HOCHSCHULE OSNABRÜCK

UNIVERSITY OF APPLIED SCIENCES

Fakultät Ingenieurwissenschaften und Informatik

Studiengang Medieninformatik

Modul Audio- und Videotechnik

Praktikumsbericht

Meilenstein 4
Datenkompression

Wintersemester 2020/2021

Dozent:

Prof. Dr. Julius Schöning

Verfasserin:

Patrick Felschen, Julian Voß

Matrikelnummer:

932056, 934505

Datum der Abgabe:

03.12.2020

I. Inhaltsverzeichnis

II. Abbildungsverzeichnis

III. Tabellenverzeichnis

1 Aufgabe: Huffman-Kodierung

Dieser Abschnitt befasst sich mit der Umsetzung der Huffman-Kodierung. Zunächst wurde ein Codebaum (S. Anhang 1) aus der gegebenen Wahrscheinlichkeitsverteilung erstellt. Daraus ließen sich die einzelnen Codewörter ablesen (S. Tabelle 1). Aus den einzelnen Codewörtern lässt sich anschließend eine mittlere Codewortlänge L_c berechnen.

Symbol	Wahrscheinlichkeit	Codewort
a	35%	11
b	25%	10
c	20%	01
d	10%	001
e	5%	0001
f	3%	00000
g	2%	00001

Tabelle 1: Huffman-Kodierung

Mittlere Codewortlänge:

$$L_c = (0{,}35 + 0{,}25 + 0{,}2) \cdot 2 + 0{,}1 \cdot 3 + 0{,}05 \cdot 4 + (0{,}03 + 0{,}02) \cdot 5 = 2{,}35$$

Bei der Huffman-Kodierung handelt es sich um eine Entropiekodierung. Diese Kodierung ist verlustfrei, alle redundanten Informationen werden entfernt.

2 Aufgabe: Lauflängenkodierung (RLE)

Um eine Lauflängenkodierung in MATLAB zu erstellen, wird als erstes die BMP-Datei eingelesen. Dadurch wird, in diesem Fall, eine 512x512 Matrix erstellt, welche mittels zweier For-Schleifen iteriert wird. In den Schleifen werden nun die Zeichen gezählt, welche in einer Reihe hintereinander vorkommen. Diese Information wird abschließend in eine Text-Datei getrennt abgespeichert.

Es handelt sich bei diesem Verfahren ebenfalls um eine verlustfreie Entropiekodierung. Alle redundanten Informationen werden entfernt.

Um eine effektive Kompression zu erstellen, ist zu beachten, dass es nur dann effektiv ist, wenn genug lange Runs[1] in der originalen Datei vorhanden sind.

2.1 Auswirkung auf die Dateigröße

Beim Betrachten der „moon.bmp“, werden lange Runs oberhalb und unterhalb des Mondes direkt im schwarzen Hintergrund erkennbar. Wobei hingegen in der „space.bmp“ viele kleine Sterne sichtbar sind, welche die Runs in kleinere Stücke unterbricht. Dies wirkt sich stark auf die Dateigröße aus. Je länger die Runs, desto kleiner wird die Dateigröße.

[1] Wiederholdende Zeichen in einer Datei

3 Aufgabe: Quantisierung

Über MATLAB wird eine PNG-Datei eingelesen, die daraus erhaltene Matrix, bestehend aus Pixelkoordinaten und dazugehörigen RGB-Werten, wird jeweils mit 8, 32 und 128 Farbstufen je Farbkanal quantisiert. Entsprechend der Farbstufe wird vorher eine Matrix mit gleichmäßigen Werten zwischen 0 und 1 erstellt. Die eingelesenen RGB-Werte werden auf diese Werte gerundet. Somit verringert sich die Anzahl der einzelnen Farben im Bild.

Bei der Quantisierung handelt es sich um eine Quellenkodierung, da hier Informationen verloren gehen. Im Endergebnis ist zu sehen, dass im Bild Informationen (in diesem Fall Farben) fehlen.

3.1 Auswirkung auf die Dateigröße

Die Dateigröße verringert sich stärker, je niedriger die Anzahl an Farbstufen gewählt wurde. Es treten jedoch starke Quantisierungsfehler auf, da bei zu niedriger Quantisierung zu wenig Farben dargestellt werden s. Abbildung 3.

Abbildung 3: 8 Farbstufen

Abbildung 2: 32 Farbstufen

Abbildung 1: 128 Farbstufen

VI. Anhang

Anhang 1: Codebaum Aufgabe 1