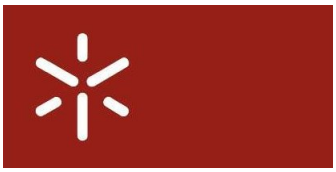

Universidade do Minho

Mestrado Integrado em Engenharia Informática

Visão Por Computador
Tutorial 2: Image Recognition

Grupo 7

Adriana Meireles
A82582

Mariana Pereira
A81146

3 de Janeiro de 2020

Conteúdo

1 Introdução

O segundo tutorial da unidade curricular Visão por Computador tem como principal objetivo a implementação de técnicas, lecionadas nas aulas da disciplina, de reconhecimento de imagem, incluindo image segmentation e categorization. Para isto, utilizou-se a ferramenta *Matlab*.

2 Conteúdos

O objetivo deste tutorial é criar um programa que consiga contar o número de moedas existentes numa determinada imagem e reconhecer o tamanho numa tentativa de quantificar o dinheiro na imagem.

2.1 Design a program for image recognition of circular objects

Neste tutorial criou-se um script com o nome *image recognition* que por sua vez iria chamar uma função com o nome *main image recognition.m* responsável por executar as funções que estão descritas no ponto 2 do enunciado.

2.1.1 Inputs

1. grey-scale image
2. type of noise
   - Salt and Pepper
   - Gaussiana
3. noise parameters
   - Densidade do Barulho (SP)
   - média (Gauss)
   - Variância (Gauss)

2.1.2 Algorithm

No segundo ponto do enunciado são pedidas várias funcionalidades que serão explicadas de seguida:

- Criação de uma uma função para adicionar ruído a uma imagem.
  - **addnoiseSP** - necessita da densidade do barulho (noise_,den) (Barulho SaltPepper)
  - **addnoiseG** - necessita da média e da variância (var_g) (Ruído Gaussiana Noise)
- Usar técnicas de pré-processamento, como métodos de normalização,filtragem, equalização de contraste, para processar a imagem:

  - **Normalização** - As imagens quando lidas são *resized* por causa do seu tamanho ser diferente uma das outras
  - **Filtragem** - Dependendo do tipo de ruído a filtragem é feita de forma diferente:
    * Para ruído do tipo Salt and Pepper é aplicado um filtro mediana. Para tal,utiliza-se a função *medfilt2* com um kernel de tamanho [5 5]
    * Para ruído do tipo Gaussiana é aplicado um filtro gaussiano utilizando a função *imgaussfilt*com sigma 0.5
  - **Equalização de contraste** - A equalização do contraste é realizada através da equalização adaptativa do histograma. Para tal,utiliza-se a função *adapthisteq* com o parâmetro 'NumTiles' com o valor [20 20].

- Processo de segmentação
  - **Deteção de edges** - Aplica-se o Canny edge detector com threshold [0.25 0.35].
  - **Deteção de círculos** - De modo a encontrar as moedas que existem na imagem utiliza-se uma função com o nome *find_circ* que recebe o número da imagem (cada uma precisa de parâmetros diferentes para ser possível detetar) e a própria imagem. Dentro desta função é utilizada a função pré definida *imfindcircles* responsável por usar a transformada de Hough para detetar círculos. Os parâmetros utilizados dentro desta função são:
    * **Imagem**
    * **Intervalo de raios** - 50 até 150
    * **Sensitivity** - corresponde à sensibilidade do array acumulador da Circular Hough Transform.Quanto maior o factor, mais objetos circulares são detetados pela função,o que pode provocar um aumento do risco de deteção falsa. Valor usado = 0.95.
    * **EdgeThreshold** - responsável por configurar o gradient threshold para a determinação de edge pixels na imagem. A função deteta mais objetos circulares,com edges fortes e fracas,para um threshold com um valor baixo. Os valores usados para a primeira,segunda e terceira imagens são respetivamente 0.32, 0.42 e 0.22.
    * **ObjectPolarity**- Exibe se são mais claros ou escuros os objetos circulares comparativamente ao background. Valor utilizado = 'bright'
      O output da função é o seguinte:
      · centros - Coordenadas dos centros dos círculos (x e y).
      · raio - Raio estimado para os centros dos círculos.

· métrica - Strength relativa para o centro dos círculo

  - Para terminar, os círculos são desenhados na imagem usando a função *vis_circ*. Por último, é colocada uma label com o número de cada moeda.

- **Contar moedas** - Existe uma função com o nome *count_coins*, que recebe o raio(output da função imfindcircles radii), responsável por contar as moedas e separá-las entre 2 tipos . As moedas do tipo1 valem 1 ponto enquanto que as moedas do tipo2 valem 2 pontos. O resultado final é apresentado numa caixa de mensagem que contêm o número de moedas encontradas, quantidade de moedas do tipo1 e tipo2 assim como uma estimativa do valor total das moedas por pontos.

- **Desenhar o Histograma** - Para desenhar o histograma utiliza-se a função *ar_hist* que recebe os raios dos círculos,output da função imfindcircles radii.Deste modo desenha dois histogramas: um com a diferença entre os raios dos círculos e outro com a área dos cícrulos. Foram utilizadas as funções *categorical* para criar categorias por cada valor único do raio ou área bem *histogram* para desenhar o próprio histograma.

2.1.3 How To Use?

Para testar este programa é necessário correr o script image recognition e responder às perguntas que aparecem :

- Qual é a imagem que pretende? Escreva 1 para coins.jpg, 2 para coins3.jpg ou 3 para coins2.jpg

- Tipo de barulho? Escreva 1 para barulho Salt-and-Pepper ou 2 para barulho Gaussiana

  - Caso tenha escolhido Salt-and-Pepper é necessário especificar a densidade do barulho. É aconselhável valor inferior a 0.4 para se obter melhores resultados.
  - Caso tenha escolhido Gaussiana é necessário especificar a média e a variância. É aconselhável valor para a média inferior a 0.1 e para a variâcia inferior a 0.05 para a obtenção de melhores resultados.

2.2 Outputs

Para obtenção dos resultados apresentados de seguida foram introduzidos ambos os ruídos na figura coins1, com uma densidade de barulho de 0.3 para Salt and Pepper, e uma média de 0 e variância de 0.03 para a Guassiana.

Nas restantes figuras são apresentados apenas os resultados da aplicação de densidade de barulho de 0.3 para Salt and Pepper.

2.2.1 Coins1

Figura 1: Após sofrer pré-processamento Coins1 com Sal and Pepper

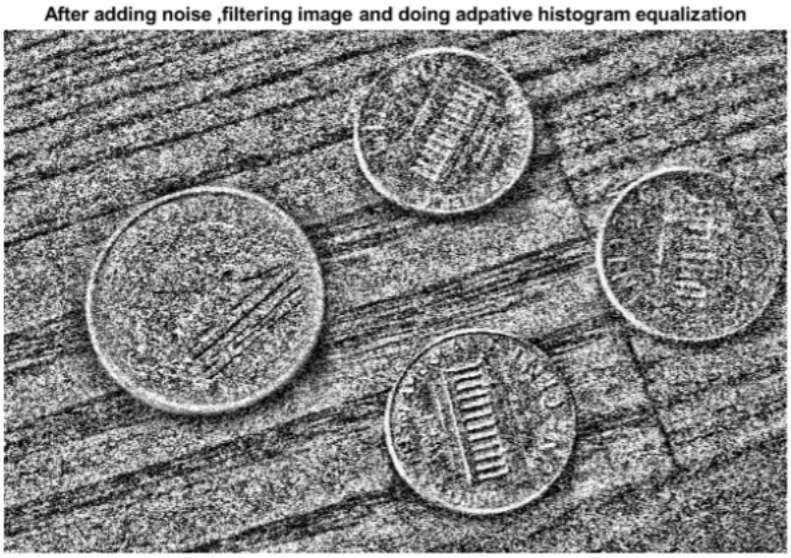

Figura 2: Após sofrer pré-processamento Coins1 com Barulho Gaussiana

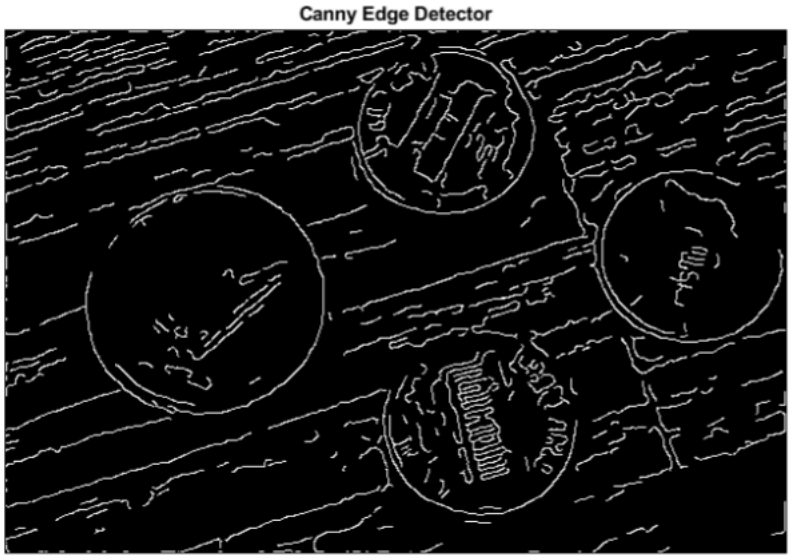

Figura 3: Canny Edge da figura com barulho Salt and Pepper

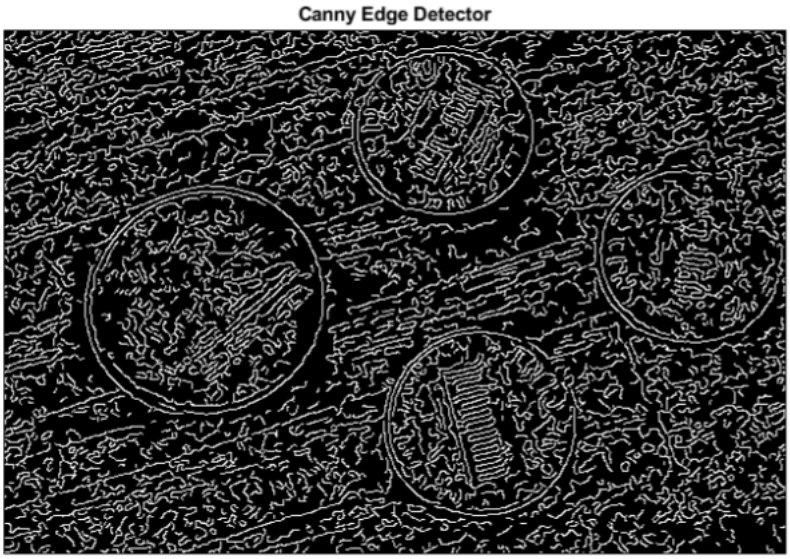

Figura 4: Canny Edge da figura com barulho Gaussiana

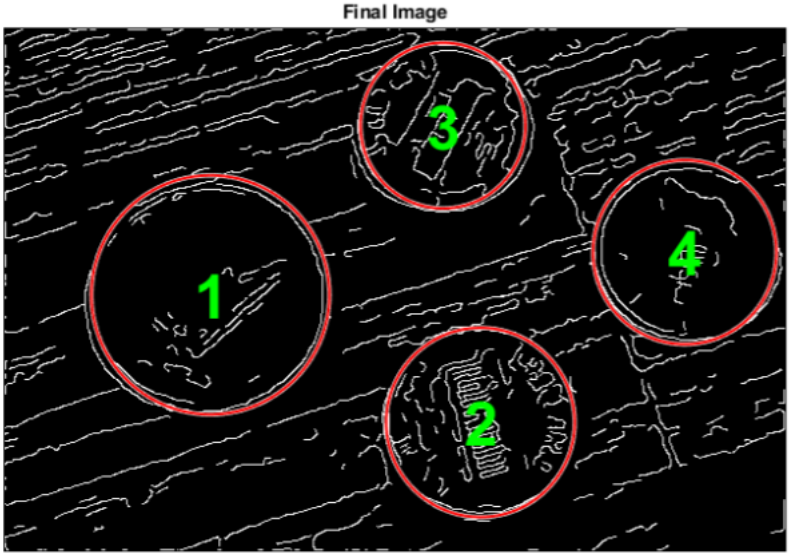

Figura 5: Resultado da Segmentação

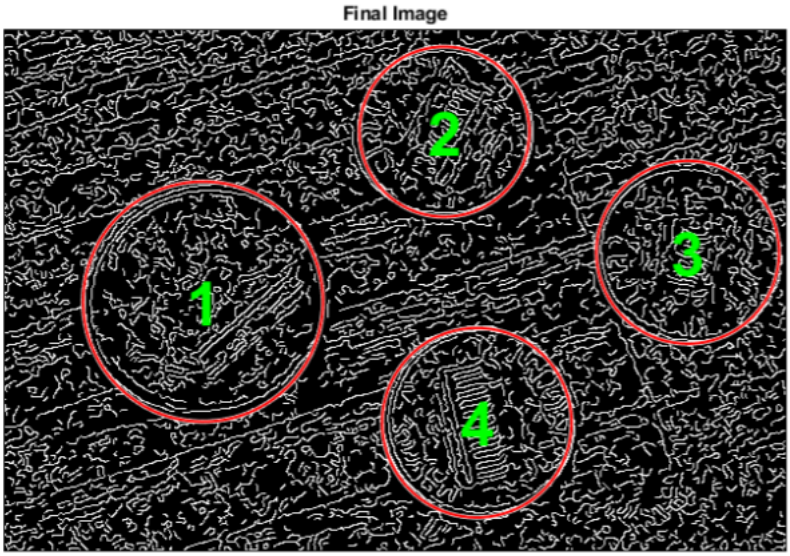

Figura 6: Resultado da Segmentação

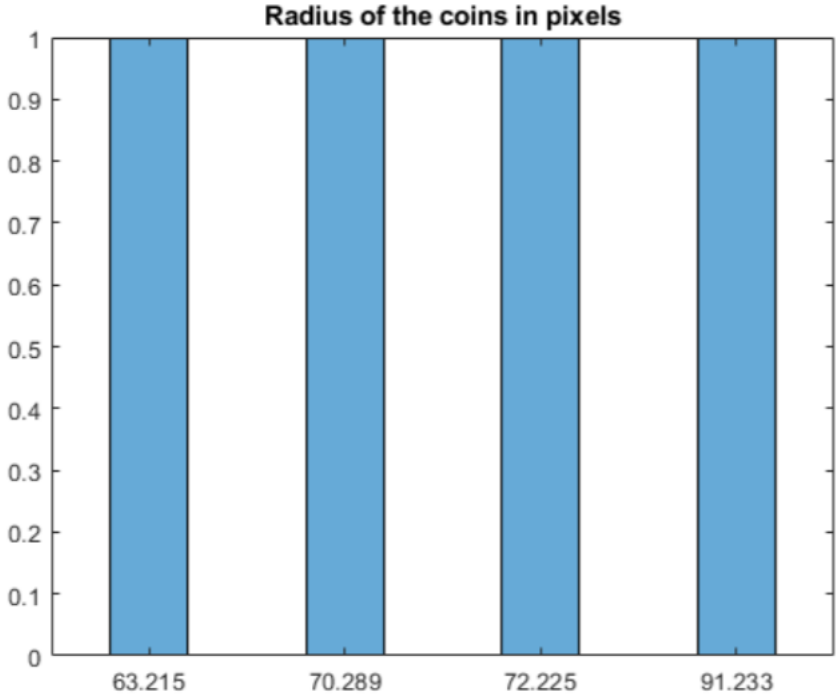

Figura 7: Histograma do Raio das moedas

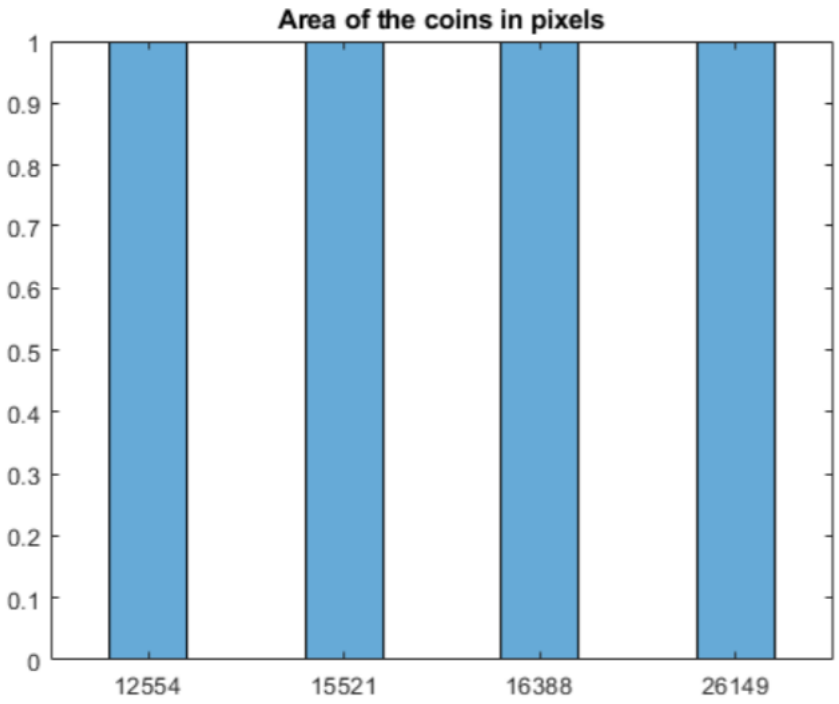

Figura 8: Histograma da área das moedas

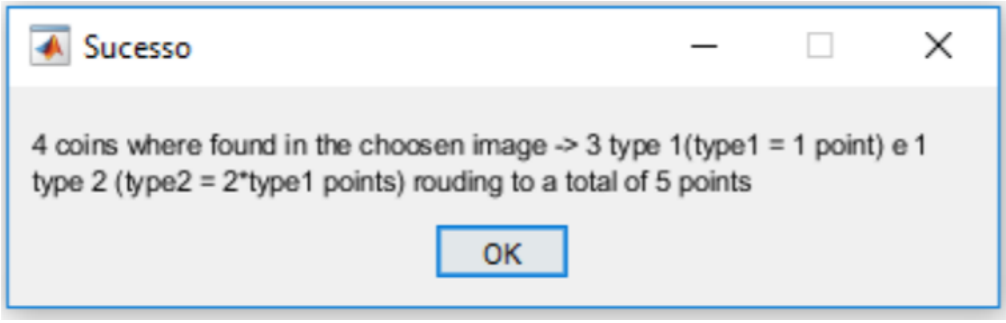

Figura 9: Contagem do valor das moedas

Os resultados obtidos para os dois tipos de ruídos, foram satisfatórios sendo possível idenficar todas as moedas presentes na figura. Foram realizados diferentes testes com variância e barulho de densidade diferentes com sucesso na identificação das moedas, apesar de não estarem apresentados aqui. Foi criado um sistema monetário, visto que não conseguimos resolver o ponto do problema da classificação de objetos. Funcionando este com base nos reais e calculando a quantidade presente na figura. É somado o valor das diferentes quantidades para obter o total, sendo que as moedas de valor 1 com raio inferior a 75 pixéis e as restantes de valor 2.

2.2.2 Coins2

Figura 10: Após pré-processamento Coins2

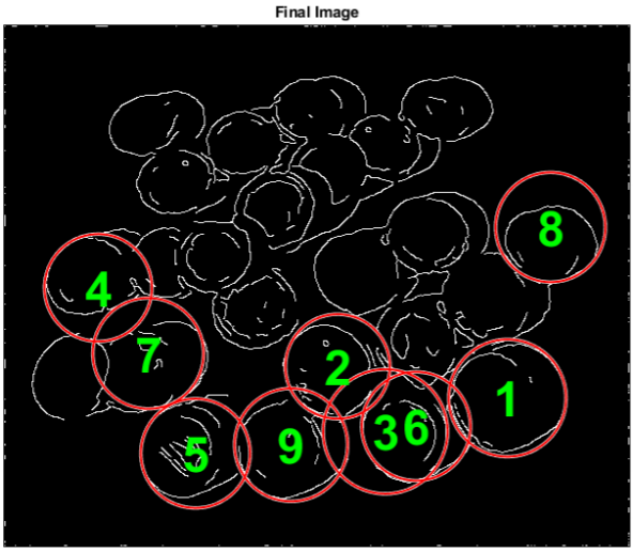

Figura 11: Resultado da segmentação

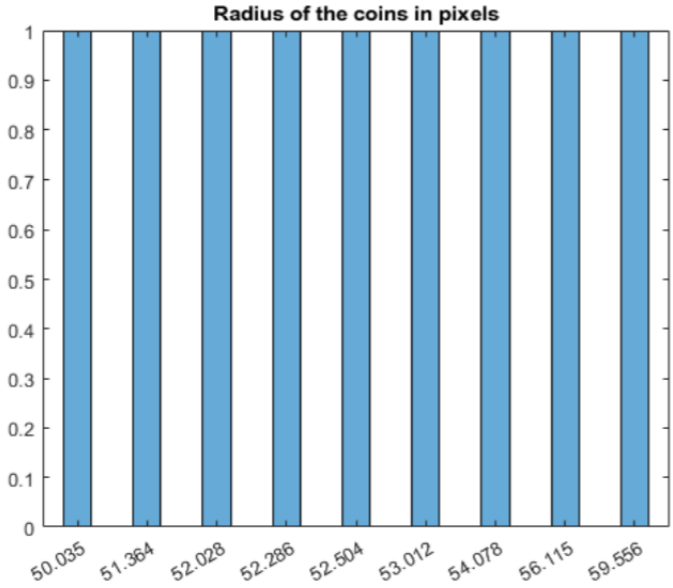

Figura 12: Histograma com os raios das moedas encontradas

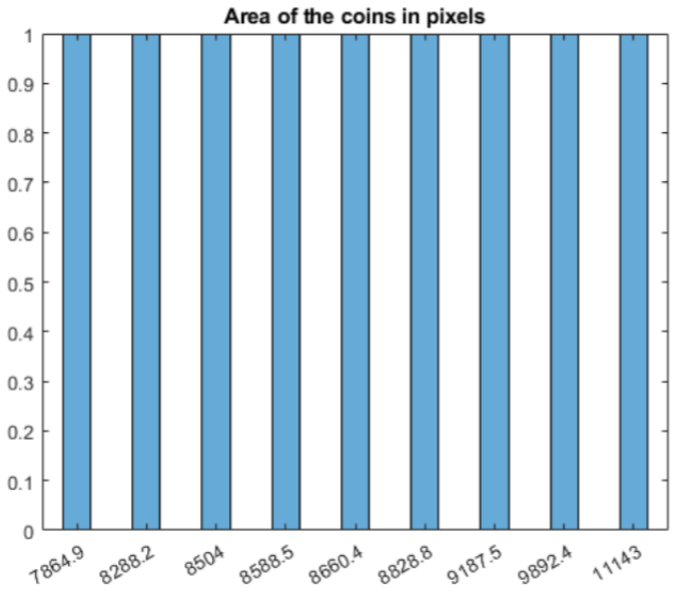

Figura 13: Histograma com a área das moedas

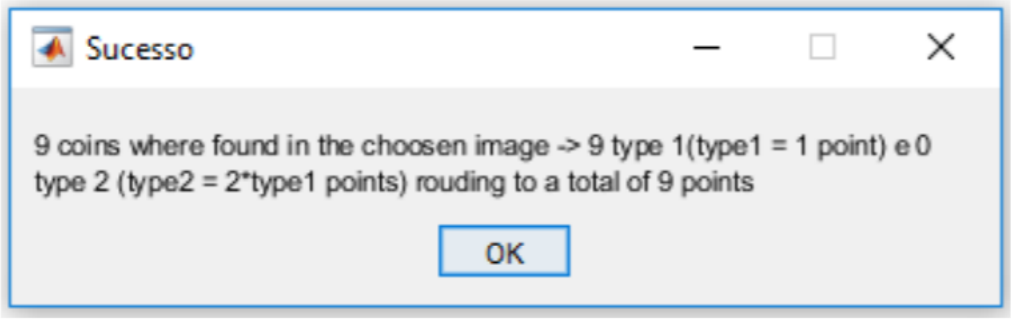

Figura 14: Contagem do valor das moedas

Os resultados obtidos não foram satisfatórios para esta figura, uma vez que não conseguimos identificar todas as moedas presentes na mesma. Devendo-se isto ao facto de aparecer muitas sombras o que dificulta a segmentação das moedas. Para obtenção de sucesso na segmentação desta figura era necessária outra abordagem do que a feita para as restantes figuras. Embora a realização de remoção de sombras e ruídos foi feita de diversas maneiras, mesmo assim não obtemos sucesso. Na figura resultante da segmentação só conseguimos identificar algumas moedas, e analisando a figura original concluímos que algumas dessas moedas são sim sombras de si mesmas. A quantia calculada não corresponde realmente à que se encontra na figura.

2.2.3 Coins3

Figura 15: Após pré-processamento Coins3

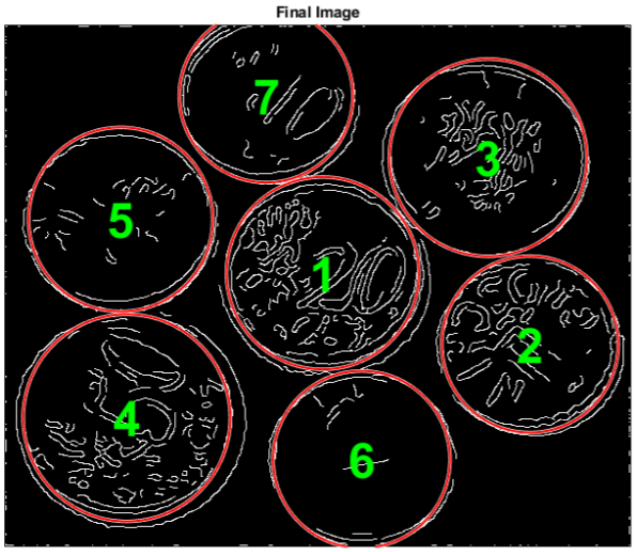

Figura 16: Resultado da segmentação

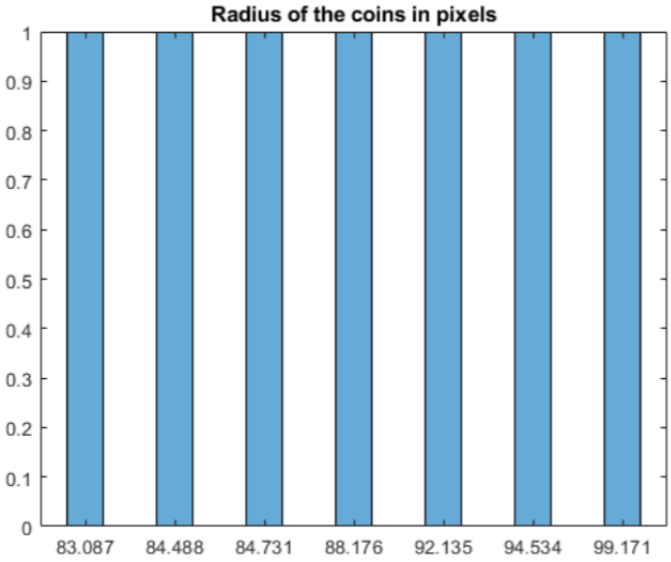

Figura 17: Histograma com os raios das moedas encontradas

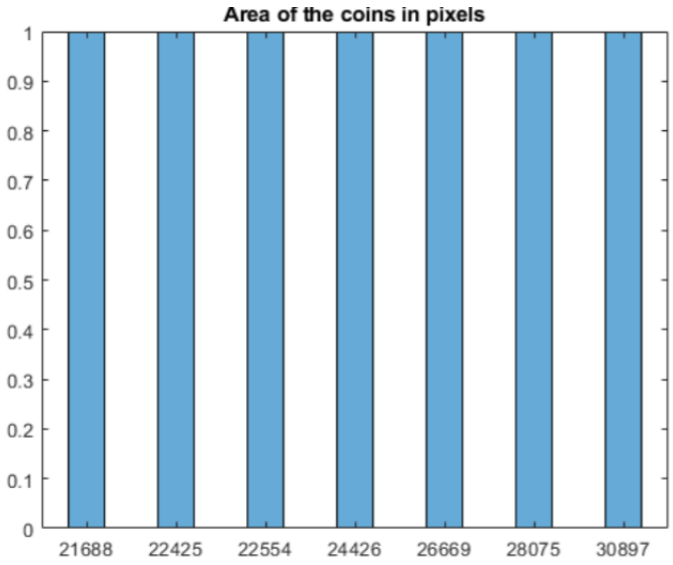

Figura 18: Histograma com a área das moedas

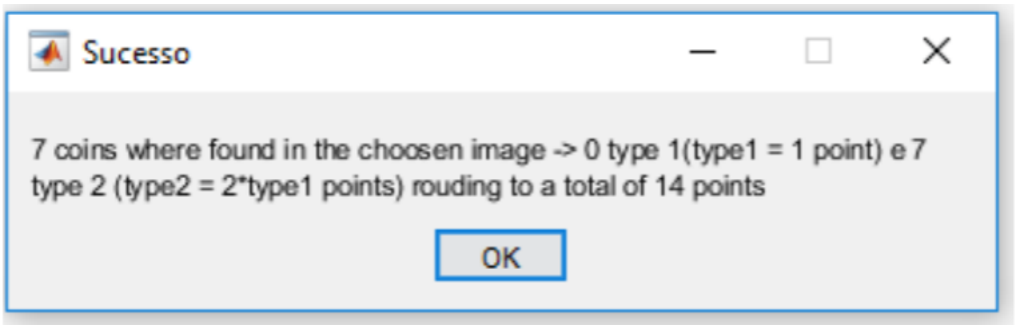

Figura 19: Contagem do valor das moedas

À semelhança do que aconteceu a Coins1, os resultados obtidos foram os pretendidos, identificando assim todas as moedas e cálculo da sua quantidade. Esta figura tem presente uma moeda que se encontra muito mais escura que as outras, portanto nem sempre é possível a identificação recorrendo ao processo de segmentação.

3 Conclusão

Com a realização deste trabalho conseguimos consolidar os conhecimentos das técnicas de pré-processamento de imagem, como normalização e filtragem, entender o processo de segmentação de uma imagem e combater o contraste na imagem através da equalização adaptativa do histograma.

Apesar dos obstáculos encontrados na realização do tutorial estamos satisfeitas com os resultados obtidos no programa final.