

## Aufgabe 1

Die CDF(2x2)-Wavelet-Transformation wurde für diese Aufgabe erfolgreich in octave implementiert. Nach der Transformationsvorschrift wird auf nicht zulässige Feldelemente zugegriffen. Diese dürfen nicht einfach auf Null gesetzt werden, da dann die Rücktransformation nicht mehr richtig funktioniert. Setzt man das Pixel bei  $s_j(2n + 2)$  auf 0 so, ist das letzte Pixel bei der Rücktransformation falsch und zwar in jeder Stufe  $j$ . Dies erzeugt ein Raster von falschen Pixeln. Wenn man jedoch die vorletzte Spalte an diese Position ( $s_j(2b + 2) = 2(b)$ ) setzt, wobei  $b$  die Bildbreite ist, kann bei der Rücktransformation auf das letzte Pixel richtig geschlossen werden, wenn man die letzte Spalte der geraden berechneten Pixel an die entsprechende Position kopiert. Siehe dazu die implementation in den Dateien CDF2x22D.m und iCDF2x22D.m. Die Bilder in den Abbildungen 3 bis 8 stellen die einzelnen Stufen  $j$  dar. Es ist anzumerken, dass sie normalisiert sind. Die nicht normalisierten Ausgaben können im Ordner './octave/Ausgabe' gefunden werden.



Abbildung 1: CDF(2x2)-Schritt 1

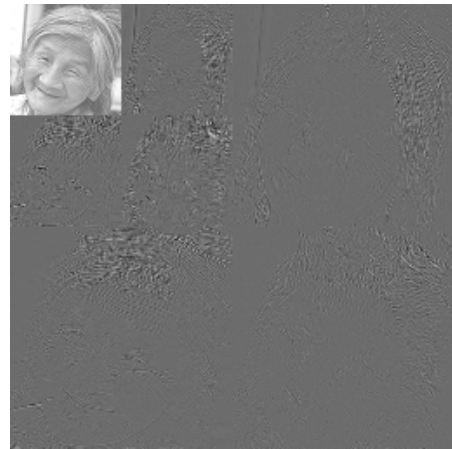


Abbildung 2: CDF(2x2)-Schritt 2

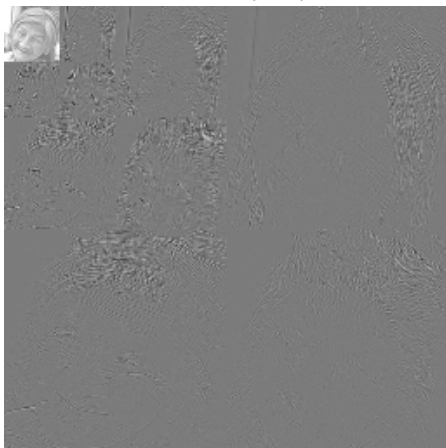


Abbildung 3: CDF(2x2)-Schritt 3

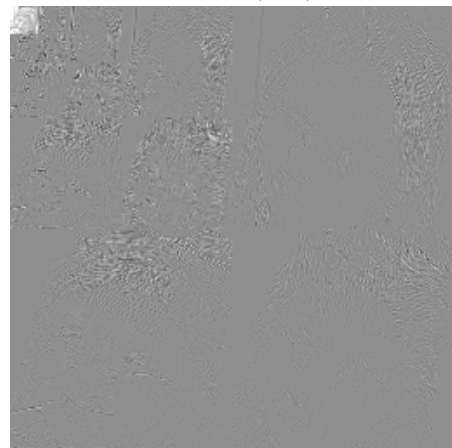


Abbildung 4: CDF(2x2)-Schritt 4

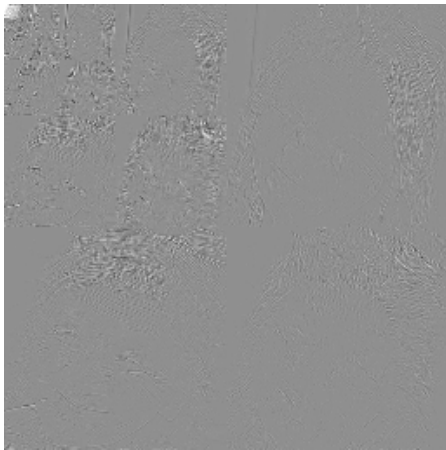


Abbildung 5: CDF(2x2)-Schritt 5

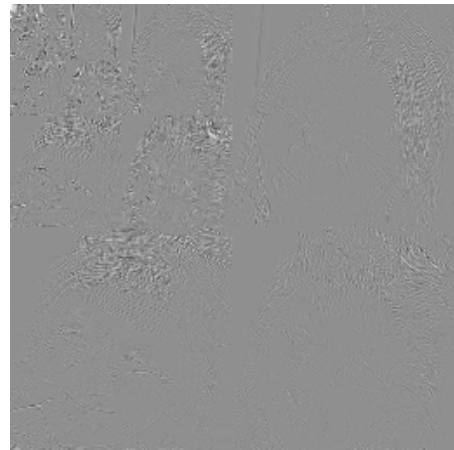


Abbildung 6: CDF(2x2)-Schritt 6

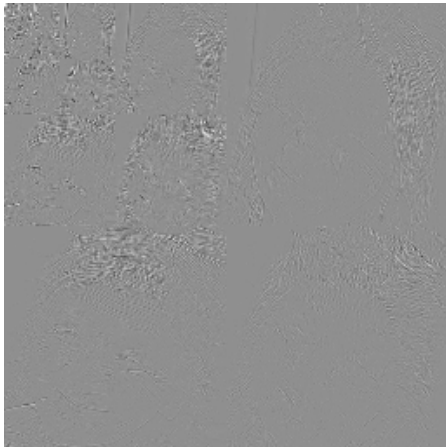


Abbildung 7: CDF(2x2)-Schritt 7

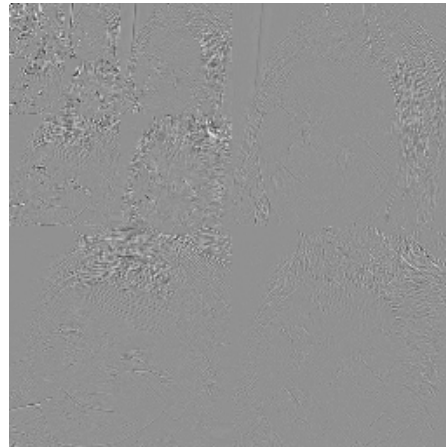


Abbildung 8: CDF(2x2)-Schritt 8

## Aufgabe 2

In dieser Aufgabe ist das Bild selektiv zu Glätten bzw. die Kanten zu zeigen, und zwar jeweils in den vier Quadranten des Bildes. Die Idee ist nun, dass man beim Transformieren die entsprechenden Bereiche löscht. Am einfachsten erklärt sich das an folgendem wavelettransformierten (nur eine Stufe) Bild (9:



Abbildung 9: Wavelet-Transformierte

Wenn man nun im Originalbild z.B. den Quadranten links-oben glätten will, so muss man in den Differenzkomponenten in Bild 9, die sich auf der rechten Seite und links unten befinden, jeweils die linken oberen Bereiche löschen. Dieses Prinzip kann man dann bei beliebig viele Stufen anwenden.

Wenn man dann zurücktransformiert, bekommt man dann das gewünschte Ergebnis.

In Bild 10 sieht man das transformierte Bild nach einer Stufe mit den weggeschnittenen Bereichen. In Bild 11 sieht man Bild 10 wieder zurücktransformiert. In Bild 12 sind nur jeweils nur noch ein Bereich weggeschnitten. Das letzte Bild ist dann wieder die rücktransformierte von 12.

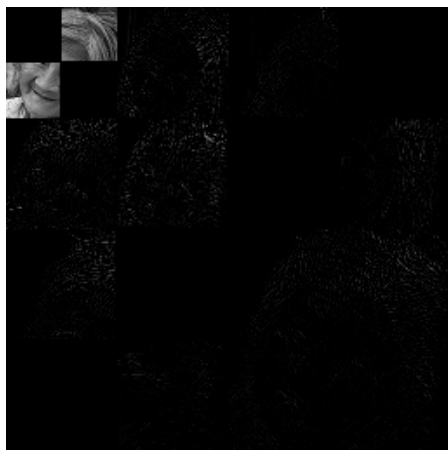


Abbildung 10: Transformierte mit  
weggeschnittenen  
Bereichen, eine Stufe

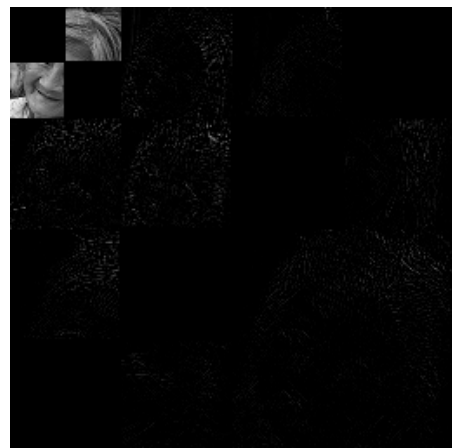


Abbildung 11: Rücktransformierte  
von 11

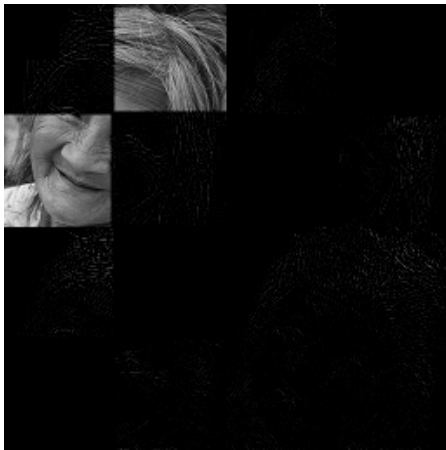


Abbildung 12: Transformierte mit  
nur einem Bereich  
gelöscht



Abbildung 13: Rücktransformierte  
von 12

### Aufgabe 3

Wir haben den Lucy-Richardson-Algorithmus implementiert, aber die Laufzeit ist so groß, dass Ergebnisse, vor allem mit mehreren Iterationen nicht durchgeführt werden konnten. Der Quellcode der Versuche ist in den Dateien `LucyRichardson*` zu finden. Die `LucyRichardson2Dslow` arbeitet auf dem ganzen 2D-Array, wobei die anderen Zeilen und Spalten getrennt verarbeiten.