



Lehrstuhl für
Kommunikations-
technik

Masterarbeit M 04-2017

**Untersuchung des Einflusses der
Quellencodierung auf verdeckt
übertragene Inhalte bei der Screen
-Camera Visible Light Communication**

von

Ying Wang

Abgabedatum: 22. May 2018

Prof. Dr.-Ing Rüdiger Kays • Lehrstuhl für Kommunikationstechnik • TU Dortmund

Inhaltsverzeichnis

1 Einleitung	1
1.1 Motivation	1
1.2 Zielsetzung	2
1.3 Aufbau der Arbeit	2
2 Grundlagen	3
2.1 Farträume	3
2.1.1 RGB-Farbraum	3
2.1.2 YUV-Farbraum	5
2.1.3 Farbmodellumrechnung	5
2.2 Wiedergabe- und Aufnahmegeräte	6
2.2.1 Zeitliche Auflösung	7
2.2.2 Örtliche Auflösung	8
2.2.3 Farbübersprechen	9
2.2.4 Gradation	11
2.3 Quellencodierung für Standbilder und Bewegtbilder	14
2.3.1 Standbildcodierung	14
2.3.2 Bewegtbildcodierung mittels H.264/AVC	21
2.4 Qualitätsbewertung der übertragene Daten	26
2.5 Systembeschreibung von DaViD	27
3 Simulation des Einflusses der Quellencodierung auf ideale Datensignale	30
3.1 Einfluss vor der Quellencodierung	30
3.2 Einfluss der Quellencodierung für Standbilder	35
3.2.1 Analysieren des Einfluss auf DCT-Codierung	35
3.2.2 Simulation der Standbildcodierung in FFMPEG	39
3.2.3 Auswertung	43
3.3 Einfluss der Quellencodierung auf Bewegtbilder	44
3.3.1 Simulation in FFMPEG	45
3.3.2 Auswertung	47

4 Experimente	52
4.1 verwendete Geräte und Testumgebung	52
4.2 Experiment mit Industriekamera	54
4.2.1 Quellencodierung auf Videos	54
4.3 Experiment mit Smartphonekamera	57
4.3.1 Quellencodierung auf Videos	57
5 Zusammenfassung	66
A Erster Anhang	68
A.1 Abbildungen	68
Abbildungsverzeichnis	86
Tabellenverzeichnis	89
Quellcodeverzeichnis	90
Literaturverzeichnis	91

1 Einleitung

Drahtlose Übertragung findet bei zunehmender Produktvielfalt eine immer größere Bedeutung. Dadurch können ein effizienter Ersatz sein für die aufwendige kabelgebundene Systeme und den spontanen und mobilen Datenaustausch ermöglichen. Des Weiteren kann die kabellose Übertragung eine große Übertragungsstrecke (von wenigen Metern bis zur Satellitenkommunikation) abdecken. Mit heute verfügbarer drahtloser Technik sind viele Mobilitätsansprüche sowohl im industriellen Bereich als auch für Privatnutzung realisierbar. Alle drahtlosen Geräten arbeiten in ihren eignen Frequenzbändern, in denen die Signale zwischen Sender und Empfänger individuelle übertragen werden. Mit der wachsenden Nachfrage von drahtlose Übertragung tritt das Problem allerdings darauf, dass nicht genügend Frequenzen zur Verfügung stehen.[1]

1.1 Motivation

Die aktuelle Forschungen wie VLC (Visible Light Communication) ist eine alternative drahtlose Datenübertragungstechnologie. Dabei werden Daten mit Hilfe des Lichtes durch Modulation übertragen. Das zur Übertragung benutztes Licht befindet sich im sichtbaren Spektrum. Zur Erzeugung des Lichtes werden LEDs verwendet, womit eine höhere Übertragungsraten erreicht werden.[2] Das Forschungsprojekt DaViD (Data transmission using Video devices), das vom Lehrstuhl für Kommunikationstechnik der TU Dortmund in den letzten Jahren vorangetrieben wird, werden die Daten anhand Bildwiedergabe- und Aufnahmegeräte in der SC-VLC (Screen-Camera Visible Light Communication) übertragen. Die Modulation im Projekt geht davon aus, dass das Datensignal in Form einer differentiellen Amplitudenänderung dem Chrominanz-Kanälen eines Videosignales überlagert wird und die Datenübertragung simultan mit einer Videowiedergabe erfolgen soll, ohne dass das überlagerte Datenframe im Video von Betrachter wahrgenommen wird. Als Sender kommen herkömmliche Wiedergabegeräte zum Einsatz, wie z.B. Fernsehen und Computer-Bildschirm. Als Empfänger wird der Bildinhalt anhand Smartphonekamera aufgenommen. Daraus werden die Datensignale nach der Aufnahme von Videoframes extrahiert und decodiert. Um die unkomprimierte Bildaufnahme zu codieren, benötigt eine sehr große Datenmenge. Daher spielt die Quellencodierung bei der

Aufnahmegeräte, wie zum Beispiel Smartphones eine wichtige Rolle. Die Herausforderung besteht darin, dass die komprimierte Daten nach der Aufnahme nicht nur für die Betrachter möglichst nicht wahrnehmbar sondern auch detektierbar sind. Da der Einfluss von Kompression auf der verdeckte Inhalt variiert unter verschiedene Bedingung, z.B. mit unterschiedlicher Modulationsamplitude und Größe der Datenblock als auch Bitrate.

1.2 Zielsetzung

In dieser Arbeit wird der Einfluss verlustbehafteter Kompression auf verdeckt übertragene Bildinhalt bei der SC-VLC berücksichtigt, somit bei der Aufnahme Information noch detektierbar wird. Zur vorgestellten Untersuchung werden einigen relevante Parameters zur Kompression benutzt und die erhaltene Ergebnisse bewertet. Dadurch können die geeignete Parameters zur Übertragung und Detektion der Daten festlegt werden.

1.3 Aufbau der Arbeit

Die Arbeit ist in fünf Kapitel eingeteilt. Im Kapitel 1 werden die Motivation, Zielsetzung und Aufbau kurz vorgestellt. Danach folgt im zweiten Kapitel die relevante Grundprinzipien zur Farbräume sowie die Umrechnungen zwischen Farbräumen, die Technologie von Wiedergabe- und Aufnahmegeräten, Quellencodierung für Standbilder und Bewegtbilder. In der letzten zwei Abschnitt im diesen Kapitel werden die Bewertungsverfahren und das untersuchtes System DaViD erklärt.

In Kapitel 3 wird zuerst der Einfluss vor der Quellencodierung analysiert. Danach wird die Simulation der Codierung für ideale Einzelbilder, Standbilder sowie Bewegtbilder durchgeführt und die Auswertung ist nach jeder Simulation angegeben.

Anschließend wird der Einfluss anhand zwei Experimente weiter mit Kameras weiter untersucht. Die Durchführung des ersten Experiments wird anhand einer Industriekamera realisiert, während der zweiter mit einer Smartphonekamera. Die Auswertung der beide Ergebnisse wird am Ende jedes Experiment beschreibt.

Im letzten Kapitel 5 werden die Ergebnisse von Simulation und Experimente verglichen und auf Basis einer komparativen Auswertung von geeigneter Parameters zur Übertragung und Detektion ermittelt.

2 Grundlagen

In diesem Kapitel werden zunächst die Grundprinzipien von Farbräumen und die Umrechnungen dazwischen erläutert. Der anschließende Abschnitt thematisiert die zeitliche und örtliche Auflösung sowie die Gradation und Farbkorrektur in den Wiedergabe- und Aufnahmesystemen. In den nächsten zwei Abschnitten werden die Quellencodierung für Standbilder und Bewegtbilder (Videos) behandelt. Zuletzt folgt die Vorstellung des Projekts DaViD und dessen Modulationsverfahren.

2.1 Farbräume

Die Farbwahrnehmung ist ein in den Rezeptoren des Auges bearbeiteter Prozess. Es gibt drei verschiedene Arten von Zapfen im menschlichen Auge, die jeweils auf rot, grün oder blau reagieren und denen ein großer Helligkeitsbereich wahrgenommen werden kann. Die Zapfen sind nur für das Tagsehen zuständig, wohingegen die Stäbchen beim farblosen, photonischen Sehen im Dunkeln zum Einsatz kommen. Erkennbar ist, dass eine Helligkeitsänderung empfindlicher wahrgenommen wird, als eine Farbänderung[3]. Dies wird als Referenz zur Untersuchung verschiedener Farbräume dienen. Mit mathematischen Methoden können die empfundenen Farben eindeutig beschrieben werden, sodass eine präzise Berechnung möglich ist. Der Farbraum ist eine gewählte Methode zur Darstellung von Helligkeit, Luminanz oder Luma und Chrominanz. In dieser Arbeit wird hauptsächlich mit RGB- und YUV-Farbräumen beherrscht. Weitere Farbräume im Bereich der Digitalbildverarbeitung sind zum Beispiel XYZ- und Lab-Farbräume.

2.1.1 RGB-Farbraum

Der RGB-Farbraum basiert auf drei Grundfarben: R (Rot), G (Grün) und B (Blau). Die Grundfarben werden auch als Primärvalenzen bezeichnet. In diesem Zusammenhang wird ein Koordinatensystem mit drei Achsen betrachtet, die jeweils den Farbort einer Primärvalenz darstellen. Haben alle Grundfarben gleichzeitig den Wert Null, bedeutet dies, dass kein Lichteinfall ermittelt wurde. Es ergibt sich die Farbe Schwarz. Werden der Rot-, Grün- und Blaukanal

gleichzeitig in höchster Intensität überlagert, erscheint Weiß. Eine Addition dieser drei Farben mit unterschiedlicher Gewichtung erzeugt den gesamten Farbeindruck. Weil der Farbeindruck nicht von der Helligkeit abhängt, kann der Farbeindruck (RGB-Farbanteile) zu 1 festgelegt[4]. Auf diese Weise wird der Übergang durch Umstellung der Gleichung von drei auf zwei Dimensionen vereinfacht, da die dritte Farbe leicht durch Kenntnis zweier gemischter Farben berechnet werden kann.

$$R + G + B = 1 \quad (2.1)$$

Abbildung 2.1 veranschaulicht die Normierung von RGB-Anteilen. Zwei Farbvektoren (F_1 und F_2) werden miteinander addiert. Dazu wird ein Vektor F erzeugt, welcher durch seine Ausrichtung die Chrominanz und durch seinen Betrag die Luminanz darstellt. Auf diese Weise ist zu erkennen, dass sich die drei Farben in einer Ebene aufspannen lassen, deren Schnittpunkt mit dem Vektor einen Farbeindruck zeigt. Allerdings sind nicht alle vom Menschen wahrnehm-

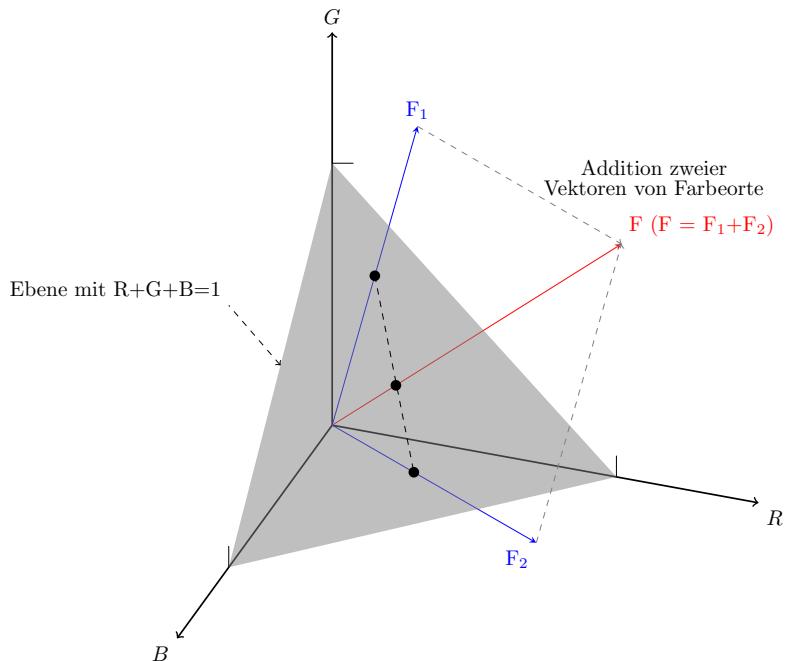


Abbildung 2.1: Vektorielle Farbaddition[4].

baren Farben im RGB-Farbraum darstellbar. Manche Farben konnten auch durch negative Anteile gemischt werden. Um absolute Werte zu erhalten, die für eine exakte Reproduktion eines Farbtöns benötigt werden, muss eine Definition die genaue Position der Primärfarben und des Weißwertes festlegen. Im RGB-Farbraum erfüllt z.B. sRGB (Standard-RGB) oder Adobe-RGB diese Normierung.

2.1.2 YUV-Farbraum

Die Dreifarbentheorie besagt aber auch, dass nicht nur die drei Grundfarben zur Farbdarstellung herangezogen werden können, sondern auch andere geeignete Farben. Es gibt auch andere Farbmodelle, die eine Farbe nicht durch die Grundfarben, sondern andere Beschreibungen ausdrücken, wie beispielsweise der YUV-Farbraum, welcher zu den Helligkeit-Farbigkeit-Modellen gehört.

Anders als im RGB-Farbraum wird die Farbe im YUV-Farbraum in eine Helligkeitskomponente Y (Luminanz) und zwei Farbkomponenten U und V (Chrominanzkomponenten) eingeteilt, wobei U ein Maß für die Abweichung von der „Mittelfarbe“ Grau in Richtung Blau darstellt. V ist die entsprechende Maßzahl für die Differenz zu Rot. Weil die Änderung der Helligkeit im menschlichen Auge empfindlicher wahrgenommen ist als Farbänderung, wird die Darstellung in diesem Farbraum typischerweise für die Bild- oder Videokomprimierung und Übertragung verwendet, indem die Datenmenge in einem Schritt ohne sichtbare Qualitätsveränderungen um gut ein Drittel reduziert wird (Der Grund wird in Abschnitt 2.2.2 erläutert).

2.1.3 Farbmodellumrechnung

Die wahrgenommene Helligkeit wird Luminanz und die Farbdifferenz als Chrominanz bezeichnet. Die Änderung der Struktur in Farbe ist weniger zu erkennen als in Helligkeit im menschlichen Auge. Daher können Farbinformationen reduziert werden, während die die Helligkeitsinformationen nicht ändern. Um eine höhere Kompressionseffizienz und gute visuelle Qualität zu erzielen, werden die RGB-Farbanteile in Luminanz- und Chrominanzkomponenten umgewandelt. Aufgrund der Unabhängigkeit von jeder Komponente, können die Falschfarben nach hoher Kompression verhindert werden.

Umrechnung nach ITU-R BT.709

ITU-R BT.709 ist ein Standard für das Format von HDTV (high-definition television), das durch den International Telecommunication Union (ITU) spezifiziert wird. Die Umrechnung vom RGB-Farbraum im Matrixformat in den YUV-Farbraum im HDTV-Format nach ITU-R BT.709 wird in folgender Matrix dargestellt:

$$\begin{bmatrix} Y' \\ U \\ V \end{bmatrix} = \begin{bmatrix} 0.2126 & 0.7152 & 0.0722 \\ -0.09991 & -0.33609 & 0.436 \\ 0.615 & -0.55861 & -0.05639 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \quad (2.2)$$

Die Luminanzkomponente (Y) wird aus einer gewichteten Summe von R, G, B-Komponenten durch lineare Transformation ermittelt. Dies gilt auch für die U- und V-Komponenten. Die inverse Transformation vom YUV-Farbraum in den RGB-Farbraum für HDTV nach ITU-R BT.709 wird in der folgenden Matrix dargestellt:

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1.28033 \\ 1 & -0.21482 & -0.38059 \\ 1 & 2.12798 & 0 \end{bmatrix} \begin{bmatrix} Y' \\ U \\ V \end{bmatrix} \quad (2.3)$$

Die Rot-, Grün- und Blaukomponenten können aus einer gewichteten Summe von Luminanz- und Chrominanzkomponenten wiederhergestellt werden.

Umrechnung nach ITU-R BT.601

ITU-R BT.601 ist ein durch den ITU-R veröffentlichter Standard, der in JPEG, MPEG und H.26X-Codierungsverfahren verwendet wird. Die Umrechnung von RGB-Farbraum in den YUV-Farbraum nach ITU-R BT.601 ist im Folgenden dargestellt:

$$\begin{bmatrix} Y' \\ U \\ V \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.14713 & -0.28886 & 0.436 \\ 0.615 & -0.51499 & -0.10001 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \quad (2.4)$$

Bei der Umrechnung in ein HDTV-Format wird die Luminanzkomponente mit der gewichteten Summe von R-, G- und B-Komponenten linear in den YUV-Farbraum transformiert. Die Chrominanzanteile werden im gleichen Fall transformiert. Die inverse Transformation vom YUV-Farbraum in den RGB-Farbraum nach ITU-R BT.601 ist im Folgenden beschrieben:

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1.13983 \\ 1 & -0.39465 & -0.58060 \\ 1 & 2.03211 & 0 \end{bmatrix} \begin{bmatrix} Y' \\ U \\ V \end{bmatrix} \quad (2.5)$$

Die Rot-, Grün- und Blaukomponenten können aus einer gewichteten Summe von Luminanz- und Chrominanzkomponenten zurück transformiert werden.

2.2 Wiedergabe- und Aufnahmegeräte

Bildaufnahmegeräte stellen eine Umsetzung der Lichtintensitätsänderung innerhalb einer Abbildung zur Verfügung. Dazu gehören digitale Kameras und Smartphonekameras. Die Kamera

kann mit dem Licht arbeiten, in dem das aufzunehmende Objekt reflektiert wird. Jedoch ist es erforderlich, viele lichtempfindlicher Elemente flächig anzuordnen. Jedes Element stimmt dann mit einem Bildpunkt (Picture Element, Pixel) überein und hat die Aufgabe, eine elektrische Ladung in Abhängigkeit von der Lichtintensität zu verändern.

Bildwiedergabegeräte dienen zur Rekonstruktion der aufgenommenen Bilder, d. h. die Umwandlung des Videosignals in sichtbares Licht ist erfolgt. Daher erscheinen die Bildpunkte anhand einiger Verfahren nacheinander auf dem Bildschirm. Die Trägheit des Wiedergabewandlers und des menschlichen Auges hat den Eindruck eines ganzen Bildes zur Folge.

Die Darstellung eines Farbbildes entsteht bei allen durch diskrete Variierung der Grundfarben RGB eines Pixels in ihrer Helligkeit. Die 8-Bit-Codierung für die Informationen über die Helligkeit eines RGB-Subpixels wird in der Regel in Stufen von 0 bis 255 angegeben. Infolgedessen kann ein Großteil der wahrgenommenen Farben angezeigt werden.

2.2.1 Zeitliche Auflösung

Eine Funktionsweise von Wiedergabegeräten, wie z.B. ein CRT-basierter Display, ist das Rasterscanning, wobei ein von links nach rechts und von oben nach unten verlaufender Scanmechanismus verwendet wird. Der vollständige Scan eines Bildes wird als Frame bezeichnet. Das gesamte Frame besteht aus 625 Zeilen im Standardfernsehsystem. Jedes Frame wird in zwei Halbbilder zerlegt, die miteinander verschachtelt werden. Dies wird als Zeilensprungverfahren (engl.: Interlaced-Mode) bezeichnet. D.h. die örtliche zweite Zeile wird zeitlich als Zeile 313 übertragen. Um einen fließenden Bildeindruck zu erzeugen, wird die Bildwiederholfrequenz definiert, wobei zur entscheidenden Wirkung der Flimmereindruck eines Bildschirms entscheidend ist. Unter Verwendung des Zeilensprungverfahrens wird z.B. beim CRT-Fernsehen die Bildwiederholfrequenz (50/60 Hz) im Vergleich zur Bildrate (25i/30i fps) verdoppelt, um das Flimmen zu verhindern. Die heutigen Wiedergabegerät, - wie etwa bei Kinofilmen - besitzen eine Bildwiederholfrequenz von 24 Hz, bei aufwendig produzierten Kinofilmen - wie 3D-Filmen - liegt die Frequenz hingegen bei 48 Hz. Fernseher haben eine Bildwiederholfrequenz von 25i/30i, Computerspiele von 60-240 Hz. Die in dieser Arbeit thematisierte Modulation (siehe folgenden Abschnitt 2.5) wird unter Ausnutzung eines Smartphones zur Erfassung der Daten aus Fernsehgeräten durchgeführt. Die meisten heutigen Smartphones haben unterschiedliche Modi in Bezug auf ihre Videoaufnahmegeräte, wie beispielsweise beim Google Pixel XL mit HD 720p (bis zu 240 fps), FHD 1080p video (bis zu 120 fps) sowie 4K 2160p video (bis zu 30 fps). Beim 4K-Modus stehen nur maximal 30 fps zur Verfügung. Der Grund liegt darin, dass der Chip von Smartphones eine Begrenzung hat, wobei limitierte Dateien pro Sekunde geschrieben werden können. Die Aufnahme eines Videos mit hoher Auflösungen bei hoher Framerate erfordert eine Kamera, die nicht nur hohe Auflösung erfassen kann und eine sehr

kurze Verschlusszeit besitzt, sondern auch die Daten sehr schnell speichern kann. In solchen Fällen ist die Begrenzung der Aufnahme von Kamera besteht in nicht ihrer Sensoren oder Verschlusszeit, sondern die Schreibgeschwindigkeit.

2.2.2 Örtliche Auflösung

Die Tabelle 2.1 stellt die Auflösung sowie die physikalische Kenngröße von Videoformaten dar, die im DaViD-Projekt(siehe Abschnitt 2.5) für weitere Simulationen und Experimente verwendet werden. Die Breite beschreibt die gesamte Anzahl der Spalten eines Bildes, während die

Tabelle 2.1: Einige Videoformate.

Format	Technik	Breite	Höhe	Seitenverh.	Pixel
DVB (PAL), DVD-Video (PAL)	digital	720	576	4:3 oder 16:9	414.720 (0,41MP)
HDTV(„720p“)	digital	1280	720	16:9	921.600 (0,92 MP)
FullHD („1080p“)	digital	1920	1080	16:9	2.073.600 (2,07 MP)
UHD 4K, 2160p	digital	3840	2160	16:9	8.294.400 (8,30 MP)

Höhe die gesamte Zeile beschreibt. Das Seitenverhältnis bezeichnet das Verhältnis der Breite eines Bild zu seiner Höhe. Im Videoformat wird das Seitenverhältnis häufig als Bruch N:M (z.B. 4:3) angegeben. Die Pixel in der Tabelle sind die Gesamtanzahl der Bildpunkte. Sie werden errechnet, indem man die Anzahl der Bildpunkte je Zeile mit der Anzahl der Bildpunkte je Spalte multipliziert. Die Ergebnisse werden mit der Einheit Megapixel (Abkürz. MP) dargestellt.

Da feine örtliche Farbunterschiede vom menschlichen Auge nur schwer wahrgenommen werden, kann die Auflösung in die Chrominanzanteile U und V reduziert werden, ohne, dass sich eine merklicher Qualitätsänderung ergibt. Dies hat den Vorteil für die unmittelbare Verringerung der Datenmenge. Eine gängige Technik ist die Farbunterabtastung. Die Helligkeitsinformation (Luminanz) und Farbinformation (Chorminanz) kann erst unter Verwendung eines geeigneten Farbmodells getrennt beschrieben werden. Die Chrominanz wird mit einer gegenüber der Luminanz reduzierten Abtastrate gespeichert, während die Helligkeit mit voller Rate abgetastet wird. Es existieren verschiedene Formen der Farbunterabtastung. Die unterschiedlichen Modi zur Unterabtastung sind in Abbildung 2.2 zu erkennen.

Die numerischen Bezeichnungen in Abbildung 2.2 weisen die Abtastraten Y (gelber Kreis) und U(rot halbes Kreis) und V (blau halbes Kreis) für die Farbdifferenzsignale auf.

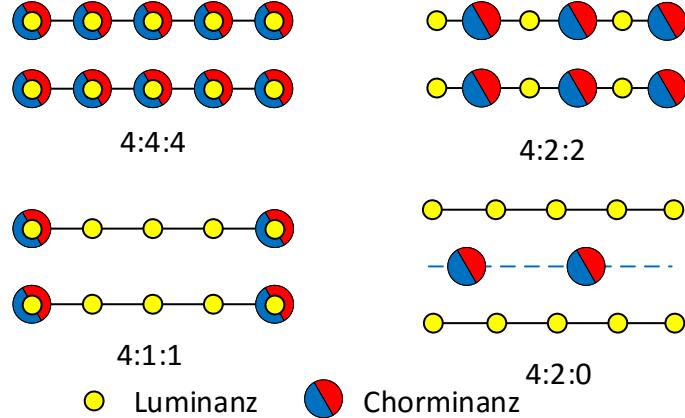


Abbildung 2.2: YUV-Abtastmodus.

Beim Standard ITU-R BT.601 ergibt sich die Unterabtastung der Farbsignale im digitalen Videostandard. Im 4:4:4-Modus werden alle Komponenten mit voller Abtastrate gearbeitet, während im 4:2:2-Modus nur im Luminanz-Zweig mit voller Abtastfrequenz und in den beiden Farbdifferenzsignalen mit halber Rate abgetastet wird. Im 4:1:1-Modus wird die Luminanz mit voller Rate abgetastet und bei der Chrominanz eine Quartal Abtastrate durchgeführt wird. Dieses Schema hat unterschiedliche Farbaufösungen in der horizontalen und vertikalen Richtung zur Folge. Um diesen Missstand zu beheben, wurde der 4:2:0-Modus definiert, wobei die Farbdifferenzsignale in beiden Raumrichtungen mit einer identischen Abtastrate zeilenweise abwechselnd übertragen werden.

Die Auflösung bei der Aufnahmegeräten wird dadurch definiert, dass kleinste Strukturen durch die Sensoren wiedergegeben werden können. Zur Beschreibung der Auflösung bei den Sensoren von Digitalkameras wird die Anzahl der Pixel zur Orientierung benutzt, die meistens in Megapixeln spezifiziert wurden. Tatsächlich hängt die erreichte Auflösung nicht nur von der Anzahl der Pixel, sondern auch von der Sensorgröße sowie der verwendeten Hard- und Software in der Kamera für die Bildverarbeitung ab. Die Tabelle 2.2 zeigt die Konfiguration einiger Smartphones über Bildsensoren und Auflösungen:

2.2.3 Farbübersprechen

Theoretisch wird die Empfindlichkeit des menschlichen Auges und des Sensors in der Kamera auf das sichtbare Spektrum so dargestellt, dass das dreifarbiges Licht sich nicht gegenseitig beeinflusst und sich jedes Licht nicht miteinander überlagert (siehe Abb.2.3). Tatsächlich ist

Tabelle 2.2: Einige Konfigurationen von Smartphones.

Name	Bildschirmauflösung	Anzahl der effektiven Pixel	Pixelgröße	Sensorgröße
Google Pixel XL	2560 × 1440 (534 ppi)	12,35 MP	1,55 μm	1/2,3"
Google Pixel XL 2	2880 × 1440 (538 ppi)	15,82 MP	1,4 μm	1/2,6"
Samsung S8	2960 × 1440 (572 ppi)	12 MP	1,4 μm	1/2,55"
Huawei P20	2244 × 1080 (408 ppi)	Dual: 20 MP	1,55 μm	1/1,7"

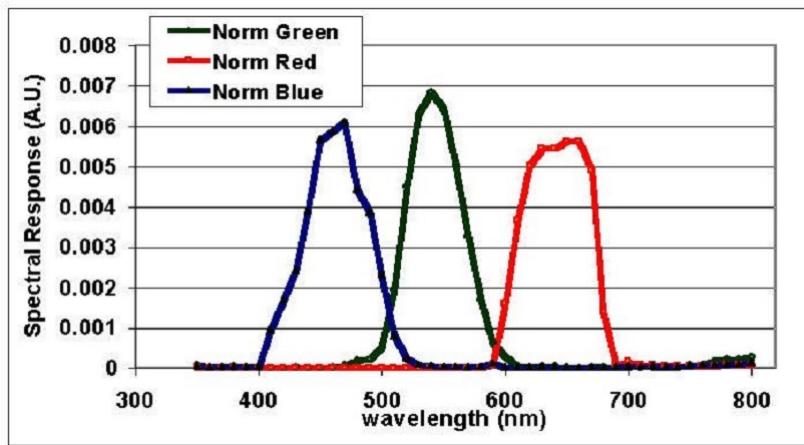
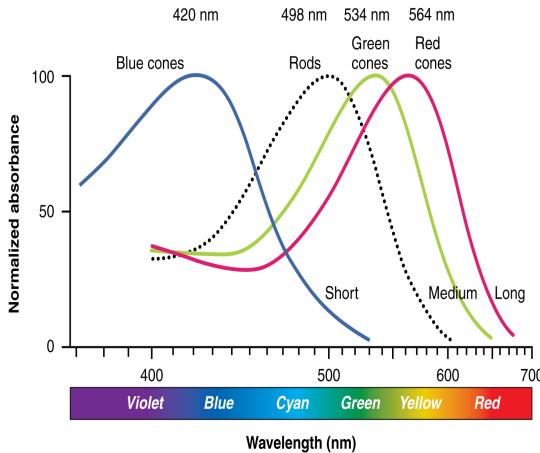
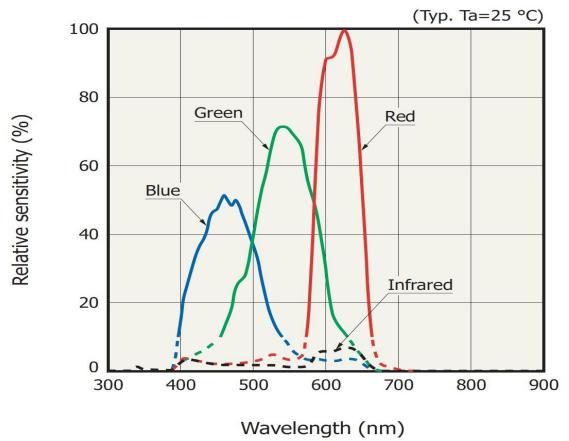


Abbildung 2.3: Ideale spektrale Antwortkurven von drei Zapfen.

es jedoch ganz anders (siehe Abb.2.4a). Jede Kamera hat eine individuelle spektrale Empfindlichkeit (siehe Abb. 2.4b). Die folgende Abbildung zeigt die spektrale Empfindlichkeit des menschlichen Auges und des Sensors in der Kamera. Es ist ersichtlich, dass die Empfindlichkeit nicht vollständig unabhängig voneinander ist und eine Verzerrung auf das Spektrum zwischen Sensoren und Auge bestehen bleibt. Im DaViD-Projekt werden die übertragenen Daten mit Smartphonekamera aus Fernsehgeräten erfasst. Die Farträume zwischen dem Fernseher und Kamera sind nicht gleich. Nach der Transformation der Farträume entsteht eine Farbdifferenz. Daher ist es nötig, die Überlagerung der Farbe und die Intensität der Empfindlichkeit jeder Farbkomponente zu korrigieren. Die RGB-Ausgabe einer Kamera ist also nur spezifisch für diese Kamera. Um konsistente Ergebnisse von allen Kameras zu erhalten, muss die Kamera zuerst in einen definierten und bekannten Farbraum umgewandelt werden. In den meisten Fällen handelt es sich dabei um den sRGB, es kann sich jedoch auch um andere Farträume handeln. Als nächstes kommt üblicherweise eine Farbkorrekturmatrixt (CCM, Color Correction Matrix) zum Einsatz. Unter Verwendung einer CCM, die durch minimale mittlere quadratische Farbfehler zwischen einem korrigierten Testchart und den entsprechenden Referenzwerten definiert ist, kann eine optimale Farbreproduktion erreicht werden. Die Korrektur der Farbe



(a) normierte Antwortkurven von RGB-Kanäle.



(b) normierte Antwortkurven von Kamera.

geschieht durch folgende Umrechnung:

$$P = \begin{bmatrix} P_{R1} & P_{G1} & P_{B1} \\ P_{R2} & P_{G2} & P_{B2} \\ \dots & & \\ P_{Rk} & P_{Gk} & P_{Bk} \end{bmatrix} = \underbrace{\begin{bmatrix} O_{R1} & O_{G1} & O_{B1} \\ O_{R2} & O_{G2} & O_{B2} \\ \dots & & \\ O_{Rk} & O_{Gk} & O_{Bk} \end{bmatrix}}_O \cdot \underbrace{\begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix}}_A \quad (2.6)$$

Die Einträge jeder Zeile in die Matrix O repräsentieren unkorrigierte Pixelwerte der normalisierten R-, G-, B-Komponenten. Die Matrix A stellt die Color Correction Matrix dar, während die Matrix P die korrigierten Pixelwerte angibt. Jeder der R-, G- und B-Pixelwerte in der Matrix P ist eine lineare Kombination der drei Eingangsfarbkomponenten, multipliziert mit der Color Correction Matrix.

2.2.4 Gradation

Die CRT-Röhren weisen die Charakteristik auf, dass die Beziehung zwischen Videosignal und Leuchtdichte nicht linear ist, sondern anhand einer Potenzfunktion mit dem Exponenten von Display-Gamma (γ_M) angenähert beschrieben wird. Der Verlauf dieser Kennlinie wird Gradation bzw. Gammaverzerrung genannt. Dies bewirkt eine dunklere Wiedergabe des Originalbildes und eine Erhöhung der Sättigung.

Die in dieser Arbeit benutzte Modulation, DaViD-Projekt, wobei das Video im Fernsehen abgespielt wird, wobei noch eine Gradation von Bildschirm besteht. Im Folgenden wird ein Beispiel zur Erläuterung des Einflusses durch Gammaverzerrung gegeben. Wenn sich ein Datenwert auf das Videoframe addiert und das gleiche Frame mit demselben Datenwert subtrahiert wird,

ergibt sich folglich eine Gradation bei beiden Ergebnissen. Es ist zu erkennen, dass die erhaltenen Ergebnisse U_1 und U_2 im Vergleich mit den linearen Ergebnissen deutlich kleiner sind. Durch das Subtrahieren der Paare von Videoframes wird die vorher eingefügte Modulationsamplitude geändert. Sie weist nun eine dunklere Wiedergabe auf.

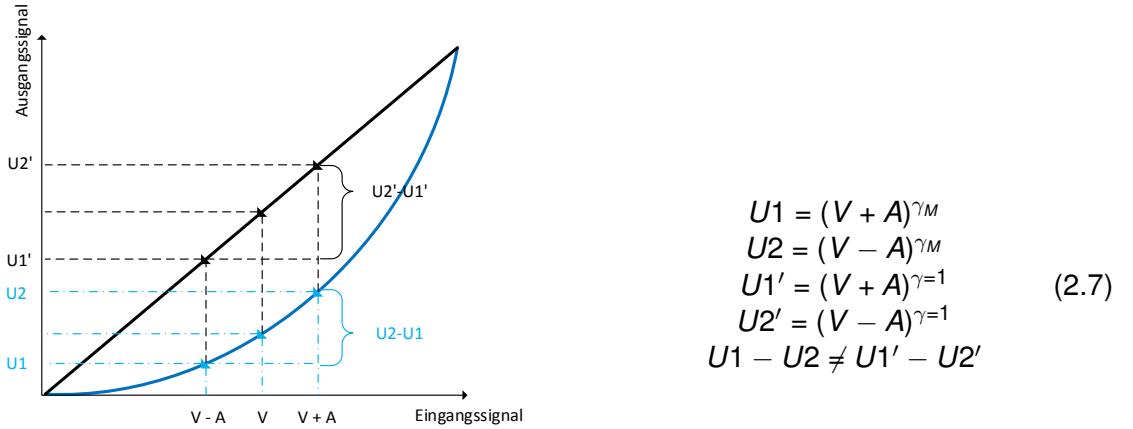


Abbildung 2.5: Beispiel zur Erklärung des Einflusses von Gradation.

Da das menschliche Auge empfindlicher gegenüber Veränderungen im dunklen Detailbereich ist und bei der Bildverarbeitung und Bildspeicherung zwischen Aufnahme- und Wiedergabeseite gibt es nur limitierte Speicherplatz und Bandbreite. Basierend auf den oben genannten Gründen werden mehr dunkle Details gespeichert. Dadurch wird erst beim Aufnahmegerät eine Gammavorverzerrung eingeführt, wodurch mehr detaillierte Strukturen im dunklen Bereich behalten werden können. Das erhaltene Bild wird heller als das Originalbild wahrgenommen (siehe Abbildung 2.7, das Bild in oben mittel). Die Abbildung 2.6 zeigt den Aufbau eines Bildübertragungssystems, wobei die Gammavorverzerrung und Gammakorrektur angewendet werden.

Um oben genannte Gammaverzerrung auszugleichen und die richtige Bilder zu rekonstruieren, wurde eine inverse Kennlinie zur Gammaverzerrung in die Aufnahmesysteme eingeführt. Dies führt zur Kompensation des nichtlinearen Zusammenhangs zwischen Videosignalpegel und entstehender Leuchtdichte bei der zur Wiedergabe eingesetzten Kathodenstrahlröhre. Die Nichtlinearität wird mit Hilfe des Exponenten von Encoding-Gamma (γ_K) angegeben. Daraus benötigt ein komplettes Bildsystem zwei Gammawerte: γ_K als Encoding-Gamma für die Gammavorverzerrung im Aufnahmesystem, γ_M als Display-Gamma für die Gamma-Korrektur im Wiedergabesystem. Für eine realistische Grautonwiedergabe steht eine korrekte Gradation

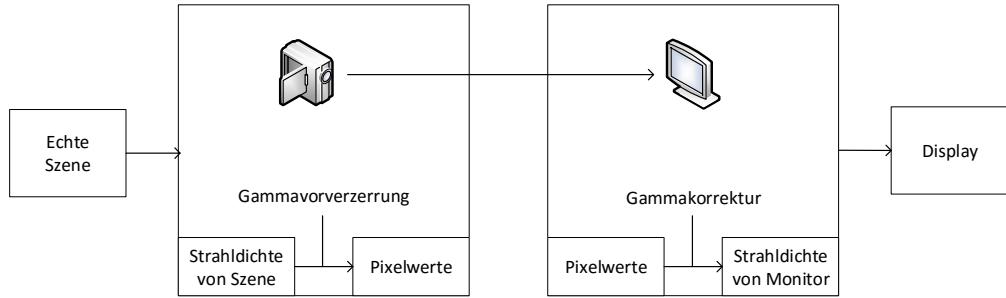


Abbildung 2.6: Gammavorverzerrung und Gammakorrektur beim Bildübertragungssystem.

des Gesamtsystems im Vordergrund. Zur Vermeidung der Farbverfälschung in den Amplituden müssen die Kennlinien der Farbwertsignale R, G und B einen ausreichenden Gleichverlauf haben. In Abbildung 2.7 ist zu erkennen, dass nach der Gamma-Korrektur in der Wiedergabeseite das Bild gleich der realen Szene wieder gegeben wird.

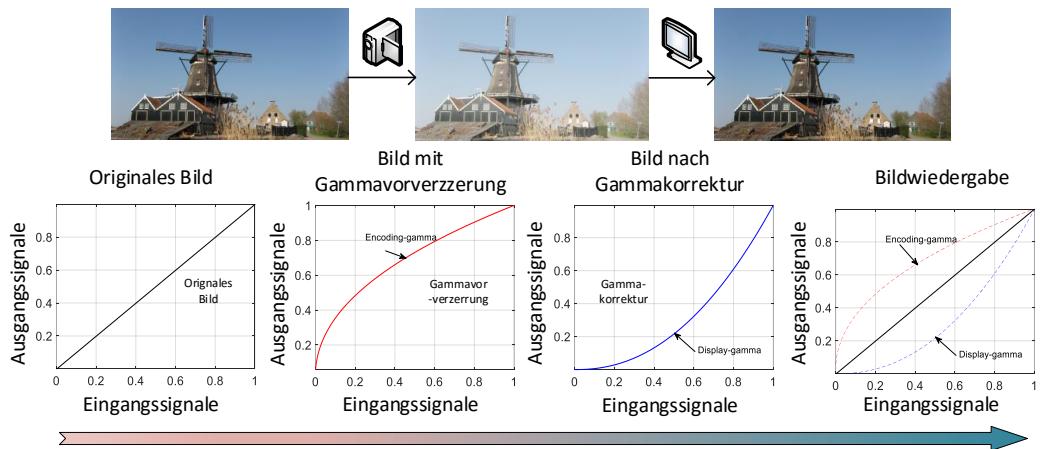


Abbildung 2.7: Gammavorverzerrung und Korrektur im Aufnahme- und Wiedergabesystem.

Abbildung 2.7 zeigt, dass im Standardsystem der Gammawert $\gamma_M = 2,2$ im Wiedergabegerät vorgegeben wird. Um eine lineare Gesamtübertragung zu erhalten, muss auf der Aufnahmeseite anschließend ein umgekehrter Wert $\gamma_M = 1/2,2 \approx 0,45$ eingeben werden. Dadurch lässt sich die Kennlinie wieder linear halten und die Bildern präzise nachbilden.

2.3 Quellencodierung für Standbilder und Bewegtbilder

Mit dem steigenden Bedarf nach besserer Qualität von Bildern und Videos wurde auf Festplatten mehr Speicherplatz und Übertragungsbandbreite benötigt. Die Quellencodierung spielt eine wichtige Rolle bei der Bild- und Videoverarbeitung, sodass eine bitsparende Beschreibung des Nachrichtensignals erzeugt wird und dabei noch gute Qualität in komprimierten Daten erhalten bleibt.

Es bestehen derzeit viele wichtige Codierungs- und Decodierungsverfahren, wie z.B. MPEG2 (Moving Picture Experts Group). Hierbei handelt es sich um ein 1994 veröffentlichtes Standard zur verlustbehafteten Video- und Audiocodierung. Obwohl MPEG-2 nicht so effizient wie das neuere Standard H.264 / AVC ist, wird es beispielsweise noch bei der Übertragung von digitalen Fernsehsendungen über Funk und im DVD-Video-Standard verwendet. Unter hoher und effizienter Videokompression bietet H.264/MPEG-4 AVC den Vorteil an, dass sich mehr Videokanäle und bessere Videoqualität über die vorhandenen Medien mit niedriger Bitrate ergeben.

2.3.1 Standbildcodierung

Nach den oben genannten Vorgängen wird nun die Quellencodierung für das Standbild durchgeführt. Das Grundprinzip der Standbildcodierung ist eine Durchführung einer blockbasierten DCT (Discrete Cosine Transform) im Frequenzbereich. Die nachfolgende Erklärung basiert auf der Intra-Codierung von MPEG-2, die unabhängig von benachbarten Bildern verarbeitet werden konnte. Eine auf DCT basierte verlustbehaftete Codierungsmethode kommt in diesem Standard zum Einsatz. Andere Codierungsverfahren (z.B. JPEG, MPEG-1 und H.264) haben ein fast ähnliches - hier vorgestelltes - Prinzip. Die Abbildung 2.8 zeigt das Kernverfahren einer Quellencodierung für Standbilder.

DCT-Codierung

Die einzelnen Schritte der DCT-Codierung in der Quellencodierung sind in Abbildung 2.8 zusammengefasst.

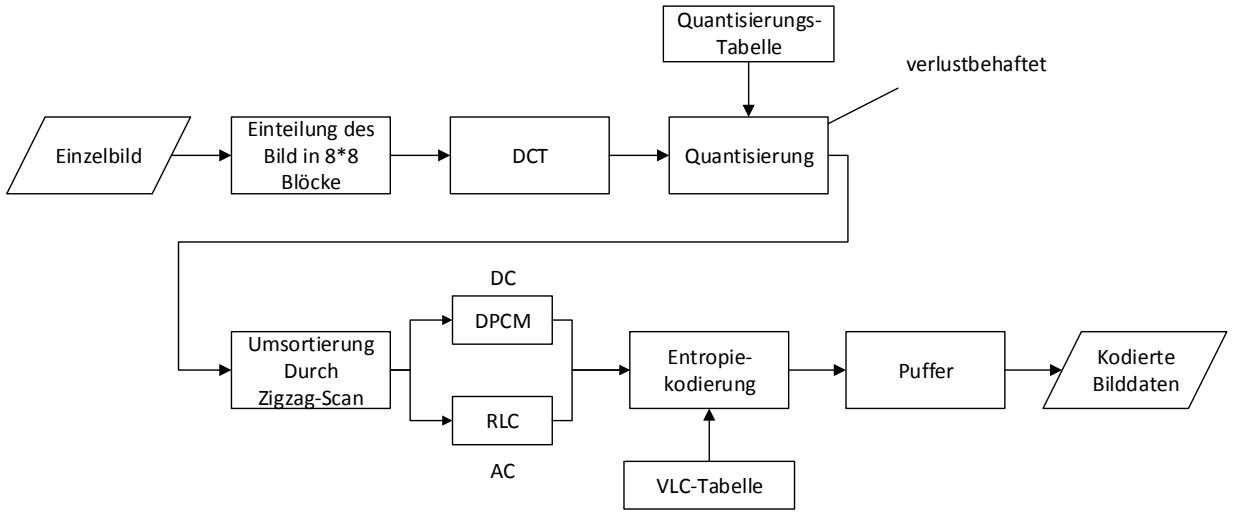


Abbildung 2.8: Blockschaltbild der DCT-Codierung.

Blockbildung

Als nächstes folgt nun die Einteilung von Bilddaten. Das Bild wird in kleine Blöcke unterteilt, die zumeist eine jeweilige Pixelgröße von 8*8 besitzen, um die Durchführung von DCT vorzubereiten. Hierbei wird die Blockzuweisung anhand eines Beispiels mit 1920 Spalten und 1080 Zeilen verdeutlicht. Durch die Unterteilung des Bildes in 8×8 kleine Blöcke werden nun 240 Blöcke in der horizontalen Richtung und 135 in der vertikalen Richtung vorhanden.

DCT (Discrete Cosine Transform)

Im nächsten Schritt beginnt die blockbasierte 2D-DCT (zweidimensionale Diskrete Kosinustransformation), welche eine Überführung der 8×8 Blöcke von Bildpunktwerten in 8×8 Koeffizienten in einem Ortsfrequenzbereich einleitet. In der digitalen Bildverarbeitung, besonders bei Farbbildern, steht die auf DCT-II basierende 2D-DCT im Vordergrund. Die mehrdimensionalen Elemente in der Bildmatrix werden zeilen- oder spaltenweise für spalten transformiert. Die folgende Funktion verdeutlicht den effizienteren Algorithmus einer 2D-DCT für Blöcke mit der Größe 8×8 . Daher ist es erfolgreich, durch diese Transformation der Bildsignale in den Frequenzbereich keine örtliche Korrelation zwischen benachbarte Bildpunkte behaltet.

$$S(p, q) = \frac{1}{4} c_p \cdot c_q \cdot \sum_{i=0}^7 \sum_{k=0}^7 s(i, k) \cdot \cos \left(p \cdot \frac{\pi}{16} \cdot (2i + 1) \right) \cdot \cos \left(q \cdot \frac{\pi}{16} \cdot (2k + 1) \right). \quad (2.8)$$

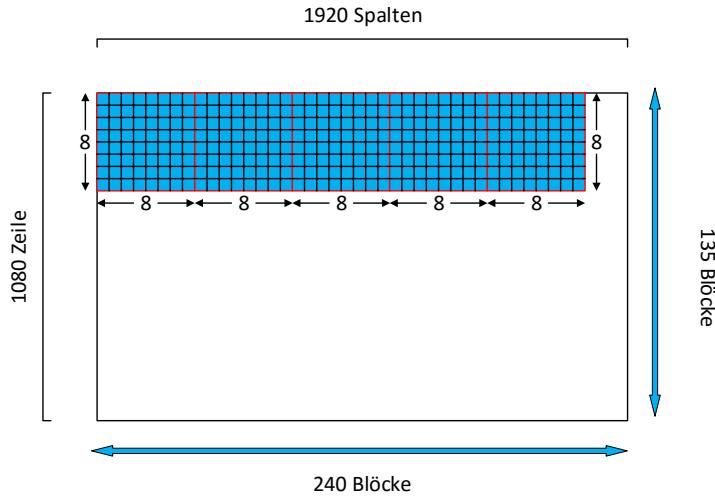


Abbildung 2.9: Beispiel einer Blockbildung.

für

$$0 \leq p \leq 7 \quad \text{und} \quad 0 \leq q \leq 7$$

mit

$$c_p, c_q = \begin{cases} 1/\sqrt{2} & \text{für } p, q = 0 \\ 1 & \text{sonst} \end{cases}$$

wobei $S(p,q)$ die erfassten DCT-Koeffizienten sind. Eine 1D-DCT wird zuerst in der vertikalen Richtung auf Spalten angewendet und dann noch einmal in der horizontalen Richtung auf Reihen.

Es können hierbei Bilddetails nach ihrer Ausdehnung und Detailhaltigkeit in entsprechend niedrig- oder höherwertige Pixelfrequenzen ausgedrückt werden. Das menschliche Auge kann niedrige Frequenzbereiche besser unterscheiden, als hohe Frequenzbereichen. Da die meisten der Hochfrequenzkomponenten im Bild relativ klein sind, liegen die DCT-Koeffizienten der entsprechenden Komponenten oft nahe zu 0. Zusätzlich enthalten die Hochfrequenzkomponenten nur feine Änderungsinformationen des Bildes. Diese Verzerrung beim menschlichen Auge ist schwer wahrnehmbar. Die Abbildung 2.10 stellt ein Basisbild nach der DCT dar. Der kleine Block in der oberen linken Ecke zeigt den DC-Anteil, welcher die Amplitude der mittleren Helligkeit aufweist. Die Werte innerhalb DC-Anteils sind größer als alle anderen Frequenzanteile (AC-Anteile). Die Energie wird im Bereich der linken oberen Ecke gesammelt. Dadurch erfolgt die Dekorrelation der benachbarten Bildpunkte.

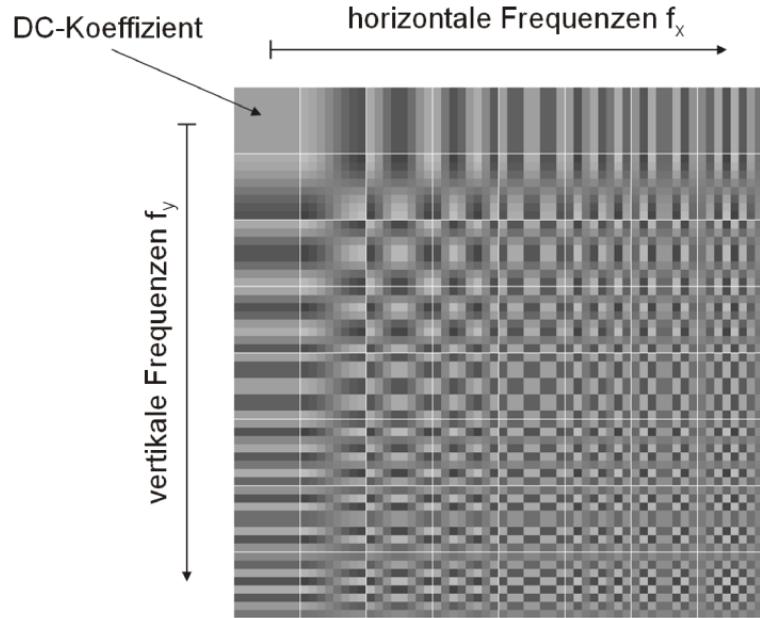


Abbildung 2.10: Basisbilder nach der DCT.

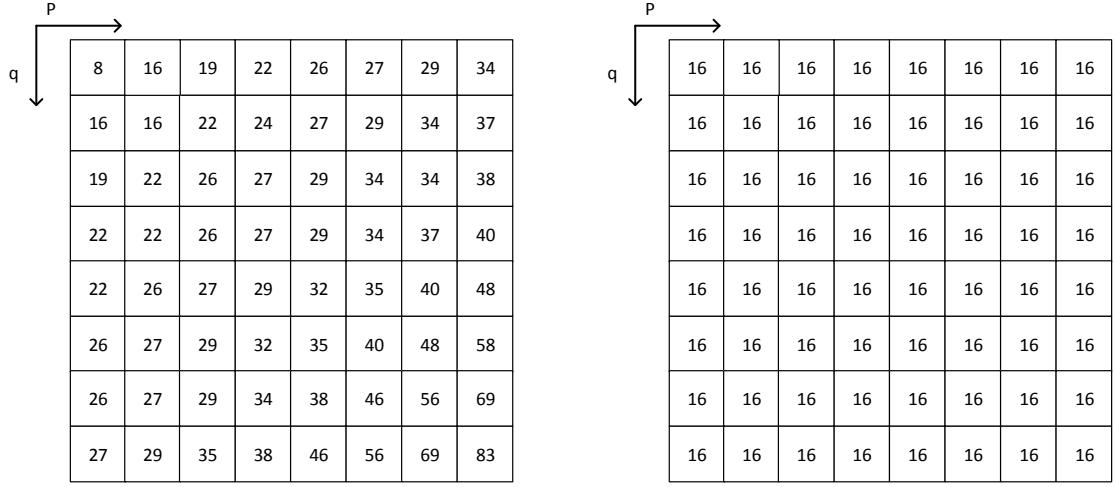
Quantisierung

Nach der DCT wird die nachfolgende Quantisierung nach der Ortsfrequenz vorgekommen. Dieser Schritt ist verlustbehaftet in der Codierung, welche den Vorteil zur Datenreduktion bietet. In diesem verlustbehafteten Prozess werden die ermittelten DCT-Koeffizienten $S_{(p,q)}$ durch die Quantisierungsmatrix $Q_{(p,q)}$ geteilt (elementweise dividiert) und das Resultat auf die nächstliegende Ganzzahl gerundet. Die Elemente in der Matrix, die zum Wert Null geworden sind, sind meistens die Hochfrequenzsignale.

$$S_q(p, q) = \text{round}\left(\frac{S_{(p,q)}}{Q_{(p,q)}}\right) \quad (2.9)$$

Die Division kommt mittels einer Quantisierungstabelle vor, welche der Farb- und Helligkeitsempfindlichkeit des menschlichen Auges entspricht. Das menschliche Auge hat die Fähigkeit, kleine Helligkeitsänderung bei relativ groben Strukturen wahrzunehmen, aber ist nicht so gut darin, die genaue Stärke einer hochfrequenten Helligkeitsschwankung anzuzeigen. Daher sind die Elemente in der Quantisierungstabelle für niedrige Frequenzen kleiner als die für hohe Frequenzen und die Koeffizienten der höheren Frequenzbereiche werden durch höhere Zahlen geteilt als bei den Niedrigen.

Die Quantisierungsmatrix ist nach Bedarf auswählbar. Erkennbar ist, dass die Elemente, die in der Diagonale bei der Quantisierungsmatrix liegen, größer als andere sind. Diese Werte wer-



	P																																																																
q	<table border="1" style="border-collapse: collapse; width: 100%; height: 100%;"> <tr><td>8</td><td>16</td><td>19</td><td>22</td><td>26</td><td>27</td><td>29</td><td>34</td></tr> <tr><td>16</td><td>16</td><td>22</td><td>24</td><td>27</td><td>29</td><td>34</td><td>37</td></tr> <tr><td>19</td><td>22</td><td>26</td><td>27</td><td>29</td><td>34</td><td>34</td><td>38</td></tr> <tr><td>22</td><td>22</td><td>26</td><td>27</td><td>29</td><td>34</td><td>37</td><td>40</td></tr> <tr><td>22</td><td>26</td><td>27</td><td>29</td><td>32</td><td>35</td><td>40</td><td>48</td></tr> <tr><td>26</td><td>27</td><td>29</td><td>32</td><td>35</td><td>40</td><td>48</td><td>58</td></tr> <tr><td>26</td><td>27</td><td>29</td><td>34</td><td>38</td><td>46</td><td>56</td><td>69</td></tr> <tr><td>27</td><td>29</td><td>35</td><td>38</td><td>46</td><td>56</td><td>69</td><td>83</td></tr> </table>	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
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																																																										

	P																																																																
q	<table border="1" style="border-collapse: collapse; width: 100%; height: 100%;"> <tr><td>16</td><td>16</td><td>16</td><td>16</td><td>16</td><td>16</td><td>16</td><td>16</td></tr> <tr><td>16</td><td>16</td><td>16</td><td>16</td><td>16</td><td>16</td><td>16</td><td>16</td></tr> <tr><td>16</td><td>16</td><td>16</td><td>16</td><td>16</td><td>16</td><td>16</td><td>16</td></tr> <tr><td>16</td><td>16</td><td>16</td><td>16</td><td>16</td><td>16</td><td>16</td><td>16</td></tr> <tr><td>16</td><td>16</td><td>16</td><td>16</td><td>16</td><td>16</td><td>16</td><td>16</td></tr> <tr><td>16</td><td>16</td><td>16</td><td>16</td><td>16</td><td>16</td><td>16</td><td>16</td></tr> <tr><td>16</td><td>16</td><td>16</td><td>16</td><td>16</td><td>16</td><td>16</td><td>16</td></tr> <tr><td>16</td><td>16</td><td>16</td><td>16</td><td>16</td><td>16</td><td>16</td><td>16</td></tr> </table>	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16
16	16	16	16	16	16	16	16																																																										
16	16	16	16	16	16	16	16																																																										
16	16	16	16	16	16	16	16																																																										
16	16	16	16	16	16	16	16																																																										
16	16	16	16	16	16	16	16																																																										
16	16	16	16	16	16	16	16																																																										
16	16	16	16	16	16	16	16																																																										
16	16	16	16	16	16	16	16																																																										

(a) für MPEG-Intrablöcke.

(b) für MPEG-Non-Intrablöcke.

Abbildung 2.11: Quantisierungstabelle für MPEG.

den bei der Quantisierung am stärksten komprimiert. Dies entspricht Oblique-Effekt, der eine wichtige Rolle bei der Bildverarbeitung spielt. Dies lässt sich dadurch begründen, dass die horizontalen und vertikalen Strukturen in Bildern empfindlicher wahrgenommen werden können als schräge (oblique) Konturen. Um die Qualität der quantisierten Bilder zu kontrollieren, ist es möglich, einen Quantisierungsfaktor QF einzufügen. Dieser kann die Quantisierungseffizienz und auch die Datenrate je nach Bedarf verändern. Damit ergeben sich die Quantisierungswerte durch:

$$S_q(p, q) = \text{round}\left(\frac{S_{(p,q)}}{Q_{(p,q)}} \cdot QF\right) \quad (2.10)$$

Die Abbildung 2.11 zeigt zwei Quantisierungstabellen: die linke Matrix (siehe 2.11a) zeigt die MPEG-Intrablöcke und die rechte Tabelle 2.11b stellt die Matrix zur Non-Intracodierung in MPEG vor. Es ist ersichtlich, dass alle Komponenten in der Non-Intrablöcke-Tabelle gleich sind. Die Non-Intracodierung bzw. Intercodierung bezieht sich auf die Codierung von Differenzbildern. Die Differenzbilder sind bereits dekorreliert und im Wesentlichen weißes Rauschen. Daher haben ihre DCTs ungefähr den gleichen Wert für alle Koeffizienten. Dies ist der Grund für die Verwendung eines konstanten Wertes in der Quantisierungsmatrix für Non-Intracodierung.

DPCM für DC-Anteile im Standbild

Wie in Abbildung 2.10 verzeichnet, steht in der oben linken Ecke die DC-Anteile, welche der größter Wert in der Matrix besitzt. Die DC- und AC-Anteile werden nun separat mit anders

Verfahren weiter verarbeitet. Im Standbild (Intraframe-Codierung) ergibt sich die räumliche Korrelation zwischen den benachbarten Pixeln. Die nebeneinander liegende 8×8 Blöcke der DC-Koeffizienten sind meistens ähnlich. Gemäß dieser Eigenschaft wird die DPCM (Differential pulse code modulation) verwendet, wodurch der Koeffizient zu Nachbarblock übernommen werden kann, was zu einer niedrigere Bitrate erzielen. DPCM ist entsprechend auch einer Prädiktion für Videosignale (Interframe-Codierung für Bewegtbildcodierung). Bei Videosignalen liegt die zeitliche Korrelation bei den gleichen Pixeln in aufeinanderfolgenden Frames. In der Interframe-Codierung wird DPCM mit dem gleichen Pixelwert in zwei aufeinanderfolgenden Frames durchgeführt. Beide DPCM-Verfahren verwendet die zuvor codierten benachbarten Pixel als Prädiktion für die nächste kommende Pixel. Die Abbildung 2.12 stellt eine typische DPCM für DC-Anteile in der Standbildcodierung. Durch DPCM werden nur die Differenz bei benachbarten Koeffizienten transformiert. Demnach liefert dieser Verfahren zugleich gute Kompressionsraten. Anhand der DPCM-Konzept wird dieser Verfahren zur Bewegtbildcodierung

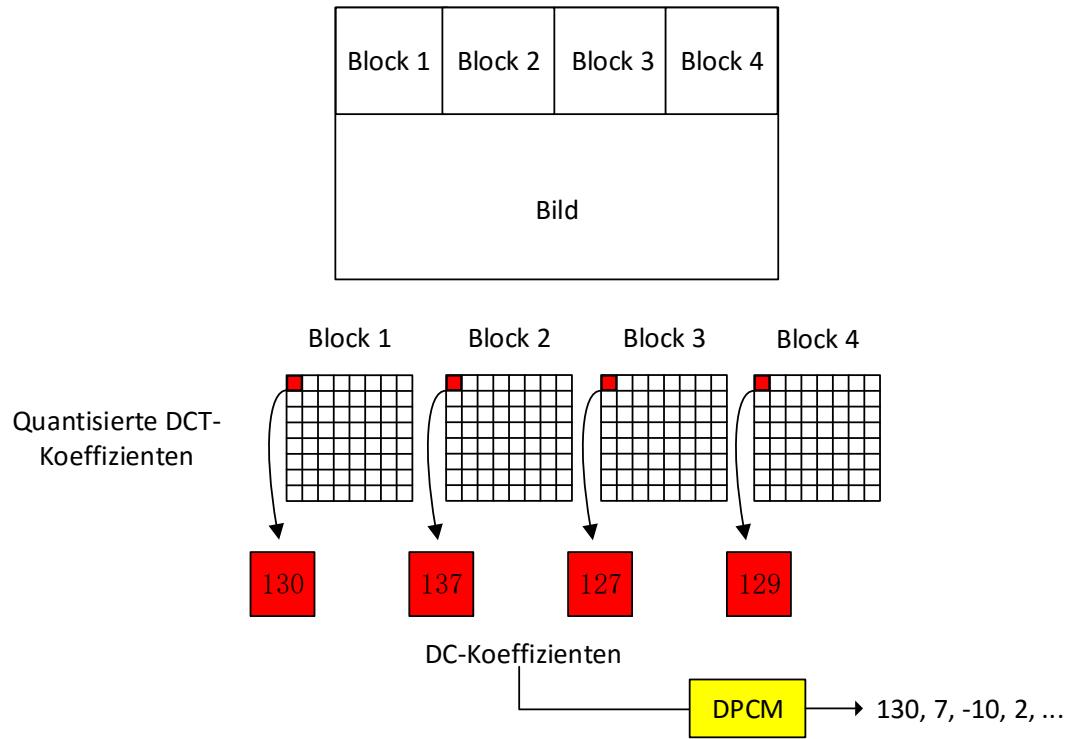


Abbildung 2.12: DPCM für DC-Anteile in der Standbildcodierung.

sinnvoll, welche unter Ausnutzung der zeitliche Korrelation der aufeinander kommende Bildern nur geringen Differenzen zur vorherigen Frames übertragen werden. Dies wird im folgenden Abschnitte von Bewegtbildcodierung mittels H.264/AVC weiter vorgestellt.

RLC für AC-Koeffizienten

Die DC-Anteile in Quantisierte Matrix sind korreliert, sodass diese Teile getrennt mittels DPCM verarbeitet werden. Für die AC-Anteile sind die Koeffizienten nach Quantisierung und Umsortierung durch Zig-Zag-Scan meistens lange Nullfolge angezeigt. Weswegen ist eine Lauflängencodierung (RLC, Run-Length Coding) sehr effizient zur Zusammenfassung viele vorhandene Nullen in der Matrix. Die Funktionsweise wird in Abbildung 2.13 verdeutlicht. Wie das

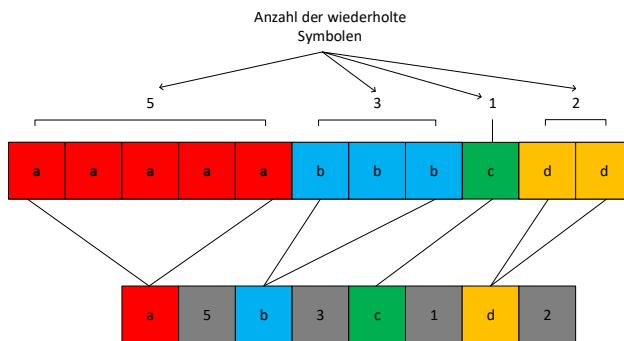


Abbildung 2.13: Beispiel der Lauflängencodierung.

Beispiel in der Abbildung gezeigt, werden die wiederholte Symbole zusammen gebündelt. Dies eignet sich auch für die Zusammenfassung vielen Nullen in der DCT-Codierung nach Quantisierung. Eine Beispieldsequenz wird anhand dieser Codierungsverfahren weiter erläutert.

...0 0 12 0 22 0 0 0 0 35 18 0 0 0 0 0 0 0 89 0 0 0 10...

Alle Nullen in der Sequenz werden mittels RLC zusammengefasst und paarweise mit der Koeffizienten gebündelt.

...(2, 12) (1, 22) (4, 35) (0, 18) (8, 89) (3, 10)...

wobei die linke Zähler in einer Klammern bezeichnet die Länge der Nullfolge und die rechte einen nachfolgenden Koeffizient. Am Ende eines Blocks wird mit einem EOB (end of block) kennzeichnet, wodurch dieser Block beendet ist. Die RLE ist auch eine verlustfrei Codierungs-methode zur Datenkompression.

Entropiecodierung

Als letzter Schritt behandelt sich eine Entropiecodierung bei der Standbildcodierung, welches einer verlustfreien Datenkomprimierung ist. Die Elemente, welche nach der Quantisierung in Hochfrequenzbereich liegen, werden fast alle zu Null sein. Danach wird die Elemente anhand

der aufsteigenden Frequenzen sortiert. Unter Verwendung von Huffman-Codierung können die Daten ohne Verlust komprimiert werden.

2.3.2 Bewegtbildcodierung mittels H.264/AVC

H.264/AVC steht für die Abkürzung von H.264 or MPEG-4 Part 10, Advanced Video Coding (MPEG-4 AVC) , die durch ITU-T Video Video Coding Experts Group (VCEG) und ISO/IEC JTC1 Moving Picture Experts Group (MPEG) zusammen entwickelt wird. Im Vergleich zum Standbildern enthalten die Bilder im Video nicht nur örtliche Korrelationsbeziehung sondern noch die Korrelation entlang der Zeitänderung. Aus diesem Grund werden die zeitliche Abhängigkeit aufeinander kommender Bildern ausgenutzt, nur die Differenzbilder zu übertragen, um die Datenmenge zu reduzieren und die Datenraten einzusparen. Daher ist dieses Standard zurzeit in Multimedia-Entwicklungsanwendungen sehr weit verbreitet.

Das Standard von H.264/AVC war erfolgreich zu schaffen, eine gute Videoqualität bei wesentlich niedrigeren Bitraten als bisherige Standards (wie z.B MPEG-1 und MPEG-2) zu liefern. D.h unter den gleichen Bildqualitätsbedingungen ist H.264/AVC das Komprimierungsverhältnis mehr als doppelt so hoch wie bei MPEG-2. Ein weiteres Ziel war es, genügend Flexibilität zu bieten, damit der Standard auf einer großen Auswahl von Anwendungen in einer Vielzahl von Netzwerken und Systemen zum Einsatz kommt, einschließlich niedrige und hohe Bitraten, Videos mit niedriger und hoher Auflösung, Broadcast, DVD-Speicher, RTP/IP packet networkse und ITU-T-Multimedia-Telefoniesysteme.

H.264-Encoder

Die verwendete Kernalgorithmen in der H.264/AVC Codierung sind Intra-und Interframe-Codierung. Hierzu werden drei Arten von Frame definiert: I Frame (so genannte das Keyframe), P Frame (Vorwärts-Prädiktion), B Frame (bidirektonaler Prädiktion).

- **I Frame:** ein Bild, das unabhängig von allen anderen Bildern codiert ist.
- **P Frame:** enthält Differenzinformation aus Bewegungskompensation, nimmt das vorherigen I-Frame oder P-Frame als Referenz zum Durchführen einer Interframe-Codierung.
- **B Frame:** stellt die höchsten Komprimierungsrate bereit. Zur Bewegungsschätzung werden nicht nur das vorherigen I- oder P-Frame, sondern auch das nachfolgendes P-Frame benötigt, um eine bidirektonaler Prädiktion durchzuführen.

Die Kombination der oben geschriebene Frame bildet zusammen eine GOP, die Abkürzung von Group of Pictures, die den Intervall zwischen zwei I-Frames vorstellt. Die Abbildung 2.14 bezeichnet ein Beispiel der GOP-Struktur. Innerhalb einer GOP ist nur ein I-Frame. Unter

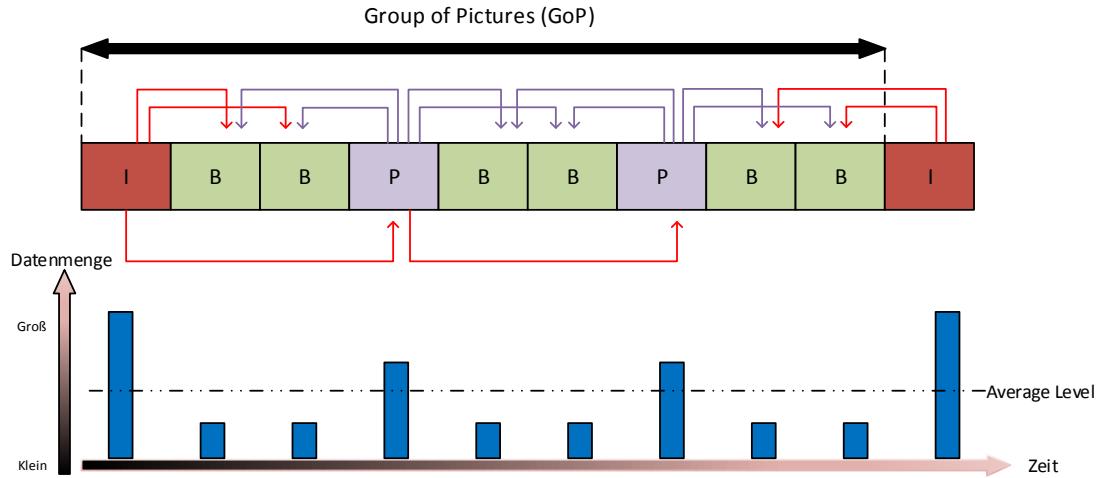


Abbildung 2.14: typische Struktur einer GOP.

der gleichen Bitrate gilt daher, je länger der GOP ist, desto größer wird Anzahl der P- und B-Frames darin und mehr Bytes belegt jedes I-, P-, B-Frame. Dies hat den Beitrag zur Realisierung einer besseren Bildqualität. Es ist nicht immer praktisch, eine höhere Qualität durch Verlängerung der GOP zu erhalten. Es gibt dazu eine Grenze, besonders in H.264-Encoder. Wenn das Szene wechselt, wird ein I-Frame automatisch eingefügt. In diesem Fall wird die tatsächliche GOP-Länge verkürzt. Andererseits werden P- und B-Frames in einer GOP aus I-Frame vorhergesagt. Des Weiteren konnte das B-Bild auch als Referenz zur Codierung anderer P- und B-Bildern in H.264/ AVC. Diese zusätzliche Flexibilität hat eine bessere Komprimierungseffizienz entwickelt, kann jedoch Fehlern bei der Transformation verursachen, wenn Daten verloren oder beschädigt werden. Sodass ist es sehr notwendig, in regelmäßigen Abständen „I-Bilder“ einzufügen und die passende GOP-Länge zu entscheiden, um Programmwechsel zu realisieren und die Zeitliche Auswirkung von Übertragungsfehler zu beseitigen.

Im neuen Version von H.264/MPEG-4 AVC ist es mehr flexibler für Referenzstruktur. Die zuvor definierte Referenzstruktur kommt auch im Einsatz. Was anders ist, dass mehr Bilder als Referenzframe und eine flexible Codierungsreihenfolge relativ zur Anzeigesreihenfolge verwendet werden können. Intraframe-Codierung stellt den Schritt zum Erzeugen von I-Frames dar. Sein Prinzip lassen sich folgendermaßen zusammenfassen: bei der Codierung eines Bildes nur die Bilddaten dieses Frames betrachtet werden, ohne die redundante Information zwischen benachbarten Frames zu berücksichtigen, da dieser Algorithmus ein vollständiges

Bild wie Standbildcodierung codiert, so dass dieses Bild unabhängig voneinander decodiert und angezeigt werden kann. Bei der Interframe-Codierung werden P- und B-Frames erzeugt. Das Prinzip ist es, die Daten mit den zwei benachbarten Frames zu vergleichen und die Differenzbildung zu generieren, um die Komprimierung weiter zu erhöhen und das Komprimierungsverhältnis zu verringern. Die Kompression wird anhand der im Abschnitt 2.3.1 erklärte DPCM-Konzept realisiert. Beim Bewegtbild kann somit nur Differenzbilder unter Ausnutzung der zeitlichen Korrelation durch DPCM generiert und übertragen. Diese Methode gehört auch zu einer Prädiktion für nachkommende Bilder. Die Implementierung der DPCM bei Bewegtbildcodierung zeigt in Abbildung 2.15. Allgemeine werden nur die berechnete Differenzen aus Prädiktion übertragen, wenn die Bilder wenige Bewegungen aufweisen. Während die Szene stark wechselt und mehr Bewegungen auftreten, erzeugt große Datenmenge hingegen. Zur

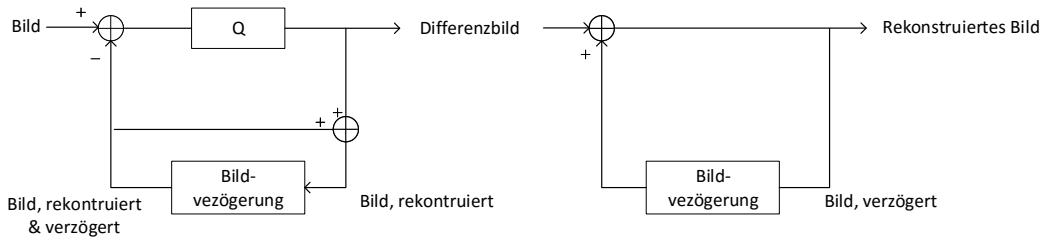


Abbildung 2.15: Blockschaltbild der DPCM für Bewegtbildcodierung.

Erstellung einer Bildvorhersage, die sehr ähnlich wie aktuelles Bild, ist erforderlich, einige Regelung zu verfolgen. Weil die Änderung zwischen vorherige und aktuellem Bild meisten durch Bewegung geschehen. Darüber hinaus wird nur die translatorische Bewegung für die erstellte Vorhersage gestattet. Somit werden die eine blockbasierte Bewegungsschätzung und Kompen-sation eingeführt. Die Verfahren gehen davon aus:

ein Bereich wird im Referenzframe (vorheriges oder kommendes Frame, das schon codiert und übertragen wird) festgelegt, um eine entsprechende $M \times N$ -Sample-Region zu find. Dies geschieht durch Vergleich der $M \times N$ Block im aktuellen Frame mit einigen oder allen der möglichen $M \times N$ Regionen in der Suchbereich, um die beste passende Bereich zu liefern. Die häufig eingehaltene Kriterium ist es ,dass die restlichen Energie durch Subtrahieren der Kandidatenregion von dem aktuellen $M \times N$ Block gebildet wird, so dass der Kandidatenregion, die die Restenergie minimiert, wird als die bestemögliche Übereinstimmung gewählt. Dieser Prozess wird als Bewegungsschätzung bezeichnet.

Die ausgewählte Kandidatenregion wird als Prädiktor für den aktuellen $M \times N$ Block und von dem aktuellen Block subtrahiert, um einen $M \times N$ Restblock zu bilden. (Bewegungskompen-sation).

Der Restblock wird nun codiert und übertragen. Der Positionoffset (Bewegungsvektor) zwischen dem aktuellen Block und Kandidatenregion wird ebenfalls auch übertragen und nicht mehr der Block selbst.

In H.264/AVC-Standard wird die Bewegungsschätzung in Makroblock durchgeführt. Der Makroblock, der einem 16×16 Pixelgröße eines Frame entspricht, ist die Grundeinheit für eine Prädiktion der Bewegung. Die Größe eines Makroblock für Quellvideo in YUV 4:2:0 Format ist 16×16 für Luminanz, und 8×8 für Chrominanz U und V (Referenz zur Abbildung 3.6). Wie vorher vorgestellt, wird eine 16×16 -Sample-Region in einem Referenzframe gesucht, das dem aktuellen Makroblock sehr nahe kommt. Die übrige Schritte gehen gleich wie oben geschriebenes aus. Das Standard von H.264/AVC erlaubt vorheriges und nachkommendes Frame als Referenz zur Bewegungsschätzung, eine so genannte bidirektionales Blockmatching (siehe Abbildung 2.16).

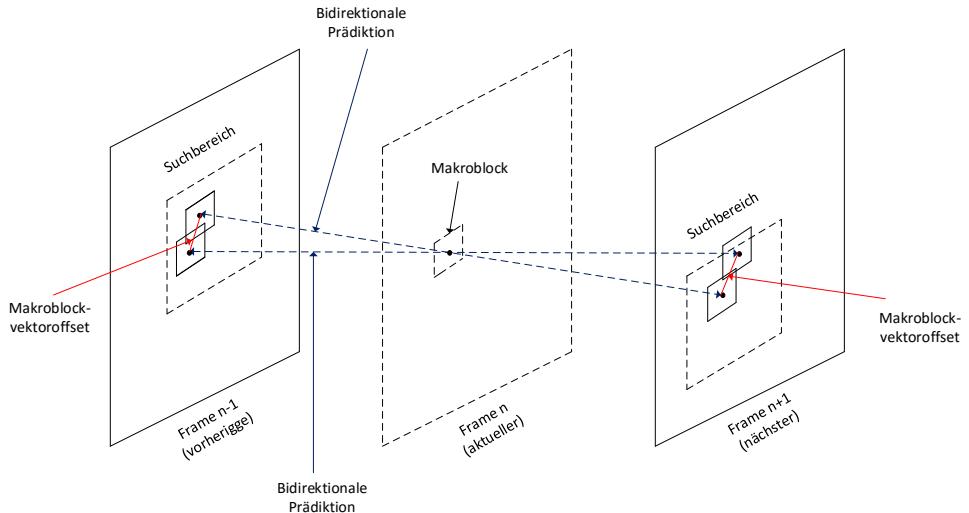


Abbildung 2.16: Prinzip des bidirektionalen Blockmatching.

Abbildung 2.17 zeigt das Blockschaltbild eines H.264/AVC-Encoder. Der Encoder hat zwei Datenflusswege: von links nach rechts ist die Bildcodierung, von rechts nach links die Bildrekonstruktion. Der Verlauf zur Bildcodierung ist wie folgt:

Die einzelne Frames werden am Anfang in den 16×16 Makroblöcke von Luminanz und Chrominanz eingeteilt. Das in dieser Arbeit behandelte Bilddaten werden nach dem Norm ITU-R BT.601 in Form Y:U:V 4:2:0 getrennt und weiter unterabgetastet. Die Farbdifferenzsignale bzw. Chrominaz werden jedoch unterabgetastet und in den 8×8 Makroblöcke unterteilt. In der Mitte wird ein Schalter eingebettet, um den Modus festzustellen, ob reine Intra-Bilder oder Inter-Bilder gesendet werden. Wenn das Szenen mehr als vorheriges Bild wechselt oder die

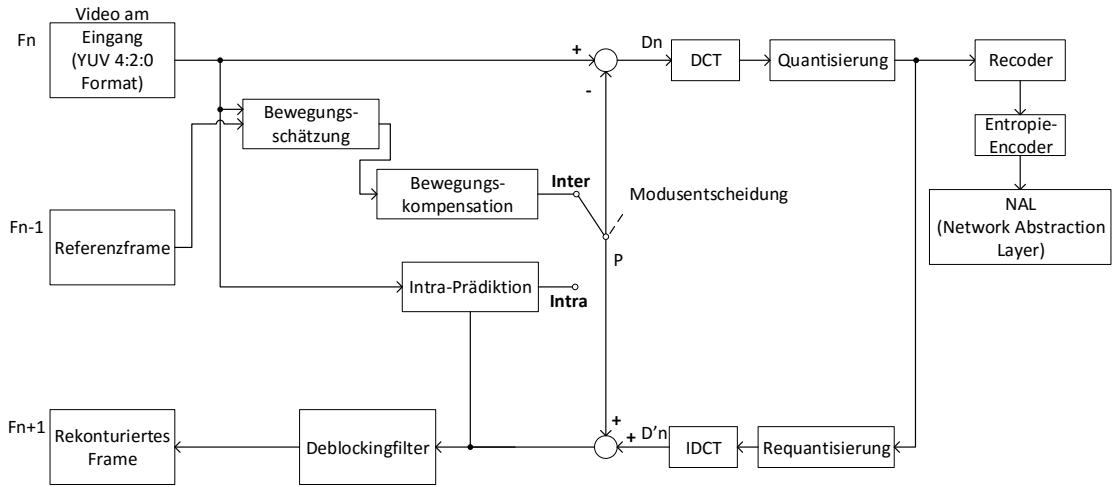


Abbildung 2.17: H.264/MPEG-4 AVC-Encoder.

Resultaten des Blockmatchings sehr schlecht sind, wird die Intraframe-Codierung (I-Frame) durchgeführt. Wenn der Schalter nach oben zieht, wird Blockmatching verwendet, die ein Suchverfahren bezeichnet, um die Bewegungsvektoren zu finden. In der Bewegungsschätzung wird ein bestmögliche übereinstimmte Bereich aus Referenzframe gesucht, der ähnlich wie im aktuellen Makroblock ist. Der Positionsoffset im gefundene Bereich aus aktuellen und vorherigen Makroblock ist der Bewegungsvektor. Somit entsteht die Prädiktion mit Bewegungskompensation P . Als nächster wird der Differenzblock D_n mittels Subtrahierung der aktuelle Makroblock vom erzeugte P erhalten. Dieser D_n wird weiter in den 8×8 oder 4×4 Subblöcke unterteilt und durch die DCT gut quantisiert. Nach der Quantisierung werden die DCT-Koeffizienten anhand einem ZigZag-Scan umsortiert und anschließend fängt die Lauflängenkodierung an. Die codierte Koeffizienten und verwandte Headerinformationen von Makroblock mit den Bewegungsvektoren werden durch Entropie weiter codiert (verlustfrei) und ins NAL (Network Abstraction Layer) geschrieben. Der Rekonstruktionsdatenfluss beschreibt wie folgt: Jeder quantisierte Makroblock wird requantisiert. Nach einer inverse DCT entsteht ein decodierte Differenzmakroblock D'_n . Der Quantisierungsprozess ist immer irreversibel und infolgedessen ist dieser D'_n nicht gleich wie der ursprüngliche D_n (verlustbehaftet). Das Frame n (F_n) wird durch Addition von Differenzmakroblock D'_n und Prädiktion mit Bewegungskompensation P wieder rekonstruiert. Dieses Frame wirkt als Referenzframe zur Codierung für das folgende Frame, nämlich F_{n+1} .

2.4 Qualitätsbewertung der übertragene Daten

Zur Bewertung der Bildqualität werden zwei Methoden in Anspruch genommen, die objektive und subjektive Bewertung. Bei subjektive Test werden hauptsächlich die Bewertungen durch Wahrnehmung der Bildern beim menschlichen Auge gegeben. In dieser Arbeit wird die subjektive Bewertung dadurch realisiert, dass die Betrachtung zwischen unkomprimierte- und komprimierte Datenblöcke, die innerhalb eines Bild im unterschiedlicher Bereiche ausgewählt werden, um die detaillierten Änderungen vor- und nach der Verarbeitung deutlich zu betrachten. Die Bereiche in einem Bild lassen sich so festlegen, dass einige kleine Bildblöcke jeweils aus dunklen, helleren, schnell und langsam bewegten Bereich extrahiert werden. Die objektive Methode wird durch die mathematische Algorithmen repräsentiert, wie z.B die Berechnung von PSNR (Peak-Signal-to-Noise-Ratio), der Vergleich von BER (Bit Error Ratio) je nach verschiedene Modulationsamplitude und Datenblocksize zwischen originale und komprimierte Daten und die Verteilungsänderung der Daten vor und nach der Verarbeitung.

PSNR wird am häufigsten bei verlustbehafteter Bildkompression verwendet, um die Qualität des Bild nach der Verarbeitung zu bewerten. Beim Vergleich von Kompressionsverfahren ist PSNR eine Approximation näher zur menschliche Wahrnehmung für Rekonstruktionsqualität. Die Berechnung von PSNR von Bild wird so berechnet:

$$MSE(D, n) = \frac{1}{b \cdot h} \sum_{x=1}^b \sum_{y=1}^h (D_c(x, y, n) - D_{org}(x, y, n))^2 \quad (2.11)$$

$$PSNR(D, n)_{frame} = 10 \log_{10} \left(\frac{(255)^2}{MSE(D, n)} \right) \quad (2.12)$$

Die PSNR von Daten ist somit errechnet:

$$PSNR(D, n)_{Daten} = 10 \log_{10} \left(\frac{(255 \cdot 2)^2}{MSE(D, n)} \right) \quad (2.13)$$

wobei, MSE steht für die Mittlere quadratische Abweichung, b und h sind die Spalte und Zeile von Daten, n die Dimension der Frame oder Daten. Weil die Werte von Daten den Bereich von -255 bis +255 ([-255, 255]) umfassen, wird die möglichst maximale Pixelwert davon $255 \times 2 = 510$

Eine andere Maßnahme ist BER. Die Grundprinzip beschreibt wie folgend, dass die Anzahl der falschen Bits durch gesamte übertragene Bits innerhalb der Untersuchungszeit geteilt wird. Der Ergebnisse ist die Bitfehlerrate (BER), die oft in Prozent dargestellt wird. N_e bezeichnet Gesamtanzahl der Fehlerbits und N_{total} zeigt alle im Kanal in gleicher Zeit übertragene Bits.

Die mathematische Darstellung der BER ergibt sich zu

$$BER = \frac{N_e}{N_{total}} \times 100\% \quad (2.14)$$

Nach Quellencodierung der Daten werden die Pixelwerte geändert. Daraus trifft Differenz bei Nachbildung der Daten. Weswegen werden die Daten aus verarbeitetem Video ausgenommen und vergleichen mit den originale Daten je nach vorgegebene Blockgröße. Der Differenz wird geteilt durch die uncodierte Daten und somit kann die Rekonstruktionsfehler der Daten durch absolute Mean Error berechnet. Die Berechnung der Fehler ist wie folgende definiert:

$$\text{Rekonstruktionsfehler} = \sum_{i=1}^N \sum_{k=1}^M \left(\sum_{x=1}^{BS} \sum_{y=1}^{BS} \left| \left(Data_{cod}(x, y, n) - Data_{org}(x, y, n) \right) / Data_{org}(x, y, n) \right| \right) \quad (2.15)$$

mit

$$\begin{cases} N = \text{Length}/\text{Blocksize} \\ M = \text{Width}/\text{Blocksize} \end{cases}$$

wobei die BS steht für die Blockgröße von Daten. Die Abbildung 2.18 zeigt schematische Aufbau einer objektiven Qualitätsbewertung für Bildern, die sich auf dieser Arbeit bezieht. Die

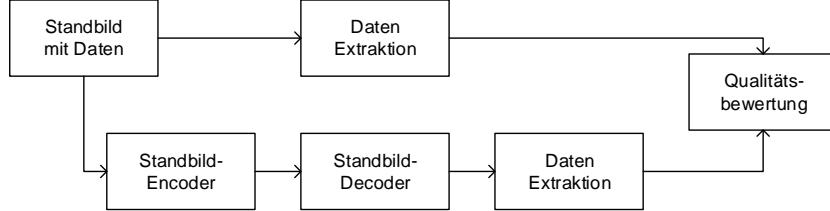


Abbildung 2.18: schematische Aufbau einer objektiven Qualitätsbewertung für übertragene Daten.

Bewertung läuft wie folgend, dass die originale Daten aus Standbild vor der Quellencodierung extrahiert werden und als Referenz zum Vergleich mit codiere Daten dienen. Die untere Encoder und Decoder werden zur Kompression des Video und Daten verwendet.

2.5 Systembeschreibung von DaViD

DaViD ist die Abkürzung von Data transmission using Video devices. Das Projekt von DaViD bezieht sich auf der Datenübertragung, wobei mit der Modulationsamplitude angesteuert wird. Die mit Modulationsamplitude multiplizierte Daten werden individuelle auf der R-, G- und

B-Kanal addiert. Die Bildinhalt wird im Fernseher wiedergegeben. Die Gestaltung dieses Projekts beruht sich darauf, die bearbeitete Daten ziemlich unsichtbar für die Betrachter. Daraus kommt die digitale Kamera mit hohe Auflösung für die Aufnahme zum Einsatz.

In dieser Arbeit wird nur die zeitliche-differentielle Modulation erläutert. Aufgrund der schwächeren Wahrnehmung von Farbdifferenz bei menschlichen Augen ist es möglich, die Modulation in der Chrominanz durchzuführen, sodass eine geringer Empfindlichkeit bei starker Quantisierung ermöglicht und geringere Anforderung für Kameraauflösung.

Modulationsmethode in DaViD

Die Abbildung 2.19 zeigt die komplette Übertragungsstrecke von DaViD. Die Arbeitsverlauf beschreibt wie folgend: Jedes Datenstrom sollte am Anfang in Pakete $d(l) \in -1, +1$ mit der

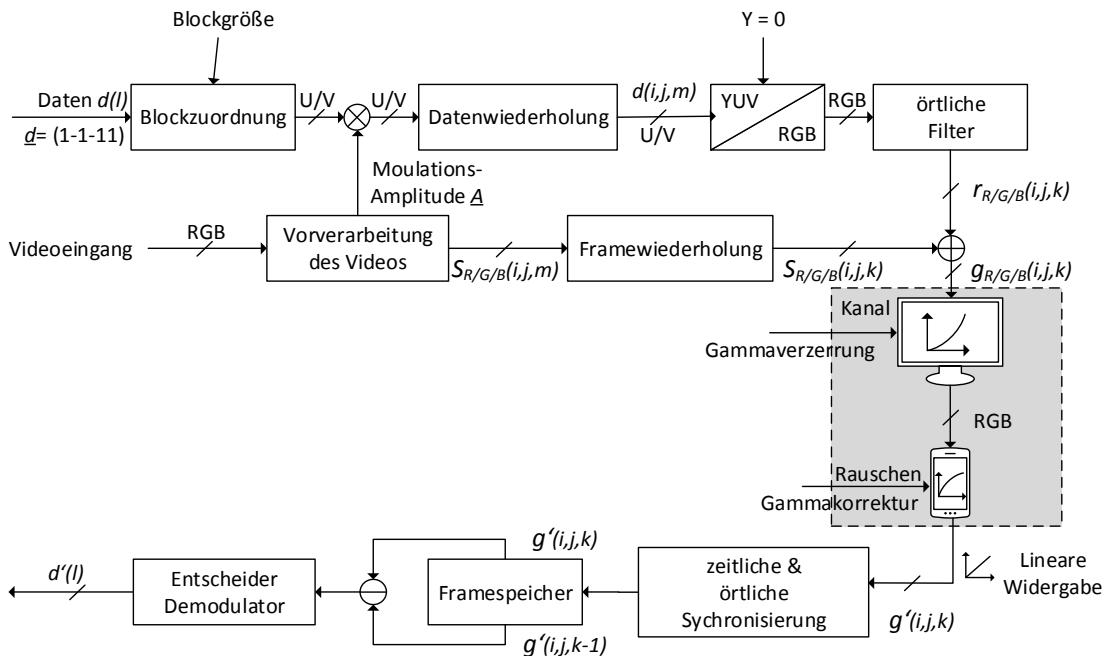


Abbildung 2.19: schematische Aufbau von DaViD-System.

maximale Länge von L pro Frame eingesetzt. Jedes Frame enthält Anzahl von m Datenpakete. Zusätzlich besitzt die Größe jedes Datenblocks von $B_x \times B_y$ Pixeln. Danach werden die Daten

in der Eingabe an den Bildschirmgröße $N_x \times N_y$ zugeordnet.

$$\begin{aligned}
 d(l) &\Rightarrow d(i, j, m) \quad 0 \leq l \leq L \\
 L &= \left\lfloor \frac{N_x}{B_x} \right\rfloor \cdot \left\lfloor \frac{N_y}{B_y} \right\rfloor \quad (2.16) \\
 i &= (l \cdot B_x) \bmod N_x + r_x, \quad r_x = 0 \dots (B_x - 1) \\
 j &= \left\lfloor l / \left\lfloor \frac{N_x}{B_x} \right\rfloor \right\rfloor \cdot B_y + r_y, \quad r_y = 0 \dots (B_y - 1)
 \end{aligned}$$

Als nächst wiederholt das originales Videoframe einmal, um ein Frame mit der positiven und das andere mit negative Modulationsamplitude zu multiplizieren. Dies entspricht der zeitlich-differentielle Modulationsverfahren.

$$s_{R/G/B}(i, j, k+1) = s_{R/G/B}(i, j, k), \quad k = 2m, m \in \mathbb{Z} \quad (2.17)$$

Aufgrund der geringe Wahrnehmung von Farbdifferenz bei menschlichen Augen wird auf dieser Stelle die Daten nur in Chrominanzzkanäle eingefügt während die Modulation der Helligkeit in Datenkanal unverändert bleibt. Eine Transformation des Farbraums wird hier durchgeführt, um die Daten auf die Videoframes zu addieren, ein Frame mit positiver und ein mit negativer Modulationsamplitude. Dies entspricht der Modulationsverfahren in Chorminanz in DaViD. Die Implementierung der Farbraumumstellung T^{-1} lässt sich im Abschnitt 2.1.3 verdeutlichen.

$$\begin{aligned}
 g_{R/G/B}(i, j, k) &= s_{R/G/B}(i, j, m) + T_{YUV \rightarrow RGB, BT.709} \cdot d(i, j, m) \cdot A \\
 g_{R/G/B}(i, j, k+1) &= s_{R/G/B}(i, j, m) - T_{YUV \rightarrow RGB, BT.709} \cdot d(i, j, m) \cdot A
 \end{aligned} \quad (2.18)$$

wobei für die zeitlich-differentiell Modulation gilt hier $T = T^{-1}$ und für Modulation von Luminanz in RGB speziell $A_R = A_G = A_B = A$. T^{-1} stellt die Farbraumtransformation dar. Auf dieser Stelle wird die Transformation der Daten mit Matrix T laut BT.709 Standard von YUV-Farbraum in den RGB-Farbraum umgestellt. Daraus erfolgt die Addition der Daten auf Videoframes.

Demodulationsmethode

Die modulierte Bildsignale werden nun bei dem Bildschirm wiedergegeben. Diese Signale g' ist jedoch gammaverzerrt (zur Erklärung siehe Abschnitt 2.2.4). Idealerweise sind die übertragene Daten bei Abspielen des Videos für die Betrachter nicht wahrnehmbar. Die Bildsignale werden im Übertragungskanal verrauscht, verzerrt, gefiltert und abgetastet. Danach folgt eine zeitliche Synchronisierung und die Datenwerte werden durch Subtrahieren vom zwei Frames eines Paare wiederhergestellt.

3 Simulation des Einflusses der Quellencodierung auf ideale Datensignale

Im diesem Kapitel wird eine Simulation durchgeführt, um die Einflüsse der Quellencodierung auf verdeckt übertragene Bildinhalt bei der Screen-Camera VLC zu untersuchen und daraus entsprechende beeinflusste Faktoren festzulegen. Nach der Simulation und Bewertung der Ergebnisse werden geeignete Modulationsamplitude und Datenblockgröße erfasst, die weiter zur Durchführung und Bewertung der experimentelle Maßnahmen eingesetzt werden könnten.

3.1 Einfluss vor der Quellencodierung

Bei der Erzeugung eines in Modulation benutztes Video werden noch ein paar Einflüsse auf der Bildinhalt und Daten vor Quellencodierung eingefügt. Um das System Nähe realen Szene zu modulieren, sind solche Bewirkungen davor zu berücksichtigen.

Die folgende Erklärung sind alle Schritte, die zur Erzeugung der zu modulierende Videoframes dienen. Das Einfügen dieser Schritte in der Simulation hat den Vorteil, die Ergebnisse präzis nahe zur reale Szene und weiter eine gute Referenz im Vergleich mit Kapital 4 implementierte Experimente.

Als ersten wird ein Bild (in .bmp Format, siehe Abbildung 3.1) im Matlab zu JPEG-Format geschrieben, um zu betrachten, wie die vorgegebene Parameters auf der Bildrekonstruktion nach JPEG-Codierung beeinflusst, ohne Rauschen und die im Kapitel 2 vorgestellte Korrektur. Wie in Abbildung 3.2 zeigt, wird ein zufälliges Data erzeugt und individuell durch ein Mapper in U und V-Kanal geschrieben. Dies ist entsprechend der vorher im Abschnitt 2.1.2 erläuterten Prinzip, dass menschliches Auge die Änderung der Helligkeit viel intensiver als Farbe wahrnimmt. Deswegen wird keine Daten in das Y-Kanal eingefügt. Es folgt dazu noch eine Blockweisung, wodurch das Blockgröße ausgewählt werden kann. Die erzeugte Daten wiederholt einmal, um eine umgekehrt Datenwerte zu generieren. Zum nächsten fängt ein Transformation von YUV-Kanäle in den RGB-Kanäle an, um die Daten auf das originales Bild



Abbildung 3.1: zu codierendes Bild .

zu addieren. Am Ende wird mit Hilfe der Matlabfunktion „*imwrite*“ je nach Variierung der Modulationsamplitude, Datenblockgröße.

Die erste Simulation wird anhand Codierung für Einzelbild implementiert, die sogenannte JPEG-Codierung, um zu betrachten, den Einfluss der Quellencodierung auf einzelnes Bild. Bei der Simulation werden die Modulationsamplitude, Datenblockgröße und Qualität zum Auswahl eingesetzt. Hier steht die Qualität in JPEG-Codierung für die Kompressionsration in Standbildcodierung, ein Skalar im Bereich [0,100], wobei 0 eine niedrigere Qualität bezeichnet und dazu die höchste Komprimierung eingeführt wird, dagegen 100 eine höhere Qualität erfasst, aber die niedrigste Kompression aufweist. Zur Bewertung des Einflusses gibt sich dort eine Rekonstruktionsfehler an, sodass die codierte Datenwerte mit originale Werte vergleichen kann. Diese Simulation wird über 20 mal das ganzes System laufen lassen, um die Verlauf der Rekonstruktionsfehler von Daten weitgehend zu glätten. Zur Durchführung der Simulation kommen die folgend Parameters zum Einsatz:

- **Modulationsamplitude (Abkürzung: A):** die Modulationsamplitude beeinflusst die Intensität der Eingangsdaten. Die Amplitude variiert von 4 bis 15.
- **Datenblocksize (Abkürzung: BS):** Die Größe der in jedes Bild eingefügte Datenblock. Das Datenblock wird in Größe 4, 5, 6, 8, 10, 12 eingegeben.
- **Bildqualität (Abkürzung: Q):** steuert die Kompressionsration bei der Bildverarbeitung an.
- **Bitrate:** die Anzahl der pro Zeiteinheit übertragenen oder verarbeiteten Bits

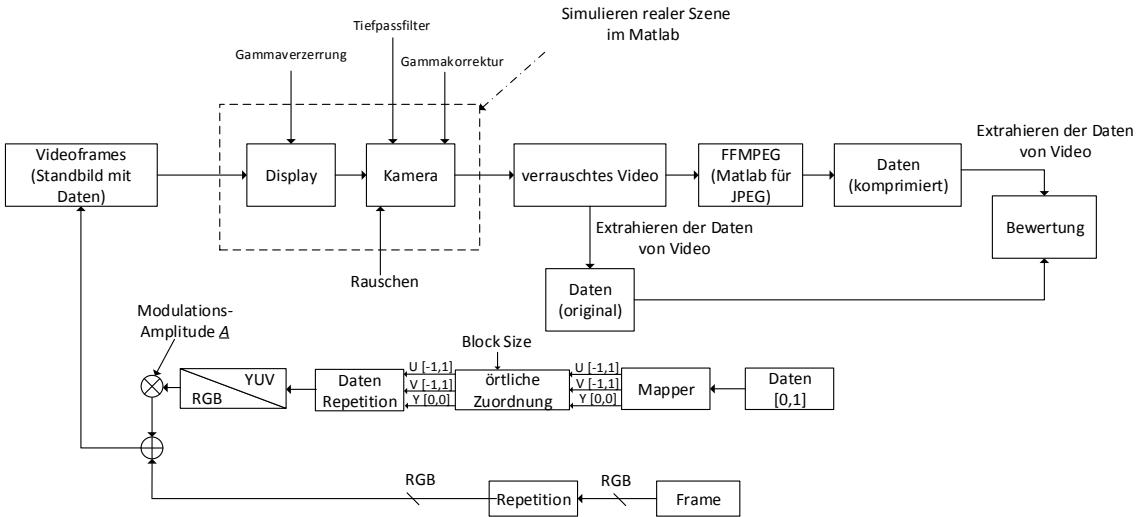


Abbildung 3.2: Simulationsablauf.

Die Abbildung 3.3 zeigt die Ergebnisse von JPEG-Codierung in U-Kanal unter 4 verschiedene Qualität ($Q = 30$, $Q = 50$, $Q = 80$ und $Q = 100$). Der hauptsächlicher Verlauf des Rekonstruktionsfehlers stellt wie folgend dar, dass sich der Fehler mit erhöhter A und vergrößertem BS verringert. Wenn BS von 3 bis 8 vergrößert, ist der Fehler schnell gefallen, insbesondere BS von 3 bis 4, stürzt der Fehler um fast 80% ab, während sich der Fehler bei $BS = 3$ sehr hoch, besonders bei $BS = 3$ und $A = 4$. Ab $BS = 8$, reduziert sich der Fehler langsam. Aber wenn Q unter 20 ist, macht die Kompression kein Sinn. Die Datei habe nur wenig Bits (c.a 37KB bis 73KB) zum Speicher und weswegen sind die meisten Hochfrequenzinformation unterdrückt und resultiert ein sehr unscharfes nachgebildetes Bild. Die Abbildung der Rekonstruktionsfehler für V-Kanal stellt im Anhang dar. Der durchschnittlicher Fehler in U-Kanal unter der gleicher Bedingung, nämlich mit gleicher BS , A und Q geringer als der in V-Kanal. Zur Bewertung der Bildqualität kommt PSNR zum Einsatz. Die Abbildung 3.4 stellt die PSNR in vier verschiedene Qualität dar. Mit der abnehmende A und vergrößerte BS steigt die PSNR. Wenn sich die Amplitude erhöht, steigt die MSE von Differenzbild. Laut der Gleichung von PSNR (siehe Abschnitt 2.4) verursacht dies eine verschlechterte PSNR. Wenn sich BS von 3 bis 4 und 5 bis 6 unter $Q = 100$ erhöht, dass sich PSNR schnell verbessert, Ab $BS \leq 8$ zieht PSNR schwach an. Die PSNR in V-Kanal hat die ähnliche Verhalten wie in U-Kanal aber einige besser. Die Datenübertragung in DaViD beruht sich auf das Abspiel eines Videos. Wie Abbilding 3.2 gezeigt, kommt in der nächsten Simulaiton ein Video zum Einsatz. Um das reale Wiedergabesystem zu nähern und die Einflüsse weiter zu untersuchen, werden bei der ursprünglichen Videoframes eine Gammaverzerrung und Rauschen eingerührt. Danach geht die verrauschte Signale eine Tiefpassfilter durch. Die entstehende Videoframes werden in FFMPEG (siehe Abschnitt 3.2.2) anhand Videocodec bearbeitet aber vor der Bearbeitung werden die Daten davon zu-

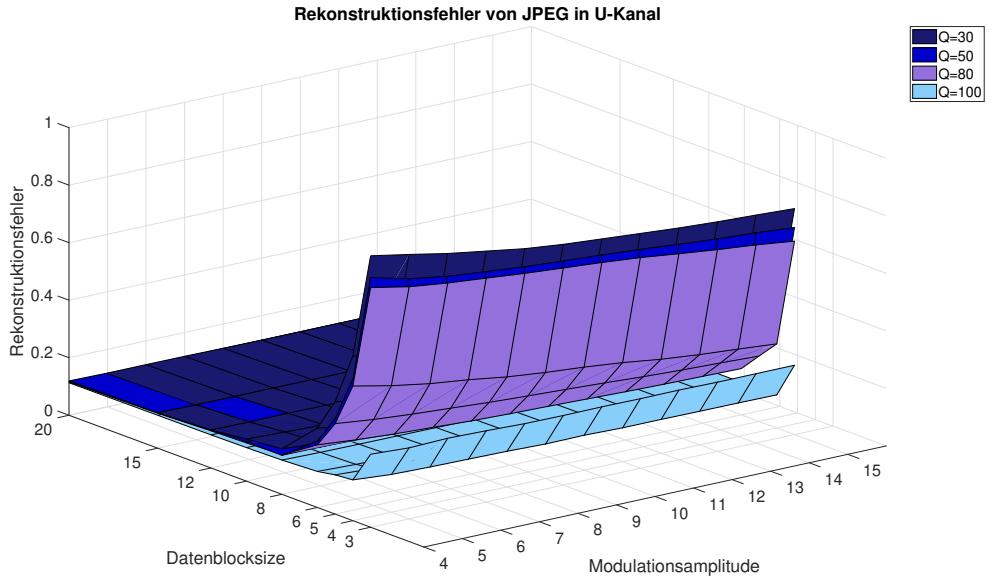


Abbildung 3.3: Rekonstruktionsfehler von JPEG-Codierung in U-Kanal.

erst extrahiert, die später als Referenz zum Vergleich mit komprimierten Daten. Die Daten sollten auch von dem durch FFmpeg behandelte Video herausgezogen, um die Bewertung der beide Daten durchzuführen (siehe Abbildung 3.2).

Gammaverzerrung

Die Abbildung 2.5 zeigt einige Pixelwerte vor- und nach der Verzerrung. Wie im Abschnitt 2.2.4 vorgestellt, ändert die Pixelwerte relativ langsam im dunklen Bereich, während im hellen Bereich mehr schneller. D.h. auf Grund der Nichtlinearität zwischen Videosignal und Leuchtdichte führt dieses Prozess zu große Differenz nach der Gammaverzerrung. Abbildung 3.5 stellt die Änderung der Dataframes ohne und mit Gammaverzerrung dar. Außerdem ist es erkennbar, dass nach Gammaverzerrung noch ein paar Bildinhalte beim Datenframe bleibt und nicht die gleiche Datei wie vorher aussieht. Das Grund liegt darin, dass die Verzerrung im Wiedergabesystem zu einer höhere Sättigung und Änderung der Modulationsamplitude führt.

Gaußtiefpassfilter

Nach Gaußfilterung wird das Bild glättet, besonders bei der Seitenkante von Datenblöcke. Der Grad von Glättung eines Bildes hängt von der Standardabweichung des Gaußfilters ab. Das

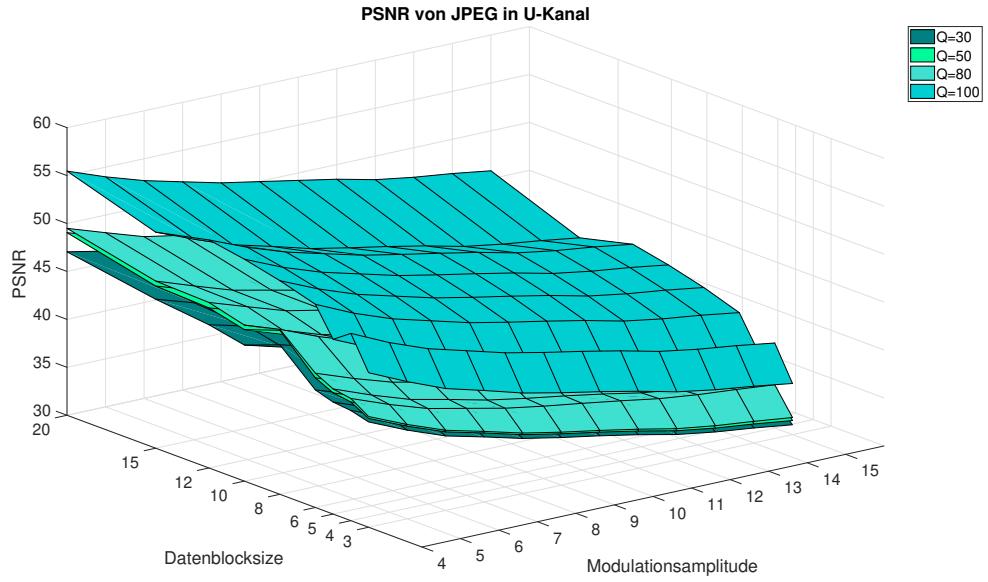


Abbildung 3.4: PSNR von JPEG-Codierung in U-Kanal.

Rauschen bleibt meistens in der Hochfrequenz. Daher lässt sich ein groÙe Menge des Rauschens gefiltert. Es bewirkt jedoch eine unschöne Glättungseffekt auf der Datenblockkante, besonders bei der Kante nebeneinander Datenblöcke sehr beträchtlich.

Farbraumumrechnung

Das erzeugtes Eingangssignal vor der Quellencodierung wird zuerst in den Zielraum, bzw. in YUV-Farbraum umgewandelt. Dadurch lassen sich die Helligkeitsinformation und Farbdifferenzsignale voneinander trennen. Wie im 2.1 schon erwähnt, bietet die Umrechnung danach für die Verarbeitung der Bildern den Vorteil, in relativ unabhängige Kanälen zu komprimieren und codieren.

Farbunterabtastung

Weil der Ortsauflösung des menschlichen Auges für Farben deutlicher geringer als für Helligkeitsinformation ist, beträgt dies zur Reduktion der Datenmenge. Die detailliert Erklärung ist schon im Abschnitt 2.2.2 erklärt. In MPEG-2 und H.264 können noch mehrere Profile nach selber Bedingung entschieden werden. Die in Abbildung 3.6 gezeigte Profile stehen bei der

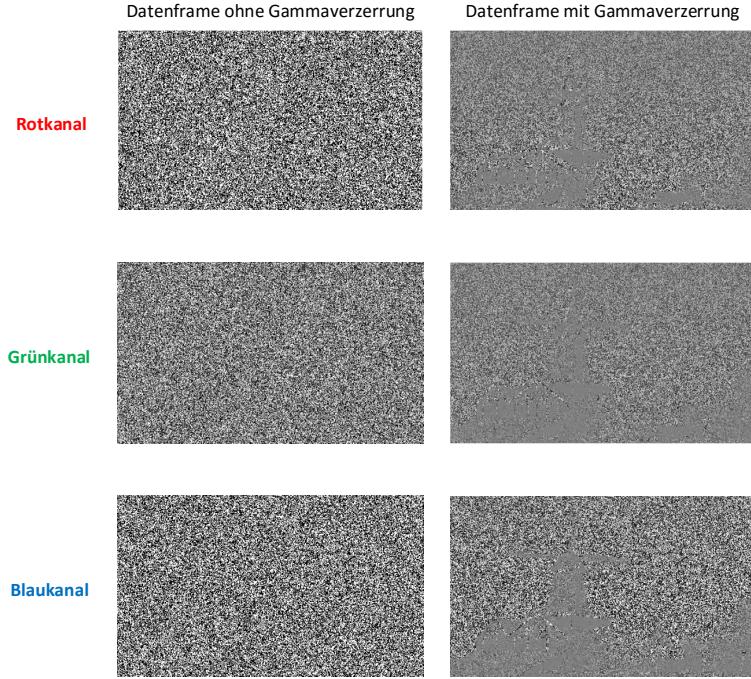


Abbildung 3.5: Dataframe mit und ohne Gammaverzerrung in jedem Kanal.

Farbunterabtastung für Anordnung verschiedene Makroblock zur Verfügung. Anhand Ausnutzung von YUV 4: 2:0 nach ITU-R BT.601, bzw. die Farbdifferenzsignale nur in jeder zweiten Zeile vorliegen, wird die Damenvolumen um Faktor 4 reduziert.

3.2 Einfluss der Quellencodierung für Standbilder

Um den Einfluss der Quellencodierung auf Standbilder zu untersuchen, wird einzelne Schritte aus Standbildcodierung simuliert, um den Verlust im Schritt zu analysieren und Ergebnisse der BER und PSNR je nach Änderung der oben gegebene Modulationsamplitude, Größe der Datenblock und Bitrate zu erfassen.

3.2.1 Analysieren des Einfluss auf DCT-Codierung

Wie schon in Abb.2.8 gezeigt, besitzen der Schritt bei der Quantisierung den größten Verlust, somit die codierte Dataframes schwierig zum ursprüngliche Datei zu rekonstruieren. Um Quel- lecodierung für Standbildern zu simulieren, wurden im Hinblick auf dieser Arbeit ein Matlab-

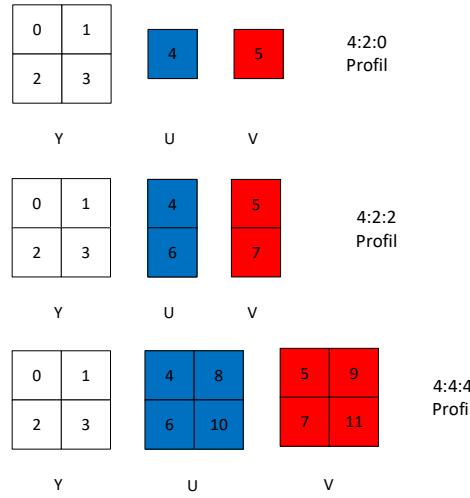


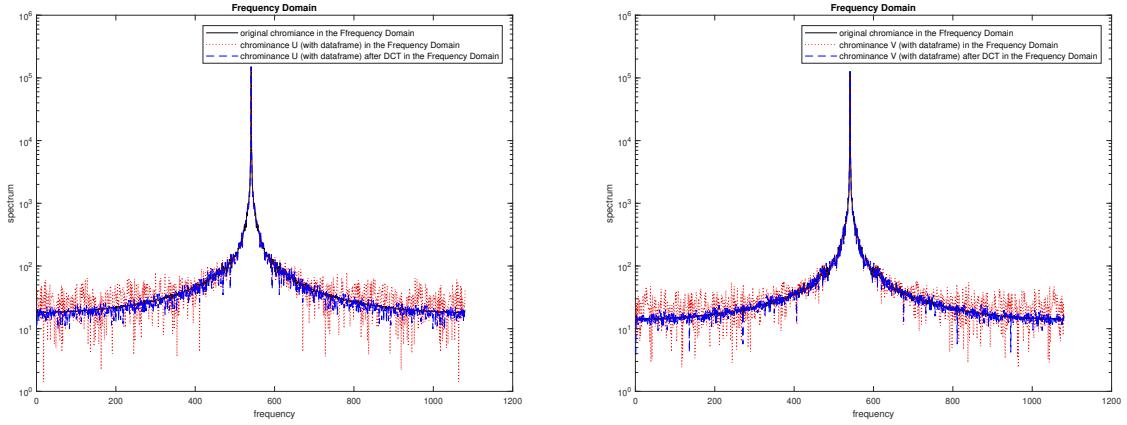
Abbildung 3.6: Makroblockstruktur der MPEG Profile.

Skript entworfen, in dem die einzelne Schritte von DCT und Quantisierung gezeigt werden, so dass die Datenänderung und der Verlust bei diesen Prozessen festgelegt werden können.

DCT

Die Durchführung einer DCT ist die Transformation der in 8×8 kleine Blöcke zerlegte Bildern vom Orts- in den Frequenzbereich. Die Abbildung 3.7 zeigt die Histogramm des Bildsignales im originalen Fall, nach FFT und DCT im Frequenzbereich. Daher ist es bemerkbar, dass eine große Menge der Signalenergie nach der DCT im Vergleich zur FFT (Fast Fourier Transformation) in der niedrigen Frequenzbereich konzentriert werden. Die DCT ermöglicht, die Energie zu bündeln. Um die Detaillierung der Änderung von Matrixelementen in U- und V-Kanal (siehe Abbildung 3.8) zu betrachten, wird anhand 4 Matrizen (jeweils mit Blockgröße 8×8) von U und V-Kanal vor und nach DCT die Änderung angezeigt. Es ist zu sehen, dass die Koeffizient in der oben linken Ecke nach der DCT den größten Wert besitzt. In der originale Matrizen sind nebeneinander Koeffizienten sehr ähnlich. Aber nach der DCT ist diese Ähnlichkeit dekorriert.

Weil unterschiedliche Blockgröße und Modulationsamplitude noch Einfluss auf der DCT haben, wird dort die Änderung der Bildinhalt anhand der ausgewählte 8×8 Blockgröße weiter angezeigt. Die Abbildung 3.9 zeichnet die Datenwerte nach DCT mit Datenblockgröße 4, 8 und 12, wobei die Modulationsamplitude 9 eingesetzt wird. Jeder Block wird in der gleiche Position mit gleicher Modulationsamplitude 9 und Hintergrund extrahiert. Was anders ist das unterschiedliches Blockgröße. Erkennbar ist, dass die Energie bei Blockgröße 8 fast alle in



(a) spektrale Antwort der Bildsignale in U-Kanal. (b) spektrale Antwort der Bildsignale in U-Kanal.

Abbildung 3.7: spektrale Antwort der Bildsignale vor und nach der DCT.

originale Pixelwerte in U-Kanal										DCT-Koeffizienten in U-Kanal									
15.7268	15.3918	15.2234	15.3912	15.6756	15.6762	15.1768	15.6768	233.9899	-4.6964	1.9041	-0.2408	-0.5615	0.2307	-0.9013	1.5740				
15.7268	15.8912	15.2234	15.3912	15.6762	15.8452	15.8394	15.0078	0.9560	-88.5875	-0.1547	32.9280	0.9642	-21.5168	0.2438	17.7737				
16.0590	15.3912	15.3912	15.8912	15.6768	15.1774	16.6768	15.3446	-0.7693	0.3366	-0.8675	1.1643	-0.0919	-0.2384	0.1325	0.0082				
15.3912	15.8906	15.5596	15.3912	14.6774	14.6774	16.6774	16.6762	-0.2859	31.8280	-0.3013	-10.9342	0.0530	6.7771	0.2335	-6.3661				
42.0102	42.5990	42.6710	42.1722	15.3918	15.3918	16.8912	17.0596	0.3134	-0.3830	0.2354	-0.6106	0.2714	-0.4840	0.5123	-0.1406				
41.5102	42.5990	42.6710	42.1722	15.3918	15.3918	16.8912	16.8912	0.2696	-21.2594	0.1717	6.8657	0.2030	-4.8575	0.0751	4.3917				
41.5108	42.0096	42.5990	42.0096	15.2292	15.2286	17.0602	16.3924	0.2515	0.2790	-0.1581	0.2077	-0.1660	0.3029	-0.3045	0.2513				
41.5108	42.0096	42.5990	42.0096	15.2286	15.8970	17.0602	16.3930	-0.1948	17.8625	-0.1956	-6.0371	-0.2062	4.4844	-0.4059	-3.2655				
originale Pixelwerte in V-Kanal										DCT-Koeffizienten in V-Kanal									
-22.8754	-21.8759	-21.3760	-21.8758	-11.5266	-11.5267	-11.4455	-11.5268	-147.4932	37.0733	-0.2776	-13.4807	-0.0859	8.9896	0.0694	-7.0198				
-22.8754	-21.9571	-21.3760	-21.8758	-11.5267	-11.8641	-10.6882	-10.9455	2.6860	10.0976	-0.0313	-2.9587	-0.8567	2.3287	0.0859	-1.7738				
-22.4569	-21.8758	-21.8758	-21.9571	-11.5268	-11.4456	-11.6894	-11.9453	-1.0887	0.1147	1.4518	-0.8026	0.1373	-0.0157	-0.3628	0.3336				
-21.8758	-21.9570	-22.3757	-21.8758	-11.3643	-11.3643	-11.6895	-11.6893	0.0505	-3.8230	-0.4862	1.0135	-0.3764	-0.4136	0.3517	0.8031				
-15.3361	-15.4172	-14.4985	-14.4174	-21.8759	-21.8759	-22.1197	-22.6196	0.2282	0.3653	-0.1607	-0.0076	-0.1032	0.2204	-0.2244	0.2638				
-15.2548	-15.4172	-14.4985	-14.4174	-21.8759	-21.8759	-22.1197	-22.1197	0.1232	2.6197	0.0785	-0.4280	-0.0557	0.2716	0.1328	-0.6959				
-15.2549	-15.3360	-15.4172	-15.3360	-22.7945	-22.7944	-22.6197	-22.0386	0.2701	0.1855	-0.5518	-0.0810	0.1681	-0.1511	0.2744	-0.0545				
-15.2549	-15.3360	-15.4172	-15.3360	-22.7944	-23.3756	-22.6197	-22.0387	-0.6424	-1.7882	-0.2155	0.6505	0.0750	-0.5723	0.2539	0.2139				

Abbildung 3.8: Änderung der Pixelwerte nach DCT.

der linken oberen Ecke gebündelt wird, während bei anderen Blockgröße die Koeffizient in der ober links nicht so hoch wie Blockgröße 8 konzentriert ist. Bei Größe 4, besonders in U-Kanal wird die Energie dezentral. Dies ist dadurch begründet, dass sich DCT oftmals auf Verarbeitung von 8×8 Blockgröße basiert. Wenn die Datenblockgröße gerade gleich wie die von DCT gegebene Blockgröße besitzt, wird die ganze Daten innerhalb einem Block verarbeitet, sonst ist eine Daten in zwei Blöcke eingeteilt oder innerhalb einem 8×8 Block zwei Daten vorhanden sind. Die nebeneinander Daten sind oft anders. Weswegen gibt die Daten meist den Einfluss gegeneinander an. Im Vergleich zum vorherigen Beispiel wird der Block mit Größe 8 festgesetzt. Die veränderte Parameter ist die Modulationsamplitude, in Höhe von 4, 9 und 12. Wie in Abbildung 3.10 dargestellt, wird die Koeffizient nach Änderung der Amplitude geändert. Bei der ausgewählte Modulationsamplitude gibt sich relativ hohe Konzentration der Energie in V-Kanal bei Größe 9 an, während in U-Kanal bei Amplitude 4 eine bessere Konzentration

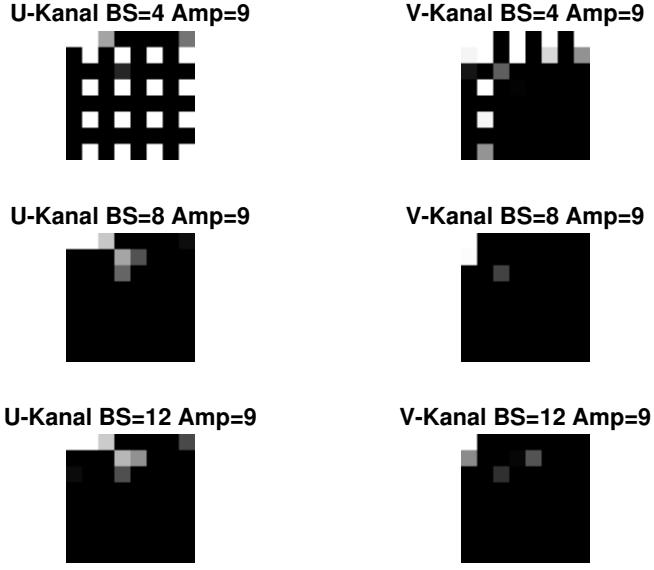


Abbildung 3.9: DCT-Koeffizienten nach Änderung der Blockgröße in U- und V-Kanal.

gegeben wird.

Quantisierungsfehler

Nach DCT werden die DCT-Koeffizienten durch Division mit den entsprechende Werte aus der Quantisierungsmatrix, welche der Wahrnehmung des menschlichen Auges für Farb- und Helligkeitsänderung entspricht. Die Abbildung 3.11 stellt die Histogramm von Daten vor- und nach Quantisierung im Frequenzbereich. Nach Quantisierung wird die Datenmenge reduziert. Da ist die Quantisierung jedoch ein inhärent nichtlinearer und irreversibler Prozess. D.h. der selbe Ausgabewert von mehreren Eingabewerten geteilt wird, ist es im Allgemeinen unmöglich, den genauen Eingabewert wiederherzustellen, wenn gegeben nur den Ausgangswert. Dieser Schritt verursacht den größten Verlust bei ganzer Standbildcodierung.

Zusätzlich verhält die Qualität der quantisierten Bild anders, wenn unterschiedliche Quantisierungsfaktor eingeführt werden. Mit großer Faktor werden die Daten stark komprimiert und liefert nachher schlechte Qualität. Im Gegensatz entsteht gute Qualität mit niedrige Kompression. Die Abbildung 3.12 zeigt die ausgewählte Blöcke im Bild mit hohe und niedrige Quantisierungsfaktor. Q bezeichnet hier die Quantisierungsfaktor. Je größer der Q ist, desto leichter wird das Bild komprimiert und dazu eine bessere Qualität liefert. Anschaulich werden bei $Q = 20$ mehr Hochfrequenzkoeffizienten unterdrückt.

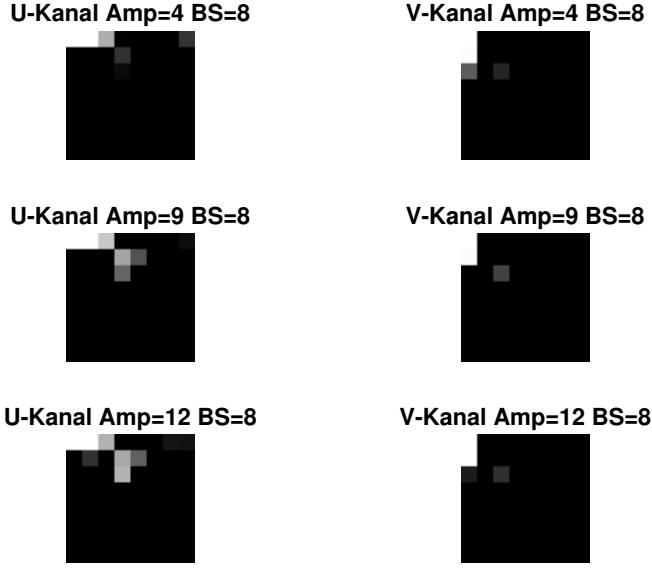


Abbildung 3.10: DCT-Koeffizienten nach Änderung der Blockgröße in U- und V-Kanal.

Nach der Quantisierung wird die DPCM für DC-Werte durchgeführt, die zur weitere Reduktion der Datenmenge dient. Dies lässt sich jeweils in der nächsten kommenden Abschnitt 3.2.2 für Standbilder und Abschnitt 3.3 weiter untersucht.

3.2.2 Simulation der Standbildcodierung in FFMPEG

Die nächste Simulation basiert sich darauf, dass ein Frame aus einem Video extrahiert und weiter zur Erzeugung eines neuen Videos benutzt wird. Das originales Videoauflösung ist eine Format von 4K (2160×3840). Um das neue Video zu erzeugen, wird das Software FFMPEG und den Videocodec von H.264/ AVC verwendet. Diese Einstellung entspricht der in weitere in Experiment benutzte Smartphonekamera (siehe Abschnitt 4.1). Das Standard H.264/ AVC ist für die Videoformat von HDTV (1080×1920) zuständig. Deswegen wird das ursprüngliches die Auflösung von (2160×3840) auf (1080×1920) verkleinert. Das Hintergrund in dieser Simulation bleibt beim Abspielen des neuen Video unverändert. Die Änderung im Video ist nur die wechselnde Datenwerte. Um relative Nähe Testumgebung zu gewährleisten, werden die hinter liegende 34 Frames aus dem gleichen originalen Video ausgenommen.

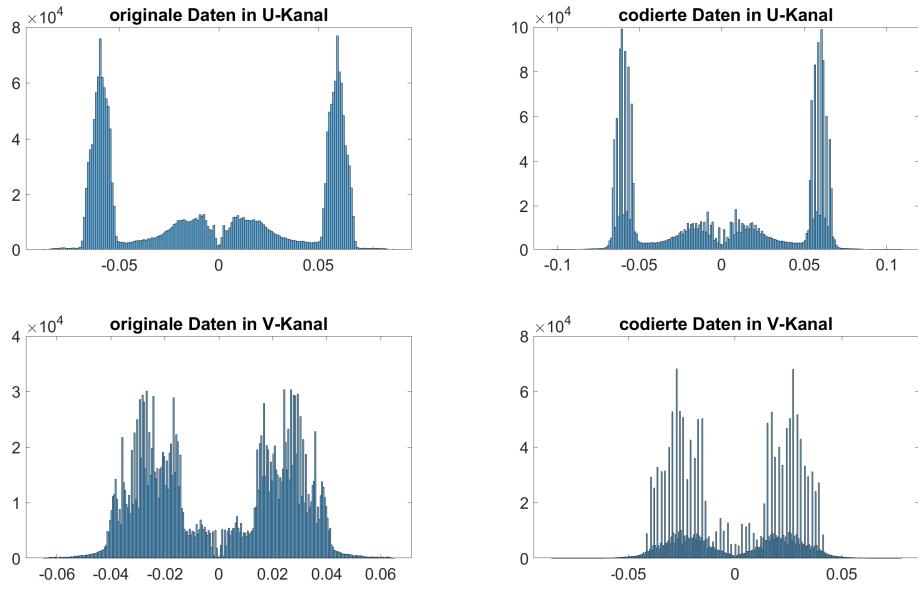


Abbildung 3.11: Histogramm der Daten vor-und nach Quantisierung.

FFMPEG

FFmpeg ist ein freies Software-Projekt, wobei eine umfangreiche Computerprogrammen und Programmbibliotheken zur Aufnahme, Konvertierung und Verpacken von Video-, Audio- und anderen Multimediadateien und Streams angeboten werden. Des Weiteren umfasst es mit libavcodec eine große Variante von Audio- und Videocodecs. Das libx264 wird hauptsächlich in dieser Arbeit als Videocodec benutzt, was ein kostenfreie Videocodec entsprechend der H.264 ist. Die Abbildung 3.13 zeigt ein Beispiel der Verarbeitungsprozess in FFMPEG. Die in Abbildung 3.2 generierte Standbilder mit Daten werden nun Frame für Frame in FFMPEG gelesen und weiter zu Video geschrieben. Ein Beispiel der Befehle zeigt wie folgend.

```
ffmpeg -r 30 -f image2 -i RGB%d.bmp -c:v libx264 -preset fast -crf 18
-x264-params bframes=0 -b:v 38M -maxrate 38M -minrate 38M
-bufsize 38M -pix_fmt yuv420p video.mp4
```

Die oben verwendete Parameter und die eingegebene Werte werden anhand einem Test in Google Pixel XL (siehe Abschnitt 4.1) festlegt. Einige wichtige Befehle zur Videocodierung werden in Tabelle erläutert.

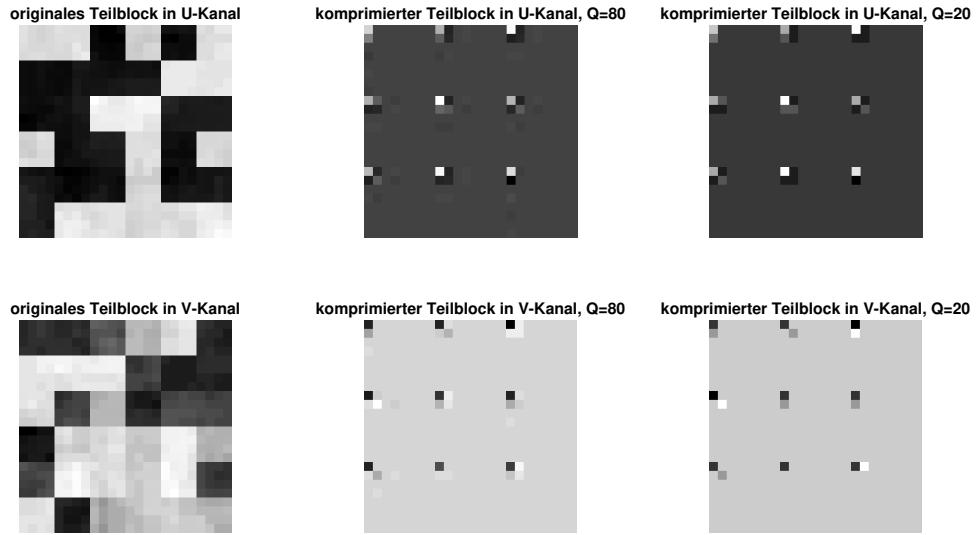


Abbildung 3.12: Einfluss der Kompressionsfaktor auf Quantisierung.

Rekonstruktionsfehler der Daten

In der Simulation werden entlang einer aufsteigender Folge die Modulationsamplitude und Blockgröße eingegeben, um die Einflüsse dieser Parameter auf die Codierungsprozess zu analysieren. Die Simulation wird mit 4 Bitrate durchgeführt. Die Abbildung 3.14 und 3.15 sind erhaltene Ergebnisse in U-Kanal und V-Kanal in 3D-Version unter 4 verschiedene Bitrate (5 Mbit/s, 10Mbit/s, 32Mbit/s und 38Mbit/s,). Anschaulich ist der Fehler bei kleiner Blockgröße und Amplitude sehr hoch. Besonders wenn $BS = 3$ und $A = 4$ sind, wird der Differenz zwischen originale und codierte über 50%, auch auch unter hohe Bitrate (38 Mbist/s). Dies gibt die Schwierigkeit zur erfolgreichen Rekonstruktion der Daten an. Je höher der Bitrate ist, desto leichter des Video quantisiert wird und weniger Fehler bei der Nachbildung auftritt. Wie der JPEG-Codierung verringert sich der Fehler von $BS = 3$ bis $BS = 4$ sehr erheblich, gegen 30% bis 40%. Ab $BS = 8$ fällt der Fehler langsam unter alle Bitrate. Auffällig ist, dass der Fehler unter hoher Bitrate (32 Mbit/s und 38 Mbit/s) nicht so grobe wie in niedrige Bitrate.

PSNR

Nach der Verarbeitung von Frame mit Daten kann durch PSNR die Qualität auswerten. Obwohl im Video belebt das Hintergrund immer gleich, beeinflusst die veränderte Daten auch auf das Standbild. Nach der Kompression verschlechtert sich die Datenqualität. Die Abbildung 3.16

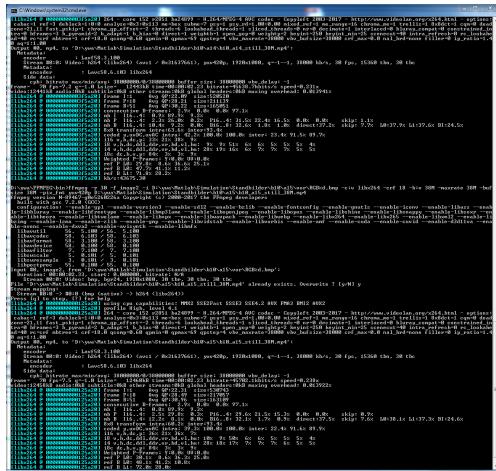


Abbildung 3.13: Videoverarbeitung in FFMPEG.

und 3.17 zeigen den Verlauf der PSNR von Datenwerte in U- und V-Kanal mit vier unterschiedlichen Bitrate (5Mbit/s, 10Mbit/s, 32Mbit/s, 38Mbit/s) je nach Änderung der Datenblockgröße und Modulationsamplitude. Mit der erhöhten Amplitude verringert sich die Qualität der Datenframes. Die Änderung der PSNR weist eine Schwankung auf, wenn sich die eingegebene *BS* vergrößert. Die PSNR ist jedoch bei *BS* 3, 5, 10 und 15 im Vergleich zur andere offensichtlich niedrig. Die entspricht auch vorher in Abschnitt 3.2.1 erklärt Grund. Mit *BS* = 8 ergibt sich eine relativ bessere PSNR. Von *BS* = 5 bis *BS* = 8 stellt eine erhebliche Erhöhung maximal bis zum c.a 20% dar.

DPCM für Bewegungsvektor und Prädiktion

Das Hintergrund dieses Video bleibt immer unverändert. Die Änderung der Bewegung ist nur die wechselte Daten, die durch geänderte Größe von Datenblock und Amplitude generiert werden. Deswegen bezieht sich die Bewegung prinzipiell nur auf dem Wechsel der Datenwerte.

Wie im Abschnitt 2.3.2 erwähnt, wird die Bewegungsschätzung und Bewegungskompensation im Encoder durchgeführt, um die Bewegung zwischen vorherige und nachkommende zu schätzen, sodass die Datenmenge noch weiter reduziert werden kann. Die Abbildung 3.18 zeigt die dazu erzeugte Bewegungsvektor. Weil viele zufällige Datenwerte am Anfang generiert werden, stellt hier die Ausrichtung der Daten auch sehr zufällig dar. Das mit Bewegungsschätzung und Bewegungskompensation entstehendes Frame behaltet nur wenige Information. Unter gleicher Bitrate ist die meisten Information die Bewegungsvektoren. Es ist ersichtlich, dass die Anzahl der übertragene Daten mit DPCM deutlich weniger als die ohne DPCM. Dadurch werden nur das Bildinhalt mit Bewegungsvektoren transformiert und nicht das ganze

Tabelle 3.1: Befehle von Video Codec in FFMPG.

Befehle	Beschreibung
-r	set the video framerate.
-f image2	convert sequente pitctures to video.
-c:v	set video codec model, libx264 will be chosed for this simulation.
-crf	Constant Rate Factor.
-b	set the video bitrate in bit/s for CBR (Constant Bit Rate).
-maxrate	set the maximal bitrate in bit/s.
-bufsize	set the rate control buffer.
-pixfmt	Blame the RGB to YUV color space conversion in 4: 2:0 format.
-preset	a collection of options that will provide a certain encoding speed to compression ratio.

Bild. Daraus werden mehr Platz gespart und dazu eine bessere Kompression realisiert. Die MSE mit Bewegungsschätzung ist nur 15,898, aber wenn die Prädiktion ohne Bewegungsschätzung zieht sich die MSE auf 372,624 an.

3.2.3 Auswertung

Vor der Datenkompression beeinflussen die Vorschritte schon auf der Daten. Die hochfrequente Datenwerte werden durch Gaußfilter untergedrückt und nach Gammaverzerrung verdunkelt die Wiedergabe und eine steigende Sättigung resultiert. Während der Datenverarbeitung bietet Die DCT keine Datenreduktion an, sondern spielt eine wichtige Rolle für Vorbereitung zur folgenden Quantisierung. DCT hat die Energie sehr hoch konzentriert. Bei Blockgröße 8 konzentriert die Energie in den 8×8 Koeffizientenblock fast alle in der oben linke Ecke. Dies entspricht der blockbasierte DCT, welche sich oftmals auch mit 8×8 Datengröße behandelt. Der Verlauf von PSNR stellt eine Verschlechterung mit steigender Modulationsamplitude und Verbesserung mit zunehmenden Datenblockgröße dar. Wenn die Modulaitonsamplitude er-

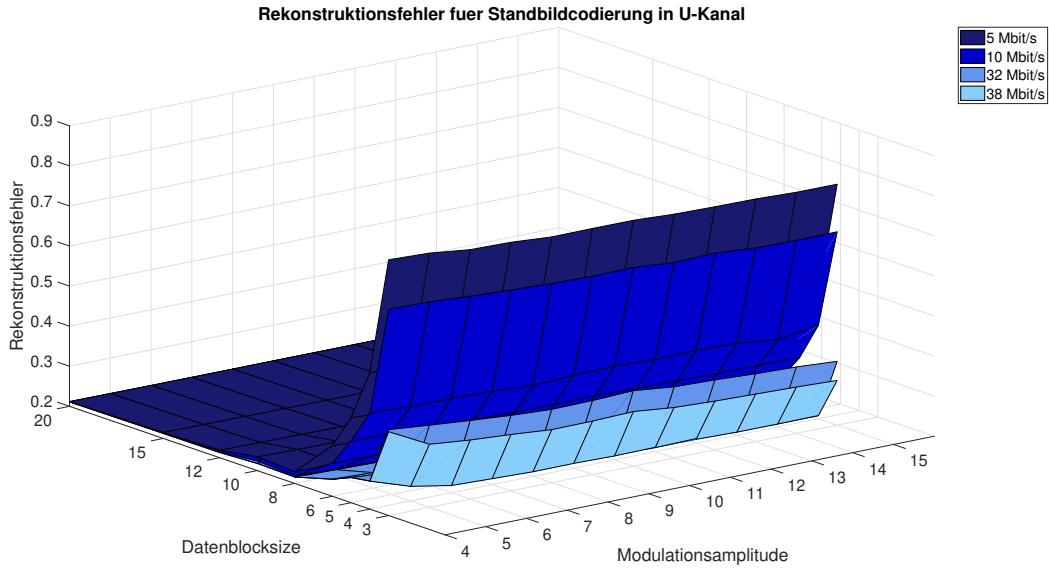


Abbildung 3.14: Rekonstruktionsfehler in U-Kanal nach Verarbeitung in FFMPEG.

höhrt, vergrößert die Differenz zwischen originalen und codierten Daten gleichfalls. Laut der Gleichung 2.4 ist die MSE ebenfalls angestiegen und führt zu schlechterem PSNR.

während die Rekonstruktionsfehler eine abnehmende Tendenz mit vergrößerte Datenblock und erhöhte Modulationsamplitude. Wie in letzten Absatz erklärt, ist Rekonstruktionsfehler relativ klein ab Blockgröße 8, die gilt auch für PSNR. Bei anderen Größen der Datenblock, besonders bei 3, 5, 10, 15 verringert sich die PSNR. Wenn die Datenblockgröße von 8 bis 5 verkleinert, reduziert sich die PSNR sehr schnell und die Rekonstruktionsfehler auch schnell steigt. Dies lässt sich so begründen, dass die ab DCT-Codierung das Prozess immer blockbasiert durchgeführt. Wenn die Blockgröße gleich oder ein ganzzahliges Vielfach von 8 ist, wird eine ganze Daten in 8×8 Block oder eine Daten in zwei oder auch mehrere 8×8 nebeneinander unterteilt, ohne die Daten aus der Mitte ausgeschnitten werden. Dazu weist eine relativ bessere PSNR und bezüglich geringere Rekonstruktionsfehler auf. Außerdem wenn die Datenblockgröße kleiner als 5 ist, wird die Datenwerte mehr von Modulationsamplitude beeinflusst. Wenn sich die Amplitude anzieht, verstärkt das Rauschen gleichzeitig. Daraus verschlechtert die SNR und führt zu einer hohen Rekonstruktionsfehler.

3.3 Einfluss der Quellencodierung auf Bewegtbilder

Die Simulation im letzten Abschnitt basiert sich auf Codierung für Standbilder mit geänderte Daten. Die Bewegung zwischen Frames ist nur durch Wechsel der Daten erzeugt. Beim Ab-

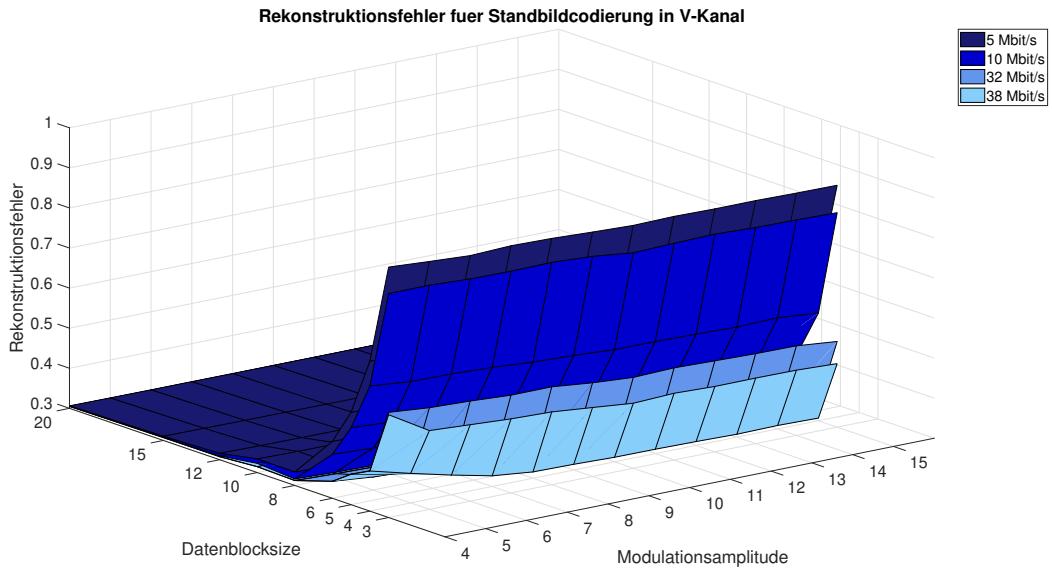


Abbildung 3.15: Rekonstruktionsfehler in V-Kanal nach Verarbeitung in FFMPEG.

spielen des Videos beeinflusst das hinterlegende Standbild nicht auf der Daten. Wenn sich das Bildinhalt selbst auch bewegt, werden die Daten noch von Bewegung der Bildern beeinflusst. In diesem Abschnitt behandelt sich die Simulation auf ein bewegtes Hintergrund mit geänderten Daten.

3.3.1 Simulation in FFMPEG

Um die Daten auf originales Video zu addieren, wird die Frames zuerst aus dem gleichen Video, das in Standbildcodierung verwendet extrahiert. Um die Umgebung relativ gleich zu bleiben, wird das Startframe gleich wie vorher benutzt und die andere Frames sind nur die hinterlegende Frames von dem Startframe. Als nächsten Schritt wiederholt jedes Frame einmal und addiert sich jeweils eine Paar Daten auf. Die andere Schritte laufen gleich wie die Standbildcodierung.

Rekonstruktionsfehler der Daten

Im Vergleich zur Standbildcodierung stellt der Fehler zur Rekonstruktion einen ähnlichen Verlauf wie in der Standbildcodierung dar. Wenn die Größe der Datenblock kleiner als 5 ist, bleibt die Fehler auch sehr hoch, von 30 bis 50 Prozent, bei Blockgröße 3 und Amplitude 4 wird der

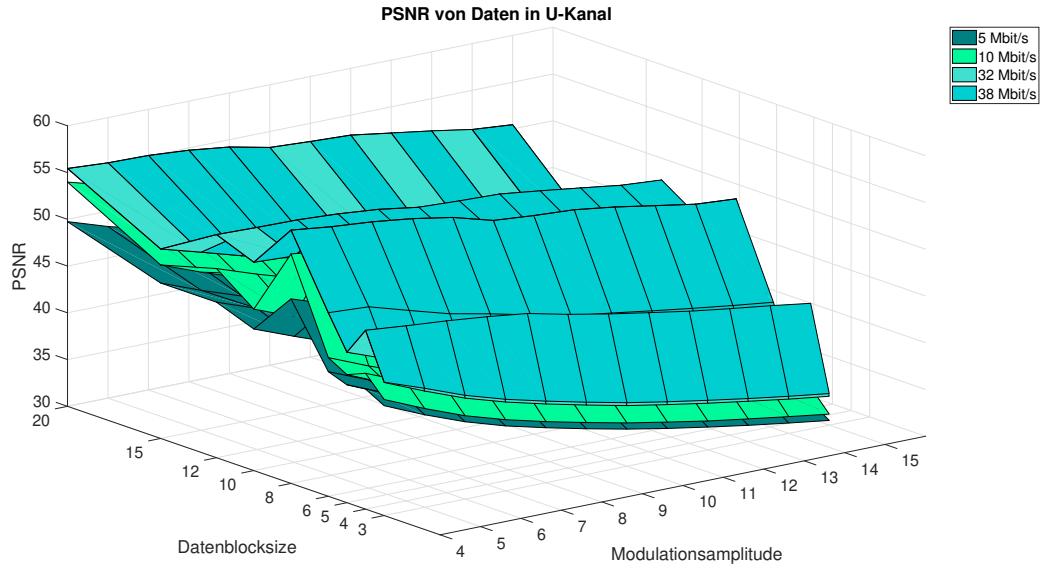


Abbildung 3.16: PSNR von Daten in U-Kanal.

Fehler am höchsten, aber bei Amplitude 8 und 11 wird ein relativ niedriger Fehler aufgezeichnet. Bei Blockgröße 3, 5, 10, 15 erhöht die Rekonstruktionsfehler, insbesondere vergrößert der Block von 5 bis 8, steigt der Fehler sehr stark. (siehe Abbildung 3.21). Die Abbildung der Rekonstruktionsfehler in V-Kanal hat den ähnlichen Verhalten wie in U-Kanal, nur die durchschnittliche Fehler relativ niedrig als in U-Kanal(siehe Anhang A.3).

PSNR

Die Abbildung 3.22 zeigt die PSNR in U-Kanal. Mit vergrößerte Datenblock steigt die PSNR. Die gleiche Abnahmetendenz trifft jedoch auch bei Blockgröße 3, 5, 10 und 15 auf. PSNR nimmt mit der erhöhte Modulationsamplitude ab. Das Grund besteht darin, dass die Mean Square Error von Daten wegen steigender Amplitude auch angewachsen ist. Laut der Gleichung von PSNR (siehe 2.4) nimmt die PSNR ab. Die PSNR in V-Kanal verhältet sehr ähnlich wie in U-Kanal (siehe Anhang A.4)

DPCM für Bewegungsvektor und Prädiktion

Die Abbildung A.11 zeigt die dazu erzeugte Bewegungsvektor. Weil viele zufällige Datenwerte am Anfang generiert werden und sich das Hintergrund zwischen zwei Frames auch verändert, stellt hier die Ausrichtung der Daten auch sehr zufällig dar. Die Abbildung 3.23 ist Differenzbild

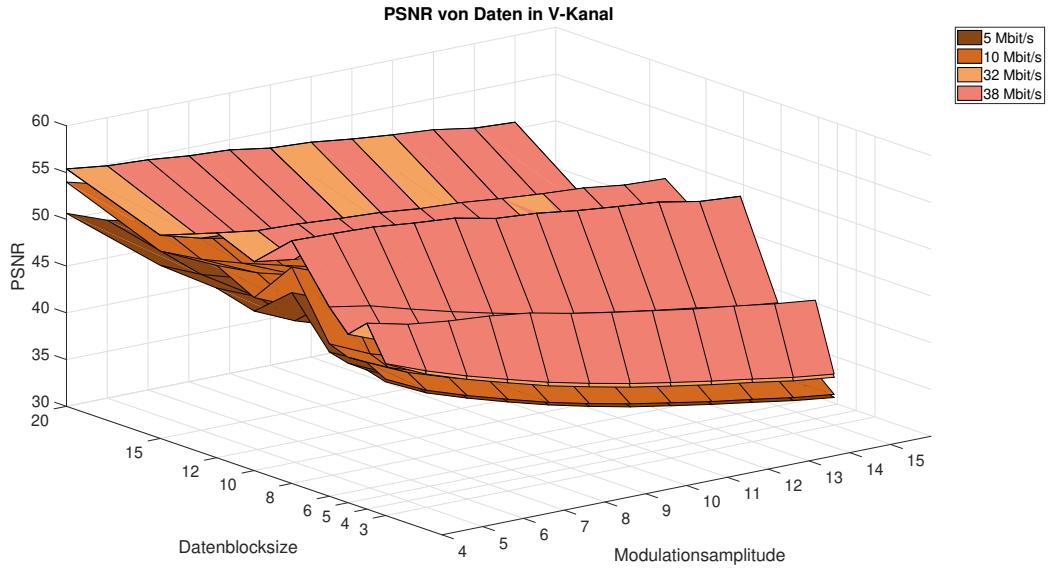


Abbildung 3.17: PSNR von Daten in V-Kanal.

zwischen vorherigem und aktuellem Frame. Das linke Bild ist ohne Bewegungsschätzung sowie Bewegungskompensation erzeugt, während das linke mit der Bewegungsschätzung sowie Bewegungskompensation. Das durch Prädiktion erzeugte Bild zeigt in der Abbildung 3.24.

Das mit Bewegungsschätzung und Bewegungskompensation entstehendes Frame behält mehr Information als die Standbilder. Weil sich das hinter legendes Bild auch bewegt, nimmt die Anzahl der zu übertragenden Information zu.

Gleich wie in der Standbildercodierung verringert sich die Anzahl der übertragene Daten mit DPCM deutlich weniger als die ohne DPCM (siehe Abb.A.12 im Anhang). Die MSE mit Bewegungsschätzung ist nur 13.953, aber wenn die Prädiktion ohne Bewegungsschätzung zieht sich die MSE auf 194.969 an. Dadurch ist die zu liefernder Differenzbild ganz klein als das originales.

3.3.2 Auswertung

Unter der gleichen Bitrate wird der Rekonstruktionsfehler deutlicher höher als der in Standbildcodierung. Das Grund liegt darin, dass im Bewegtbildern mehr Bewegungsvektoren zur Prädiktion erzeugt und gespeichert werden. Wenn mehr Speicherplatz dafür ausgenutzt wird, steht daher geringe Platz für Speicherung von Daten. Dies resultiert schlechte Qualität und PSNR von nachgebildeter Bildern und Daten. Wenn sich die Szene sehr schnell wechselt und

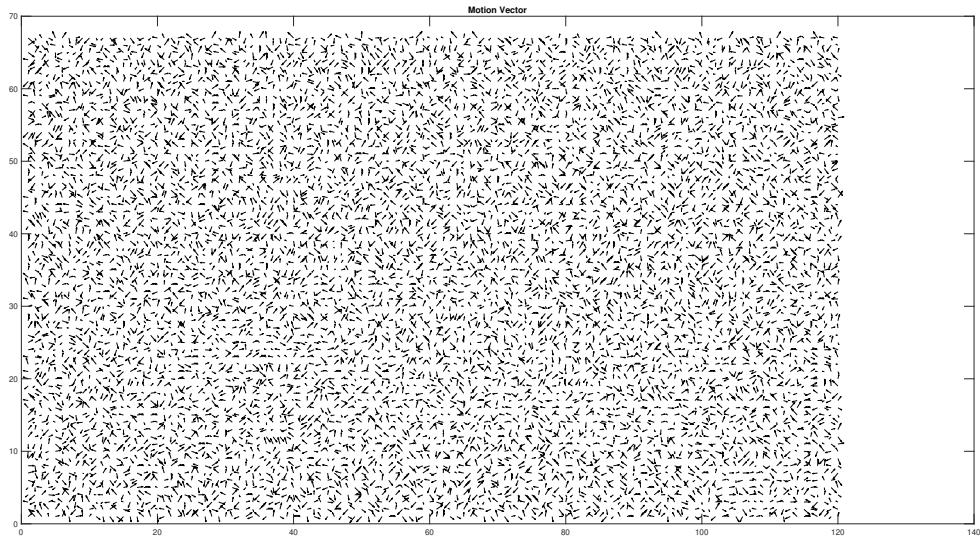


Abbildung 3.18: die entstehende Bewegungsvektoren.

die Bitrate niedrig ist, verursacht eine starke Quantisierung und dazu resultiert einen sichtbaren Übergang zwischen vorherigem und aktuellem Frame. Aufgrund der hohe Quantisierung wird die Datenmenge vom Video sehr klein. Unter dieser Bedingung wird die meisten hochfrequente Information in Quantisierung und DPCM unterdrückt. Die genaue Struktur, wie z.B die Kanten zwischen zwei Datenblock und der Übergang zwischen Himmel und Boden sind meistens die Hochfrequenzsignale. Das Video sieht nicht so kontinuierlich aus, sonder viele scharfe Kanten. Nach der Analyse und Untersuchung wird in drei Simulation (JPEG-Codierung, Standbildercodirung und Bewegtbildercodierung) anhand der Ergebnisse von Rekonstruktionsfehler und PSNR angezeigt, dass sich bei Blockgröße 8 und Amplitude 9 geringer Fehler und PSNR ergibt. Andererseits hat das hinter liegende Grundbild auch Einfluss auf der Übertragung der Daten. Vergleich mit die Codierung von Standbildern und Bewegtbildern ist daraus zu folgen, dass geringer Fehler bei Standbildcodierung auftritt, wenn die Bitrate, Datenblockgröße und Modulationsamplitude von bei der Codierungsverfahren gleich sind. Dies lässt sich in Kapitel 4 in Experiment weiter untersuchen.

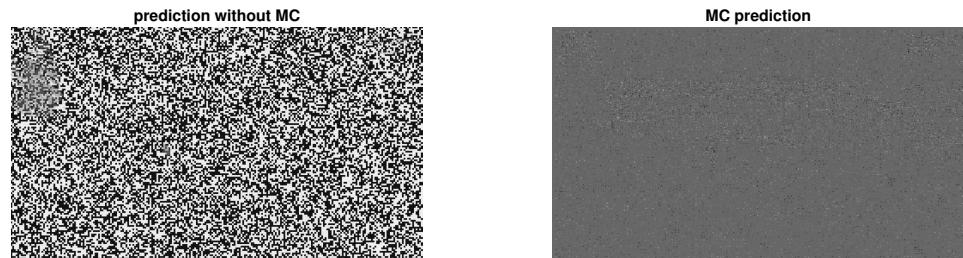


Abbildung 3.19: das erzeugtes Bild mit und ohne Bewegungsschätzung.

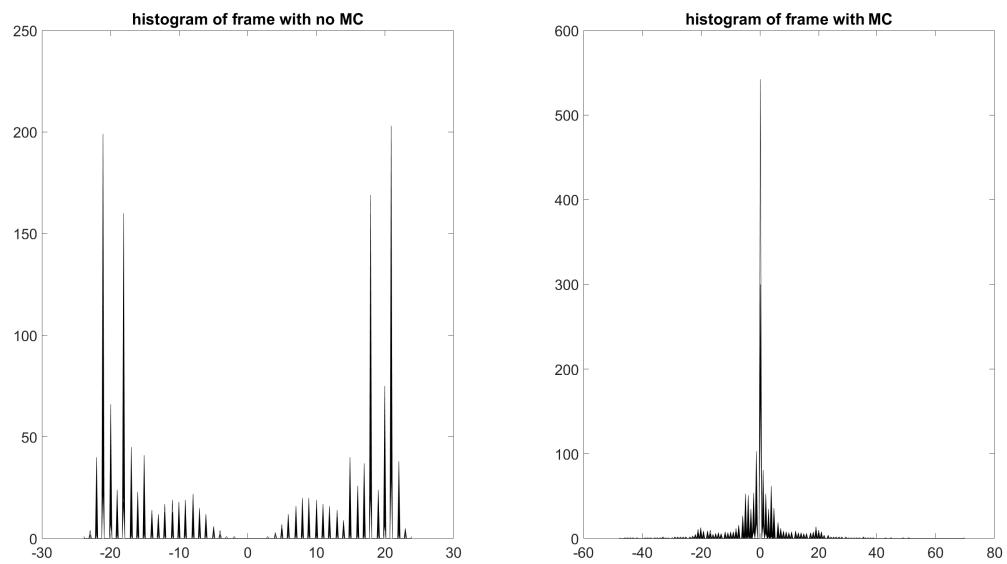


Abbildung 3.20: Histogramm des Frames mit und ohne Bewegungsschätzung.

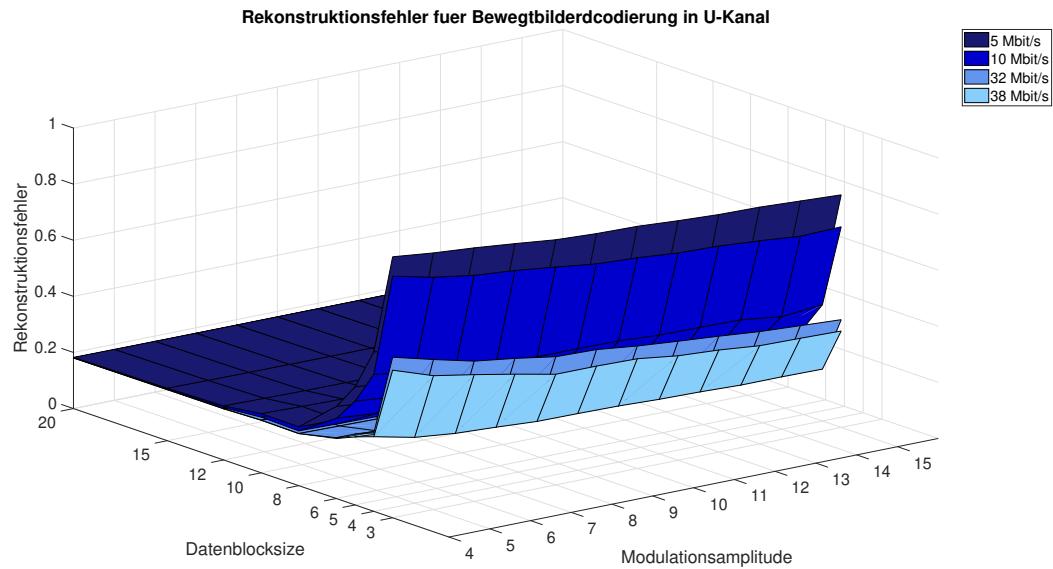


Abbildung 3.21: Rekonstruktionsfehler in U-Kanal nach Verarbeitung in FFMPEG.

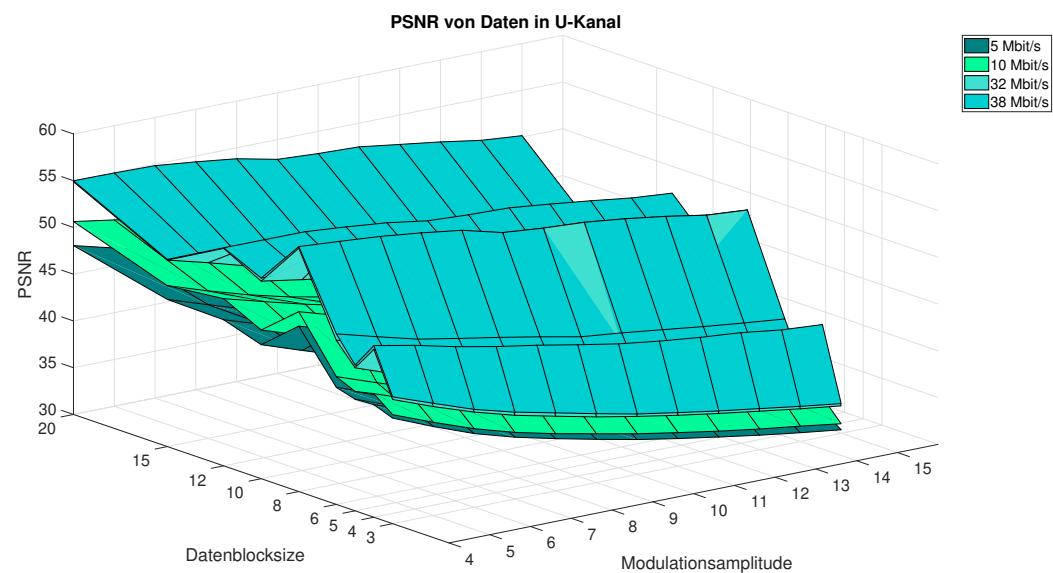


Abbildung 3.22: PSNR von Daten in U-Kanal.

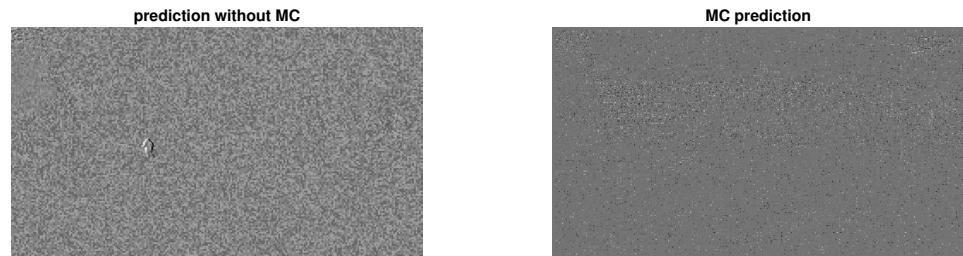


Abbildung 3.23: das Differenzbild mit und ohne Bewegungsschätzung.

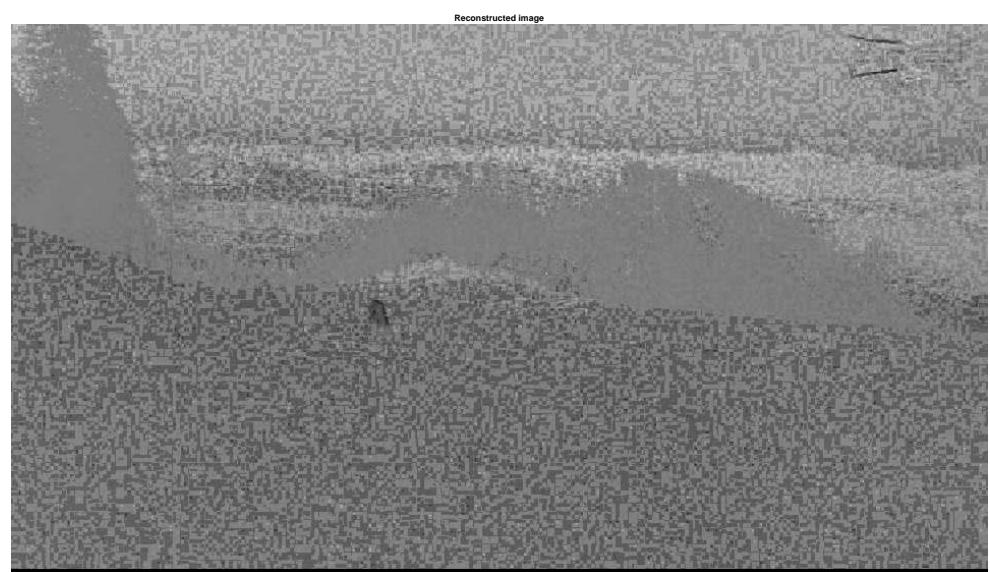


Abbildung 3.24: durch Bewegungsschätzung entstehendes Bild.

4 Experimente

Die im Kapitel 3 durchgeführte Simulation bewirkt sich als Referenz zur folgende Experiment, wobei die Videoframes mit digitale Kameras aufgenommen werden und weiter in FFMPEG bearbeitet, um der Einfluss der Quellencodierung in reale Szene zu untersuchen. Danach werden der Ergebnisse dargestellt, wie z.B Rekonstruktionsfehler, PSNR von Daten, BER zwischen Raw-Daten und decodierten Daten und Vergleich der ausgewählte Blöcke im unterschiedlichen Bereich innerhalb einem Frame (langsam bewegt dunkler und heller Bereich, schnell bewegte dunkler und heller Bereich) und die Besonderheiten im Verlauf der oben erwähnten Ergebnisse aufgezeigt und erklärt. Die zur Experimente benutzte Parameters, die die Video- und Datenqualität beeinflussen, sind die Datenblockgröße, Modulationsamplitude und die Bitrate.

4.1 verwendete Geräte und Testumgebung

verwendete Geräte

Die in Experimente verwendete Geräte und die herstellerspezifische Konfiguration werden in Tabelle 4.1 aufgeführt. Die Auflösung bezeichnet die Auflösung von Wiedergabegerät. Die Aufnahmemodus beschreibt die Aufnahmerate pro Sekunde und die Auflösung der aufgenommenen Videos. In dieser Arbeit wird das Modus von 4K 2160p video in Google Pixel XL ausgewählt.

Testumgebung

Die originale Videoframes wurden mit digitalem Kamera aufgenommen (Industriekamera und Smartphonekamera) und zum PC transformiert. Das aufgenommene Frames wurden als YUV- oder Raw-Datei Format gespeichert und weiter zur Codierung in FFMPEG benutzt. Danach lassen sich die Frames in Matlab und FFMPEG weiter verarbeiten. Die ursprüngliche Frames dienen als Referenz zur Vergleich mit die codierte Videoframes. Da die übertragene Daten DaViD-System werden durch Smartphone wieder erhalten, ist es notwendig, die Video Codec

Tabelle 4.1: die in Experimente verwendete Geräte und deren Konfiguration.

Kategorie	Bezeichnung	Auflösung	Aufnahmemodus
Monitor	OLED LG 55EF950V	2160*3840	
Industriekamera	Emergent Vision HS-12000C	bis zu 12 Megapixels	87, 179, 338 Bilder/s
Smartphonekamera	Google Pixel XL	12,35 MP	4K 2160p video (Up to 30 FPS), FHD 1080p video (Up to 120 FPS), HD 720p (up to 240fps)
Kategorie	Bezeichnung	Brennweite	Lichtstärke
Objektiv	AFS-Nikkor	24 bis 70mm	1:2,8

für Smartphonekamera vor der Videocodierung in FFMPEG zu erfassen, um die entsprechende Parameters in FFMPEG einzustellen und die präzis Testumgebung wie in Smartphone zu simulieren.

Um die Videocodec in Google Pixel XL einzusammeln, werden 4 Modus für Videoaufnahme ausgewählt und jeweils zwei Video aufnehmen, eine still bleibt und die andere schnell bewegt. Daraus kann die Parameters umfänglich festlegt werden. In Tabelle 4.2 zeigt die erfassende Videocodec und die relevante Parameters.

Tabelle 4.2: Videocodec und Parameters von aufgenommene Videos in Google Pixel XL.

Videoname	Modus	Videocodec	Profile	GOP	Bitrate
30Hz-schnell	4K 2160p video (Up to 30 FPS)	H.264/AVC	High	nur I und P-Frames	$\approx 38Mb/s$
30Hz-langsam	4K 2160p video (Up to 30 FPS)	H.264/AVC	High	nur I und P-Frames	$\approx 38Mb/s$
60Hz-schnell	FHD 1080p video (Up to 60 FPS)	H.264/AVC	High	nur I und P-Frames	$\approx 32Mb/s$
60Hz-langsam	FHD 1080p video (Up to 60 FPS)	H.264/AVC	High	nur I und P-Frames	$\approx 32Mb/s$
120Hz-schnell	FHD 1080p video (Up to 120 FPS)	H.264/AVC	High	nur I und P-Frames	$\approx 10.3Mb/s$
120Hz-langsam	FHD 1080p video (Up to 120 FPS)	H.264/AVC	High	nur I und P-Frames	$\approx 10.3Mb/s$
240Hz-schnell	HD 720p (up to 240fps)	H.264/AVC	High	nur I und P-Frames	$\approx 5.2Mb/s$
240Hz-langsam	HD 720p (up to 240fps)	H.264/AVC	High	nur I und P-Frames	$\approx 5.2Mb/s$

Die Parameters in Tabelle erscheint, dass die Bitrate in gleichem Modus fast konstant bleibt und alle Videocodec das Standard von H.264/AVC benutzt. Außerdem irgend wie schnell sich der Inhalt im Video bewegt, enthaltet eine GOP nur I und P-Frames. Daraus werden die Videocodec und relevante Parameters zur Codierung festlegt.

4.2 Experiment mit Industriekamera

Emergent Vision HS-12000C ist das in diesem Experiment verwendetes Kamera. Das ist ein 10GigE-Kamera für die industrielle Bildverarbeitung. Typisch sind Prüfanwendungen in automatisierten Qualitätsprüfsystemen mit hohen Geschwindigkeit (87, 179, 338 Bilder/s) und Auflösung (2048×1088 , 2048×2048 , 4096×3072 Pixel). Das Industriekamera in diesem Experiment kann im geeigneten Abstand den Bildsequenz in 4K-Format (3860×2160 Pixel) aufnehmen und mit Raw-Datei im PC gespeichert werden. Vor Generierung neue Videosignale plus Datensignale werden alle Frames aus Video zur HD-Format (1920×1080 Pixel) verkleinert. Dies erfüllt das in H.264/AVC definierte Standardisierung.

Die Kamera ermöglicht, dass die Framerate an die Wiedergabeseite von 60 Hz angepasst wird. Die Experiment berücksichtigt sich nun nicht mehr auf der Standbildcodierung. Weil die Aufnahmerate gleich wie originale Rate ist und das originales Bild direkt zum Einsatz kommen könnte, macht dann die Untersuchung für Einzelstandbild keinen Sinn. Die Experiment mit Industriekamera beschäftigt sich nun auf eine bewegte Videosignale und nur auf der Blockgröße 8 und 16 in 4K-Format sowie die Modulaitonsamplitude von 4, 9, 13, 15, 20. Der Auswahl dieser Parameters basiert sich auf der in Simulaiton festgelegte Größe, die relativ schlechte und auch gute Ergebnisse von Rekonstruktionsfehler und PSNR aufweist. Weil die Datenblockgröße größer als 8 ist, nimmt die Fehler und PSNR langsam ab, während die Größe der Datenblock kleiner als 5 ist, tritt mehr Fehler auf. Bei Blockgröße 4 erhalte eine relative bessere PSNR und weniger Fehler im Vergleich zur benachbarte Blockgröße von 3 und 5. Daraus lässt sich ein anscheinender Vergleich zwischen dieser beide Datenblockgröße. Mehr Untersuchung der Einfluss von Blockgröße und Simulationsamplitude wird mit Smartphonekamera durchgeführt.

4.2.1 Quellencodierung auf Videos

Die Bewertungskonzept in diese Experiment ist gleich wie in Abbildung 2.18 gezeigt. Nach Kompression des Videos in FFmpeg werden die Daten in Matlab ausgenommen und vergleichen mit der ursprüngliche Datenwerte. Wie im Abschnitt 2.3 schon erwähnt, wirkt die Quantisierung stark auf hochfrequente Komponenten ein. Die Übergang zwischen zwei Datenblock gehört meistens zur Hochfrequenz. Weswegen werden Alle Daten, auch in codierte Daten in der Mitte im Abstand der Blockgröße abgetastet und die abgetastete Punkte kommen für die Berwertung zum Einsatz, um besser zu beobachten, wie die Quellencodierung auf der übertragene Daten beeinflusst.

Rekonstruktionsfehler

Die Abbildung zeigt die Rekonstruktionsfehler in U-Kanal unter 4 Bitrate (5Mbit/s, 10Mbit/s, 32Mbit/s und 38Mbit/s), welcher an die Bitrate von Smartphonekamera (Google Pixel XL) angepasst.

PSNR

Vergleich der Blöcke unterschiedlichem Bereich in einem Bild

Nach einer objektiven Bewertung durch mathematische Methode werden jetzt einige Blöcke, die innerhalb eines Bildes aber in unterschiedlichem Bereich ausgewählt, z.B. in dunklem, hellem, mehr und wenig bewegtem Bereich. Die unterschiedlichen Bildinhalt bewirkt auch anders bei der Datenübertragung. Die Datenblöcke werden bei eingegebene Datenblockgröße (später in Abkürzung *BS* bezeichnet) 8 mit Amplitude 9(*A*) unter Bitrate 5 Mbit/s und 38 Mbit/s in zwei Szene (mit dunklem und hellem Hintergrund) (siehe Abbildung). Ersichtlich ist, dass die Blöcke aus schnell bewegtem Hintergrund nach der Kompression mit Bitrate 5 Mbit/s verzerrt werden. Die Datenstrukturen ändert sehr unklar. Wenn die Bitrate erhöht, bzw. eine niedrige Kompression durchgeführt wird, ergibt sich eine bessere Datenstrukturen sowie sichtbare Kanten zwischen Datenblöcke (siehe Abbildung A.14). Die Differenzbild enthält nur wenige Kanteninformation. Unter Betrachtung der Histogramm zwischen beider Bitrate besteht auch weniger Änderung der Datenwerte unter hohe Bitrate.

Wenn jetzt ein helles Hintergrund des Videos berücksichtigt wird, weist die ausgewählte Blöcke schon eine gute Strukturen sonst unter niedrige Bitrate von 5 Mbit/s (siehe Abb.A.13).

Auswertung

Aufgrund der hohe Auflösung von Industriekamera kann der Datenstruktur deutlich wieder nachgebildet. Die Qualität der rekonstruierte Daten ist auch von der Bewegung und Kolorit des Videos abhängig. Nach Auswahl der Datenblockgröße und Modulationsamplitude lässt sich daraus folgen, dass eine klare Datenstrukturen in folgende Fällen wieder rekonstruiert werden: Entweder eine hohe Bitrate angibt, kann eine gute Rekonstruktion auch bei kleiner *BS* und *A* erzielt oder unter niedrige Bitrate sind große *BS* und hohe *A* erforderlich.

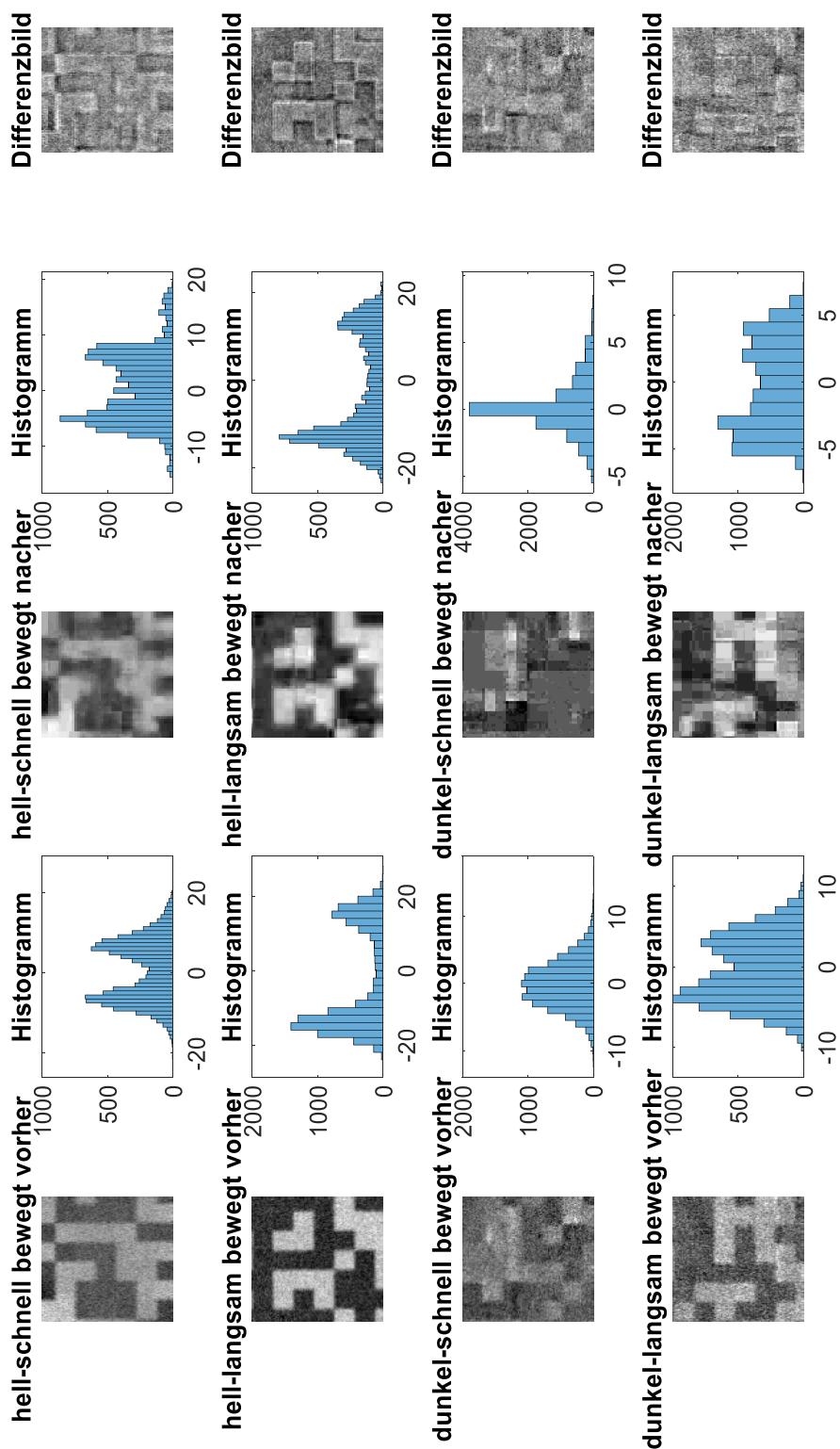


Abbildung 4.1: ausgewählte Blöcke aus Dataframes mit dunklem Hintergrund unter 5 Mbit/s.

4.3 Experiment mit Smartphonekamera

Mit dem von Lehrstuhl für Kommunikationstechnik entwickelte Anwendung „interactive.m“ kann jedes Frame im aufgenommen Video in Raw-Datei sowie YUV-Format in Smartphone gespeichert. Diese Bilder werden weiter zur Untersuchung der Einfluss von Quellencodierung in FFMPEG bearbeitet. Danach vergleichen die ursprüngliche Daten mit der verarbeitete Daten. Daraus lassen sich BER und PSNR errechnen. Um die Einfluss aus unterschiedliche Szene zu untersuchen, werden in dieser Experiment vier verschiedene Videoszene ausgewählt: ein Paar bestehen zwei Szene, eine mit schnelle Bewegung und die andere relativ langsame und beide mit dunklem Hintergrund. Die andere Paar besteht aus der gleichen Bewegungssituation aber mit hellerer Hintergrund. Die in Simulation benutzte $BS = 4$ ist wegen Problem bei der Decodierung weist eine nicht stabilen Verhalten auf und oft nicht decodierbar. Weswegen werden dort nur $BS = 5, 6, 8, 12$ verwendet.

4.3.1 Quellencodierung auf Videos

Die Abbildung 4.2 zeichnet den Ablauf der Experiment mit Smartphonekamera. Um den reinen Bildinhalt aus Video zu extrahieren, wurden vier QR-Pattern in jedem Frame eingebettet und auf der Ecke gestellt. Das vom Lehrstuhl für Kommunikationstechnik entwickelte Programm „SmartphoneDecoder.m“ in Matlab ermöglichen dazu, die Monitorkanten mittels vier QR-Pattern zu detektieren und nur den Bildinhalte im Monitor herauszuziehen. Um die gleiche Bedingung für Ausschneiden der Monitorkanten in codierter Videoframes zu gewährleisten, werden die gleiche Position von QR-Pattern in codierter Bilder detektiert. Die vorher erhaltene Frames können nun mit dem Programm örtlich abgetastet und decodiert werden. Daraus lassen sich die BER und PSNR für ursprüngliche Datenwerte nach Decodierung errechnen. Bei komprimierter Videoframes sollte das Frame auch in der gleichen Position wie in originales Frame örtlich abgetastet. Die komprimierte Daten in FFMPEG werden auch von verarbeiteten Video herausgenommen. Dadurch können die BER von ursprünglicher Daten und der komprimierte Daten im Programm berechnet und miteinander vergleichen. Alle zur mit Smartphonekamera aufgenommene Videoframes sind in 4K-Format mit Auflösung 2160×3840 . Wie die Experiment mit Industriekamera werden alle Frames zuerst mit Befehle „imresize“ und Einstellung von „nearest“ zu 1080×1920 verkleinert.

BER

Die BER wird nach der Decodierung durch Vergleich der übertragene Datenframes mit originaler Dataframes berechnet. Aber das in FFMPEG gearbeitete Video kann nicht immer

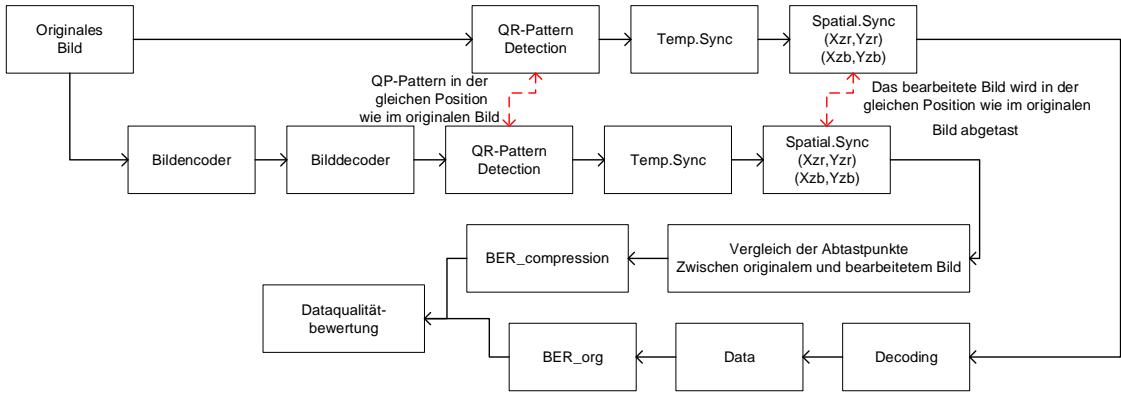


Abbildung 4.2: Aufbau der Experiment und Bewertung der Ergebnisse.

decodiert werden, weil sich die Qualität nach Komprimierung weiter verschlechtern lassen. Manche Daten wurde defekt geworden. Daher wird die BER für komprimierte Daten durch Vergleich der Abtastpunkt in beide Videos ausgerechnet. Die Punkte lassen sich in Mitte jedes Datenblocks im Abstand der Blockgröße abtasten. Die folgende Abbildungen zeigt die BER, die in vier Bitrate und Szene herauskommt. Die Abbildung 4.3 zeigt BER der Datenframes, die

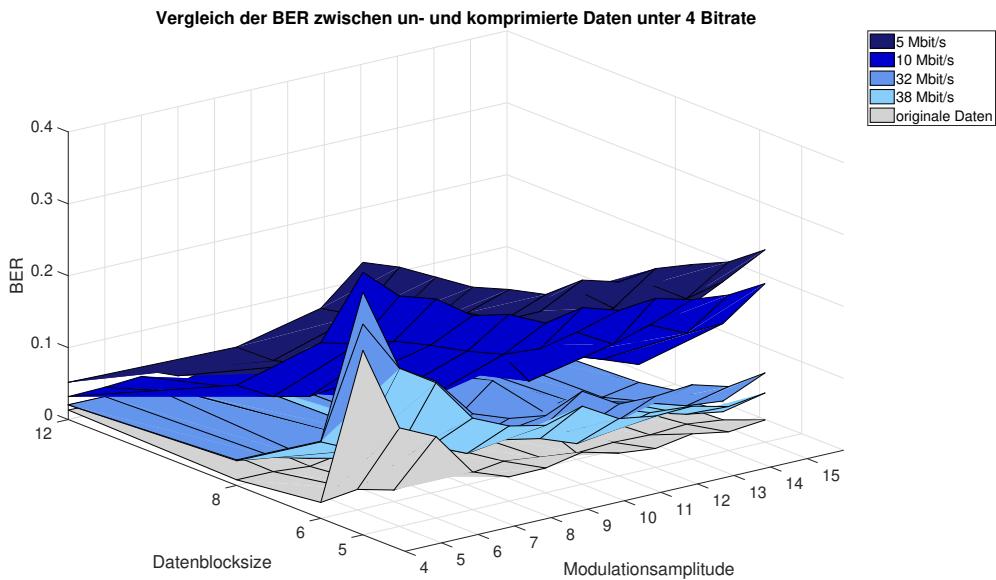


Abbildung 4.3: BER mit schnell bewegtem und hellem Hintergrund.

beim Abspielen eines Videos mit schnell bewegtem und hellem Hintergrund übertragen werden. Die BER abhängig von vergrößerter Datenblock (für spätere Erklärung mit Abkürzung *BS* geschrieben) und erhöhte Modulationsamplitude (für spätere Erklärung mit Abkürzung *A* ge-

schrieben) verhaltet in abgeschwächtem Tendenz. Das ursprüngliches Datenframe bei $BS = 5$ und $A = 4$ (in 4K-Format: $= 10$) hat den größte Fehler, gegen 20%. Wenn eine Quellencodierung bei der gleiche Situation durchgeführt wird, steigt der Fehler bis zum c.a 40% (unter Bitrate 5 Mbit/s). Wenn sich BS von 5 bis 6 vergrößert, verhaltet der Fehler einen schnell abnehmende Tendenz in allen Fällen. Ab $BS = 8$ bleibt der Fehler ganz niedrig unter Bitrate 32 Mbit/s und 38 Mbit/s. Der Fehler ist jedoch bis zu 20% mit $BS = 8$ unter Bitrate 5 Mbit/s. Je höher A eingegeben wird, desto geringer tritt der Fehler auf. Bei kleiner BS und niedrig A verhalten die Daten ähnlich wie Rauschen. Daraus verursacht dieser Fall niedrige SNR und führt zu hohe Fehler. Zusätzlich wird BER in Abhängigkeit von steigerte Bitrate niedriger. Unter Bitrate 32Mbit/s und 38Mbit/s ist der Fehler schon wenig und unterscheidet sich nicht so große wie unter niedriger Bitrate. Die Abbildung 4.4 zeigt die BER der Datenframes mit relativ

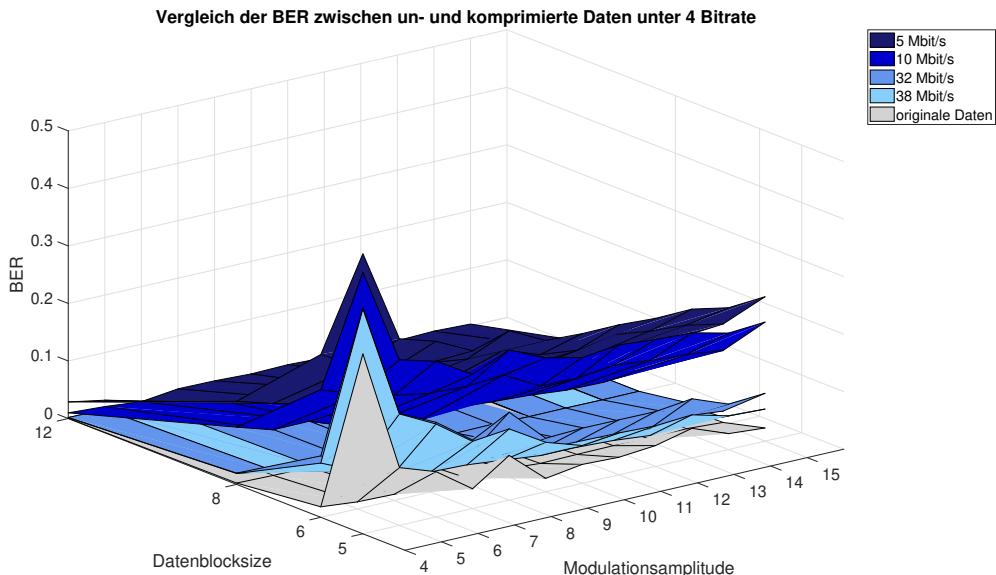


Abbildung 4.4: BER der Datenframes mit langsam bewegtem und hellem Hintergrund.

langsam bewegt und hellem Hintergrund. Mit der erhöhte A und vergrößerte BS nimmt die BER ab. Wenn sich BS von 5 auf 8 vergrößert, wird BER stark reduziert. Für die komprimierte Daten ergibt sich eine relativ kleine BER ab $BS \geq 8$, maximal bis 2% unter Bitrate 32 Mbit/s. Wenn $BS = 8$ ist, weist eine niedrige BER mit $A = 9$; Gleich wie vorher zwei Abbildungen gezeigt , sinkt BER (siehe Abb. ??) mit abnehmende A und BS . Wenn $BS \geq 8$ 8 ist, bleibt BER sehr klein (kleiner als 2%, unter Bitrate 32 Mbit/s), besonders von $BS = 5$ bis $BS = 8$ geht BER schnell zurück. Wenn sich die Bitrate verringert, verursacht eine abnehmende BER. Bei $BS = 12$ ergibt sich eine ganz niedrige BER unter allen Bitrate. Der Verhalten in Abbildung 4.6 ist ähnlich wie in Abbildung 4.5. Wenn BS größer gleich 8 und Bitrate höher gleich 32 Mbit/s ist, besitzt die Datenframes eine sehr kleiner BER. Aber die BER bleibt auch relativ hoch, wenn

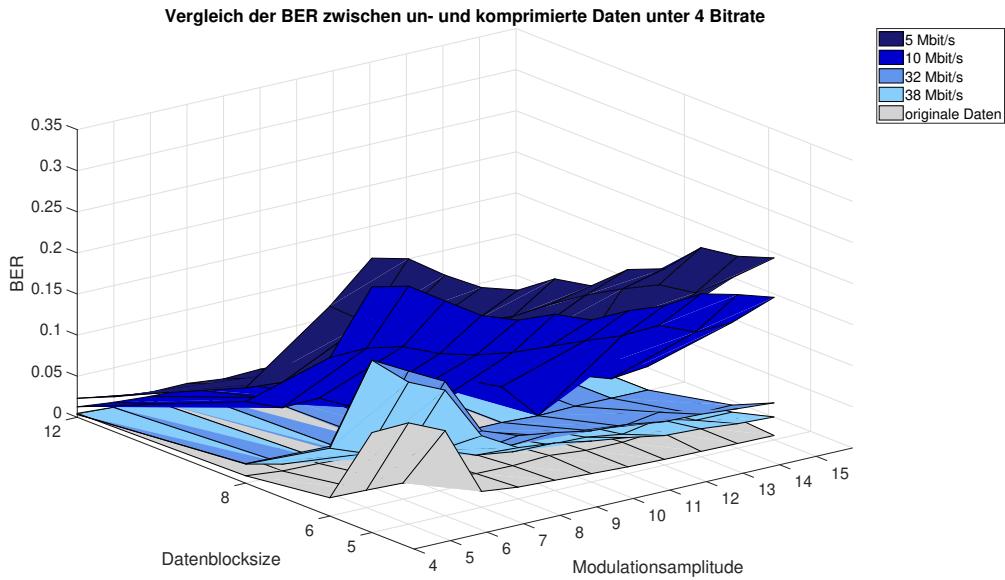


Abbildung 4.5: BER der Datenframes mit schnell bewegtem und dunklem Hintergrund.

die Bitrate niedrig ist, von 25% bis 30% unter 5 Mbit/s und 10 Mbit/s mit $BS = 5$ und $A = 4$. Bei Bitrate 5 Mbit/s wird eine größte BER in alle Szene angezeigt. Je höher der Bitrate ist, desto geringer Fehler auftritt. Auch bei $BS = 12$ ist der Fehler sonst unter Bitrate 5 Mbit/s auch niedrig.

PSNR

PSNR ist auch eine wichtige Kenngröße zur Bewertung der Bildqualität. Wie in der Simulation und letztem Experiment wird PSNR für Datenframes vor und nach der Kompression für Smartphone evaluiert. Die Abbildung 4.7 zeichnet die PSNR von Datenframes in U-Kanal mit schnell bewegtem und hellem Hintergrund. Mit der steigende A und abnehmende Bitrate ermäßigst sich die PSNR. Bei $BS = 5$ verhaltet PSNR nicht eine stabile reduzierte Tendenz, sondern von der Amplitude stark beeinflusst. Es gibt eine schnelle Reduktion von PSNR mit $A = 10 \text{ und } 13$ und $BS = 5$, während bei $A = 9 \text{ und } 11$ wieder ein bessere PSNR angezeigt wird. Für Änderung der BS von 6 bis 12 verbessert sich die PSNR langsam, allerdings von $BS = 5$ bis $BS = 6$ verschlechtert sich die PSNR relativ schneller. Außerdem, wenn die Bitrate nimmt ab, fällt PSNR gleichfalls, besonders bei 5Mbit/s und auch in kleiner BS und A , nur 25,7 dB. Die Abbildungen für alle andere drei Szene werden im Anhang A.5, A.6 und A.7 gezeigt. In der Abbildung A.5 zeigt eine schlechte PSNR bei $BS = 5$ mit $A = 8, 10, 13$. Wählen ber alle dunklen Szene ist die Änderung der PSNR nicht so schwankend, sonder bleibt relativ stabil. Die PSNR in allen Szene in V-Kanal hat fast ähnliche Verhalten wie in U-Kanal.

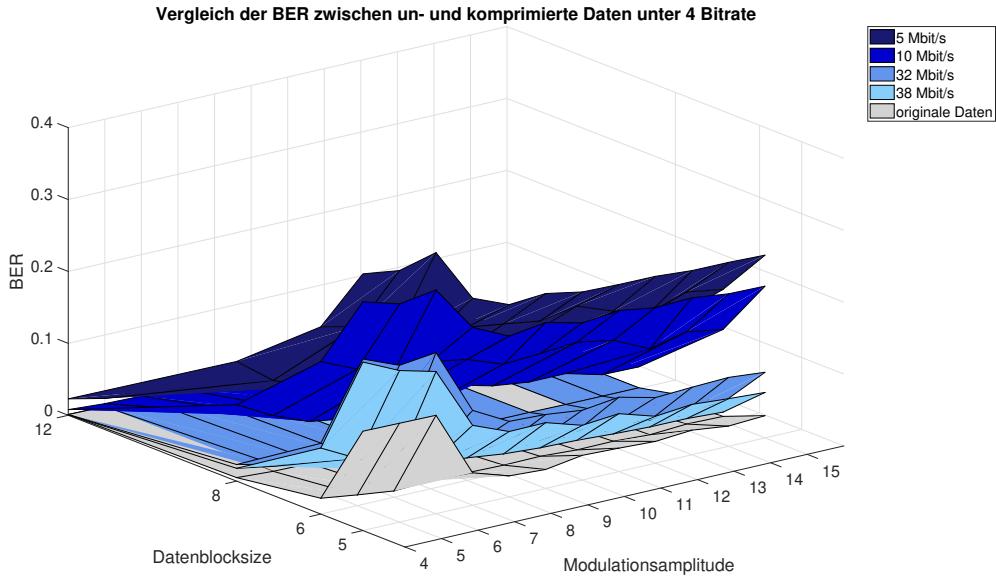


Abbildung 4.6: BER der Dataframes mit langsam bewegtem und dunklem Hintergrund.

Vergleich der Blöcke unterschiedlichem Bereich in einem Bild

Gleich wie die Experiment mit Industriekamera werden die Blöcke auch in verschiedenen Bereich ausgewählt und die Differenzbild sowie Histogramm angezeigt. Laut der in vorher analysierte Ergebnisse von PSNR und BER werden die Blöcke bei $BS = 8$ und $A = 8$ unter Bitrate 5 Mbit/s und 38 Mbit/s, $BS = 5$ und $A = 15$ unter Bitrate 38 Mbit/s, $BS = 12$ und $A = 4$ unter Bitrate 5 Mbit/s ausgewählt. Die Abbildung 4.8 zeigt die ausgewählte Blöcke (Blockgröße 100) von Videoframes mit hellem und schnell bewegtem Hintergrund. Die extrahierte Blöcke sind jeweils innerhalb einem Frame und aus gleicher Position vom nächst kommenden Frame. Die Eigenschaft dieser Bereiche werden in jedem Block oben bezeichnet. Unter Bitrate 38 Mbit/s erscheint, dass die Differenz zwischen komprimiertem und originalem Frame meistens aus der Kanteninformation besteht. Der Kanten zwischen zwei Datenblöcke sind die hochfrequente Information, die nach der Kompression eine große Menge gefiltert werden. Daraus verursacht einen unklaren Übergang zwischen zwei Datenblöcke. Zusätzlich sehen die Struktur der Blöcke von beide schnell bewegtem Bereiche nicht so klar wie die andere langsam bewegten Bereichen . Das Grund liegt darin, dass mehr Platz zur Speicherung der Bewegungsvektoren benutzt und dadurch mehr Information in Hochfrequenz unterdrückt wird. Daraus verursacht eine verschwommene Struktur, besonders bei der Kanten. Aus der Histogramme sind ebenfalls mehr Pixelwerte zu Null gezogen. Dies treten oftmals bei dem Übergang zwischen in Abbildung 4.8 gezeigter weißem und schwarzem Block. Um mehr Änderung dazwischen zu betrachten, sind noch die gleiche Blöcke vom Video mit 5 Mbit/s, 10Mbit/s und 32Mbit/s aus-

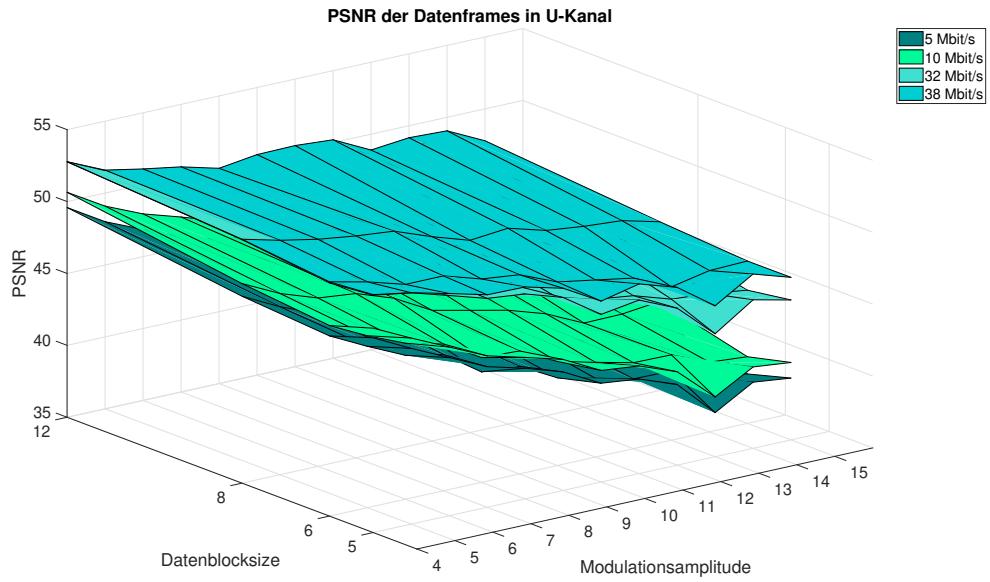


Abbildung 4.7: PSNR von Dataframes in U-Kanal mit schnell bewegtem und hellem Hintergrund.

genommen (siehe Anhang A.15, A.16 und A.17). Bei niedrige Bitrate wie 5Mbit/s und 10Mbit/s bleiben nur weniger Dateninformation, wobei sich die Struktur in alle ausgewählte Bereiche sehr unscharf geworden sind. Auffällig ist, dass große Menge der Datenwerte zu Null geschoben sind. Als nächst werden einige Blöcke aus der Datenframes mit dunklem und schnell bewegtem Hintergrund herausgezogen. Die Abbildung 4.8 zeigt die Blöcke unter Bitrate 38 Mbit/s. Im Vergleich zur Abbildung 4.8 sind bei dieser Szene, obwohl eine hoch Bitrate besitzt, weist dies schlechte Nachbildungen auf. Das Differenzbild enthält viele Struktur wie originalen Block. Die Differenz besteht nun nicht nur aus der Kanteninformation, sondern noch den Inhalten des originalen Datenframe. Aus der Histogramm ist es auch sehr sichtbar, dass die Pixelwerte nach der Kompression meistens zu Null geworden sind. Dadurch führt dies so viele unklare Struktur und Grauwerte im Block. In dieser Szene und unter Bitrate 5 Mbit/s sind alle Blockstruktur nicht erkennbar und fast keine Kanten zwischen Blöcke. Die Abbildung für die Bitrate 32 Mbit/s, 10Mbit/s und 5Mbit/s werden im Anhang A.18, A.19 und A.20 gezeigt. Um den Einfluss der eingegebene BS und A auf Kompression weiter zu betrachten, werden noch einige Abbildungen von $BS = 5$, $A = 15$ unter Bitrate 38 Mbit/s (siehe Abb. A.21) und $BS = 12$, $A = 4$ unter Bitrate 5 Mbit/s (siehe Abb. A.22) angezeigt. Die Strukturen der Daten werden sonst unter eine hohe Bitrate und mit hohe A auch sehr unscharf, während der Strukturen der Daten bei $BS = 12$ unter Bitrate 5 Mbit/s schon relativ klar sind.

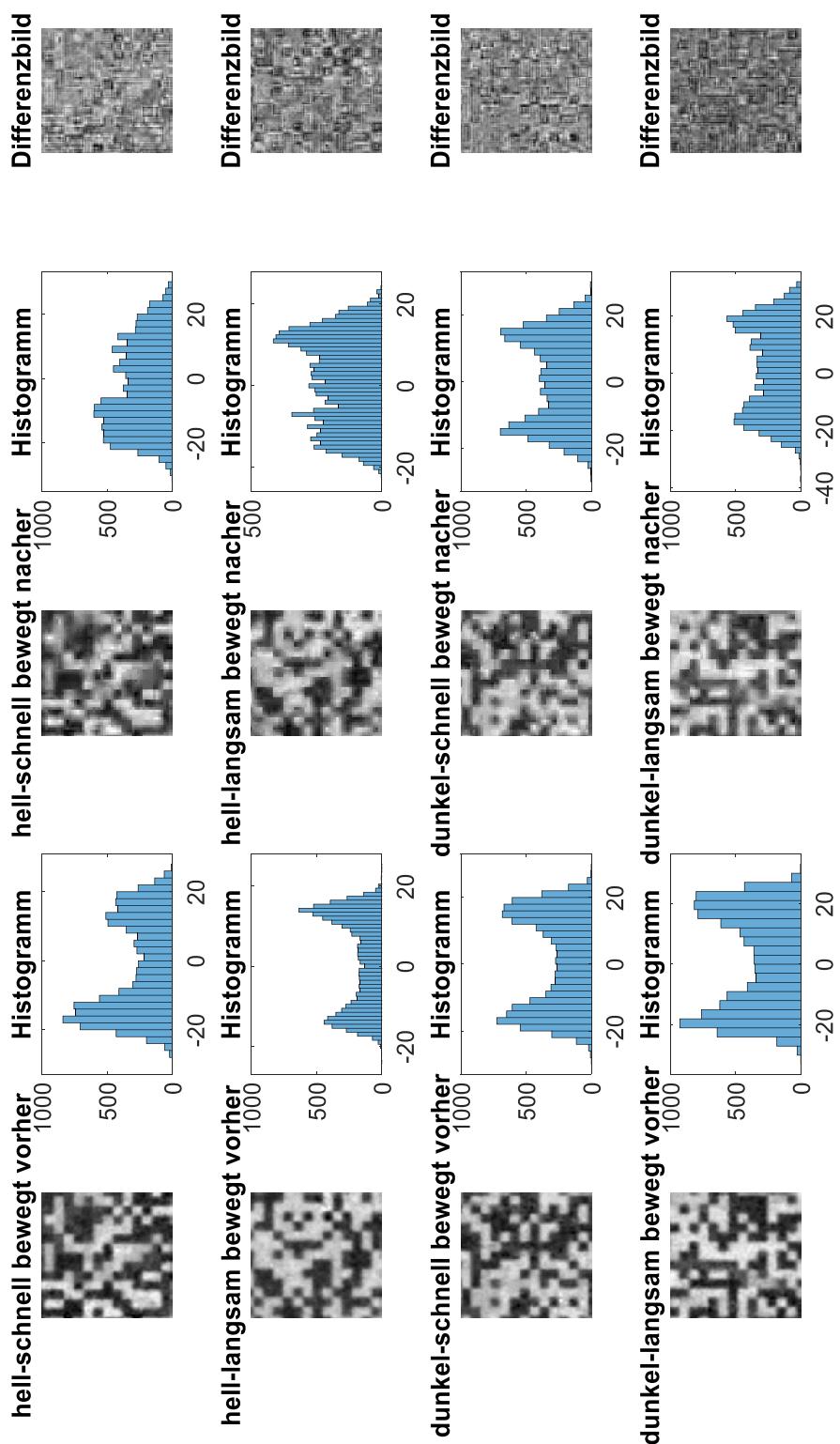


Abbildung 4.8: ausgewählte Blöcke aus Dataframes mit schnell bewegtem und hellem Hintergrund.

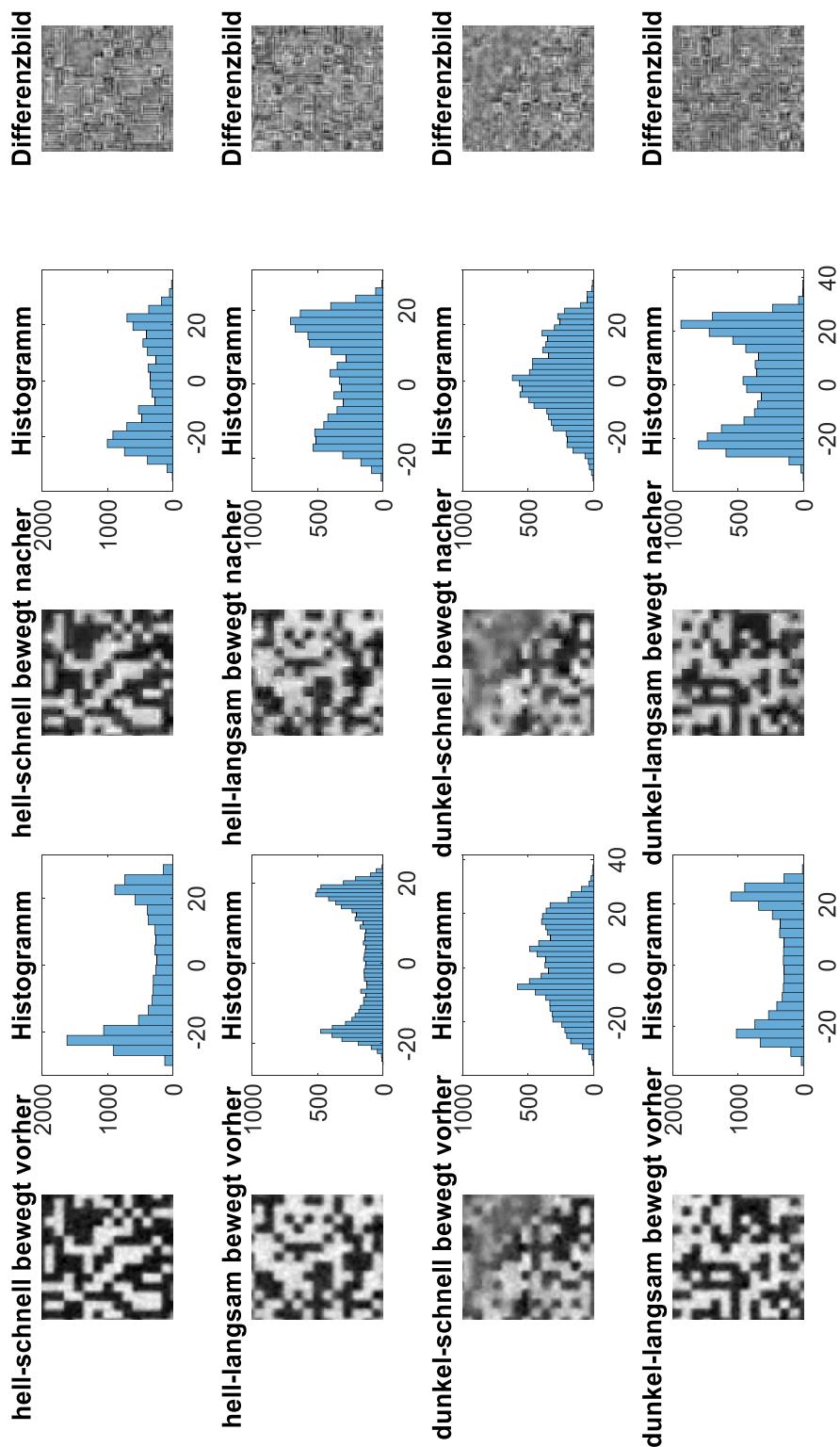


Abbildung 4.9: ausgewählte Blöcke aus Dataframes mit schnell bewegtem und dunklem Hintergrund.

Auswertung

Im Vergleich zum Experiment mit Industriekamera wird das Smartphonekamera nicht so hohe Auflösung besitzt. Daraus, besonders bei niedrige Bitrate tritt mehr Fehler auf, bei alle 4 Szenen schon über 20%. Dies führt Schwierigkeit für die Detektion der übertragene Datenframes. Mit kleiner BS und A , z.B. $BS = 5$ und $A = 4$ bis 6 ist der BER obwohl unter einer hohen Bitrate auch hoch. Bezugs auf der Simulation hat die Datenframes bei $BS = 5$ auch hohe Rekonstruktionsfehler. Deswegen ist die Datenframe mehr beeinflusst in dieser Blockgröße. Hingegen wird BER in allen Fällen ab $BS = 8$ deutlich gering. Bei manche plötzlich erhöhte BER, welche die benachbarte BER sehr klein sind, wird das Problem bei der Aufnahme von Raw-Datei erzeugt. Des Weiteren verzerrt die Datenstruktur in schnell bewegtem und dunklem Bereich stärker als die in relativ hellerem und langsamer bewegtem Bereich. Wenn die Blockgröße sehr groß (hier $BS = 12$) ist, kann eine niedrige BER sonder gute PSNR sowie klare Strukturen der Daten unter niedrige Bitrate erreichen werden. Dagegen ist bei sehr kleiner Datenblockgröße höhere Modulationsamplituden erforderlich. Dies kann allerdings auch nicht eine gute PSNR, niedrige BER und erkennbare Datenstrukturen wie mit großer Datenblockgröße erzielen. Wie die Untersuchung in der Simulation, dass nach der Quantisierung mehr Kanteninformation gefiltert wird, besonders unter hohe Kompression. Wenn die Blockgröße zu klein ist, wird der Abstand zwischen benachbarte Datenkanten sehr nahe. Dadurch wird das Signal mehr von Änderung der Kanten beeinflusst. Die ganze Struktur wird unscharf geworden und nach der Kompression schwierig erkannt.

5 Zusammenfassung

Zwei Quellencodierungsverfahren kommen in dieser Arbeit zum Einsatz, die JPEG-Codierung und H.264/ AVC. Zur Untersuchung des Einfluss unterschiedlicher Datenblockgröße, Modulationsamplitude und Bitrate auf der Quellencodierung, wird Skipt in Matlab zur Bewertung benutzt und je nach Änderung der oben genannte Parameters die resultierende Rekonstruktionsfehler, PSNR, BER und genaue Struktur von selbst ausgewählter Blöcke errechnet und verglichen werden. Durch der Simulation für Einzelbild, Standbilder und Bewegtbilder sind ungeeignete Parameters, die zur schlechte Ergebnisse führt, gefiltert. Wie z.B. die Datenblockgröße gleich 5 ist, tritt in alle Fälle mehr Fehler auf und verursacht schlechte PSNR. Außerdem ist die Daten mit dieser Blockgröße mehr von Amplitude beeinflusst. Hingegen in aller Simulation weist die Blockgröße ab 8 eine verbesserte PSNR und niedrige Fehler auf. Dies entspricht gerade auch die blockbasierte verarbeitete DCT-Codierung, sodass alle Energie innerhalb einem Datenblock am höchsten gebündelt wird. Wenn die Blockgröße in 8 festlegt, wird eine niedrige BER sowie Rekonstruktionsfehler bei Modulationsamplitude 7,9 und 11.

Für die Einzelbildcodierung, nämlich JPEG-Codierung, ist der Rekonstruktionsfehler bei Qualität 100 mit Blockgröße 8 auch bis zu ungefähr 10%. Obwohl die Codierung mit höchste Qualität durchgeführt, reduziert die Datenmenge noch um c.a 4.797 KB, Kompressionsverhältnis bis zu 6:1. Für die mit H.264/ AVC codierte Videosignale sowie Datensignale stellt eine hohe Rekonstruktionsfehler in der Simulation und BER in der Experiment dar, wenn die Bitrate niedrig (hier von 5 Mbit/s) bei kleiner Blockgröße von 3 bis 6. Auffällig ist, die Datenstrukturen sehr unklar bei kleiner Datenblockgröße. Weil die Kanten so nah gegen einander sind und die Signale in der Kanten häufig nach der Kompression unterdrückt werden, hat dies sehr starker Einfluss auf der ganzen Datenstrukturen. Dies erschwert die Detektierbarkeit von komprimierten Daten. Daraus hat der Auswahl von Datenblockgröße eine große Bedeutung. Des Weiteren ist die Bewertung der Bildqualität signifikant von der Bewegung und Kolorit der Videosignale beeinflusst.

Die Experiment mit Smartphonekamera ist nur mit Google Pixel XL durchgeführt und die festlegte Parameters passt allerdings nicht zu eine umfangreiche Smartphonekameras. In Zukunft wären mehr Smartphonekamera zur Experimente für eine umfassende geeignete Modulationsamplitude, Datenblockgröße und Bitrate denkbar. Außerdem werden die Codes zur

5 Zusammenfassung

Bewertung der Rekonstruktionsfehler und BER optimiert, um eine schnellere Rechen zu erzielen. Auch sollten sowohl mehr Bewertungsmethode als auch mehr Video mit unterschiedlicher Szene zum Einsatz kommen.

Anhang A

Erster Anhang

A.1 Abbildungen

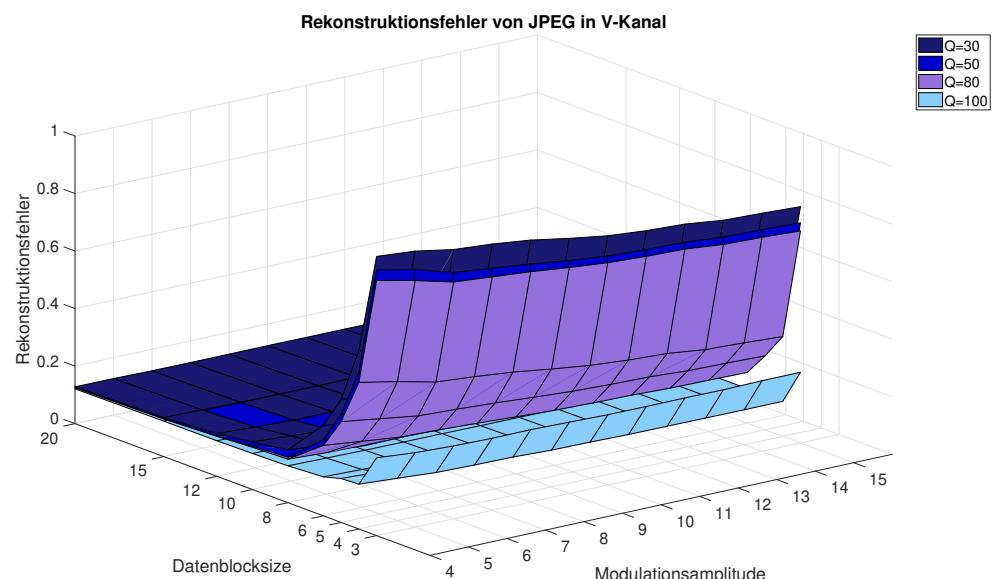


Abbildung A.1: Rekonstruktionsfehler von JPEG-Codierung in V-Kanal.

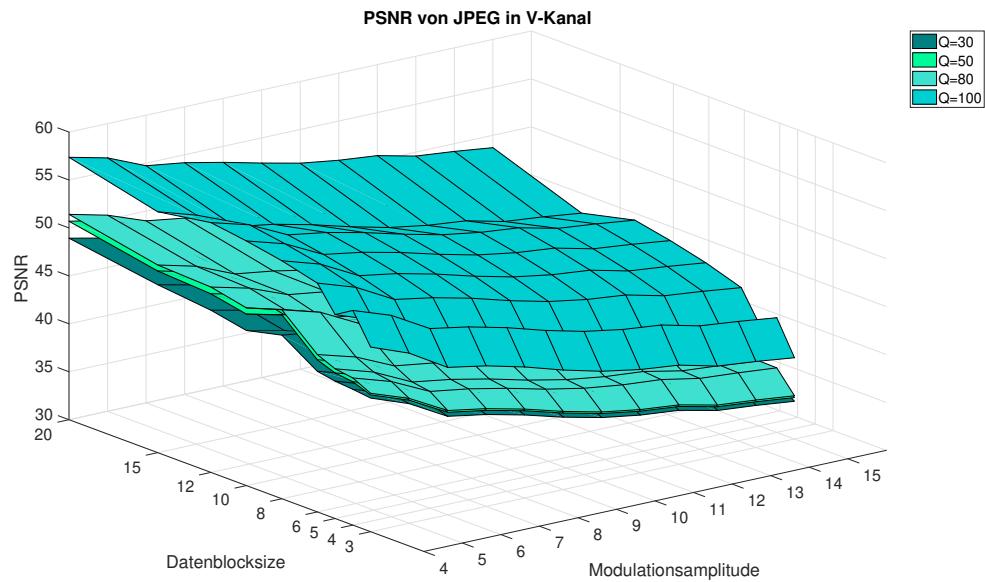


Abbildung A.2: PSNR von JPEG-Codierung in V-Kanal.

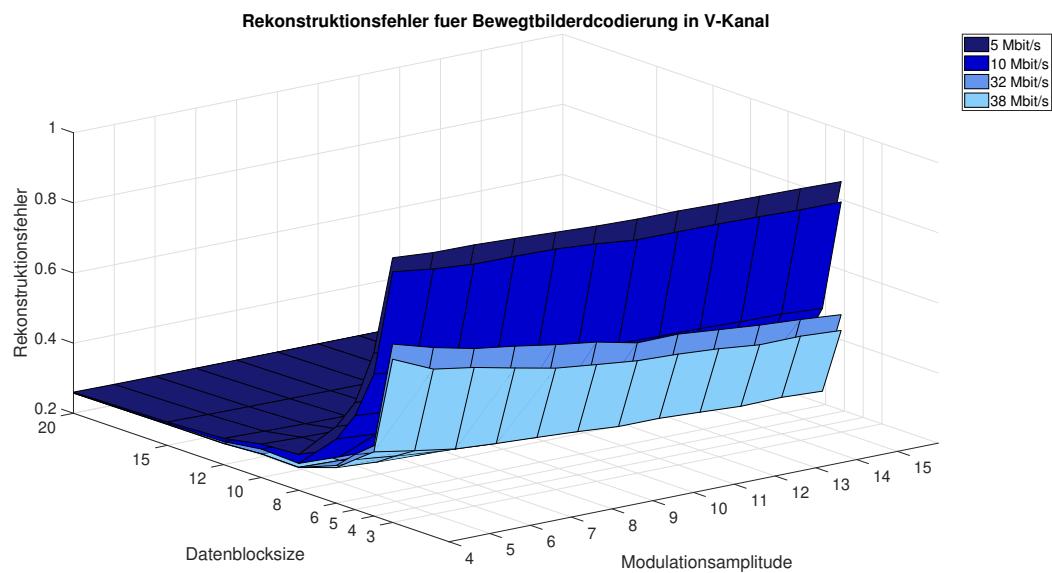


Abbildung A.3: Rekonstruktionsfehler in V-Kanal nach Verarbeitung in FFMPEG.

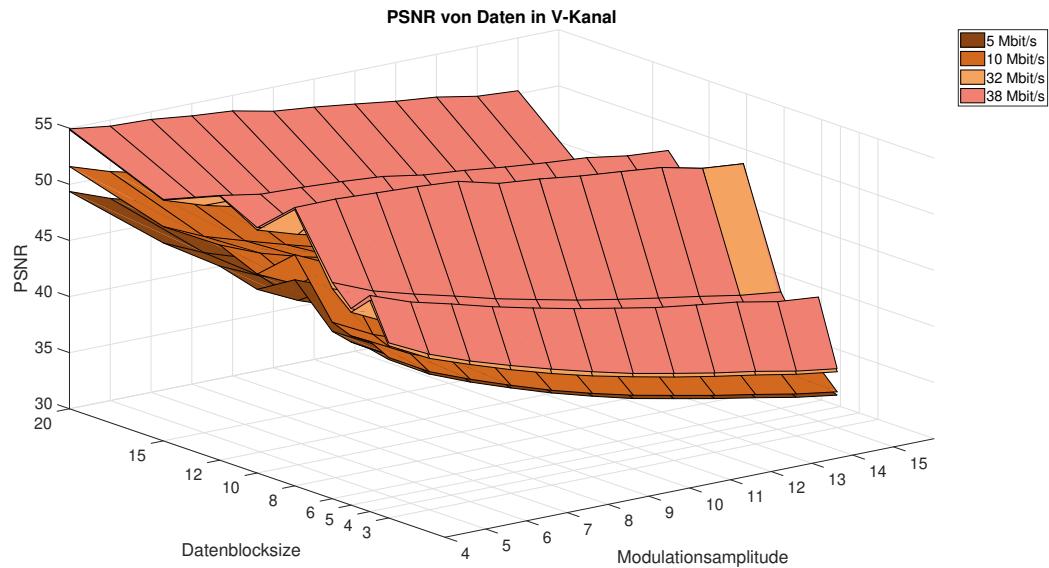


Abbildung A.4: PSNR von Daten in V-Kanal.

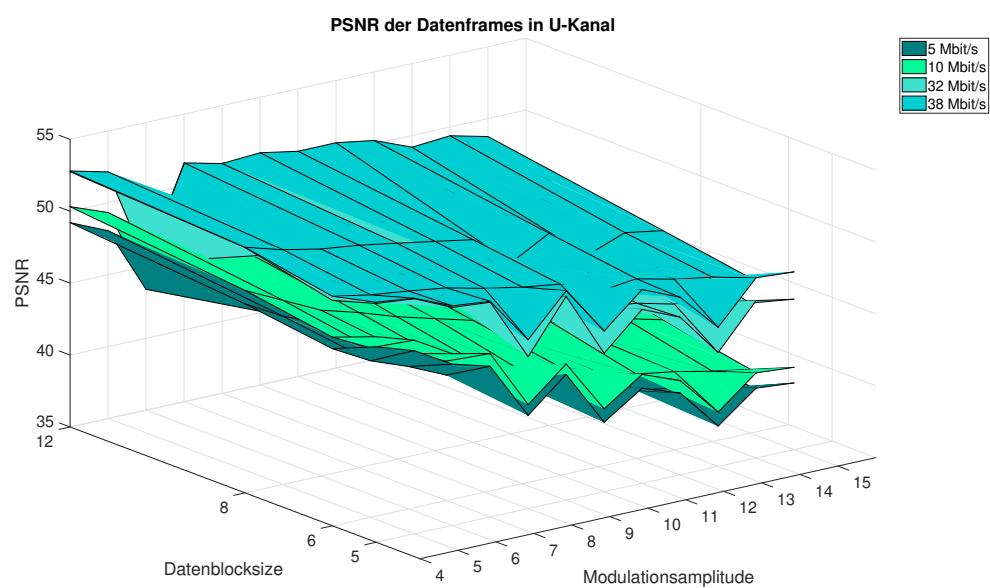


Abbildung A.5: PSNR von Dataframes in U-Kanal mit langsam bewegtem und hellem Hintergrund.

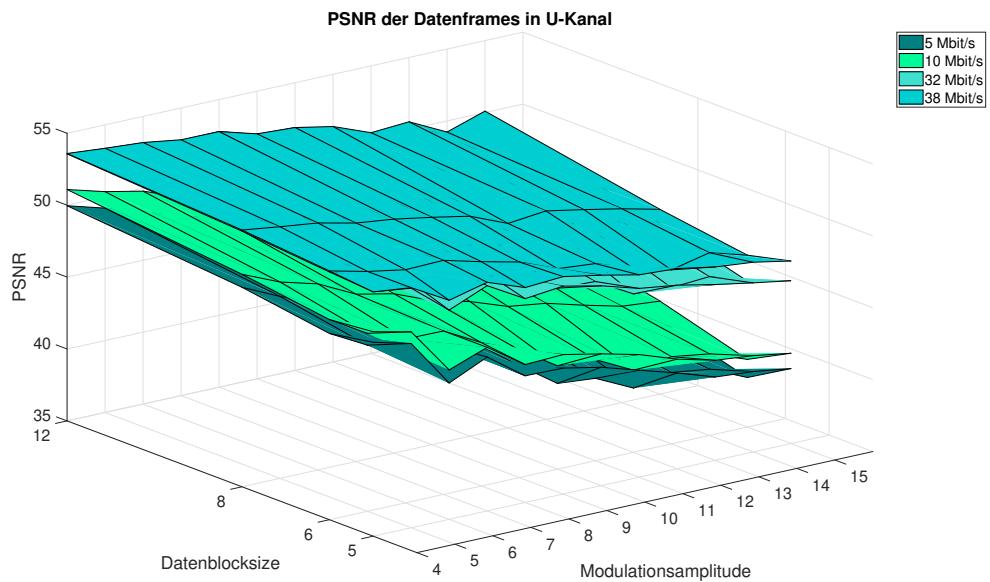


Abbildung A.6: PSNR von Dataframes in U-Kanal mit schnell bewegtem und dunklem Hintergrund.

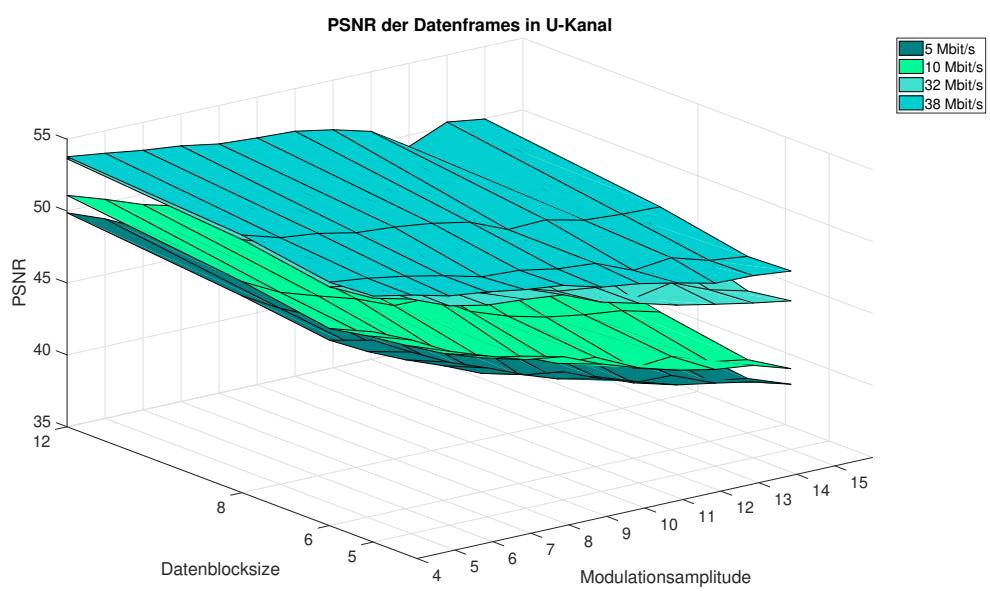


Abbildung A.7: PSNR von Dataframes in U-Kanal mit langsam bewegtem und dunklem Hintergrund.

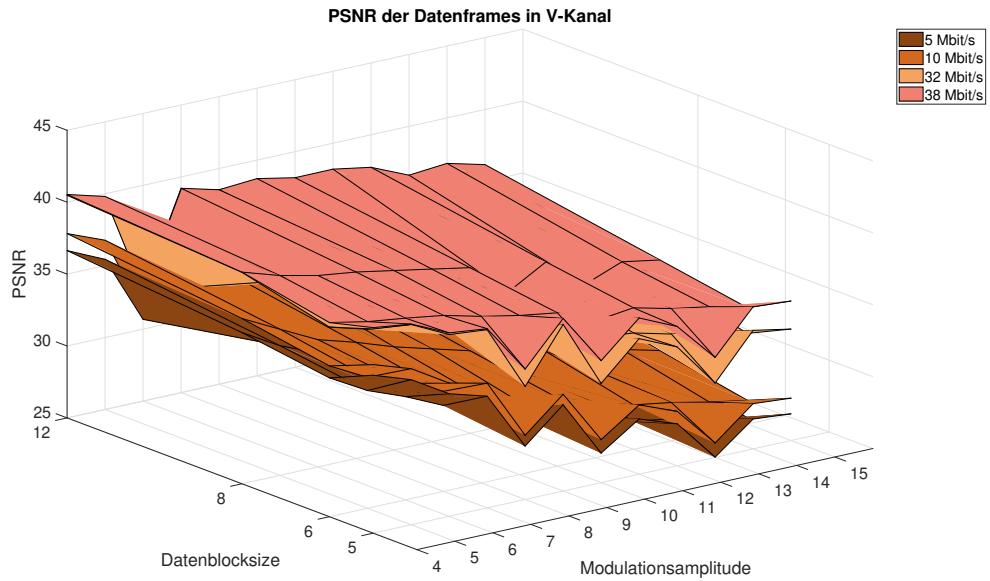


Abbildung A.8: PSNR von Dataframes in V-Kanal mit langsam bewegtem und hellem Hintergrund.

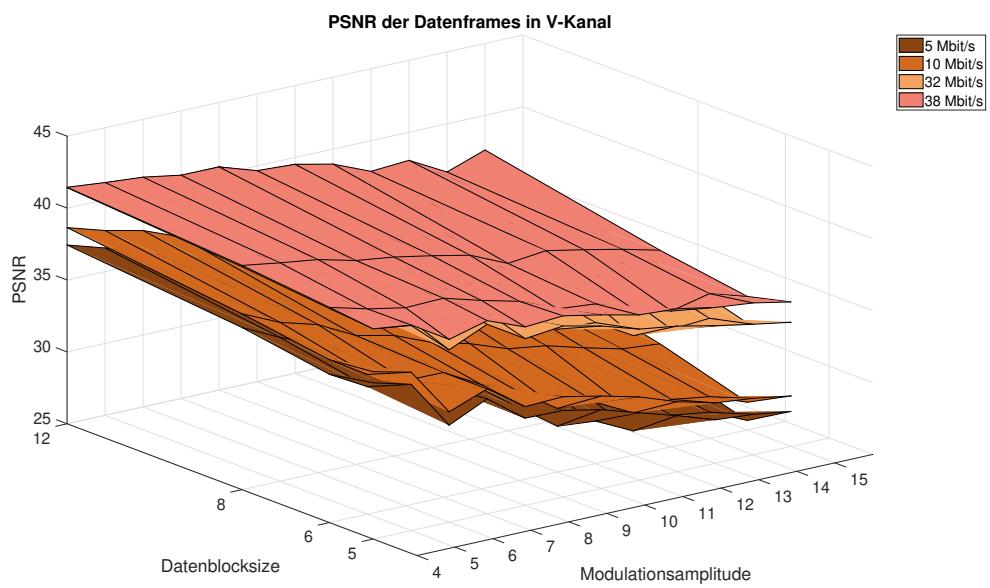


Abbildung A.9: PSNR von Dataframes in V-Kanal mit schnell bewegtem und dunklem Hintergrund.

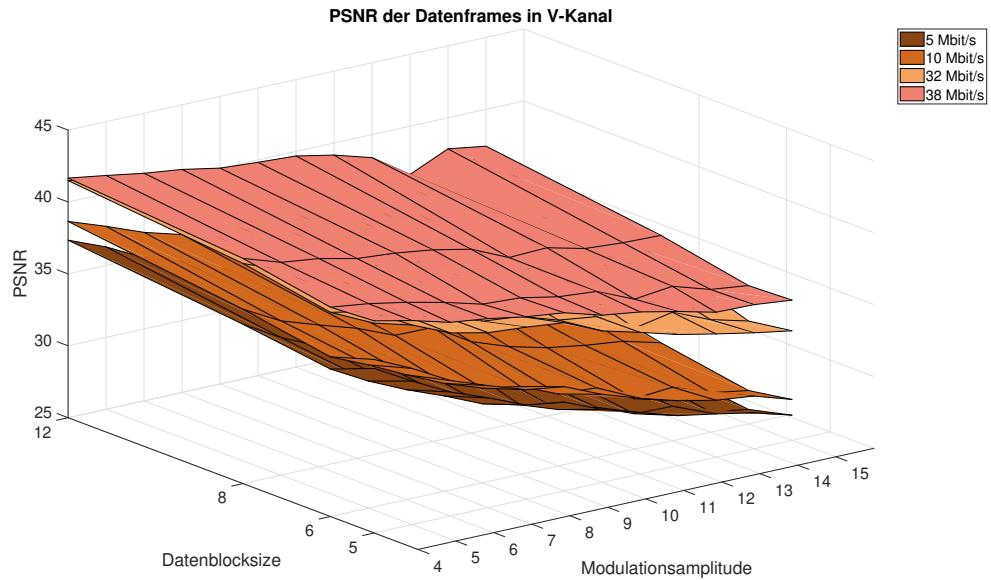


Abbildung A.10: PSNR von Dataframes in V-Kanal mit langsam bewegtem und dunklem Hintergrund.

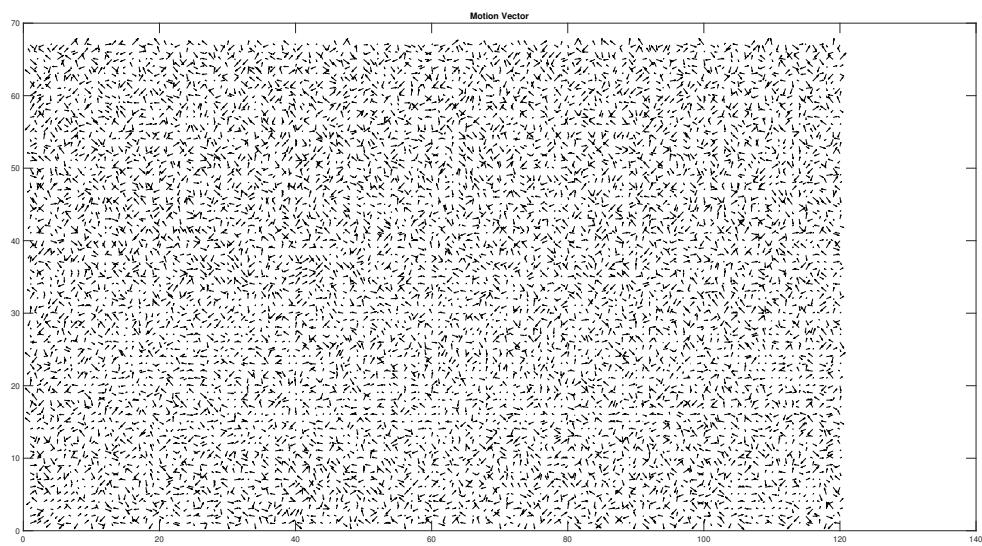


Abbildung A.11: Bewegungsvektoren von Bewegtbildern.

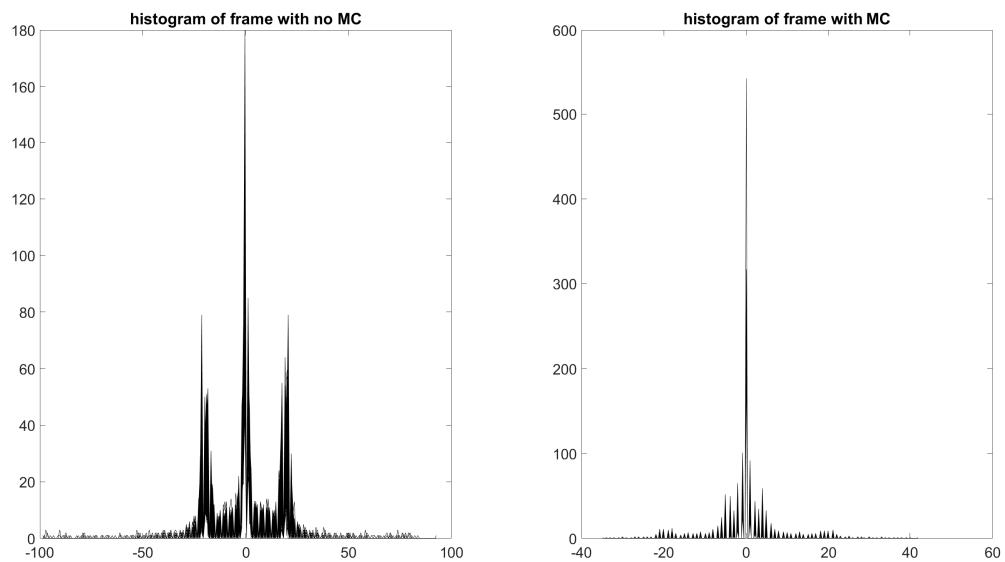


Abbildung A.12: Histogramm des Frames mit und ohne Bewegungsschätzung.

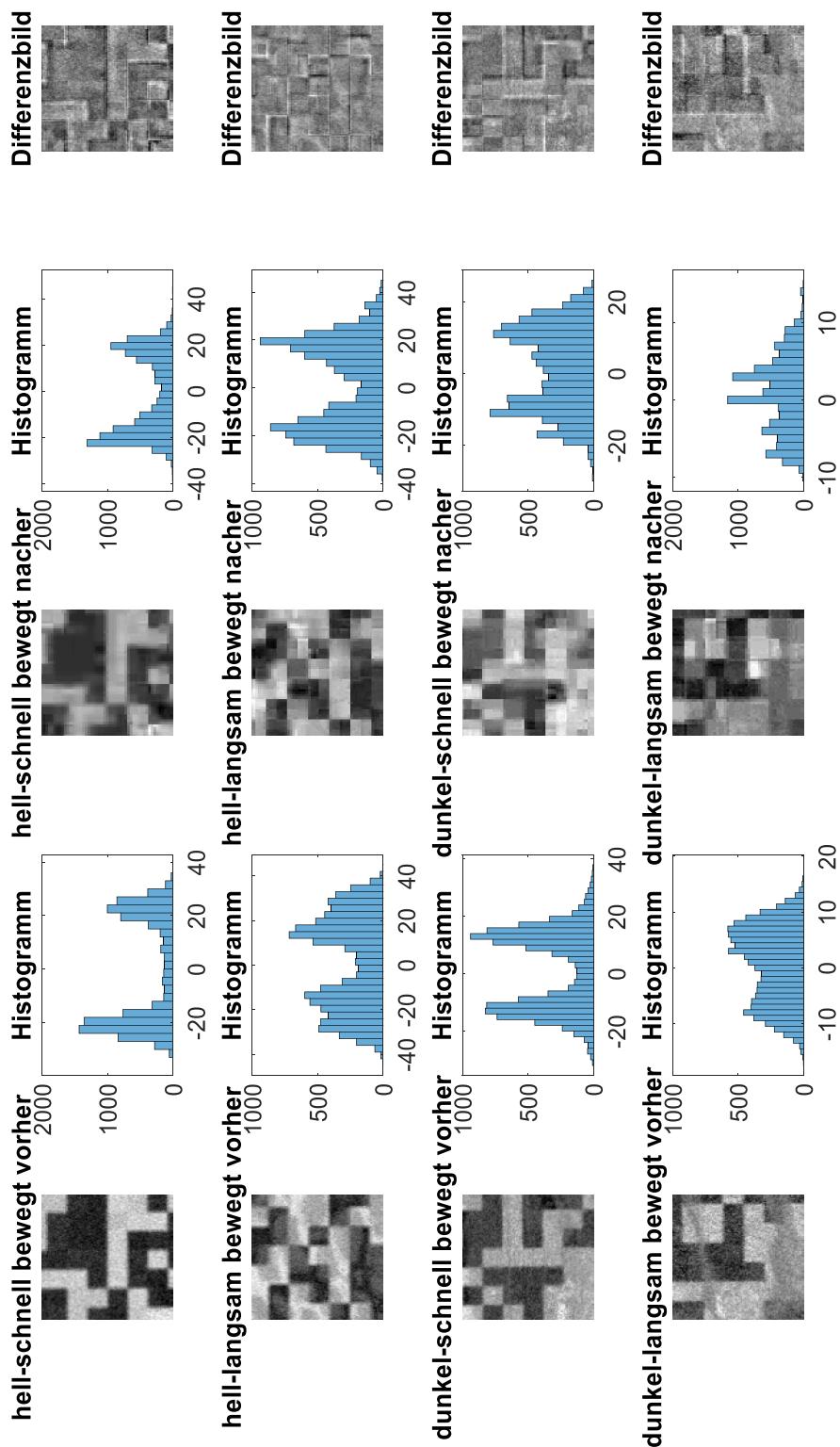


Abbildung A.13: ausgewählte Blöcke aus Dataframes mit hellem Hintergrund unter 5 Mbit/s.

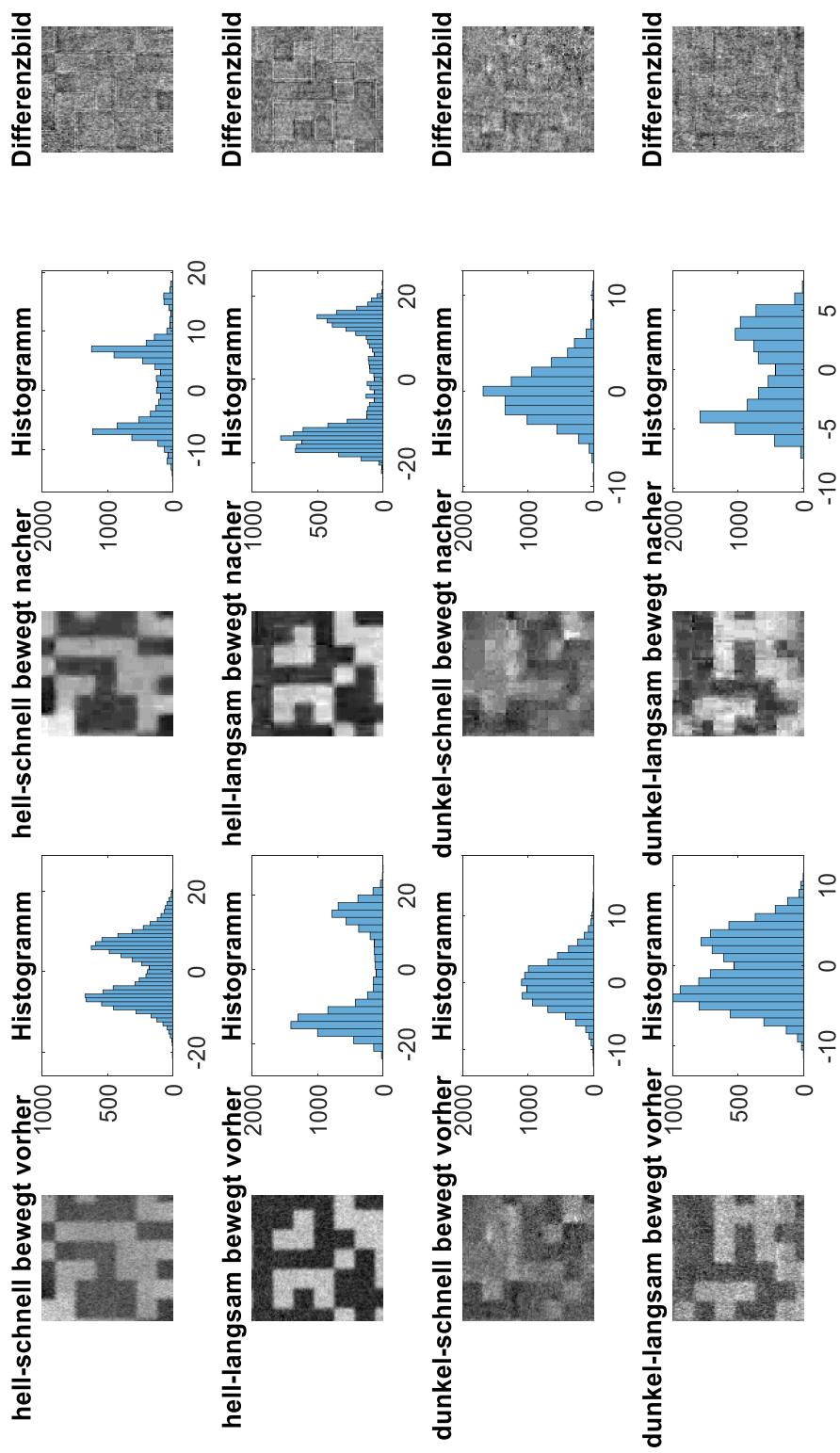


Abbildung A.14: ausgewählte Blöcke aus Dataframes mit dunklem Hintergrund unter 38 Mbit/s.

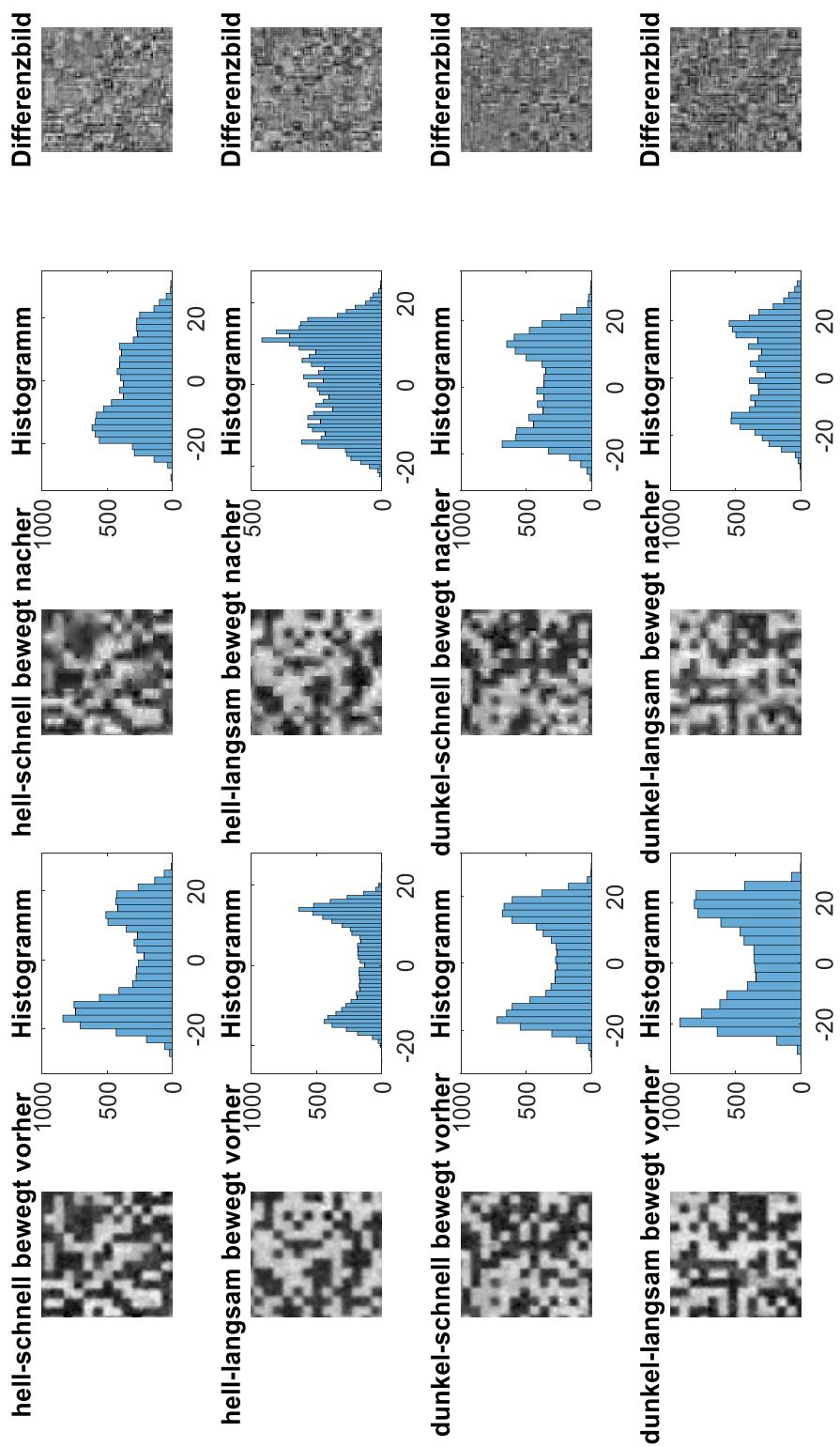


Abbildung A.15: ausgewählte Blöcke aus Dataframes mit langsam bewegtem und dunklem Hintergrund unter 32 Mbit/s.

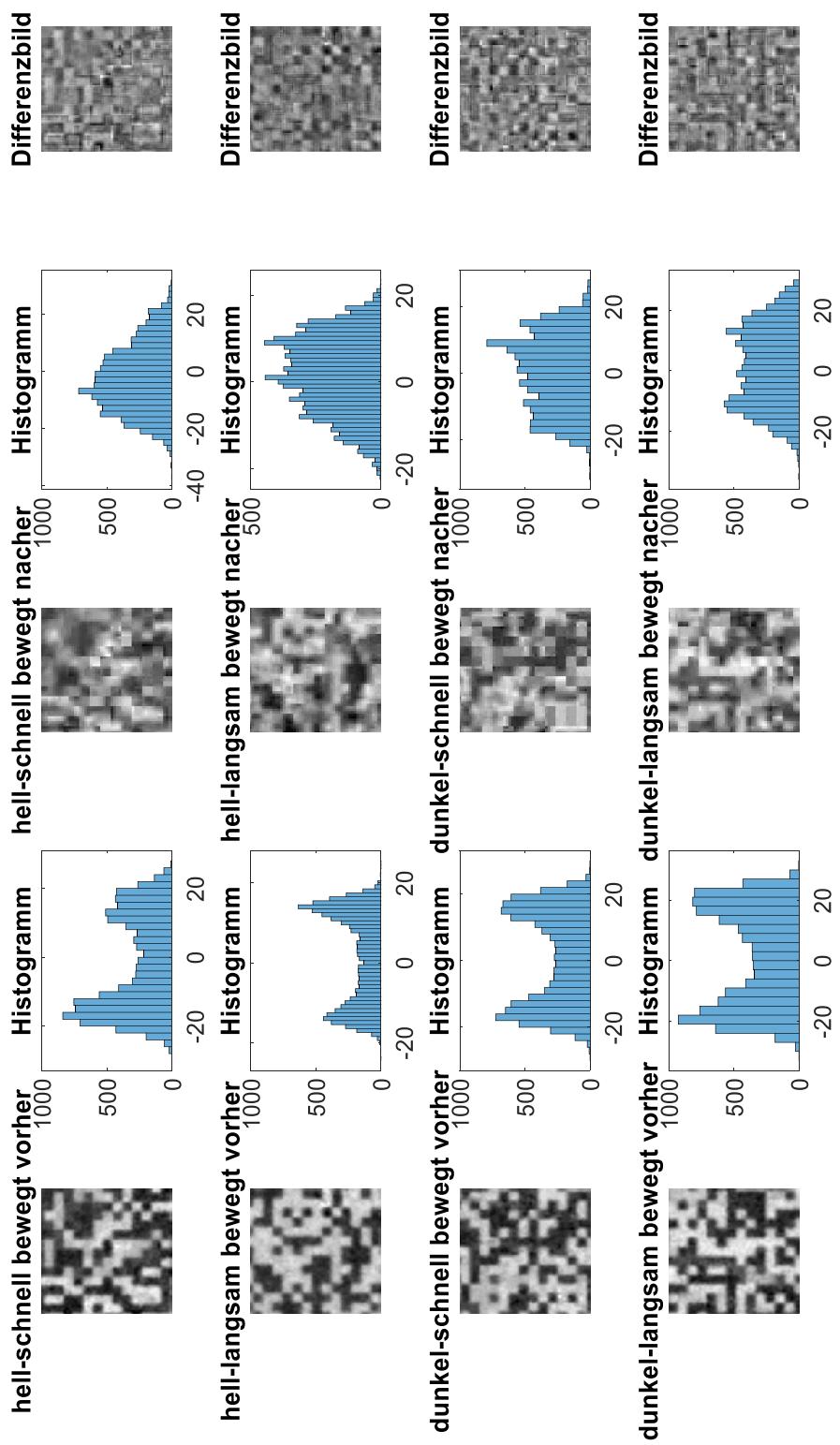


Abbildung A.16: ausgewählte Blöcke aus Dataframes mit langsam bewegtem und dunklem Hintergrund unter 10 Mbit/s.

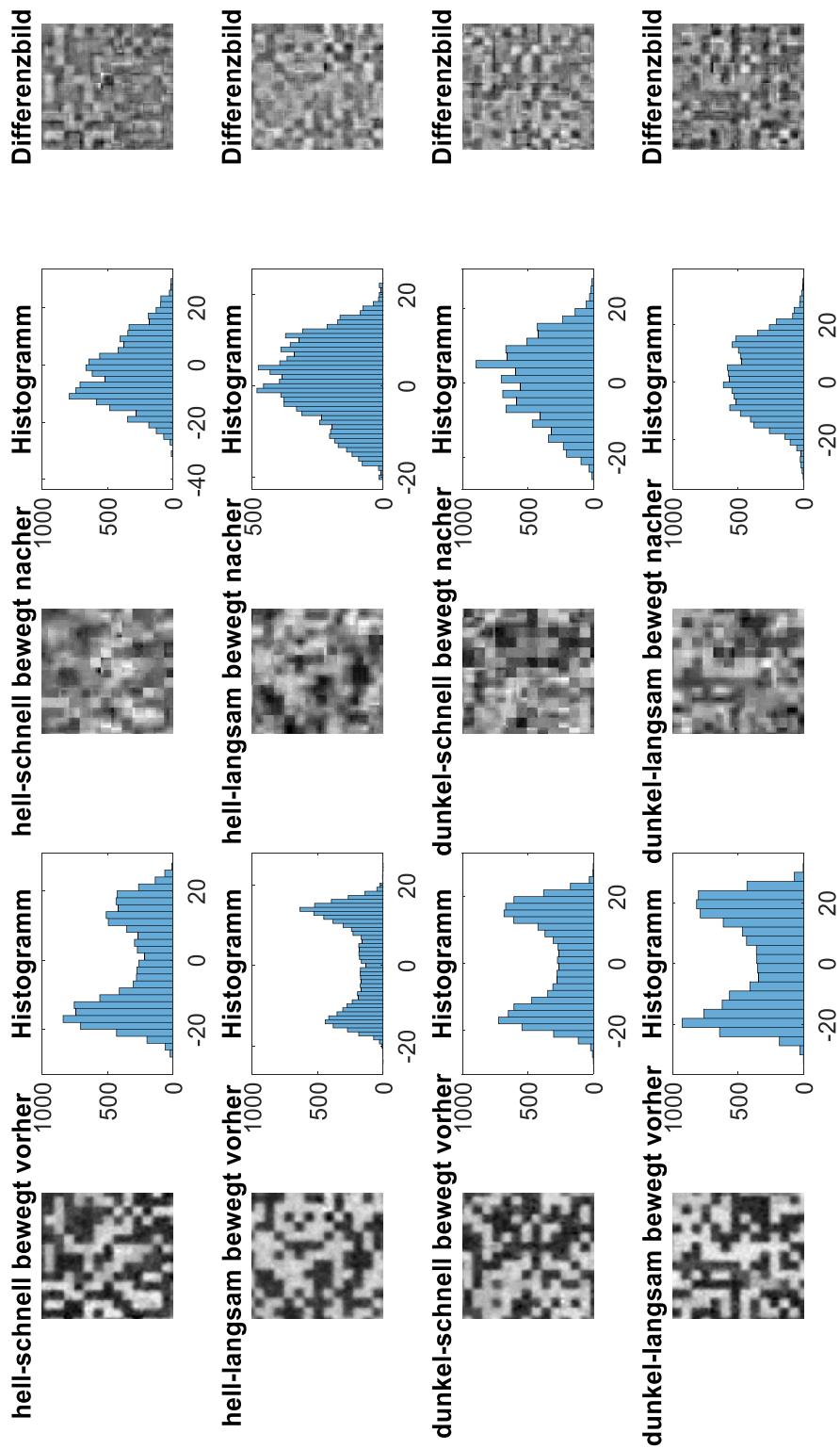


Abbildung A.17: ausgewählte Blöcke aus Dataframes mit langsam bewegtem und dunklem Hintergrund unter 5 Mbit/s.

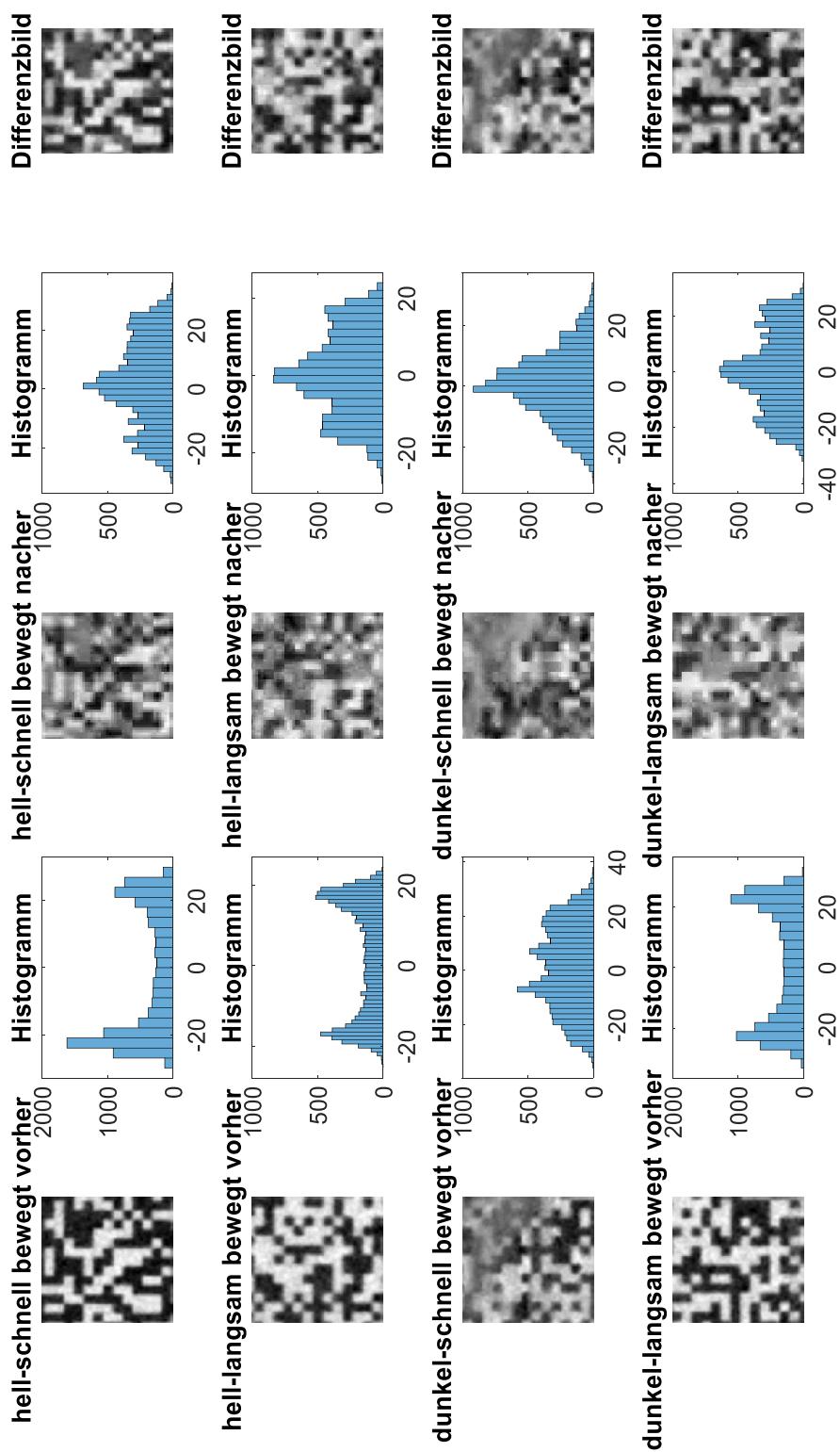


Abbildung A.18: ausgewählte Blöcke aus Dataframes mit langsam bewegtem und dunklem Hintergrund unter 32 Mbit/s.

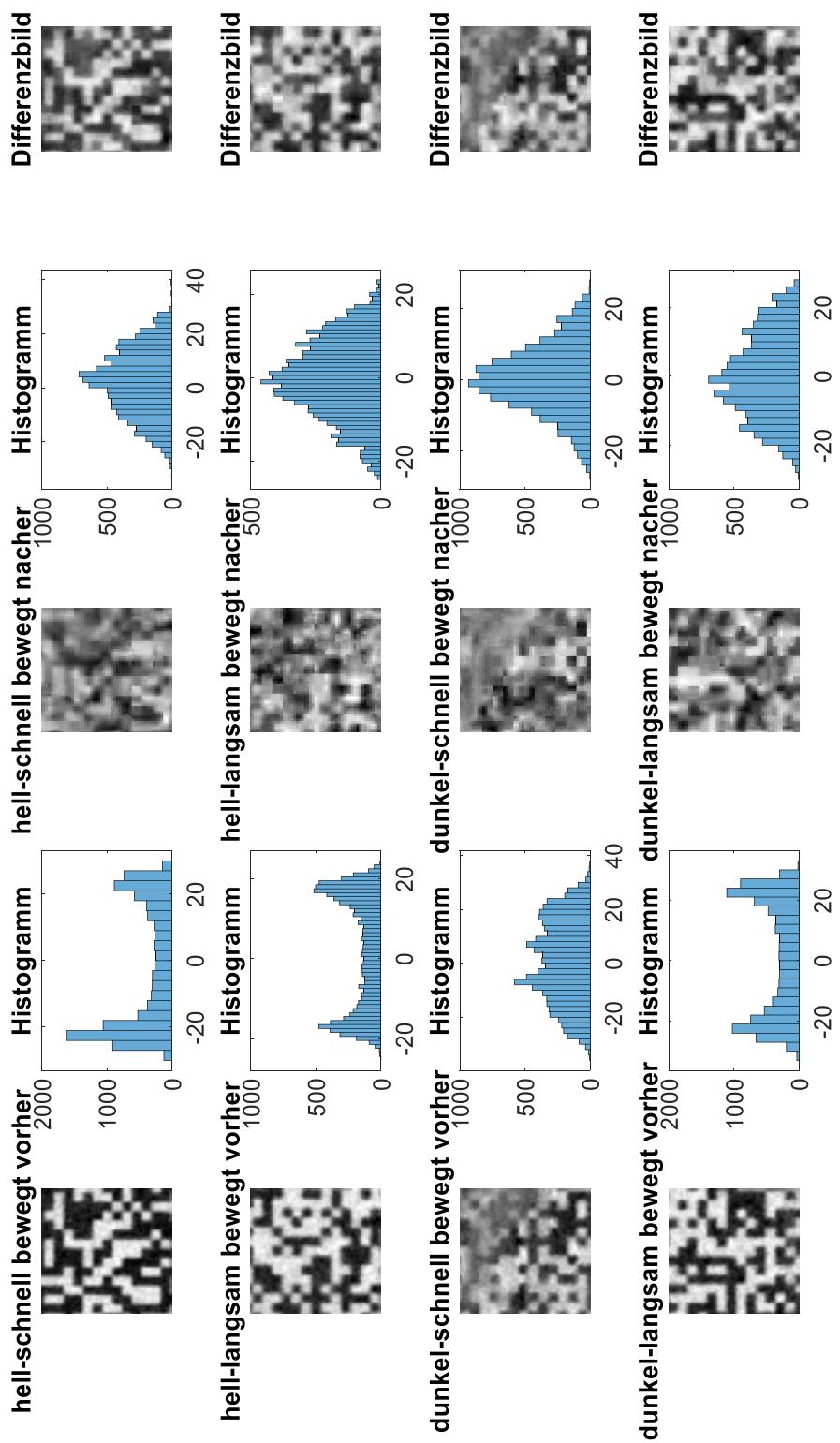


Abbildung A.19: ausgewählte Blöcke aus Dataframes mit langsam bewegtem und dunklem Hintergrund unter 10 Mbit/s.

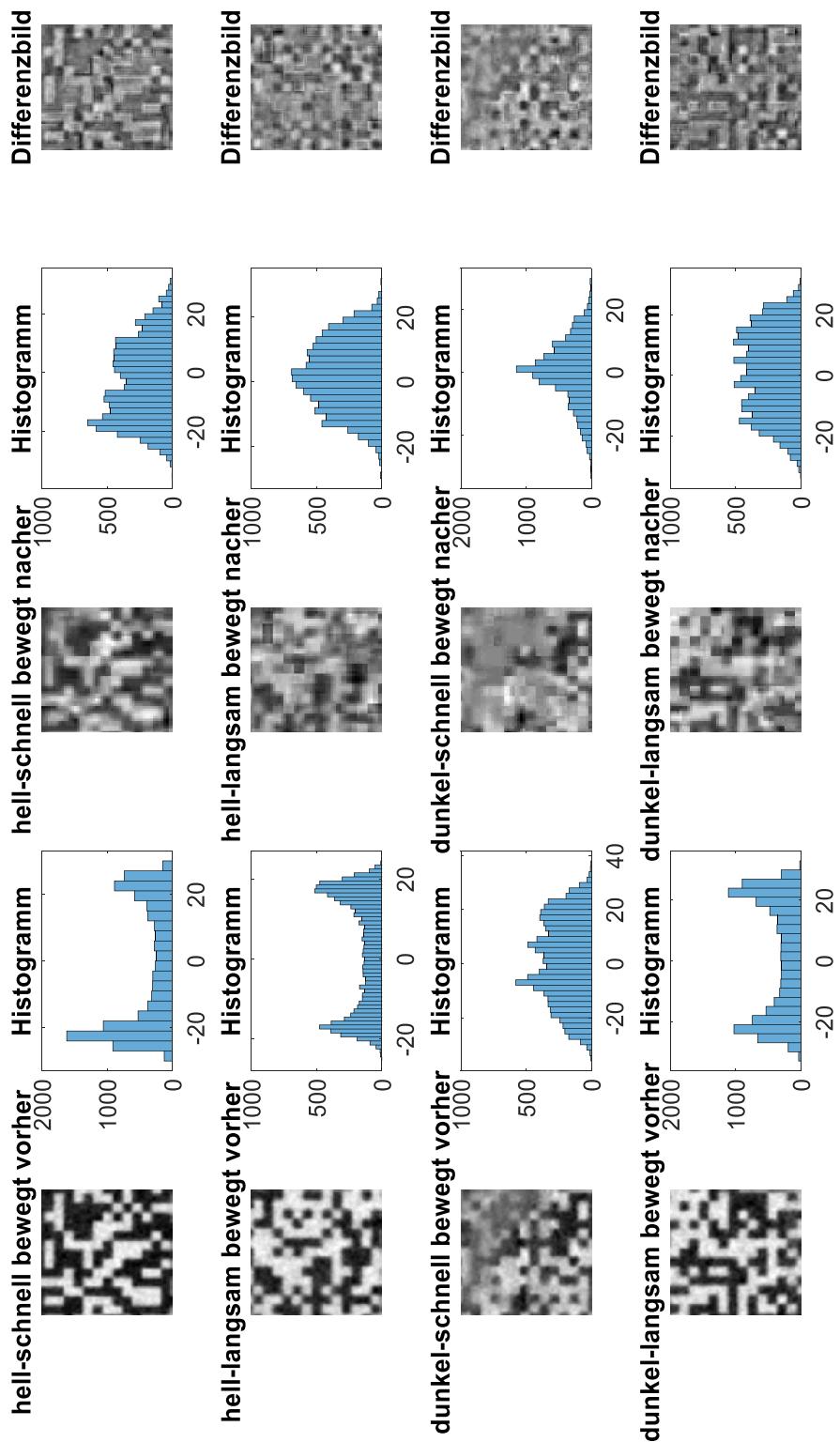


Abbildung A.20: ausgewählte Blöcke aus Dataframes mit langsam bewegtem und dunklem Hintergrund unter 5 Mbit/s.

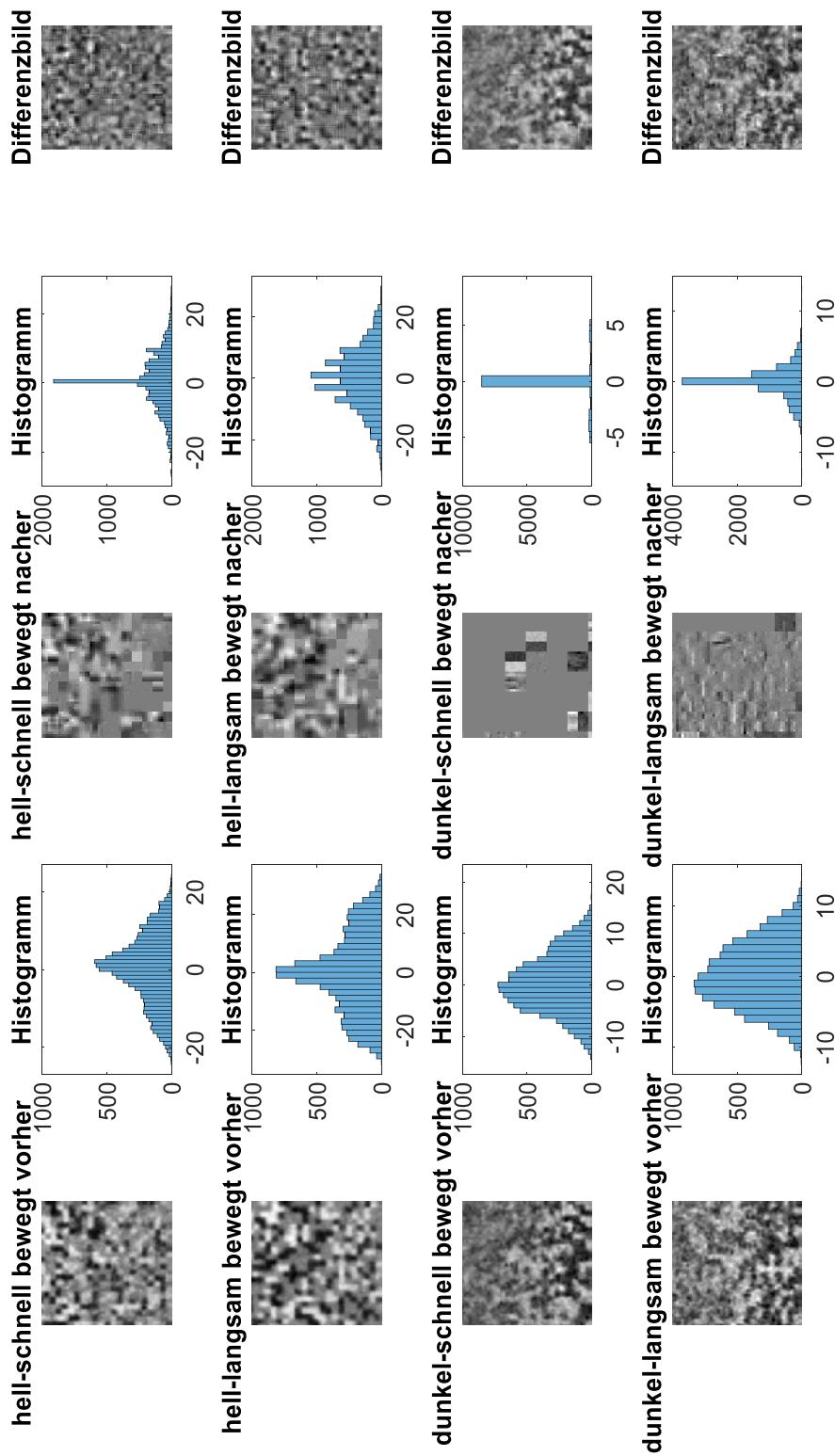


Abbildung A.21: ausgewählte Blöcke aus Dataframes mit langsam bewegtem und dunklem Hintergrund unter 38 Mbit/s mit $BS = 5$ und $A = 15$.

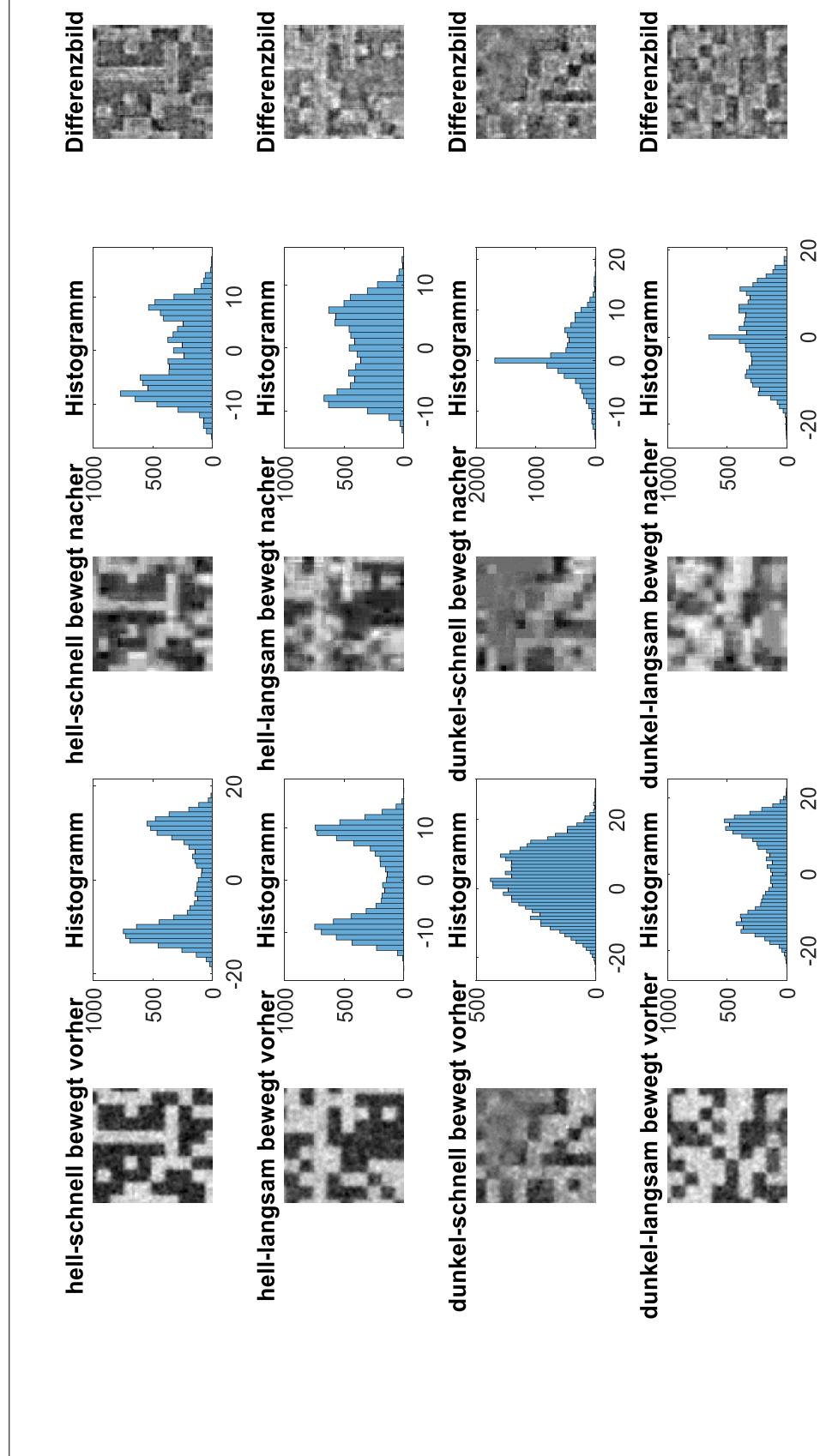


Abbildung A.22: ausgewählte Blöcke aus Dataframes mit langsam bewegtem und dunklem Hintergrund unter 5 Mbit/s mit $BS = 12$ und $A = 4$.

Abbildungsverzeichnis

2.1	Vektorielle Farbabaddition[4]	4
2.2	YUV-Abtastmodus	9
2.3	Ideale spektrale Antwortkurven von drei Zapfen	10
2.5	Beispiel zur Erklärung des Einflusses von Gradation	12
2.6	Gammavorverzerrung und Gammakorrektur beim Bildübertragungssystem . .	13
2.7	Gammavorverzerrung und Korrektur im Aufnahme- und Wiedergabesystem . .	13
2.8	Blockschaltbild der DCT-Codierung	15
2.9	Beispiel einer Blockbildung	16
2.10	Basisbilder nach der DCT	17
2.11	Quantisierungstabelle für MPEG	18
2.12	DPCM für DC-Anteile in der Standbildcodierung	19
2.13	Beispiel der Lauflängencodierung	20
2.14	typische Struktur einer GOP	22
2.15	Blockschaltbild der DPCM für Bewegtbildcodierung	23
2.16	Pirnzip des bidirektionalen Blockmatching	24
2.17	H.264/MPEG-4 AVC-Encoder	25
2.18	schematische Aufbau einer objektiven Qualitätsbewertung für übertragene Daten	27
2.19	schematische Aufbau von DaViD-System	28
3.1	zu codierendes Bild	31
3.2	Simulationsablauf	32
3.3	Rekonstruktionsfehler von JPEG-Codierung in U-Kanal	33
3.4	PSNR von JPEG-Codierung in U-Kanal	34
3.5	Dataframe mit und ohne Gammaverzerrung in jedem Kanal	35
3.6	Makroblockstruktur der MPEG Profile	36
3.7	spektrale Antwort der Bildsignale vor und nach der DCT	37
3.8	Änderung der Pixelwerte nach DCT	37
3.9	DCT-Koeffizienten nach Änderung der Blockgröße in U- und V-Kanal	38
3.10	DCT-Koeffizienten nach Änderung der Blockgröße in U- und V-Kanal	39
3.11	Histogramm der Daten vor-und nach Quantisierung	40
3.12	Einfluss der Kompressionsfaktor auf Quantisierung	41

3.13 Videoverarbeitung in FFmpeg	42
3.14 Rekonstruktionsfehler in U-Kanal nach Verarbeitung in FFmpeg	44
3.15 Rekonstruktionsfehler in V-Kanal nach Verarbeitung in FFmpeg	45
3.16 PSNR von Daten in U-Kanal	46
3.17 PSNR von Daten in V-Kanal	47
3.18 die entstehende Bewegungsvektoren	48
3.19 das erzeugtes Bild mit und ohne Bewegungsschätzung	49
3.20 Histogramm des Frames mit und ohne Bewegungsschätzung	49
3.21 Rekonstruktionsfehler in U-Kanal nach Verarbeitung in FFmpeg	50
3.22 PSNR von Daten in U-Kanal	50
3.23 das Differenzbild mit und ohne Bewegungsschätzung	51
3.24 durch Bewegungsschätzung entstehendes Bild	51
4.1 ausgewählte Blöcke aus Dataframes mit dunklem Hintergrund unter 5 Mbit/s	56
4.2 Aufbau der Experiment und Bewertung der Ergebnisse	58
4.3 BER mit schnell bewegtem und hellem Hintergrund	58
4.4 BER der Datenframes mit langsam bewegtem und hellem Hintergrund	59
4.5 BER der Datenframes mit schnell bewegtem und dunklem Hintergrund	60
4.6 BER der Dataframes mit langsam bewegtem und dunklem Hintergrund	61
4.7 PSNR von Dataframes in U-Kanal mit schnell bewegtem und hellem Hintergrund	62
4.8 ausgewählte Blöcke aus Dataframes mit schnell bewegtem und hellem Hintergrund	63
4.9 ausgewählte Blöcke aus Dataframes mit schnell bewegtem und dunklem Hintergrund	64
A.1 Rekonstruktionsfehler von JPEG-Codierung in V-Kanal	68
A.2 PSNR von JPEG-Codierung in V-Kanal	69
A.3 Rekonstruktionsfehler in V-Kanal nach Verarbeitung in FFmpeg	69
A.4 PSNR von Daten in V-Kanal	70
A.5 PSNR von Dataframes in U-Kanal mit langsam bewegtem und hellem Hintergrund	70
A.6 PSNR von Dataframes in U-Kanal mit schnell bewegtem und dunklem Hintergrund	71
A.7 PSNR von Dataframes in U-Kanal mit langsam bewegtem und dunklem Hintergrund	71
A.8 PSNR von Dataframes in V-Kanal mit langsam bewegtem und hellem Hintergrund	72
A.9 PSNR von Dataframes in V-Kanal mit schnell bewegtem und dunklem Hintergrund	72
A.10 PSNR von Dataframes in V-Kanal mit langsam bewegtem und dunklem Hintergrund	73
A.11 Bewegungsvektoren von Bewegtbildern	73
A.12 Histogramm des Frames mit und ohne Bewegungsschätzung	74

A.13 ausgewählte Blöcke aus Dataframes mit hellem Hintergrund unter 5 Mbit/s	75
A.14 ausgewählte Blöcke aus Dataframes mit dunklem Hintergrund unter 38 Mbit/s	76
A.15 ausgewählte Blöcke aus Dataframes mit langsam bewegtem und dunklem Hin- tergrund unter 32 Mbit/s	77
A.16 ausgewählte Blöcke aus Dataframes mit langsam bewegtem und dunklem Hin- tergrund unter 10 Mbit/s	78
A.17 ausgewählte Blöcke aus Dataframes mit langsam bewegtem und dunklem Hin- tergrund unter 5 Mbit/s	79
A.18 ausgewählte Blöcke aus Dataframes mit langsam bewegtem und dunklem Hin- tergrund unter 32 Mbit/s	80
A.19 ausgewählte Blöcke aus Dataframes mit langsam bewegtem und dunklem Hin- tergrund unter 10 Mbit/s	81
A.20 ausgewählte Blöcke aus Dataframes mit langsam bewegtem und dunklem Hin- tergrund unter 5 Mbit/s	82
A.21 ausgewählte Blöcke aus Dataframes mit langsam bewegtem und dunklem Hin- tergrund unter 38 Mbit/s mit $BS = 5$ und $A = 15$	83
A.22 ausgewählte Blöcke aus Dataframes mit langsam bewegtem und dunklem Hin- tergrund unter 5 Mbit/s mit $BS = 12$ und $A = 4$	84

Tabellenverzeichnis

2.1 Einige Videoformate	8
2.2 Einige Konfigurationen von Smartphones	10
3.1 Befehle von Video Codec in FFMPG	43
4.1 die in Experimente verwendete Geräte und deren Konfiguration	53
4.2 Videocodec und Parameters von aufgenommene Videos in Google Pixel XL . .	53

Quellcodeverzeichnis

Literaturverzeichnis

- [1] George Krebs. The Clock Is Ticking, year = 2011, url = <http://reboot.fcc.gov/blog?categoryId=840092.>, urldate = 2011-03-16.
- [2] Forscher schaffen mit weißer led 500 mbit/s.
- [3] Robert Snowden, Robert J Snowden, Peter Thompson, and Tom Troscianko. *Basic vision: an introduction to visual perception*. Oxford University Press, 2012.
- [4] R.kays. Bildkommunikation. 2016.

Eidesstattliche Versicherung

Wang, Ying

Name, Vorname

180854

Matr.-Nr.

Ich versichere hiermit an Eides statt, dass ich die vorliegende Masterarbeit mit dem Titel

Untersuchung des Einflusses der Quellencodierung auf verdecktübertragene Inhale bei der Screen-Camera Visible Light Communicati

selbstständig und ohne unzulässige fremde Hilfe erbracht habe. Ich habe keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie wörtliche und sinngemäß Zitate kenntlich gemacht. Die Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Dortmund, 14. Juni 2018

Ort, Datum

Unterschrift

Belehrung:

Wer vorsätzlich gegen eine die Täuschung über Prüfungsleistungen betreffende Regelung einer Hochschulprüfungsordnung verstößt, handelt ordnungswidrig. Die Ordnungswidrigkeit kann mit einer Geldbuße von bis zu 50.000,00 € geahndet werden. Zuständige Verwaltungsbehörde für die Verfolgung und Ahndung von Ordnungswidrigkeiten ist der Kanzler/die Kanzlerin der Technischen Universität Dortmund. Im Falle eines mehrfachen oder sonstigen schwerwiegenden Täuschungsversuches kann der Prüfling zudem exmatrikuliert werden. (§ 63 Abs. 5 Hochschulgesetz - HG -)

Die Abgabe einer falschen Versicherung an Eides statt wird mit Freiheitsstrafe bis zu 3 Jahren oder mit Geldstrafe bestraft.

Die Technische Universität Dortmund wird ggf. elektronische Vergleichswerkzeuge (wie z.B. die Software „turnitin“) zur Überprüfung von Ordnungswidrigkeiten in Prüfungsverfahren nutzen.

Die oben stehende Belehrung habe ich zur Kenntnis genommen:

Dortmund, 14. Juni 2018

Ort, Datum

Unterschrift