Praktikum Multimedia-Technik

Blatt 1 - Huffman-Decodierung

Aufgabe 1 - `GetBits()`

Schreiben Sie Funktionen um Dateien auf Bitebene auslesen zu können. Dazu zählen insbesondere diese drei Funktionen:

- `int GetBits(int n);` Liefert die nächsten n Bits der Datei (wobei $n \leq 24$).
- `int PeekBits(int n);` Liefert wie `GetBits()` die nächsten n Bits, ohne aber die aktuelle Position in der Datei zu ändern. D.h. mehrfache Aufrufe von PeekBits() hintereinander liefern die gleichen Werte.
- `int SkipBits(int n);` Überspringt die nächsten n Bits in der Datei.

Tip: `GetBits()` läßt sich offensichtlich trivial durch eine Kombination aus `PeekBits()` und `SkipBits()` implementieren.

Aufgabe 2 - Huffman-Decoder

Schreiben Sie eine Funktion, die unter Benutzung der Funktionen aus Aufgabe 1 und gegebener Huffman-Tabelle das nächste Symbol aus einer Datei liest. Eine Huffman-Tabelle ist dabei eine Zuordnung von Bitcode zu einem Symbol (in unserem Fall einfach eine Integer-Zahl). Sieht die Huffmantabelle z.B. so aus:

code	symbol
1	42
010	2003
001	4711
011	-1
0001	64

dann sollte das Programm für das Eingabefile `00101010 00101101 10100001 ...` die Werte 4711, 2003, 42, 64, -1, -1, 2003, 64, ... liefern.

Aufgabe 3

Lesen Sie den H.261 Standard (nur der Inhalt bis Abschnitt 4 ist relevant).

Dirk Farin, farin@uni-mannheim.de

Praktikum Multimedia-Technik

Blatt 2 - Bitstream Decodierung

Aufgabe 1 - Intra-Decoding

Schreiben Sie einen Bitstream-Decoder für H.261 Intra-Frames. Beachten Sie, dass Sie damit nur das erste Bild der H.261 Streams decodieren können, da in den folgenden Bildern Inter-Macroblöcke enthalten sind.
Anfang der "Hall-and-Monitor" Sequenz:

```
0000000 00 01 00 97 00 80 40 00 00 22 e4 6c 8b 4a 5f 84
0000020 a4 f1 f6 31 2d 20 4c b0 ba c0 34 fe 26 c3 9c e8
0000040 9a 3a dc b2 c0 25 2c 2e ce 26 16 5b 80 e5 21 76
0000060 73 59 05 69 c2 8a a2 c8 4b 08 a4 08 60 3b e5 06
0000100 60 9e 7c fa ea ae 00 31 01 c0 60 9f 37 7a 26 8d
0000120 92 00 77 fa bf e1 76 9a 79 79 20 39 2e e2 26 71
0000140 ac 80 ab 13 8e e3 e7 2a 1a 8e 67 48 23 7e 24 2e
0000160 ca 1a 08 df 89 0b bc 43 49 3e fe 9d de 84 a3 7d
```

Ausgabe des Decoders für "Hall-and-Monitor" Sequenz:

```
---------- PICTURE HEADER ----------
temporal reference: 1
split screen:       no
document camera:    no
freeze pic release: yes
image format:       Q-CIF
still image mode:   yes
extra data: 0
extra data: 0
extra data: 0
GOB 1
GQUANT 14
MBA:   1 = last + 1
MTYPE: intra                            , tcoeff
intra DC:  1424
run,level: 0 (1), -3
run,level: 0 (2), -2
run,level: 0 (3), -2
run,level: 1 (5), -1
run,level: 0 (6), -1
run,level: 2 (9), -2
run,level: 0 (10), -2
run,level: 3 (14), -1
run,level: 5 (20), -1
EOB
intra DC:  1584
run,level: 1 (2), -3
run,level: 0 (3), -2
run,level: 5 (9), -2
run,level: 0 (10), -3
run,level: 9 (20), -1
```

```
EOB
intra DC:  1408
run,level: 0 (1), -4
run,level: 3 (5), -1
run,level: 0 (6), -1
run,level: 7 (14), -1
EOB
intra DC:  1560
EOB
intra DC:  920
EOB
intra DC:  1096
EOB
MBA:   2 = last + 1
MTYPE: intra                          , tcoeff
intra DC:  1712
run,level: 0 (1), -1
run,level: 0 (2), -3
run,level: 0 (3), -3
run,level: 5 (9), -3
run,level: 0 (10), -3
run,level: 9 (20), -1
EOB
intra DC:  1648
run,level: 0 (1), 5
run,level: 0 (2), -3
run,level: 0 (3), -3
run,level: 1 (5), -1
run,level: 3 (9), -3
run,level: 0 (10), -2
run,level: 9 (20), -1
EOB
intra DC:  1648
run,level: 1 (2), 1
EOB
intra DC:  1600
run,level: 0 (1), 3
run,level: 0 (2), 1
EOB
intra DC:  896
EOB
intra DC:  1104
EOB
MBA:   3 = last + 1
MTYPE: intra                          , tcoeff
intra DC:  800
run,level: 1 (2), -3
run,level: 2 (5), 2
EOB
intra DC:  1152
run,level: 0 (1), -6
run,level: 0 (2), -8
run,level: 0 (3), -1
...
```

Praktikum Multimedia-Technik

Blatt 3 - Intra-Block Decodierung

Aufgabe 1 - Dequantisierung

Bei der Syntax-Decodierung haben Sie die Koeffizienten der Blöcke als Run/Level-Paare erhalten. Füllen Sie nun die Koeffizienten anhand der Zigzag-Ordnung an die richtige Stelle der 8×8-Blöcke ein und dequantisieren Sie die Koeffizienten $\hat{c}$ mit dem aktuellen MQUANT Wert. Die Formel für die dequantisierten Koeffizienten c lautet dabei:

$$c = \begin{cases} MQUANT \cdot (2\hat{c} + sign(\hat{c})) & \text{für MQUANT ungerade} \\ MQUANT \cdot (2\hat{c} + sign(\hat{c})) - sign(\hat{c}) & \text{für MQUANT gerade} \end{cases}$$

Die dequantisierten Werte werden auf den Bereich $[-2048; 2047]$ begrenzt. Beachten Sie, dass die DC-Koeffizienten davon abweichend durch einfache Multiplikation mit 8 dequantisiert werden. Referenzausgaben zum Vergleich der Ergebnisse finden Sie auf der Web-Seite.

Aufgabe 2 - Inverse-DCT

Transformieren Sie die Koeffizienten-Blöcke durch Anwenden der IDCT in den Ortsraum. Die IDCT ist dabei definiert durch:

$$s(x,y) = \sum_{u=0}^{7} \sum_{v=0}^{7} \frac{C(u)}{2} \frac{C(v)}{2} \mathcal{S}(u,v) \cos((2x+1)u\pi/16) \cos((2y+1)v\pi/16)$$

mit

$$C(u) = \begin{cases} 1/\sqrt{2} & \text{für } u = 0 \\ 1 & \text{für } u > 0 \end{cases}.$$

Benutzen Sie aus Gründen der Effizienz unbedingt vorberechnete Tabellen für die Cosinusterme. Durch geschickte Klammerung lässt sich die Transformation in zwei Rechenschritte separieren:

$$s(x,y) = \sum_{u=0}^{7} \frac{C(u)}{2} \left(\sum_{v=0}^{7} \frac{C(v)}{2} \mathcal{S}(u,v) \cos((2y+1)v\pi/16) \right) \cos((2x+1)u\pi/16)$$

Das bedeutet, dass anstelle einer 2D-Transformation zuerst 1D-Transformationen auf den Zeilen der Matrix und danach auf den Spalten angewendet werden. Dadurch verringert sich die Anzahl der Berechnungen erheblich. Referenzausgaben zum Vergleich der Ergebnisse finden Sie auf der Web-Seite.

Dirk Farin, farin@uni-mannheim.de

Aufgabe 3 - YCbCr nach RGB Farbraumkonvertierung

Nach der IDCT erhalten Sie direkt die Bilddaten im YCbCr-Farbraum. Konvertieren Sie diesen nach RGB. Da die Chroma-Kanäle im 4:2:0-Format gespeichert wurden, benutzen Sie einfach jedes Chroma-Pixel für jeweils 4 Luminanzpixel. Die Transformation von YCbCr nach RGB ergibt sich als:

$$\begin{pmatrix} R \\ G \\ B \end{pmatrix} = \begin{pmatrix} 1.164 & 0 & 1.60 \\ 1.164 & -0.39 & -0.81 \\ 1.164 & 2.02 & 0 \end{pmatrix} \begin{pmatrix} Y - 16 \\ Cb - 128 \\ Cr - 128 \end{pmatrix}.$$

Aufgabe 4 - Ausgabe

Stellen Sie das decodierte Bild auf dem Bildschirm dar. Hinweis: Sie können Aufgabe 4 unabhängig von Aufgabe 3 lösen, indem Sie nur die Y-Komponente als Graustufenbild darstellen.

Bonusaufgabe - schnelle DCT

Sie werden bemerken, dass der Rechenzeit-intensivste Teil die IDCT sein wird (auch mit dem Row/Column-Ansatz). Ersetzen Sie die 1D-IDCTs durch einen schnellen Algorithmus. Ein solcher ist in Abb. 1 abgebildet. Im Falle der IDCT

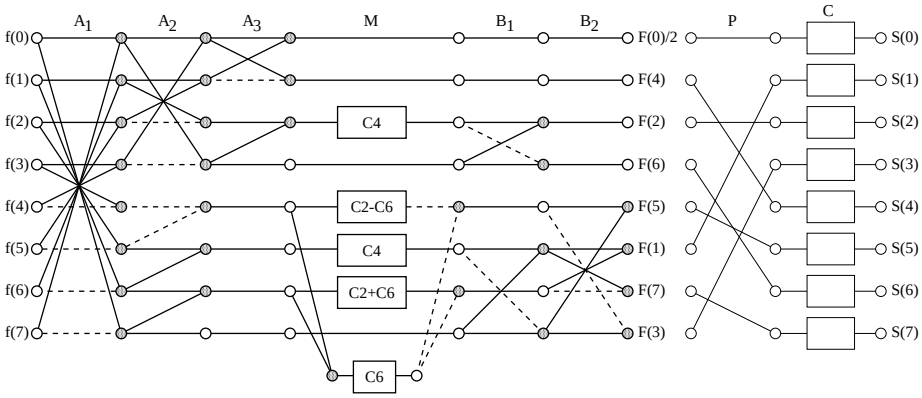

Abbildung 1: Flußgraph für 8-Punkt DCT nach Arai, Agui, Nakajima

wird dieser von rechts nach links durchlaufen. Führen zwei Linien zusammen, so werden die entsprechenden Werte addiert. Gestrichelte Linien negieren den Wert und die Kästchen entsprechen einer Multiplikation mit einer Konstanten. Sie sehen, dass für eine 1D-DCT nun nur noch 13 Multiplikationen nötig sind. Die erste Stufe (rechte Spalte) kann dabei auch noch im Row/Column Ansatz für beide Schritte zusammengefasst werden, so dass nur noch 5 Multiplikationen übrig bleiben. Die normale 1D-DCT benötigt dahingegen 64 Multiplikationen. Die Werte für die Kästchen in Stufe M sind dabei (von oben nach unten): $1/\sqrt{2}, \cos(\pi/8) - \cos(3\pi/8), 1/\sqrt{2}, \cos(\pi/8) + \cos(3\pi/8), \cos(3\pi/8)$ und die Werte der Spalte C ergeben sich zu $\sqrt{2}/4$ für den ersten Wert und $1/2 \cos(\pi \cdot n/16)$ für $n = 1, \ldots, 7$.

Praktikum Multimedia-Technik

Blatt 4 - Inter-Blöcke, Bewegungskompensation

Aufgabe 1 - Inter-Block-Decodierung

Erweitern Sie den Bit-Stream-Decoder so, dass er auch Inter-Blöcke decodieren kann. Dazu zählen die Syntaxelemente MVD (motion-vector displacement), CBP (coded-block pattern) und die restlichen Macroblocktypen. Beachten Sie auch, dass viele Restriktionen von Intra-Blöcken bei Inter-Codierung nicht mehr vorhanden sind. So kann z.B. MBA> 1 sein. Beachten Sie auch die unterschiedliche Behandlung des DC-Koeffizienten und dass der spezielle Huffman-Code "1s" in Tabelle 5 nun tatsächlich auftreten kann.
Einen Referenz-Trace des Decoders finden Sie auf der Web-Seite.

Aufgabe 2 - Motion-Compensation

Implementieren Sie die Bewegungskompensation. Dazu kopieren Sie die Bilddaten, ggf. verschoben gemäß dem angegebenen Motionvektor, an die aktuelle MB-Position. Codierte DCT-Differenzdaten werden zu diesen addiert. Vernachlässigen Sie zunächst das Loop-Filter, d.h. behandeln sie Macroblöcke mit FIL im MTYPE genauso wie Macroblöcke ohne FIL.

Aufgabe 3 - Loop-Filter

Implementieren Sie den Loop-Filter in der Bewegungskompensation als einfachen separierten 1-2-1 Tiefpassfilter. Pixel am Rand des Macroblocks werden nicht gefiltert.

Herzlichen Glückwunsch, damit sollte Ihr H.261-Decoder komplett sein!

Dirk Farin, farin@uni-mannheim.de