Grundidee: Die umliegenden Pixel beeinflussen die Farbe eines Pixels

Für die Berechnung der Intensität eines neuen Pixels werden die Intensitäten der umliegenden Pixel mit einem festgelegten Faktor multipliziert und dann alle addiert. Dies wird als Faltung bezeichnet.

Hinweis: auf den Rand kann man den Filter nicht so einfach anwenden, wir lassen die Randpixel daher so wie sie sind.

Beispiel:

	0	1	2	3	4	5	6
0	1	3	2	1	2	3	1
1	2	3	2	1	2	3	2
2	2	3	3	1	2	3	3
3	3	4	4	2	5	4	3
4	6	2	3	8	5	2	1
5	2	1	2	2	1	1	2
6	2	1	2	3	4	2	1

Intensität der Originalpixel: I_{alt}

-1	-2	-1
0	0	0
1	2	1

Filtermatrix

	0	1	2	3	4	5	6
0	1	3	2	1	2	3	1
1	2	2					2
2	2						3
3	3						3
4	6					-11	1
5	2						2
6	2	1	2	3	4	2	1

Intensität der neuen Pixel: I_{neu}

Pixel an Position (1|1):

$$I_{neu}(1,1) = \begin{matrix}(-1)\cdot I_{alt}(0,0) & +(-2)\cdot I_{alt}(1,0) & +(-1)\cdot I_{alt}(2,0)\\ +(0)\cdot I_{alt}(0,1) & +(0)\cdot I_{alt}(1,1) & +(0)\cdot I_{alt}(2,1)\\ +(1)\cdot I_{alt}(0,2) & +(2)\cdot I_{alt}(1,2) & +(1)\cdot I_{alt}(2,2)\end{matrix} = -1-6-2+2+6+3 = 2$$

Pixel an Position (5|4):

$$I_{neu}(5,4) = \begin{matrix}(-1)\cdot I_{alt}(4,3) & +(-2)\cdot I_{alt}(5,3) & +(-1)\cdot I_{alt}(6,3)\\ +(0)\cdot I_{alt}(4,4) & +(0)\cdot I_{alt}(5,4) & +(0)\cdot I_{alt}(6,4)\\ +(1)\cdot I_{alt}(4,5) & +(2)\cdot I_{alt}(5,5) & +(1)\cdot I_{alt}(6,5)\end{matrix} = -5-8-3+1+2+2 = -11$$

Auftrag:

1. *Berechne die restlichen Werte des oben angegebenen Bildes.*
2. *Öffne die Datei 01_faltung.ods. Das Beispiel aus Aufgabe 1 ist ein Teil des Bsp. 1 (zufällige Verteilung).*
3. *Kontrolliere Deine Berechnung, indem du die Filtermatrix einträgst.*
4. *Untersuche, wie sich diese Filtermatrix auf die anderen Beispiele auswirkt. Beschreibe, wozu diese Filtermatrix verwendet werden könnte. Untersuche, welchen Effekt es hat, wenn du jeden Wert der Matrix verdoppelst*[1]*.*
5. *Untersuche genauso auch folgende Filtermatrizen:*

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

0	-1	0
-1	4	-1
0	-1	0

0	-1	0
-1	5	-1
0	-1	0

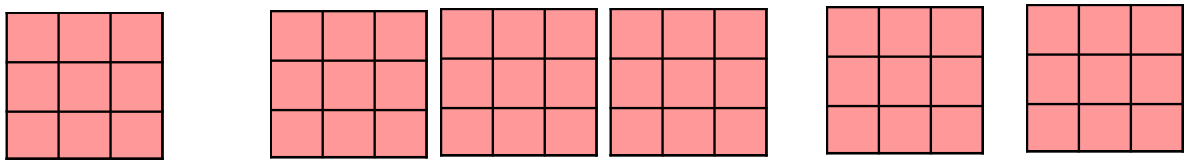
-2	-1	0
-1	1	1
0	1	2

1 Man kann die Filtermatrizen auch in GIMP direkt an Bildern ausprobieren. Unter Filter->Generic->Convolution Matrix bzw. Filter->Allgemein->Faltungsmatrix kann man die Matrizen ausprobieren.

6. *Die Summe der Einträge der Matrix ist in der Regel 0 oder 1. Beschreibe, welchen Effekt es auf das einfarbige Bild hat, wenn die Summe 0 ist. Beschreibe, was passiert, wenn die Summe 1 ist. Untersuche, was passiert, wenn die Summe zwischen 0 und 1 liegt bzw. größer als 1 ist.*

Faltungsfilter

Glätten/Weichzeichnen | Kantenfinden | Schärfen | Relief



Bildbearbeitungsalgorithmen 4: Mehrpixeloperationen (Faltung)

Ziel: Es soll eine Methode `public Picture faltung(Picture originalbild, double[][] filter)` implementiert werden, die den Filter auf ein Bild anwendet. Durch Aufruf mit geeigneten Filtern können dann die verschiedenen Bildbearbeitungsalgorithmen realisiert werden. Die Größe des Filters muss dabei nicht unbedingt 3x3 sein.

Programmierauftrag:

7. *Du erhältst ein Puzzle mit dem Quelltext der Methode. Ordne die Bauteile richtig an.*

8. *Kommentiere die einzelnen Abschnitte hinter den "//" - Kommentarzeichen. Folgende Fragen können dabei helfen. Du solltest auf jeden Fall Antworten auf die Fragen haben.*
   - *Welches Pixel hat die Koordinaten (x|y)? Welches Pixel hat die Koordinaten (xx|yy)?*
   - *Was speichern die Variablen laenge und halb?*
     Hinweis: Teilt man zwei ganze Zahlen durcheinander, werden die Nachkommastellen abgeschnitten: 3/2 = 1.
   - *Was gibt die Position (i | j) an?*
   - *Warum wird rot auf 0.0 gesetzt, wenn es kleiner als 0.0 ist? Warum auf 255.0, wenn rot größer als 255.0 ist?*
   - *Warum müssen die Variablen rot, gruen und blau als double definiert werden?*
   - *Warum beginnt die Schleife mit der Zählvariable x nicht bei 0, sondern bei halb?*
   - *Warum wird pixelNeu mit* original.getPixelArray() initialisiert?

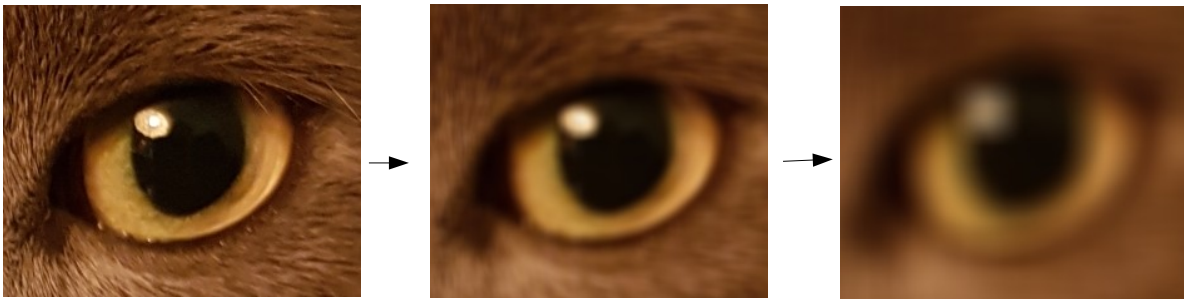
9. *Implementiere die Methode in Java. Erzeuge dazu eine neue Klasse "Mehrpixel-operationen". Importiere ganz am Anfang der Datei die notwendigen Hilfsklassen:*
   ```
   import imp.*;
   import java.awt.Color;
   ```
   Füge dann die Methode faltung ein. Hinweis: Du kannst die Begrenzung eines Wertes auf den Bereich 0-255 in einer extra Methode auslagern, da dies mehrfach benötigt wird.

10. *Implementiere eine Methode* `public Picture weichzeichnen(Picture originalbild)`. *Definiere dazu die Filtermatrix (vgl. direkte Initialisierung von Arrays) als zweidimensionales Array und rufe damit die Methode faltung auf.*

11. *Implementiere analog die Methoden* `schaerfen`, `relief`, `kantenfindenHorizontal`, `kantenfindenVertikal` *und* `kantenfinden`.

Bildbearbeitungsalgorithmen 5: Größere Filtermatrizen

Die bisherigen Filtermatrizen waren immer 3x3 Pixel groß. Bei Bildern mit hoher Auflösung sieht man den Effekt kaum, wenn nur die direkt benachbarten Pixel herangezogen werden. Daher kann man die Filtermatrizen auch größer wählen.



11x11 Mittelwertfilter *31x31 Mittelwertfilter*

Weichzeichnen mit Mittelwertfilter:

Um das Weichzeichnen zu realisieren, haben wir eine Matrix mit identischen Werten in jeder Zelle verwendet. Die Summe ergab 1. Damit wird automatisch der Mittelwert der neun Pixel berechnet.

Diese Idee kann einfach auf größere Filter übertragen werden: Man bestimmt die Anzahl der betroffenen Pixel und trägt den Kehrwert in jede Zelle der Matrix ein.

Programmierauftrag:

1. *Implementiere eine Methode public Picture weichzeichnen(Picture originalbild, int groesse), die einen Mittelwertfilter der Größe groesse x groesse einsetzt.*

Tipps:

- *Deklariere die Matrix in der richtigen Größe.*
- *Berechne die Anzahl der Pixel der Matrix.*
- *Setze jeden Wert der Matrix auf 1.0/(Anzahl der Pixel), indem du zwei Schleifen verwendest.*
  ***Achtung:** 1.0/anzahl ist etwas anderes als 1/anzahl, da bei der ersten Rechnung eine Kommazahl berechnet wird, bei der zweiten eine ganzzahlige Division durchgeführt und der Rest abgeschnitten wird.*

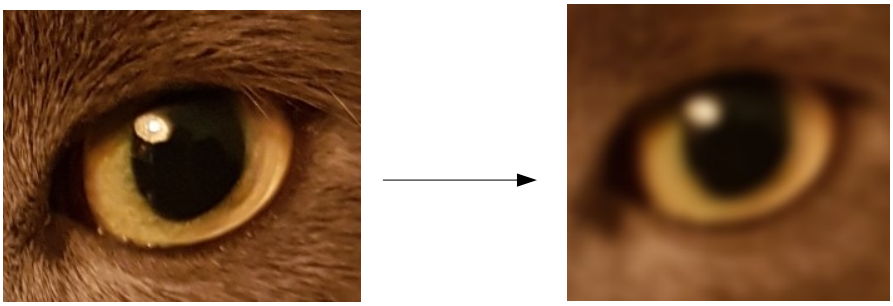
Bei größeren Filtern bleibt am Bildrand ein sichtbarer Streifen von Pixeln, die nicht verändert werden, da der Filter über die Bildgrenze hinausragen würde. Möchte man dieses Problem beheben, gibt es drei Möglichkeiten:

- Man setzt die Pixel außerhalb des Bildes alle auf "Schwarz".
- Man nimmt für die Pixel außerhalb des Bildes die Farbe des Pixels am Rand.
- Man spiegelt das Bild nach außen: z.B. bekommt das Pixel mit Koordinaten (-4|10) die Farbe des Pixels mit den Koordinaten (4|10).

Programmierauftrag:

2. *Passe die Implementierung der Methode faltung so an, dass eine drei Varianten umgesetzt wird. Überprüfe, ob das Bild nun wie gewünscht aussieht.*

Bildbearbeitungsalgorithmen 6: Weichzeichnen mit Gaußfiltern

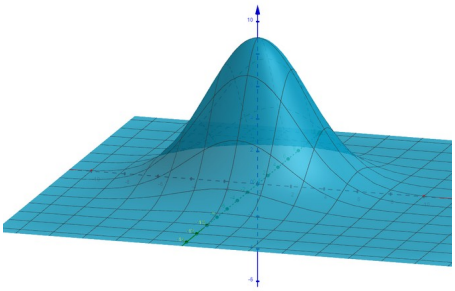


31x31 Gaußfilter

Bei größeren Filtern sollte der Einfluss weit entfernter Pixel geringer sein als der nahe liegender Pixel. Bei Mittelwertfiltern haben aber alle Pixel den gleichen Einfluss. Der Gaußfilter behebt dieses Problem.

Nahe an der Mitte liegende Zellen des Filters erhalten größere Werte als weiter außen liegende[2].

$$Gewicht(i,j) = K \cdot e^{-\frac{r^2}{2\sigma^2}}$$

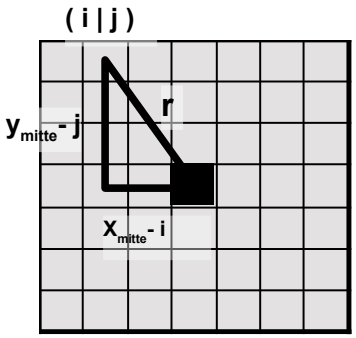


Gewichte des Gaußfilters

Dabei ist r der Abstand vom mittleren Punkt des Gaußfilters und berechnet sich mit dem Satz des Pythagoras. Sigma (σ) ist die Standardabweichung der Gaußverteilung und ergibt sich aus der Größe des Filters. Wählt man

$$\sigma = \frac{groesse}{6}$$

stellt man sicher, dass die Werte außerhalb des Filters nahezu keinen Einfluss auf die Berechnung hätten. K wird so gewählt, dass die Summe aller Gewichte 1 ergibt.



Programmierauftrag

1. *Implementiere eine Methode public Picture gaussfilter(Picture originalbild, int groesse), die einen Gaußfilter der Größe groesse x groesse generiert und für die Faltung verwendet.*

Tipps:

- Deklariere die Matrix in der richtigen Größe.
- Berechne Sigma.
- Berechne die Koordinaten der Mitte der Matrix.
- Berechne für jede Zelle der Matrix den Abstand r von Mitte mit dem Satz des Pythagoras.

  Setze den Wert der Zelle auf $e^{-\frac{r^2}{2\sigma^2}}$.

  (In Java e^x: Math.exp(x), $\sqrt{x}$: Math.sqrt(x), x^y: Math.pow(x,y))
- Berechne K, indem du alle Werte der Matrix addierst und den Kehrwert bildest.
- Multipliziere jede Zelle der Matrix mit K.

2 Auf der Seite *http://dev.theomader.com/gaussian-kernel-calculator/* (Juni 2020) können die Gewichte eines Gaußfilters berechnet werden.