Studienarbeit

Technische Universität Ilmenau

Entwicklung und Implementierung von Methoden zur Erkennung von Werbeblöcken im digitalen Fernsehen

Andreas Regel

10. April 2003

Kurzfassung

Diese Studienarbeit befasst sich mit den Möglichkeiten in Aufzeichnungen digitaler Fernsehsendungen vorhandene Werbung zu erkennen und zu entfernen. Zur Ermittlung der Werbeblöcke werden verschiedenen Methoden spezieller und allgemeiner Art vorgestellt. Diese werden in einer Softwarelösung implementiert, getestet und bewertet.

Die Arbeit entstand an der Technischen Universität Ilmenau, Fakultät für Informatik und Automatisierung, im Fachgebiet Prozessinformatik im Rahmen des Digitalen Video Projektes.

Inhaltsverzeichnis

Abbildungsverzeichnis

1. Einleitung

Das Programmangebot des Fernsehens wurde durch das Entstehen privater Fernsehanstalten stetig erweitert. Im Gegensatz zu den öffentlich-rechtlichen Sendern finanzieren sich diese nicht durch Gebühren, sondern vorwiegend durch die Einnahmen aus der Ausstrahlung von Werbung. Da die Werbung nicht nur zwischen einzelnen Sendungen ausgestrahlt wird, sondern auch das laufende Programm unterbricht, stellt sie ein Ärgernis für den Zuschauer dar.

Beim Fernsehen während der Ausstrahlung muss der Zuschauer mit der Werbung leben. Anders ist es bei der Aufnahme von Sendungen, z.B. zum Zwecke des späteren Sehens oder der Archivierung von Programmen. Hier ist es möglich die Werbung zu entfernen, was während der Aufnahme oder nach deren Abschluss geschehen kann. Für die Automatisierung dieses Prozesses wurden Methoden entwickelt, welche unter Berücksichtigung akustischer und optischer Merkmale des Fernsehsignals eine Unterscheidung zwischen Werbung und restlichem Programm ermöglichen. Da die Fernsehanstalten aber daran interessiert sind, dass der Zuschauer möglichst die Werbung auch zu Gesicht bekommt, werden immer wieder kleine Veränderungen dieser Merkmale vorgenommen. Das führt dazu, dass die Verfahren zur Werbeblockerkennung mit der Zeit unzuverlässiger werden.

Mit der Einführung des digitalen Fernsehens lässt sich Erkennung von Werbeblöcken verbessern. Einerseits können durch die gesteigerte Bild- und Tonqualität die bekannten Verfahren verbessert werden und andererseits lassen sich neue Methoden finden, Werbung vom normalen Programm zu unterscheiden.

1.1. Digitales Video Projekt

Das Digitale Video Projekt (kurz DVP) ist ein Projekt des Fachgebiets Prozessinformatik der Fakultät für Informatik und Automatisierung an der TU Ilmenau. Im Rahmen des DVP soll basierend auf existierenden Hard- und Softwarekomponenten ein digitaler Videorecorder auf einer Standard-PC-Plattform entstehen. Im Vordergrund stehen dabei ein innovatives und ergonomisches Bedienkonzept

und eine flexible modulare Architektur, die es erlaubt verschiedenste Funktionen und Ausstattungsmerkmale zu implementieren.

1.2. Aufgabenstellung

Ziel dieser Arbeit ist es im Rahmen des DVP eine Software zu entwickeln, die es ermöglicht, das Herausschneiden der Werbung aus aufgezeichneten Fernsehsendungen zu automatisieren. Das beinhaltet die Dekodierung des aufgezeichneten Datenstroms sowie die Entwicklung, Verbesserung und Implementierung bekannter sowie neuer Methoden zur Unterscheidung der Werbung vom restlichen Programm. Weiterhin soll eine Integration in die Menüstruktur der Software VDR (Video Disc Recorder) erfolgen, damit die entstehende Funktionalität auch intuitiv vom Anwender benutzt werden kann.

2. Grundlagen

2.1. Digitales Fernsehen nach dem DVB-Standard

Die Einführung des digitalen Fernsehens in den letzten Jahren bringt eine Verbesserung der Bild- und Tonqualität mit sich. Außerdem werden die Möglichkeiten erweitert, da zusätzlich zum Fernsehsignal auch Daten übertragen werden können. Dazu wurde für Europa der DVB-Standard definiert, der sich auch zunehmend weltweit durchsetzt. Dieser gliedert sich noch auf, je nachdem ob das Fernsehsignal über Satellit (DVB-S), Kabel (DVB-C) oder terrestrisch (DVB-T) übertragen wird.

Der DVB-Standard definiert, dass Bild, Ton und Zusatzdienste nach dem MPEG-2 Verfahren (ISO/IEC 13818) komprimiert übertragen werden. Damit ist es möglich in einem herkömmlichen Fernsehkanal je nach Qualität 6 bis 10 digitale Fernsehkanäle unterzubringen[1].

2.1.1. Sender

Vor dem Senden des digitalen Fernsehsignals erfolgt die MPEG-2 Kompression der digitalen Audio- und Videodaten. Falls diese in analoger Form vorliegen, müssen sie vor der Kompression erst noch digitalisiert werden. Die entstehenden komprimierten AV-Daten werden mit anderen Daten zu einem MPEG-2 Transport Stream (TS) multiplext.

Da der Transport Strom fast keine Redundanzen enthält und der kleinste Fehler die Übertragung ruinieren könnte, werden noch Informationen zur Fehlerkorrektur hinzugefügt. Danach erfolgt die Modulation auf die Trägerfrequenz des Kanals, bevor schließlich das Signal auf den verschiedenen Übertragungsmedien (Satellit, Kabel, terrestrisch) gesendet wird.

[1] Die Übertragungskapazität eines Fernsehkanals beträgt ca. 38 Mbps. Bei MPEG-2 Komprimierung wird eine gute Qualität bei Fernsehauflösung (720 x 576) bei etwa 4 bis 5 Mbps erreicht.

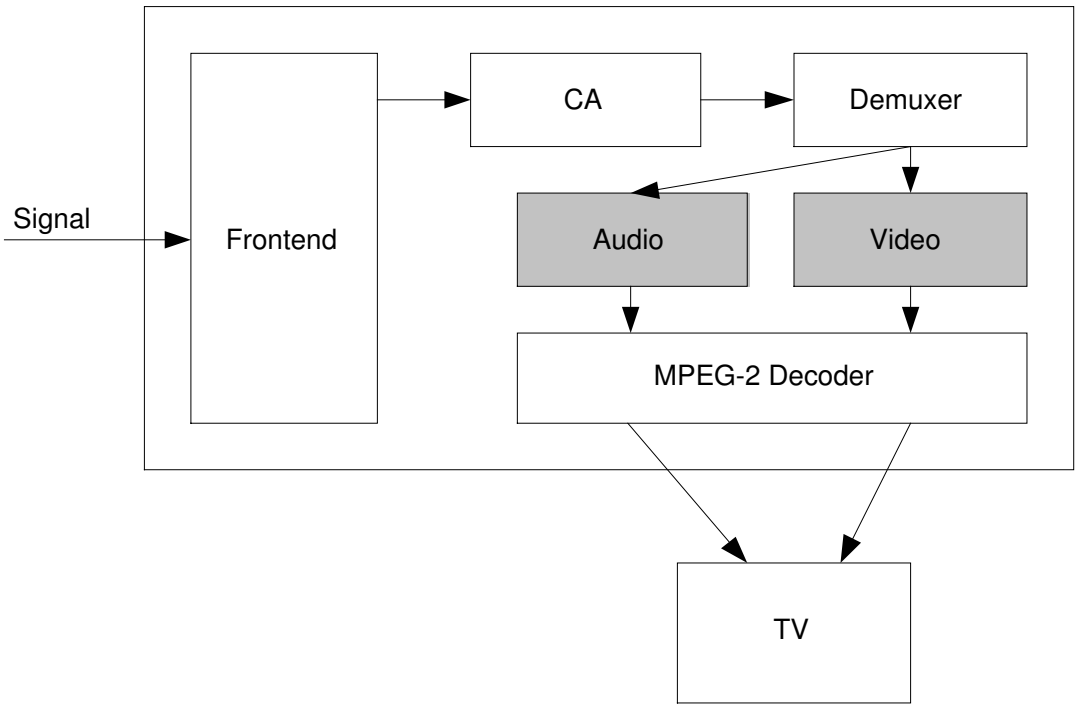


Abbildung 2.1.: Aufbau eines DVB-Empfängers

2.1.2. Empfänger

Der DVB-Empfänger setzt sich aus folgenden Hardwarekomponenten zusammen:

- das Frontend bestehend aus Tuner und DVB Demodulator
- Conditional Access (CA) Hardware zur Echtzeit-Entschlüsselung von Pay-TV-Programmen
- Demultiplexer zur Filterung des eingehenden DVB-Datenstroms
- MPEG-2 Audio- und Videodecoder

Das gesendete Signal wird im Frontend empfangen und zu einem MPEG Transport Stream demoduliert. Der gesamte TS durchläuft die CA Hardware. Dort werden die Programme, zu denen der Nutzer Zugang hat, in Echtzeit decodiert und wieder in den Transport Stream eingefügt. Der Demultiplexer teilt diesen in seine Komponenten auf. Das können Audio- und Videoströme sein, aber zusätzlich sind auch Datenströme mit Informationen zu den bereitgestellten Programmen möglich. Die AV-Daten werden im MPEG-2 Decoder dekomprimiert und können dann auf ein Fernsehgerät ausgegeben werden.

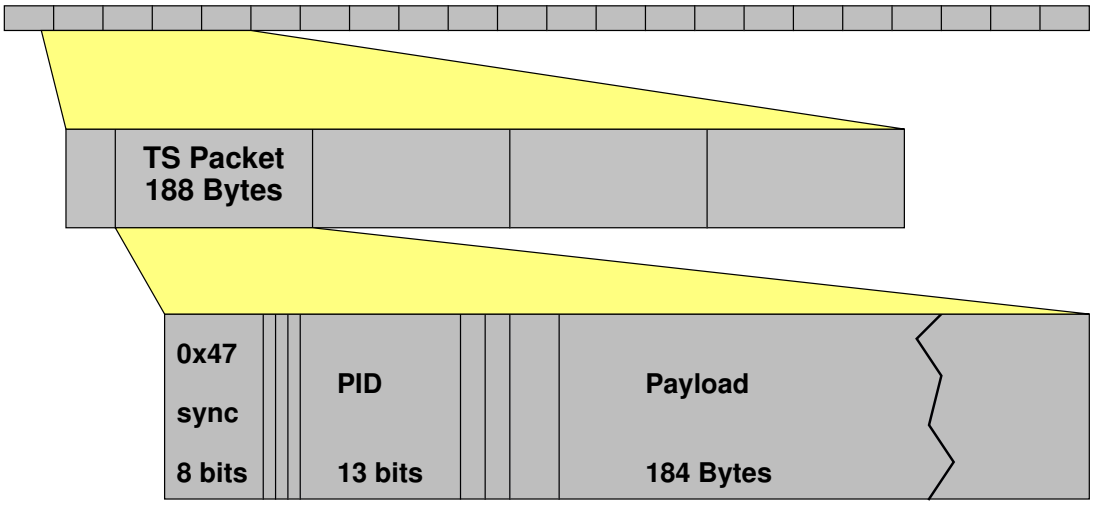


Abbildung 2.2.: Aufbau eines Transport Stream

2.1.3. Datenformat

Transport Stream

Ein Transport Stream setzt sich aus 188 Byte großen Paketen zusammen. Diese können verschiedenste Arten von Daten enthalten, wobei dies in der Mehrheit Audio- und Videoströme der Fernsehprogramme sind. Dabei kann ein Transport Strom viele Programme beinhalten, abhängig von der Bandbreite des Übertragungskanals und der Bitrate der einzelnen Programme. Die Audio/Video-Daten werden als Packetized Elementary Streams (PES) übertragen, welche in die TS Pakete aufgesplittet werden. Jedes dieser Pakete besitzt einen 4 Byte großen Header, beginnend mit einem Synchronisationsbyte[2]. Neben Bits zur Fehlererkennung beinhaltet der Header die 13 bit große PID, die den Datenstrom bezeichnet, zu dem das Paket gehört. So gehören beispielsweise alle Pakete mit der PID 110 zu dem Videostrom des Senders ZDF auf dem Kanal mit der Frequenz 394 MHz im digitalen Kabel[3]. Da der PES sich auch aus Paketen zusammensetzt, enthält der Header noch Informationen darüber, ob ein PES Paket im aktuellen TS Paket beginnt.

[2] Dieses enthält immer den Wert 47hex
[3] Stand März 2003

Stream ID (hex)	Stream Type
BC	Program Stream Map
BD	Private Stream 1
BE	Padding Stream
BF	Private Stream 2
C0 - DF	Audio Stream
E0 - EF	Video Stream

Tabelle 2.1.: Stream IDs eines PES Pakets

Packetized Elementary Stream

Die Packetized Elementary Streams bestehen aus MPEG-Datenströmen, die in Pakete aufgesplittet und durch Multiplexing ineinander verschachtelt sind. Das sind meist Audio- und Videoströme, aber zusätzlich sind auch so genannte Private Streams möglich, die prinzipiell beliebige Daten transportieren können und bei einigen Sendern zur Übertragung von Dolby Digital Ton genutzt werden.

Jedes PES Paket besitzt einen Header, der neben der Paketlänge die Stream ID zur Kennzeichnung des enthaltenen Datenstroms enthält. Die wichtigsten Identifikationsnummern sind in Tabelle 2.1 aufgeführt. Nach dem Header können Zeitinformationen in Form des Presentation Time Stamp (PTS) folgen, die zur Synchronisation von Bild und Ton bei der Wiedergabe dienen. Die eigentlichen Video- und Audioframes werden im Nutzdatenteil der PES Pakete übertragen.

Video Stream

Die Video Streams sind Bestandteil des PES. Sie enthalten aufeinander folgende MPEG Video Frames. Jedes dieser komprimierten Einzelbilder wird durch eine Reihe von Headern eingeleitet, in welchen Informationen über das Frame wie Bildformat, Auflösung, Bildrate enthalten sind, die zur Dekodierung benötigt werden. Daran anschließend folgen die eigentlichen komprimierten Bilddaten. Diese werden im YUV-Format übertragen, d.h. sie bestehen aus der Helligkeitskomponente Y, auch Luminanz genannt, und den Farbdifferenzsignalen U und V, welche auch Chrominanz genannt werden.

Audio Stream

Die Audio Streams, auch Bestandteil des PES, enthalten aufeinander folgende MPEG Audio Frames. Jedes dieser Frames beginnt mit einem 4 Bytes großen Header, dem die komprimierten Tonsignale folgen. Der Header enthält für den Decoder benötigte Informationen wie Layer, Bitrate, Samplingrate und Größe des Frames.

Private Stream

Der Private Stream des PES kann prinzipiell beliebige Daten enthalten. Meist wird er jedoch benutzt um Dolby Digital Tonsignale zu übertragen. Wenn das der Fall ist, enthält der Private Stream eine fortlaufende Folge von Audio Frames im AC3 Format, die jeweils mit einem Header eingeleitet werden.

2.2. Video Disc Recorder

Der Video Disc Recorder, kurz VDR, ist eine Software für das Betriebssystem Linux, welche auf einem Standard-PC die komplette Funktionalität eines digitalen Videorecorders bereitstellt. Es können digitale Fernseh- und Radioprogramme empfangen und geschaut bzw. gehört werden, wobei auch die Zusatzinformationen im DVB-Datenstrom ausgewertet werden um dem Anwender einen elektronischen Programmführer[4] zur Verfügung zu stellen.

VDR bietet die Möglichkeit die Fernseh- bzw. Radioprogramme auf die Festplatte des PCs aufzuzeichnen, was auch timergesteuert geschehen kann. Die Aufzeichnungen können natürlich wiedergegeben werden, was sogar während deren Stattfinden zeitversetzt möglich ist (Time-Shift-Verfahren). Während der Wiedergabe können Schnittmarken gesetzt werden, woraufhin ein Schneiden der Aufzeichnung durchgeführt werden kann um beispielsweise die Werbeblöcke aus einem Film zu entfernen.

Die Steuerung der Funktionen des VDR erfolgt per Infrarotfernbedienung oder PC-Tastatur. Über ein Menü, das im OSD dargestellt wird, kann auf alle Funktionalitäten des VDR zugegriffen werden, hier sind auch Einstellungen vorzunehmen, die diese beeinflussen. Im OSD wird auch der EPG dargestellt, wie in Abbildung 2.3 zu sehen.

[4] Electronical Programme Guide, kurz EPG

Abbildung 2.3.: OSD des VDR mit Electronical Programme Guide

Damit VDR funktionieren kann, muss eine DVB-PCI-Karte im Computer installiert sein. Diese beinhaltet alle der oben genannten Komponenten eines DVB-Empfängers. VDR greift auf diese Komponenten über das vom Linux DVB Treiber zur Verfügung gestellte API zu.

In der aktuellen Entwicklerversion von VDR wurde eine Plugin-Schnittstelle integriert. Damit wird es möglich die Funktionalität zu erweitern ohne den Quellcode von VDR direkt ändern zu müssen. Es sind Erweiterungen verschiedenster Art möglich. So existieren beispielsweise Plugins zur Wiedergabe unterschiedlicher Medien, wie DVD, Video-CD und MP3-Dateien, zum Streamen von Videodaten über Netzwerke und zur Darstellung von Teletext. Sogar einfache Spiele sind möglich, wie das Tetris- und TicTacToe-Plugin beweisen.

3. Stand der Technik in der Werbeblockerkennung

3.1. Merkmale zur Unterscheidung von Werbung und Programm

3.1.1. Senderlogo

Jeder Fernsehsender ist verpflichtet, sich anhand seines Logos zu identifizieren. Bei der kommerziellen Werbung hingegen bleibt diese Identifikation aus, es sei denn, es handelt sich um Eigenwerbung. Da die Werbenden für die Inhalte ihrer Spots verantwortlich sind und nicht die Sender, die sie ausstrahlen, muss kein Logo dargestellt werden. Deshalb bietet sich eine Logoerkennung an um die Werbung zu identifizieren.

Die Logos können folgende Charakteristiken aufweisen:

- weiß, nicht transparent
- weiß, transparent
- farbig, nicht transparent
- farbig, transparent

Da schon einige Ansätze existieren, die mit Hilfe der Logoerkennung versuchen die Werbung zu erkennen, überlegen sich die Sender immer wieder neue Tricks, damit diese Verfahren unzuverlässiger werden. So wird bei vielen Sendungen das Logo erst einige Sekunden[1] nach dessen Beginn wieder eingeblendet. Außerdem werden des öfteren Programmhinweise eingeblendet, während denen das Logo häufig ausgeblendet wird. Auch variiert die Position des Logos bei vielen Sendern, sie ist oft unterschiedlich bei 4:3 und 16:9 Programmen.

[1] Das können auch schon mal bis zu 30 Sekunden sein.

Abbildung 3.1.: Jingle zu Beginn der Werbung

Abbildung 3.2.: Jingle am Ende der Werbung

3.1.2. Jingle

Ein Jingle ist eine kurze Melodie, wie sie von den Fernseh- bzw. Radiosendern zu Beginn der Werbesendung eingespielt wird. Im Bereich des Fernsehens wird dieses akustische Merkmal oft um sein optisches Pendant erweitert. Da per Gesetz vorgeschrieben ist, dass die Werbung für den Zuschauer eindeutig als diese zu erkennen sein muss, zeigen bei vielen Fernsehanstalten kurze Videosequenzen den Wechsel von der eigentlichen Sendung zur Werbung an. Umgekehrt gibt es auch meist einen sichtbaren Hinweis auf das Ende der Werbepause.

Da bei den meisten Sendern je mehrere Jingles existieren und diese sich auch des öfteren ändern, bietet sich dieses Merkmal nicht unbedingt für ein Verfahren zur Identifizierung der Werbung an. Allerdings enthalten viele dieser Videosequenzen den Schriftzug „Werbung“ irgendwo sichtbar im Bild, der wenn er durch einen Algorithmus zu erkennen wäre, sich für ein solches Verfahren anbieten würde.

Für weitere Informationen sei auf die Diplomarbeit von F. Pelster [Pel98] verwiesen, in der eine Erkennung von Jingles im Hörfunkbereich realisiert wurde.

3.1.3. Austastzeilen

Während das Fernsehen nach PAL, NTSC oder SECAM Standard ein Bildformat mit dem Seitenverhältnis[2] 4:3 hat, verwenden Kinofilme andere Formate. Diese reichen von 1,34:1 (Normalformat) bis hin zu 2,35:1 (Cinemascope). Die meisten Filme auf der Leinwand haben also ein sehr viel breiteres Format als das

[2]Länge der Zeilen im Verhältnis zur Höhe des Bildes

normale Fernsehbild. Damit das Bild verzerrungsfrei auf einem herkömmlichen Fernseher wiedergegeben werden kann, wird bei solchen Filmen, aber auch bei zahlreichen Musikvideos und einigen Fernsehserien, der obere und untere Rand des Fernsehers „schwarz ausgetastet“. Wenn nun ein Verfahren zur Erkennung von Werbeblöcken feststellt, das bei einem Filme diese schwarzen Balken mitgesendet werden, kann das reichen um die Werbung eindeutig zu identifizieren. Da aber auch einige Werbespots und die Programmvorschau eines Fernsehsenders diese Austastzeilen vorweisen, wird ein Verfahren, welches kein weiteres Merkmal heranzieht, nicht sehr genau arbeiten können.

3.1.4. Mono-/Stereoton

Bei vorwiegend alten Filmen und Serien kann es vorkommen, dass sie nur mit Mono aufgezeichnet und daher auch nur mit Mono wiedergegeben werden können. Die heutige Werbung wird hingegen fast ausschließlich im Stereoverfahren ausgestrahlt. Bildet man die Differenz zwischen dem linken und rechten Audiokanal, wird bei einer Mono-Sendung die Differenz null, dagegen bei einer Stereo-Sendung von null abweichen.

3.1.5. Lautstärkeunterschied

Bei vielen Sendern ist festzustellen, dass die Werbung lauter gesendet wird als das normale Programm. Eine Audiostufe sollte die Lautstärkezunahme zu Beginn der Werung feststellen können. Allerdings ob sich anhand dieses Kriteriums schnell genug detektieren lässt, ob die Lautstärkezunahme vom Wechsel Sendung-Werbung stammt oder ob die Lautstärke im Fernsehprogramm erhöht wurde, ist ohne eine längerfristige Analyse durch Vergleichsmessungen nicht vorhersehbar.

3.2. Realisierte Verfahren

3.2.1. Reklame Ade

Im Rahmen des Jugend Forscht Wettbewerbs hat Tobias Kramer 1995 eine Elektronik entwickelt, die, zwischen Fernseher und Videorecorder geschaltet, bei den Aufnahmen den Recorder stoppt, sobald die Werbung beginnt [Kra95]. Das Verfahren basiert auf einer Logoerkennung folgender Art: Die linke obere Ecke des

aktuellen Videosignals wird digitalisiert und mit einer gespeicherten Schablone des Senderlogos Byte für Byte verglichen. Dies führt zu Schwierigkeiten bei der Erkennung bei transparenten Logos, die aber zunehmend eingesetzt werden. Auch ist die Fixierung auf eine feste Position nicht mehr zeitgemäß, da die Senderlogos sehr häufig in der oberen rechten aber auch in den unteren Ecken auftreten können.

3.2.2. NoAd

Die von Fabian Zink 1998 im Rahmen einer Diplomarbeit entwickelte Software NoAd läuft unter dem Betriebssystem Windows auf einem Standard-PC [Zin98]. Das aktuelle Fernsehsignal wird über eine im PC installierte TV-Karte aufgenommen, die Software entscheidet ob Werbung vorliegt oder nicht und per an die serielle Schnittstelle angeschlossenem Infrarot-Transmitter werden an einen Videorecorder die Kommandos zum Stoppen oder Starten der Aufnahme gesendet. NoAd trifft seine Entscheidung basierend auf einer Logoerkennung und einer Messung der Austastzeilen.

Der Ablauf des Programms beginnt mit einer vom Benutzer gesteuerten Lernphase, die nur einmal pro Logo durchgeführt werden muss. Als erstes wird per Mittelwertbildung über die Helligkeitskomponente fortlaufender Fernsehbilder der auffälligste Punkt[3] gesucht um die Position des Logos festzustellen. Dazu sind je nach Hintergrund etwa 50 bis 80 Einzelbilder nötig. Ist das Logo gefunden, erfolgt die Ermittlung dessen Eigenschaften in einem Bereich von 100×100 Pixeln um die gefundene Position. Dazu gehören die horizontalen und vertikalen Abmessungen, sowie bis zu 3000 Referenzpunkte, die zur späteren Identifikation herangezogen werden. Das geschieht wieder durch Mittelwertbildung über mehrere fortlaufende Fernsehbilder, diesmal werden aber die einzelnen Farbkomponenten berücksichtigt. Die Ergebnisse der Lernphase werden zur späteren Verwendung abgespeichert.

Der Algorithmus zur Entscheidung ob Werbung vorliegt oder nicht, gliedert sich in zwei Teile. Als erstes werden die Austastzeilen ermittelt, wenn das zu keinem Ergebnis führt erfolgt die Logoidentifizierung.

Zur Ermittlung der oberen Austastzeilen werden die Helligkeiten der Bildzeilen von oben nach unten berechnet. Solange diese unter einem Schwellwert liegen, werden sie als Austastzeilen gewertet. Steigt die Helligkeit darüber ist der Beginn des eigentlichen Bildes gefunden. Die unteren Austastzeilen werden ent-

[3] Der auffälligste Punkt ist der mit der größten Abweichung vom Mittelwert.

sprechend umgekehrt von unten nach oben gesucht. Zur Sicherheit wird noch überprüft ob die näher zur Bildmitte liegenden Zeilen auch hell genug sind und nicht nur ein dunkles Bild vorliegt.

Die Identifizierung des Logos erfolgt über einen Vergleich der Helligkeiten der Messpunkte im aktuellen Bild mit denen der Referenzpunkte. Je nachdem ob das Logo hell oder dunkel ist, müssen die Messpunkte heller oder dunkler als die Referenzpunkte sein um als Logopunkte interpretiert zu werden. Für eine sichere Logoerkennung muss die Anzahl der erkannten Logopunkte mindestens drei mal so groß sein wie die der nicht erkannten. Wenn die Anzahlen etwa gleich sind, wird davon ausgegangen, dass zur zeit kein Logo gesendet, also Werbung vorliegt. Zur Sicherheit müssen zwei aufeinander folgende Fernsehbilder zur gleichen Entscheidung führen, damit diese akzeptiert wird.

Die Software NoAd arbeitet prinzipiell sehr schnell und zuverlässig. Schwierigkeiten gibt es aber insbesondere, wenn die im Abschnitt 3.1 beschriebenen Probleme bei der Logoerkennung, wie die verspätete Einblendung und das kurzzeitige Fehlen bei Programmhinweisen, vorliegen. Außerdem ist die Erkennung der Austastzeilen nur sehr grob gefasst, was zur Aufzeichnung von im Breitbildformat ausgestrahlten Werbespots und Programmvorschauen führt.

3.2.3. NoAd2

NoAd2 ist eine auf NoAd aufbauende Software, die 1999 im Rahmen einer Diplomarbeit von Thorsten Janke und Markus Koppers entwickelt wurde [JK99]. Im Gegensatz zur ersten Version läuft sie unter dem Betriebssystem Linux. Das Hauptanliegen war es die Algorithmen zur Erkennung der Werbung zu optimieren. Als Grundlage dient hier nur die Logoerkennung.

Zum Finden des Logos in der Lernphase wird als erstes eine Tiefpassfilterung über eine Folge von Einzelbildern durchgeführt, was zu einer Minimierung des dynamischen Anteils führt. Da sich das Senderlogo in gewisser Weise von seiner Umgebung unterscheidet, entsteht an den Stellen der Übergänge zwischen zwei Bildpunkten eine große Helligkeitsdifferenz. Nach der Betrachtung einer Sequenz von Bildern bleiben nur noch gültige Logoübergänge übrig. Diese werden zur Ermittlung der Referenzpunkte genutzt. Bei der Identifikation des Logos werden nun die Helligkeiten der ermittelten Referenzpunkte innerhalb mit denen außerhalb des Logos verglichen. Wenn das Logo im Bild vorhanden ist, sollten diese bei möglichst vielen der Referenzpunkte über einem Schwellwert liegen.

Die Logoerkennung in NoAd2 ist sehr zuverlässig bei allen Arten von Senderlogos. Da es aber auf die Logoerkennung als einziges Merkmal beschränkt ist,

gibt es auch hier Schwierigkeiten bei den oben genannten Problemen bei Senderlogos.

3.2.4. Digitale Videorecorder

Mit der zunehmenden Digitalisierung des Fernsehens erscheinen auch immer mehr Geräte auf dem Markt, die eine digitale Speicherung der Fernsehprogramme auf Festplatte oder DVD ermöglichen, die digitalen Videorecorder. Dabei existieren 2 Arten solcher Geräte:

- Analog-Empfänger, die die analogen Signale digitalisieren, mit einer wählbaren Qualität komprimieren und aufzeichnen.
- DVB-Empfänger, die das empfangene Signal direkt ohne Qualitätsverlust aufzeichnen.

Bei der Wiedergabe dieser Aufzeichnungen bringen so gut wie alle digitalen Videorecorder Funktionen mit, die dem Nutzer ermöglichen schnell die Werbeblöcke zu überspringen, was durch den wahlfreien Zugriff auf das Speichermedium möglich wird. Das können zum Beispiel minutenweises Springen durch die Aufzeichnung sein.

Manche Geräte erkennen die Werbeblöcke und erlauben es bei der Wiedergabe einer Aufnahme das automatische Überspringen. Genau dies bietet die Commercial Advance genannte Funktionalität des ReplayTV 5000 von Sonicblue. Der Hersteller gibt an, dass die Korrektheit dieser Funktion unter Laborbedingungen bei über 95% liegt. In praktischen Test hat sich gezeigt, dass sie je nach Material eher bei 70% bis 90% einzuordnen ist.

3.3. Fazit

Die hier vorgestellten Verfahren zur automatischen Werbeblockerkennung funktionieren alle nicht annähernd hundertprozentig korrekt. Das liegt größtenteils daran, dass sie alle schon einige Jahre alt sind. Die Fernsehanstalten schlafen nicht und versuchen durch leichte Änderungen im Sendeverhalten, wie die verspätete Einblendung oder das zwischenzeitliche Fehlen des Senderlogos, die bekannten Algorithmen zu täuschen. Wenn das passiert, wird Werbung an Stellen im Programm erkannt, an denen z.B. ein Film schon wieder begonnen hat oder noch läuft. Den Verfahren gerät dann zum Nachteil, dass sie meist nur ein

Erkennungsmerkmal, das Senderlogo, berücksichtigen. Ein weiteres Problem ist, dass alle vorgestellten Ansätze auf dem analogen Fernsehen basieren. Damit kann die Zuverlässigkeit der einzelnen Erkennungsmethoden stark sinken, wenn das Signal durch ein mehr oder weniger starkes Rauschen gestört ist, was bei der Übertragung des analogen Fernsehens nicht zu vermeiden ist.

4. Lösungsansatz

Das hier vorgestellte Verfahren zur Erkennung von Werbeblöcken basiert auf dem digitalen Fernsehsignal. Damit erhöht sich prinzipiell schon die Zuverlässigkeit der einzelnen Methoden, da das Signal immer die gleiche gute Qualität besitzt und nicht durch ein bei der Übertragung hinzukommendes Rauschen gestört wird. Außerdem ergeben sich durch den Aufbau der digitalen Datenströme einige neue Merkmale zur Unterscheidung von Werbung und der eigentlichen Sendung.

Um die Zuverlässigkeit des Algorithmus zu steigern wird nicht nur ein Erkennungsmerkmal berücksichtigt, sondern es werden die Ergebnisse mehrerer miteinander kombiniert. Unter anderem werden auch einige Spezialfälle behandelt, die nur bei bestimmten gesendeten Programmen zum Tragen kommen.

Im folgenden werden als erstes neue Unterscheidungsmerkmale sowie erweiterte und verbesserte Erkennungsmethoden aufgezeigt, bevor dann der Algorithmus in seiner Gesamtheit vorgestellt wird.

4.1. Unterscheidungsmerkmale und Erkennungsmethoden

4.1.1. Senderlogo

Durch Beobachtungen wurde festgestellt, dass die Logos der meisten deutschen Fernsehsender hell und mehr oder weniger transparent sind. Der folgend vorgestellte Algorithmus wurde deshalb auf diesen Typ optimiert. Die Erkennung dunkler Senderlogos wird vom zugrunde liegenden Verfahren zwar prinzipiell unterstützt, zum vollständigen Funktionieren wären aber kleine Änderungen notwendig.

Die Erkennung und Identifikation des Senderlogos erfolgt über ein so genanntes Template Matching[1]. Im ersten Schritt wird auf der Helligkeitskomponente des aktuellen Fernsehbildes eine einfache Kantendetektion mittels Differenzope-

[1] vgl. [SOS00], Kapitel 3.8

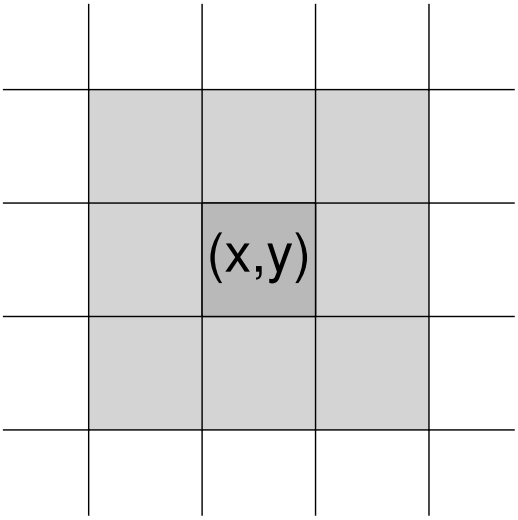


Abbildung 4.1.: 8-ter Nachbarschaft des Punktes (x,y)

ratoren durchgeführt. Hierfür wird der Sobel-Operator verwendet (4.1).

$$\Delta_1 f(x,y) = \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix} \qquad \Delta_2 f(x,y) = \begin{pmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{pmatrix} \tag{4.1}$$

Dabei werden zwei Faltungsmasken auf das Bild angewendet, $\Delta_1 f(x,y)$ für die Detektion der vertikalen und $\Delta_2 f(x,y)$ für die der horizontalen Kanten. Die Faltung geschieht wie folgt: Die 8-ter Nachbarschaft[2] eines jeden Bildpunktes wird mit der Matrix $\Delta_1 f(x,y)$ bzw. $\Delta_2 f(x,y)$ multipliziert und die Elemente des Ergebnisses aufsummiert. Dies führt zu dem Resultat, dass ein Punkt um so stärker hervorgehoben wird, je mehr er sich in der Helligkeit von seinen vertikalen bzw. horizontalen Nachbarn unterscheidet. Die nach Anwendung der zwei Matrizen entstehenden zwei Kantenbilder werden per Addition zu einem vollständigem Kantenbild kombiniert. Aus diesem wird schließlich noch durch Schwellwertrechnung[3] ein Binärbild gewonnen, in dem nur Pixel mit den Grauwerten 0 und 255 vorkommen. Die Kantendetektion ist damit abgeschlossen, ein mögliches Ergebnis ist in Abbildung 4.2 zu sehen.

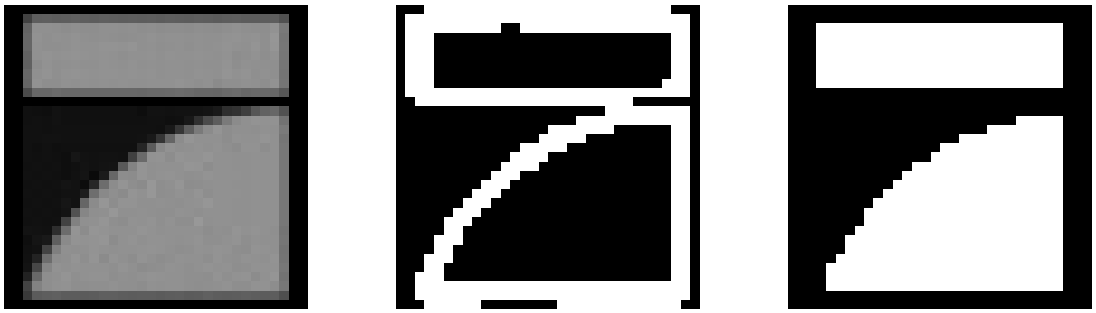
Nach der Kantendetektion erfolgt das Template Matching. Bevor dies durchgeführt werden kann, muss für jedes zu identifizierende Logo eine Schablone und eine Maske existieren. Die Schablone, die für das eigentliche Template Matching

[2] vgl. Abbildung 4.1

[3] Als Schwellwert wird 127 verwendet. Alle Pixel mit größerer Helligkeit werden weiß, alle anderen werden schwarz.

(a) (b)

Abbildung 4.2.: Helligkeitskomponente eines Fernsehbildes (a) und Ergebnis nach der Kantendetektion mit Schwellwertrechnung (b)



(a) (b) (c)

Abbildung 4.3.: Helligkeitskomponente (a) des Logos von Pro7 mit zugehöriger Schablone (b) und Maske (c)

benötigt wird, entspricht einem Kantenbild des Senderlogos[4]. In der Maske sind die Bereiche des Logos weiß dargestellt, in denen keine Kanten verlaufen[5]. Sie dient der Korrektheitssteigerung der Logoerkennung.

Das Template Matching wird benutzt um Objekte bekannter Form, Größe und Orientierung in einem Bild zu erkennen. Diese Methode ist einfach und effektiv. Eine Filtermaske, dessen Koeffizienten ein Teilbild darstellen, das dem zu findenden Objekt entspricht, wird über das gesamte Bild angewendet. Dies entspricht einer Kreuzkorrelation wie in Formel (4.2), wobei I das Kantenbild und F die Filtermaske darstellt. N ist der Normalisierungsfaktor, der der Anzahl der weißen Punkte in der Filtermaske entspricht.

$$I'(x,y) = \frac{\sum_m \sum_n F(m,n) I(m-x, n-y)}{N} \tag{4.2}$$

[4]vgl. Abbildung 4.3(b)
[5]vgl. Abbildung 4.3(c)

Abbildung 4.4.: Ergebnis nach Anwendung des Template Matching auf Kantenbild 4.2(b) mit Schablone 4.3(b). In der oberen rechten Ecke ist ein deutliches Maximum mit einer Helligkeit von 220 entstanden.

Wenn die Maske im Originalbild I gefunden wird, entsteht an dieser Stelle im Resultat I' ein Intensitätsmaximum.

Auf die Logoerkennung angewendet, wird beim Template Matching die Schablone des Senderlogos mit dem Kantenbild des aktuellen Videosignales kreuzkorreliert. Danach wird im entstandenen Bild[6] der Punkt der größten Intensität gesucht und als Position des Senderlogos akzeptiert, falls seine Helligkeit über einem Schwellwert liegt[7].

Zur Steigerung der Korrektheit der Logoerkennung sind noch einige zusätzliche Überprüfungen nötig. Damit der Algorithmus an Stellen mit vielen, dicht beieinander liegenden Kanten nicht irrtümlich ein Senderlogo erkennt, wird mit Hilfe der Logomaske an diesen Stellen die Helligkeit im Kantenbild überprüft. Sie darf einen bestimmten Wert nicht überschreiten, damit der Punkt als Position des Logos akzeptiert wird. Weiterhin wird eine Logoposition erst akzeptiert, wenn die Helligkeit im Videosignal an dieser Stelle einen bestimmten Wert nicht unter-

[6] vgl. Abbildung 4.4

[7] Durch Testen hat sich ein Schwellwert von 200 als gut erwiesen

schreitet. Ein vorher gefundenes Logo wird erst als nicht vorhanden gewertet, wenn die Helligkeit im Bild an dieser Stelle einen bestimmten Wert unterschritten hat, da ein helles Senderlogo sich von einem hellen Hintergrund nicht stark genug abhebt und damit nur sehr schlecht erkannt werden kann.

4.1.2. Schwarze Ränder

Bei der Untersuchung von aufgenommenen Fernsehprogrammen ist aufgefallen, dass nicht nur Filme im Breitbildformat die so genannten Austastzeilen aufweisen. Nahezu bei jeder Sendung, auch im normalen 4:3 Format, lassen sich mehr oder weniger breite schwarze Ränder feststellen und das nicht nur oben und unten sondern auch links und rechts, wie in den Abbildungen 4.5 und 4.6 zu sehen ist. Positiv dabei ist, dass während einer Sendung, insbesondere bei Filmen und Serien, diese Ränder relativ konstant bleiben, währenddessen sie in der Werbung mit nahezu jedem Spot wechseln. Aufgrund dieser Eigenschaft lässt sich das Merkmal sehr gut für die Werbeblockerkennung nutzen.

Die Randerkennung funktioniert folgendermaßen: Sofern die mittlere Helligkeit des Fernsehbildes einen bestimmten Wert nicht unterschreitet, wird mit dem Suchen der horizontalen Ränder begonnen. Dafür wird die Luminanzkomponente des Bildes zeilenweise durchlaufen, für den oberen Rand von oben nach unten und für den unteren Rand entsprechend umgekehrt. Während dieses Durchlaufs wird die durchschnittliche Helligkeit pro Pixel sowie die Anzahl der schwarzen Punkte[8] der aktuellen Zeile berechnet. Es werden zwei Kriterien überprüft, die zum Erkennen des schwarzen Randes führen können. Wenn sich die durchschnittliche Helligkeit oder der prozentuale Anteil von schwarzen Punkten von einer Zeile zur nächsten verdoppeln, ist das Ende des Randes gefunden. Die Randerkennung wird dagegen erfolglos abgebrochen, wenn der Anteil der schwarzen Pixel 70 Prozent überschreitet oder 150 Zeilen durchlaufen wurden.

Das Finden der vertikalen Ränder erfolgt spaltenweise nach den gleichen Kriterien von links nach rechts bzw. umgekehrt, wobei je maximal 50 Spalten durchlaufen werden.

4.1.3. Schwarz-Weiß Sendungen

Ein Spezialfall für die Werbeblockerkennung sind Schwarz-Weiß Programme. Bei solchen Sendungen ist die Werbung leicht identifizierbar, da diese fast immer

[8] Ein Pixel wird als schwarz gezählt, wenn seine Helligkeit kleiner als 21 ist

Abbildung 4.5.: Schwarze Ränder während einer Fernsehserie

Abbildung 4.6.: Schwarze Ränder während eines Werbespots

farbig ist.

Um zu Erkennen ob ein Schwarz-Weiß Programm vorliegt, werden die Chrominanzkomponenten des Fernsehbildes untersucht. Wenn das Bild keinen Farbanteil besitzt liegen die Werte dieser Komponenten um den Mittelpunkt des zulässigen Bereiches also zwischen 126 und 130. Der Algorithmus zählt nun die Punkte, die in diesem Bereich liegen. Das es ab und zu mal Ausreißer nach oben und unten geben kann, wird gefordert, dass der Anteil dieser Punkte mindestens 90 % beträgt, damit ein Schwarz-Weiß Bild erkannt wird. Ansonsten wird es als Farbbild gewertet.

4.1.4. Dolby Digital Ton

Der Sender Pro7 strahlt im Private Stream 1 Dolby Digital Ton im AC3 Format aus. Dies geschieht meist mit 2-Kanal Stereo-Ton. Einige neuere Filme werden jedoch im Surround-Format gesendet, wobei die unterbrechende Werbung weiterhin im Stereo-Ton übertragen wird. Da dies in Zukunft immer häufiger der Fall sein wird und sicherlich auch bei anderen Sendern genutzt werden wird, findet sich dieser Spezialfall im Gesamtalgorithmus wieder.

Die Ermittlung des aktuellen Kanalmodus ist sehr schnell und einfach möglich. Sie erfordert lediglich die Auswertung einiger Bits im Header der AC3 Frames.

4.1.5. Mono Sendungen

Bei Filmen und Serien, die nur mit Mono-Ton vorliegen und deshalb auch nur so ausgestrahlt werden können, lässt sich die Werbung gut erkennen, da diese vorwiegend im Stereo-Format vorliegen. Nun kann solche eine Sendung direkt im Mono-Format oder im Stereo-Format ausgestrahlt werden, wobei bei letzterem der linke und der rechte Tonkanal gleich sind.

Ob Mono-Ton ausgestrahlt wird, kann in zwei Schritten ermittelt werden. Als erstes wird im Header des MPEG Audio Frames nachgesehen, ob dieses schon im Mono-Format vorliegt. Ist dies nicht der Fall, wird im dekodieren Audio Frame der linke Kanal mit dem rechten verglichen und bei annähernder Übereinstimmung wird es als Mono-Signal gewertet. Ansonsten liegt ein Stereo-Signal vor.

4.1.6. Weitere Merkmale

Hier werden Unterscheidungsmerkmale vorgestellt, die im DVB-Datenstrom prinzipiell auftreten können, aber zur Zeit nicht im Gesamtalgorithmus verwendet

werden, da sie entweder überhaupt nicht oder nur von den öffentlich-rechtlichen Sendeanstalten genutzt werden.

Mehrkanalton

Bei einigen Filmen werden mehrere Tonkanäle übertragen, beispielsweise andere Sprachen oder Zusatzinformationen für Blinde. Wenn diese Programme durch Werbung unterbrochen würden, wäre während der Werbepause nur ein Kanal nötig, oder alle würden das gleiche Signal übertragen. Ob ein Film mit mehreren Tonkanäle ausgestrahlt wird, lässt sich einfach durch die Auswertung von Headerinformationen der PES Pakete entscheiden. Da aber nur die öffentlich-rechtlichen Sender diese Möglichkeit auch nutzen und diese Filme nicht durch Werbung unterbrochen werden, ist dieses Merkmal für die Werbeblockerkennung nicht von Nutzen.

PAL Plus Sendungen

Programme im PAL Plus Format, also im 16:9 Breitbildformat, werden im digitalen Fernsehen auf besondere Art und Weise gesendet. Das Bild wird anamorph codiert, d.h. nur das aktive Bild ohne Austastzeilen wird gestreckt im 4:3 Format übertragen. Auf Empfängerseite wird das verzerrte Bild auf die richtige Größe skaliert. Auf diese Art und Weise wird die Bildqualität gesteigert, da das aktive Bild in vertikaler Richtung mit höherer Auflösung übertragen wird. Programme, die in diesem Format gesendet werden, sind durch Informationen im Header des Videostroms gekennzeichnet und lassen sich deshalb sehr einfach erkennen. Da Werbung im normalen 4:3 Format gesendet wird, wäre eine einfache und genaue Unterscheidung möglich. Allerdings wird diese Möglichkeit auch nicht von den privaten Sendeanstalten genutzt, hier wird immer im Normalformat gesendet und Breitwandfilme werden mit Austastzeilen übertragen.

Verschiedene Tonformate

Das Format des MPEG Audio Datenstroms erlaubt es, das Tonsignal mit unterschiedlichen Parametern zu übertragen. Damit ist es möglich bei bestimmten Sendungen, wie Spielfilmen oder Konzerten, die Qualität zu erhöhen, wogegen bei Werbung die Qualität zwecks Bandbreiteneinsparung wieder gesenkt werden könnte. In welchem Format das Tonsignal übertragen wird, ist durch eine Auswertung des Headers der Audio Frames einfach und schnell möglich. Da die privaten

Sendeanstalten von dieser Möglichkeit keinen Gebrauch machen, ist auch dieses Merkmal für die Werbeblockerkennung nicht von Nutzen.

4.2. Angewandter Algorithmus

Der Gesamtalgorithmus zur Erkennung der Werbeblöcke läuft in 2 Phasen ab, die Datensammlung und die Analyse. Zu den berücksichtigten Merkmalen zählen das Senderlogo, die schwarzen Ränder, Schwarz-Weiß Programme und Dolby Digital Ton. Die Erkennung von Sendungen im Mono-Format konnte wurde nur teilweise implementiert und konnte mangels ausgiebiger Tests nicht im Gesamtverfahren berücksichtigt werden.

Die Phase der Datensammlung findet statt, während die Daten der zu untersuchenden Aufnahme durchlaufen und dekodiert werden. Für jedes Audio bzw. Video Frame erfolgt die Untersuchung der zutreffenden Merkmale, also die Erkennung des Senderlogos, der schwarzen Ränder, ob ein Schwarz-Weiß Bild vorliegt und wie der Kanalmodus des Dolby Digital Tons ist. Falls Änderungen in diesen Merkmalen erkannt werden. erfolgt die Speicherung der neuen Daten mit deren Zeitindex, indem sie der Liste der Änderungen des jeweiligen Merkmals hinzugefügt werden. Dabei findet eine Vorverarbeitung statt, bei der kurzfristige Schwankungen erkannt und gegebenenfalls ignoriert werden.

Nach dem Durchlaufen der Aufnahmedaten aber noch vor der Analysephase erfolgt eine Untersuchung der gesammelten Änderungen der schwarzen Ränder. Wenn beispielsweise Teile der Fernsehbilder zu dunkel sind, kann es passieren, dass nur einige Seitenränder richtig erkannt werden und es treten Unsymmetrien in den gesammelten Daten auf. Um diese zu finden, wird bei jeder Änderung der obere mit den unteren und der linke mit dem rechten Rand verglichen. Beträgt dieser Unterschied mehr als 15 Pixel, liegt eine Unsymmetrie vor. Zu deren Beseitigung wird der aktuell untersuchte Rand mit dem vorigen und dem folgenden verglichen und der mit der größten Übereinstimmung gewählt. Dessen Werte werden dem untersuchten Rand zugewiesen. Dadurch treten Doppelungen in der Liste der Randänderungen auf, die in einem abschließenden Durchlauf beseitigt werden.

In der folgenden Analyse wird aufgrund der gesammelten Daten entschieden welche Teile der Aufnahme Werbung und welche Teile normale Sendungen sind und entsprechende Schnittmarken gesetzt. Als erstes wird geprüft, ob einer der Spezialfälle vorliegt und relevant ist. Wenn die Aufnahme ein Schwarz-Weiß Programm enthält, werden die Zeitpunkte der Wechsel von Schwarz-Weiß nach Far-

be und umgekehrt zu den Schnittmarken. Dabei wird vorausgesetzt, dass die Phasen der Schwarz-Weiß Anteile auch lang genug sind, damit nicht irrtümlich schwarz-weiße Werbespots als normale Sendung erkannt werden. Falls die Aufnahme Dolby Digital Ton enthält und in diesem Änderungen auftreten, werden an diesen Stellen die Schnittmarken gesetzt. Dabei ist die Werbung der Teil mit Stereo-Ton und die restliche Sendung besitzt ein anderes Tonformat wie z.B. Dolby Surround.

Falls keiner der Spezialfälle relevant ist, erfolgt die Analyse auf Grundlage der Logo- und Randänderungen. Während dieser kann einer von drei verschiedene Zustände auftreten: Werbung, wenn das Logo längere Zeit ausbleibt, Programmvorschau, wenn das Logo nur für kurze Zeit vorhanden ist oder wenn es kurze Schwankungen in Position und Vorhandensein gibt und normale Sendung bei längerem Vorhandensein des Senderlogos. Um nun die genauen Zeitpunkte bei den Übergängen von einem Zustand zum nächsten festzustellen oder um die Zuverlässigkeit zu steigern, werden die Randänderungen benutzt. Als Beginn der normalen Sendung wird der Zeitpunkt vor dem Erscheinen des Senderlogos angenommen, an dem sich zum letzten mal der Rand geändert hat. Falls das Logo für relativ kurze Zeit verschwindet, wird überprüft, wie oft sich der Rand in dieser Zeit ändert und abhängig von der Anzahl entschieden, ob dies eine richtige Unterbrechung des Programms ist oder nicht.

Nach der Beendigung der Analysephase werden schließlich noch die Schnittmarken gesetzt, wobei etwa eine Sekunde Reserve gelassen wird. Damit ist der Algorithmus abgeschlossen.

5. Realisierung der Software

Im folgenden wird die Entwicklung der Software CommScan[1] aufgezeigt, die die Erkennung von Werbeblöcken realisiert.

5.1. Zielsetzung

Mit der Software CommScan soll zwischen Werbesendungen und normalem Fernsehprogramm unterschieden werden. Als Eingabe dienen mit VDR erstellte Aufzeichnungen im MPEG-2 Format. Durch Anwendung des in Kapitel 4.2 vorgestellten Algorithmus soll entschieden werden, welche Teile der Aufnahme zum eigentlichen Programm gehören und welche Werbung sind und deshalb herausgeschnitten werden können. Dabei liegt der Schwerpunkt auf Spielfilmen und Serien. Als Ergebnis liefert CommScan Schnittmarken in einem für VDR verständlichen Format, so dass ein Schnitt der Aufzeichnung mit VDR möglich ist, bei dem nur die Teile der Aufzeichnung erhalten bleiben, die zur eigentlichen Sendung gehören.

Die Integration der Software in VDR soll sich durch folgende Eigenschaften auszeichnen:

- Über die Menüstruktur von VDR wird eine Aufzeichnung ausgewählt und der Commercial Scan und damit das Programm CommScan gestartet.
- Die Software wird im Hintergrund ausgeführt, d.h. VDR kann währenddessen normal weiter benutzt werden.
- Während CommScan arbeitet, informiert es den Anwender durch Meldungen auf dem OSD über seinen Fortschritt.
- Ist das Programm beendet, stehen die Schnittmarken in einem für VDR verständlichen Format zur Verfügung. Diese können bei der Wiedergabe der untersuchten Aufzeichnung überprüft werden, worauf der Schnitt gestartet werden kann.

[1] kurz für Commercial Scan

5.2. Systemumgebung

Die gesamte Software ist auf einem PC der momentan mittleren Leistungsklasse, einem Pentium 4 1,5 GHz mit 512 MB RAM, realisiert worden. Zu den üblichen Hardwarekomponenten wird zusätzlich noch eine DVB-PCI-Karte benötigt, damit VDR ordnungsgemäß funktionieren kann.

Das Programm CommScan wurde unter dem Betriebssystem Linux[2] entwickelt. Linux ist ein freies Unix-System und unterliegt der GNU GPL[3] [Fre91]. Kurz zusammengefasst sichert diese Lizenz, dass ein frei zur Verfügung gestellter Quellcode auch immer und für jedermann frei bleibt. Sie erlaubt es, die Software zu ändern oder Teile davon in neuen Programmen zu verwenden, die wiederum der GNU GPL unterliegen müssen. Mit dem Medium Internet kombiniert, kann eine schnelle und globale Verbreitung der offenen Quellen und hierdurch eine nahezu synchrone Weiterentwicklung erfolgen. Das Betriebssystem Linux ist ein Musterbeispiel für diese Philosophie.

Für die entwickelte Software CommScan wurde auch dieses Lizenzmodell gewählt. Einerseits soll dies die rasche Verbreitung und Weiterentwicklung der vorhandenen Verfahren ermöglichen. Andererseits war durch die Verwendung von Quellcode aus VDR diese Lizenz vorgegeben, da VDR auch der GNU GPL unterliegt.

Als Programmiersprache wurde die Hochsprache C++ verwendet. Diese unterstützt das objektorientierte Paradigma und bringt auch einige hilfreiche Datentypen in seiner Standard Bibliothek mit. Insbesondere da auch VDR in dieser Sprache entwickelt wurde, bot sie sich an.

5.3. Programmablauf

Der grobe Programmablauf ist in Abbildung 5.1 dargestellt. Beim Start von CommScan aus VDR heraus wird die zu untersuchende Aufzeichnung als Kommandozeilenparameter übergeben. Die Aufnahmedateien werden zunächst geöffnet, dies geschieht mit Hilfe der VDR-Klassen *cFileName* und *cIndex*. Danach wird in einer Schleife vom ersten bis zum letzten Frame der Aufzeichnung jeweils ein einzelnes PES Frame eingelesen. Für jeden dieser Rahmen erfolgt ein Demultiplexen in seine Bestandteile, also Video-, Audio- und Private-Streams. Die in den Datenströmen enthaltenen einzelnen Frames werden extrahiert und teilweise

[2] genau: SuSE Linux 8.0
[3] General Public License

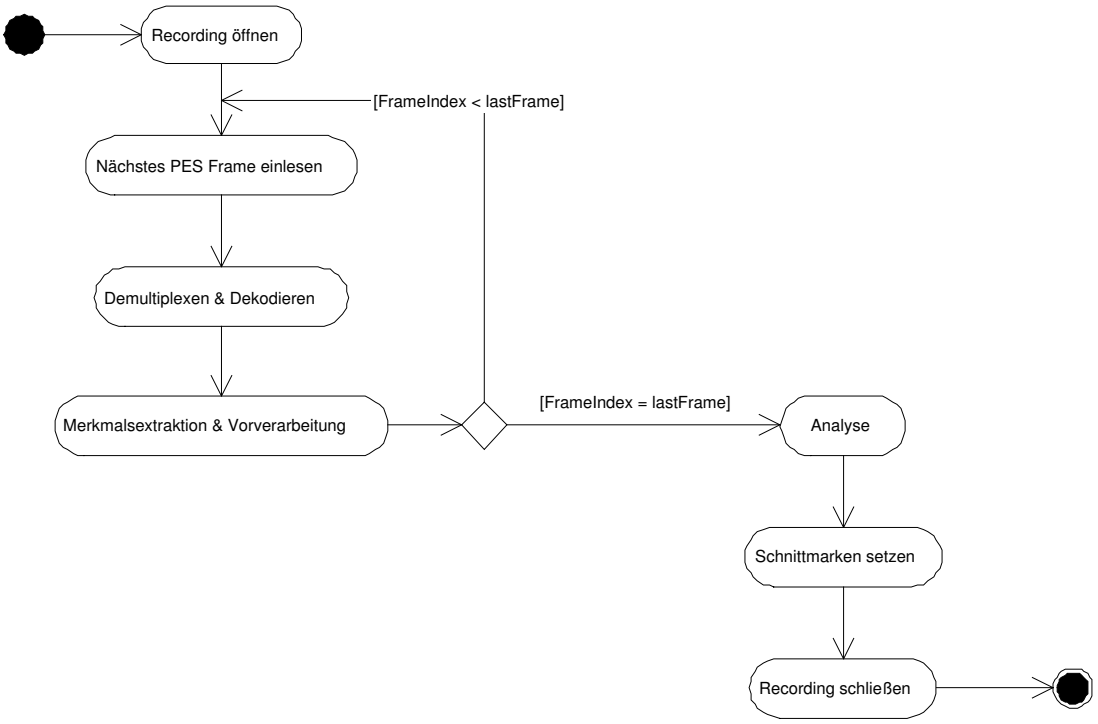


Abbildung 5.1.: Programmablauf

auch dekodiert. Ist dies abgeschlossen, erfolgt die Merkmalsextraktion mittels der Feature-Klassen. Die so ermittelten Daten durchlaufen eine Vorverarbeitung bevor sie zur späteren Verwendung abgespeichert werden. Ist das Bearbeiten aller Frames der Aufzeichnung abgeschlossen, erfolgt die Analyse der gesammelten Daten, entsprechend des in Kapitel 4.2 vorgestellten Verfahrens. Nach deren Ende werden die ermittelten Schnittmarken mit Hilfe der VDR-Klasse *cMarks* in dem von VDR geforderten Format auf die Festplatten geschrieben. Nach dem Schließen der Aufnahmedateien ist das Programm beendet. Die Marken können dann bei der Wiedergabe der Aufzeichnung mit VDR überprüft werden.

Als Rahmen für den Programmablauf wurde die Klasse *cCommercialScanner* (Abbildung 5.2) entworfen. In ihr ist der Ablauf folgendermaßen: Im Konstruktor erfolgt das Öffnen der Aufzeichnungsdateien. Die Methode *Process* beinhaltet den Hauptanteil des Algorithmus. Hier erfolgt das Einlesen der PES-Frames, das Demultiplexen und Dekodieren und die Merkmalsextraktion. Ist das abgeschlossen, wird in der Methode *Analysis* die Analyse der gesammelten Daten vorgenommen. Nach dem Schreiben der Schnittmarken mit *SetCuttingMarks* erfolgt schließlich im Destruktor der Klasse das Schließen der Aufnahmedateien.

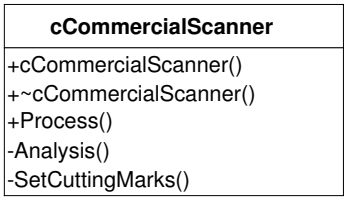


Abbildung 5.2.: Die Klasse cCommercialScanner

Fileoffset	File-number	Frame-type	reserved
4 Bytes	1 Byte	1 Byte	2 Bytes

Abbildung 5.3.: Aufbau eines Indexeintrags

5.3.1. Aufzeichnungsformat von VDR

VDR speichert die aufgenommenen Daten in einem Format, dass einen schnellen und beliebigen Zugriff darauf erlaubt. Die PES Pakete der aufgezeichneten Sendung werden fortlaufend in eine nummerierte Datei beginnend bei *001.vdr* geschrieben. Dabei werden sie zu den so genannten PES Frames zusammengefasst. Zu einem solchen Rahmen gehören Daten eines kompletten Video-Frames, also Video-, Audio- und Private-Streams. Wächst die Größe der Aufzeichnungsdatei über einen bestimmten Wert[4], wird die Nummer um eins erhöht und die Speicherung in der nächsten Datei fortgesetzt. In einer weiteren Datei namens *index.vdr* werden parallel dazu so genannte Indexeinträge gespeichert. Jeder dieser Einträge[5] enthält Informationen über das zugehörige PES Frame. Das sind die Nummer der Aufzeichnungsdatei und das Offset in dieser Datei, in der das Frame gespeichert ist, sowie dessen Typ, also ob es sich um ein I-, P-, oder B-Frame handelt. Mit diesen Informationen sind problemlos Sprünge innerhalb der Aufzeichnung möglich.

Eine Aufzeichnung kann zusätzlich noch weitere Dateien beinhalten. In *marks.vdr* werden die Schnittmarken gespeichert. Das sind Zeilen von ASCII-Einträgen im Format *Stunde*:*Minute*:*Sekunde*.*Frame*. Die Datei *resume.vdr* enthält die Position, an der die Wiedergabe der Aufnahme unterbrochen wurde, damit sie an dieser Stelle fortgesetzt werden kann. In *summary.vdr* können Informa-

[4] konfigurierbar, Standardwert ist 2000 MB.
[5] vgl. Abbildung 5.3

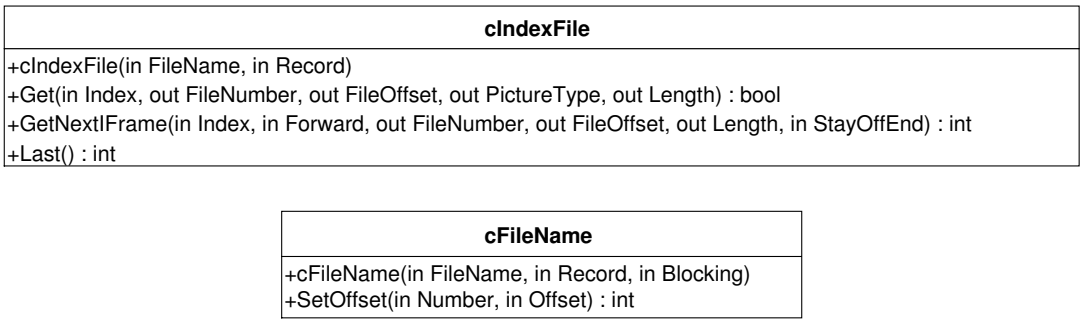


Abbildung 5.4.: VDR-Klassen für das Einlesen der Aufzeichnung und deren wichtigste Methoden

tionen über den Inhalt der Aufzeichnung gespeichert werden. Diese werden von VDR bei der Aufnahme automatisch vom EPG übernommen, soweit vorhanden.

5.3.2. Einlesen der Aufzeichnung

Für das Einlesen der Aufzeichnungen wurden die in Abbildung 5.4 gezeigten Klassen des VDR in CommScan integriert. Die Klasse *cFileName* repräsentiert die die PES Frames enthaltenen Aufzeichnungsdateien, die Indexdateien werden durch *cIndexFile* behandelt.

Das Öffnen der Aufzeichnung erfolgt mit dem Instantiieren dieser Klassen per Konstruktoraufruf, wobei der Name des Recordings in *fileName* übergeben wird, dessen Wert dem Kommandozeilenparameter von CommScan entspricht:

```
recFile = new cFileName(fileName, false);
recIndex = new cIndexFile(fileName, false);
```

Durch diese Anweisungen wird die erste Aufzeichnungsdatei geöffnet und es werden die Einträge aus der Indexdatei zum schnelleren Zugriff in den Speicher geladen.

Für das Einlesen der einzelnen PES-Frames werden die Indexeinträge vom ersten bis zum letzten in einer Schleife durchlaufen:

```
for (index = 0; index < recIndex->Last(); index++)
{
    if (recIndex->Get(index, &fileNumber, &fileOffset,
                      &pictureType, &length))
    {
        file = recFile->SetOffset(fileNumber, fileOffset);
```

```
        if (file != -1)
        {
            length = ReadFrame(file, frameBuffer,
                               length, 256 * 1024);
        }
    }
}
```

Die Methode *Get* der Klasse *cIndexFile* liefert die Daten des aktuellen Indexeintrags und damit Position, Typ und Länge des nächsten PES-Frames. Mit der Funktion *SetOffset* von *cFileName* wird der Dateizeiger an diese Position verschoben und zurückgeliefert. Dabei muss unter Umständen ein Wechsel der Aufzeichnungsdatei erfolgen, z.B. von *001.vdr* nach *002.vdr*. Von dieser Stelle wird mit der VDR-Funktion *ReadFrame* das komplette PES-Frame in den Speicherbereich, der in *frameBuffer* übergeben wird, eingelesen.

Vor dem Beenden von CommScan erfolgt das Schließen der Aufzeichnung. Dazu müssen die Objekte mittels delete zerstört werden:

```
delete recFile;
delete recIndex;
```

In den dabei erfolgenden Destruktoraufrufen werden die entsprechenden Dateien geschlossen und der Speicherbereich für die Indexeinträge wieder freigegeben.

5.3.3. Demultiplexen & Dekodieren

Für das Demultiplexen und Dekodieren der PES-Frames wurden die Klassen in Abbildung 5.5 entworfen und implementiert. Dabei wurde der genaue Aufbau dieser Datenpakete dem ISO/IEC MPEG-2 Standard [MPE] entnommen und berücksichtigt. Der Vorgang des Demultiplexen und Dekodierens mit Hilfe dieser Klassen wird im Folgenden beschrieben.

Das Demultiplexen beginnt mit dem Instantiieren der Klasse *cPESFrame*. wobei die Daten des zuvor eingelesenen PES-Frames mit dem Konstruktoraufruf übergeben werden. Das Paket wird analysiert, dabei werden die einzelnen Audio-, Video- und Private-Streams, soweit vorhanden, erkannt, extrahiert und in Form von Objekten der Klasse *cMPEGStream* gespeichert. Im Konstruktor dieser Klasse erfolgt, wenn es sich um einen Audio-Stream oder einen Private-Stream mit Dolby Digital Ton handelt, die Identifikation und Zählung der einzelnen Audio-Frames. Sollten dabei Überlappungen von einem PES-Frame zum Nächsten auftreten, werden diese erkannt und entsprechend behandelt.

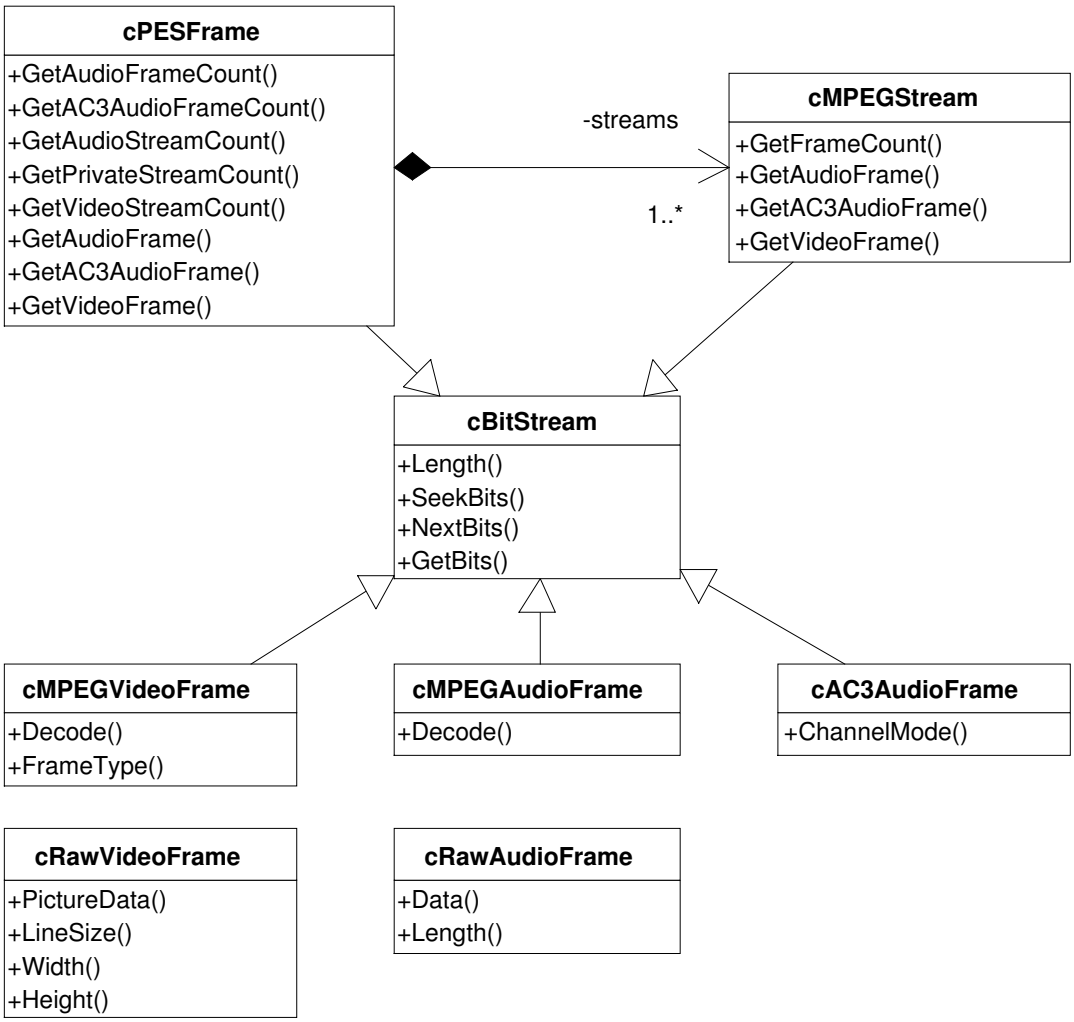


Abbildung 5.5.: Klassen für das Demultiplexen und Dekodieren und deren wichtigste Methoden

Die Anzahl der MPEG-Datenströme eines PES-Frames können über die *Get...StreamCount* Methoden von *cPESFrame* abgefragt werden, genau wie die Anzahl der entsprechenden Frames mittels *Get...FrameCount*. Diese werden gebraucht, damit die gewünschten Video-, Audio- und AC3-Frames aus den Streams extrahiert werden können, was mit den Methoden *Get...Frame* geschieht. Diese liefern zu gegebener Stream- und Frame-Nummer Objekte der Klassen *cMPEGVideoFrame*, *cMPEGAudioFrame* und *cAC3AudioFrame*, welche die gewünschten Daten beinhalten. Dabei erfolgt noch die Analyse der in allen drei Arten von Paketen vorhandenen Headerdaten.

Alle fünf bisher vorgestellten Klassen sind Subklassen von *cBitStream*. Dies erlaubt es die Daten der Ströme bitweise zu durchlaufen und auszulesen, was

das Parsen der Header vereinfacht.

Das Dekodieren der Video- und Audio-Frames erfolgt über die Methode *Decode* der Klassen *cMPEGVideoFrame* und *cMPEGAudioFrame*. Diese liefern die unkomprimierten Einzelbilder und Tonstücke in Form von Objekten der Klassen *cRawVideoFrame* und *cRawAudioFrame*. Dabei werden nur die Bilder dekodiert, die als I-Frame vorliegen, also unabhängig von anderen sind.

Dekodieren mit der libavcodec

Für das eigentliche Dekodieren der Daten werden Funktionen der Bibliothek *libavcodec* genutzt. Diese ist Bestandteil des „FFmpeg video and audio converter“, welcher in Version 0.4.6 verwendet wurde und eine Reihe von Audio- und Video-Decodern für die verschiedensten Formate zur Verfügung stellt. Die Verwendung dieser Decoder ist im folgenden Programmbeispiel für ein Video-Frame aufgezeigt. Dabei sind dessen Daten mit *data* und *length* gegeben.

```
AVCodec *codec;
AVCodecContext *c = NULL;
int got_picture;
AVFrame *picture;

// MPEG video decoder finden
codec = avcodec_find_decoder(CODEC_ID_MPEG1VIDEO);
// Datenstrukturen initialisieren
c = avcodec_alloc_context();
picture = avcodec_alloc_frame();
// MPEG Video Decoder oeffnen
avcodec_open(c, codec);
// Video-Frame dekodieren
avcodec_decode_video(c, picture, &got_picture,
                     data, length);
if (got_picture)
{
    // Video-Frame wurde erfolgreich dekodiert.
    ...
}
// MPEG Video Decoder schliessen
avcodec_close(c);
// Datenstrukturen freigeben
```

```
free(c);
```

Als erstes wird der gewünschte Video-Codec für MPEG-Video gesucht. Nach dem Initialisieren weiterer benötigter Datenstrukturen wird dieser geöffnet. Daraufhin werden dem Codec die Daten des Video-Frames übergeben, woraufhin die Dekodierung erfolgt. Ist diese erfolgreich steht das unkomprimierte Einzelbild in einer Struktur vom Typ *AVFrame* zur Verfügung.

5.3.4. Merkmalsextraktion

In der Phase der Merkmalsextraktion erfolgt die Untersuchung der in Kapitel 4.1 vorgestellten Erkennungsmerkmale. Dabei werden Veränderungen festgestellt und mit ihrem Zeitpunkt in Listen zwischengespeichert, wovor noch eine Vorverarbeitung der Daten stattfindet.

Die Merkmalsextraktion erfolgt während des Demultiplexen und Dekodierens. Sobald ein bestimmter Typ von Daten vorliegt, erfolgt die Untersuchung der Merkmale die für diesen Typ zutreffend sind. Dabei sind 6 verschiedene Typen möglich:

- **PES-Frame**, hierfür werden noch keine Merkmale berücksichtigt, die Erkennung von Mehrkanalton wäre möglich
- **MPEG Video-Frame**, hierfür werden noch keine Merkmale berücksichtigt, die Erkennung von PAL-Plus Sendungen wäre möglich
- **dekomprimiertes Video-Frame**, dies ist Basis für die Logo-, schwarze Rand- und Schwarz-Weiß-Erkennung
- **MPEG Audio-Frame**, hierfür werden noch keine Merkmale berücksichtigt, die Erkennung verschiedener Tonformate wäre möglich
- **dekomprimiertes Audio-Frame**, darauf basiert die nur teilweise realisierte Mono/Stereo-Erkennung
- **AC3 Audio-Frame**, dies ist Grundlage für die Erkennung des Dolby Digital Tonformats

Die Untersuchung der Merkmale der verschiedenen Typen erfolgt in den Methoden *Check...Features* der Klasse *cCommercialScanner*[6]. Sollte sich dabei ein Merkmal im Vergleich zur letzten Betrachtung geändert haben, werden dessen

[6] vgl. Abbildung 5.6

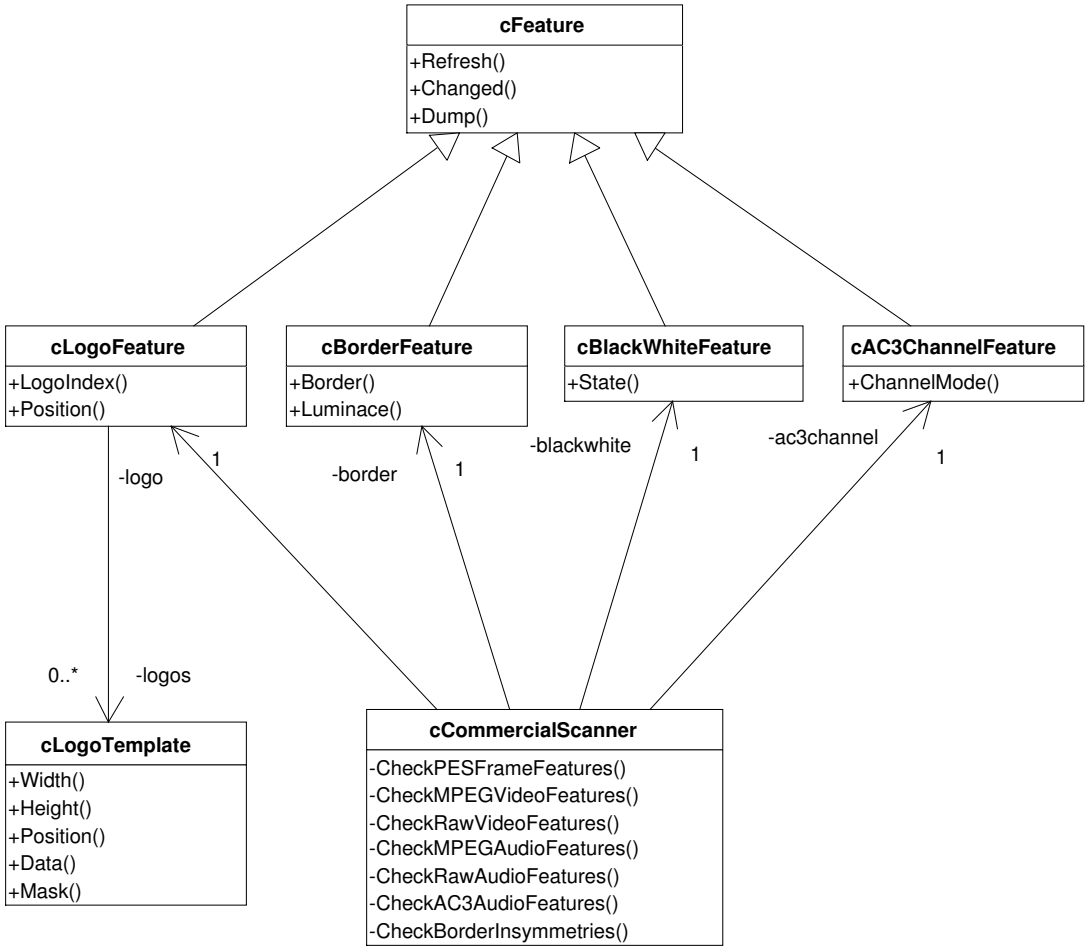


Abbildung 5.6.: Klassen für die Merkmalsextraktion und deren wichtigste Methoden

Zeitpunkt und Daten, nachdem sie die Vorverarbeitung erfolgreich durchlaufen haben, in die entsprechende Liste zur Zwischenspeicherung aufgenommen.

In der Vorverarbeitung wird als erstes überprüft, wie viel Zeit seit der letzten Veränderung vergangen ist. Liegt dieser Wert unter einer bestimmten Schwelle[7], werden die neu ermittelten Eigenschaften des betrachteten Merkmals mit denen der vorletzten Änderung verglichen. Sollten diese übereinstimmen, handelt es sich nur um eine kurzzeitige Schwankung, die ignoriert wird, indem die aktuelle Änderung nicht gespeichert und die letzte aus der Liste gelöscht wird. Ansonsten haben die neuen Merkmalseigenschaften die Vorverarbeitung passiert.

Zur Durchführung der Merkmalsextraktion wurden die Klassen in Abbildung 5.6 entworfen. Für jedes zu untersuchende Merkmal entstand eine der *c...Feature*-

[7] Der Schwellwert ist je nach Merkmal unterschiedlich, liegt aber immer im Bereich weniger Sekunden.

Klassen:

- *cLogoFeature*, für die Logoerkennung. Das aktuelle Senderlogo und dessen Position können über *LogoIndex* und *Position* abgefragt werden. Die zu berücksichtigen Senderlogos werden lokal in einem Array von Objekten der Klasse *cLogoTemplate* aufbewahrt. Über deren Methoden stehen die Attribute der Logos, also Breite, Höhe, Schablone, Maske und mögliche Positionen, dem Erkennungsalgorithmus zur Verfügung.

- *cBorderFeature*, für die Erkennung der schwarzen Ränder. Der aktuelle Rand und die Helligkeit des letzten Bildes können mittels *Border* und *Luminance* abgefragt werden.

- *cBlackWhiteFeature*, für die Erkennung von Schwarz-Weiß-Sendungen. Der aktuelle Zustand kann über *State* ermittelt werden.

- *cAC3ChannelFeature*, für die Erkennung unterschiedlicher Dolby Digital Kanalmodi. Der aktuelle Modus kann durch *ChannelMode* erfragt werden.

Da diese in ihrem grundlegendem Aufbau sehr ähnlich waren, wurden zur Vermeidung von Redundanzen die gemeinsamen Attribute und Methoden in der Basisklasse *cFeature* zusammengefasst. Mit der Methode *Refresh*, die in den Subklassen überschrieben werden muss, erfolgt die Untersuchung des jeweiligen Merkmals. Sollte dabei eine Veränderung auftreten, wird der Zustand der Objekte entsprechend gesetzt, welcher dann mit der Methode *Changed* abgefragt werden kann.

5.3.5. Analyse

Nachdem die Aufzeichnung komplett Frame für Frame verarbeitet wurde und dabei alle Merkmalsdaten extrahiert und gespeichert wurden, erfolgt vor der Analyse noch eine Vorverarbeitung der gesammelten Randänderungen. Dies geschieht über die Methode *CheckBorderInsymmetries* der Klasse *cCommercialScanner*[8]. Diese ermittelt Unsymmetrien in den erkannten Randbreiten nach den in Kapitel 4.2 beschriebenen Kriterien und versucht diese zu beseitigen.

Die eigentliche Analyse zur Ermittlung der Übergänge zwischen den verschiedenen Zuständen Werbung, Vorschau und Programm erfolgt nach Abschluss der

[8] vgl. Abbildung 5.6

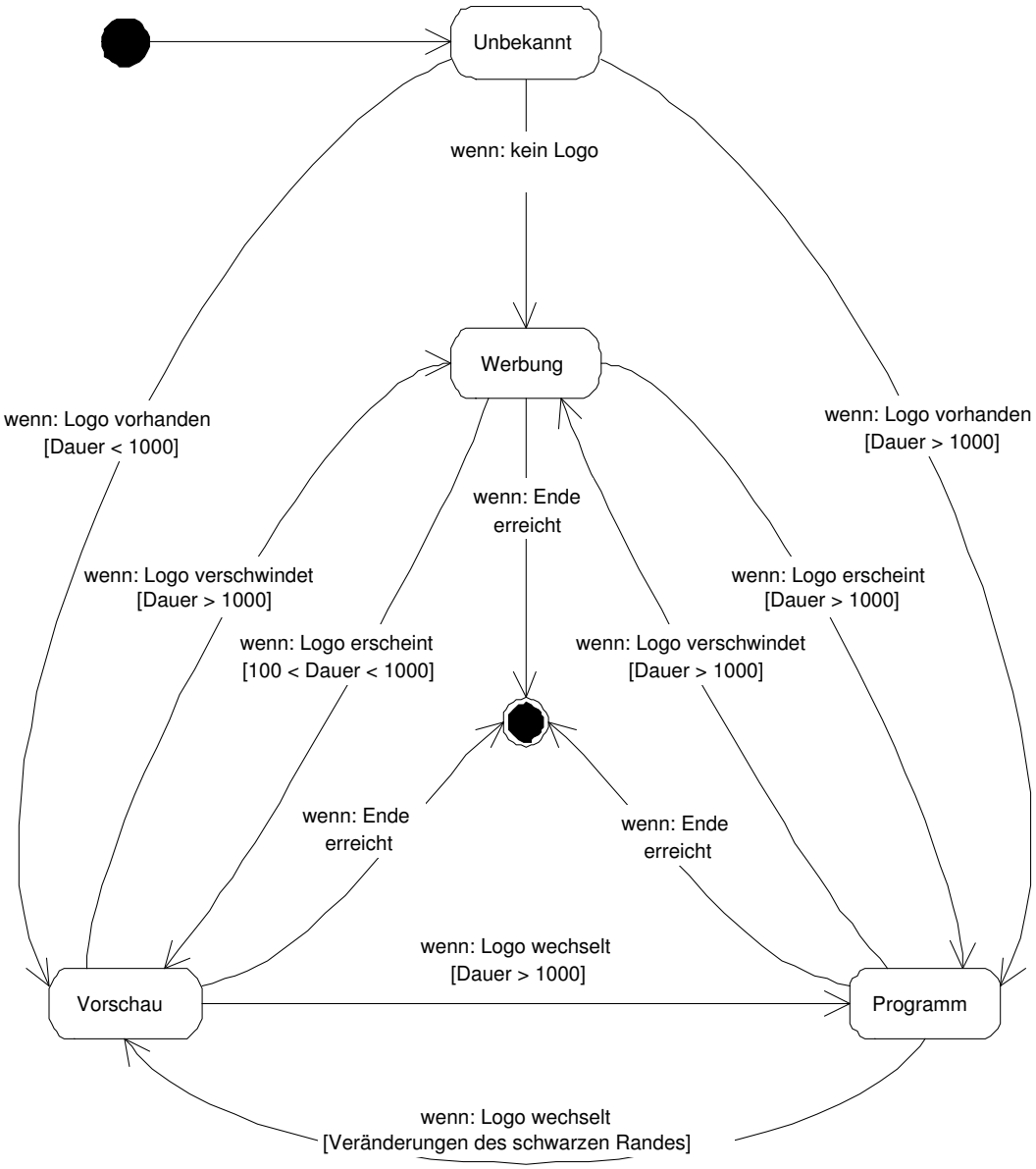


Abbildung 5.7.: Mögliche Zustände und wesentliche Übergänge bei der Analyse ohne Berücksichtigung von Spezialfällen

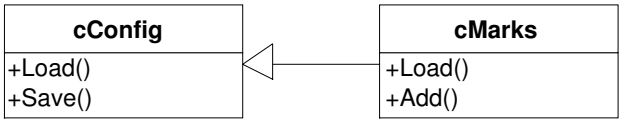


Abbildung 5.8.: VDR-Klassen für das Setzen der Schnittmarken und deren wichtigste Methoden

Vorarbeiten in der Methode *Analysis*[9]. Dies läuft nach dem in Kapitel 4.2 beschriebenen Verfahren. Als erstes wird geprüft, ob einer der Spezialfälle Dolby Digital Ton oder Schwarz-Weiß vorliegt und dementsprechend die Zeiten der Übergänge zwischengespeichert. Sollte dies nicht der Fall sein, erfolgt die Analyse anhand der Logo- und Randänderungen. Die wesentlichen Zustandswechsel sind in Abbildung 5.7 dargestellt, wobei noch einige Spezialfälle berücksichtigt werden. Die erkannten Zustandswechsel werden mit ihrem Zeitindex in einer Liste zur späteren Verwendung zwischengespeichert.

5.3.6. Setzen der Schnittmarken

Aus der Liste der Zustandsübergänge werden in der Methode *SetCuttingMarks* der Klasse *cCommercialScanner* die Schnittmarken generiert. Übergänge vom Programm zur Werbung oder Vorschau und umgekehrt werden als Beginn und Ende der Werbung interpretiert und deren Zeitindizes als Schnittmarken genommen.

Das Schreiben der Marken in dem für VDR verständlichen Format erfolgt mit der Klasse *cMarks*[10], die aus VDR übernommen wurde. Die Methode *Load* öffnet die Datei *marks.vdr* im Verzeichnis der untersuchten Aufzeichnung. Sie wird automatisch erstellt, falls sie noch nicht existieren sollte. Mittels *Add* werden der Datei die ermittelten Schnittmarken hinzugefügt, wobei zur Sicherheit je etwa 1 Sekunde[11] Reserve gelassen wird. Mit der Methode *Save* wird die Datei schließlich gespeichert und geschlossen. Damit ist der Algorithmus abgeschlossen und nach dem Schließen der Aufnahmedateien wird CommScan beendet.

[9] vgl. Abbildung 5.2

[10] vgl. Abbildung 5.8

[11] Genau genommen sind dies 2 GOPs, also in der Regel 24 Frames, was einer knappen Sekunde entspricht.

Abbildung 5.9.: Aufzeichnungsmenü von VDR mit der roten Schaltfläche „Befehle“

5.4. Integration in VDR

VDR bietet die Möglichkeit beliebige Kommandos über das Menü zu starten. Diese müssen in der Konfigurationsdatei *commands.conf* eingetragen werden. Sie werden beim nächsten Start von VDR automatisch in dessen Menüstruktur integriert und stehen unter dem Menüpunkt „Befehle“ zur Verfügung. Eine weitere Möglichkeit eigene Kommandos aus VDR heraus auszuführen sind die Recording-Commands, die Befehle darstellen, die auf einzelne Aufzeichnungen angewendet werden können. Sie werden in der Datei *reccmds.conf* definiert. Ein Eintrag in dieser Datei besteht aus einem Namen gefolgt von einem Doppelpunkt und dem auszuführenden Befehl. Die angegebenen Kommandos stehen dann für jede Aufnahme im Aufzeichnungsmenü unter der roten Taste zur Verfügung, wie in Abbildung 5.9 zu sehen. Beim Aufruf eines solchen Befehls wird automatisch der Dateiname der aktuell gewählten Aufzeichnung als Kommandozeilenparameter übergeben.

Um CommScan in VDR zu integrieren wurde die Möglichkeit der Recording-Commands gewählt. Da VDR bei der Ausführung eines Kommandos normaler-

weise auf dessen Beendigung wartet und dadurch für weitere Benutzung blockiert ist, musste eine Möglichkeit gefunden werden, CommScan im Hintergrund zu starten. Dafür bot sich der Dienst *at* an, der zur Grundausstattung von nahezu jedem Linux-System gehört. Mit der Befehlszeile

```
at now -f <Befehl>
```

wird der angegebene Befehl in die Warteschlange von *at* eingereiht und bei nächster Gelegenheit im Hintergrund ausgeführt. Mit dem Start von CommScan im Hintergrund wäre es nun möglich gewesen, dieses mehrmals parallel auszuführen. Deshalb muss vor dem Start noch geprüft werden, ob es schon läuft und gegebenenfalls mit Fehlermeldung abgebrochen werden.

Zur Verwirklichung der genannten Forderungen wurde das folgende Shell-Script commscan.sh geschrieben:

```
01 #!/bin/bash
02
03 RUNNING="‘ps -A | grep -w ’commscan’ | wc -l‘"
04
05 if [ $RUNNING -gt 2 ] ; then
06 echo "Commercial Scan is already running!"
07 exit 0
08 fi
09
10 echo "/usr/local/bin/commscan $1" > \
/usr/local/bin/commscan_tmp.sh
11 chmod +x /usr/local/bin/commscan_tmp.sh
12 at now -f /usr/local/bin/commscan_tmp.sh
13 echo "Commercial Scan started"
```

Zeile 1 bestimmt, dass zur Ausführung dieses Scripts die Shell *bash* benötigt wird. In den Zeilen 3 bis 8 erfolgt die Prüfung ob CommScan schon ausgeführt wird. Es wird die Anzahl der Prozesse mit „commscan“ im Namen[12] ermittelt. Dafür werden die Linux-Tools *ps*, *grep* und *wc* benutzt. Mit *ps* wird eine Liste aller momentan laufenden Prozesse erzeugt. Aus dieser werden mit *grep* die Einträge ausgefiltert, die den Text „commscan“ enthalten. Das Programm *wc* ermittelt schließlich die Anzahl der Zeilen und schreibt diese in die Variable RUNNING. Falls diese Zahl größer als 2 ist, dann wird CommScan bereits ausgeführt und

[12] Dies können die Prozesse *commscan* und *commscan.sh* sein.

Abbildung 5.10.: Befehlsmenü einer Aufzeichnung

nach Ausgabe einer Fehlermeldung, welche dann im OSD von VDR erscheint, wird das Scripts beendet. Ansonsten erfolgt der Start des Programms. Dafür wird in Zeile 10 ein neues Shell-Script erzeugt, bestehend aus dem Befehl zum Starten von CommScan und dem in *$1* enthaltenen Dateinamen der zu untersuchenden Aufzeichnung. Dieses Script wird in Zeile 11 ausführbar gemacht, bevor es an den Dienst *at* zur bald möglichsten Ausführung übergeben wird (Zeile 12). Schließlich erfolgt in Zeile 13 noch die Ausgabe einer Erfolgsmeldung, welche dann im OSD von VDR angezeigt wird.

Bevor CommScan von VDR aus gestartet werden kann, muss der Datei reccmds.conf der folgende Eintrag hinzugefügt werden:

```
Commercial Scan?:/usr/local/bin/commscan.sh
```

Dieser besteht aus dem Titel, der im Befehlsmenü erscheinen soll und der Befehlszeile. Das Fragezeichen bewirkt, dass vor dem Start noch eine Abfrage erfolgt, die mit der Taste <OK> bestätigt werden muss. Das Kommando steht nach dem nächsten Start von VDR im Befehlsmenü der Aufzeichnungen zur Verfügung, wie in Abbildung 5.10 gezeigt.

Abbildung 5.11.: Fortschrittsanzeige während der Ausführung von CommScan

Während CommScan im Hintergrund ausgeführt wird, kann VDR ganz normal weiterverwendet werden. Dabei sollte CommScan den Benutzer über seinen Fortschritt informieren. Um dies zu realisieren wurde das Simple VDR Protocol (SVDRP) verwendet. Dies ist ein Protokoll, welches die Steuerung von VDR über ein Netzwerk ermöglicht. Unter anderem können mit dem Kommando „MESG“ beliebige Textnachrichten an VDR gesendet werden, deren Darstellung im OSD erfolgt. Dafür läuft in VDR ein Server auf Port 2001, der entsprechende Kommandos entgegen nimmt und verarbeitet.

Zum Senden einer Nachricht an VDR, die im OSD dargestellt werden soll, sind folgende Schritte nötig:

1. Verbindungsaufbau zu IP 127.0.0.1, Port 2001
2. Senden der gewünschten Meldung mit „MESG <Meldung>“
3. Verbindungsabbau

Die so gesendete Nachricht wird im OSD angezeigt, sofern dieses nicht gerade in Gebrauch ist. CommScan nutzt diese Möglichkeit für eine Fortschrittsanzei-

ge, wie sie in Abbildung 5.11 zu sehen ist. Außerdem wird bei Beendigung der Software die Meldung „Commercial Scan finished“ ausgegeben.

6. Ergebnisse

Das grundlegende Ziel bestand darin das Herausschneiden der Werbung aus aufgezeichneten Fernsehsendungen zu automatisieren. Die Konzentration lag dabei auf Spielfilmen und Serien, die wohl den häufigsten Anwendungsfall der Aufzeichnung und Archivierung darstellen. Dafür wurden Verfahren vorgestellt und implementiert, die die Differenzierung zwischen Werbung und normaler Sendung ermöglichen. Einige davon stellen Spezialfälle dar, andere sind allgemein anwendbare Erkennungsmethoden.

Die entstandene Applikation CommScan enthält die erforderlichen Daten (Logos und deren Positionen) um die Aufzeichnungen von den drei Fernsehsendern Pro 7, Kabel 1 und RTL zu untersuchen. Allerdings ist es relativ einfach Daten weiterer Sender zu integrieren. Dazu muss die Schablone und Helligkeitsmaske sowie die möglichen Positionen des Logos angegeben werden.

Die Güte der implementierten Algorithmen wurde anhand einer Reihe verschiedener aufgezeichneter Programme der drei Sender überprüft. Dabei wurde festgestellt, dass das System im Allgemeinen sehr genaue Resultate liefert, insbesondere wenn einer der Spezialfälle Schwarz/Weiß oder Dolby Digital Ton zur Anwendung kommt. Diese treten zwar relativ selten auf, liefern aber in der Regel hundertprozentig genaue Ergebnisse. Vor allem letzteres Merkmal wird wohl in Zukunft häufiger Auftreten, auch auf anderen Sendern[1].

6.1. Probleme und mögliche Lösungen

Wenn keiner der Spezialfälle relevant ist, werden die allgemeinen Erkennungsmerkmale Senderlogo und schwarzer Rand verwendet um die Werbeblöcke zu ermitteln. Dabei können unter Umständen Probleme auftreten, auf die im Folgenden eingegangen wird.

[1] Etwa zum Zeitpunkt der Fertigstellung dieser Arbeit hat z.B. der Sender SAT 1 damit begonnen, einige Filme in diesem Tonformat auszustrahlen.

6.1.1. Allgemeine Probleme

Bei der Untersuchung von Aufnahmen des Senders RTL wird häufig die Programmvorschau zur normalen Sendung hinzu gezählt und nicht wie die Werbung zum Entfernen markiert. Das liegt daran, dass das Senderlogo während Vorschau und nachfolgender Sendung meist ohne Unterbrechung und immer an derselben Position im Bild vorhanden ist. Zur Lösung dieses Problems müssten die Änderungen des schwarzen Randes berücksichtigt werden, in der Art, dass die Sendung erst ab der Stelle als begonnen angesehen wird, ab der der Rand konstant bleibt.

6.1.2. Logoerkennung

Die Logoerkennung ist relativ einfach gestrickt. Deshalb kann es vorkommen, dass ein Logo erkannt wird, obwohl keines vorhanden ist, beispielsweise bei eng beieinander liegenden Kanten im Bild. Grund dafür ist unter anderem die Art der Kantendetektion. Der verwendete Sobel-Operator erzeugt sehr dicke Linien im Kantenbild. Beim Template Matching können diese zu einem starken Peak im Ergebnisbild und damit zu einer falschen Logodetektion führen. Mit den Überprüfungen auf der Helligkeitsmaske des Logos wird versucht dem entgegen zu wirken. Es können aber trotzdem noch Fälle fälschlicherweise erkannter Logos auftreten.

Eine Möglichkeit dies zu vermeiden ist das Verwenden eines besseren Kantendetektionsalgorithmus. Ein Beispiel eines erweiterten Verfahrens ist das von Boie-Cox [BC87]. Es bietet eine genauere Erkennung, da es auch diagonal verlaufende Kanten berücksichtigt. Ein weiterer Vorteil ist die Erzeugung deutlich dünnerer Kanten als beim Sobel-Operator. Damit können auch Senderlogos mit feinerer Struktur gut detektiert werden. In ersten Tests an Einzelbildern hat sich gezeigt, das die Erkennung mit Boie-Cox deutlich genauer arbeitet, vor allem wenn sich das Logo nur relativ schwach vom Fernsehbild abhebt. Das Verfahren hat nur einen Nachteil: Es ist sehr rechenaufwändig. Um es sinnvoll in CommScan integrieren zu können, sollte die Kantendetektion in der Logoerkennung nicht mehr auf einem Großteil des Bildes sondern nur noch lokal an möglichen Logopositionen erfolgen.

Ein weiteres unter Umständen auftretendes Problem der Logoerkennung ist die Detektion eines völlig falschen Logos eines anderen Senders. Dies führt dazu, dass im Folgenden nur noch Logos dieses Senders berücksichtigt werden, was natürlich nicht besonders vorteilhaft ist. Mit der oben vorgestellten Verbesserung

mittels anderer Kantendetektion kann man dem entgegenwirken. Eine Möglichkeit die falsche Erkennung zu verhindern ist die im folgenden Kapitel kurz erläuterte Merkmalsextraktion während der Aufzeichnung eines Programms, da damit der Sender feststeht und nur dessen Logos berücksichtigt werden müssen. Außerdem bringt dies eine Beschleunigung der Erkennung mit sich, insbesondere wenn deutlich mehr Sender und deren Logos zu betrachten sind.

6.1.3. Randermittlung

Die Ermittlung der schwarzen Ränder des Fernsehbildes ist schwierig bis unmöglich, wenn das untersuchte Bild zu dunkel ist. Im Moment werden diese Fälle durch CommScan nicht berücksichtigt. Dies führt zu einer gesteigerten Ungenauigkeit des Gesamtalgorithmus. Wenn beispielsweise ein Film mit einer dunklen Szene beginnt und das Logo erst verspätet eingeblendet wird, dann wird der Beginn unter Umständen nicht genau erkannt. Besser wäre es sich die Stellen zu merken, an denen das Bild zu dunkel für die Randerkennung war und dies bei der Analyse zu berücksichtigen. Wenn man zusätzlich noch den Helligkeitsschwellwert für die Randermittlung erhöht, führt dies zu einer weiteren Verbesserung, da die schwarzen Ränder bei einem helleren Bild leichter erkannt werden.

7. Ausblick

Die vorliegende Studienarbeit sollte als Grundlage für weiterführende Arbeiten dienen. Es wurden verschiedene Methoden entwickelt und vorgestellt, Werbung vom laufenden Programm zu unterscheiden. Diese arbeiten meist sehr zuverlässig, einige bieten sogar hundertprozentige Korrektheit, soweit sie auch genutzt werden können. Andere können noch verbessert werden, unter anderem mit den im vorigen Kapitel erläuterten Möglichkeiten.

Im Vordergrund für die weitere Entwicklung sollte zunächst die Integration von CommScan in die Software VDR in Form eines Plugins dienen, was einige Vorteile mit sich bringt:

- Die genutzten Klassen des VDR müssen nicht mehr im Programmcode von CommScan enthalten sein, da diese allen Plugins automatisch zur Verfügung stehen. Damit werden Änderungen in den Datenstrukturen und damit in der Implementierung der Klassen automatisch berücksichtigt.

- Das OSD kann direkt für Statusausgaben, wie die Fortschrittsanzeige, verwendet werden, es muss nicht mehr das SVDRP genutzt werden. Damit sind nicht nur einfache Textmeldungen möglich, sondern Ausgaben beliebiger Art, da das OSD pixelweise angesteuert werden kann.

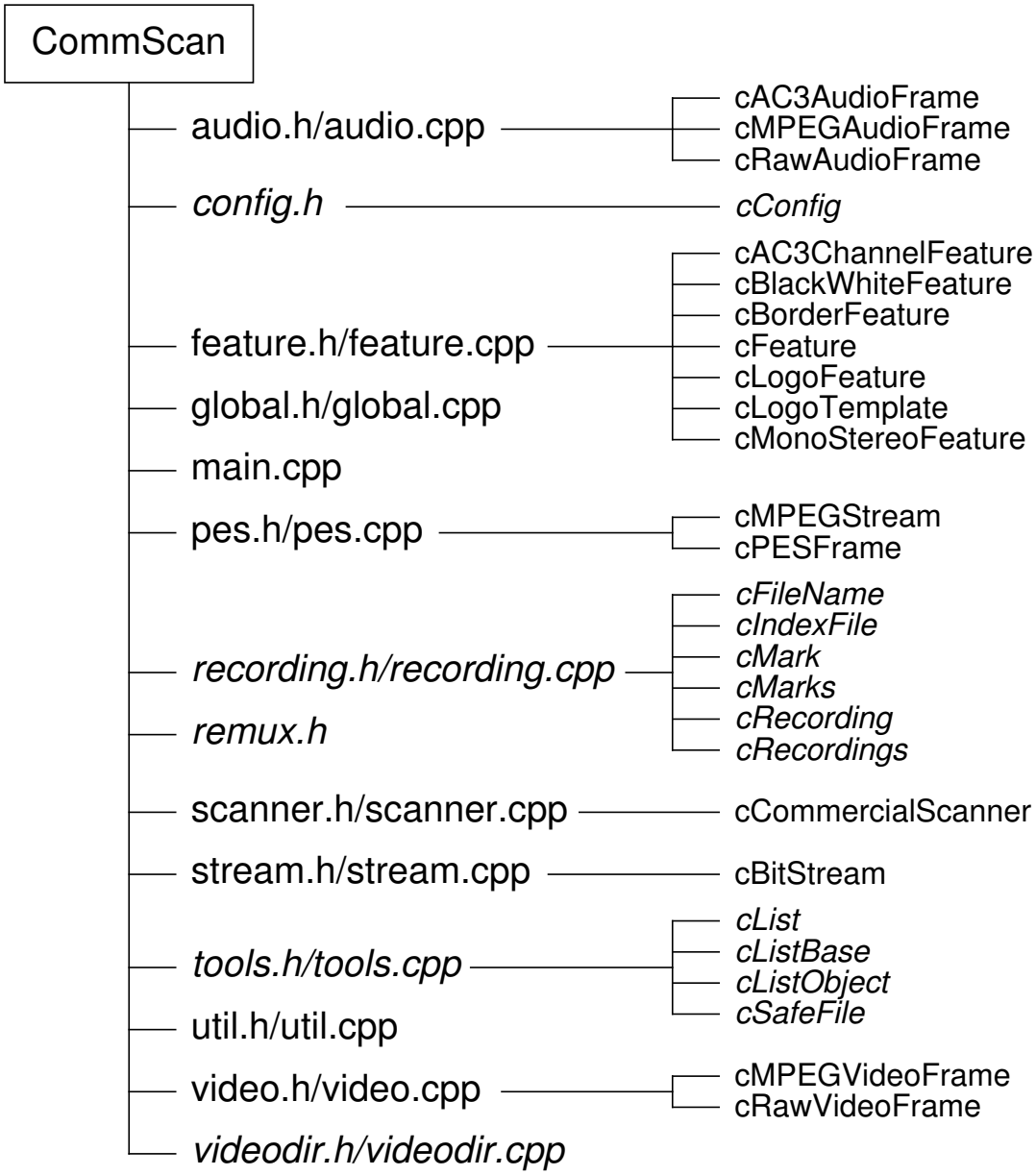
- Einstellungsmöglichkeiten diverser CommScan-Parameter können in die Setup-Menüs von VDR integriert werden, was den Test verschiedener Parameter, wie Schwellwerte, erleichtert, da keine Neukompilation nötig ist.

Ein nächster möglicher Schritt in der Integration wäre es, die Sammlung der Merkmalsdaten bereits während der Aufzeichnung einer Sendung durchzuführen. Damit stehen die Schnittmarken direkt nach Beendigung der Aufnahme zur Verfügung und die Wartezeit bei einem gesonderten Start von CommScan entfällt. Außerdem wird die Logoerkennung vereinfacht und beschleunigt, da feststeht, welcher Sender aufgezeichnet wird und damit nur noch dessen Logos im Bild gesucht werden müssen.

Wie man sieht ist das Projekt „Werbung im Fernsehen erkennen“ mit der vorliegenden Arbeit noch nicht abgeschlossen. Es bietet noch viele Möglichkeiten für weitere darauf aufbauende Arbeiten.

A. Programmstruktur

Im folgenden ist der Aufbau der entwickelten Software CommScan angegeben. Dabei sind aus VDR übernommene Dateien bzw. Klassen kursiv dargestellt.



B. Softwareinstallation

Systemvoraussetzungen

Damit CommScan ausgeführt werden kann, müssen folgende Voraussetzungen im System gegeben sein:

- Betriebssystem Linux auf einem Standard PC
- vollständig eingerichtete und funktionstüchtige Software VDR

Benötigte Pakete

Zum Installieren der Software werden die Pakete

- „FFmpeg video and audio converter“ Version 0.4.6[1] vorliegend in der Datei `ffmpeg-0.4.6.tar.gz` und
- „CommScan“ vorliegend in der Datei `commscan.tgz`

benötigt

Installationsschritte

1. Die Dateien `ffmpeg-0.4.6.tar.gz` und `commscan.tgz` in ein beliebiges[2] Verzeichnis kopieren und in dieses Verzeichnis wechseln.

   ```
   cp ffmpeg-0.4.6.tar.gz /usr/local/src
   cp commscan.tgz /usr/local/src
   cd /usr/local/src
   ```

2. FFmpeg und CommScan mit *tar* entpacken.

[1] FFmpeg kann von http://ffmpeg.sf.net bezogen werden.
[2] Im folgenden wird `/usr/local/src` verwendet.

```
tar -xzf ffmpeg-0.4.6.tar.gz
tar -xzf commscan.tgz
```

3. Ins Verzeichnis von FFmpeg wechseln und FFmpeg compilieren.

```
cd ffmpeg-0.4.6
./configure
make
```

4. Die Header-Dateien der libavcodec installieren.

```
mkdir /usr/local/include/ffmpeg
cp libavcodec/avcodec.h /usr/local/include/ffmpeg
cp libavcodec/common.h /usr/local/include/ffmpeg
mkdir /usr/local/include/ffmpeg/i386
cp libavcodec/i386/mmx.h /usr/local/include/ffmpeg/i386
```

5. Die statische Bibliothek libavcodec.a ins Verzeichnis von CommScan kopieren.

```
cp libavcodec/libavcodec.a /usr/local/src/commscan
```

6. Ins Verzeichnis von CommScan wechseln, CommScan compilieren und installieren.

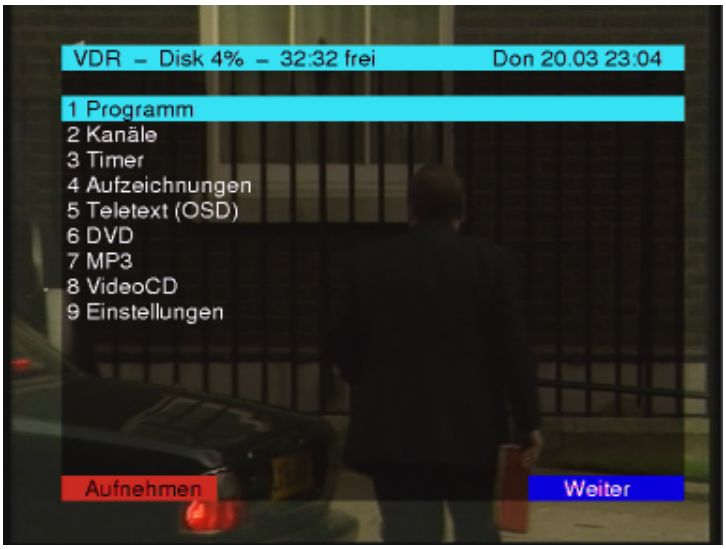
```
cd /usr/local/src/commscan
make
make install
```

7. Inhalt der Datei reccmds.conf.sample an die Datei reccmds.conf im Config-Verzeichnis von VDR anhängen. Gegebenenfalls diese Datei erstellen.

C. Bedienungsanleitung

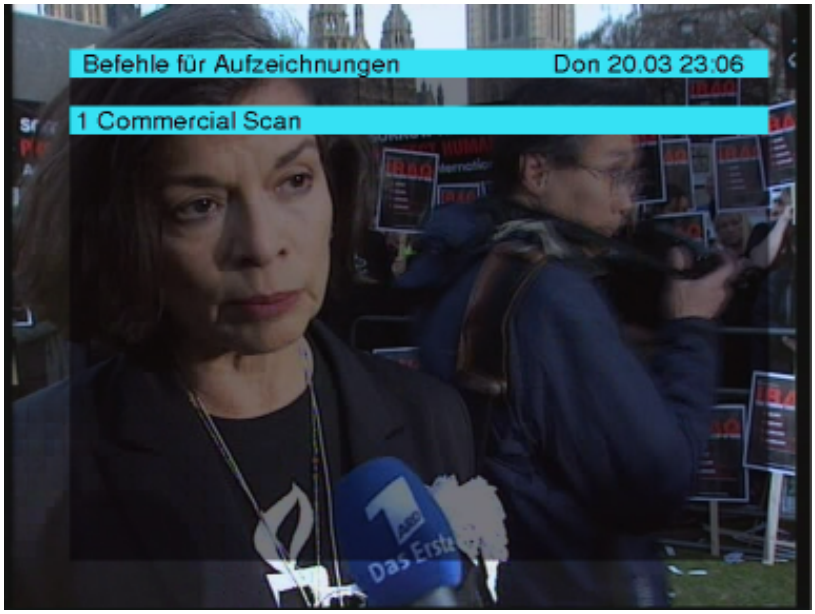
Im folgenden wird Schritt für Schritt gezeigt, wie die Software CommScan verwendet wird. Voraussetzung dafür ist die erfolgreiche Installation nach der Anleitung in Anhang B.

1. Aufrufen des Menüs von VDR.



2. Ins Aufzeichnungs-Untermenü wechseln.

3. Die gewünschte Aufzeichnung auswählen und mit Taste <Rot> das Befehlsmenü öffnen. Dies sollte nach erfolgreicher Installation von CommScan einen Menüeintrag namens „Commercial Scan“ beinhalten.



4. Nach Druck auf die <OK>-Taste und nochmaligem Bestätigen mit <OK> wird CommScan gestartet. Wenn dies erfolgreich ist, erscheint die Meldung „Commercial Scan started“. Sollte die Software vorher schon laufen, wird dies erkannt und es erscheint statt dessen die Meldung „Commercial Scan is already running!“.

5. CommScan wird im Hintergrund ausgeführt, so dass VDR ganz normal weiter benutzt werden kann, während die Analyse der Aufzeichnung durchgeführt wird. Über den aktuellen Fortschritt informiert es mittels Meldungen im OSD des VDR.

6. Ist CommScan beendet, erscheint die Meldung „Commercial Scan finished“ im OSD

7. Nun kann bei der Wiedergabe der Aufzeichnung mit der <OK>-Taste die Zeitleiste mit den gesetzten Schnittmarken angezeigt und diese überprüft

und falls nötig korrigiert werden. Mit der Taste <2> kann schließlich der Schnitt gestartet werden.

Abkürzungsverzeichnis

AV	Audio/Video
API	Application Programming Interface
CA	Conditional Access
DVB	Digital Video Broadcast
DVD	Digital Versatile Disc
DVP	Digital Video Project
EPG	Electronical Programme Guide
GPL	General Public License
IEC	International Electrotechnical Commission
ISO	International Standard Organization
MPEG	Motion Picture Experts Group
MP3	MPEG-1 Audio Layer III
OSD	On Screen Display
PES	Packetized Elementary Stream
PID	Packet Identification
TS	Transport Stream
VDR	Video Disc Recorder

Literaturverzeichnis

[BC87] R. A. Boie and I. Cox. Two dimensional optimum edge recognition using matched and wiener filters for machine vision. In *Proceedings of the IEEE First International Conference on Computer Vision*, pages 450–456, 1987.

[Fre91] Free Software Foundation. GNU General Public License. `http://www.gnu.org/copyleft/gpl.html`, 1991.

[JK99] Thorsten Janke and Markus Koppers. Optimierung und Implementierung eines Systems zur Unterdrückung von Werbeblöcken bei Aufzeichnungen mit dem Videorecorder, September 1999.

[Kra95] Tobias Kramer. Reklame ade. `http://www.jugend-forscht.de/`, 1995.

[MM02] Marcus Metzler and Ralph Metzler. The Linux DVB API, August 2002.

[MPE] ISO/IEC 13818 (MPEG-2). Coding of motion pictures and associated audio.

[Pel98] F. Pelster. Entwicklung einer Einheit zur Detektion von Audiosequenzen unter Verwendung des DSP TMS320F206, April 1998.

[SOS00] Michael Seul, Lawrence O'Gorman, and Michael J. Sammon. Practical algorithms for image analysis, 2000.

[Zin98] Fabian Zink. Entwicklung und Implementierung einer Windows-Applikation zur Erkennung von Werbeblöcken, September 1998.