Processamento Digital de Imagens
Trabalho 2

Mariana Mendanha da Cruz
Engenharia de Controle e Automação
Faculdade de Tecnologia - UNB
Matrícula: 160136784
Email: mariana.mendanha.mm@gmail.com

***Abstract*—Trabalho de Processamento Digital de imagens utilizando Octave.**

I. Introdução

O objetivo deste trabalho é abordar o tema Processamento Digital de imagens, mais especificamente a binarização, morfologia matemática e a segmentação contando com a ferramenta Matlab/Octave e também por em prática os conhecimentos passados em aula na disciplina de Introdução a Processamento de imagens.

MATLAB é um software interativo de alta performance voltado para o cálculo numérico. Ele integra análise numérica, cálculo com matrizes, processamento de sinais e construção de gráficos em ambiente simples. Utilizando-se do pacote Image Processing Toolbox do Matlab/Octave é possível ler imagens como matrizes e com isso aplicar diversas operações, manipulando a imagem.

Com relação ao tema, é importante abordar alguns conceitos, sendo eles: Binarização, Morfologia e Transformada de Watershed.

Para separar os objetos que desejamos analisar de uma imagem, utilizamos técnicas de binarização ou limiarização. A binarização é o método mais simples de segmentação de imagens. Resumidamente consiste em separar uma imagem em regiões de interesse e não interesse através da escolha de um ponto de corte. Os métodos mais simples de binarização utilizam um único ponto de corte, conhecido por threshold (limiar). Em alguns casos, no entanto, não se consegue apenas um limiar que resulte em uma boa segmentação para toda a imagem. Para esses casos existem técnicas de limiarização variáveis e multiníveis baseadas em várias medidas estatísticas. Como sabemos que uma imagem digitalizada pode ser escrita como uma função *f(x,y)* (histograma), a função binarização pode ser escrita como:

$$g(x,y) = \left\{ \begin{array}{l} R_1 sef(x,y) \leq T \\ R_2 sef(x,y) > T \end{array} \right\} \quad (1)$$

Onde T é o limiar, R_1 e R_2 são os valores estipulados para os níveis de cinza da imagem binarizada, neste caso utiliza-se 0 (preto) e 255 (branco).

A Morfologia Matemática se refere ao estudo e análise de imagens usando operadores não lineares. Ela consiste em extrair informações de uma imagem de geometria desconhecida pela transformação com uma outra imagem completamente definida, chamada elemento estruturante.

Ao contrário da convolução genérica, na Morfologia Matemática o formato do elemento estruturante terá impacto sobre o resultado. Dentre as atividades que a morfologia executa no processamento de imagens estão: aumento de qualidade, restauração, detecção de falhas, análise da textura, análise particular, análise de formas, compreensão, etc. Essas ferramentas são aplicadas em diversas áreas como visão robótica, inspeções, microscopia, biologia, metalurgia e documentos digitais.

A ideia básica é percorrer uma imagem com um elemento estruturante e quantificar a maneira com que este se encaixa ou não na imagem. No caso afirmativo, marca-se o local ou nível de cinza onde o elemento estruturante coube na imagem. Dessa forma, pode-se extrair informações relevantes sobre o tamanho e forma de estruturas na imagem. As operações morfológicas que utilizaremos serão: erosão, dilatação, abertura e fechamento.

A *Transformada de Watershed* propõe uma abordagem morfológica para o problema de segmentação de imagens, interpretando estas como superfícies, onde cada pixel corresponde a uma posição e os níveis de cinza determinam as altitudes.

A partir desta noção, identificam-se simbolicamente ”bacias hidrográficas”, definidas por mínimos regionais e suas regiões de domínio. E nos pontos onde águas provenientes de mínimos diferentes se tocam, ergue-se uma barreira, que constitui a linha de segmentação de uma seção da imagem.

II. Metodologia

Nessa seção será abordada a resolução dos Problemas:

A. Problema 1.1

Foi pedido a criação de um programa que dada a imagem *morf_test.png* binarize-a diretamente, em que seu fundo fique

branco e as letras pretas.

Então, recebeu-se a matriz da imagem transformado-a em precisão dupla, calculou-se o limiar por meio da função já implementada *"graythresh"* e fazendo uso do limiar, utilizou-se a função *"im2bw"* para exercer a binarização direta da imagem.

B. Problema 1.2

Foi pedido a criação de um algoritmo Morfológico em níveis de cinza de forma a eliminar (ou reduzir) as variações no fundo da imagem e em seguida binariza-la.

Então, criou-se um algoritmo que primeiramente faz um realce das letras por meio da erosão (já que as letras são mais escuras que o fundo), removendo assim, possíveis falhas da escrita contida na imagem. Em seguida, optou-se pelo fechamento para a obtenção do fundo da imagem, com o fundo, operou-se uma subtração com a imagem resultante da erosão obtendo-se assim uma imagem com o fundo preto. Por fim, com essa mesma imagem, calculou-se o valor de seu limiar e então aplicou-se a binarização, chegando-se a imagem final.

C. Problema 2.1

Foi pedido para se fazer a leitura da imagem *"cookies.tif"*.

Então, recebeu-se a matriz da imagem transformado-a em precisão dupla utilizando as funções *"im2double"* e *"imread"*.

D. Problema 2.2

Foi pedido para binarizar a imagem de tal forma que sejam identificados os dois cookies.

Então, calculou-se o limiar por meio da função já implementada *"graythresh"* e fazendo uso do limiar, utilizou-se a função *"im2bw"* para exercer a binarização direta da imagem.

E. Problema 2.3

Pediu-se para eliminar por completo o cookie mordido, deixando pelo menos uma parte do cookie completo na imagem binarizada utilizando algoritmos morfológicos.

Para isso, utilizou-se da erosão com elemento estruturante sendo um disco de raio 60 escolhido através de testes manuais.

F. Problema 2.4

Pediu-se para recuperar a forma inicial do cookie completo na imagem resultante do problema anterior.

Desta forma, utilizou-se a dilatação para reverter o processo anterior com o elemento estruturante sendo um disco de raio 60.

G. Problema 2.5

Pediu-se para, a partir da imagem resultante do problema anterior, obter uma imagem final em níveis de cinza com apenas o cookie completo.

Então, fez-se uma subtração da imagem dilatada do problema anterior com a imagem original, obtendo-se assim, a imagem final.

H. Problema 3.1

Pediu-se para binarizar a imagem *img_cells.jpg*, onde as células ficam pretas e o fundo branco.

Então, recebeu-se a matriz da imagem transformado-a em precisão dupla utilizando as funções *"im2double"* e *"imread"*. Em seguida, calculou-se o limiar por meio da função já implementada *"graythresh"* e fazendo uso do limiar, utilizou-se a função *"im2bw"* para exercer a binarização direta da imagem.

I. Problema 3.2

Pediu-se para retirar os buracos nas células da imagem, onde as células ficam pretas e o fundo branco.

Primeiramente, inseriu-se bordas na parte superior, lateral direita e inferior da imagem binarizada, percorrendo bit a bit por meio de laços preenchendo esses bits com valor 0 (preto). Em seguida utilizou-se a função pronta *"imfill"* na imagem binária invertida para preencher os buracos da imagem. Por último, retirou-se as bordas criadas anteriormente por meio de um algoritmo em laço que verifica se o bit próximo - por exemplo, para a borda superior o bit verificado é o bit logo abaixo - tem valor um ou zero e reproduz esse mesmo valor para o bit em questão, numa tentativa de consertar a informação que foi perdida com a inclusão dessa "borda de tratamento".

J. Problema 3.3

Pediu-se para, se necessário, utilizar a função *"bwareaopen"* para se preencher espaços conectados.

Nesse passo, utilizou-se a função enunciada pois havia um ou dois pontos na imagem com pequenos aglomerados de bits que não eram verdadeiramente células, tal função retirou

esses pontos da imagem. Para ela, utilizou-se o inverso da imagem obtida no item anterior e seu resultado foi invertido novamente para se ter a imagem como especificada (fundo branco, células pretas), obtendo-se assim, a imagem binária final.

K. Problema 3.4

Pediu-se então, para computar a segmentação *Watershed* utilizando como parâmetro o resultado da função de distância *"bwdist"*.

Primeiramente, calculou-se a transformada da distância por meio de *"bwdist"* em que o valor de cada pixel na imagem de saída é a distância entre este mesmo pixel e o pixel 1 mais próximo na imagem. Em seguida pega-se o complemento dessa imagem resultante para que os pixels claros representem às altas elevações e os pixels escuros as baixas para a Transformada de Watershed. Então, calculou-se a Transformada por meio da função *"watershed"* obtendo-se uma imagem com repartições e setou-se em zero os pixels que estão fora da zona de interesse. Por último, transformou-se a imagem obtida em RGB para melhor visualização da segmentação.

III. Resultados

A. Problema 1

Visto anteriormente como foram resolvidos os problemas apresentados, vamos aos resultados:

Iniciou-se com a imagem original:

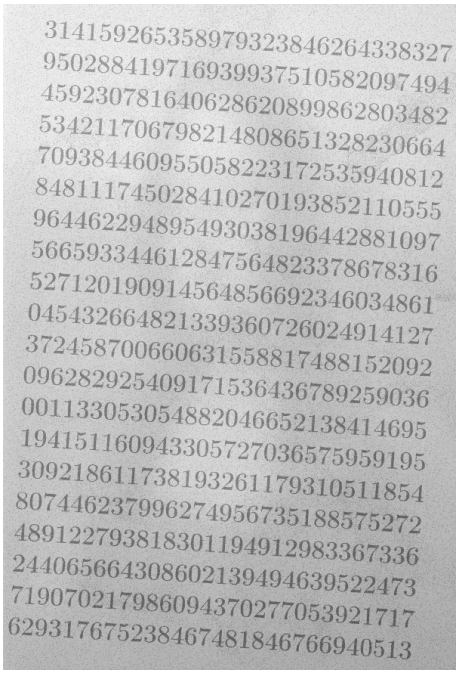


Fig. 1. Imagem original

Binarizando-a diretamente temos:

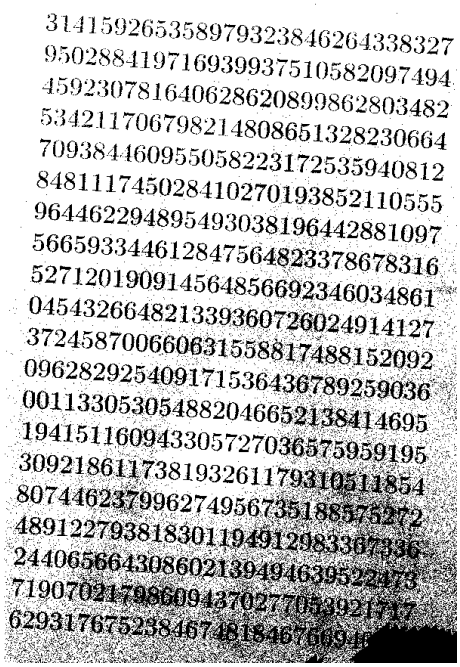


Fig. 2. Imagem binarizada diretamente

Como pode ser visto, apesar de binarizada com o melhor valor encontrado, a imagem ainda deixa a desejar, pois possuí uma quantidade razoável de ruído, o que atrapalha na leitura do texto.

Utilizando o algoritmo morfológico temos:

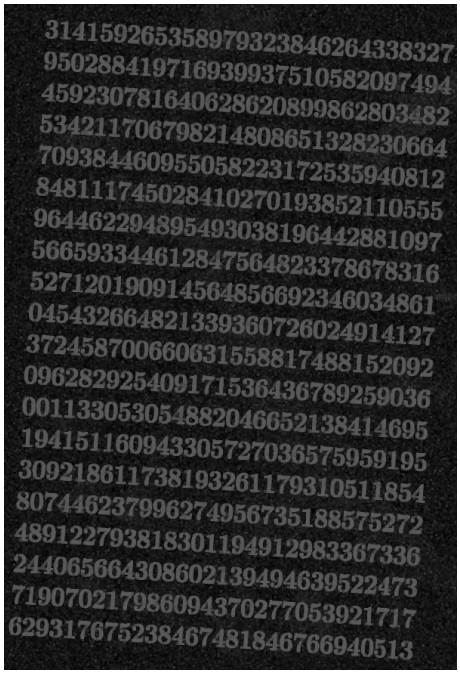


Fig. 3. Imagem após algoritmo morfológico

É possível perceber que o fundo está escuro por conta da subtração de fundo realizada e também parece estar mais homogêneo.

Por fim, após a binarização da imagem anterior, temos:

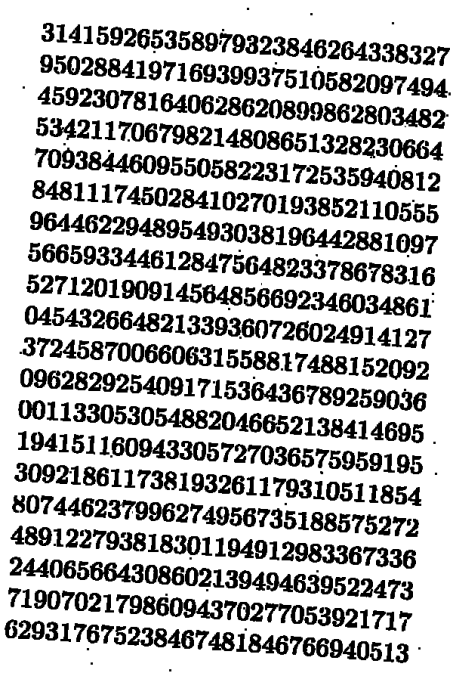


Fig. 4. Imagem binarizada

Comparando com a imagem diretamente binarizada conseguimos perceber uma melhora visível da legibilidade da imagem por conta do fundo que foi retirado quase que completamente. Pode-se ver que no algoritmo implementado que há um trecho de código comentado com título *Retirada de ruídos*, descomentando este trecho obtém-se uma imagem ainda mais livre de ruídos. De qualquer forma é notável a melhora da imagem com algoritmo morfológico.

B. Problema 2

Visto anteriormente as resoluções dos problemas apresentados, vamos aos resultados: Iniciou-se com a imagem original:

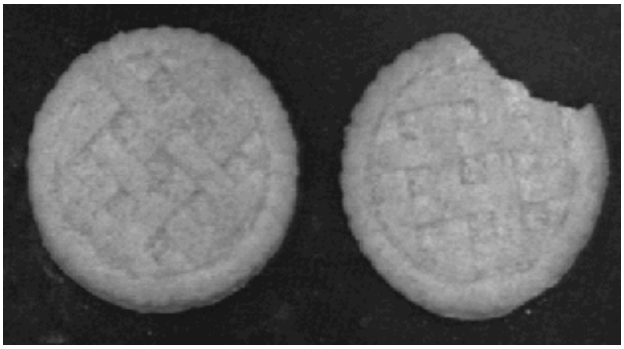


Fig. 5. Imagem original

Binarizando a imagem temos:

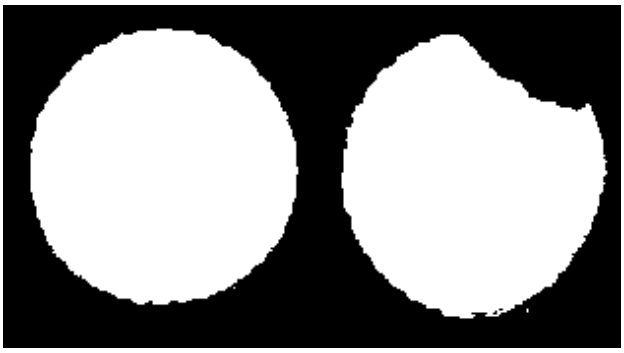


Fig. 6. Imagem binarizada

Vemos os formatos dos cookies bem definidos em branco, com fundo preto.

Em seguida, a imagem binarizada sofre erosão, resultando:

Fig. 7. Imagem erodida

Nota-se que o cookie mordido sumiu completamente e sobrou apenas uma pequena parte do cookie inteiro.

Recuperou-se o cookie inteiro por dilatação, resultando:

Fig. 8. Imagem dilatada

Podemos perceber que pelo cookie mordido ter sumido totalmente na imagem obtida no passo anterior, quando fazemos a operação inversa apenas o cookie inteiro volta ao

seu tamanho inicial.

Por fim, utilizando a imagem anterior como máscara para a imagem original, obtêm-se a imagem final:

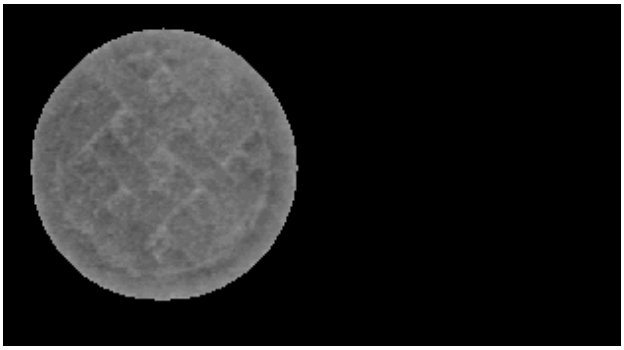


Fig. 9. Imagem final

Vemos que a imagem resultante obtida tem possui apenas o cookie inteiro em níveis de cinza e o fundo da imagem é preto.

C. Problema 3

Visto anteriormente como foram resolvidos os problemas apresentados, vamos aos resultados:

Iniciou-se com a imagem original:

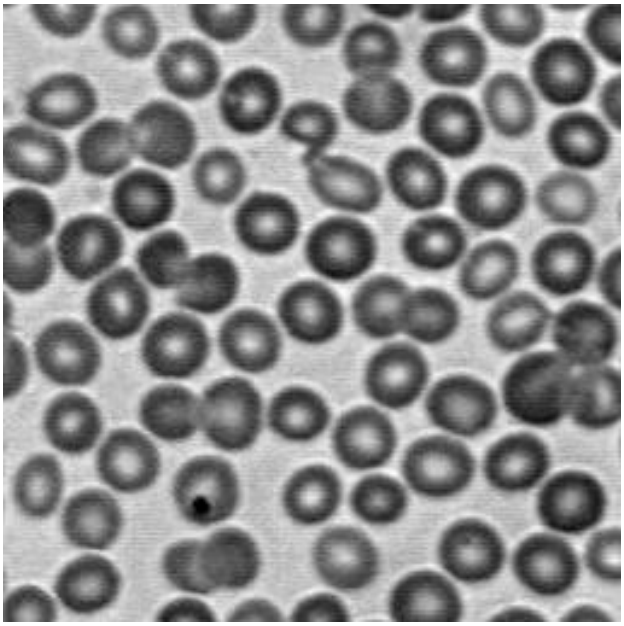


Fig. 10. Imagem original

Binarizando-a diretamente temos:

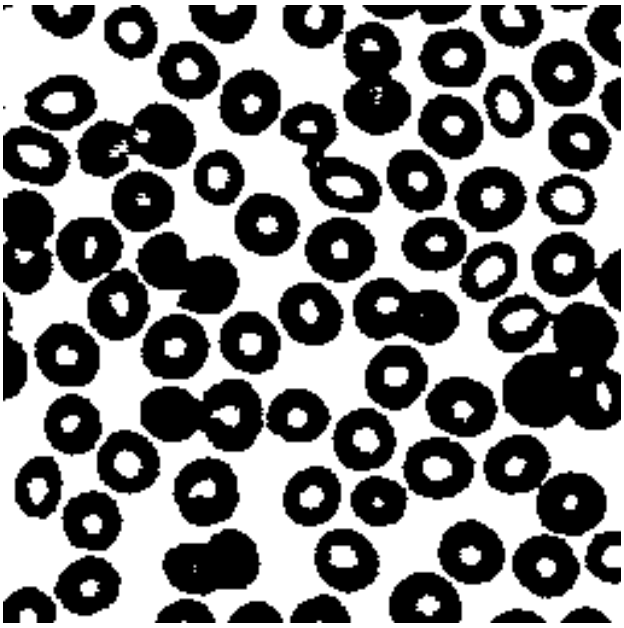


Fig. 11. Imagem binarizada diretamente

A imagem foi binarizada e as células ficaram pretas e o fundo branco assim como foi pedido.

Fez-se, então, o preenchimento dos buracos:

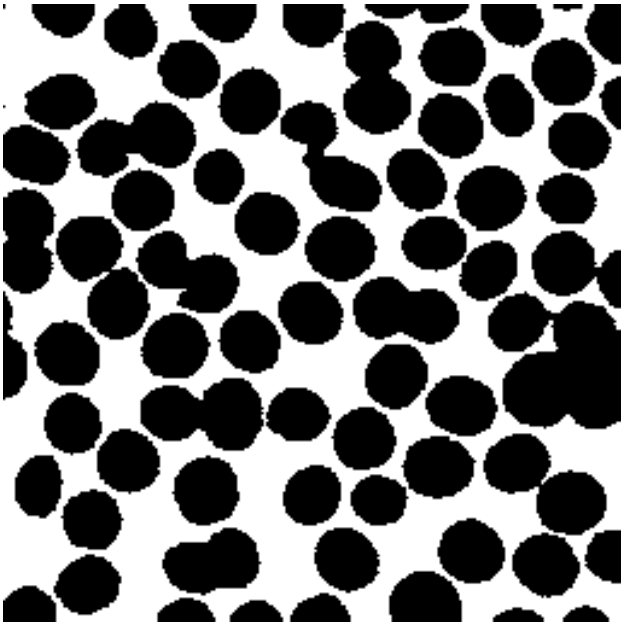


Fig. 12. Imagem preenchida

Como pode ser visto, todos os buracos foram preenchidos, isso por conta do algoritmo de tratamento de borda criado que pode ser visto na seção anterior.

Em seguida usou-se *bwareaopen* para a retirada de pontos que não representam células:

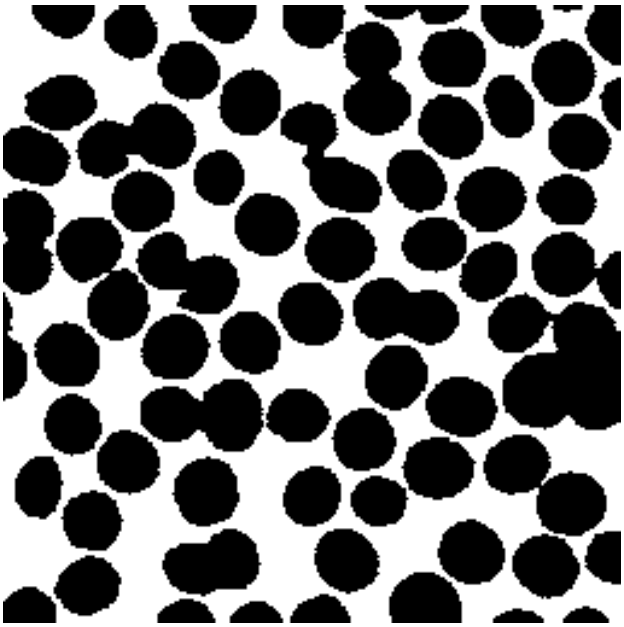


Fig. 13. Imagem binária final

Podemos ver que alguns bits localizados no canto superior esquerdo não se encontram mais lá, pois não foi considerado que eles representariam células.

Com o cálculo da segmentação obtemos:

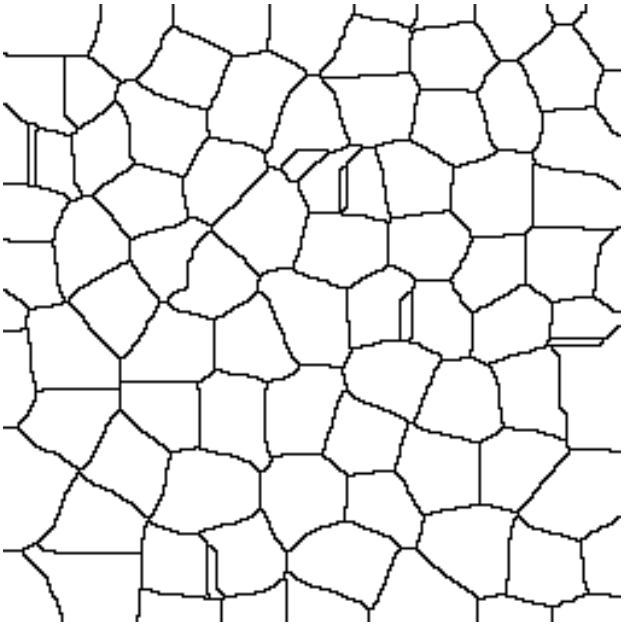


Fig. 14. Imagem da segmentação

Com essa imagem é possível visualizar as repartições criadas mesmo sem a presença das células

A segmentação vista anteriormente foi empregada na imagem final binarizada, resultando:

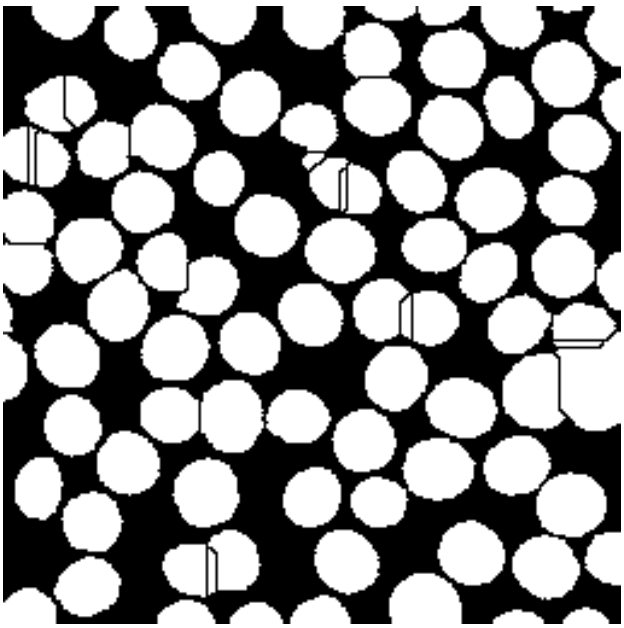


Fig. 15. Imagem células segmentadas

Vemos que a imagem obtida nada mais é que a própria imagem segmentada, em que cada célula foi dividida

Aqui a imagem foi transformada em rgb:

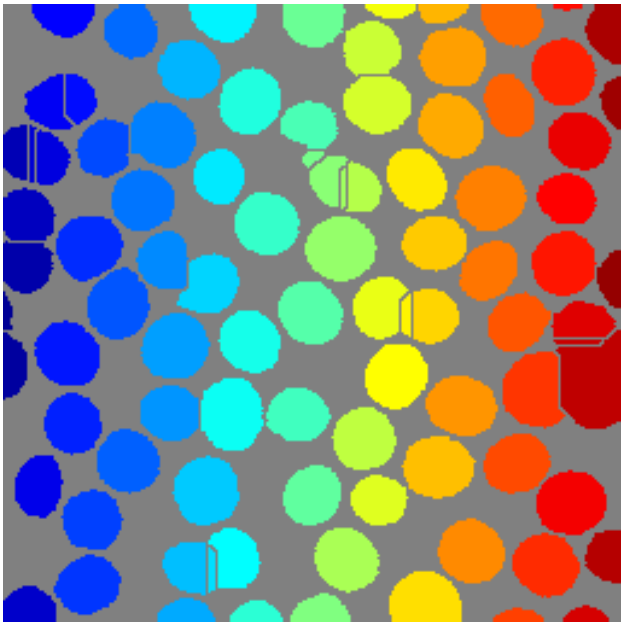
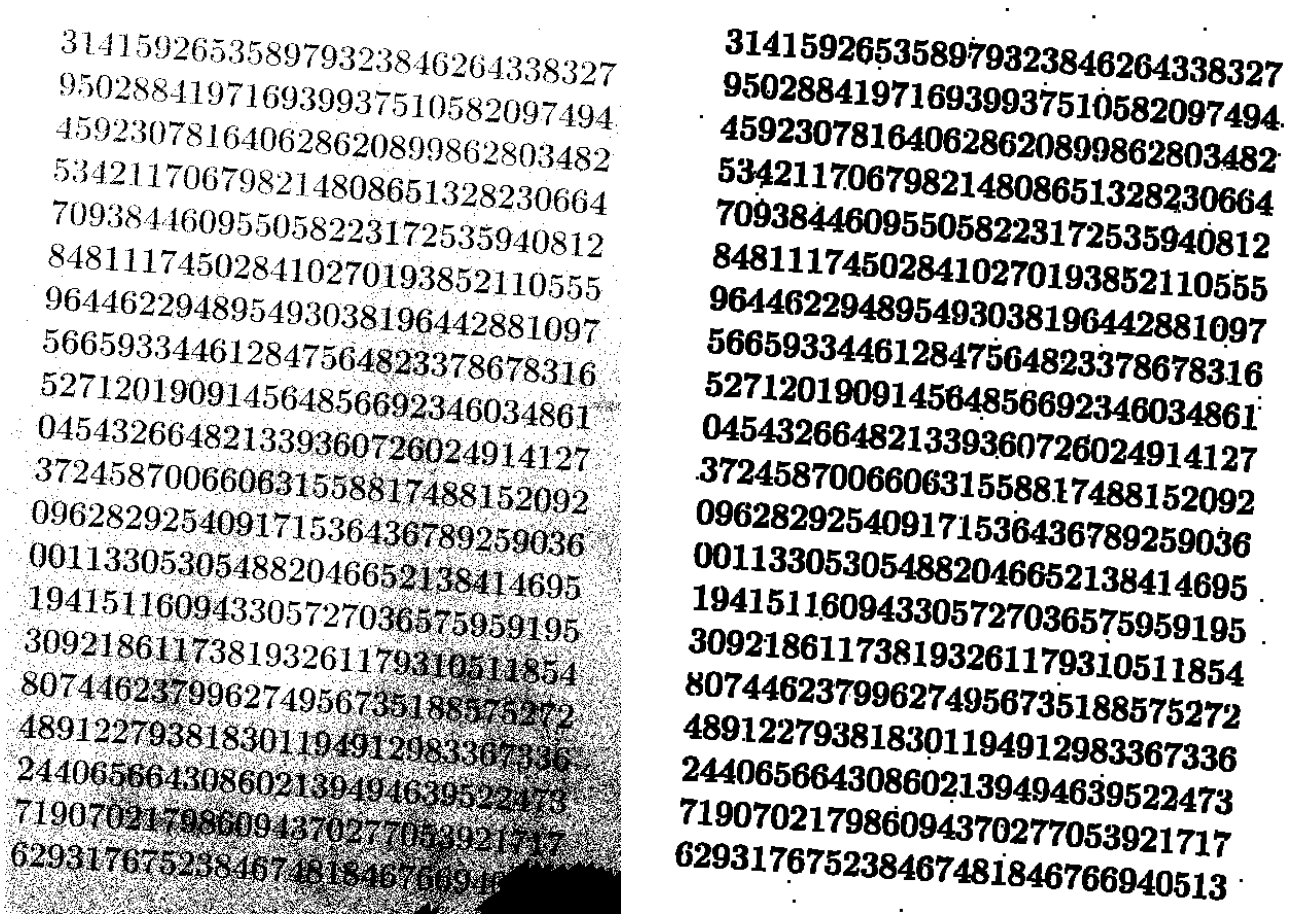


Fig. 16. Imagem células segmentadas colorida

Para melhor visualização das segmentações separou-se as partes com cores onde fica mais nítido ainda a separação.

IV. Conclusão

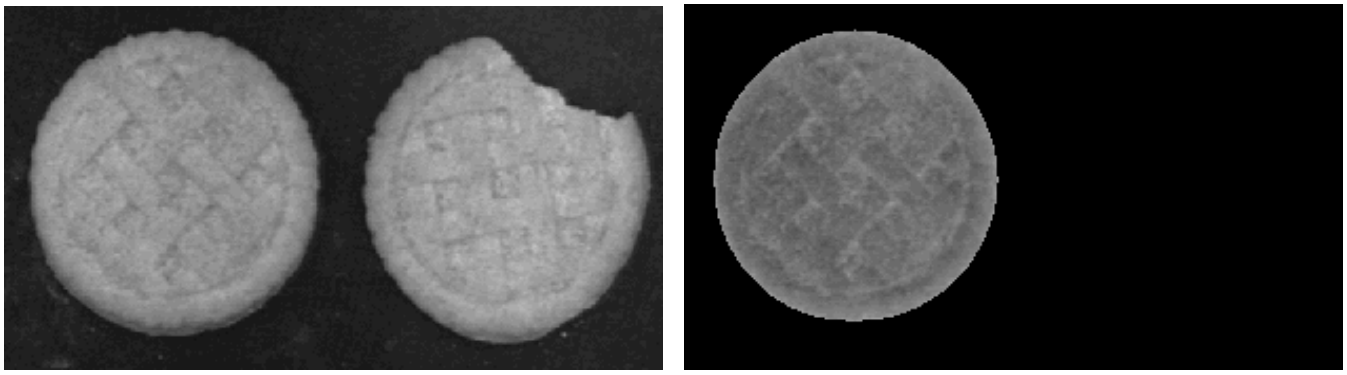
Podemos concluir no problema 1 que houve uma grande melhora na nitidez e legibilidade do texto, o que pode ser visto comparando as imagens a seguir, portanto a resolução com algoritmo morfológico foi um sucesso.



(a) Binária direta (b) Binária com morfologia

Fig. 17. Comparação da imagem binária direta com imagem obtida

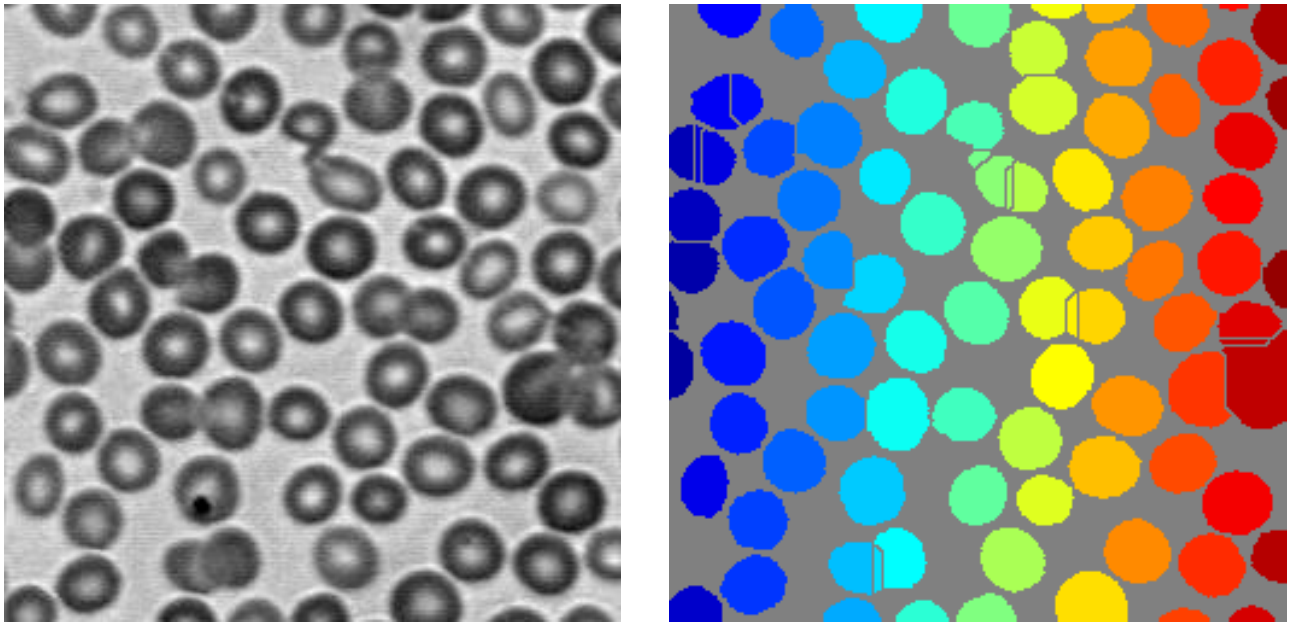
Podemos concluir no problema 2 que o objetivo de obter uma imagem final em níveis de cinza apenas com o cookie inteiro foi alcançado, portanto a resolução do problema foi um sucesso, como pode ser visto abaixo:



(a) Imagem original (b) Imagem final

Fig. 18. Comparação da imagem original com imagem obtida

Podemos concluir no problema 3 que houve uma segmentação real da imagem e esta bem definida. É possível observar também que houve áreas em que a segmentação das células não foi a mais correta, pela imagem não se tratar de corpos circulares perfeitos, felizmente não foram muitos casos desse tipo. Por fim, foram feitos alguns testes por conta própria para a melhora da segmentação, porém os resultados sempre seguiam de melhoras em alguns pontos e pioras em outros, por conta disso, decidiu-se deixar como está, até porque a melhora do *Watershed* não era o objetivo do problema em si, portanto, podemos dizer que a resolução do problema que consistia na aplicação do *Watershed* foi um sucesso, como pode ser visto no resultado abaixo:



(a) Imagem original (b) Imagem final Segmentada

Fig. 19. Comparação da imagem original com imagem obtida

Por último, podemos dizer que o objetivo do trabalho foi alcançado, os temas propostos foram trabalhados, praticou-se os conceitos e conhecimentos abordados em aula e encontrou-se todos os resultados requisitados.

V. Referências

Roteiro do trabalho.
Slides de Introdução a Processamento de imagens.
Livro texto - Digital Image Processing.
http://www.lapix.ufsc.br/ensino/visao/morfologia-matematica/
https://infoescola.com.br
https://tecnoblog.net
https://stackoverflow.com
http://www.ime.unicamp.br/~valle/PDFfiles/valente10.pdf
https://octave.sourceforge.io
https://mathworks.com
http://www.ic.uff.br/~aconci/OTSUeOutras.pdf
https://teses.usp.br/teses/disponiveis/18/18133/tde-10072006-002119/publico/Capitulo_4_mestrado.pdf