt werden, wobei hier v.a. bei leisen Stellen (Nutz)Signal und Störgeräusch oft nicht mehr unterschieden werden können. Eine qualitiativ hochwertige Aufarbeitung der Daten ist unter diesen Voraussetzungen nicht möglich.

Nachdem die “Knackser” (Samples, die Anomalie im Vergleich zum restlichen Frequenzspektrum aufweisen) erkannt sind, können die zugehörigen Samples ersatzlos entfernt werden. Da diese “fehlerhaften” Samples im Vergleich zur restlichen Datenmenge verschwindend gering (kurz) sind entsteht kein hörbarer Verlust, und es müssen hier keine Ersatz-Daten eingefügt werden.

Synthese:

Je nach angestrebtem Format und Qualität könnnen die verbleibenden Samples entweder unkomprimiert abgespeichert werden, etwa als WAV oder AIFF Datei. Alternativ können auch redundante Frequenzen, Klangstrukturen etc. durch Codierungen entfernt und so das Datenvolumen drastisch reduziert werden.

Durchsuchen:

Eher unüblich, v.a. nach der Synthese, da die Knackser keine “Nutzdaten” sind und deswegen auch weggeworfen werden. Am Endes des Verfahrens sind eher statistische Daten interessant, etwa die Anzahl der entfernten Knackser insgesamt, Durchschnittliche Anzahl der Samples pro Knackser oder der prozentualer Anteil der entfernten Samples am gesamten Musikstück.

4.5 Voice over IP

Segmentierung:

In diesem Beispiel werden analoge Daten in Form von Audio-Signalen übertragen. Die Segmentierung geschieht dabei automatisch zeitbasiert bei der Konvertierung (Analog-Digital-Wandlung) der Audio-Signale in digitale Daten.

Selektion:

Wie bei allen audiobasierten Verfahren kann auch hier eine Aufwertung der Klangqualität durch diverse Filter erfolgen. Diese sind wahlweise vor oder nach der Analog-Digital-Wandlung (= Segmentierung) geschaltet. Neben der Verbesserung der Klangqualität werden aus dem Eingangs-Datenstrom als eigentliche Selektion diejenigen Daten ausgewählt, die wirklich übertragen werden sollen. Bei analogen Telefonsystemen ist hier z.B. der Frequenzbereich zwischen 300Hz und 3.4kHz interessant, bei ISDN wird alles zwischen 0Hz und 120kHz(!) übertragen. [6] Die restlichen Klangdaten werden nicht übertragen/gesampelt, da sie nicht im von jeweiligen Dienst umfassenden Qualitätsbereich sind, und zu grossen Teilen auch für die angebotenen Dienste - üblicherweise Übertragung von Sprache - nicht relevant sind.

Die so ausgewählten Daten können noch weiter codiert werden, um bessere Qualität in den vom menschlichen Gehör eher wahrgenommenen “tiefen” Tönen zu erzielen, z.B. anhand gängiger Codierungsverfahren wie den im Telefonie-Bereich eingesetzten MU-LAW oder A-LAW Verfahren. [7]

Synthese:

Nachdem die (Sprach-)Daten mit den diversen Methoden gefiltert und codiert wurden werden sie für den Transport in Chunks gruppiert, um die Weiterleitung und Wiederzusammensetzung in paketorientierten Netzen wie z.B. TCP/IP, GPRS/i-Mode oder UMTS mit Hilfe von Protokollen wie H.323 oder SIP[1] zu erleichtern. Die Pakete werden vom jeweiligen Netz aufgenommen, und zum Endgerät des Gesprächspartners transportiert. Dort angekommen werden die einzelnen Pakete wieder zu einem Datenstrom zusammengesetzt, der anschließend wieder mittels Digital/Analog Wandlung in Audio-Signale z.B. im Lautsprecher eines Telefons ausgegeben wird.

Durchsuchen:

Das reine Durchsuchen von Voice-Streams ist für Anwender kaum sinnvoll. Für die Sicherung der Übertragung müssen jedoch Mechanismen zur Fehlerkorrektur und Erkennung von Übertragungsstörungen vorhanden sein. Dies kann z.B. durch numerieren der einzelnen Sprachzellen (Chunks, Pakete) geschehen, vgl. Sequenznummern im TCP/IP Protokoll.[2] Dies ist jedoch nur als Maßnahme zur Übertragungssicherung zu verstehen, nicht um den übertragenen Datenstrom durchsuchbar zu machen.

Ist dies gewünscht, so sind seperate Spracherkennungs-Systeme einzusetzen, wie z.B. Echelon. [9]

5 Fazit

Betrachtet man das eingangs vorgestellte Projekt aus dem Blickwinkel der Informationswissenschaften, so ergeben sich eine Reihe interessanter Verfahren, auf die die genannten und im Projekt angewandten Schritte übertragen werden können.

Die beschriebenen Methoden der Segmentierung, Selektion und Synthese lassen sich in den Beispielen wiederfinden, die Indexierung zur Durchsuchung ist abhängig vom angestrebten Ziel ebenfalls bei den meisten Anwendungen realisierbar. Unterschiede ergeben sich hier jeweils aus den genauen Anforderungen der Anwendung.

Die ursprüngliche Anwendung aus dem Bereich des Video- und Image-Retrievals ist auch durchaus auf andere Bereiche und Datenformate übertragbar, wie anhand der Beispiele aus dem Audio-Bereich gesehen werden kann. Weitere Beispiele die das illustrierte Verfahren benutzen können mit hoher Wahrscheinlichkeit auch in vielen anderen Anwendungsgebieten gefunden werden.

[1] Siehe [18] Seite 17.
[2] Siehe [8] Seite 149.

A Literatur

[1] Feyrer, Hubert (2001): Mit dem Regensburger Marathon-Cluster durch's Ziel.
URL: http://www.feyrer.de/marathon-cluster/ [Stand: 2002-06-27]

[2] NOAH - Notfall Organisations- und Arbeits-Hilfe: Rettungsweste des Uni-Klinik.
URL: http://www.noah-regensburg.de/ [Stand: 2002-06-27]

[3] AR Quake Project:
URL: http://wearables.unisa.edu.au/projects.php [Stand: 2002-05-23]

[4] NSF/ARPA Science and Technology Center for Computer Visulaisation and Scientific Visualisation: Kamera mit See-Through-Display.
URL: http://www.cs.unc.edu/ azuma/azuma_AR.html [Stand: 1999-12-17]

[5] Knapp, Louise: Finding Your Inner Fast Eddy. Augmented Reality-gestuetztes Billiard.
URL: http://www.wired.com/news/gizmos/0,1452,52990,00.html [Stand: 2002-06-15]

[6] Sokoll, Thorsten: Telefone, ISDN, xDSL.
URL: http://graphics.cs.uni-sb.de/Courses/ws9900/cg-seminar/Ausarbeitung/Thorsten.Sokoll/ [Stand: 2002-06-27]

[7] Mobile Markeding Ressource Center: Telephony Applications. MU-LAW Audio Encoding.
URL: http://resources.windwire.com/standards/audio.html [Stand: 2002-06-27]

[8] Stevens, W. Richard (1994): TCP/IP Illustrated, Vol. 1 - The Protocols. Reading, MA: Addison Wesley.

[9] Echelonwatch.
URL: http://www.echelonwatch.org/ [Stand: 2002-06-27]

[10] g4u - Harddisk Image Cloning for PCs.
URL: http://www.feyrer.de/g4u/ [Stand: 2001-09-19]

[11] Werbung muß sein - sonst bleibt alles beim alten.
URL: http://www.dooyoo.de/review/419723.html [Stand: 2002-06-27]

[12] Kuhmünch, Christian (1996): Automatische Erkennung von Werbung im Fernsehen.
URL: http://www.informatik.uni-mannheim.de/informatik/pi4/publications/library/Kuhmuench1996a.pd
[Stand: 06-1996]

[13] Antkowiak, Adalbert; Oettl, Sonja ; Santiago, Laurence (2001): Projektseminar Informationretrieval
URL: http://www-nw.uni-regensburg.de/ reb01193.3.stud/v11_2210/default.html [Stand: 2001-10-22]

[14] NetBSD runs a Marathon: Slashdot-Artikel mit Diskussion zu den Themen Marathon & Cluster Computing.
URL: http://slashdot.org/article.pl?sid=01/05/30/2034212&mode=thread&tid=122 [Stand: 2002-07-05]

[15] dumpmpeg: Zerlegt MPEG-Streams in einzelne Bilder.
URL: http://sourceforge.net/projects/dumpmpeg [Stand: 2002-07-06]

[16] mpeg_encode: Berechnet MPEGs aus einzelnen Bildern.
URL: http://bmrc.berkeley.edu/frame/research/mpeg/mpeg_encode.html [Stand: 2002-07-06]

[17] NetBSD: Open Source Betriebssystem unter dem die Client-Rechner des Regensburger Marathon-Clusters liefen.
URL: http://www.netbsd.org [Stand: 2002-07-06]

[18] Wiler, Lars (2002): Voice over IP. Die Datenschleuder #77, ISSN 0930-1054.

[19] del Bimbo, Alberto: Visual Information Retrieval. San Francisco, Californien, USA: Morgan Kaufmann Publishers.