Projeto 3 - MC920A - Medidas de imagens

Marcelo Biagi Martins - 183303

1 Introdução

Este projeto de Processamento de Imagens teve como objetivo a implementação de diversas funções pertinentes para a análise de uma imagem ou extração de dados desta. As funções implementadas no código conseguem obter os subcontornos de uma imagem, transformar a imagem original em uma monocromática ou binária, calcular área, excentricidade, solidez e perímetro das regiões da mesma; e retornar um histograma em forma de gráfico de barras que agrupa as áreas das regiões da imagem de entrada.

Para obter essas medidas, a biblioteca *opencv*, utilizada neste projeto, apresenta nativamente funções que obtém medidas à partir dos momentos de uma imagem, que são medidas ponderadas sobre as intensidades dos pixels das imagens que permitem ao usuário a extração de informações e posterior análise; e estas funções foram necessárias para obter a maioria das medidas requisitadas.

2 Especificações e funcionamento do programa

O script python criado para este projeto pode ser executado com o comando *python3 proj3.py + flags*. As flags em questão servem para alterar os parâmetros de execução do código e estão dispostas na tabela 1:

Tabela 1: **Flags de execução**

Flag	Opções	Valor default
--folder	Diretório com as imagens png	imagens/
--image	Qualquer nome de imagem dentro do folder	objetos3

Além disso, existe a flag obrigatória *--option OPT* que recebe a função que o usuário deseja executar, conforme descrito na tabela 2:

Tabela 2: **Métodos de limiarização**

OPT	Função
1	transformar imagem para binária
2	transformar imagem para escala de cinza
3	gerar os contornos das regiões
4	imprimir número de regiões, área, perímetro, excentricidade e solidez da imagem no terminal e numera as regiões
5	imprimir número de regiões grandes, médias e pequenas e gerar histograma de área
6	todas as opções acima

Assim, por exemplo, o comando *python3 proj3.py --option 4 --image objetos2* executa o script para a imagem *objetos2.png* e gera a área, perímetro, etc das regiões da imagem. A seguir, cada função é explicada mais a fundo.

3 Resultados

As opções 1 e 2 são diretas: salvam a imagem (binária ou escala de cinza) com o nome *grayscale_objetos1.png* ou *binary_objetos1.png*, por exemplo. De forma semelhante, a opção 3 gera os "contours"da imagem com a função *cv2.findContours()* e retorna os mesmos em um fundo branco com o nome *edges_objetos1.png*, por exemplo. A imagem teste *objetos2.png* na forma binária, escala de cinza e com seus contornos estão na figura 1 e a imagem original está na figura 4.

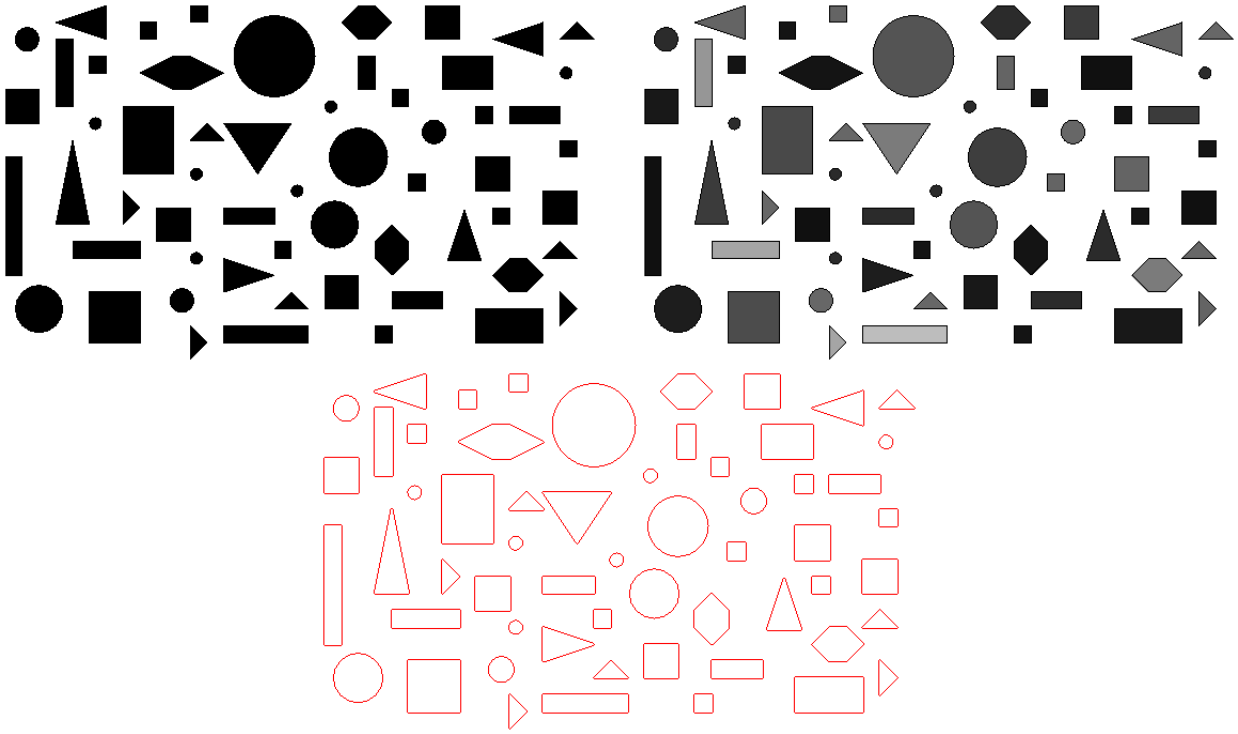

Figura 1: Objetos 2 binário, escala de cinza e seus contornos

A opção 4 gera e imprime no terminal o número de regiões, área, perímetro, excentricidade e solidez das regiões da imagem, utilizando funções como *cv2.contourArea()* para área, *cv2.arcLength()* para perímetro e outras funções do opencv. Além disso, gera uma imagem com cada região da imagem numerada na ordem inversa com que elas são percorridas, com o uso da função do opencv *cv2.putText()* com o nome *numbers_objetos1.png*, por exemplo. Para facilitar a visualização dos números, as imagens são transformadas em binárias antes da escrita e seus números são escritos na cor branca. Essas imagens, geradas a partir das imagens teste, estão dispostas na figura 2. Um exemplo de saída da opção 4 utilizando como imagem teste a figura *objetos3.png* é:

```
Número de regiões: 9

Região 0: Área: 4107 Perímetro: 319.421354 Excentricidade: 0.738345 Solidez: 0.754963
Região 1: Área:  844 Perímetro: 125.639609 Excentricidade: 0.750543 Solidez: 0.904558
Região 2: Área: 3690 Perímetro: 265.119838 Excentricidade: 0.907458 Solidez: 0.978264
```

```
Região 3: Área:  584 Perímetro: 104.911687 Excentricidade: 0.871986 Solidez: 0.913928
Região 4: Área:  478 Perímetro:  94.426406 Excentricidade: 0.883524 Solidez: 0.925460
Região 5: Área: 1762 Perímetro: 179.781745 Excentricidade: 0.875949 Solidez: 0.971862
Região 6: Área:  688 Perímetro: 108.669047 Excentricidade: 0.877953 Solidez: 0.972458
Região 7: Área: 4067 Perímetro: 311.078208 Excentricidade: 0.881921 Solidez: 0.780689
Região 8: Área:  716 Perímetro: 101.982755 Excentricidade: 0.625314 Solidez: 0.980164
```

Para obter essas medidas, os momentos das imagens foram obtidos com a função *cv2.moments()* e um laço percorria cada região de cada imagem. Com os momentos "m00", "m01"e "m10", por exemplo, pode-se obter o centroide para cada região da forma $(\frac{m10}{m00}, \frac{m01}{m00})$.

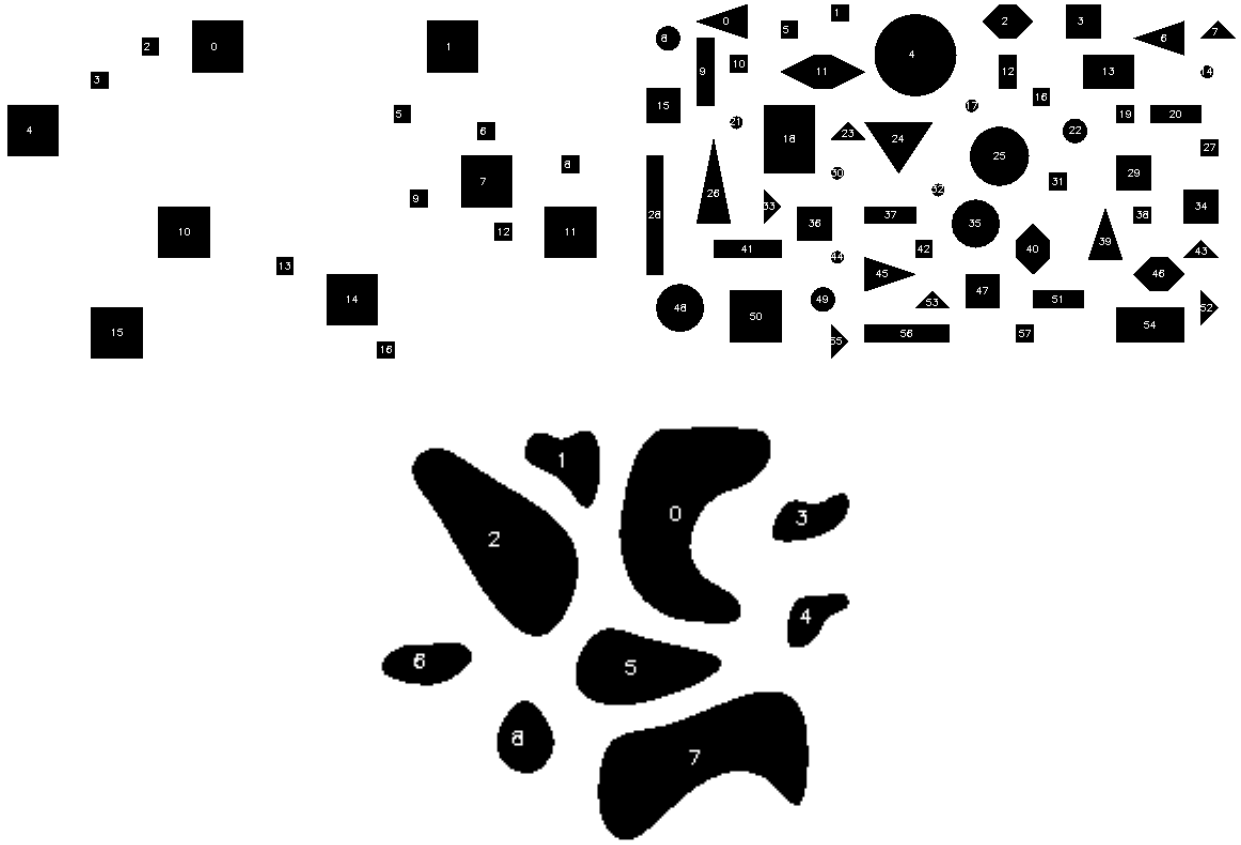

Figura 2: Imagens com regiões numeradas

A opção 5 gera e imprime no terminal a quantidade de regiões agrupadas por área e também um histograma na forma de gráfico de barras. Um exemplo de saída desta opção é

```
número de regiões pequenas: 5
número de regiões médias: 1
número de regiões grandes: 3
```

E os histogramas gerados estão na figura 3. Como as imagens teste possuem suas regiões bem nítidas, é simples contar a quantidade de regiões na mão para conferir alguns dados e é possível notar que os resultados são condizentes.

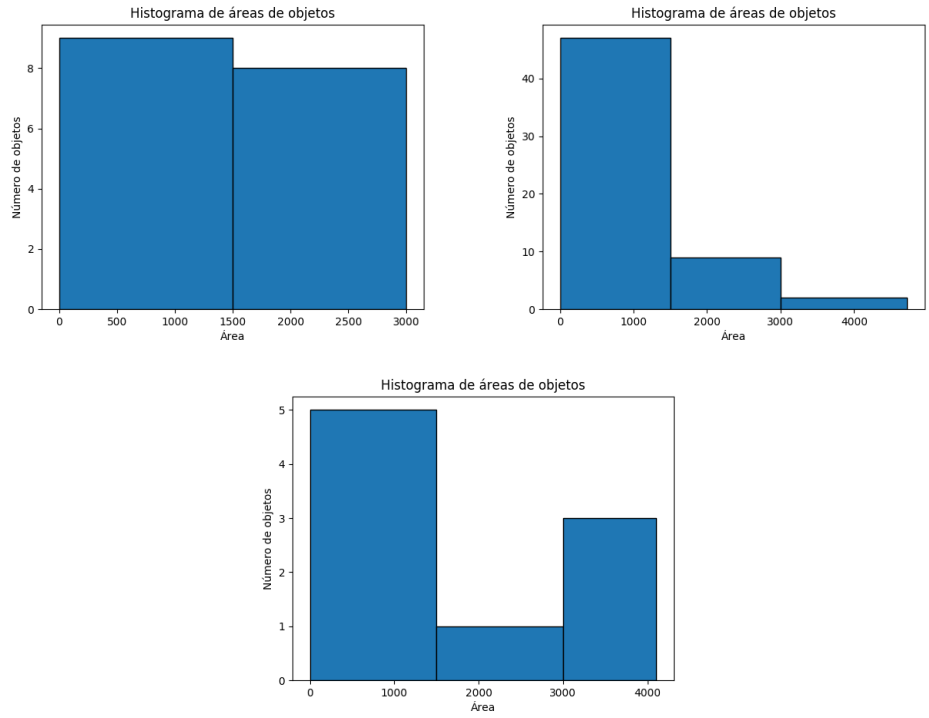

Figura 3: Histogramas gerados

4 Conclusões

Esse trabalho possibilitou a obtenção de diversos dados das imagens teste, além de ter possibilitado uma melhor compreensão da biblioteca opencv-python e de conceitos como momentos de imagens. Como as imagens teste foram simples, os resultados esperados foram atingidos e assim, as medidas obtidas foram precisas.

Referências

[1] *Contours e momentos*, Em https://docs.opencv.org/2.4/modules/imgproc/doc/structural_analysis_and_shape_descriptors.html

[2] *Funções do Opencv para calcular medidas das regiões*, Em https://docs.opencv.org/trunk/dd/d49/tutorial_py_contour_features.html

5 Anexos

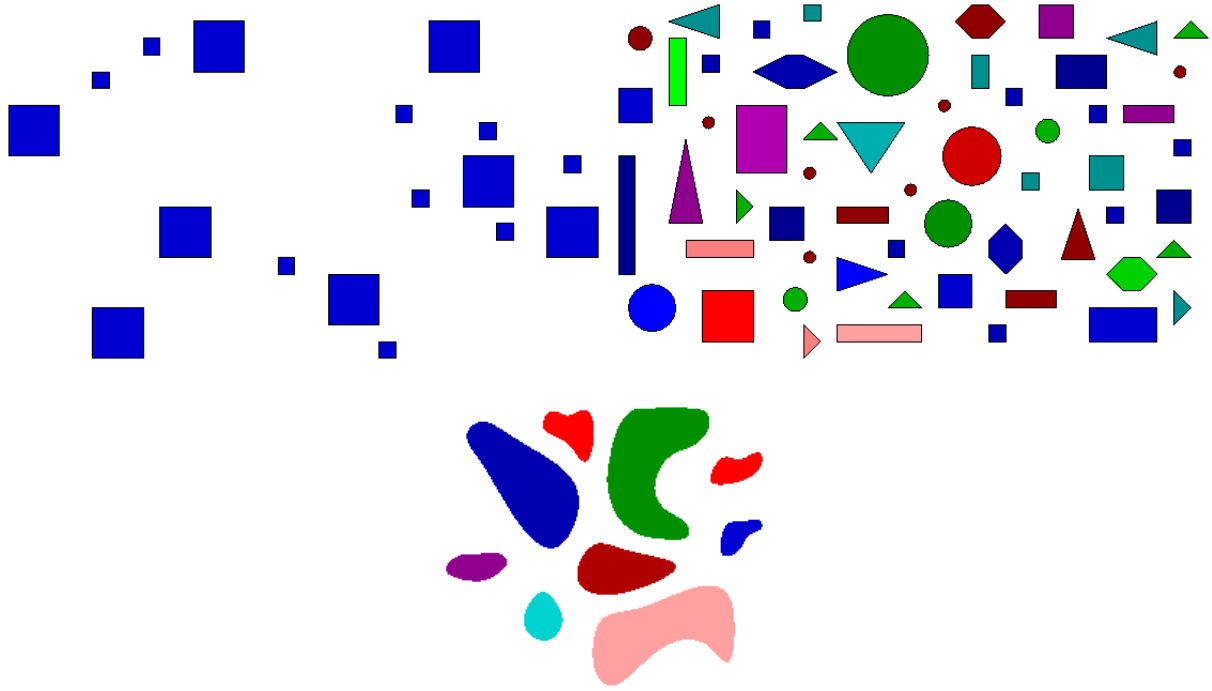

Figura 4: Figuras teste