Bishop’s University

CS463 winter 2023

Computer Vision-Image processing

Instructor: Dr. Elmehdi Aitnouri

Assignment management

Number: 2

Name: Analysis of binary images

Posting date: 2023-03-16

Delivery date: 2023-03-30, midnight

Team member

Last name	First name	ID	Email
Sparnaay	Jerome	002320723	jsparnaay21@bishops.ca

Finally a somewhat good UI has been implemented. Unfortunately, displaying the image automatically is harder with a compiled language like C, so for the sake of intercompatibility with windows, one still will needs to open the file manually at the specified place. This should corrected with the intelligent scissor assignment, where I'll implement a program using ssl to enable the required functionality. The previous assignment functions have been unchanged and is still available by choosing it at the opening of the program.

The whole UI is terminal based and runs in a loop to enable all the assignment's tools to be used. It loads the image in with the name of it and the name has the root at the root of the program.

Edge detection

Edge detection is all made using 2 functions in file edgeDetector.c . Both edge detector have a direction and reverse parameter. The directions have defined macros specified in the file. The direction uses binary & to determine if the values should be added to the sum of the pixels mask. The horizontal direction has bit 1 and vertical 2, so when 3 is given, both values pass and adds to there respective sum. The reversed simply multiply by 1 or -1 the array that is used to calculate the mask of the pixel. Finally both functions copy the Img to keep the data safe during the convolution the values are used to get the values of the convolution and the Img parameter is the one where the values are to applied. This is where the similarities end.

The gaussian based one takes the sigma, direction and whether it is reversed or not. The mask is calculated with the sigma and the resulting array is stored to be used as a mask. The magnitude of both is calculated by squaring each directions sum and square rooting the sum of both them.

The other edge detection are all made possible by the generic edge detector. This one has two additional parameter: centered and sobel. Without going too far into it, both are used in a way to create a mask and logic that will allow to have the appropriate effect.

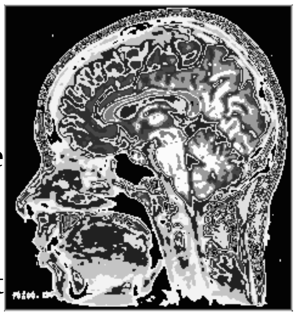
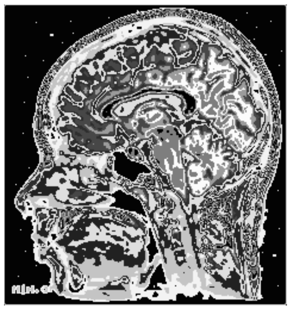
Filters

Preprocessing filters are separated in two parts: histogram manipulation and smoothing.

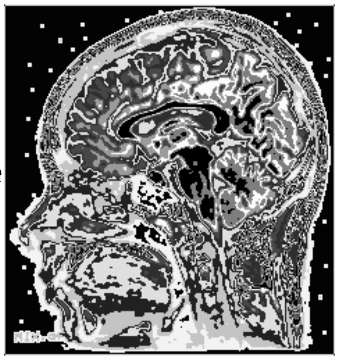
The histogram manipulation filters both take two parameter. For the linear one it takes a and b according to the function ax + b = y, where x is the current value of the pixels and y the value after the filter. a and b are given by the user. The power law also need 2 values from the user and applies as per the formula. Both are called with the histogramManipulation function that it itself takes the img, the function to apply the filter (this allows for a future upgrade to use opencl or cuda kernels to apply filters), the variables for the filters and the filter function itself as a void pointer.

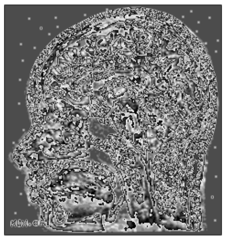
For the smoothing filters, they are called using the smoothingFilter function and passing either the applyFilter and a filter (like gaussian or averaging) or no filter function and another function (median). The averaging and gaussian are quite similar and has their respective functions that will set a filter for the function to use. The median takes all the pixel in the mask and uses the standard qsort* and returns the middle element to be inserted in the Img.

*Fun fact that I learned by opening an old standard c library book from 1998, qsort is a function that is in the stdlib.c and is a generic c function that can sort arrays of any type.

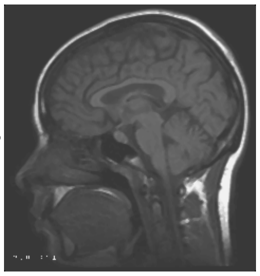



Applying a power function, a Gaussian edge detector and a power function again these are images I was able to produce to overblown the contrast of the brain and the wrinkles. They also has a nice look visually and simplifies the structures, which although makes the information that can be gather less scientific, makes the structures easier the separate visually. For example the two small sausages are clearly defined in the middle of the brain and the main brain on top and the part at the back are very easily discernible. These could be an image that one could use to show and explain the head and the brains to a younger audience with the image being more appealing and using an intelligent scissor, in could be recolored. The files are located in output/assign1/comp1. These might not be recreatable with current code, as an unintended bug was present when these were created where the values to be put at each pixel spot could overflow and go back to 0 if they were bigger than 255 instead of being simply 255.





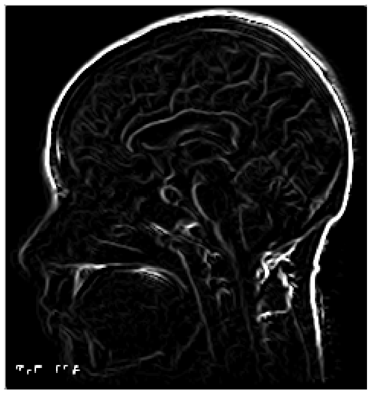
Last for the nice looking, but not not really useful is these, which happened when using a mix of power law, gausian filter and edge detection that sharpened the image too much that is made noise, but keeps the edges and general form of parts. It looks somewhat arty in a weird “this is a complete failure that works” way.

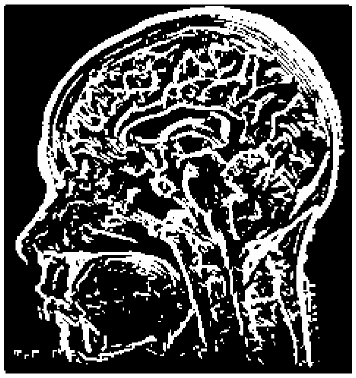


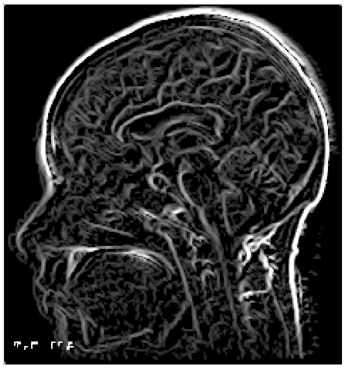
For more serious result, here is a preprocessed image with linear histogram and median smoothing preprocessing applied to remove the pepper and salt. The linear histogram manipulation was used to brighten the image visually and makes the edge detection more pronounced. The median filter was used to remove the pepper and salt and keep the edges as opposed to the 2 other smoothing filters.

We can see that the result is quite good, but the text is made unreadable by the median smoothing. This is the image that the rest of the edge detection will use.

Here is 3 photo of the edge detection using a sobel operator of 2. The first is one is the one right after the edge detection. The second is thresholded to be a binary image and the third has a power law filter applied.







For the Gaussian edge detection, again 3 images. In order, original after edge detection, thresholded, thresholded and (power law followed by linear).