Trabalho 3

Fundamentos de Programação – 2017.1

Gabriel Maia Gondim – 1936352

Contribuição dos membros da equipe

De toda a equipe, Gabriel ficou responsável por pensar em uma solução, fazer a função, os testes, otimizar os códigos, verificar e corrigir os erros, comentar e escrever o pdf.

Ideia geral, desafios e como foram superados

Os filtros

Estava pensando em um jeito de encontrar os centros dos círculos, mas percebi que precisava "limpar" a imagem antes, pois o ruído dificultaria a busca pelos círculos.

Criei dois filtros, o filtroMediana, que passa por todos os pixels exceto os das bordas e coloca nesses pixels a mediana do seu valor com o de todos os 8 pixels ao seu redor, e o filtroPixel, que zera todos os pixels com valor menor que maior*FRAC (FRAC é uma macro). Depois tirei o filtroPixel, pois percebi que poderia apenas colocar nas condições para pegar apenas valores maiores que maior*FRAC.

A função

Para encontrar os centros pensei em procurar um ponto que não seja escuro (antes, quando usava filtroPixel, procurava por pixels com valor diferente de zero, agora procura valores maiores do que maior*FRAC) e quando encontrar esse ponto, o programa cria um "quadrado" em volta do pixel, olhando na esquerda, direita e abaixo do pixel. Se tiverem pixels brancos em uma dessas direções, ele aumenta o tamanho do quadrado nessa direção e continua nessa busca até não encontrar mais pixels brancos. Nessa parte, antes tinha pensado em pegar o ponto médio do quadrado, mas o quadrado nem sempre estava bem ajustado, o que causava vários erros, então resolvi usar um cálculo semelhante ao do centro de massa, em que ele incrementava sempre que achava um pixel claro e somava a sua distância, tirando a "média" entre eles depois.

Após o programa encontrar o primeiro círculo na matriz de dados, ele guarda o centro desse círculo e o seu tamanho em números de pixels claros em cord e tam, respectivamente. Sempre que o programa encontra um centro de algum aglomerado claro, além de guardar em uma posição desses vetores o tamanho e as coordenadas, ele ordena os vetores com relação aos tamanhos, deixando os centros e tamanhos dos dois maiores aglomerados nas duas últimas posições do vetor, sendo que as duas primeiras são usadas apenas para guardar temporariamente as coordenadas e para trocar a ordem destas. Isso faz com que sempre que o programa encontra todos os aglomerados na imagem, ficam na ponta do vetor os dois maiores, que são os dois círculos.

Agora ele simplesmente guarda as coordenadas nos parâmetros de saída *l e *r e retorna o ângulo entre os círculos.

Correção de erros

Quando testei a função pela primeira vez, percebi que todos os centros tinham um erro exatamente 1 pixel abaixo do normal nas duas coordenadas. Depois de verificar bastante o código, percebi que as coordenadas estavam sendo passadas como int e por isso sempre arredondavam para baixo. Então, somei 0.5 nas duas para caso dê uma coordenada com parte decimal acima de 0.5, seja arredondada para cima.

Depois de ter feito o programa testei sem os filtros e percebi que o número de erros era o mesmo, então resolvi apagar os filtros e deixei o programa sem os filtros (para acelerar o tempo de execução).

Também tive um problema com relação ao ângulo, pois tinha várias vezes que ele resultava em zero, mas eu sabia que não devia haver tantos círculos exatamente na horizontal, foi aí que percebi que estava fazendo o cálculo do ângulo com números inteiros e estava arredondando sempre para baixo, aí mudei o código para que calculasse como double.

Outro problema foi que fiz o programa no linux, mas quando fui testar no windows, quase todos os testes estavam dando errado, tendo números aleatórios nas coordenadas. Depois de procurar por um tempo percebi que não tinha inicializado o vetor tam[4], e que quando tinha uma posição com um valor maior que os tamanhos dos centros, o programa organizava o vetor e ele ficava como sendo um dos centros, tendo assim números “perdidos” na memória como coordenadas dos centros.