

# EP 2 - Relatório

Felipe Constantino de Oliveira

MAC0417/5768 - Visão e Processamento de Imagens - IME-USP

## 1 Introdução

Este trabalho contém arquivos *.py* e foi utilizada a seguinte configuração:

- Python 2.7
- OpenCV 3.4.1
- Numpy 1.14.3
- Matplotlib 2.1.2
- Scipy 1.1.0

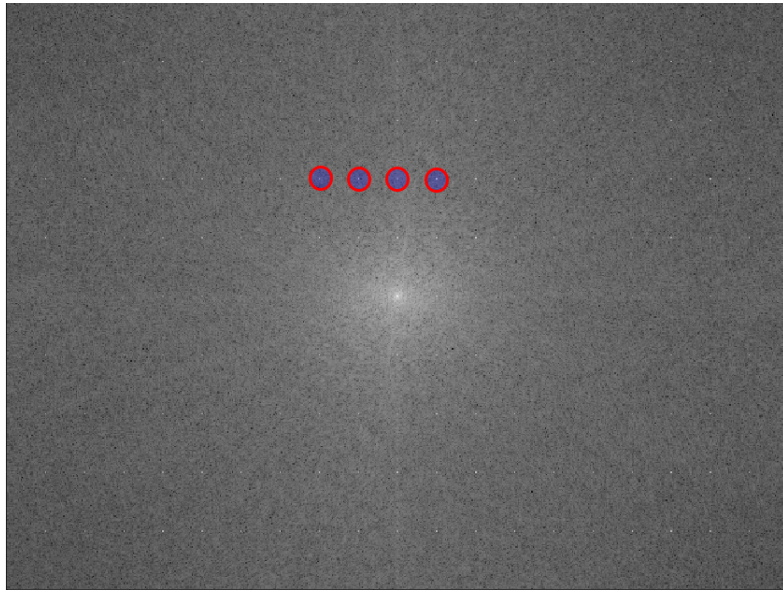
## 2 Problema 1

O primeiro desafio era utilizar os conceitos de Fourier para eliminar o ruído periódico na imagem do leopardo abaixo:

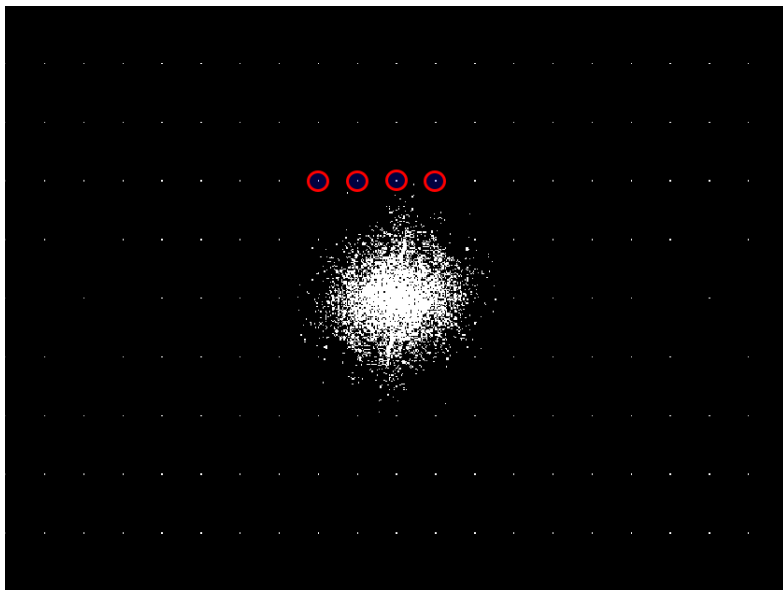


Figura 1: Imagem com Ruído periódico

Para isso, vamos analisar o espectro em Fourier. Nota-se pequenos pontos (alguns circulos em vermelho para facilitar a observação) que parecem estar afetando a qualidade da imagem. Podemos também aplicar uma limiarização para facilitar a visualização conforme imagem abaixo:



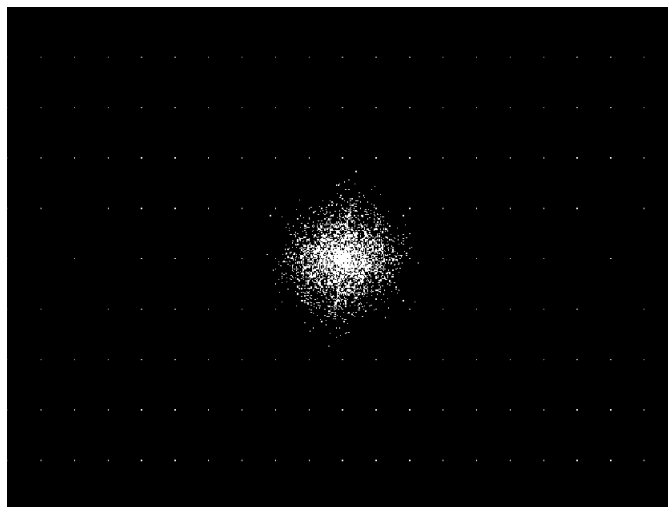
(a)



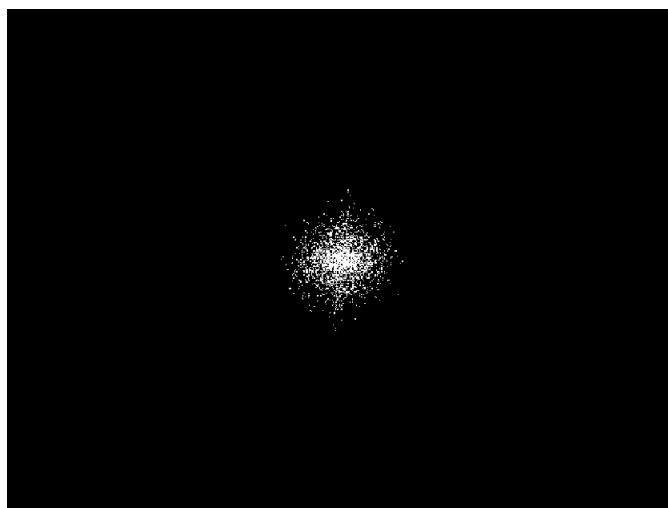
(b)

Figura 2: Figura (a) contém o espectro e (b) com threshold de 60%

Além disso, podemos comparar o espectro em sua parte real e imaginária. Neste ponto é importante notar que existem tais pontos somente na parte real.



(a)



(b)

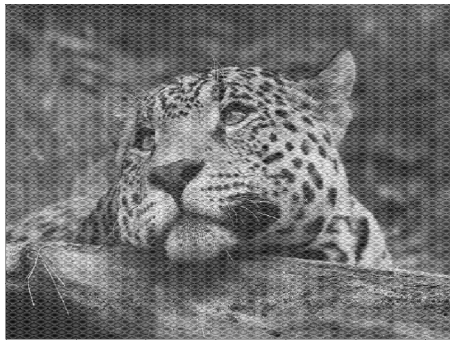
Figura 3: (a) Spectrum com threshold da parte real e (b) Spectrum com threshold da parte imaginária

Como somente a parte real possui estes valores acima do comum (que são os que geram os ruídos periódicos) podemos trabalhar somente em remover tal ruído periódico somente na parte real do espectro.

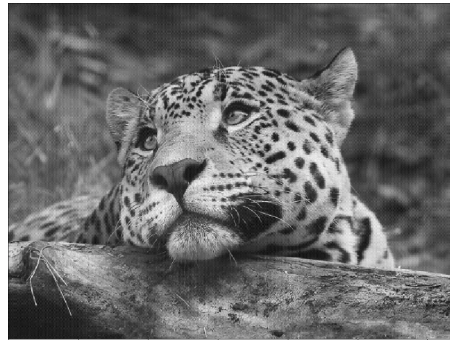
Para remover os pontos que geram o ruído periódico, o algoritmo precisa de 3 parâmetros: (1) o espectro, (2) Um valor  $s$  para o tamanho de um lado ímpar de um kernel quadrado ( $k = s \times s$ ) e (3) um valor  $t$  que é um limiar de aceitação, onde  $t > 1$ . Este limiar é necessário para não obter pixels

O algoritmo compara o valor absoluto do centro do kernel com os valores absolutos de seus vizinhos dado o kernel  $k$ . Se o pixel no centro do kernel é maior que todos os seus vizinhos vezes o limiar  $t$ , o novo valor do centro é a média entre os vizinhos.

Na prática, alguns testes foram feitos e o melhor foi utilização de uma janela  $5 \times 5$  onde  $m = 2$ , ou seja, o valor do centro é pelo menos 2 vezes maior que valor máximo de seus vizinhos. Como resultado, temos a imagem (b) abaixo:



(a)



(b)

Figura 4: Figura (a) imagem de entrada (b) imagem de saída.

### 3 Problema 2

O problema número 2 também tinha como objetivo restaurar uma imagem corrompida, só que desta vez com um ruído impulsivo com diversas regiões pretas.

Para restaurar, foi pedido que se utilizasse o atributo área das bacias da Max-tree, no entanto como os ruídos a serem removidos são pretos, é necessário utilizar a Min-tree (dual da Max-tree) ou pode-se complementar a imagem e utilizar a Max-tree. Como estamos utilizando na matéria o *siamxt Toolbox*, que só trabalha com Max-tree a segunda opção parece mais adequada.

Como os ruídos são bem pequenos, ou seja, possuem poucos pixels conexos, um atributo de área pequeno já é o suficiente. Isso pode ser demonstrado inclusive na imagem abaixo:

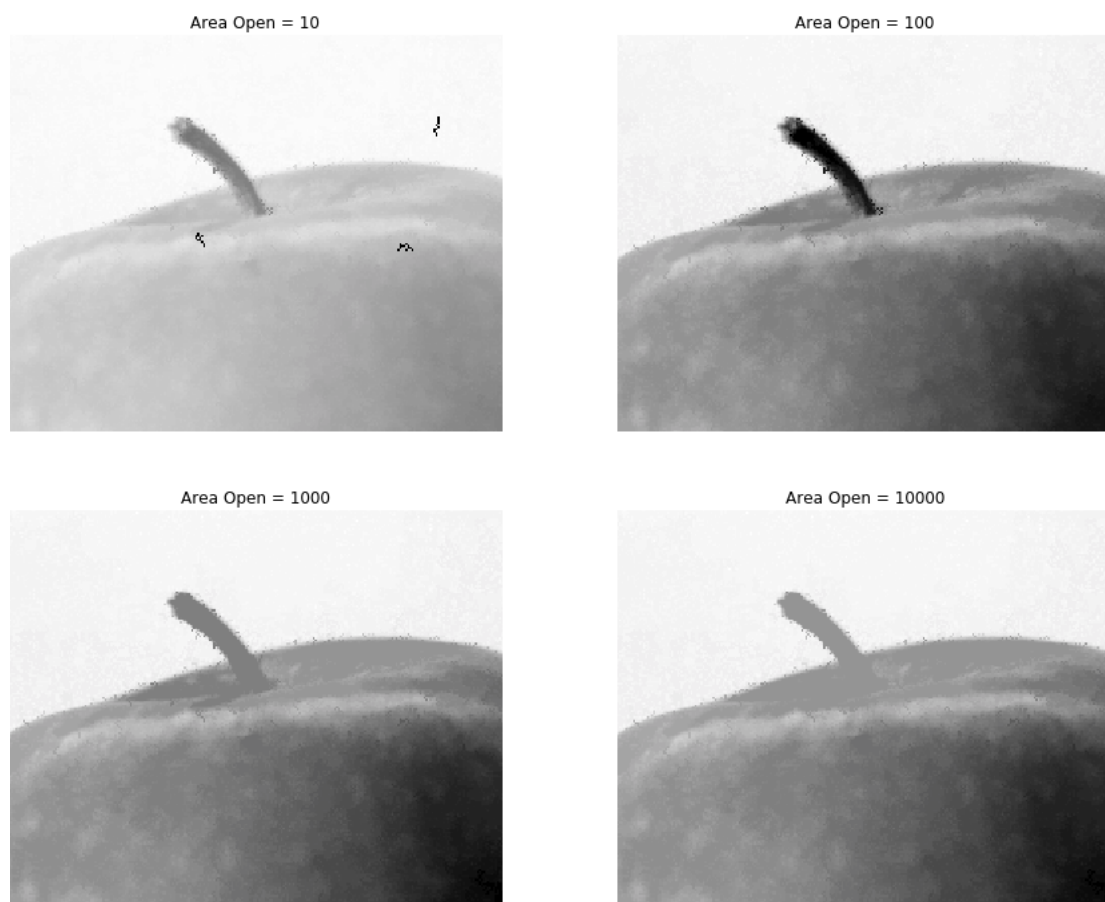


Figura 5: Detalhes da maçã após diversas podas com valores diferentes do atributo área

Podemos ver que com o atributo área = 10, temos uma boa quantidade de ruído eliminado. Porém ainda sobraram alguns grandes. Quando utilizamos uma poda com valor do atributo área muito grande como 1000 já acabamos afetando a imagem. Veja que o talo da maçã começou a perder o verdadeiro contraste. Além disso, se exageramos, como no exemplo utilizando 10000, além do talo, outros componentes da maçã acabam sendo prejudicados.

Como o valor 100 pareceu adequado durante os testes, a imagem resultante após a aplicação da poda por área = 100 é a seguinte:

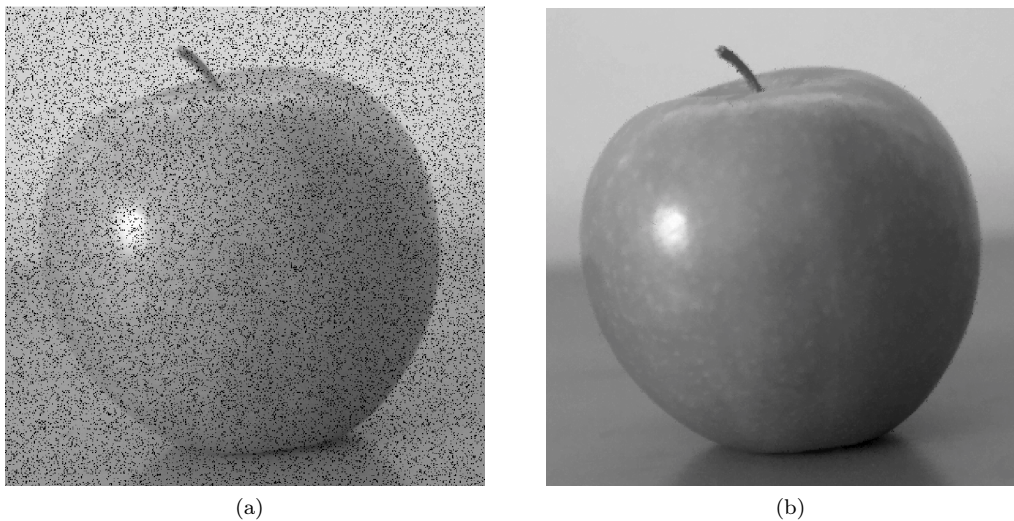


Figura 6: Figura (a) imagem de entrada (b) imagem de saída.

## 4 Problema 3

O problema 3 tem como objetivo segmentar o joelho utilizando watershed. Só que para isso, acaba-se utilizando diversos conceitos apresentados na matéria a qual detalharemos a seguir.

### 4.1 Cálculo do Gradiente Morfológico

Em morfologia matemática, Seja uma imagem em escala de cinza  $I$  e um kernel  $k$ , Gradiente Morfológico é a diferença entre a dilatação e a erosão desta imagem, ou seja,  $G(I) = (I \oplus k) - (I \ominus k)$ .

Conforme pedido pelo exercício, utilizaremos um disco de raio unitário como kernel, ou seja,

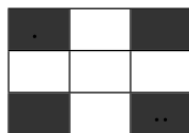


Figura 7: Disco de raio unitário

Gerando a seguinte imagem:

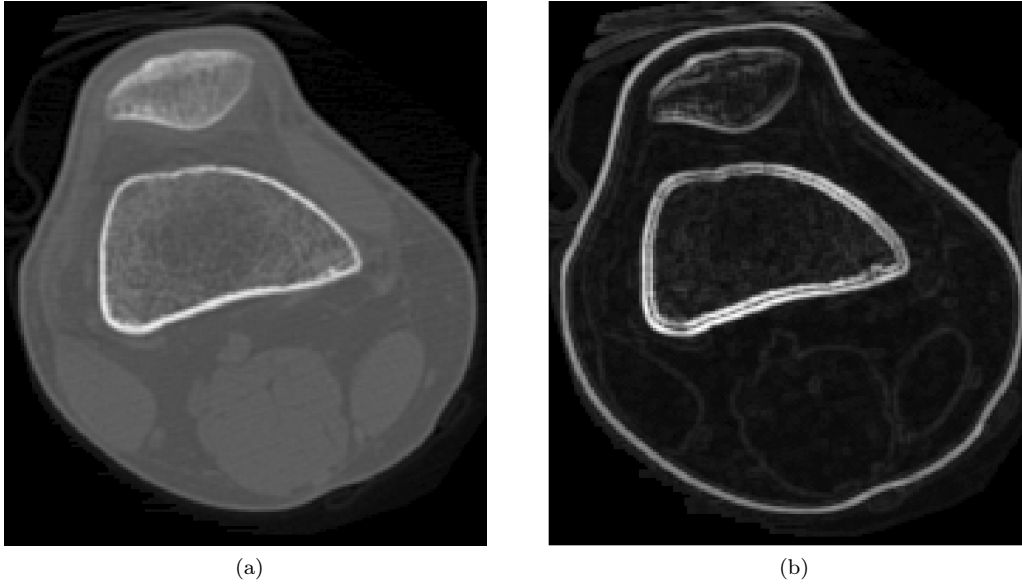


Figura 8: Figura (a) imagem de entrada (b) imagem de saída.

## 4.2 Filtro das 8 bacias e Marcadores

Agora precisamos aplicar um filtro em cima do resultado do gradiente morfológico. Similarmente ao problema 2, foi necessário complementar a imagem e utilizar a Max-tree, conservando as 8 folhas da Max-tree com a imagem complementada de acordo com o atributo calculado *volume*.

Como resultado, tivemos outra outra imagem muito parecida com a do gradiente só que filtrada. Após isso, novamente calculamos a Max-tree da imagem resultante do passo anterior e dessa vez selecionamos somente as folhas. Como resultado, temos as 8 menores bacias da imagem.

Estas bacias serão utilizadas como marcadores para o watershed. Para isso, foi necessário rotular cada um dos componentes conexos resultantes com uma cor, conforme imagem (a) abaixo. Após isso, rodamos o algoritmo de watershed, resultando na imagem (b) abaixo

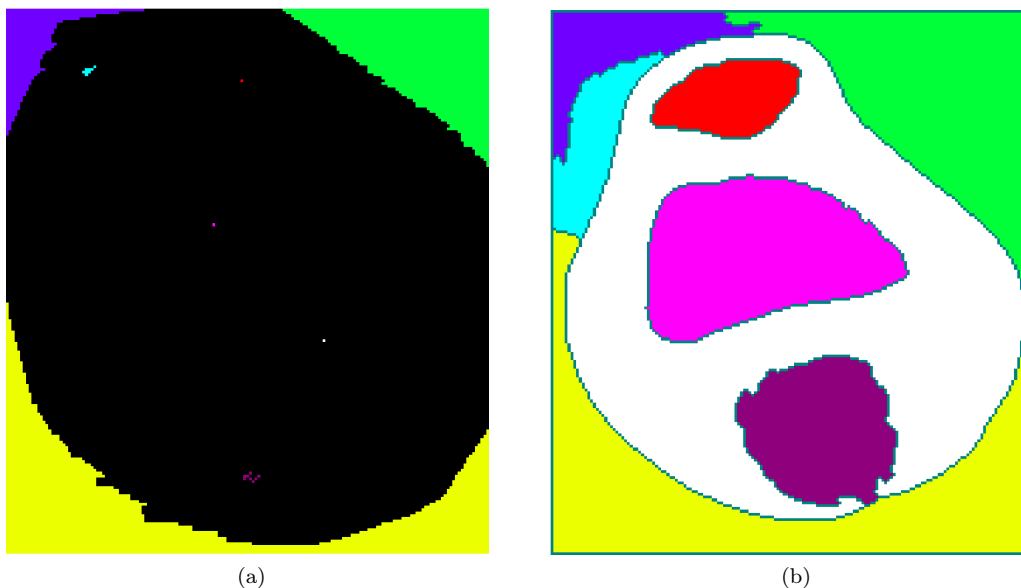


Figura 9: Figura (a) imagem de entrada (b) imagem de saída.

## 5 Problema 4

O problema 4 é um desafio de conseguir selecionar os caracteres alfanuméricos da melhor forma possível. Como primeiro passo, foi necessário utilizar filtros morfológicos para remover os fios de cabelo. O filtro com melhor resultado foi um fechamento de um elemento estruturante  $5 \times 5$  circular, conforme ilustração abaixo

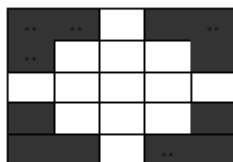


Figura 10: Elemento estruturante circular  $5 \times 5$

A motivação por ele ser circular é tentar pegar a ondulação do cabelo, porém mantendo o aspecto arredondado das letras. Como resultado ainda tivemos alguns cabelos, principalmente aqueles que tiveram ficaram com sobra, resultando em um fio um pouco mais alongado



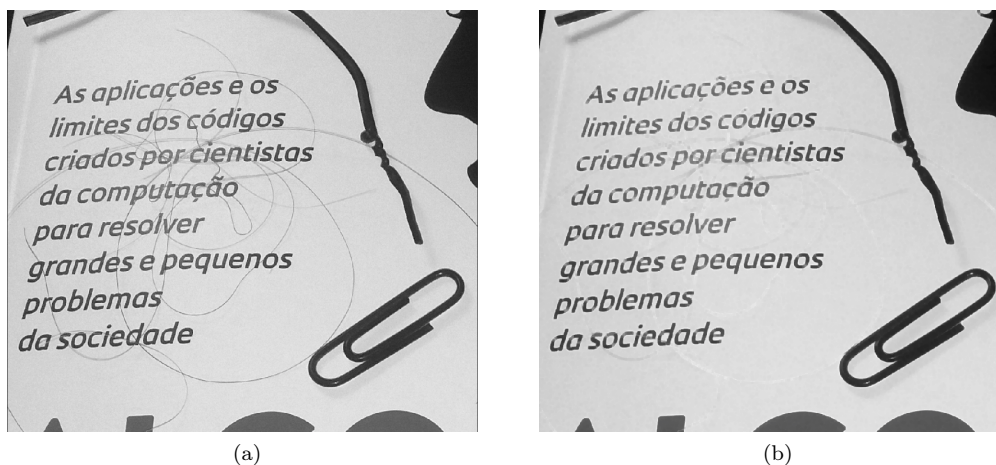


Figura 11: Figura (a) imagem de entrada (b) imagem de saída.

Após este filtro, o objetivo era conseguir retirar as letras da revista. Utilizou-se diversos filtros detalhados abaixo:

- **Largura e altura mínimas e máximas da Bounding-box:** A bounding-box, ou caixa delimitadora, é uma boa forma de eliminar letras que não possuem um certo aspecto. O filtro foi pensado levando-se em consideração diferenças por exemplo entre a letra “T” e a letra “m”. Veja que elas possuem aspectos bem diferentes: o “T” tem altura grande, porém largura pequena, Já o “m” possui uma boa largura mas não muita altura.
- **Razão de retangularidade mínima:** tem por objetivo encontrar qual a porcentagem de preenchimento dentro da bounding-box, sendo assim possível eliminar o restante dos cabelos finos, já que eles provavelmente geraram uma caixa delimitadora mínima grande porém utilizando muito pouco seu preenchimento
- **Altura mínima:** A altura remove ruídos indesejados da imagem, removendo componentes isolados com padrão de brilho fora do esperado.

Como resultado dos filtros acima, pudemos gerar a imagem abaixo

As aplicações e os  
limites dos códigos  
criados por cientistas  
da computação  
para resolver  
grandes e pequenos  
problemas  
da sociedade

Fabrica da  
Qualcomm  
poderá inserir  
o Brasil na  
cadeia produtiva  
do chip

Queda de  
meteorito há 250  
milhões de anos  
no centro do país  
gerou terremoto  
e tsunami

A MEI forum  
de empresários  
tornou-se  
protagonista  
no debate  
sobre inovação

Pesquisadores  
repensam lugar  
do modernismo  
paulistano  
na história da  
literatura brasileira

Nova política de  
biocombustíveis  
premia produtores  
mais eficientes e  
deve ampliar  
a oferta

Figura 12: Resultado após filtros na Max-tree