Projeto II - Introdução ao Processamento de Imagens

1st Lucas Junior Ribas
Departamento de Ciências da Computação
Universidade de Brasília - UnB
Brasília, Brasil
ribasprojectt@gmail.com

***Resumo*—Foram desenvolvidas uma série de técnicas específicas em processamento de imagens digitais para este projeto. Neste contexto, iremos entender os conceitos envolvendo morfologia matemática aplicado ao processamento de imagens, bem como suas aplicações práticas para extração e descrição de regiões específicas de uma imagem. Além disso, serão abordados em detalhes os processos de segmentação de imagens, explorando o algoritmo K-Means para K classes em multidimensões.**

***Index Terms*—k-means, morfologia matemática, binarização, clusters, abertura, fechamento, classes, seguimentação**

I. Introdução

O objetivo proposto para o este segundo em projeto de processamento de imagens digitais, é compreender os conceitos de morfologia matemática em processamento de imagens, e além disso entender a utilidade de utilizar o K-Means para segmentação de imagens.

Sendo uma das técnicas mais utilizadas no processamento de imagens em operações de transformação em formas geométricas, a morfologia matemática é bastante útil para extrair regiões de uma imagem, bem como descrever formas e objetos. De início essa técnica foi desenvolvida para trabalhar em imagens binárias, mas posteriormente a técnica evoluiu para lidar com imagens em escala de cinza. Neste projeto iremos focar em imagens binárias para a área de morfologia matemática.

As principais técnicas dentro da morfologia matemática em imagens são: Dilatação, Erosão, Abertura, e Fechamento. Essas operações morfológicas operam em conjuntos de pixels de uma imagem com base em sua forma e tamanho dentro da imagem.

Para entendermos as operações de morfologia, primeiro precisamos entender o que é um objeto estruturante. O objeto estruturante é geralmente uma matriz pequena, também conhecida como elemento estruturante ou kernel. Essa matriz tem uma forma específica, como um quadrado, círculo ou linha, e é usada para definir a operação de morfologia matemática. Esse objeto estruturante percorre a imagem pixel a pixel e dependendo da operação morfológica, expande ou contrai uma determinada região da imagem.

Dilatação: Ao realizar uma dilatação, o objeto estruturante percorre a imagem e se pelo menos um pixel do objeto estruturante se sobrepuser ao objeto na imagem, o objeto se expande ao tamanho dos pixels do objeto estruturante.

Erosão: O objeto estruturante ao percorrer a imagem, ele verifica se cabe totalmente dentro do objeto presente na imagem, se ele couber, o objeto é erodido para o pixel de origem do objeto estruturante, ou seja, supondo que um objeto quadrado caiba dentro de um objeto na imagem, o objeto na imagem terá o tamanho do pixel central desse objeto estruturante.

Abertura: É simplesmente uma erosão que é seguida por uma dilatação. Por exemplo, se você tiver imagens com pequenos pontos e objetos principais maiores. Quando a abertura é aplicada, a erosão removerá os pequenos pontos e a dilatação restaurará subsequentemente o tamanho do objeto maior, mas não reintroduzirá os pequenos pontos. O resultado é uma imagem mais limpa, na qual pequenos detalhes foram eliminados.

Fechamento: O fechamento é uma dilatação seguida por uma erosão. Supondo que tenhamos uma imagem binária onde o objeto principal tem um pequeno furo no centro. O objetivo é preencher esse buraco e tornar o objeto completo. Primeiro, a dilatação é usada para ampliar o objeto principal. Isso fará com que o espaço do buraco seja ocupado e as áreas próximas sejam conectadas. Após isso, a erosão é feita para restaurar o tamanho do objeto principal, mas agora com o buraco preenchido. A erosão remove os detalhes extras introduzidos pela dilatação.

Adicionalmente precisamos entender como é feito a detecção de objetos conexos. No que compreende a detecção de objetos, e identificar as áreas onde os pixels estão conectados entre si, no caso de imagens binárias, onde os pixels '1s' estão interligados. Por exemplo, levando em conta dois pixels, eles estarão conectados se compartilharem uma mesma aresta ou um vértice. Além disso, os objetos conectados podem ser partes da imagem que compartilham características em comuns, como áreas brancas em um fundo preto, ou vice-versa.

No contexto de processamento de imagens, o K-Means é utilizado para segmentação de imagens, sendo seu objetivo principal agrupar os pixels com características parecidas em clusters ou classes. Essas características normalmente são as cores ou a intensidade dos pixels.

De início, o pixel da imagem é definido como um ponto de dados no espaço. Levando em consideração uma imagem RGB, cada pixel é representado por um vetor, onde cada valor

representa o valor das cores no espaço.

Normalmente K centróides iniciais são escolhidos para iniciar o algoritmo de K-Means, mas para este projeto foi realizada a inicialização automática. Os pixels vão sendo atribuídos aos seus centros mais próximos à medida que o algoritmo é executado, essa proximidade é definida pela distância euclidiana que é calculada com base nas características dos pixels, por exemplo, diferenças nas componentes RGB. Os centróides vão sendo recalculados como a média das características dos pixels desses clusters. Todos esses passos são repetidos até que a convergência da média das características seja atingida. Por fim, teremos uma imagem segmentada em K classes definidas inicialmente, com camadas contendo pixels parecidos agrupados em um mesmo cluster(ou classe).

No âmbito da metodologia, os tópicos a serem abordados incluirão morfologia matemática em processamento de imagens, e segmentação de imagens baseada no algoritmo K-Means. A seção de resultados apresentará saídas gráficas de cada tópico tratado na metodologia, culminando, por fim, na conclusão do estudo.

II. Metodologia

A. *Morfologia Matemática em Processamento de Imagens*

Primeiramente é feito a leitura da imagem 'brain.png', convertendo a mesma para uma imagem em níveis de cinza, ou seja, monocromática. Com essa imagem em níveis de cinza formada, aplicamos um filtro Gaussiano no domínio da frequência com um Sigma no valor de '100', para limitar as altas frequências e borrar minimamente a imagem. Após isso, é aplicado um filtro de mediana que irá recalcular os valores dos pixels de acordo com seus vizinhos, removendo uma quantidade significativa de ruído. Foi utilizada uma máscara de [7, 7] como argumento de uma função padrão do MatLab, a 'medfilt2()'.

Finalmente temos a imagem 'brain' suavizada e pronta para ser analisada e realizar a binarização.

Como a imagem possui uma borda branca que aparentemente tem uma área maior que a área do tumor, é preciso remover essa borda antes de realizar a detecção. É feita a análise do histograma da imagem 'brain' e é constatado que a maioria dos pixels de bordas estão com valores de brilho próximos de 255, que define o branco absoluto. Com isso é feito uma binarização com o limiar no valor de '250', ou seja, todos os pixels acima de 250 serão definidos como 1, e o restante será 0. Novamente analisando o histograma, temos que os pixels do tumor estão se diferenciando do restante da imagem a partir dos valores de brilho de 150 aproximadamente. Como isso, o limiar é definido como '151', para definir os pixels do tumor com 1. Infelizmente os pixels de bordas estarão incluídos nessa binarização, mas é exatamente por esse motivo que binarizamos a borda separadamente. Agora basta subtrair a borda da imagem 'brain', e teremos a imagem 'brain_bin' final, binarizada e sem a maioria da borda.

Com a imagem binarizada, podemos aplicar as operações morfológicas para melhorar esse resultado. Primeiro foi criado um objeto estruturante 'ponta' manualmente, com um formato de ponta. O objetivo é criar uma ponta nos contornos da seleção do tumor, rumo a melhorar os detalhes. Além disso, temos outros elementos estruturantes, como o 'square0' com tamanho 3, e o 'disk0' com tamanho 3.

Primeiramente aplicamos uma abertura com o 'disk0', que remove praticamente todos os resquícios da borda branca. Depois disso é aplicado mais uma abertura, agora com 'meu_elemento' estruturante, que começa a inserir pequenas pontas nos objetos. Depois é aplicado uma dilatação com o 'meu_elemento' com o objetivo de aumentar os contornos com a ponta. E por fim uma erosão com o elemento estruturante 'square0' melhorando os resultados dos contornos em ponta.

Agora, com o máximo de operações aplicadas sem afetar o tumor, teremos que detectar o tumor dentre os objetos presentes na imagem. Para isso utilizamos a função padrão do MatLab que ira retornar características dos objetos presentes na imagem 'CC'. Com esse retorno obtido, conseguimos ter o número total de objetos presentes na imagem.

A partir da quantidade total de objetos totais presentes na imagem, criamos um vetor para armazenar a quantidade de pixels de cada objeto. Os tamanhos são obtidos dentro de um 'for' onde calculamos o 'size(CC.PixelIdxListi, 1)' e armazenamos em 'sizes_objs'. Após isso basta encontrar o maior tamanho presente no vetor 'sizes_objs' e teremos o objeto de maior área da imagem.

Depois disso, utilizamos a função padrão 'find' do MatLab para encontrar o índice 'maior_objeto_id' desse objeto dentro do vetor 'sizes_objs'. E de posse do índice podemos obter os pixels desse objeto, 'CC.PixelIdxListmaior_objeto_id'

É criado uma imagem toda preta de uma única dimensão para armazenar o objeto encontrado. Com a imagem apenas com o objeto, é feita uma dilatação e uma erosão separadamente. Com isso subtraímos a dilatação 'a' da erosão 'b', obtendo as fronteiras do objeto.

Agora é criado uma imagem toda preta RGB para armazenar as fronteiras do objeto na cor vermelha dentro dessa imagem.

Por fim, somamos a imagem original 'img_gray' a 'borda_red', destacando na cor vermelha o tumor do cérebro de Brain.

B. *Segmentação de Imagens : K-Means*

Primeiro lemos a imagem 'onion.png' armazenada em 'onion'. Obtemos as dimensões da imagem, sendo sua altura definida como 'l' (linhas), e a largura definida como 'c' (colunas), e por fim pegamos o número de canais guardando em 'camada'.

Após isso realizamos o 'reshape' para transformar o imagem 'onion' RGB para um vetor de dados. Com o vetor de dados em mãos, chamamos a função 'kmeans' várias vezes com um valores de k arbitrários, com o intuito de chegar a melhor segmentação da pimenta. Conseguindo obter um valor final igual a 8 classes. Agora chamamos 'kmeans' novamente com o valor ideal, só que agora retornamos o vetor de centróides, para obter com a função 'centroide_cor_mais_proximo' o

‘id_centroide’ da pimenta. Para isso passamos o vetor de centróides ‘C’ e a cor aproximada da pimenta.

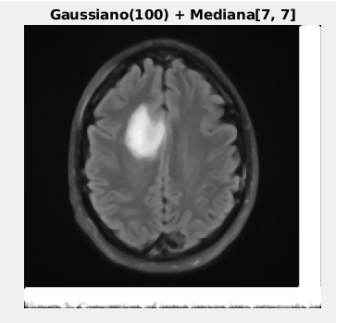
Realizamos o mesmo processo para a cebola, só que agora como valor de classes ideal igual a 6.

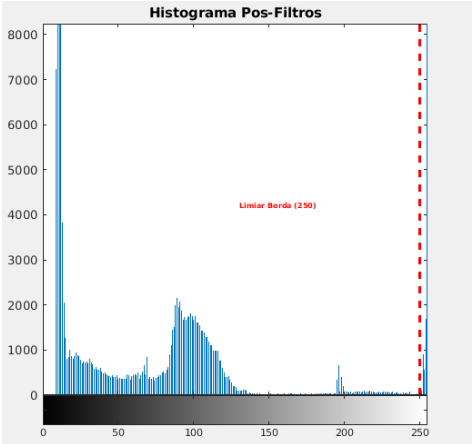
Como o objetivo de segmentar todas as verduras de uma vez, foi utilizado o Método do Cotovelo, que visa analisar o ponto no gráfico que define a relação da inércia e da quantidade de classes definidas para o ‘kmeans’. Quando a inércia para de sofrer quedas significativas no decorrer do aumento do número de classes, quer dizer que esse é o ponto perfeito para melhor segmentar a imagem, ou seja, o valor ideal de K. Cada classe tem sua distância quadrática intra-classe, que é a soma das distâncias quadráticas entre cada pixel dentro de uma classe. A inércia, em termos de k-means, refere-se à soma das distâncias quadráticas intra-cluster. Portanto, a soma das distâncias quadráticas intra-classe é uma medida da inércia para cada cluster individual. A inércia total para todos os clusters é a soma das inércias individuais.

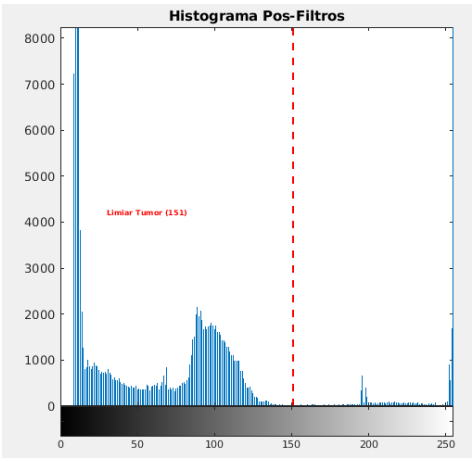
Com o número de classes K ideal obtido, podemos segmentar todas as verduras de uma vez. Executando o ‘kmeans’ para ‘K == 9’ classes.

III. Resultados

A. Resultados: ***Morfologia Matemática em Processamento de Imagens***







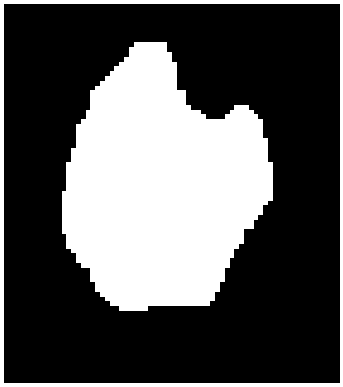


Figura 1. Abertura ”disk”tam : 3

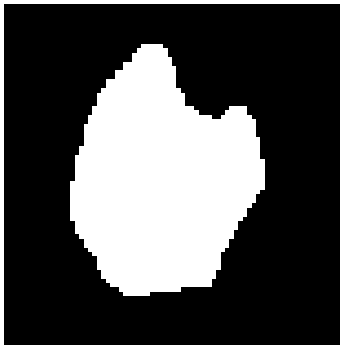


Figura 2. Abertura meu elemento

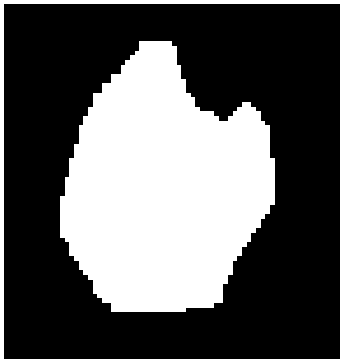
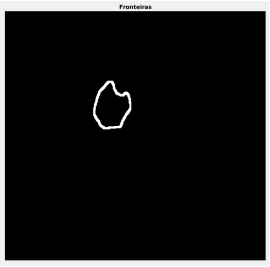


Figura 3. Dilatação meu elemento

Figura 4. Erosão square0 : FINAL



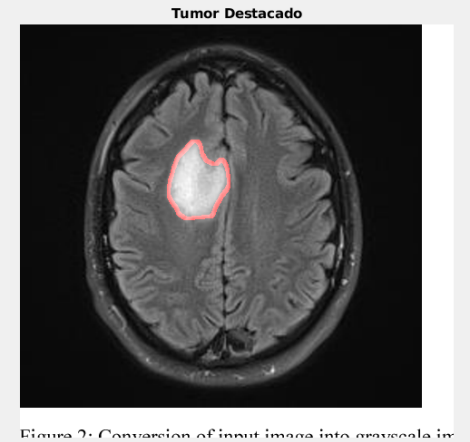
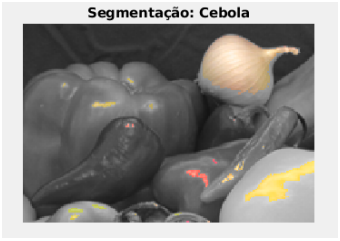


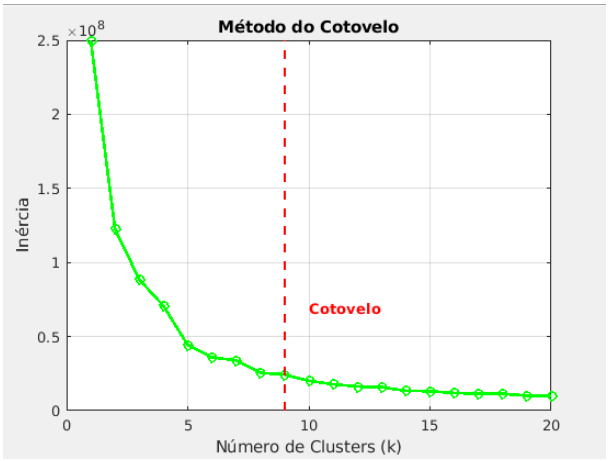
Figure 2: Conversion of input image into grayscale im

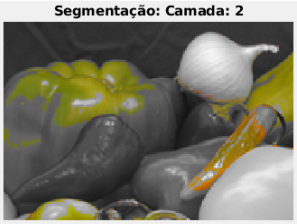
As mudanças foram mínimas, porém é perceptível que as bordas foram detalhadas. Os resultados não incluem a remoção da borda branca da imagem original, pois iria exceder o limite de páginas. Porém ao executar o código terão todos os passos.

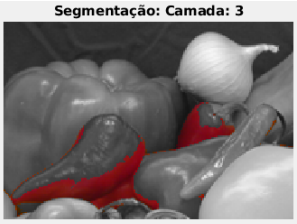
B. Resultados: ***Segmentação de Imagens : K-Means***

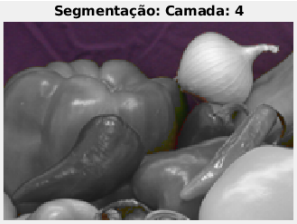


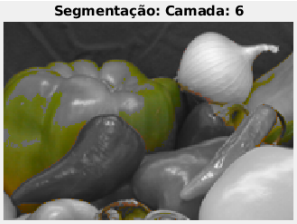
1) Segmentação do Restante das Verduras: O restante das verduras foram segmentadadass em 9 clusters, porém será exibido abaixo somente os melhores resultados. As pimentas e a Cebola ja foram exibidas acima, portanto não exibi novamente.

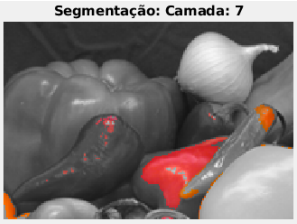












Conclusões

Após analisar os resultados, percebi que as técnicas de morfologia matemática são muito úteis em processamento de imagens, podendo ajudar na detecção de objetos de maneira eficiente e otimizada. O maior desafio foi pensar em uma maneira de remover a bordas sem afetar o tumor. O método de subtração da borda foi a melhor maneira que encontrei. O algoritmo K-means é uma estratégia interessante para segmentar imagens, sendo útil para agrupar os pixels das imagens por um característica de cor ou brilho, facilitando o processo de segmentação. Houve alguns problemas para segmentar perfeitamente, mas no geral a eficiência do K-Means é satisfatória.