

Assignment 3
Image Manipulation — using Java
Due: 12 noon on Monday 23rd October, 2006

Outline

Your aim in this assignment is to write some computer image manipulation tools for **Portable PixMap** (ppm) images. The *assignments* page of the COMP1100 web site has a description of the representation of images on computers and the ppm format in particular.

A number of Java classes to represent channels, pixels and images, and a driver program to test your work, are provided in `/dept/dcs/comp1100/public/assts/asst3/` and on the course web site.

Some sample ppm images are available in `/dept/dcs/comp1100/public/assts/asst3/images` and on the course web site. If you wish to make your own images (make sure they're not too big) you can convert from other formats to ASCII ppm with a tool like Photoshop or the Gimp. *Eye Of Gnome* (eog) is a useful image display tool usually distributed with Linux, but there are others. Be sure to turn off interpolation to see true pixels when zooming in.

Required Tasks

You are required to design, develop and test several image manipulation tools. Since they are naturally classified as PPM methods, they will belong in the PPM class. Some of them need to manipulate individual pixels, so those methods will be included in the Pixel class.

1. Threshold

Add a method to the PPM class which implements the threshold filter. The method will have the following signature:

```
void threshold (int level)
```

The brightness of a pixel is the average of the RGB values (see the `desaturate` function). The argument to `threshold` is a value between 0 and 255. All pixels at least as bright as this value are sent to white, and all pixels darker than it are sent to black.

Here is the Borat .ppm sample image after having the threshold filter applied to it:



2. Horizontal fold

Add a method to the PPM class which takes an image and adds a copy to the left of it. The copy should be flipped horizontally. The method should have the following signature:

```
void foldH()
```

Here is the Boratt .ppm sample image after having the horizontal fold filter applied to it.



Hint: This method will have the same overall structure as the `doubleScale()` method included in the PPM class. For each pixel in a given row, add it twice to the new row. Once at the front of the new row, and once at the back. Check the Java API documentation on the course web site to find out how to do this.

Free Choice Tasks

Successful completion of the required tasks above, to the satisfaction of your tutor and lecturer is sufficient to achieve up to 80% on this assignment. For marks better than that, you will need to develop some image manipulation tools of your own design.

There are two basic categories: *pixel-based*, where you modify each pixel in some manner (such as desaturation); and *image-based*, where the structure of the image is modified in some way (such as scaling). You should develop one of each kind.

Pixel-based

Here are a few suggestions for possible pixel-based manipulation functions. You are welcome — encouraged — to come up with your own ideas, but be sure to include an explanation of the technique and intentions in the comments of your code. You may need to do some research to be sure you understand the technical definition of each manipulation.

- **invert**. Convert an image to its colour negative. Look at the information on colour representation on the web page accompanying this assignment. For example, cyan is the inverse of red and yellow is the inverse of blue.
- **darken**. Make an image twice as dark as it presently is.
- **contrast**. Increase the contrast of the image. You should be able to find a formula for changing contrast on the web or in the library.
- **posterize**. Reduce the color depth of an image. First you will need to find out this means.

Image-based

Here are a few suggestions for image-based manipulation functions. Once again, you are encouraged to come up with your own ideas, but give us a clear explanation.

- **Rotate the image 180°**. This is easy.
- **Rotate the image 90°**. This is not easy.
- **Scale again**. Rather than simply doubling the size of the image, provide the function with an argument to specify the scaling factor.
- **Create a grid of four versions of the original image flipped along the edges of the original**. Other variations are possible.

Submission

You should submit all the modules that you have written and wish to be included in your assessment, by the published deadline. Late submissions will be penalised 10% for each day (or part thereof) past the deadline.

Assignment submission will be electronic. Submit your assignment with the command:

```
submit comp1100 Asst3 <file names>
```

You should submit a `.tgz` archive file containing all your code, as per Assignment 2.

The filename should be: `Asst3-uXXXXXXX.tgz`, where `XXXXXXX` is replaced by your own student number. *Do not include any image files in your archive.*

You should check that your submissions have been successful by clicking the **View Marks** button in **StReaMS** (<http://cs.anu.edu.au/streams/>). You should see a list of files together with the time and date of submission.

Important: For the sake of speedy marking, please modify `Driver.java` to run each of your filters in turn. Follow the same process as the example filters already there.

To invoke the main program on an image file, use the command:

```
java Driver someFile.ppm
```

This should save number of output files of the form `someFile-FILTER.ppm` where `FILTER` is replaced by the names of the filters supported by your program.

Assessment

This assignment forms 10% of your total assessment for COMP1100.

Successful completion of the *Required Tasks* to the satisfaction of your tutor and lecturer is sufficient to achieve a mark of 80% on this assignment. For marks better than a that, you will need to develop some image manipulation tools of your own design, as explained in *Free Choice Tasks*.