MC920 – Trabalho 3

Rafael Sartori M. Santos, 186154

17 de outubro de 2019

1 Introdução

O objetivo deste trabalho é separar, rotular, contar objetos desconexos utilizando limiarização e análise de vizinhança, verificar propriedades, como área, perímetro, centroide e, por fim, classificar entre pequeno, médio e grande pela área ocupada.

Faremos isso utilizando um programa escrito em *Python* utilizando a biblioteca padrão, *OpenCV* e *scikit-image* para processar as imagens e *Matplotlib* para histograma da área calculada dos objetos da imagem.

2 Método

O trabalho é facilmente divisível em várias etapas. Para cumprir todos os objetivos, será necessário:

- monocromatizar a imagem de entrada,
- rotular objetos desconexos,
- contá-los,
- extrair propriedades,
- classificá-los quanto à área.

A razão para aplicação de cada um desses passos será explicada junto com sua metodologia.

Para a aplicação de fato, no entanto, iremos utilizar funções já implementadas de *OpenCV* e *scikit-image*, já que seus funcionamentos não diferem tanto dos métodos descritos nesta seção.

2.1 Monocromatização

Como a imagem de entrada é colorida, teremos mais de uma camada de cor e isso tornará difícil a identificação de um objeto ou fundo. Portanto, qualquer tipo de transformação que produza uma imagem de camada única é suficiente.

Para produzir uma saída em apenas uma camada, podemos utilizar uma função que tem como parâmetro as várias camadas ou ainda fazer uma limiarização.

Na aplicação, abrimos a imagem em modo de escala de cinza através da *flag* em *OpenCV*.

2.2 Rotulação e contagem

Com a imagem monocromática, podemos identificar os objetos através do agrupamento usando `vizinhança-4` ou `vizinhança-8`. Podemos fazer isso seguindo este plano:

- inicializamos uma variável que é o número do objeto que estamos identificando atualmente (começa com zero);
- inicializamos a matriz de "resposta" de mesma dimensão que a entrada com zero (ela guardará o rótulo e quais pontos pertencem a esse objeto);
- percorremos a imagem toda; ao encontrarmos um objeto (não fundo), fazemos:
  - incrementamos a variável do objeto que estamos identificando, esse é o número do objeto atual;
  - navegaremos dentro dele utilizando a vizinhança selecionada, marcando numa matriz de "resposta" o número do objeto atual;
  - ao não possuir mais pontos que não são fundo alcançáveis pela vizinhança, continuamos percorrendo a imagem.
- terminamos a imagem tendo percorrido todos os pontos e rotulado todos os objetos desconexos entre si.

Dessa forma, automaticamente já contamos os objetos presentes, é o valor final da variável do número de objetos que identificamos.

Com essa matriz, será possível contar a área, identificar perímetro e encontrar centroide de cada objeto.

Essa etapa é realizada pelo *scikit-image* através da função `label`, que identifica e rotula objetos desconexos dada uma vizinhança.

2.3 Medir área e perímetro

Do resultado do método anterior (2.2), podemos medir a área simplesmente contando os pontos que possuem mesmo valor ao rótulo do objeto. Por exemplo, para o rótulo 4, contamos os pontos cujo valor é 4 na matriz retornada.

Já para o perímetro, é mais difícil: será necessário encontrar o contorno desses objetos e depois contar os pontos pertencentes a ele. Há algumas maneiras de se fazer isso, por exemplo: encontrando os que possuem alguma vizinhança com o fundo, utilizando morfologia matemática na imagem.

Essa etapa é realizada através das funções `getprops` (utilizando a rotulação anterior) e `perimeter` (isolando os pontos dos rótulos individualmente) do *scikit-image*.

2.4 Calculando centroide

A centroide da imagem é a posição central (C_x, C_y) do objeto na imagem bidimensional, ou seja, a posição mediana em relação a área. Pode ser calculada por decomposição geométrica em pequenos retângulos (os pontos) através da eq. (1).

$$C_l = \frac{\sum C_{i,l} A_i}{\sum A_i} \tag{1}$$

Onde l é a dimensão em que queremos calcular a centroide, $C_{i,l}$ é a coordenada na dimensão l da i-ésima parte da decomposição do objeto C. Essa decomposição é feita automaticamente pela digitalização da imagem (em pequenos quadrados, os *pixels*).

Esse passo no código já é realizado na seção 2.3 pelo `getprops`.

2.5 Classificação quanto a àrea

A classificação é mais simples. Com a área A do objeto calculada, basta verificar em qual categoria se encaixa. São elas:

- pequeno, se $A < 1500$;
- médio, se $A \in [1500, 3000)$;
- grande, se $A \geq 3000$.

Com a classificação, produzimos um histograma em que, no eixo horizontal, temos o tamanho do objeto e, no vertical, a quantidade de objetos.

O histograma é feito pelo *Matplotlib*.

3 Resultados

.

4 Conclusão

O trabalho mostra que, a partir de uma simplificação de uma imagem (que pode ser facilmente obtida se as condições são controladas), é rápido determinar a área, posição e tamanho de objetos, informações muito requisitadas em automação industrial, por exemplo, para verificação.

Com as informações obtidas, como o contorno, conseguimos comparar com o que era esperado através de funções matemáticas e/ou outros algoritmos estudados na disciplina.

Os resultados foram bastante satisfatórios e pouca correção foi necessária dado que as funções já foram implementadas e testadas pelas bibliotecas.