

---

## Bildverarbeitung SoSe 2012

### Übung 7

Max Michels, Sebastian Kürten

---

## 1 Aufgabe 1

## 2 Aufgabe 2

Es wurde die 2D-Wavelet-Transformation mit Haar-Wavelets implementiert. Dabei wurde zur Implementierung das Lifting-Schema verwendet. Kern der Implementierung ist die Version für Hin- und Rücktransformation einer einzelnen Spalte, auf der die Gesamttransformation aufgebaut ist.

In der Hauptdatei werden dann für die beiden Eingabebilder im Wavelet-Raum kleine Werte im Sinne des Absolutbetrags auf Null gesetzt. Dies geschieht in der Funktion 'compress'. Es wurden anteilig die kleinsten vorkommenden absoluten Werte im Bild auf Null gesetzt, d.h. bspw. die Angabe 0.9 bedeutet, dass 90 Prozent aller Werte auf Null gesetzt wurden. Unten sind die Abbildungen der auf diese Art gewonnen Bilder zu sehen.

Listing 1: Hauptdatei a2.m

```
1 values = [0.5 0.6 0.7 0.8 0.9 0.95 0.97];
2
3 for pic = {"lena", "old-lady"}
4 %for pic = {"lena64"}
5     pic = pic{1};
6     src = sprintf("%s-bw.png", pic)
7
8     B = normalize(imread(src));
9     w = size(B, 2);
10    C = compress(B, values);
11
12    for i = [1:size(values, 2)]
13        value = values(i);
14        name = sprintf("%s.compressed.%.2f.png", pic, value)
15        p = ((i-1) * w);
16        Ci = uint8(denormalize(C(:, [p + 1:p + w])));
17        imwrite(Ci, name);
18    end;
19 end;
```

Listing 2: Kompression

```
1 function A = compress(B, values)
2     A = B;
3     H = haar(B);
4
5     for i = values
6         cutOff = percentile(H, i);
7         HM = setToZeroAbs(H, cutOff);
8         M = ihaar(HM);
9         A = horzcat(A, M);
10    end;
11
```

```
12 end;
```

Listing 3: Haar transformation with Lifting schema

```
1 function A = haar(B)
2     A = haarCols(haarRows(B));
3 end;
```

Listing 4: Haar for rows (based on column version)

```
1 function A = haarRows(B)
2     A = haarCols(B')';
3 end;
```

Listing 5: Haar for columns

```
1 function A = haarCols(B)
2     [h, w] = size(B);
3     A = zeros(h, w);
4     for i = [1 : w]
5         A(:, i) = haarSingleCol(B(:, i));
6     end;
7 end;
```

Listing 6: Haar transform for a single column

```
1 function A = haarSingleCol(B)
2 % apply the haar transform to a single column
3     h = size(B, 1);
4     A = zeros(h, 1); % overall result
5     gpu = zeros(h / 2, 1); % result of add
6     gmu = zeros(h / 2, 1); % result of sub
7     for i = [1 : h/2] % calculate add
8         gpu(i) = (B((i-1)*2 + 1) + B((i-1)*2 + 2)) / sqrt(2);
9     end;
10    for i = [1 : h/2] % calculate sub
11        gmu(i) = (B((i-1)*2 + 1) - B((i-1)*2 + 2)) / sqrt(2);
12    end;
13    A(h/2+1:end) = gmu;
14    if h == 2
15        % stop for length 2
16        A(1) = gpu;
17    else
18        % recurse into one half
19        A(1:h/2) = haarSingleCol(gpu);
20    end;
21 end;
```

Listing 7: Inverse haar transformation

```
1 function A = ihaar(B)
2     A = ihaarCols(ihaarRows(B));
3 end;
```

Listing 8: Inverse haar transformation (based on columns)

```
1 function A = ihaarRows(B)
```

```

2     A = ihaarCols(B')';
3 end;

```

Listing 9: Inverse haar transformation for columns

```

1 function A = ihaarCols(B)
2     [h, w] = size(B);
3     A = zeros(h, w);
4     for i = [1 : w]
5         A(:,i) = ihaarSingleCol(B(:,i));
6     end;
7 end;

```

Listing 10: Inverse Haar for a single Column

```

1 function A = ihaarSingleCol(B, depth = 0)
2 % revert the haar transform of a single column
3     h = size(B, 1); % the size of the current vector
4     if h == 2 % stop recursion if h = 2
5         % the first two values are calculated from
6         % the last two entries of the vector
7         A = [B(1) + B(2); B(1) - B(2)] / sqrt(2);
8     else
9         % we deduct one part by adding and the
10        % other by subtracting parts of the vector
11        S = ihaarSingleCol(B(1:h/2), depth + 1);
12        T = B(h/2+1:end);
13        A = [S + T; S - T] / sqrt(2);
14        % fancy permutation to reorder rows
15        sel = [rem([0:h-2]*h/2, h-1) h-1] + 1;
16        A = A(sel);
17    end;
18 end;

```

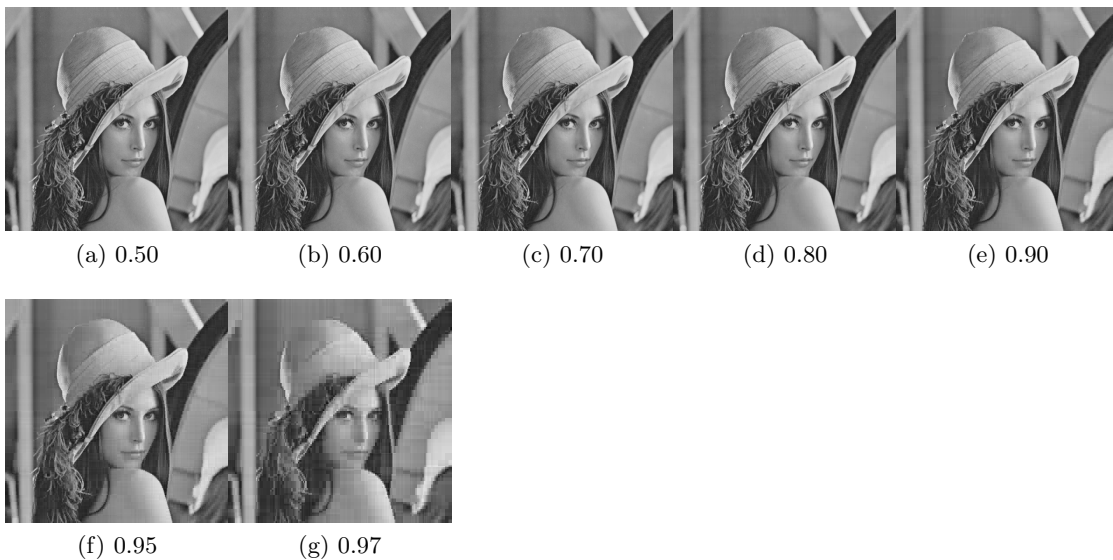


Figure 1: lena

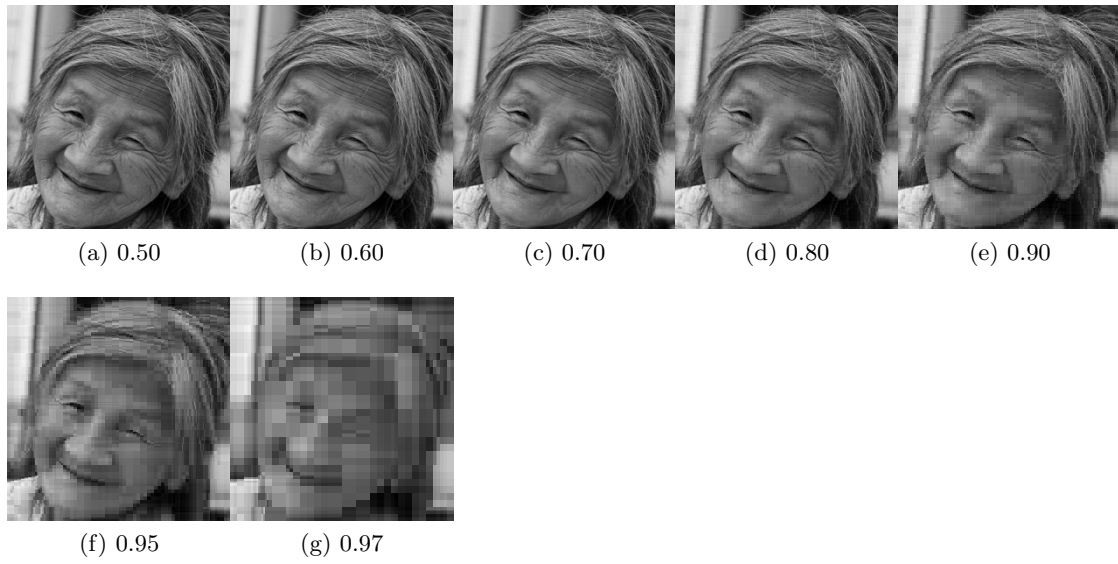


Figure 2: old-lady

### 3 Aufgabe 3