Łukasz Turowski
N1 L_431
45136

1. Implementacja funkcji dla obrazów

1. Funkcja colorFit

```
def colorFit(color, n):
    fit = np.linalg.norm(n - color, axis=1)
    return n[np.argmin(fit)]
```

2. Funkcję realizujące Dithering

- Losowy

```
def randomDithering(image):
    imgcopy = image.copy()
    randomValue = np.random.rand(image.shape[0], image.shape[1])

    for x in range(image.shape[0]):
        for y in range(image.shape[1]):
            imgcopy[x, y] = image[x, y] > randomValue[x, y]

    return imgcopy
```

- Zorganizowany

```
def organizedDithering(image, palette):
    imgcopy = image.copy()
    m2tresholdMap = np.array([[0, 8, 2, 10],
                              [12, 4, 14, 6],5
                              [3, 11, 1, 9],
                              [15, 7, 13, 5]])
    mpre = 1 / 16 * (m2tresholdMap + 1) - 0.5

    for x in range(image.shape[0]):
        for y in range(image.shape[1]):
            imgcopy[x, y] = colorFit(imgcopy[x, y] + mpre[x % 4, y % 4], palette)

    return imgcopy
```

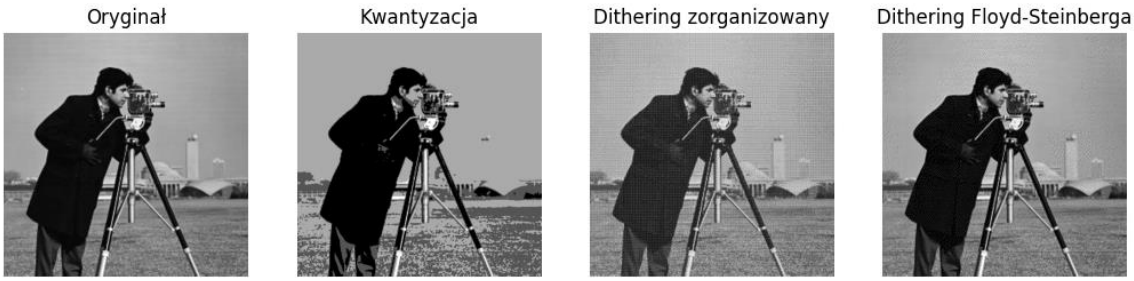
- Floyd-Steinberga

```
def floydSteinbergDithering(image, palette):
    imgcopy = image.copy()

    for x in range(1, image.shape[0] - 1):
        for y in range(1, image.shape[1] - 1):
            oldPixel = imgcopy[x, y].copy()
            newPixel = colorFit(oldPixel, palette)
            imgcopy[x, y] = newPixel
            quant_error = oldPixel - newPixel

            imgcopy[x + 1, y] = max_value(imgcopy[x + 1, y] + quant_error * 7 / 16)
            imgcopy[x - 1, y + 1] = max_value(imgcopy[x - 1, y + 1] + quant_error * 3 / 16)
            imgcopy[x, y + 1] = max_value(imgcopy[x, y + 1] + quant_error * 5 / 16)
```

2. Badanie skuteczności algorytmów na próbkach

1. Skala szarości

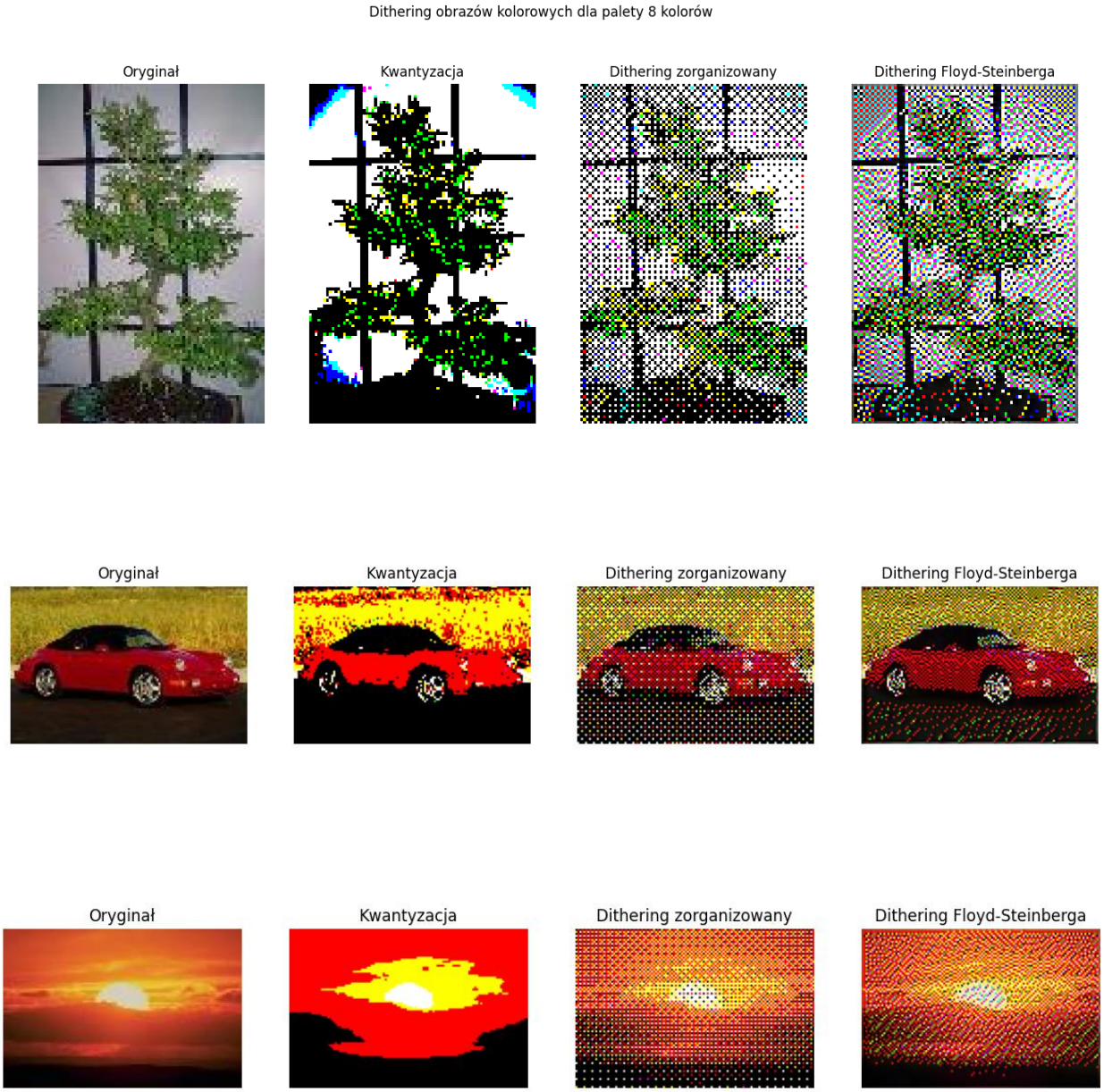
Przy palecie, która wykorzystuje 1 bit, najbliżej oryginału jest algorytm Floyd-Steinberga, najgorzej zaś wypada kwantyzacja oraz metoda losowa.



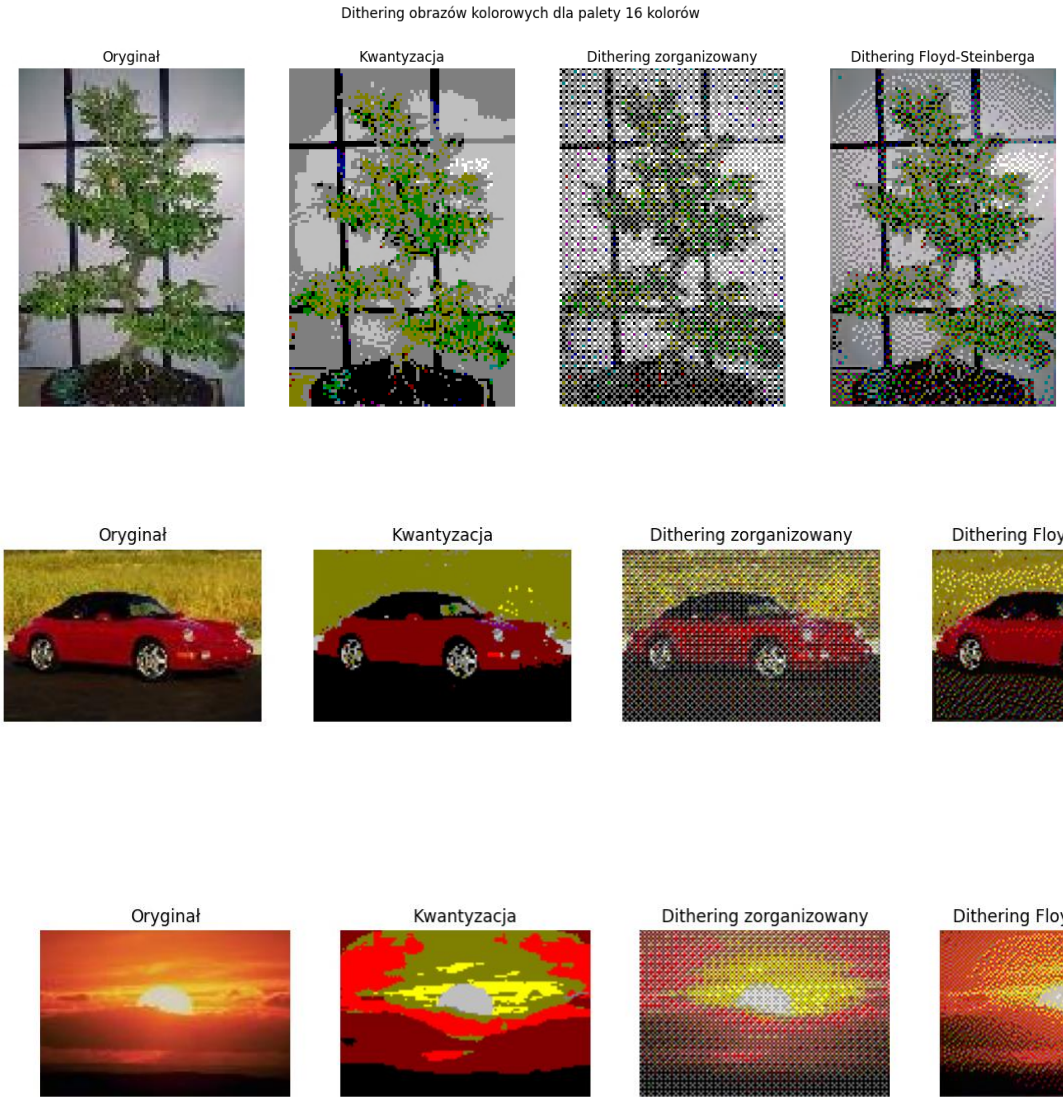
Przy palecie 2 bitu koloru kwantyzacja lepiej wypada niż w poprzednim. Jednak algorytm Floyd-Steiberga dalej jest bliższy oryginału, dithering zorganizowany też jest szczegółowy, ale rozjaśniony

Dla palety 4 bitu kwantyzacja jest mocno zbliżona do oryginału, chociaż troszeczkę są zakłócenia na tle samego zdjęcia.

2. Kolorowe zdjęcia



Przy obu zdjęciach z wykorzystaniem palety 8 kolorów za równo kwantyzacja jak i Dithering wypadają słabo, zdjęcia są mocno uproszczone oraz przy kwantyzacji mamy skrajne odcienie kolorów.



Zarówno kwantyzacja jak i oba algorytmy Ditheringu nie są jakoś podobne do oryginału, przy kwantyzacji mamy uproszczenie kolorów, a przy algorytmach występuje w dużej mierze zarys samego obrazu, bez mocnych szczegółów

3. Kwantyzacja: paleta ręczna vs automatyczna

Do tego zadania wybrałem zdjęcie samochodu, użyłem strony internetowej do sprawdzenia kolorów, żeby jak najbardziej były bliskie oryginałowi. Za równo paleta ręczna oraz automatyczna zostały przedstawione w 16 bitach

Ciężko było dobrać kolory, żeby zachować jak największą szczegółowość samego zdjęcia.

4. Wnioski

Badania przebiegły w mojej ocenie dobrze, zdjęcie w dużej mierze pokrywały się z załączonymi w instrukcji.