h_da
HOCHSCHULE DARMSTADT
UNIVERSITY OF APPLIED SCIENCES
fbi
FACHBEREICH INFORMATIK

Hochschule Darmstadt
- Fachbereich Informatik -

Grundlagen der Videokompression

Seminararbeit im Kurs
Wissenschaftliches Arbeiten in der Inforamtik I

vorgelegt von
Justin Böhm und Matthias Greune

Referent: Michael Kröhn

Ausgabedatum: 21.10.2016
Abgabedatum: 16.12.2016

Erklärung

Ich versichere hiermit, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die im Literaturverzeichnis angegebenen Quellen benutzt habe. Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten oder noch nicht veröffentlichten Quellen entnommen sind, sind als solche kenntlich gemacht. Die Zeichnungen oder Abbildungen in dieser Arbeit sind von mir selbst erstellt worden oder mit einem entsprechenden Quellennachweis versehen. Diese Arbeit ist in gleicher oder ähnlicher Form noch bei keiner anderen Prüfungsbehörde eingereicht worden.

Justin Böhm
Darmstadt, den 14. Dezember 2016

Matthias Greune
Darmstadt, den 14. Dezember 2016

Abstrakt

Diese Arbeit gibt einen Überblick über die grundlegenden Methoden von Videokompressionsverfahren. Hierfür werden, sich am Encoding-Prozess von MPEG-1 orientierend, zunächst Arten der Irrelevanzreduktion, anschließend die wichtigsten Ansätze der Redundanzreduktion vorgestellt und anhand von Beispielcode erläutert. In eigenen Tests wurden unter Anwendung der vorgestellten Methoden zur verlustbehafteten und partieller Anwendung von verlustfreien Kompressionsalgorithmen Kompressionsraten mit einer Ratio bis zu 1:20 erreicht. Diese Arbeit zeigt somit, wie mittels weniger Grundlagen bereits vergleichsweise hohe Einsparungen im Speicherverbrauch von Videos erreicht werden können. Insbesondere mit Blick auf die stetig steigenden Forderungen nach höheren Framerates, und besserer Auflösung wird deutlich, welch hohe Relevanz das Thema Videokompression auch in Zukunft haben wird.

Inhaltsverzeichnis

Abbildungsverzeichnis

1. Einleitung

Videos sind seit der Entwicklung des Fernsehers zum Massenmedium kaum noch aus dem alltäglichen Leben wegzudenken. Seit dem Aufstieg des Internets als zentrales Kommunikationsmedium haben sich allerdings die Anforderungen an geeignete Speichertechniken von Videos drastisch verändert. Die heutigen Abspielgeräte haben noch immer begrenzten Speicherplatz und sind häufig nur mit schmalbandigen Internetanbindungen ausgestattet. Die Auflösung der Videos ist hingegen stark gestiegen. Um diese Ansprüche zu adressieren wurden Kompressionsalgorithmen entwickelt, die eine effiziente Speicherung speziell für bewegte Bilder ermöglichen. Die resultierenden Probleme aus dieser Art der Speicherung, wie Bildartefakte, sind heutigen Nutzern wohlbekannt. Die eigentliche Funktionsweise von Videokompression bleibt aber oft unbemerkt.

Diese Arbeit widmet sich den Grundlagen dieser Kompressionsverfahren. Zunächst werden die Methoden der Irrelevanzreduktion vorgestellt, die auf der Ausnutzung von psychovisuellen Effekten basieren. Hierbei werden gespeicherte Informationen entfernt, welche der menschliche Sehsinn nur schwer wahrnimmt und somit weniger relevant für das Erkennen des Bildes sind. Anschließend wird das Themenfeld der Redundanzreduktion beleuchtet, welche sich mit der Thematik beschäftigt, wie mehrmals vorkommende Informationen reduziert und entfernt werden können.

Diese Reihenfolge der Themen orientiert sich am MPEG-1 Encoding Prozess. MPEG-1 war einer der ersten relevanten digitalen Videokompressionsstandards, und wurde hauptsächlich für die Übertragung von digitalem Satellitenfernsehen eingesetzt. Im Vergleich zu aktuellen Standards sind die vorgeschlagenen Prozesse weniger komplex. Dieser Umstand macht MPEG-1 auch für die verwendeten Codebeispiele gut nutzbar.

2. Irrelevanzreduktion

Videos bestehen im Wesentlichen aus einer Aneinanderreihung von einzelnen Bildern. Dieses Kapitel untersucht, wie die in diesen Einzelbildern enthaltenen Daten möglicherweise reduziert werden können.

Die rohe Aufnahme eines Bildes bietet eine Fülle an Informationen. Mit Blick auf die Eigenschaften des menschlichen Sehsinns lässt sich hierbei allerdings feststellen, dass einige Informationen relevanter für das Erkennen eines Bildes sind, als andere. Die Irrelevanzreduktion beschäftigt sich mit der Trennung und Reduzierung von weniger wichtigen Informationen und bietet damit Methoden zur verlustbehafteten Datenkompression an.

Bei der Videokompression werden im wesentlichen zwei Eigenschaften zur Reduktion von Daten ausgenutzt. Zum einen nimmt das Auge Varianzen in der Helligkeit (Luminanz) stärker wahr, als Änderungen im Farbton (Chrominanz). Zum Anderen ist das Auge besser in der Lage niedrige Ortsfrequenzen zu erkennen, als hohe - erkennt also grobe Strukturen eher als feinere. Diese Eigenschaften können nun ausgenutzt werden, um einen guten Kompromiss aus akzeptabler Bildqualität und guter Datenreduktion zu finden [**akramullah_digital_2014**].

2.1. Chroma Subsampling

Das Chroma Subsampling nutzt den Umstand aus, dass Helligkeitsvarianzen besser wahrgenommen werden, als Farbvarianzen. Zumeist liegen die Bildinformationen im Ausgangsformat jedoch im RGB Farbmodell vor, wobei hier die Helligkeitswerte in jeden Kanal eingehen. Um nun aber die Chrominanz bei gleichbleibender Auflösung der Luminanz zu reduzieren wird eine getrennte Darstellung dieser Informationen benötigt. Hierfür wird im MPEG-1 Standard die $\mathrm{YC}_B\mathrm{C}_R$ Darstellung verwendet, wobei das Y für die Luminanz steht und in C_B und C_R die Farbwerte codiert werden. Die Umrechnung lässt sich mittels folgender Formeln realisieren [**itu-t_recommendation_1995**]:

$$
\begin{aligned}
Y &= 0.299 \cdot R + 0.587 \cdot G + 0.114 \cdot B \\
U &= (B - Y) \cdot 0.493 \\
V &= (R - Y) \cdot 0.877
\end{aligned}
$$

Nun kann das eigentliche Subsampling stattfinden, welches bei MPEG-1 bei einer Auflösung von 4:2:0 realisiert wird. Die erste Zahl gibt hierbei die horizontale Abtastrate der Luminanz an. Die zweite Zahl steht für die horizontale Abtastrate der C_B und C_R Kanäle in Relation zum ersten Wert. Die dritte Zahl gibt die vertikale Sampling Rate an, wobei diese entweder 2 oder 0 betragen kann, also entweder kein vertikales Subsampling, oder vertikales Subsampling von 2:1 stattfindet. Für den Fall von 4:2:0 Subsampling bedeutet dies, dass jeweils 2x2 Bildpunkte des C_B und C_R Kanals auf einen Bildpunkt in der Ergebnismenge abgebildet werden. Hiermit wird also die Auflösung des C_B und C_R Kanals halbiert, was zu einer Datenreduktion von insgesamt 50% führt. [**poynton_chroma_????**]

Das Chroma Subsampling bietet somit eine gute Möglichkeit der Kompression, die allerdings nicht verlustfrei abläuft. Artefakte können, wie in Abbildung A.1 im Anhang dargestellt, bei Verwendung dieser Methode vor allem bei scharfen, farbigen Kanten entstehen, wenn diese durch einen gesubsampleten Block verlaufen.

2.2. Diskrete Kosinus Transformation

Wie bereits oben beschrieben neigt der menschliche Sehsinn dazu niedrige Ortsfrequenzen eher zu erkennen, als höhere. Eine Ortsfrequenz ist definiert als „Anzahl bestimmter periodischer Erscheinungen bezogen auf einen räumlichen Abstand" [**atmwiki_ortsfrequenz_????**]. Wir erkennen also gröbere Strukturen mit einer niedrigen Ortsfrequenz eher als feinere Strukturen mit einer höheren. Um diesen Umstand nun auszunutzen muss das Ausgangsbild von der räumlichen Ebene auf eine Frequenzebene transformiert werden, damit anschießend, in dem darauf folgenden Schritt der Quantisierung, die höheren Frequenzen reduziert werden können. Diese Transformation lässt sich mittels einer zweidimensionalen Diskreten Kosinus Transformation (DCT) bewerkstelligen.

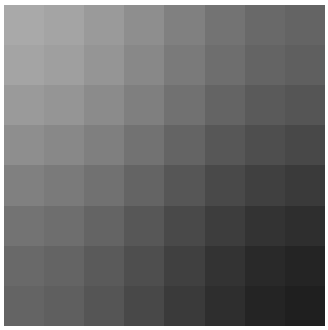
Die DCT ist eine Sonderform der Fouriertransformation, bei der eine Funktion mittels Sinusschwingungen approximiert wird. Die Fouriertransformation hat allerdings unter anderem den Nachteil, dass für jeden abgetasteten Punkt ein Tupel aus Amplitude und Phase bzw. Sinus und Kosinus Koeffizienten gespeichert werden muss. Die DCT nutzt nun den Umstand aus, dass das betrachtete Intervall begrenzt ist. Durch eine vertikale Spiegelung dieses Intervalls lassen sich die Sinus Anteile heraus kürzen, wobei am Ende lediglich Kosinus Anteile übrig bleiben, also nur ein Koeffizient pro abgetasteten Punkt gespeichert werden muss. Des Weiteren bewirkt die Spiegelung, dass Start- und Endpunkt äquivalent sind. Da die Fouriertransformation von einer unendlichen Folge ausgeht, muss der letzte Koeffizient den ggf. großen Unterschied zwischen Start- und Endpunkt ausglei-

chen. Sind diese Punkte aber äquivalent, wird die Kraft des letzten Koeffizienten nicht verschwendet [**symes_peter_digital_2004**]. Eine mögliche Implementierung ist in Listing B.1 im Anhang zu sehen. Verarbeitet werden mit der zweidimensionalen DCT immer 8x8 Blöcke eines jeden Kanals mit der Formel:

$$F(u,v) = \frac{1}{4} C_u C_v \sum_{x=0}^{7} \sum_{y=0}^{7} f(x,y) \cos\left(\frac{(2x+1)u\pi}{16}\right) \cos\left(\frac{(2y+1)v\pi}{16}\right)$$

$$\text{wobei} \begin{cases} C_u = \frac{1}{\sqrt{2}} \text{ für } u=0, \text{ ansonsten } C_u = 1 \\ C_v = \frac{1}{\sqrt{2}} \text{ für } v=0, \text{ ansonsten } C_v = 1 \end{cases}$$

Die Abbildung 2.1 zeigt das Resultat einer angewandten DCT auf einen schwarz-weißen 8x8 Pixelblock, welcher aus jeweils einer horizontalen und einer vertikalen Kosinus Schwingung besteht. Der sogenannte DC Wert ist der erste Wert der Matrix und gibt die mittlere Helligkeit an. Alle anderen Komponenten beschreiben die relative Abweichung zu diesem Wert und werden gemeinhin als AC Werte betitelt, wobei diese zugleich die zum unteren rechten Rand hin höher werdenden Ortsfrequenzen repräsentieren. Wie bereits zu erkennen führt die DCT oftmals selbst schon durch Rundung auf ganzzahlige Ergebnisse zu einer Matrix mit einer erhöhten Anzahl gleicher Werte, die sich für die Anwendung weiterer, verlustfreier, Kompressionsmethoden eignet.



800	200	0	0	0	0	0	0
200	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Abbildung 2.1.: Mittels DCT gut komprimierbarer 8x8 Pixelblock
Links: Ausgangsbild, Rechts: Resultierende DCT-Matrix

2.3. Quantisierung

Im vorigen Schritt wurde durch Anwendung der Diskreten Kosinus Transformation eine Matrix mit den korrespondierenden Ortsfrequenzen eines 8x8 Pixelblocks gewonnen. Um

nun tatsächlich eine Reduktion der höheren Ortsfrequenzen zu erreichen wird die Methode der Quantisierung angewandt. Hierbei wird eine ganzzahlige Division eines jeden DCT Koeffizienten mit einem Quantisierungswert vorgenommen. Das gerundete Ergebnis ist dann der quantisierte Wert. Durch diese Division und Rundung wird versucht die bisher noch hohen Koeffizienten zu verkleinern, sowie in den höheren Frequenzbereichen möglichst auf Ergebnisse gleich Null zu kommen.

Im Fall von MPEG-1 wird hierfür ein Uniform Scalar Quantizer verwendet, bei dem die Eingangswerte durch Division der Schrittgröße auf Bereiche gleicher Größe abgebildet werden, wobei eine stufenähnliche Charakteristik entsteht. [**symes_peter_digital_2004**]. Um die errechneten Ortsfrequenzen in Relation zur Wahrnehmung des menschlichen Auges zu reduzieren wird hierfür eine Quantisierungsmatrix verwendet. Diese beinhaltet separate Werte für jeden DCT Koeffizienten. Die Schrittgröße setzt sich für AC-Werte zusammen aus dem korrespondierenden Quantisierungswert der Quantisierungsmatrix und einem Quantisierungsfaktor (MQuant). Der Quantisierungsfaktor dient der Steuerung der Bildqualität und kann einen Wert zwischen 1 und 31 annehmen, wobei ein Quantisierungsfaktor von 1 für eine hohe Bildqualität sorgt, ein Faktor von 31 hingegen für eine stark reduzierte. Da das Auge sensibel gegenüber großräumigen Luminanzfehlern ist, wird der DC durch eine feste Schrittgröße von 8 dividiert. [**ISO13586**] Eine Implementierung des vorgestellten Algorithmus ist in Listing B.2 im Anhang zu sehen.

In Abbildung A.2 des Anhangs ist die angewandte Quantisierung exemplarisch an einem Beispielbild mit der im MPEG-1 Standard voreingestellten Quantisierungsmatrix (siehe Anhang, Tabelle A.1) sowie Quantisierungsfaktoren von eins, 16 und 31 dargestellt. Bei höheren Quantisierungsfaktoren sind hier deutliche Qualitätsverluste zu erkennen, wobei die groben Strukturen des Bildes aber erhalten bleiben.

Durch die Anwendung der DCT wird ein eingehender 8x8 Pixelblock also in eine Darstellung transformiert, die es erlaubt mittels der Quantisierung vor allem enthaltene höhere Ortsfrequenzen zu reduzieren. Diese Prozesse führen zunächst jedoch nicht direkt zu einer Datenreduktion, da trotz des erhöhten Anteils gleicher Werte in der Matrix eben diese Werte auch gespeichert werden müssen. Allerdings wurde erreicht, dass die Entropiekodierung, welche im nachfolgenden Kapitel erläutert wird, bessere Kompressionsergebnisse erzielen kann.

3. Redundanzreduktion

Die Redundanzreduktion ist ein weiterer wesentlicher Bestandteil der Videokompression, der schon seit den frühen Jahren digitaler Videosignale verwendet wird. Die in ihr enthaltene Entropiecodierung wurde schon mit dem ersten digitalen Videokompressionsstandard h.120, im Jahre 1984, ausgeliefert. Ein weiteres Teilstück, die Bewegungskorrektur, wurde dann 4 Jahre später, in der zweiten Version von h.120, eingeführt. Anstatt für den Menschen schlecht wahrnehmbare und somit unwichtige Bildteile zu entfernen, enfernt sie Redundanzen, sowohl in der Codierung, mittels Entropiecodierung, als auch temporaler Natur, mittels Bewegungskorrektur. Die daraus resultierende Reduzierung der Videodaten hat, in der Regel, im Gegensatz zur Irrelevanzreduktion keinen Einfluss auf die Qualität des vom Nutzer erkennbaren Bildes.

3.1. Entropiecodierung

Die Entropiecodierung ist eine verlustfreie Methode zur Kompression von Daten im Generellen. Sie findet also nicht exklusiv Anwendung in der Videokompression, bildet für sie jedoch trotzdem eine wichtige Grundlage.

$$h(i)=log(\frac{1}{p(i)}); \quad H=\sum_{i=1}^{E} p(i)*h(i); \quad L=\sum_{i=1}^{E} p(i)*l(i)$$

Ein Video kann als Nachricht, d.h. als Folge von Zeichen betrachtet werden. Jedes Zeichen an der Stelle i, hat dabei einen Informationsgehalt h, der sich aus seiner Eintrittswahrscheinlichkeit berechnet.

Die Entropie H ist der mittlere Informationsgehalt von einem Zeichen einer Nachricht. Sie berechnet sich aus der Summe aller Informationsgehälter mit einer Gewichtung der Wahrscheinlichkeit des iten-Zeichens.

Da die Zeichen einer Nachricht in den meisten Fällen nicht alle mit der gleichen Wahrscheinlichkeit auftreten, ist es sinnvoll die Codierung an die Wahrscheinlichkeiten der Auftretenden Zeichen anzupassen, sodass die am häufigst auftretenden Zeichen den jeweils kürzest möglichen Code erhalten.[**symes_peter_digital_2004**] Die mittlere Wortlänge L berechnet sich aus der Summe aller Codelängen mit einer Gewichtung der Wahrschein-

lichkeit des iten-Zeichens.

Das Shannon'sche Codierungstheorem besagt, dass, bei effizientes möglicher Kodierung mit eindeutiger Dekodierung, die mittlere Wortlänge eines Codes immer mindestens der Entropie einer Nachricht entsprechen muss. Dies bildet ein theoretisches Limit, dem sich so nah wie möglich angenähert werden soll.

Die Redundanz des Codes berechnet sich aus der Differenz von Entropie und mittlerer Wortlänge. Das Ziel für die Kompression eines Videos ist es, so wenig Code-Redundanz wie möglich zu erreichen.

Bei dem Videokompressionsstandard MPEG, wird die Entropiecodierung mittels einer Lauflängencodierung, gefolgt von einer Huffman-Codierung realisiert. Der Bitstrom eines Videos, welches zuvor quantisiert wurde, wird dabei auf aufeinanderfolgende Null-Bits untersucht. Diese werden anschließend zusammengefasst, indem ein Blockende angehängt wird. In einer eigenen Implementierung konnte hierdurch bereits eine Kompressionsrate von ca. 1:5 bei hoher Bildqualität erreicht werden. Bei niedrigerer Bildqualität und damit vermehrtem Auftreten von Nullen steig die Kompressionsrate auf 1:20 an (siehe Tabelle A.2 und Anhang C). Im Anschluss werden im resultierenden Bitstrom, im Zuge der Huffman-Codierung, mehrfach vorkommende Zahlenfolgen basierend auf ihrer Auftrittswahrscheinlichkeit mit kürzeren Codes ersetzt.

Durch das Zusammenspiel von Irrelevanzreduktion mit Quantisierung, Lauflängen-, sowie Huffman-Codierung ist eine effektive Kompression des gesamten Bitstroms, ohne eine merkbare Verminderung der Bildqualität möglich.

3.2. Bewegungskorrektur

Alle bis jetzt vorgestellten Ansätze der Videokompression beschäftigen sich mit der Kompression von Einzelbildern innerhalb eines Videos. Bei der Bewegungskorrektur hingegen, wird jenes Kompressionspotential ausgenutzt, welches innerhalb der Abhängigkeiten der Einzelbilder in einem Video steckt. Videos bestehen meist aus zusammenhängenden Szenen mit größtenteils unverändertem Inhalt innerhalb einer jeweils solchen Szene, den sogenannten temporalen Redundanzen.

Teilt man eine Szene in ihre Einzelbilder (Frames) auf, stellt man schnell fest, dass sich große Teile des Hintergrunds in mehreren Bildern wiederholen. Die Bewegungskorrektur nutzt die Redundanz des Hintergrunds aus, indem es mehrfach vorhandene Teile jeweils

nur ein Mal speichert und in den folgenden Bildern darauf referenziert um ein für den Zuschauer unverändertes Bild anzuzeigen. Da Videos üblicherweise zum Großteil mit redundanten Bildteilen in einzelnen Szenen gefüllt sind, macht die von Bewegungskorrektur erzielbare Kompression einen großen Teil des gesamt möglichen Kompressionpotentials innerhalb von Videos aus.

3.2.1. Frames

Beim Codieren mittels Bewegungskorrektur werden alle Video Einzelbilder in verschiedene Bildarten aufgeteilt. Es gibt rein intracodierte Frames, die sogenannten I-Frames. Bei ihnen handelt es sich um einzelne Vollbilder, die von keinem anderen Bild des Videos abhängen. I-Frames sind also für sich stehende Vollbilder, welche mit den üblichen Methoden der Bildkompression verkleinert wurden, somit bieten diese die geringsten Kompression. Außerdem gibt es intercodierte Frames, die nur eine vorhergesagte Differenz des Inhaltes in Abhängigkeit zu einem vorherigen I-Frame haben, die sogenannten P-Frames. Als letztes gibt es B-Frames, die sehr ähnlich zu P-Frames sind, jedoch in zwei Richtungen intercodiert wurden. Sie speichern nur die jeweils vorhergesagte Differenz des Inhaltes zum Vorherigen, sowie dem Nächsten I- oder P-Frame und benutzen somit den geringsten Speicherplatz. Die Vorhersagung wird mittels einer Anpassung der Codierungsreihenfolge realisiert, sodass diese ungleich der Anzeigereihenfolge ist.[**symes_peter_digital_2004**]

Wenn man diese Aufteilung jeweils nur einmal pro Szene anwenden würde, würden mehrere Probleme bei wahlfreiem Zugriff entstehen. So würden, wenn der I-Frame oder ein P-Frame einer Szene übersprungen wird oder gar komplett fehlt, die in den folgenden P- und B-Frames festgehaltenen Differenzen, auf den falschen I-Frame angewendet, sodass im Video störende Artefakte entstehen, die sehr denen auf Abbildung A.3 ähneln.

Da bei einem Großteil der Anwendungsfälle von Videos jedoch ein fast vollständig wahlfreier Zugriff gewünscht ist, teilt man jede Videosequenz in mehrere kleine aufeinanderfolgende Bildergruppen (Group of pictures, kurz GOP) auf. Eine GOP wird meist mit 2 Parametern angegeben, in diesem Beispiel M und N. Dabei ist N die Anzahl von Frames aus denen die GOP besteht, also die Distanz von einem I-Frame zum nächsten I-Frame. M gibt die Distanz von einem I- oder P-Frame, bis zum jeweils Nächsten an, somit ist N-1 die Anzahl von B-Frames, die nach einem I- oder P-Frame folgen.[**huszak2010analysing**] Eine Bildergruppe fängt immer mit einem I-Frame an und wiederholt sich bis zum Ende eines Videos mit einem konstanten Schema.

Mit den Parametern M=4 und N=12, würde die GOP dann aussehen wie folgt aussehen:

I BBB P BBB P BBB

Bei MPEG ist eine Aufteilung mit den Parametern M=3 bis 4 und N=11 bis 15 üblich. [**symes_peter_digital_2004**]

Betrachtet man ein Video mit einer üblichen Framerate von 25, dann ist dadurch wahlfreier Zugriff mit einer Genauigkeit von bis auf die Hälfte einer Sekunde gegeben. Außerdem wird der bei leichten Übertragungsfehlern auftretende Schaden minimiert, sodass das vom Endnutzer gesehene Video nur für sehr kurze Zeit Kompressionsartefakte aufweist, selbst bei komplettem Verlust eines I-Frames.

3.2.2. Makroblöcke

Beim Komprimieren eines Videos wird zunächst ein Frame pro GOP mittels Irrelevanzreduktion komprimiert und dann als Referenz zwischengespeichert. Die folgenden Bilder werden, um die vom Encoder benötigte Arbeit einfach aufteilen zu können, in sogenannte Makroblöcke unterteilt. Diese Makroblöcke sind bei den meisten Standards auf eine feste Größe von 16x16 Pixel gesetzt.[**symes_peter_digital_2004**] Die neusten Standards wie h.246/AVC unterstützen hingegen variable Blockgrößen und mehrere Refernzbilder. [**lin2009vlsi**]

3.2.3. Bewegungskorrektur

Das in Makroblöcke aufgeteilte Bild wird Block für Block verglichen um statische Bildinhalte zu erkennen. [**symes_peter_digital_2004**] Ein Bildinhalt ist statisch, wenn sich ein Block von einem Bild zum nächsten nicht verändert hat. Alle statischen Bildinhalte werden dann entfernt, stattdessen wird auf den Inhalt der gespeicherten Referenz verwiesen. Damit sind zwar statische Bildinhalte kein Problem, allerdings kann, zum Beispiel bei einem Schwenken der Kamera, der trotzdem in beiden Bildern identisch vorhandene Hintergrund nicht kodiert werden, da sich sein Block in der Referenz zum Block des folgenden Bildes unterscheidet. Um diese immer noch redundanten Bildinformationen ebenfalls entfernen zu können, bedarf es einer komplexeren Vorgehensweise, der Bewegungskorrektur. Die Bewegungskorrektur sucht jene Blöcke im neuem Frame mittels eines Block Matching Algorithmus heraus. Gefundene Blöcke werden mit einem Vektor, der von der Position des

neuen Blocks, auf die Position des Ursprungsblocks aus der Referenz zeigt, wie auf der Abbildung A.4 erkennbar, kodiert.[**symes_peter_digital_2004**] Beim Dekodieren kann dann mittels dieser Vektoren auf die Position des alten Blocks referenziert werden, sodass nur dieser Vektor gespeichert werden muss.

4. Zusammenfassung

In dieser Arbeit wurden die grundlegenden Methoden der Videokompression vorgestellt. Mittels der Irrelevanzreduktion lassen sich weniger relevante Informationen in einzelnen Bildern gezielt reduzieren. Hierfür werden Farbwerte mittels Chroma Subsampling ungenauer gespeichert als Helligkeitswerte. Durch Anwendung der DCT werden anschließend die Ortsfreuqenzen extrahiert und mit Hilfe der Quantisierung um die höheren Frequenzbereiche reduziert. Diese Verfahren resultieren in einer Darstellung, welche sich mittels der Entropiecodierung sehr gut komprimieren lässt. Unter Anwendung einer speziellen Form der Lauflängen- und Huffmancodierung wird die Menge der Daten möglichst nah an das theoretische Limit der enthaltenen Entropie herangeführt. Im Schritt der Motion Compensation werden schlussendlich Redundanzen in Bildsequenzen erkannt und reduziert.

In einer eigenen Implementierung der Methoden der Irrelevanzreduktion und unter Zuhilfenahme der Lauflängencodierung konnten Kompressionsraten von 1:5 bei guter Bildqualität und eine Ratio von 1:20 bei verminderter Bildqualität erreicht werden. Hierdurch wurde deutlich wie groß der Nutzen von Videokompression in Bezug auf die Speichergröße einer Datei sein kann.

** Auf Artefakte eingehen?

A. Weitere Abbildungen und Tabellen

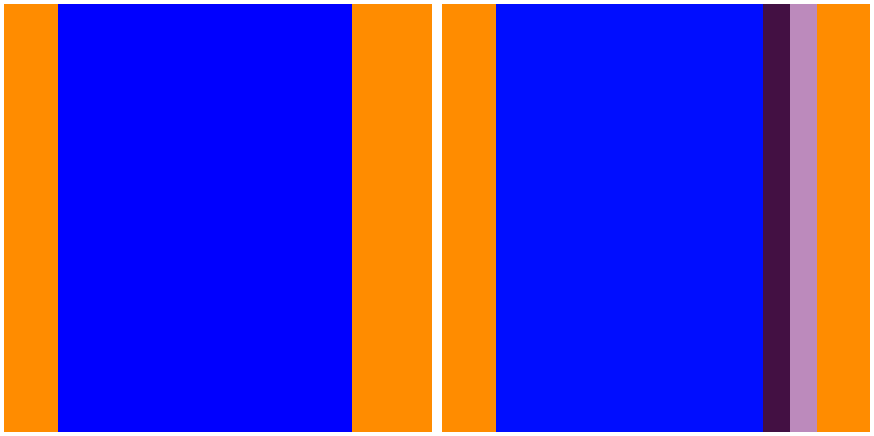


Abbildung A.1.: Artefakte durch Chroma Subsampling
Links: Original, Rechts: Subsampled. Die rechte Kante des blauen Farbblocks liegt in gesubsampleten 2x2 Blöcken, wodurch Artefakte entstehen. Die linke Kante liegt zwischen zwei 2x2 Blöcken, weshalb es zu keiner falschen Darstellung kommt.

8	16	19	22	26	27	29	34
16	16	22	24	27	29	34	37
19	22	26	27	29	34	34	38
22	22	26	27	29	34	37	40
22	26	27	29	32	35	40	48
26	27	29	32	35	40	48	58
26	27	29	34	38	46	56	69
27	29	35	38	46	56	69	83

Tabelle A.1.: Voreingestellte MPEG-1 Intracoding Quantisierungsmatrix. [**symes_peter_digital_2004**]

Abbildung A.2.: Ergebnis der Quantisierung mit verschiedenen Quantisierungsfaktoren
Oben links: Original, Oben rechts: Quantisiert mit Faktor 1, Unten links: Quantisiert mit Faktor 16, Unten rechts: Quantisiert mit Faktor 31.
Mit zunehmendem Quantisierungsfaktor ist ein ansteigender Verlust der Bildqualität zu beobachten, wobei grobe Strukturen weitestgehend erhalten bleiben. Original nach ***[brooke_cagle_2016]***

Genutzte Optionen			Größe in Kilobyte	Ratio
RLE	Chroma Subsampling	Quantisierung mit Faktor		
X		-	258.38	1.25
X	X	-	145.95	2.23
X	X	1	58.32	5.56
X	X	16	18.59	17.43
X	X	31	16.23	19.96

Tabelle A.2.: Testergebnisse der angewandten Kompressionsalgorithmen bei einer Ausgangsgröße von 324 Kilobyte des Originalbildes

Abbildung A.3.: Bildartefakte beim Auslassen eines I-Frames
Es wird jeder dritte Frame eines GIFs gezeigt, welches einen Szenenwechsel von einem Turner hinzu zu zwei springendem Delfinen darstellt. Hierbei wurde im Ursprungsvideo mit Absicht der beim Szenenwechsel erzeugte I-Frame zerstört um künstlich Bildartefakte zu erzwingen und sogenannte „Glitch-Art" zu erzeugen.
Original nach [**supergif**]

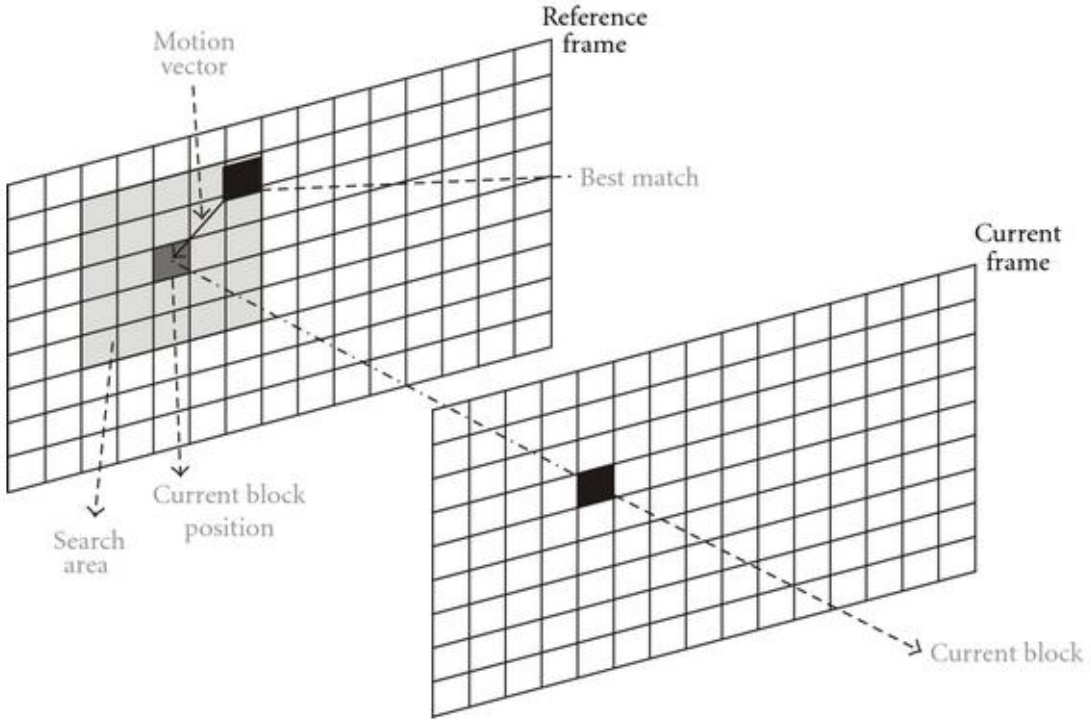


Abbildung A.4.: Suchen von identischen Blöcken in zwei Frames mittels Bewegungskorrektur
Das Referenz Bild wird anschließend mit dem resultierenden Motion Vektor codiert
Quelle: [**lopes_memory_2012**]

B. Listings

```
def dct(f):
    # initialize resulting DCT array
    F = [8*[0], 8*[0], 8*[0], 8*[0], 8*[0], 8*[0], 8*[0], 8*[0]]
    # Go through f and calculate DC/AC for each value
    for u in range(0, 8):
        for v in range(0, 8):
            if u == 0:
                cu = 1/math.sqrt(2)
            else:  # u > 0:
                cu = 1
            if v == 0:
                cv = 1/math.sqrt(2)
            else:  # v > 0:
                cv = 1

            sum = 0
            for x in range(0, 8):
                for y in range(0, 8):
                    sum += f[x][y] * math.cos((((2 * x) + 1) * u
                        * math.pi ) / 16) * math.cos((((2 * y) +
                        1) * v * math.pi) / 16)
            # save result in F
            F[u][v] = round((cu * cv * sum) / 4)
    return F
```

Listing B.1: Implementierung der DCT für ein 8x8 Array

```
def quantize(dct, quantizer, MQuant=1):
    result = numpy.empty_like(dct)
    for x, row in enumerate(dct):
        for y, coefficient in enumerate(row):
            if x == 0 and y == 0:
                result[x][y] = int(coefficient / 8)
            else:
                result[x][y] = int( 8 * coefficient / (MQuant *
                   quantizer[x][y] ))
    return result
```

Listing B.2: Implementierung des Quantisierungsprozesses nach MPEG-1 Standard ohne Clipping

C. Erläuterung des Testvorgehens

Als Grundlage der Testvorgänge diente das dieser Arbeit auf CD beiliegende Programm. Das Chroma Subsampling ist als Mittelwertberechnung von jeweils 2x2 Blöcken während der Kompression realisiert. Die DCT ist wie im Kapitel 2.2 vorgestellt implementiert. Als Quantisierungsmatrix wird die in Tabelle A.1 dargestellte MPEG-1 Intracoding Quantisierungsmatrix verwendet. Das Verfahren ist nach Listing B.2 implementiert. Als Lauflängencodierung wird eine abgewandelte Form des ZigZag Encodings nach JPEG-Standard verwendet. Hierbei werden lediglich Nullen lauflängencodiert, was in der Praxis ein Symbol für die Angabe des RLE encodierten Zeichens spart. Aufgrund der Charakteristik der aus der Quantisierung resultierenden Matrix hat sich diese Form als effizienter herausgestellt, als eine klassische RLE. Da kein Predictive Coding der DC-Werte verwendet wird, werden, anders als im JPEG Standard, auch diese Werte encodiert. Zur Berechnung der resultierenden Bildgrößen wurden folgende Werte angenommen: Jeder RGB Kanal lässt sich als 8 Bit Integer darstellen. Da jedes Pixel aus drei RGB Kanälen besteht resultiert hieraus eine Größe von 24 Bit. Zur Speicherung eines RLE encodierten Deskriptors werden aufgrund der zusätzlich benötigten Symbole 9 Bit benötigt. Als Quellbild wurde das in A.2 dargestellte Originalbild verwendet. Die Speicherung der RGB Werte benötigt nach oben beschriebener Annahme 324 Kilobyte. Das Bild hat eine Abmessung von 288x384 Pixel.