Métodos de Análise de Imagem para Contagem Automática de Células

Duarte Rodrigues 1,2, Ricardo Brioso 1,3, Mariana Xavier 1,4

1 Faculdade de Engenharia da Universidade do Porto

2 [up201708998@fe.up.pt](mailto:up201708998@fe.up.pt)

3 [up201705429@fe.up.pt](mailto:up201705429@fe.up.pt)

4 up201705177@fe.up.pt

*Resumo* — oi

Palavras-chave — contagem celular, análise de imagem, região de interesse (ROI), segmentação, outros

# Introdução

O número de células numa cultura é uma medida de controlo altamente usada para obter informações acerca da densidade celular nas mesmas, usado como medida de avaliação em casos de proliferação, viabilidade ou toxicidade [1].

Geralmente este processo é feito de forma manual, o que, apesar de simples, implica subjetividade e concentração humana, que se vai refletir na fiabilidade dos resultados, e é um processo que demora bastante tempo [2]. Assim, de forma a melhorar a exatidão e a velocidade do processo, têm vindo a ser desenvolvidos métodos automáticos baseados em análise de imagem e visão computacional.

Este projeto tem como objetivo o desenvolvimento de métodos de análise de imagem que permitam fazer a contagem automática das células cancerígenas HL60, relacionadas com a leucemia, usando imagens obtidas por microscopia e por um hemacitómetro [3] (dispositivo cuja área de contagem é conhecida, delimitada por linhas de um determinado tamanho que ajudam na contagem, permitindo determinar a densidade celular em relação a um volume específico de solução [4]). Para tal, torna-se necessário em primeiro lugar delinear apenas a área correspondente à área de interesse e só posteriormente proceder à contagem celular no seu interior.

Completar if u want

# Métodos

## Delineamento da Região de Interesse (ROI)

Nesta primeira etapa começou-se por abrir o diretório das imagens provenientes da base de dados utilizada. Estas foram, então, corridas e processadas uma a uma. Olhando para as imagens, observa-se que estas contêm uma grelha de linhas individuais, sendo a região de interesse definida pelo quadrado entre quatro conjuntos de três linhas adjacentes. Assim, tentou-se isolar essas linhas e preencher todo o seu interior, obtendo-se a máscara binária desejada.

Após ter a imagem a processar lida, esta foi passada à função *segmentROI* que, tal como o nome indica, extraiu dela a região de interesse, representando-a numa imagem binária como um retângulo branco em fundo preto.

Posto isto, para avaliar a exatidão do processo, recorreu-se à função *evaluateROI*. Esta recebe como parâmetros a ROI obtida e o índice da imagem no diretório, de forma a ler a máscara que corresponde ao seu *ground truth*, segmentada manualmente, e determinar os três valores que vão permitir a sua avaliação: o índice de Jaccard, a média e a máxima das quatros distâncias Euclidianas entre os vértices das regiões (em pixéis).

Juntamente com isto, as imagens obtidas foram gravadas numa nova pasta, juntamente com um *report* geral em ficheiro texto com o nome de todas as imagens e os respetivos índices obtidos.

### Segmentação da ROI

Na função *segmentROI*, começou-se por fazer um pré-processamento à imagem. O primeiro passo foi transformá-la numa imagem em escala de cinza, em vez de RGB, de forma a poder efetuar todas as operações seguintes. De seguida, aplicou-se um filtro de mediana, capaz de remover ruído pontual da imagem, tanto claro como escuro. Tendo a imagem suavizada, esta foi binarizada, ou seja, imagem apenas em tons preto e branco, recorrendo ao método de *Otsu*, de forma a se obter apenas as linhas microscópicas e alguns contornos celulares mais evidentes.

Após o pré-processamento, recorreu-se à operação *close* com dois elementos estruturantes diferentes: um em forma de linha horizontal e outro de linha vertical. Esta operação é usada para fundir os conjuntos de três linhas adjacentes, preenchendo os espaços entre elas, obtendo-se uma imagem constituída por uma grelha de linhas individuais, em que quatro delas são bastante mais largas que as outras. Assim, passamos de ter de identificar quatro conjuntos de linhas para identificar apenas 4 linhas.

De seguida, o objetivo era eliminar as linhas mais finas, que não delimitam a ROI. Para tal, recorreu-se a um elemento estruturante na forma de disco, aplicado na função *open*, que elimina as linhas desejadas, sem erodir as linhas mais grossas.

Depois, recorreu-se à função de Matlab *imfill* para preencher o espaço delimitado pelas 4 linhas, de forma a se obter a máscara branca. No entanto, agora tornou-se necessário cortar as linhas para lá dos seus cruzamentos, de forma a obter apenas o quadrado desejado.

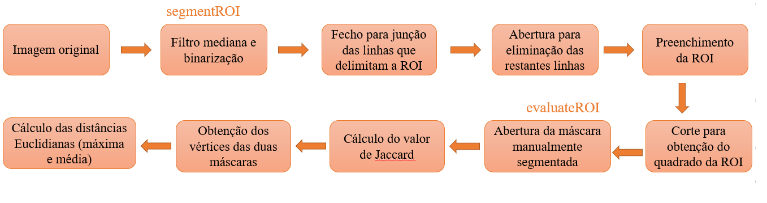
Para este corte, recorreu-se à função de Matlab *bwboundaries* que dá o conjunto das coordenadas dos pixels que delimitam um objeto. Com dois simples ciclos conseguiu-se determinar, a partir das coordenadas obtidas, tanto as linhas superior e inferior como as linhas laterais esquerda e direita que delimitam a área de interesse, procedendo ao corte de tudo o que estava para lá delas, ou seja, definindo-lhes o valor de intensidade igual a zero.

### Avaliação da Segmentação Obtida

Esta função abre, em primeiro lugar, o diretório contendo as máscaras segmentadas manualmente, lendo aquela que corresponde ao índice que lhe é passado como parâmetro. Esta máscara vai servir como termo de comparação e avaliação da exatidão da máscara obtida pelo algoritmo, ou seja, funciona como *ground truth*.

Seguidamente é feito o cálculo da similaridade, dada pelo valor de Jaccard, aplicando diretamente a função de Matlab *jaccard* com as duas imagens binárias como parâmetros.

A obtenção das distâncias Euclidianas já implica mais cálculo. Recorreu-se novamente à função *bwboundaries* para determinar os pontos na fronteira da máscara de forma a achar os quatro cantos que a caracterizam, sendo estes incluídos num vetor de vértices. Isto foi feito tanto para a máscara obtida como para o *ground truth*. Por fim, com um simples ciclo calculou-se a distância Euclidiana para cada um dos quatro vértices, achando tanto a sua média como o seu valor máximo.

Na figura abaixo encontra-se um resumo do processamento relativo à primeira tarefa.

## Segmentação e Contagem Celular

Nesta segunda etapa começou-se por abrir de novo o diretório das imagens provenientes da base de dados utilizada. Estas foram corridas e processadas uma a uma de forma a se segmentar as células nelas presentes. É de notar que, para efeitos de contagem, considera-se que as células cujo centro ultrapassar o lado direito ou inferior da ROI não entram na contagem.

Após a leitura, esta é passada a escala de cinza e a primeira função na etapa de processamento é a *getROI*. Esta tem o objetivo de, através das máscaras da região de interesse utilizadas como *ground truth* na tarefa anterior, obter apenas a parte da imagem correspondente à ROI. Esta etapa é necessária visto que se pretende proceder à contagem apenas dentro desta.

De seguida, recorreu-se à função *segmentandPlotCells* que começa por fazer o processamento necessário para identificação das células, sob a fórmula de círculos. Assim, obtém-se dois *arrays*, um correspondente aos centros destas e outro aos raios. Dentro desta função é ainda chamada a *excludeBorders* que permite garantir a condição anteriormente mencionada: células que ultrapassem os limites direito ou inferior da ROI não entram na contagem. De seguida, estes arrays dados são tratados de forma a transformar o círculo obtido num retângulo que envolve a célula em causa. Isto é possível calculando o canto superior esquerdo do quadrado, recorrendo ao centro e subtraindo o seu raio. Assim, através da função *rectangle* de Matlab, é possível desenhá-los sobre a imagem original.

De seguida, é útil desenhar também os retângulos correspondentes ao *ground truth* para efeitos de comparação visual. Isto é feito através da função *plotGroundTruth*.

Posto isto, para avaliar a exatidão do processo, recorreu-se à função *evaluateSegmentation*. Esta recebe como parâmetros a informação acerca dos retângulos obtidos tanto de forma automática como manual e permite determinar os valores que vão permitir a sua avaliação: o número de células em cada uma das segmentações (automática e manual), o número de verdadeiros positivos, falsos positivos e falsos negativos e, com isto, os valores de *recall*, *precision* e *F-measure* com β igual a 1.

Juntamente com isto, a informação relativa às posições dos retângulos que envolvem as células é guardada num ficheiro .mat, com o nome da imagem, numa nova pasta que contém os resultados desta etapa, juntamente com um *report* geral em ficheiro texto com o nome de todas as imagens e os respetivos índices obtidos. Ao avançar entre as imagens, são também mostradas numa figura em Matlab a imagem original sobreposta com as duas segmentações e os valores de avaliação obtidos.

### Obtenção da ROI

A função *getROI* recebe como *input* a imagem a processar e o seu índice no diretório. Com isto, abre o diretório das máscaras correspondentes ao *ground truth*, obtém a máscara correspondente, procedendo à multiplicação entre a imagem e a máscara.

Desta forma, obtém-se uma moldura preta, com um quadrado correspondente à ROI da imagem de *input* no seu interior, em escala de cinza.

### Segmentação Celular

Processo de segmentação celular e merdas.

Chamar a seguinte algures.

Plot da segmentação.

### Exclusão dos Bordos

Esta função é chamada aquando da segmentação celular.

Começa por aplicar um filtro de mediana, de forma a diminuir ruído na imagem, e a fazer uma equalização do seu histograma, que recorre ao algoritmo CLAHE de forma a aumentar o contraste da imagem.

De seguida, baseia-se em parte do processamento realizado na primeira parte de forma a juntar as linhas adjacentes que formam os limites da ROI (pelo fecho com um elemento estruturante em forma de linha, tanto vertical como horizontal) e eliminação das restantes linhas da grelha (pela abertura com um elemento estruturante em forma de disco).

Posto isto, recorre a *multithresholding* para binarizar apenas as linhas adjacentes, ou seja, que formam a moldura da ROI, deixando tudo o resto a zeros. Seguidamente é calculado o gradiente desta, obtendo as linhas que formam a moldura, tanto exteriores como interiores.

Estas linhas são, de seguida, identificadas através da Transformada de Hough. Esta usa ângulos de 0 para identificação de linhas verticais e -90/89 para identificação de linhas horizontais. Correndo a matriz de linhas obtidas, identificou-se, em primeiro lugar, as linhas correspondentes aos limites exteriores direito e inferior e, tendo isto, à identificação dos limites interiores direito e inferior, que são retornados pela função.

### Avaliação da Segmentação

Esta recebe como parâmetros a informação que permite recriar os retângulos envolventes das células, tanto da segmentação manual como da automática.

Em primeiro lugar, obtém o número de células em cada uma delas, através do tamanho das matrizes. Idealmente, estes valores seriam iguais.

Forma de obter TP, FP e FN.

Seguidamente calcula os valores de *recall* (R), *precision* (P) e *F-measure* com β igual a 1 (F1) através da aplicação das fórmulas diretas:

(1)

|  |  |  |
| --- | --- | --- |
|  | Média | Desvio Padrão |
| Jaccard | 0,9909 | 0,0003 |
| Máximo Distâncias | 6,2862 | 2,2987 |
| Média Distâncias | 4,5912 | 1,4376 |

(2)

(3)

# Resultados e Discussão

## Delineamento da Região de Interesse (ROI)

## Quanto às 50 imagens de treino, estas mostraram resultados de segmentação como mostra a figura 3.

## Como se pode observar, a imagem segmentada é bastante semelhante à original, dando uma boa representação da ROI. Aproximando no bordo inferior, verifica-se que este não é uma linha perfeitamente reta. Isto é algo que acontece em algumas das imagens devido a uma ligeira erosão dos bordos da ROI e ligeira inclinação de algumas imagens.

Quanto aos valores para avaliação, estes encontram-se resumidos para toda a base de dados de treino na seguinte tabela:

Quanto à similaridade, os valores de Jaccard são bastante elevados e muito constantes entre si (baixo desvio padrão), o que mostra que as ROI obtidas são muito significativas da ROI verdadeira. Quanto às distâncias, estes valores estão representados em pixéis. Tendo em conta que as imagens originais têm dimensão 1200x1600 e as ROI’s cerca de 1000x1000, então os valores obtidos são também bastante bons, estando os vértices em posições muito próximas entre si.

Recorrendo às funções *tic* e *toc* de Matlab, obteve-se um tempo total para leitura, processamento, avaliação e armazenamento dos resultados de aproximadamente 33 segundos.

Fase de teste.

## Segmentação e Contagem Celular

# Conclusão

# Trabalhos Futuros

##### References

1. “Cell Counting”. BioTek. Acedido em: https://www.biotek.pt/pt/applications/cell-counting.html

[2] “Manual and Automated Cell Counting”. Isogen. Acedido em: https://www.isogen-lifescience.com/laboratory-instruments/manual-and-automated-cell-counting.html?gclid=Cj0KCQjw-\_j1BRDkARIsAJcfmTGK\_nSFiITC5xxd7hRrY7MKzdKVuxc\_k2eoW3gFTg99ebGAymeprA0aAqBNEALw\_wcB

[3] Akin Ozkan. “Computer Vision based automated cell counting pipeline: a case study for HL60 cancer cell on hematocytometer”. Biomedical Research. 2018. Acedido em: https://pdfs.semanticscholar.org/c35c/1fb16bf42dfd3f8cb9bd9c54240c92b77fd7.pdf

[4] Yevgeniy Grigoryev. “Cell Counting with a Hematocytometer: Easy as 1, 2, 3”. BiteSize Bio. Dezembro 2014. Acedido em: https://bitesizebio.com/13687/cell-counting-with-a-hemocytometer-easy-as-1-2-3/