

Laboration 2 - TDDC78

Jonathan Karlsson - jonka293 - 890201-1991
Niclas Olofsson - nicol271 - 900904-5338

17 maj 2013

1 Program description

1.1 Threshold filter

This program calculates the average intensity of the image and makes all pixels with a higher-than-average value white, and all other pixels black.

The whole image is read on the root node. The root node also calculates the average intensity of the image. Now a data structure is created for each new thread that is to be created. The structure contains the intensity level, the image itself as well as start and end value that defines where on the image each thread is going to work. The thread is created and each of them start to work in their part, since memory is shared the whole image is done when each thread is done. Each thread is operating on different parts of the image so there is no risk of memory being overwritten.

When each thread is done the root writes the resulting image to disk.

1.2 Blur filter

For each pixel this program calculates the average color of the neighbors with a given radius, of the pixel and sets the pixels color to the average. This gives a blurred effect over the whole image with a small radius giving almost the same image back and a big radius gives a very blurry image.

The whole image is read on the root node and then the intervals that each processor is going to work in is calculated and transmitted with `scatterv`. The reason that `scatterv` is necessary is because there's a lot of dependencies between each chunk of the image so each processor need a bit more data so it can read all neighbors, otherwise there will be some broken lines in the image. Now each chunk is iterated by the blur filter and then transmitted to the root node with `gatherv`, the root node must take special care to get rid of some overhead created by the sending of extra data. This is done by pointing a bit forward in the array storing the image.

2 Execution times

Below are figures showing the execution times.

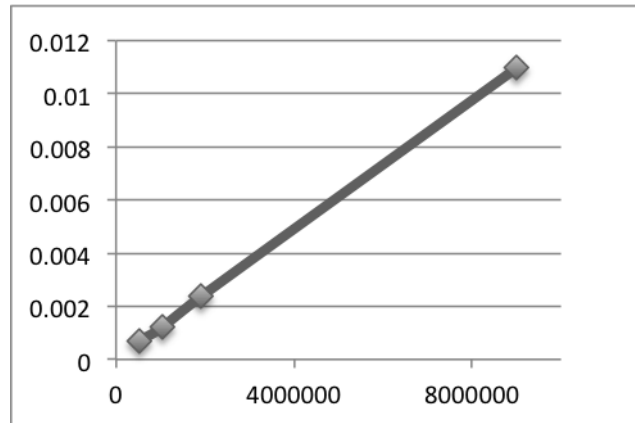


Figure 1: Using the same amount of nodes on different images. (Thresh)

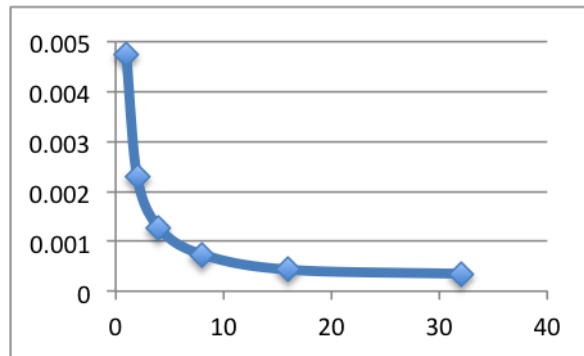


Figure 2: Using the same image with different amount of nodes. (Thresh)

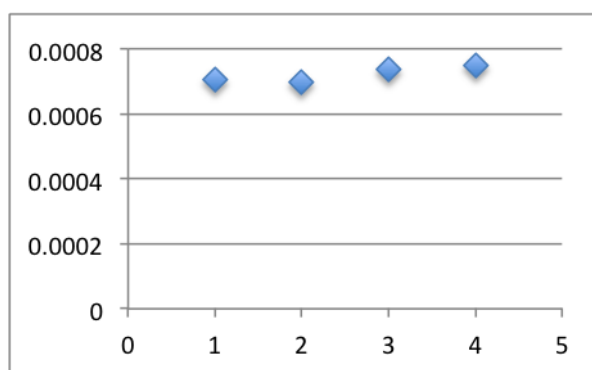


Figure 3: Scaling the image at the same rate as the number of nodes. (Thresh)

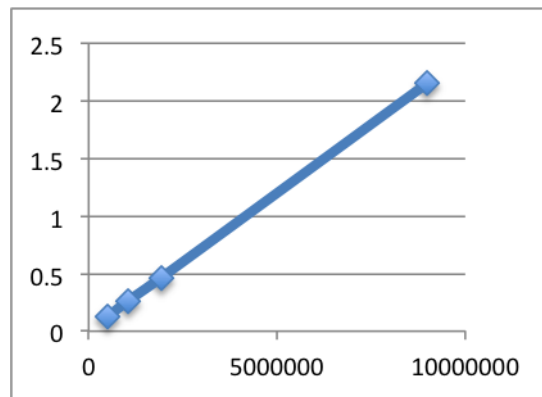


Figure 4: Using the same amount of nodes on different images. (Blur)

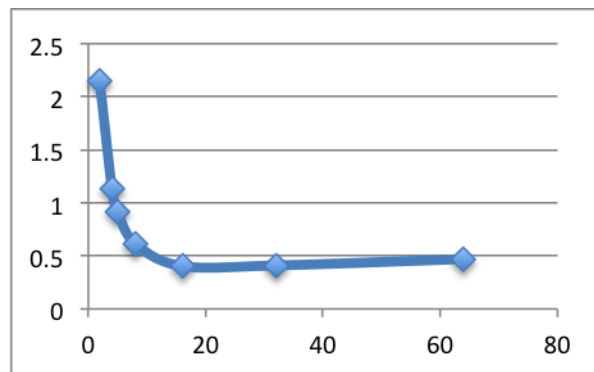


Figure 5: Using the same image with different amount of nodes. (Blur)

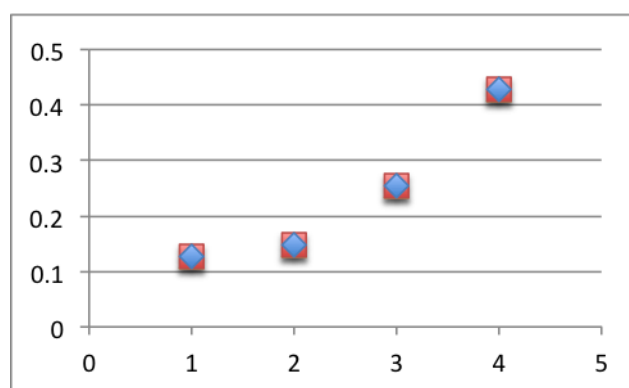


Figure 6: Scaling the image at the same rate as the number of nodes. (Blur)