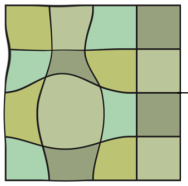

Image Processing & Understanding

Grundlagen der Bildverarbeitung

Praktische Aufgaben - Aufgabe 4

Wintersemester 15/ 16
Bildverarbeitung und Bildverstehen
Prof. Klaus Tönnies,
Tim König, Johannes Steffen

Abgabe: Abgabe erfolgt spätestens am 14.12.2015 um 07:59 Uhr. Jedes Team muss also insgesamt 5 Dateien (*p04_execute.m*, *p04_compress.m*, *p04_decompress.m*, *p04_rle.m*, *p04_rld.m*) einreichen. Allgemeine Details zur Einreichung finden Sie auf der Webseite!

In dieser Aufgabe sollen Sie ein Grauwertbild komprimieren. Dafür sollen Sie eine abgewandelte Form der JPEG-Kompression verwenden, welche durch mehrere Schritte erfolgt:

- Einteilung des Bildes in $m \times m$ Blöcke ($m = 8$ bzw. 16). Dann für jeden Block:
  - Frequenzraumüberführung mittels Diskreter Kosinustransformation (DCT)
  - Quantisierung der Koeffizienten mit der entsprechenden $m \times m$ Quantisierungsmatrix
  - Zig-Zag-Umsortierung der quantisierten Koeffizienten entlang der Diagonalen (Abbruch, wenn nur noch 0-Elemente folgen)
  - Run-Length-Encoding (RLE) des entstandenen quantisierten Koeffizientenvektors (Achtung: Eigentlich wird bei JPEG die Huffman-Kodierung benutzt!)
- Rückgabe aller RLE-Vektoren der Blöcke des Bildes

Um dies alles umsetzen zu können sollen Sie folgende Aufgaben erledigen:

1. Schreiben Sie zunächst eine Funktion *vecRLE = p04_rle(vec)*, welche Ihnen einen Zeilenvektor RLE-kodiert. Der resultierende Vektor *vecRLE* soll an den ungeraden

Indizes die *Values* und an den geraden Indizes die *Run-Lengths* abbilden.

Beispiel:

$vec = \begin{pmatrix}1 & 1 & 1 & 1 & 1 & 1 & 2 & 3 & 3 & 3\end{pmatrix}$, $p04\_rle(vec) = \begin{pmatrix}1 & 6 & 2 & 1 & 3 & 3\end{pmatrix}$

2. Schreiben Sie (passend zu Ihrer Kodierungsfunktion) eine Dekodierungsfunktion *vec = p04_rld(vecRLE)*, welche in der Lage ist aus Ihrem enkodierten Vektor den dekodierten, also den originalen Vektor, wiederherzustellen.

3. Schreiben Sie eine Funktion *compressedImage = p04_compress(image, blockSize)*, welche folgende Teilschritte umsetzt:

   a) Zerlegung des Bildes in $blockSize \times blockSize$ große Blöcke (also 8×8 bzw. 16×16)

   b) Diskrete Kosinustransformation jedes Blockes mittels der Matlab/Octave-Funktion *dct2*

   c) Quantisierung der DCT-Blöcke mit der passenden *blockSize x blockSize* Quantisierungsmatrix (siehe bereitgestellte *p04_quantm.mat*)

   d) Umsortierung der quantisierten Koeffizienten entlang der (Anti-)Diagonalen jedes Blockes mit der bereitgestellten Funktion *p04_zigzag*

   e) Kodierung des zuvor sortierten, quantisierten Koeffizientenvektors mittels Ihrer selbstgeschriebenen Funktion *p04_rle* aus Aufgabe 1

   f) Rückgabe der kodierten Vektoren in einem Cell-Array (*compressedImage*) der Größe *Vertikale Blöcke* $\times$ *Horizontale Blöcke*

4. Schreiben Sie (passend zu Ihrer Funktion *p04_compress*) eine Funktion *image = p04_decompress(compressedImage, blockSize)*, welche Ihre komprimierte Bildspeicherung dekomprimiert und Ihnen als Grauwertbild zurückgibt. Um die Komprimierungsschritte rückgängig zu machen, können Sie folgende Funktionen verwenden:

   a) *p04_rld* (Ihre Funktion aus Aufgabe 2)

   b) *p04_zagzig* (bereitgestellt)

   c) *idct2* (aus Matlab bzw. Octave)

5. Testen Sie Ihre Funktionen *p04_compress* und *p04_decompress* innerhalb eines Scripts *p04_execute* mit dem bereitgestellten Beispielbild *p04_Bild1.png* für die $blockSize = 8$ und $blockSize = 16$ und geben Sie die Kompressionsrate mittels der Funktion *disp* auf der Konsole aus. Geben Sie alle drei Bilder (Original, Block-8-komprimiert, Block-16-komprimiert) aus und vergleichen Sie deren Qualität in den Kommentaren.