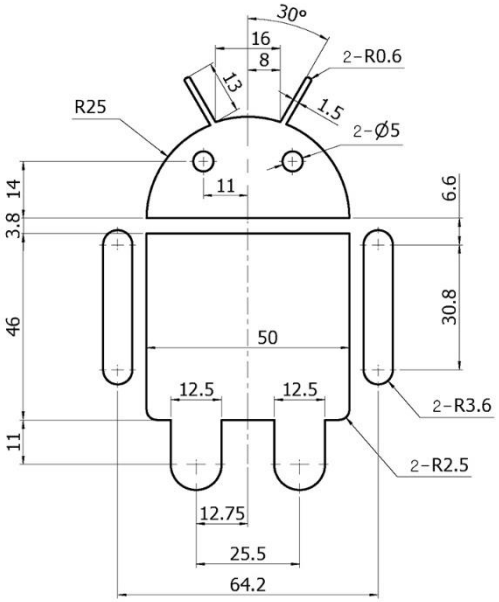
Michał Sprysak

49406

Kompresja bezstratna

1. Obrazy do przeprowadzania analizy skuteczności kompresji

a) Rysunek techniczny (705 x 1005)



b) Wzór dokumentu (1240 x 1754)

Dział VI. Tabela 4. Wprowadzający produkty w opakowaniach[1)]

1. Informacje o rodzaju opakowań, w których wprowadzane są produkty	**Informacje o sposobie wykonania obowiązku zapewnienia odzysku i recyklingu odpadów opakowaniowych**	
	samodzielnie	za pośrednictwem organizacji odzysku opakowań
z tworzyw sztucznych		
z aluminium		
ze stali, w tym z blachy stalowej, oraz z pozostałych metali		
z papieru i tektury		
ze szkła		
z drewna		
z pozostałych opakowań		
wielomateriałowe		za pośrednictwem porozumienia[2)]
środki niebezpieczne w opakowaniach		

2. Dane organizacji odzysku opakowań[3)]			
Nazwa		Numer rejestrowy[4)]	
Województwo		Powiat	
Gmina		Miejscowość	
Kod pocztowy		Ulica	
Nr domu		Nr lokalu	
Data zawarcia umowy z organizacją odzysku opakowań [DD/MM/RRRR]			
Termin obowiązywania umowy z organizacją odzysku opakowań[5)] [DD/MM/RRRR]			

3. Dane organizacji samorządu gospodarczego, z którą zawarto porozumienie			
Nazwa	Data przystąpienia do porozumienia [DD/MM/RRRR]	Termin obowiązywania porozumienia[5)] [DD/MM/RRRR]	Rok, od którego obowiązuje przejęcie obowiązku[6)] przez porozumienie

Objaśnienia:

1) Wypełnić oddzielnie dla każdego rodzaju opakowań, w których wprowadzane są produkty.
2) Dotyczy wprowadzającego produkty w opakowaniach wielomateriałowych lub wprowadzającego środki niebezpieczne w opakowaniach, który przystąpił do porozumienia, o którym mowa w art. 25 ustawy z dnia 13 czerwca 2013 r. o gospodarce opakowaniami i odpadami opakowaniowymi.
3) Wypełnić w przypadku wykonania obowiązku zapewnienia odzysku i recyklingu odpadów za pośrednictwem organizacji odzysku opakowań, na podstawie umowy, o której mowa w art. 17 ust. 4 ustawy z dnia 13 czerwca 2013 r. o gospodarce opakowaniami i odpadami opakowaniowymi.
4) Podać nadany numer rejestrowy, o którym mowa w art. 54 ustawy z dnia 14 grudnia 2012 r. o odpadach (Dz. U. z 2018 r. poz. 992, z późn. zm.).
5) W przypadku braku określonego terminu obowiązywania umowy lub porozumienia należy wpisać 00/00/0000.
6) Dotyczy przejęcia obowiązku w zakresie utworzenia i utrzymania systemu zbierania, transportu, odzysku lub unieszkodliwiania odpadów opakowaniowych powstałych z opakowań wielomateriałowych albo z opakowań po środkach niebezpiecznych.

c) **Kolorowe zdjęcie (1200 x 630)**

2. Kod dokonujący kompresji koder i dekoder

```python
def koder_RLE(img):
    img = img.astype(int)
    x = np.array([len(img.shape)])
    x = np.concatenate([x, img.shape])
    shape = x[1:int(x[0] + 1)]
    retval = []
    cnt = 0
    img = img.flatten()
    for i in range(0, len(img)):
        if (len(img) - 1 != i and img[i] == img[i + 1]):
            cnt += 1
        else:
            retval.append(cnt + 1)
            retval.append(img[i])
            cnt = 0
    return retval, shape


def dekoder_RLE(img):
    img, shape = img
    retval = []
    for i in range(0, len(img), 2):
        retval += [img[i + 1]] * img[i]
    retval = np.reshape(retval, shape)
    return retval
```

```python
def koder_byte_run(img):
    img = img.astype(int)
    x = np.array([len(img.shape)])
    x = np.concatenate([x, img.shape])
    shape = x[1:int(x[0] + 1)]
    img = img.flatten()
    retval = []
    cnt = 0
    # print(f"img len = {len(img)} ")
    for i in range(0, len(img)):
        if (len(img) - 1 != i and img[i] == img[i + 1]):
            cnt += 1
        else:
            if (cnt != 0):
                retval.append(-cnt)
            retval.append(img[i])
            cnt = 0
    return retval, shape


def dekoder_byte_run(img):
    img, shape = img
    retval = []
    for i in range(0, len(img)):
        if (img[i] < 0):
            retval += ((img[i] * -1)) * [img[i+1]]
        else:
            retval.append(img[i])
    retval = np.reshape(retval,shape)
    return retval
```

Wnioski

3.1 Sposób wykorzystania pamięci

W RLE obraz jest spłaszczany do wektora jednowymiarowego, czyli wszystkie wartości są zapisywane w jednej linii. W Każdym kroku pętli liczone są powtarzające się wartości pikseli. Wszystkie wartości zapisywane są w tablicy retval, pierwsza wartość przedstawia ile razy powtarza się dana wartość, a kolejna jaka jest wartość tego piksela i tak dla wszystkich kolejnych wartości. Oryginalny rozmiar obrazu przechowywany jest w tablicy o nazwie shape jako pojedynczy wektor.

Podobną zasadę działania ma Byte Run, w identyczny sposób przechowywany jest oryginalny rozmiar obrazu. Natomiast Byte Run, w pętli sprawdza czy są jakieś wartości, które się powtarzają, jeśli tak to sprawdza ile razy, zapisuje tą wartości jako ujemną w tablicy, a następnie wartość tego piksela. Jeśli brak powtórzeń to do tej samej tablicy przypisuje kolejne wartości.

3.2 Poprawność kodowania na danych testowych

Dane testowe

```
t1 = np.array([1, 1, 1, 2, 3, 4, 4, 1, 2, 3, 4])
t2 = np.array([1, 1, 1, 1, 2, 1, 1, 1, 1, 2, 1, 1, 1, 1])
t3 = np.array([1, 2, 3, 1, 2, 3, 1, 2, 3])
t4 = np.array([5, 1, 5, 1, 5, 5, 1, 1, 5, 5, 1, 1, 5])
t5 = np.array([[4, 4, 2, 1, 6, 6, 6], [5, 1, 2, 2, 3, 3, 9], [6, 3, 1, 7, 7, 3, 2]])
```

Działanie RLE

```
RLE
Koder RLE = ([3, 1, 1, 2, 1, 3, 2, 4, 1, 1, 1, 2, 1, 3, 1, 4], array([11]))
Dekoder RLE = [1 1 1 2 3 4 4 1 2 3 4]
Oryginalne dane = [1 1 1 2 3 4 4 1 2 3 4]

Koder RLE = ([4, 1, 1, 2, 4, 1, 1, 2, 4, 1], array([14]))
Dekoder RLE = [1 1 1 1 2 1 1 1 1 2 1 1 1 1]
Oryginalne dane = [1 1 1 1 2 1 1 1 1 2 1 1 1 1]

Koder RLE = ([1, 1, 1, 2, 1, 3, 1, 1, 1, 2, 1, 3, 1, 1, 1, 2, 1, 3], array([9]))
Dekoder RLE = [1 2 3 1 2 3 1 2 3]
Oryginalne dane = [1 2 3 1 2 3 1 2 3]

Koder RLE = ([1, 5, 1, 1, 1, 5, 1, 1, 2, 5, 2, 1, 2, 5, 2, 1, 1, 5], array([13]))
Dekoder RLE = [5 1 5 1 5 5 1 1 5 5 1 1 5]
Oryginalne dane = [5 1 5 1 5 5 1 1 5 5 1 1 5]

Koder RLE = ([2, 4, 1, 2, 1, 1, 3, 6, 1, 5, 1, 1, 2, 2, 2, 3, 1, 9, 1, 6, 1, 3, 1, 1, 2, 7, 1, 3, 1, 2], a
Dekoder RLE = [[4 4 2 1 6 6 6]
 [5 1 2 2 3 3 9]
 [6 3 1 7 7 3 2]]
Oryginalne dane = [[4 4 2 1 6 6 6]
 [5 1 2 2 3 3 9]
 [6 3 1 7 7 3 2]]
```

Działanie Byte Run

```
Byte Run
Koder Byte Run = ([-2, 1, 2, 3, -1, 4, 1, 2, 3, 4], array([11]))
Dekoder Byte Run = [1 1 1 2 3 4 4 1 2 3 4]
Oryginalne dane = [1 1 1 2 3 4 4 1 2 3 4]

Koder Byte Run = ([-3, 1, 2, -3, 1, 2, -3, 1], array([14]))
Dekoder Byte Run = [1 1 1 1 2 1 1 1 1 2 1 1 1 1]
Oryginalne dane = [1 1 1 1 2 1 1 1 1 2 1 1 1 1]

Koder Byte Run = ([1, 2, 3, 1, 2, 3, 1, 2, 3], array([9]))
Dekoder Byte Run = [1 2 3 1 2 3 1 2 3]
Oryginalne dane = [1 2 3 1 2 3 1 2 3]

Koder Byte Run = ([5, 1, 5, 1, -1, 5, -1, 1, -1, 5, -1, 1, 5], array([13]))
Dekoder Byte Run = [5 1 5 1 5 5 1 1 5 5 1 1 5]
Oryginalne dane = [5 1 5 1 5 5 1 1 5 5 1 1 5]

Koder Byte Run = ([-1, 4, 2, 1, -2, 6, 5, 1, -1, 2, -1, 3, 9, 6, 3, 1, -1, 7, 3, 2], array([3, 7]))
Dekoder Byte Run = [[4 4 2 1 6 6 6]
 [5 1 2 2 3 3 9]
 [6 3 1 7 7 3 2]]
Oryginalne dane = [[4 4 2 1 6 6 6]
 [5 1 2 2 3 3 9]
 [6 3 1 7 7 3 2]]
```

3.3 Wyniki dla zdjęć

```
    RLE
Typ zdjęcia: Rysunek techniczny
Oryginalny rozmiar: 8502300
Rozmiar po kompresji = 2126180
Rozmiar po dekompresji = 8502300
Stopień kompresji: 74.99%
Zgodność: True

    Byte Run
Typ zdjęcia: Rysunek techniczny
Oryginalny rozmiar: 8502300
Rozmiar po kompresji = 2351104
Rozmiar po dekompresji = 8502300
Stopień kompresji: 72.35%
Zgodność: True
```

```
    RLE
Typ zdjęcia: Dokument
Oryginalny rozmiar: 26099520
Rozmiar po kompresji = 24781108
Rozmiar po dekompresji = 26099520
Stopień kompresji: 5.05%
Zgodność: True

    Byte Run
Typ zdjęcia: Dokument
Oryginalny rozmiar: 26099520
Rozmiar po kompresji = 25312100
Rozmiar po dekompresji = 26099520
Stopień kompresji: 3.02%
Zgodność: True
```

```
    RLE
Typ zdjęcia: Kolorowe zdjęcie
Oryginalny rozmiar: 9072000
Rozmiar po kompresji = 102132976
Rozmiar po dekompresji = 9072000
Stopień kompresji: -1025.80%
Zgodność: True

    Byte Run
Typ zdjęcia: Kolorowe zdjęcie
Oryginalny rozmiar: 9072000
Rozmiar po kompresji = 82337696
Rozmiar po dekompresji = 9072000
Stopień kompresji: -807.60%
Zgodność: True
```