

Lab5 - Zmiana wartości pikseli – Metody na obrazie

1. **im.getpixel** i **im.putpixel**
2. **im.load**
3. **im.point**

1. Metoda **im.getpixel(i , j)** pobiera wartość piksela z miejsca o współrzędnych (i , j) obrazu im. Metoda **im.putpixel(i , j), (a, b, c)** wstawia w obraz im w miejscu (i , j) piksel o wartości (a, b, c). Oczywiście format wartości piksela zależy od trybu obrazu. Zaleca się stosować te metody, gdy chcemy dokonać podmiany niewielu pikseli, bo metody działają wolno.
2. Metoda **im.load()** ładuje wszystkie piksele do tablicy tymczasowej, na przykład **pix = im.load()** skutkuje tym, że powstaje tymczasowa tablica **pix** zawierająca wartości pikseli z obrazu im. Adresy pikseli w tej tablicy są identyczne jak adresy w obrazie, **pix[i, j]** jest równoważne **im.getpixel(i , j)**, natomiast **pix[i, j] = (a, b, c)** jest równoważne **im.putpixel(i , j), (a, b, c)**. Zaleca się stosować tę metodę, gdy zmieniamy wiele pikseli różnymi nawet skomplikowanymi funkcjami. Metoda znacznie szybsza niż getpixel i putpixel.
3. Metoda **im.point** jest szybką metodą wykorzystywaną do zastosowania funkcji na wszystkich pikselach (i każdym z osobna). W połączeniu z funkcją lambda redukuje kod „do jednej linijki”.

Na przykład filtr liniowy:

```
im.point(lambda i: i * a + b)
```

działa tak, że wartość każdego piksela i obrazu im zmienia się zgodnie z funkcją $i * a + b$, gdzie a i b są parametrami.

Zadania

Wczytaj pod nazwą **obraz** obrazek wybrany na poprzednich zajęciach. Przekształcenia wykonuj na kopii tego obrazu. Wczytaj obraz czarnobiały obraz z inicjałami (z lab1) pod nazwą **inicjały**.

1. Napisz funkcję **wstaw_inicjały(obraz, inicjały, m, n, kolor)**, gdzie m, n są współrzędnymi punktu, w którym wstawimy w **obraz**, inicjały tak, że czarne piksele obrazu **inicjały** pojawią się w kolorze równym wartości **kolor**. Utwórz i zapisz **obraz1.png**, w którym inicjały w kolorze czerwonym są wstawione tak, że prawy dolny róg obrazu **inicjały** pokrywa się z prawym dolnym rogiem obrazu.
2. Stosując metodę load napisz funkcję **wstaw_inicjały_maska(obraz, inicjały, m, n)**, gdzie m, n są współrzędnymi punktu, w którym traktując **inicjały** jako maskę zmienimy piksele odpowiadające czarnym pikselom z maski zgodnie z transformacją logarytmiczną (zob. 4b). Utwórz i zapisz **obraz2.png**, w którym maska jest wstawiona mniej więcej na środku obrazu.
3. Korzystając z funkcji **rysuj_kwadrat_srednia(obraz, m, n, k)**, (gdzie m,n - środek kwadratu, k - długość boku kwadratu, liczba nieparzysta) napisz funkcję **rysuj_kwadrat_min(obraz, m, n, k)**, która na obrazie **obraz** rysuje kwadrat o środku w punkcie m, n o długości boku k wypełniony kolorem, który powstaje jako wartość **minimalna** (na każdym kanale) wartości pikseli obrazu oryginalnego z tego samego zakresu co rysowany kwadrat. Wybierz 3 różne miejsca i 3 długości k i zastosuj do obrazu tak, żeby były widoczne wszystkie 3 kwadraty. Wynik zapisz jako **obraz3.png**.
4. Stosując metodę point i funkcję lambda:
 - a. Napisz funkcję **kontrast(obraz, wsp_kontrastu)**, która działa tak, że każdy piksel i zmienia wartość zgodnie z funkcją kontrastu

$$f(i) = 128 + (i - 128) * mn, \text{ gdzie } mn = ((255 + \text{wsp_kontrastu}) / 255) ** 2,$$

a `wsp_kontrastu` przyjmuje wartości od 0 do 100

Przetestuj różne wartości `wsp_kontrastu` oraz obraz oryginalny i 3 różne obrazy z testu umieść na diagramie plt (pod nazwą `fig1.png`). Napisz w raporcie jak wartości `wsp_kontrastu` wpływają na uzyskany efekt

- b. Napisz funkcję `transformacja_logarytmiczna(obraz)`, która działa tak, że każdy piksel `i` zmienia wartość zgodnie z funkcją

$$f(i) = 255 * \text{np.log}(1 + i / 255)$$

Obraz oryginalny, obraz uzyskany po zastosowaniu tej funkcji i obraz po zastosowaniu funkcji filtru liniowego dla $a=2$ i $b=100$ umieść na diagramie plt (pod nazwą `fig2.png`). Napisz w raporcie czym różnią się te 3 obrazy.

- c. Napisz funkcję `transformacja_gamma (obraz, gamma)`, która działa tak, że każdy piksel `i` zmienia wartość zgodnie z funkcją

$$f(i) = (i / 255) ** (1 / \text{gamma}) * 255, \text{ gdzie } \text{gamma} \text{ jest współczynnikiem większym od zera, ale może też być zarówno ułamkiem jak i liczbą całkowitą w zakresie do 100.}$$

Podobnie jak w 4a. przetestuj różne wartości `gamma` i napisz w raporcie jak wartości `gamma` wpływają na uzyskany efekt oraz umieść na diagramie plt (pod nazwą `fig3.png`).

5. Napisz, dlaczego obraz powstały z zastosowania poleceń

```
T = np.array(obraz, dtype='uint8')
```

```
T += 100
```

```
obraz_wynik = Image.fromarray(T, "RGB")
```

jest inny niż obraz powstały z zastosowania funkcji `obraz.point(lambda i: i + 100)`

Napisz funkcję, która działa na tablicy obrazu i daje taki sam efekt, co `obraz.point(lambda i: i + 100)`

6. Co się stanie, gdy funkcję `odbij_w_pionie` (podanej na ćwiczeniach) zamienimy na:

```
def odbij_w_pionie(im):
    img = im.copy()
    w, h = im.size
    px = img.load()
    for i in range(w):
        for j in range(h):
            px[i, j] = px[w - 1 - i, j]
    return img
```

Dlaczego tak się dzieje?

Raport, plik z kodem oraz wszystkie obrazy zaznaczone na zielono wstaw na Moodle.