Oblig 7 i 6121 hausten 2018

Student:

- Khoi Nguyen Hoang
- Kenneth Lindalen
- Tom Che Tran
- Per Øyvind Perry Stendal

Oppgave 1

```
1  import ij.*;
2  import ij.process.*;
3  import ij.gui.*;
4  import java.awt.*;
5  import ij.plugin.filter.*;
6  import java.util.*;
7
8  public class Median_Filter_Plugin implements
   PlugInFilter {
9
10     ImagePlus imp;
11     public int setup(String arg, ImagePlus imp) {
12         this.imp = imp;
13         return DOES_8G;     }
14
15     public void run(ImageProcessor ip) {
16         int heigth = ip.getHeight();
17         int width = ip.getWidth();
18         int filterStrH = 5; //Horisontal størrelse
19         int filterStrV = 5; // Vertikal størrelse
20         ArrayList <Integer> arrayVertikal = new
   ArrayList<Integer>();
21
22
23         int totalPxVekter = 0;
24         int[][] fasteVekterArray =
25         {
26           {1,1,1,1,1},
```

```
27                  {1,1,2,1,1},
28                  {1,2,3,2,1},
29                  {1,1,2,1,1},
30                  {1,1,1,1,1}
31          };
32
33          ImageProcessor nyttBilde = new
    ByteProcessor(width, heigth);
34
35
36          int vertStart = (int) (fasteVekterArray.length
    / 2.0);
37          int horStart = (int)
    (fasteVekterArray[0].length / 2.0);
38
39          //Går gjennom bilde horisontalt og tar alle nye
    midlertidige verdier og lagrer de midlertidig.
40          //Dvs. filter verdiene
41          for (int y = vertStart; y < heigth -
    filterStrV; y++) {
42              for (int x = horStart; x < width -
    filterStrH; x++) {
43                // Vekt matrisen
44                for(int m = -vertStart; m < vertStart;
    m++) {
45                  for(int n = -horStart; n < horStart;
    n++) {
46
47                    //int i = ip.getPixel(x+m,y+n);
48
49                    int filterVerdi =
    fasteVekterArray[vertStart+m][horStart+n];
50                    int horPos = x - horStart;
51                    int pikselVerdi = ip.getPixel(x+m,
    y+n);
52
53                    for (int i = 0; i < filterVerdi; i++)
    {
54
```

```
55                      arrayVertikal.add(pikselVerdi);
56
57                    }
58                  }
59                }
60                int median = 2;
61                Collections.sort(arrayVertikal);
62                int nyVerdi =
    arrayVertikal.get(((arrayVertikal.size()-1)/median));
63                nyttBilde.putPixel(x, y, nyVerdi);
64                arrayVertikal.clear();
65              }
66          }
67          ImagePlus im = new ImagePlus("Median
    filter",nyttBilde);
68            im.show();
69      }
70  }
```

Original

ImageJ

Fast

Min

Fast min

Max

Fast Max

Det har blitt brukt piksel størrelse 5 på alle bilder og på Fast brukte jeg 128 på offseten.

Analyse av Median bilder :

På avstand så ser ImageJ mye klare ut enn Fast metoden, men når du går nærmere inn på bildene er Fast den som fortsatt holder form best. Den største forskjellen på bildene er at imagej bruker sirkulert og Fast bruker kvadrater, som er ganske synlig på kantene i de forskjellige bildene.

Min bilder:

Med Min er det veldig lett å se forskjellene, mye mer rundheter og harde kanter. Bildene i seg selv er mye mørkere enn de forrige, som om

de mørke pikslene har blitt framhevet, utenom dette kunne jeg ikke se noen forskjeller.

Max bilder:

Dette blir mye av det samme som på Min bildene, rundheter og harde kanter(sirkler og firkanter), men i stedefor mørke bilder er de lyse pikslene framhevet.

Oppgave 3

```
1  import ij.*;
2  import ij.process.*;
3  import ij.gui.*;
4  import java.awt.*;
5  import ij.plugin.filter.*;
6  import java.util.*;
7
8  public class PseudoMedian_Plugin implements PlugInFilter {
9      ImagePlus imp;
10
11     public int setup(String arg, ImagePlus imp) {
12         this.imp = imp;
13         return DOES_8G;
14     }
15
16     public void run(ImageProcessor ip) {
17         int heigth = ip.getHeight();
18         int width = ip.getWidth();
19         int filterStrH = 5; //Horisontal størrelse
20         int filterStrV = 5; // Vertikal størrelse
21         ArrayList <Integer> arrayVertikal = new ArrayList<Integer>();
22         int median = 2;
23
24
25         int totalPxVekter = 0;
26         int[][] fasteVekterArray =
27         {
28             {1,1,1,1,1},
29             {1,1,2,1,1},
30             {1,2,3,2,1},
31             {1,1,2,1,1},
32             {1,1,1,1,1}
```

```
33          };
34
35          ImageProcessor nyttBilde = new ByteProcessor(width, heigth);
36          ImageProcessor ip2 = new ByteProcessor(width, heigth);
37
38
39          int vertStart = (int) (fasteVekterArray.length / 2.0);
40          int horStart = (int) (fasteVekterArray[0].length / 2.0);
41
42          //Går gjennom bilde horisontalt og tar alle nye midlertidige
    verdier og lagrer de midlertidig.
43          //Dvs. filter verdiene
44          for (int y = vertStart; y < heigth - vertStart; y++) {
45                  for (int x = horStart; x < width - horStart; x++) {
46
47                      //  K*1 median (horisontalt),
48                      for(int m = -vertStart; m < vertStart; m++) {
49
50                              int filterVerdi =
    fasteVekterArray[vertStart+m][horStart];
51                              int pikselVerdi = ip.getPixel(x+m, y);
52                              for (int i = 0; i < filterVerdi; i++) {
53                                      arrayVertikal.add(pikselVerdi);
54                              }
55                          }
56
57                          Collections.sort(arrayVertikal);
58                          int nyVerdi =
    arrayVertikal.get(((arrayVertikal.size()-1)/median));
59                          ip2.putPixel(x, y, nyVerdi);
60                          arrayVertikal.clear();
61                      //  1*L median (vertikalt)
62                      for(int n = -horStart; n < horStart; n++) {
63                              int filterVerdi =
    fasteVekterArray[vertStart][horStart+n];
64                              int pikselVerdi = ip2.getPixel(x, y+n);
65                              for (int i = 0; i < filterVerdi; i++) {
66                                      arrayVertikal.add(pikselVerdi);
67                              }
68                      }
69
70                      Collections.sort(arrayVertikal);
71                      nyVerdi = arrayVertikal.get(((arrayVertikal.size()-
    1)/median));
72                      nyttBilde.putPixel(x, y, nyVerdi);
73                      arrayVertikal.clear();
74                  }
75          }
76
77          ImagePlus im = new ImagePlus("Median filter",nyttBilde);
```

```
78            im.show();
79        }
80
81    }
```