**Oblig 8 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.util.*\*; | | 5 | import *java.awt.*\*; | | 6 | import *ij.plugin.filter.*\*; | | 7 | import *ij.process.*\*; | | 8 | import *java.lang.Math.*\*; | | 9 |  | | 10 | public *class* SobelOperator\_Plugin implements *PlugInFilter* { | | 11 |  | | 12 | *ImagePlus* imp; | | 13 | *int* pixelX = 0; | | 14 | *int* pixelY = 0; | | 15 |  | | 16 | public *int* setup(*String* *arg*, *ImagePlus* *imp*) { | | 17 | this.imp = imp; | | 18 | return DOES\_8G; | | 19 | } | | 20 |  | | 21 | public *void* run(*ImageProcessor* *ip*) { | | 22 |  | | 23 | *int* sobelX[][] = { | | 24 | {-1, 0, 1}, | | 25 | {-2, 0, 2}, | | 26 | {-1, 0, 1} | | 27 | }; | | 28 |  | | 29 | *int* sobelY[][] = { | | 30 | {-1, -2, -1}, | | 31 | {0, 0, 0}, | | 32 | {1, 2, 1} | | 33 | }; | | 34 |  | | 35 | *int* width = ip.getWidth(); | | 36 | *int* height = ip.getHeight(); | | 37 | *int* resultat = 0; | | 38 | *int* resultat2 = 0; | | 39 |  | | 40 |  | | 41 | *ImageProcessor* ip2 = new *ByteProcessor*(width, height); | | 42 | *ImageProcessor* ip3 = new *ByteProcessor*(width, height); | | 43 |  | | 44 | for (*int* x = 1; x < width - 2; x++) { | | 45 | for (*int* y = 1; y < height - 2; y++) { | | 46 | resultat = 0; | | 47 | pixelX = 0; | | 48 | pixelY = 0; | | 49 | resultat2 =0; | | 50 | for (*int* j = -1; j <= 1; j++) { | | 51 | for (*int* i = -1; i <= 1; i++) { | | 52 | *int* hentPx = ip.getPixel(x+i, y+j); | | 53 | // kjøres gjennom konvelusjon | | 54 | pixelX += sobelX[1+j][1+i] \* hentPx; | | 55 | pixelY += sobelY[1+j][1+i] \* hentPx; | | 56 | } | | 57 | } | | 58 | resultat = (*int*) Math.sqrt((pixelX \* pixelX) + (pixelY \* pixelY)); | | 59 | resultat2 = (*int*)(Math.atan2(pixelY,pixelX)\*180/Math.PI); | | 60 | ip2.putPixel(x, y, resultat); | | 61 | ip3.putPixel(x,y,resultat2); | | 62 | } | | 63 | } | | 64 | *ImagePlus* im = new *ImagePlus*("Sobel Strength",ip2); | | 65 | im.show(); | | 66 | *ImagePlus* im2 = new *ImagePlus*("Sobel",ip3); | | 67 | im2.show(); | | 68 | } | | 69 | } | |  |

**Oppgave 2**

|  |  |
| --- | --- |
|  | |
| |  |  | | --- | --- | | 1 | import *ij.*\*; | | 2 | import *ij.process.*\*; | | 3 | import *ij.gui.*\*; | | 4 | import *java.util.*\*; | | 5 | import *java.awt.*\*; | | 6 | import *ij.plugin.filter.*\*; | | 7 | import *ij.process.*\*; | | 8 | import *java.lang.Math.*\*; | | 9 | import *java.awt.Color.*\*; | | 10 |  | | 11 | public *class* SobelOperatorColor\_Plugin implements *PlugInFilter* { | | 12 |  | | 13 | *ImagePlus* imp; | | 14 | *int* pixelX = 0; | | 15 | *int* pixelY = 0; | | 16 | *Color* color; | | 17 |  | | 18 | public *int* setup(*String* *arg*, *ImagePlus* *imp*) { | | 19 | this.imp = imp; | | 20 | return DOES\_8G; | | 21 | } | | 22 |  | | 23 | public *void* run(*ImageProcessor* *ip*) { | | 24 |  | | 25 | *int* sobelX[][] = { | | 26 | {-1, 0, 1}, | | 27 | {-2, 0, 2}, | | 28 | {-1, 0, 1} | | 29 | }; | | 30 |  | | 31 | *int* sobelY[][] = { | | 32 | {-1, -2, -1}, | | 33 | {0, 0, 0}, | | 34 | {1, 2, 1} | | 35 | }; | | 36 |  | | 37 | *int* width = ip.getWidth(); | | 38 | *int* height = ip.getHeight(); | | 39 | *int* resultat = 0; | | 40 | *int* resultat2 = 0; | | 41 |  | | 42 |  | | 43 |  | | 44 | *ImageProcessor* ip2 = new *ColorProcessor*(width,height); | | 45 |  | | 46 | for (*int* x = 1; x < width - 2; x++) { | | 47 | for (*int* y = 1; y < height - 2; y++) { | | 48 | resultat = 0; | | 49 | pixelX = 0; | | 50 | pixelY = 0; | | 51 | resultat2 =0; | | 52 | for (*int* j = -1; j <= 1; j++) { | | 53 | for (*int* i = -1; i <= 1; i++) { | | 54 | *int* hentPx = ip.getPixel(x+i, y+j); | | 55 | // kjøres gjennom konvelusjon | | 56 | pixelX += sobelX[1+j][1+i] \* hentPx; | | 57 | pixelY += sobelY[1+j][1+i] \* hentPx; | | 58 | } | | 59 | } | | 60 | resultat = (*int*) Math.sqrt((pixelX \* pixelX) + (pixelY \* pixelY)); | | 61 | resultat2 = (*int*)(Math.atan2(pixelY,pixelX)\*180/Math.PI); | | 62 | color = (color.getHSBColor(resultat2, resultat, 50)); | | 63 | ip2.setColor(color); | | 64 | ip2.drawPixel(x,y); | | 65 | } | | 66 | } | | 67 | *ImagePlus* im = new *ImagePlus*("Sobel Strength",ip2); | | 68 | im.show(); | | 69 |  | | 70 | } | | 71 | } | |  |

**Oppgave 3**

|  |  |
| --- | --- |
| **Originale bilde:** Bridge.gif | **Gradient Retning** |
| **Gradient Styrke** | **Fargebilde** |
| "Process->Find Edges" i ImageJ. |  |
|  |  |