Universidade Federal do Agreste de Pernambuco

Bacharelado em Ciências da Computação

Prof. Tiago Buarque A. de Carvalho

Reconhecimento de Padrões

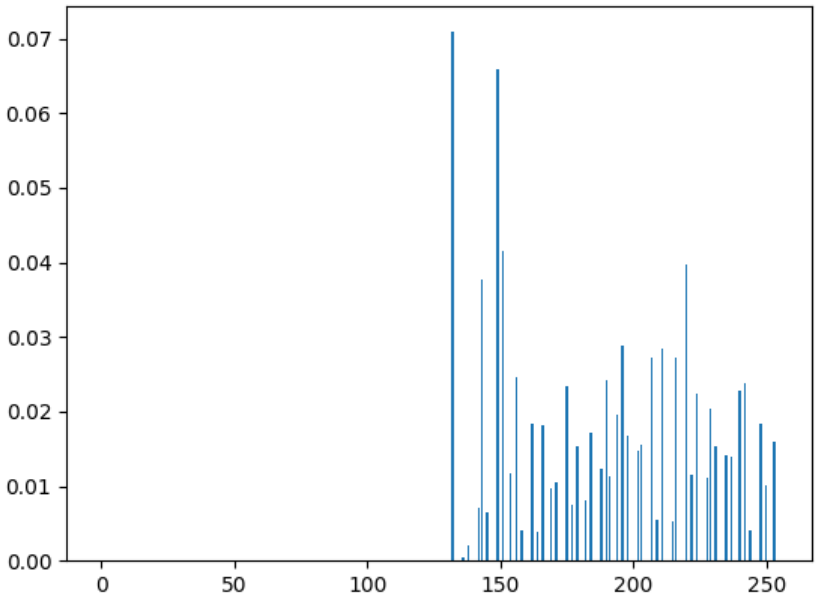
Processamento de Histograma

Limiarização

Filtros Espaciais de Suavização

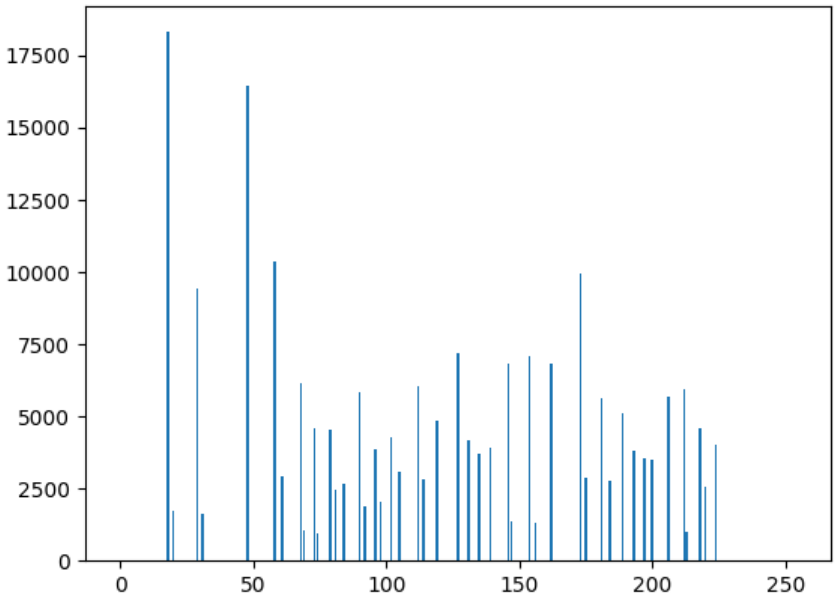
Aluno: Vinícius Santos de Almeida

- 1.
    - Fiz os algoritmos em Python para facilitar as operações e plotar mais facilmente, na letra a, carreguei cada imagem, calculei a quantidade de tons de cinza, e depois fiz o calculo dos histogramas a partir da fórmula mostrada em aula. Na letra b, pude reaproveitar a função `tons_cinza` da questão anterior, com isso criei a função `equalizar_img` que a partir de imagem, e com a fórmula passada em aula, realiza a equalização das imagens, e por último, na letra c, modifiquei a função `equalizar_img` para criar o histograma das novas imagens.
        - Resultados com explicação (letras a, b, c e d):
            - Imagem: Fig0320(1)(top_left).
                - a. Como essa imagem é mais esbranquiçada, é possível notar que os valores com mais frequência são valores maiores de 100.

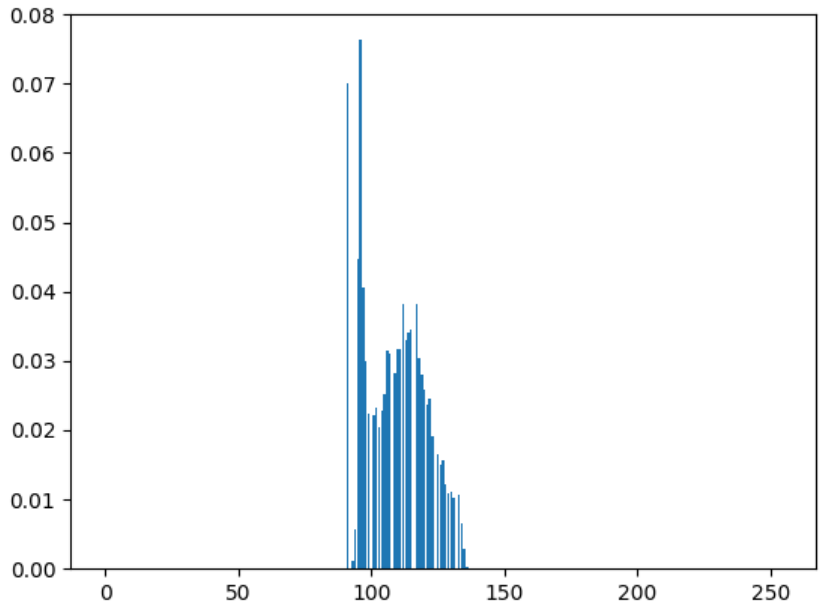


- b. Aqui é esperado que a imagem deixe de ser esbranquiçada, e esquilibre os valores, criando uma imagem menos esbranquiçada. E o resultado é muito satisfatório.

- c. É notável que os valores agora estão mais espalhados e há um equilibrio maior na frequência dos tons.



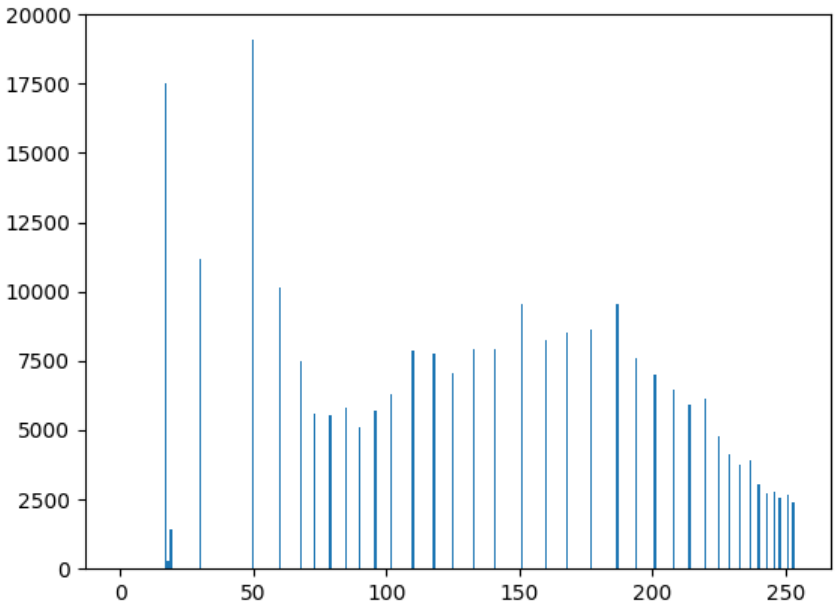
- Imagem: Fig0320(2)(2nd_from_top).
    - a. Já nesta imagem, que é bem "acinzentada", é possível notar que a maioria dos valores está em um intervalo intermediario entre os valores de 50 a 150.



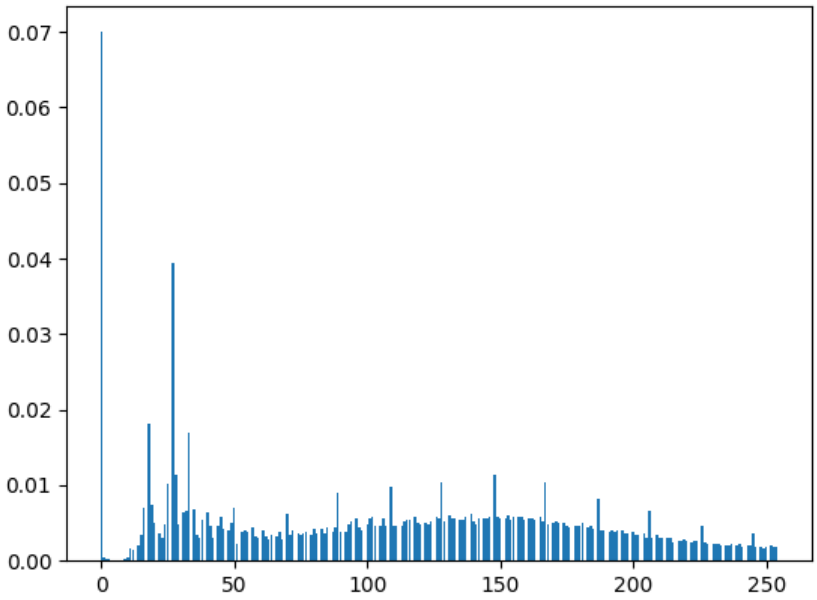
    - b. Assim como na letra a, espera-se que haja um espalhamento dos valores, criando um equilíbrio e é o que de fato acontece, o que resulta numa

imagem bem semelhante à da letra a.

- c. O histograma também, assim como esperado, sofre o espalhamento dos valores, criando o histograma equalizado.

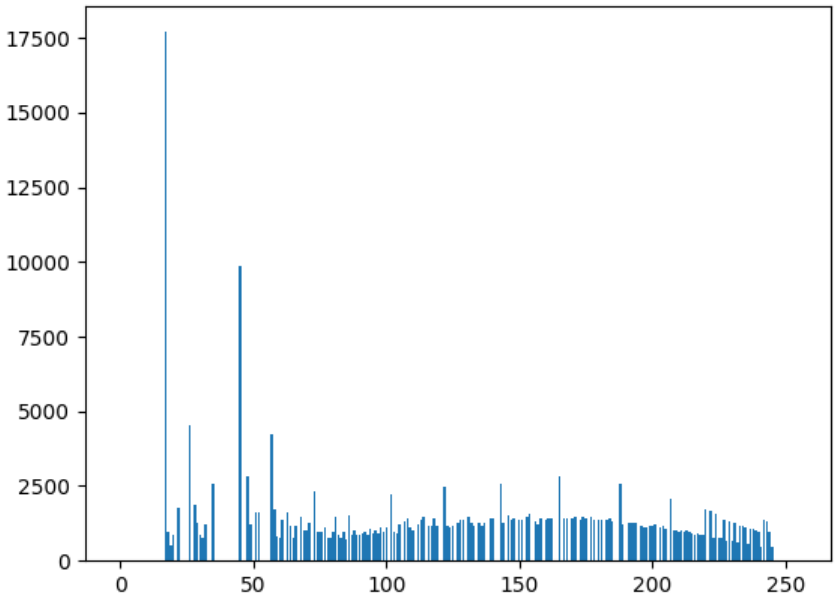


- Imagem: Fig0320(3)(third_from_top).
    - a. Essa é a imagem que tem o maior equilibrio na frenquência dos tons do preto ao branco, por isso, temos um histograma mais balanceado.

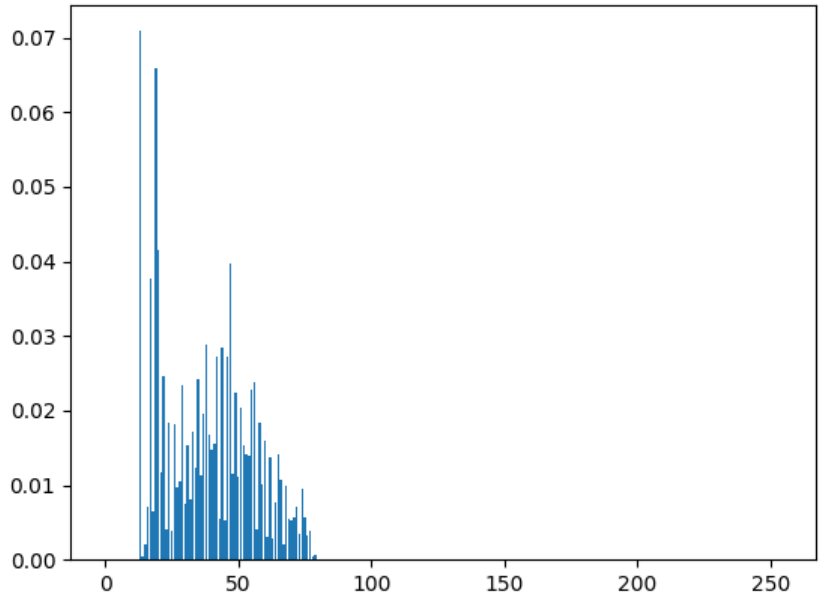


- b. Aqui não espero que haja grande diferença na imagem, já que os valores já estão bem equalizados. Mesmo assim há uma melhora que deixa as partes mais claras um pouco menos claras e as partes escuras são mais escurecidas, o resultado a semelhante aos das letras anteriores.

- c. Também não esperava grande mudança no histograma, e ao olho nu, de fato não houve grande mudanças nele em relação ao não equalizado.



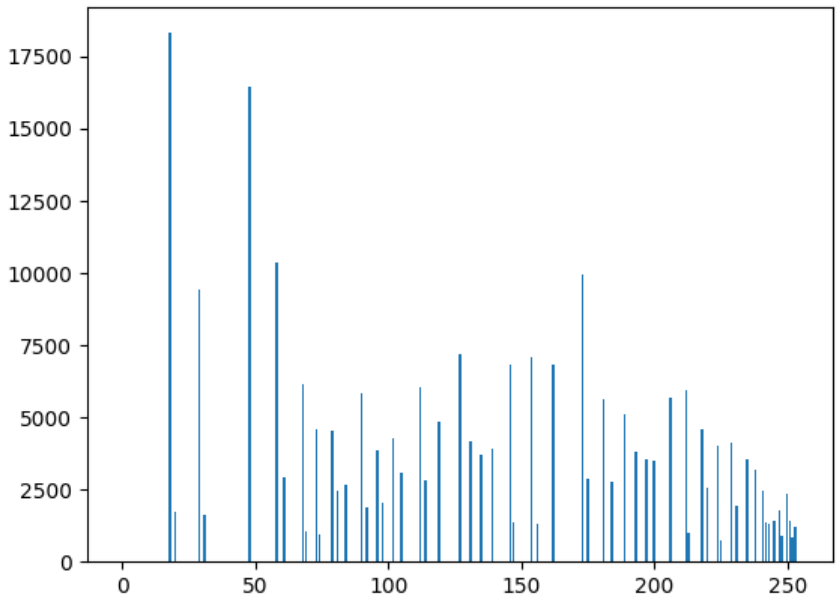
- Imagem: Fig0320(4)(bottom_left).
    - a. Nesta imagem, pelo histograma notamos que há frequência maior de tons escuros, por isso a imagem deve ser escurecida.



    - b. Aqui o que espero é que a imagem fique semelhante às anteriores, já que a equalização vai criar um equilibrio na frequência dos tons, isso é de fato o

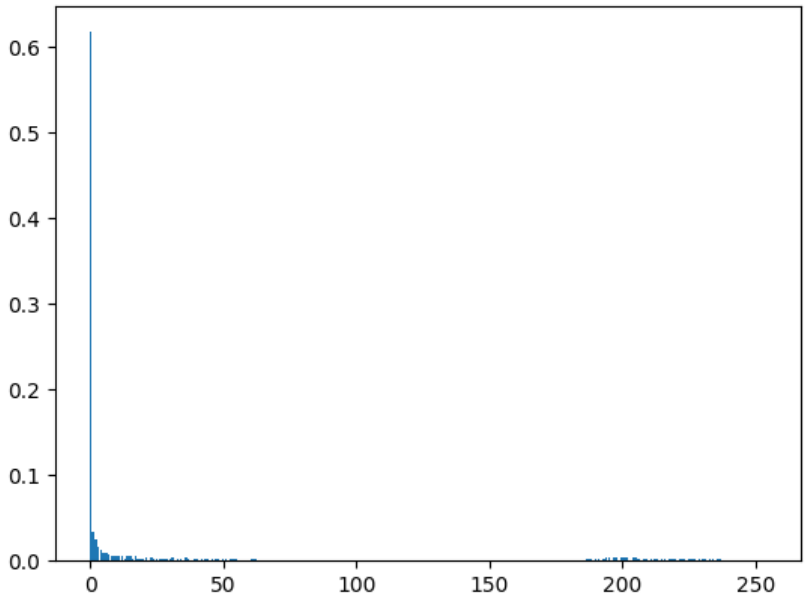
que acontece.

- c. Assim como nas letras anteriores, a equalização deixou a frequência de tons mais equilibrada pelo histograma, não há grande diferenças em relação aos resultados das duas primeiras imagens.

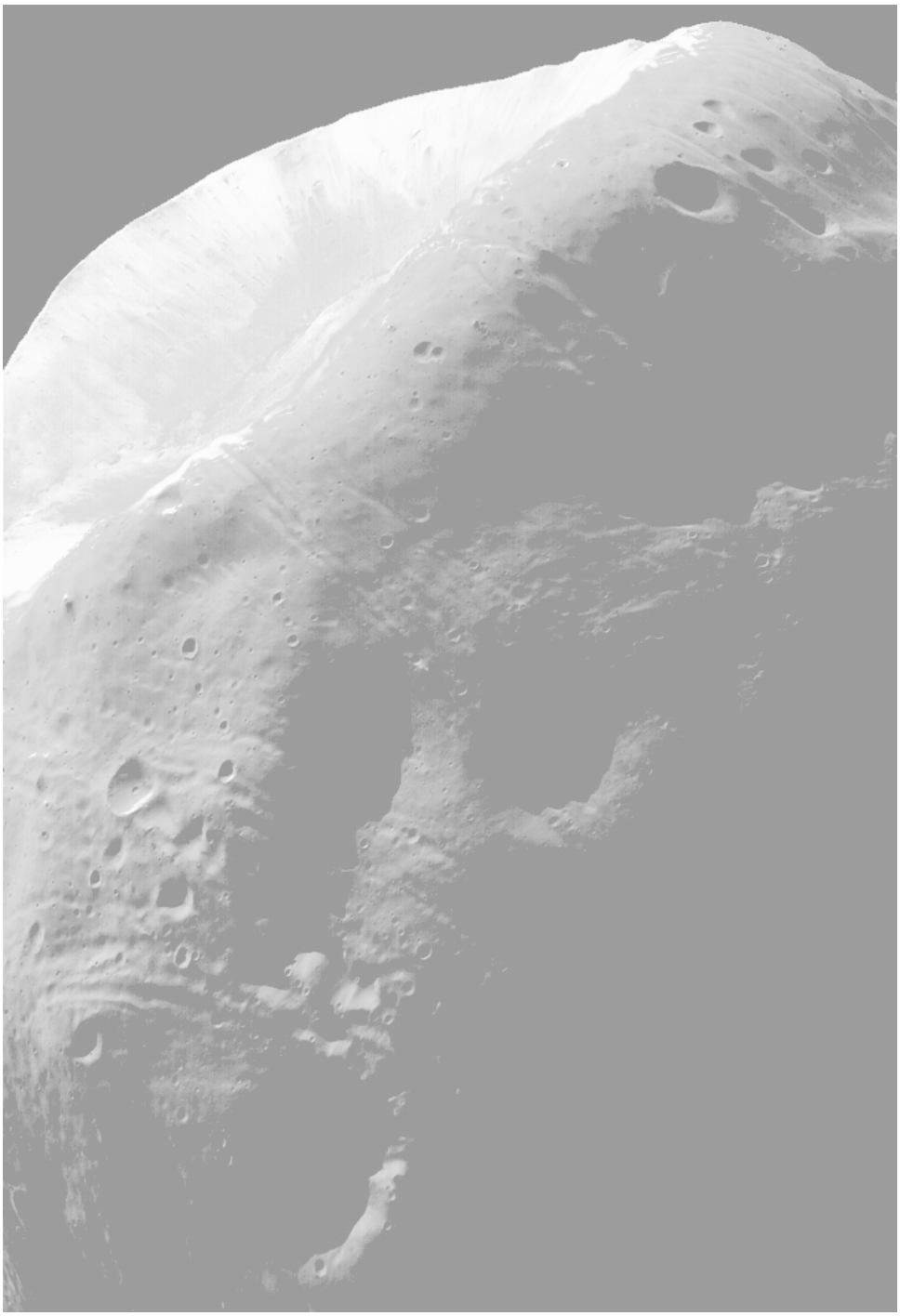


- Imagem: Fig0323(a)(mars_moon_phobos).

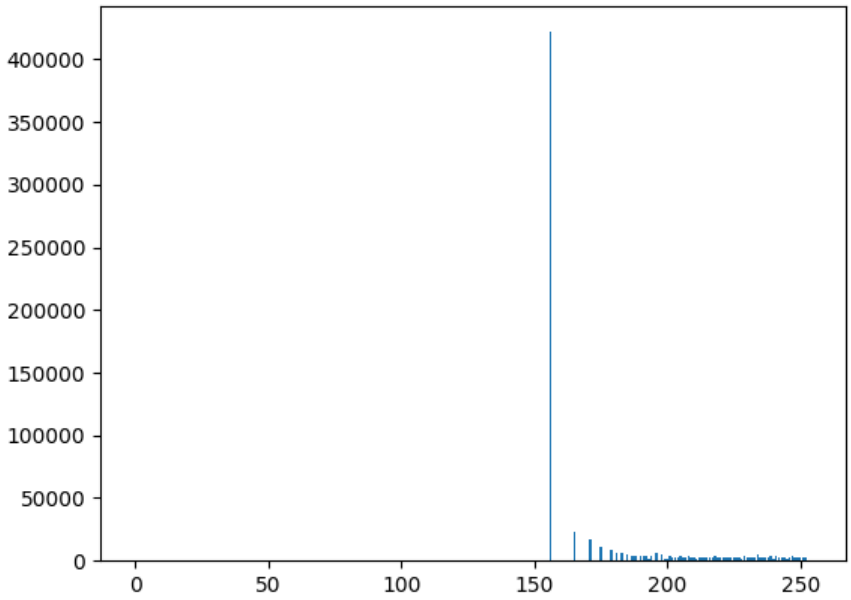
- a. Essa é uma imagem que possui muitos valores de preto por haver um objeto na imagem com fundo preto atrás, esse tipo de imagem não traz histogramas bem equilibrados, o que de fatos podemos notar ao gerar.



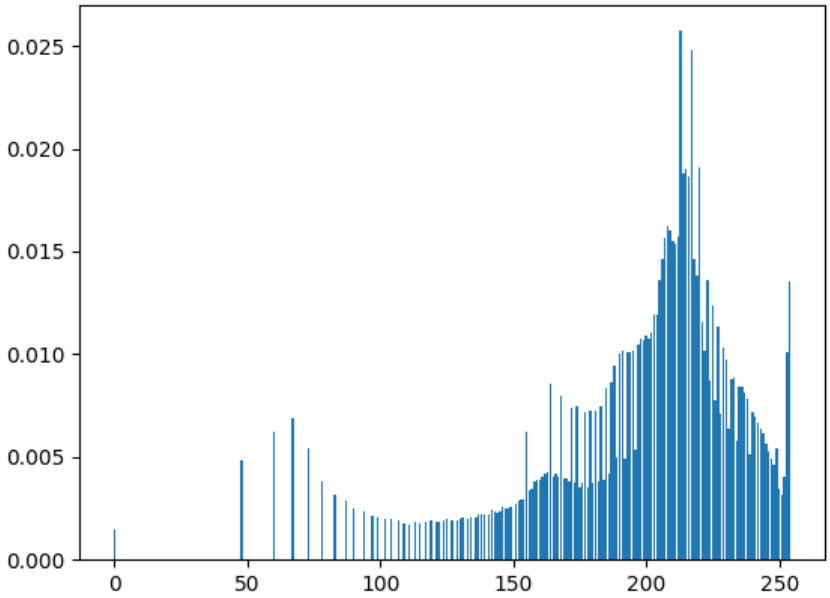
- b. O que espero é que a imagem se torne mais clara de alguma forma, talvez haja saturação, o fundo preto pode ficar mais claro, e de fato isso aocntece, o resultado não é muito agradável, não é uma imagem que a equalização melhore muito o resultado.



- c. Esperava um histograma mais equilibrado, porém fiquei surpreso ao ver que a frequência dos tons mais escuros foram movidos para tons mais claros, mantendo-se semelhante em quantidade.



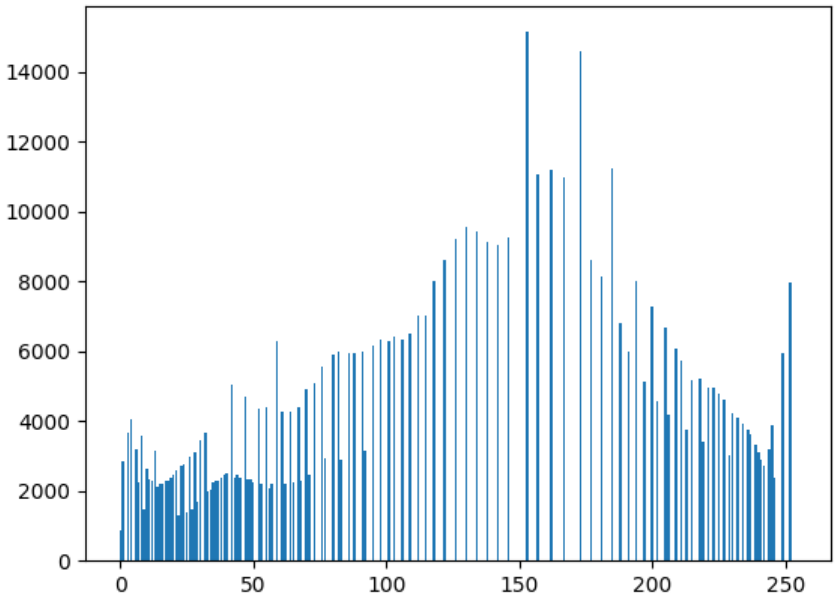
- Imagem: Fig0309(a)(washed_out_aerial_image).
    - a. Essa imagem tem muitos valores claros, o que indica que os valores do histograma estarão mais concentrados em valores maiores, por ser uma imagem esbranquiçada. O resultado de fato é como esperado, contém uma pequena quantidade de valores abaixo de 150, a maioria fica acima de 150 mesmo.



    - b. Acredito que o trabalho de equalização dessa imagem vai ajudar muito assim como nas primeiras imagens do exercício, o que de fato acontece, a

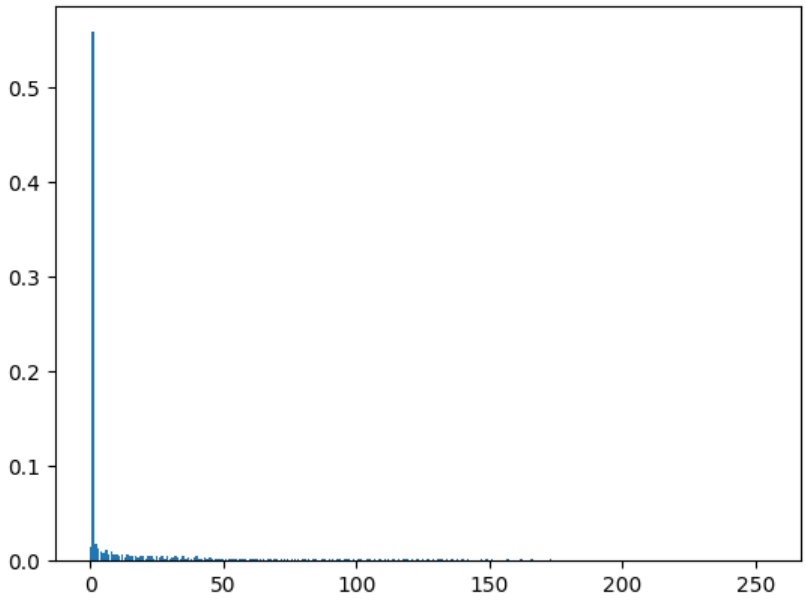
imagem deixa de ser esbranquiçada.

- c. Deve haver um equilibrio na frequência dos valores, e de fato isso acontece, embora seja notável que ainda há frequência muito alta entre 150 e 200.



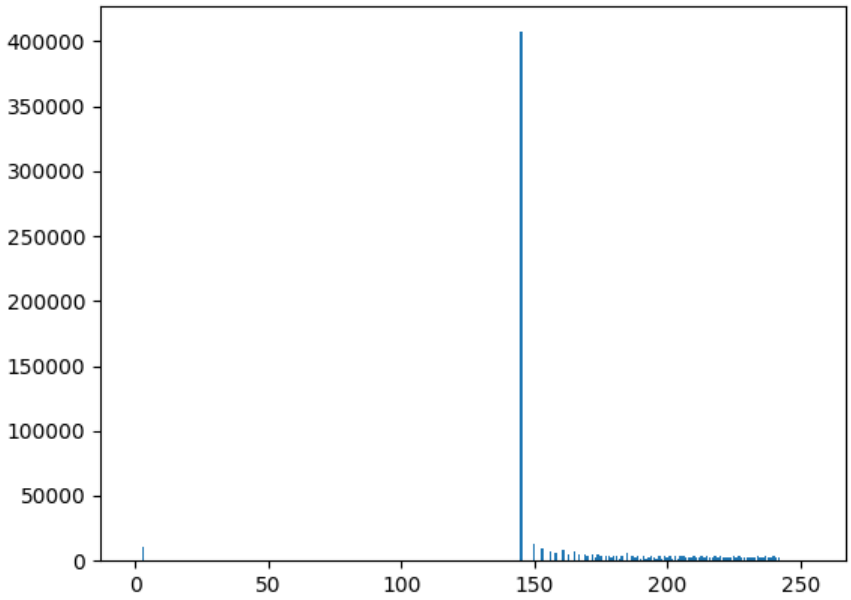
- Imagem: Fig0308(a)(fractured_spine).

- a. Situação semelhante à imagem `Fig0323(a)(mars_moon_phobos)`. Muitos valores de preto, e alguns valores escuros, pouquíssimos claros, o histograma de fato reflete isso.



- b. A imagem é bem semelhante à `Fig0323(a)(mars_moon_phobos)`, o que me leva a acreditar que o resultado deve deixar a imagem cinza nos pontos onde era preto absoluto, o que de fato acontece, o resultado não é agradável.

- c. É uma imagem que assim como a `Fig0323(a)(mars_moon_phobos)` não é ideal para se equalizar devido à quantidade de valores pretos, por isso acredito que semelhante à outra, a frequência vai ficar parecida, mas em valores mais claros. O que de fato acontece.



- Implementação letra a:

```
import numpy as np

from PIL import Image
import matplotlib.pyplot as plt

img_list = [
  'Fig0320(1)(top_left)',
  'Fig0320(2)(2nd_from_top)',
  'Fig0320(3)(third_from_top)',
  'Fig0320(4)(bottom_left)',
  'Fig0323(a)(mars_moon_phobos)',
  'Fig0309(a)(washed_out_aerial_image)',
  'Fig0308(a)(fractured_spine)',
]

def load_image_as_array(path: str):
  img = Image.open(path, 'r')
  img_array = np.asarray(img)
  return img_array

def tons_cinza(img_array) -> dict:
  tons_cinza = {}
  for row in img_array:
    for tom in row:
      if tom not in tons_cinza:
        tons_cinza[tom] = 1
      else:
        tons_cinza[tom] += 1
  return tons_cinza

def hist_data(img_array) -> dict:
```

```
  tons = tons_cinza(img_array)
  M = len(img_array)
  N = len(img_array[0])
  hist_data = {}
  for i in range(0, 255):
    if i in tons:
      hist_data[i] = tons[i] / (M * N)
    else:
      hist_data[i] = 0
  return hist_data

def plot_hist(hist_data, filename: str):
  plt.clf()
  plt.bar(hist_data.keys(), hist_data.values())
  plt.savefig(filename)

def main():
  for img in img_list:
    img_array =
load_image_as_array(f'./static/imagens/{img}.png')
    img_array = hist_data(img_array)
    plot_hist(img_array, f'./static/questao1a_{img}.png')
    print(f'''- Imagem: {img}
![{img}](./static/questao1a_{img}.png)''')

if __name__ == '__main__':
  main()
```

- Implementação letra b:

```
import numpy as np

from PIL import Image
import matplotlib.pyplot as plt

from questao1a import img_list, load_image_as_array, plot_hist,
tons_cinza

def equalizar_img(img):
  tons = tons_cinza(img)
  M = len(img)
  N = len(img[0])
  saida = {}

  tons_order = {}
  for i in range(0, 255):
    if i in tons:
      tons_order[i] = tons[i]
    else:
      tons_order[i] = 0

  new_image = img.copy()
```

```
  for i, _ in enumerate(img):
    for j, _ in enumerate(img[i]):
      soma = 0
      n = list(tons_order.items())[0: img[i][j] + 1]
      for _, qtd in n:
        soma += (len(tons_order) - 1) * (qtd / (M * N))
      new_image[i][j] = soma

  return new_image

def main():
  for img in img_list:
    img_array =
load_image_as_array(f"./static/imagens/{img}.png")
    img_array = equalizar_img(img_array)

Image.fromarray(img_array).save(f"./static/questao1b_{img}.png")
    print(f'''- Imagem: {img}
![{img}](./static/questao1b_{img}.png)''')

if __name__ == '__main__':
  main()
```

- Implementação letra c:

```
import numpy as np

from PIL import Image
import matplotlib.pyplot as plt

from questao1a import img_list, load_image_as_array, plot_hist,
tons_cinza

def equalizar_hist(img):
  tons = tons_cinza(img)
  M = len(img)
  N = len(img[0])
  saida = {}

  tons_order = {}
  for i in range(0, 255):
    if i in tons:
      tons_order[i] = tons[i]
    else:
      tons_order[i] = 0

  for i, (tom, _) in enumerate(tons_order.items()):
    soma = 0
    n = list(tons_order.items())[0: i + 1]
    for tom1, qtd in n:
      soma += (len(tons_order) - 1) * (qtd / (M * N))
    if int(soma) in saida:
```

```
        saida[int(soma)] += qtd
      else:
        saida[int(soma)] = qtd

    saida_order = {}
    for i in range(0, 255):
      if i in saida:
        saida_order[i] = saida[i]
      else:
        saida_order[i] = 0

    return saida_order

def main():
    for img in img_list:
      img_array =
load_image_as_array(f"./static/imagens/{img}.png")
      img_array = equalizar_hist(img_array)
      plot_hist(img_array, f"./static/questao1c_{img}.png")
      print(f'''- Imagem: {img}
![{img}](./static/questao1c_{img}.png)''')

if __name__ == '__main__':
    main()
```