Aplikasi pada Aljabar Linear: Konversi Citra RGB ke Grayscale

Muhammad Dicky Armansyah // 221011054
Alifwan Ananta Putra // 221011029
Firman Imran // 221011103

Metode Lightness

Lightness, mencari nilai tertinggi dan terendah dari nilai R, G, dan B, kemudian hasil penjumlahan nilai tertinggi dan terendah tersebut dikalikan dengan 0,5. Secara matematis dapat dirumuskan:

Grayscale = (max(R, G, B)) + (min(R, G, B)) * 0.5

Metode Average

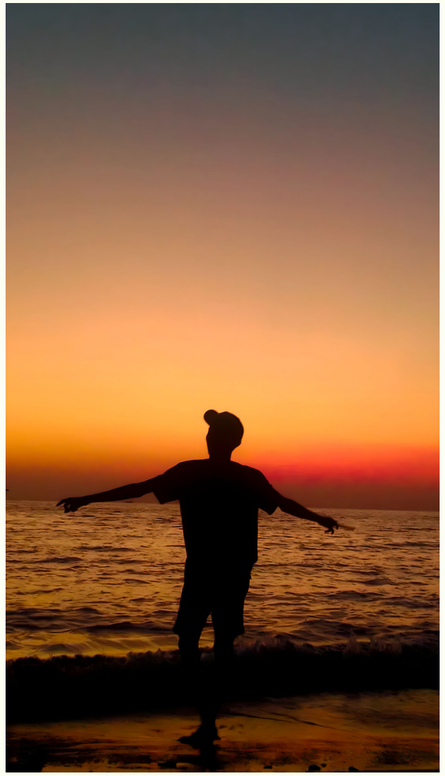
Average, mencari nilai rata-rata dari R, G, dan B. Nilai ratarata itulah yang dapat dikatakan sebagai grayscale. Rumus matematisnya adalah:

Grayscale = (R + G + B) /3

Metode Luminosity

Luminosity, mengalikan setiap nilai R, G, dan B dengan konstanta tertentu yang sudah ditetapkan nilainya, kemudian hasil perkalian seluruh nilai R, G, B dijumlahkan satu sama lain. Rumus matematisnya adalah:

Grayscale = (0.2126 x R) + (0.7152 x G) + (0.0722 x B)
Grayscale = (0.299 x R) + (0.587 x G) + (0.114 x B



Dicky.jpg

Alifwan.jpeg

Castel.JPG

```python
import cv2
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
img_path = 'Dicky.jpg'
img = cv2.imread(img_path)
print(img.shape)
fix_img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
plt.imshow(fix_img)
R, G, B = fix_img[:,:,0], fix_img[:,:,1], fix_img[:,:,2]
print(np.array(fix_img))
```

```
(4096, 2304, 3)
[[[57 67 69]
  [57 67 69]
  [57 67 69]
  ...
  [48 61 69]
  [48 61 69]
  [48 61 69]]

 [[57 67 69]
  [57 67 69]
  [57 67 69]
  ...
  [48 61 69]
  [48 61 69]
  [48 61 69]]

 [[57 67 69]
  [57 67 69]
  [57 67 69]
  ...
  [48 61 69]
  [48 61 69]
  [48 61 69]]

 ...
```

```
[[ 0  0  0]
 [ 0  0  0]
 [ 0  0  0]
 ...
 [ 1  1  1]
 [ 1  1  1]
 [ 1  1  1]]

[[ 0  0  0]
 [ 0  0  0]
 [ 0  0  0]
 ...
 [ 2  1  0]
 [ 2  1  0]
 [ 2  1  0]]

[[ 0  0  0]
 [ 0  0  0]
 [ 0  0  0]
 ...
 [ 2  1  0]
 [ 2  1  0]
 [ 2  1  0]]]
```

Metode Lightness

```
fix_img[:] = np.max(fix_img, axis = -1, keepdims = 1)/2 + np.min(fix_img, axis = -1, keepdims = 1)/2
print(np.array(fix_img[:]))

plt.axis('off')
plt.imshow(fix_img[:])
plt.savefig('metode lightness.jpg', bbox_inches = 'tight')
```

```
[[[63 63 63]
  [63 63 63]
  [63 63 63]
  ...
  [58 58 58]
  [58 58 58]
  [58 58 58]]

 [[63 63 63]
  [63 63 63]
  [63 63 63]
  ...
  [58 58 58]
  [58 58 58]
  [58 58 58]]]
```

```
[[[63 63 63]
  [63 63 63]
  [63 63 63]
  ...
  [58 58 58]
  [58 58 58]
  [58 58 58]]

 [[63 63 63]
  [63 63 63]
  [63 63 63]
  ...
  [58 58 58]
  [58 58 58]
  [58 58 58]]]
```

```
[[[ 0  0  0]
  [ 0  0  0]
  [ 0  0  0]
  ...
  [ 1  1  1]
  [ 1  1  1]
  [ 1  1  1]]

 [[ 0  0  0]
  [ 0  0  0]
  [ 0  0  0]
  ...
  [ 1  1  1]
  [ 1  1  1]
  [ 1  1  1]]]
```

```
[[[ 0  0  0]
  [ 0  0  0]
  [ 0  0  0]
  ...
  [ 1  1  1]
  [ 1  1  1]
  [ 1  1  1]]

 [[ 0  0  0]
  [ 0  0  0]
  [ 0  0  0]
  ...
  [ 1  1  1]
  [ 1  1  1]
  [ 1  1  1]]]
```

Metode Average

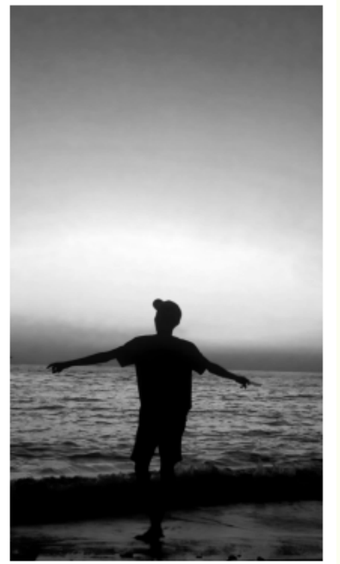
```
gray_img = np.mean(fix_img, axis = 2)
print(np.array(gray_img))

plt.axis('off')
plt.imshow(gray_img, cmap = 'gray')
plt.savefig('Metode Average.jpg' , bbox_inches = 'tight')
```

```
[[64.33333333 64.33333333 64.33333333 ... 59.33333333 59.33333333
  59.33333333]
 [64.33333333 64.33333333 64.33333333 ... 59.33333333 59.33333333
  59.33333333]
 [64.33333333 64.33333333 64.33333333 ... 59.33333333 59.33333333
  59.33333333]
 ...
 [ 0.         0.         0.         ...  1.         1.
   1.        ]
 [ 0.         0.         0.         ...  1.         1.
   1.        ]
 [ 0.         0.         0.         ...  1.         1.
   1.        ]]
```

Metode Luminosity/Weighted Average

```
lumi_img = (0.2126*R) + (0.7152*G) + (0.0722*B)
print(lumi_img)

plt.axis('off')
plt.imshow(gray_img, cmap = 'gray')
plt.savefig('Metode Luminosity.jpg' , bbox_inches = 'tight'
```

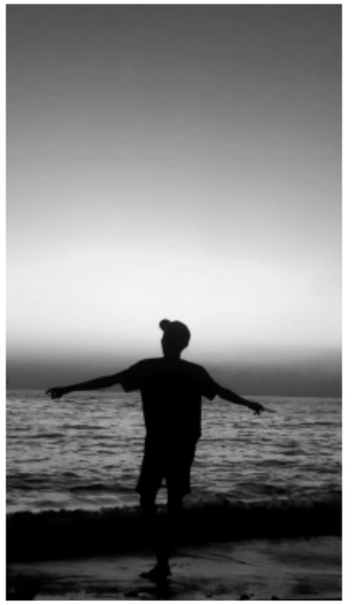
```
[[65.0184 65.0184 65.0184 ... 58.8138 58.8138 58.8138]
 [65.0184 65.0184 65.0184 ... 58.8138 58.8138 58.8138]
 [65.0184 65.0184 65.0184 ... 58.8138 58.8138 58.8138]
 ...
 [ 0.     0.     0.     ...  1.     1.     1.    ]
 [ 0.     0.     0.     ...  1.1404 1.1404 1.1404]
 [ 0.     0.     0.     ...  1.1404 1.1404 1.1404]]
```



Metode Weighted Average/Luminosity

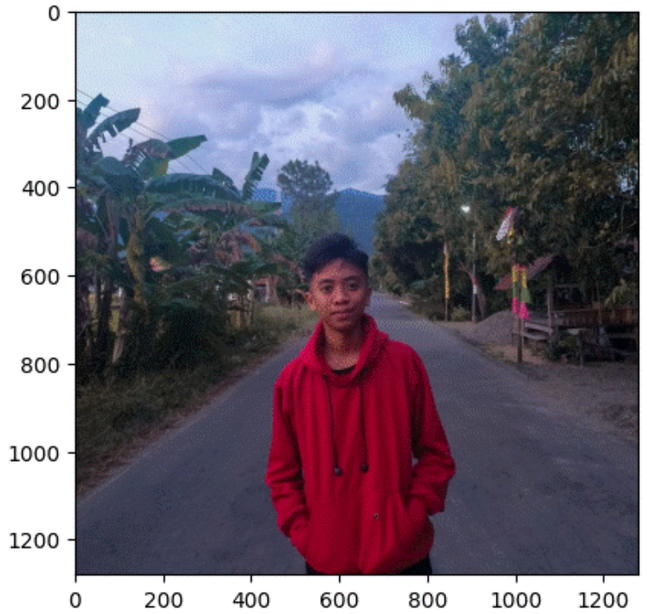
```
wav_img = (0.299*R) + (0.587*G) + (0.114*B)
# print(lumi_img)
print(np.array(wav_img))
plt.axis('off')
plt.imshow(wav_img, cmap = 'gray')
plt.savefig('Metode Weighted Average', bbox_inches='tight')
```

```
[[64.238 64.238 64.238 ... 58.025 58.025 58.025]
 [64.238 64.238 64.238 ... 58.025 58.025 58.025]
 [64.238 64.238 64.238 ... 58.025 58.025 58.025]
 ...
 [ 0.     0.     0.    ...  1.     1.     1.   ]
 [ 0.     0.     0.    ...  1.185  1.185  1.185]
 [ 0.     0.     0.    ...  1.185  1.185  1.185]]
```



```python
import cv2
import numpy as np
import matplotlib.pyplot as plt

img_path = 'alifwan.jpeg'
img = cv2.imread(img_path)
print(img.shape)

plt.imshow(img)

fix_img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
plt.imshow(fix_img)

R, G, B = fix_img[:,:,0], fix_img[:,:,1], fix_img[:,:,2]
print(np.array(fix_img))
```

```
(1280, 1280, 3)
[[[199 216 234]
  [200 217 235]
  [201 218 236]
  ...
  [ 48  59  53]
  [ 47  58  52]
  [ 41  52  46]]

 [[200 217 235]
  [200 217 235]
  [201 218 236]
  ...
  [ 44  55  49]
  [ 38  49  43]
  [ 28  39  33]]

 [[200 217 235]
  [201 218 236]
  [202 219 237]
  ...
  [ 36  45  40]
  [ 29  40  34]
  [ 19  30  24]]

 ...
```



Metode Lightness

```
fix_img = np.max(fix_img,axis = -1, keepdims = 1)/2 + np.min(fix_img,axis = -1, keepdims = 1)/2

plt.axis('off')
plt.imshow(fix_img, cmap= 'gray')
plt.savefig('metode lightness.jpg', bbox_inches = 'tight')
```

```
[[[216 216 216]
  [217 217 217]
  [218 218 218]
  ...
  [ 53  53  53]
  [ 52  52  52]
  [ 46  46  46]]

 [[217 217 217]
  [217 217 217]
  [218 218 218]
  ...
  [ 49  49  49]
  [ 43  43  43]
  [ 33  33  33]]

 [[217 217 217]
  [218 218 218]
  [219 219 219]
  ...
  [ 40  40  40]
  [ 34  34  34]
  [ 24  24  24]]

 ...
```

Metode Average

```
gray_img = np.mean(fix_img, axis = 2)
print(np.array(gray_img))

plt.axis('off')
plt.imshow(gray_img, cmap = 'gray')
plt.savefig('Metode Average.jpg' , bbox_inches = 'tight')
```

```
[[216.33333333 217.33333333 218.33333333 ...  53.33333333  52.33333333
   46.33333333]
 [217.33333333 217.33333333 218.33333333 ...  49.33333333  43.33333333
   33.33333333]
 [217.33333333 218.33333333 219.33333333 ...  40.33333333  34.33333333
   24.33333333]
 ...
 [ 60.66666667  61.66666667  64.33333333 ...  61.33333333  59.33333333
   56.33333333]
 [ 59.66666667  60.66666667  62.33333333 ...  64.33333333  61.33333333
   59.33333333]
 [ 58.66666667  59.66666667  61.33333333 ...  67.33333333  64.33333333
   62.33333333]]
```

Metode Luminosity/Weighted Average

```
lumi_img = (0.2126*R) + (0.7152*G) + (0.0722*B)
print(lumi_img)

plt.axis('off')
plt.imshow(gray_img, cmap = 'gray')
plt.savefig('Metode Luminosity.jpg' , bbox_inches = 'tight')
```

```
[[213.6854 214.6854 215.6854 ...  56.2282  55.2282  49.2282]
 [214.6854 214.6854 215.6854 ...  52.2282  46.2282  36.2282]
 [214.6854 215.6854 216.6854 ...  42.7256  37.2282  27.2282]
 ...
 [ 56.5896  57.5896  59.734  ...  57.8796  55.8796  52.8796]
 [ 55.5896  56.5896  57.734  ...  60.8796  57.8796  55.8796]
 [ 54.5896  55.5896  56.734  ...  63.8796  60.8796  58.8796]]
```

Metode Weighted Average/Luminosity

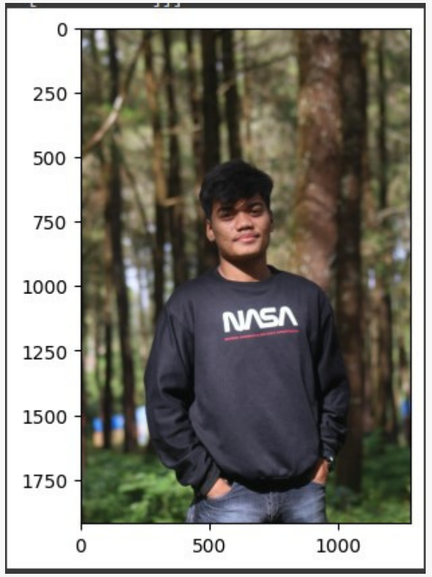
```
wav_img = (0.299*R) + (0.587*G) + (0.114*B)
# print(lumi_img)
print(np.array(wav_img))
plt.axis('off')
plt.imshow(wav_img, cmap = 'gray')
plt.savefig('Metode Weighted Average', bbox_inches='tight')
```

```
[[212.969 213.969 214.969 ...  55.027  54.027  48.027]
 [213.969 213.969 214.969 ...  51.027  45.027  35.027]
 [213.969 214.969 215.969 ...  41.739  36.027  26.027]
 ...
 [ 57.041  58.041  60.269 ...  58.03   56.03   53.03 ]
 [ 56.041  57.041  58.269 ...  61.03   58.03   56.03 ]
 [ 55.041  56.041  57.269 ...  64.03   61.03   59.03 ]]
```

```python
import cv2
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

img_path = 'Castel.JPG'
img = cv2.imread(img_path)
print(img.shape)

fix_img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
plt.imshow(fix_img)

R, G, B = fix_img[:,:,0], fix_img[:,:,1], fix_img[:,:,2]
print(np.array(fix_img))
```

```
(1920, 1280, 3)
[[[104  98  82]
  [107 101  85]
  [107 104  87]
  ...
  [173 167 143]
  [175 169 143]
  [175 171 142]]

 [[105  99  83]
  [108 102  86]
  [108 105  88]
  ...
  [176 170 146]
  [178 172 146]
  [178 174 145]]

 [[106 100  84]
  [109 103  87]
  [111 105  89]
  ...
  [179 173 149]
  [180 174 148]
  [180 176 149]]
```

```
[[ 42  52  28]
 [ 41  51  27]
 [ 40  50  26]
 ...
 [126 165  86]
 [125 164  81]
 [127 165  80]]

[[ 42  52  28]
 [ 41  51  27]
 [ 40  50  26]
 ...
 [124 163  82]
 [125 163  80]
 [126 164  79]]

[[ 42  52  28]
 [ 41  51  27]
 [ 41  51  27]
 ...
 [123 162  81]
 [124 162  77]
 [124 162  75]]]
```



Metode Lightness

```
fix_img[:] = np.max(fix_img, axis = -1, keepdims=1)/2 + np.min(fix_img, axis = -1, keepdims=1)/2

print(np.array(fix_img[:]))

plt.axis('off')
plt.imshow(fix_img[:])
plt.savefig('Metode Lightness', bbox_inches='tight')
```

```
[[[ 94  94  94]
  [ 96  96  96]
  [100 100 100]
  ...
  [156 156 156]
  [160 160 160]
  [160 160 160]]

 [[ 94  94  94]
  [ 97  97  97]
  [100 100 100]
  ...
  [158 158 158]
  [162 162 162]
  [162 162 162]]

 [[ 94  94  94]
  [ 96  96  96]
  [101 101 101]
  ...
  [161 161 161]
  [165 165 165]
  [165 165 165]]

 ...
```

```
[[ 38  38  38]
 [ 38  38  38]
 [ 38  38  38]
 ...
 [124 124 124]
 [124 124 124]
 [123 123 123]]

[[ 38  38  38]
 [ 38  38  38]
 [ 38  38  38]
 ...
 [124 124 124]
 [123 123 123]
 [123 123 123]]

[[ 38  38  38]
 [ 38  38  38]
 [ 38  38  38]
 ...
 [123 123 123]
 [122 122 122]
 [121 121 121]]]
```

Metode Average

```
gray_img = np.mean(fix_img, axis = 2)
print(np.array(gray_img))

plt.axis('off')
plt.imshow(gray_img, cmap = 'gray')
plt.savefig('Metode Average.jpg' , bbox_inches = 'tight')
```

```
[[ 96.66666667  98.66666667 102.         ... 159.         163.33333333
  163.33333333]
 [ 96.66666667  99.66666667 102.         ... 161.         165.33333333
  165.33333333]
 [ 97.33333333  99.33333333 103.         ... 164.         168.33333333
  168.33333333]
 ...
 [ 38.66666667  38.66666667  38.66666667 ... 124.66666667 124.66666667
  124.        ]
 [ 38.66666667  38.66666667  38.66666667 ... 124.66666667 124.
  124.        ]
 [ 38.66666667  38.66666667  38.66666667 ... 123.66666667 123.
  122.66666667]]
```

Metode Luminosity/Weighted Average

```
lumi_img = (0.2126*R) + (0.7152*G) + (0.0722*B)
print(lumi_img)

plt.axis('off')
plt.imshow(gray_img, cmap = 'gray')
plt.savefig('Metode Luminosity.jpg' , bbox_inches = 'tight')
```

```
[[100.3422 102.3422 105.5548 ... 164.5428 169.3984 169.3984]
 [100.3422 103.3422 105.5548 ... 166.5428 171.3984 171.3984]
 [101.913  103.913  106.5548 ... 169.5428 174.3984 174.3984]
 ...
 [ 46.1412  46.1412  46.1412 ... 149.5022 149.5022 149.3578]
 [ 46.1412  46.1412  46.1412 ... 149.5022 149.3578 149.3578]
 [ 46.1412  46.1412  46.1412 ... 148.5022 148.3578 148.9286]]
```

Metode Weighted Average/Luminosity

```
wav_img = (0.299*R) + (0.587*G) + (0.114*B)
# print(lumi_img)
print(np.array(wav_img))
plt.axis('off')
plt.imshow(wav_img, cmap = 'gray')
plt.savefig('Metode Weighted Average', bbox_inches='tight')
```

```
[[ 99.888 101.888 105.187 ... 164.058 168.83  168.83 ]
 [ 99.888 102.888 105.187 ... 166.058 170.83  170.83 ]
 [101.247 103.247 106.187 ... 169.058 173.83  173.83 ]
 ...
 [ 44.274  44.274  44.274 ... 143.045 143.045 142.817]
 [ 44.274  44.274  44.274 ... 143.045 142.817 142.817]
 [ 44.274  44.274  44.274 ... 142.045 141.817 142.176]]
```

Muhammad Dicky Armansyah

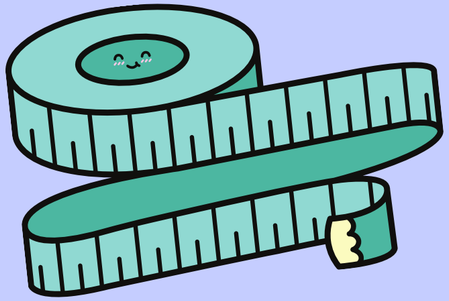
Saya memilih metode Weighted Average/Luminosity
Metode ini memberikan keseimbangan yang baik antara warna-warna yang berbeda, sehingga dapat memberikan representasi yang lebih akurat dari citra asli saya.

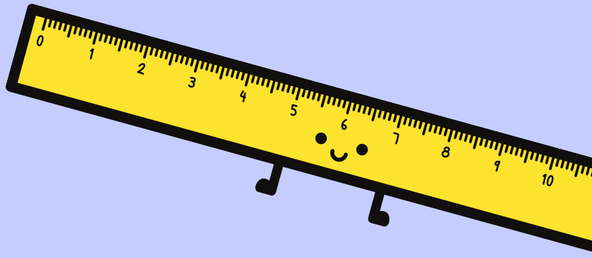
Alifwan Ananta Putra

Saya memilih Metode Lightness kareena memberikan pendekatan yang sederhana dengan mengambil rata-rata dari nilai maksimum dan minimum setiap kanal warna RGB. Hal ini dapat menghasilkan citra grayscale yang cukup seimbang secara visual tanpa memberikan bobot tertentu pada salah satu kanal warna.

Firman Imran

Dari keempat metode tersebut saya memilih metode Lightness karena metode ini menghasilkan citra grayscale yang seimbang.





TERIMA KASIH.