Computer Graphics – Exercise 1 – Image Processing

Grades

xxxx35073_xxxx00121	99
xxxx11338_xxxx66614	91
xxxx27812_xxxx22335	105
xxxx58070_xxxx57892	93
xxxx87275_xxxx11524	88
xxxx52541	76
xxxx65769_xxxx77072	85
xxxx55717	77
xxxx30590	100
xxxx56289_xxxx59441	99
xxxx97065_xxxx09618	85
xxxx54921_xxxx30159	74
xxxx70388	105
xxxx15149_xxxx34972	70
Average:	89.07

Points

- Sampling 40
- Transformation 25
- Convolution 35
- Dithering 5

Notes

- Process
    - I ran every submission the same series of tests. First I ran the example configuration files (referred to as training configurations). Surprisingly I found many mistakes there!
    - Then I ran it through a series of unseen configurations
    - Finally for each submission I created a special custom configuration file to check it's custom non-linear transformation.

- There were some common format issues that didn't follow the instructions (e.g. gaussian instead of Gaussian).
    - Common mistakes were handled and points weren't taken off
    - However any other errors resulted in loss of points.
- It took me some time to grade this exercise only because I wanted make sure that I wouldn't take away a lot of points for merely misinterpreting the instructions. I wanted to compare everyone's outputs on the same configurations and understand which mistakes were common and which were exceptions.

xxxx35073_xxxx00121

- Almost everything OK. Minor alignment issues.
- Code is too dense. Should use more OO
- Custom kernel direction wrong (-1)
- Dithering isn't Floyd-Steinberg
- Overall: **99**

xxxx11338_xxxx66614

- Rotation doesn't take sampling into account (even on train examples!) (-5)
- text_conv_laplacian.config throws exception! (-7)
    - Exception in thread "main" java.lang.NumberFormatException: For input string: "0 0 0 0 1 0 0 -1 0"

```
at sun.misc.FloatingDecimal.readJavaFormatString(FloatingDecimal.java:1242)
at java.lang.Double.parseDouble(Double.java:527)
at cg10a_ex1.cd10a_ex1.main(cd10a_ex1.java:53)
```

- Different sampling methods aligns image differently (-2)
- Dithering OK (+5)
- Code is too dense. Should use more OO
- Overall: **91**

xxxx27812_xxxx22335

- Excellent
- Good code structure and documentation
- Ditheing OK (+5)
- Overall: **105**

xxxx58070_xxxx57892

- Faild on training config!: green_small_scale_x4_NN.config. Also on another test config which used upscaling (-7)
    - Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: Coordinate out of bounds!

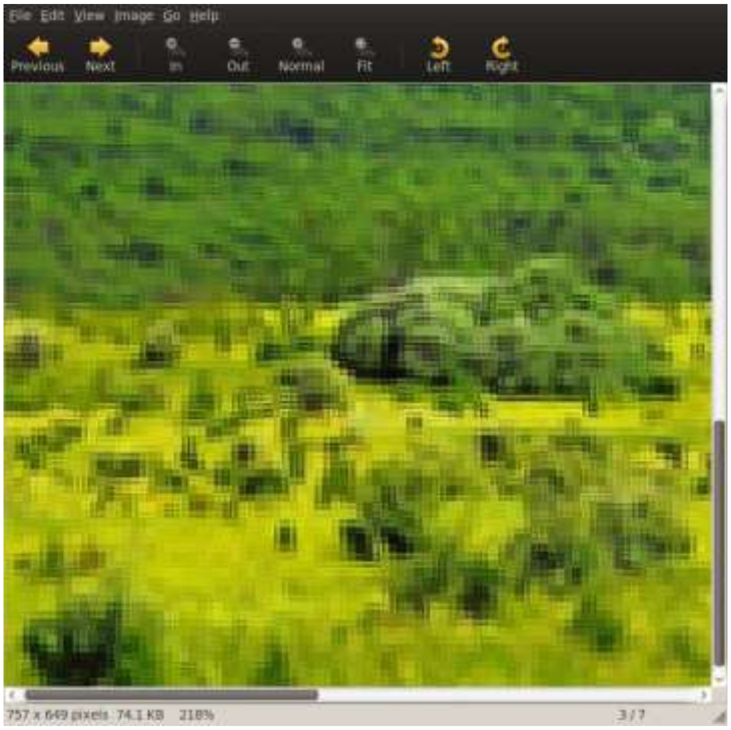
```
          at
sun.awt.image.ByteInterleavedRaster.getDataElements(ByteInterleavedRaster.java:318)
          at java.awt.image.BufferedImage.getRGB(BufferedImage.java:888)
          at cg10a_ex1.resampleNearestneighbor(cg10a_ex1.java:490)
```

```
        at cg10a_ex1.resample(cg10a_ex1.java:456)
        at cg10a_ex1.scale(cg10a_ex1.java:199)
        at cg10a_ex1.main(cg10a_ex1.java:79)
```

- Other than that looks very good
- Code formatted and commented well, but fitting it in a single .java file is very bad practice
- Overall: **93**

xxxx87275_xxxx11524

- Rotation in radians? Why? The example I gave was in degrees! (-1)
- Bilinear interpolation is wrong. Noticeable even on train examples (-15)

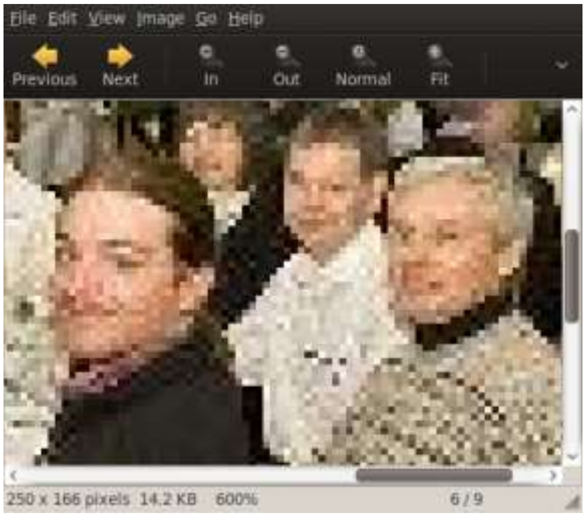


Your result on: green_small_scale_x4_bilinear,

- Custom convolution wrong direction (-1)
- Dithering OK (+5)
- Code is too dense
- Overall: **88**

xxxx52541

- In your doc you write: 1.The algorithm names are as requested, and are case sensitive. This means that "Scale_gaussian" is recognized, but "Scale_Gaussian" will result in an "Unidentified algorithm name". That's all very nice but the parameter is defined as Scale_Gaussian!!
- Why change range of image after convolution? I didn't give such an example (-1)
- Gaussian barely has an effect when downscaled to half! (-8)



Your result with a gaussian downscaling with 0.5

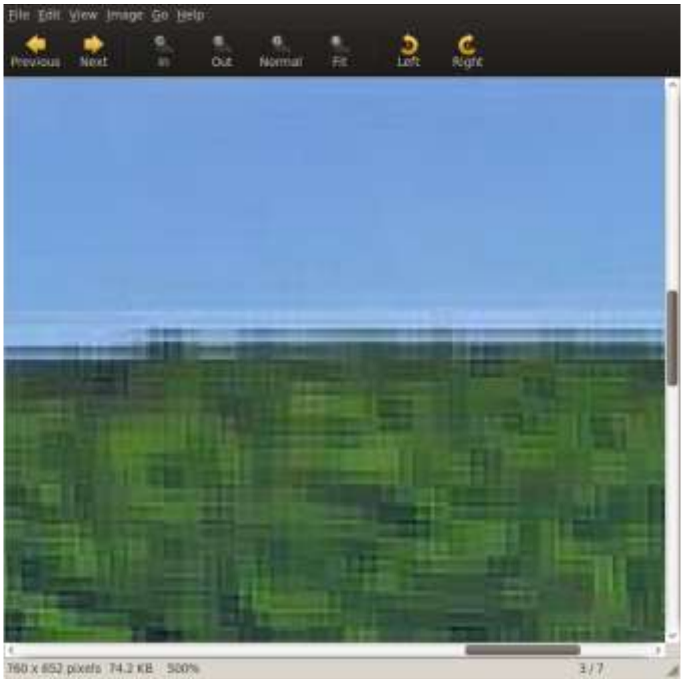
- Bilinear looks the same as linear! No chance to verify Bilinear (-15)
- Many scalings throw exceptions : (-5)

```
        Exception in thread "main" java.lang.NegativeArraySizeException
                at Testing.Testing.convertTo3DArray(Testing.java:393)
                at Testing.Testing.parseConfigurationFile(Testing.java:239)
                at Testing.Testing.main(Testing.java:61)
```

- Dither looks OK (+5)
- Overall: **76**

xxxx65769_xxxx77072

- Everything looks great and works as it should, but bilinear is simply WRONG! (-15)



Your result on: green_small_scale_x4_bilinear,

- No dithering
- Code look good, but the use of package “src” is just wrong
- Overall: **85**

xxxx55717

- Some scaling throw exceptions (-7)
  - Config: [green_small.bmp, OUT_green_small_scale_x4_NN.jpg, Scale_nearestneighbor, 4]

```
STARTING...
java.lang.ArrayIndexOutOfBoundsException: Coordinate out of bounds!
        at
sun.awt.image.ByteInterleavedRaster.getDataElements(ByteInterleavedRaster.java:318)
        at java.awt.image.BufferedImage.getRGB(BufferedImage.java:888)
        at Sampler.sampleNearest(Sampler.java:17)
        at Sampler.sample(Sampler.java:9)
        at Transforms.scale(Transforms.java:38)
        at cg10a_ex1.main(cg10a_ex1.java:58)
```

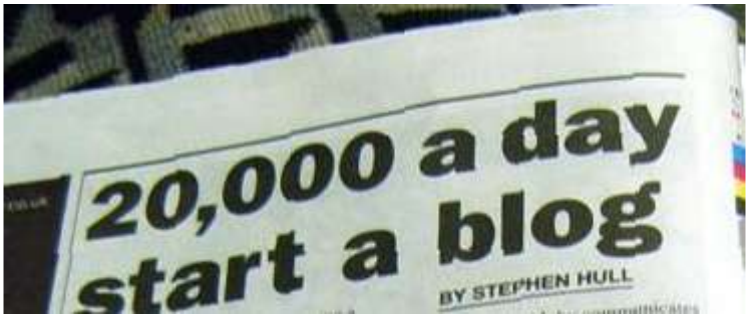
- Predefined convolution methods throw exception (-5)
  - Config: [duck.jpg, OUT_ducks_convolution_blurG.jpg, Blur_gaussian]

```
java.lang.NullPointerException
at cg10a_ex1.main(cg10a_ex1.java:36)
```

- Downscaling doesn't seem to use any sampling other than NN! thus never seen Gaussian (-15)
- Dither OK (+5)
- Overall: **77**

xxxx30590

- Rotation with bilinear looks worse than other submissions (jagged lines) (-3)



Your result on bilinear rotation degree 1

- Custom convolution direction is wrong (-1)
- Bilinear scaling border stretched (-1)
- Dither OK (+5)
- Very good coding
- Overall: **100**

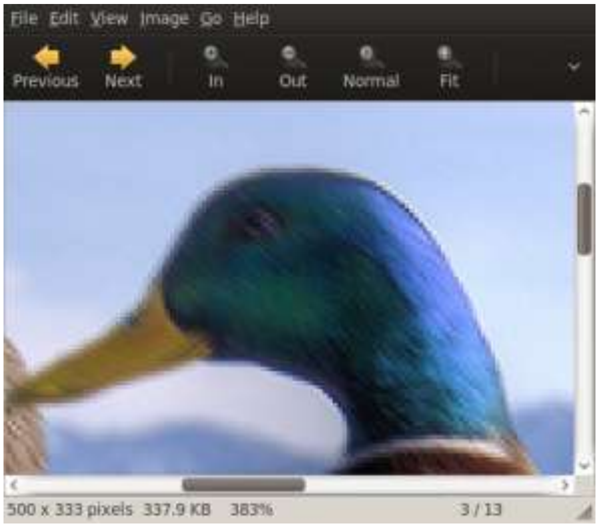
xxxx56289_xxxx59441

- Custom convolution direction is wrong (-1)
- Very good outputs

- Code in one file and no documentation...
- Overall: **99**

xxxx97065_xxxx09618

- Edge detection fails to run (-5)
- Something wrong with convolution, with both custom and predefined (-15)



Duck with your Gaussian blur. Looks more like motion blur

- Dither OK (+5)
- Overall: **85**

xxxx54921_xxxx30159

- Sampling has no effect in rotation (-5)
- Scaling with bilinear makes the image larger than with NN! Making it more cropped. Should not happen! (-5)
- Why change range of image after convolution? (-1)
- Rotated image doesn't fit into canvas! (-5)
- Gaussian doesn't run! (-10)
    - Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: Coordinate out of bounds!

```
        at sun.awt.image.IntegerInterleavedRaster.getDataElements(IntegerInterleavedRaster.java:219)
        at java.awt.image.BufferedImage.getRGB(BufferedImage.java:888)
        at cg_ex1.Sampling.gaussian(Sampling.java:60)
        at cg_ex1.Warping.Scale_gaussian(Warping.java:48)
        at cg_ex1.cg10a_ex1.main(cg10a_ex1.java:66)
```

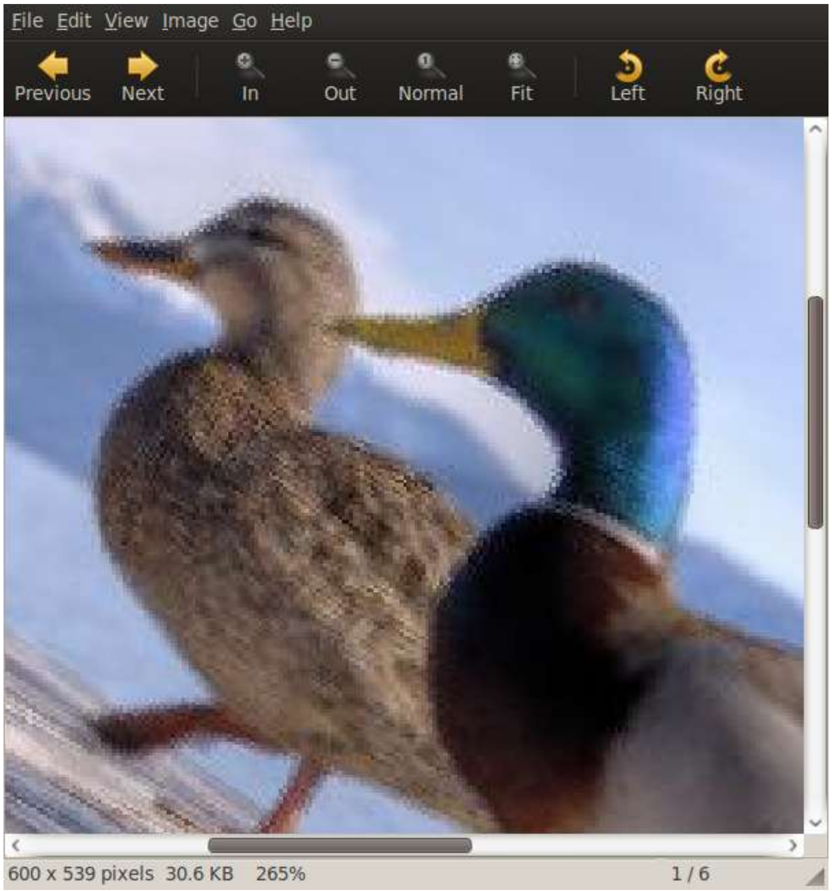
- Overall: **74**

xxxx70388

- Best and fastest work
- Dither OK (+5)
- Overall: **105**

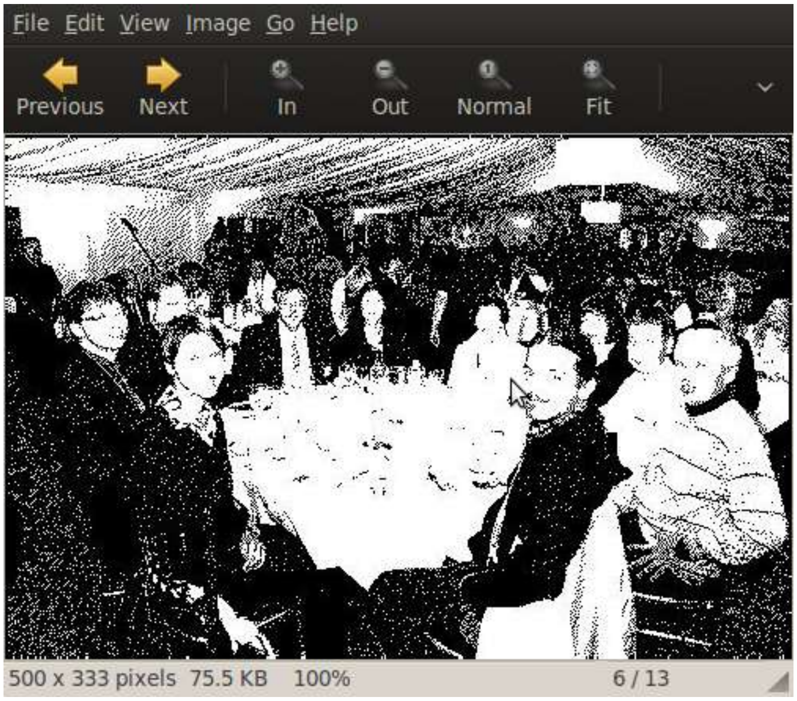
xxxx15149_xxxx34972

- Run: /home/chen/GRADER/res/input/level0/text_conv_laplacian.config Failed (-5)
  - Exception in thread "main" java.lang.NumberFormatException: For input string: "-1 -1 -1 -1 8 -1 -1 -1 -1"
    ```
    at sun.misc.FloatingDecimal.readJavaFormatString(FloatingDecimal.java:1242)
    at java.lang.Double.parseDouble(Double.java:527)
    at cg10a_ex1.main(cg10a_ex1.java:930)
    ```



Your bilinear on the training example

- Bilinear is WRONG (-15)
- Gaussian downscaling 0.5 looks exactly like NN (-10)
- Dithering is wrong (0)



Your dithering with a parameter of 2

- Overall: **70**