Міністерство освіти і науки України

НТУУ «Київський політехнічний інститут ім. Ігоря Сікорського»

Фізико-технічний інститут

Лабораторна робота №1

З дисципліни

«Автоматизація обробки ІзОД»

Варіант 9

**Виконав:**

Студент 5 курсу ФТІ

групи ФЕ-91мп

Проценко Д.О.

**Перевірив:**

Прогонов Д. О.

Київ-2020Завдання на лабораторну роботу:

1. Сформувати тестову вибірку зображень з вихідного пакета;
2. Для кожного каналу кольору кожного зображення з тестового пакета обчислити наступні характеристики:
   1. Максимальна / мінімальне значення;
   2. Математичне сподівання і дисперсію;
   3. Медіану значень, інтерквартільний розмах;
   4. Коефіцієнти асиметрії та ексцесу (нормалізований);
3. Для кожного каналу кольору кожного зображення з тестового пакета побудувати гістограму значень яскравості пікселів;
4. Провести апроксимацію отриманих гістограм з використанням відомих імовірнісних розподілів, визначити найкращу апроксимацію;
5. Побудувати розподіл типів використаних імовірнісних розподілів для яких досягається мінімальне значення помилки апроксимації з п.4.

**Хід роботи**

**1) Формування тестової вибірки зображеннь**

Лабораторна робота буде виконана мовою Python за допомогою блокового інтерпритатора Jupyter. В данній лабораторній роботі використовувались наступні бібліотеки:

* Scipy
* Matplotlib
* Os
* Numpy
* Pandas

Тестова вибірка сформована із псевдовипадкового вибору 250 кольорових зображень із пакету mirflick.

Приклад коду для вибірки зображень:

*# load all images in a directory*

random.seed(9)

random\_indexes = random.sample(range(25000), 250)

loaded\_images = list()

**for** i **in** range(250):

filename = 'im' + str(random\_indexes[i]) + '.jpg'

img\_data = image.imread('mirflickr25k/mirflickr/' + filename)

loaded\_images.append(img\_data)

print('> loaded **%s** **%s**' % (filename, img\_data.shape))

Тепер отриманий масив зображень буде знаходитись в loaded\_images в виді двомірного масиву з трьома значеннями яскравості в кожрній комірці.

Далі сформуємо матрицю для збору статистичних даних про зобьраження. Для цього створимо двомірний Numpy масив на три рядки для кожного каналу кольору та на 256 стовпчиків, що відповідатиме кількості пікселів відповідної яскравості.

values = np.zeros((3, 256))

index = 0

**for** image **in** loaded\_images:

**for** i **in** range(image.shape[0]):

**for** j **in** range(image.shape[1]):

values[0][image[i][j][0]] += 1

values[1][image[i][j][1]] += 1

values[2][image[i][j][2]] += 1

index += 1

**if** (index % 10) == 0:

print('> processed **%i** files' % (index))

**2) Обчислення характеристик для кожного каналу зображень**

1. Максимальна / мінімальне значення

Після виконання коду отимаємо наступні значення :

RED:

Max:255, Min: 0

BLUE:

Max:255, Min: 0

GREEN:

Max:255, Min: 0

1. Математичне сподівання і дисперсія

Розрахунки будуть проводитись за наступними формулами:



(1),



(2),

Де (1) - математичне очікування, а (2) – дисперсія, xi - значення яскравості, pi – ймовірність її появи. pi можна знайти як кількість пікселів даної яскравості поділену на всю кількість пікселів

*#GREEN*

sum\_value = sum(values[GREEN])

M\_green = 0

**for** index **in** range(len(values[GREEN])):

p = (values[GREEN][index] / sum\_value)

M\_green += p \* index

D\_green = 0

**for** index **in** range(len(values[GREEN])):

p = (values[GREEN][index] / sum\_value)

D\_green += p \* ((index - M\_green) \*\* 2)

print("Green: expected value (math. ochikuvanya) : **{0:.2f}**, dispersion : **{1:.2f}**"\

.format(M\_green, D\_green))

Таким чином отримуэмо значення Мат. очікування та дисперсії:

Red: expected value (math. ochikuvanya) : 110.96, dispersion : 5971.28

Green: expected value (math. ochikuvanya) : 101.85, dispersion : 5362.33

Blue: expected value (math. ochikuvanya) : 90.83, dispersion : 5764.33

с. Медіана значень та інтерквартальний розмах.

Для медіани будемо сумувати всі кількості пікселів в масиві і коли сума перевалить за половину кількості пікселів – ми будемо знати що це і є медіана

*#Median and interquartal*

**def** medianSearcher(array\_values):

sum\_val = sum(array\_values)

summa = 0

**for** index **in** range(len(array\_values)):

summa += array\_values[index]

**if** (summa > (sum\_val / 2)):

**return** index - 1

**if** (summa == (sum\_val / 2)):

**return** (2 \* index + 1) / 2

Для інтерквартального розмаху працюємо аналогічно, але будемо шукати вже сумарну імовірність пройдених пікселів.

**def** quartSearcher (array\_values, quart):

sum\_val = sum(array\_values)

summa = 0

**for** index **in** range(len(array\_values)):

prev\_sum = summa

p = array\_values[index] / sum\_val

summa += p

**if** (summa > quart):

**if** (quart - prev\_sum < summa - quart):

**return** index - 1

**elif** (quart - prev\_sum > summa - quart):

**return** index

**else**:

**return** (2 \* index - 1) / 2

Таким чином отримаємо :

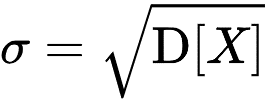
RED: median - 105, interquartal - 131

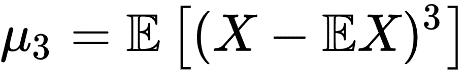
GREEN: median - 94, interquartal - 120

BLUE: median - 73, interquartal – 124

d. Коефіцієнти асиметрії та ексцесу

Використаємо відповідні формули коефіцієнту асиметрії:





**def** E\_operator(arr\_values, M, power):

sum\_val = sum(arr\_values)

ans = 0

**for** index **in** range(len(arr\_values)):

p = (arr\_values[index] / sum\_val)

ans += p \* ((index - M) \*\* power)

**return** ans

*#asymmetry, ekscess*

Asym\_red = E\_operator(values[RED], M\_red, 3) / (D\_red \*\* (3 / 2))

Asym\_green = E\_operator(values[GREEN], M\_green, 3) / (D\_green \*\* (3 / 2))

Asym\_blue = E\_operator(values[BLUE], M\_blue, 3) / (D\_blue \*\* (3 / 2))

Ekscess\_red = E\_operator(values[RED], M\_red, 4) / (D\_red \*\* 2)

Ekscess\_green = E\_operator(values[GREEN], M\_green, 4) / (D\_green \*\* 2)

Ekscess\_blue = E\_operator(values[BLUE], M\_blue, 4) / (D\_blue \*\* 2)

print('Red: asymmetry - **{0:.3f}**, ekscess - **{1:.3f}**'\

.format(Asym\_red, Ekscess\_red))

print('Green: asymmetry - **{0:.3f}**, ekscess - **{1:.3f}**'\

.format(Asym\_green, Ekscess\_green))

print('Blue: asymmetry - **{0:.3f}**, ekscess - **{1:.3f}**'\

.format(Asym\_blue, Ekscess\_blue))

Отримаємо:

RED: asymmetry - 0.213, ekscess - 1.859

GREEN: asymmetry - 0.343, ekscess - 2.033

BLUE: asymmetry - 0.582, ekscess - 2.160

1. **Для кожного каналу кольору кожного зображення з тестового пакета побудувати гістограму значень яскравості пікселів**

Використаємо бібліотеку matplotlib , таким чином отримаємо наступні значення:

**import** **matplotlib.pyplot** **as** **plt**

x = range(256)

*#RED*

plt.bar(x,values[RED],color='red')

plt.xlabel('Brightness')

plt.ylabel('Frequency')

plt.ylim(top=1500000)

plt.show()

*#GREEN*

plt.bar(x,values[GREEN],color='green')

plt.xlabel('Brightness')

plt.ylabel('Frequency')

plt.ylim(top=1500000)

plt.show()

*#BLUE*

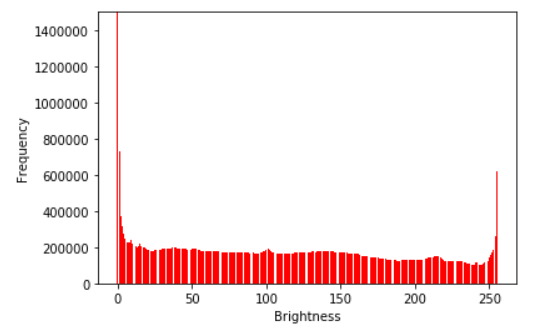
plt.bar(x,values[BLUE],color='blue')

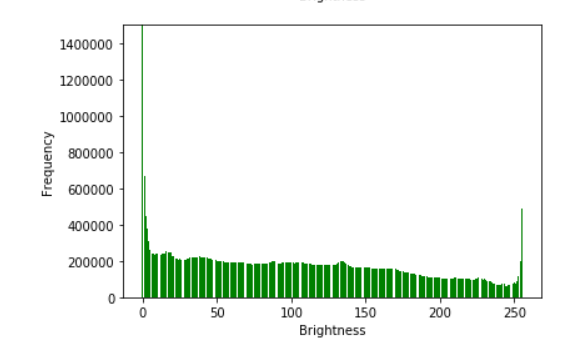
plt.xlabel('Brightness')

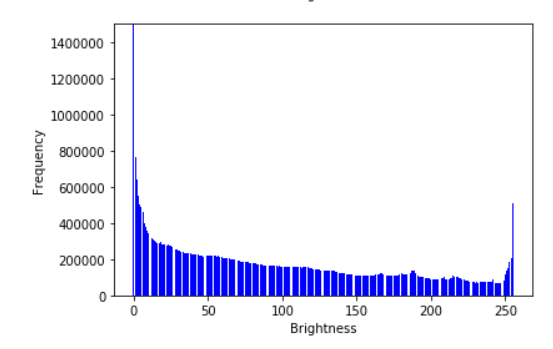
plt.ylabel('Frequency')

plt.ylim(top=1500000)

plt.show()







**ВИСНОВКИ**

У данній лабораторній роботі було розраховано значення статистичних характеристик вибірки 250 зображення для кожного каналу кольорів. Було знайдено що всі канали охоплюють увесь спектр значень.

Розраховано мат.очікування для кожного кольору:

Red: expected value (math. ochikuvanya) : 110.96, dispersion : 5971.28

Green: expected value (math. ochikuvanya) : 101.85, dispersion : 5362.33

Blue: expected value (math. ochikuvanya) : 90.83, dispersion : 5764.33

Медіану значень та інтерквартальний розподіл:

RED: median - 105, interquartal - 131

GREEN: median - 94, interquartal - 120

BLUE: median - 73, interquartal – 124

Також було знайдено коефіцієнти асиметрії та коефіцієнт ексцесу:

RED: asymmetry - 0.213, ekscess - 1.859

GREEN: asymmetry - 0.343, ekscess - 2.033

BLUE: asymmetry - 0.582, ekscess - 2.160

Так, як коефіцієнт асиметрії додатній, то напрямок грайіка буде в бік спадання значень.

Графіки приведено за допомогою бібліотеки matplotlib, вони повністю відповідають знайденим значенням.