МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Кафедра инфокоммуникаций

Отчет по лабораторной работе № 3.8

«Процессы дискретизации и квантования изображения»

по дисциплине «Технологии распознавания образов»

Выполнил студент группы
ПИЖ-б-о-21-1
Зиберов Александр
« » апреля 2023 г.
Подпись студента_____
Работа защищена
« »_____20__г.
Проверил Воронкин Р.А. _____
(подпись)

Ставрополь, 2023

Цель работы:

Изучение функций, использующихся для моделирования процессов квантования и дискретизации изображения на языке Python.

Выполнение работы:

Проработать примеры лабораторной работы в отдельном ноутбуке.

In [1]:
```
import cv2
import numpy as np
from matplotlib import pyplot as plt
```

Задание 1.

Выбрать значение шага дискретизации в пределах от 5 до 15. Продискретизировать с этим шагом дискретизации изображение и вывести его на экран.

In [2]:
```
image = cv2.imread('pictures/ab.jpg')
image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
img = image.copy()

K = 15 # размер шага
s = img.shape

h1, w1 = s[0], s[1]
h = (s[0] - s[0] % K)
w = (s[1] - s[1] % K)

img = cv2.resize(img, (w, h))

for y in range(0, h-1, K):
    for x in range(0, w-1, K):
        if len(s) > 2:
            s = np.average(img[y:(y + K), x:(x + K)], axis=0)
            img[y:(y + K), x:(x + K)] = np.average(s, axis=0)
        else:
            s = img[y:(y+K), x:(x+K)]
            img[y:(y+K), x:(x+K)] = np.average(s)

img = cv2.resize(img, (w1, h1))
res = np.hstack((image, img))
plt.figure(figsize=(10, 20))
plt.axis("off")
plt.imshow(res);
```

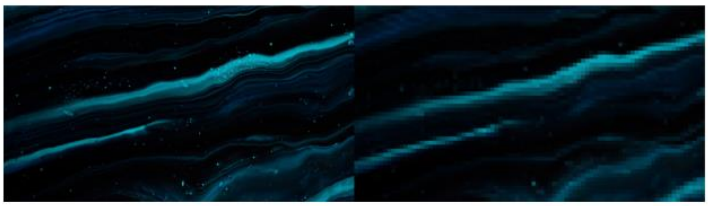

Рисунок 1 – Пример 1

Задание 2.2.

Проквантовать изображение, сократив число градаций до 4

In [3]:
```
img = cv2.imread('pictures/ab.jpg')
img = cv2.cvtColor(img,cv2.COLOR_BGR2RGB)

Z = img.reshape((-1, 3))
Z = np.float32(Z)

crt = (cv2.TERM_CRITERIA_EPS + cv2.TERM_CRITERIA_MAX_ITER, 10, 1.0)

k = 4
ret, label, center = cv2.kmeans(Z, k, None, crt, 10, cv2.KMEANS_RANDOM_CENTERS)

center = np.uint8(center)
res = center[label.flatten()]
res2 = res.reshape((img.shape))

result = np.hstack((img, res2))

plt.figure(figsize=(10, 20))
plt.axis("off")
plt.imshow(result);
```

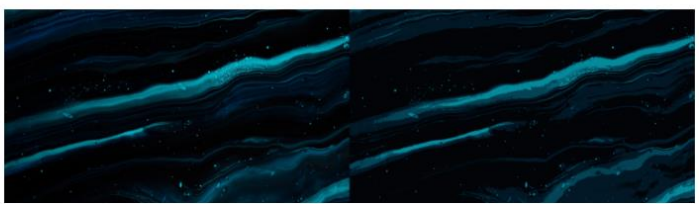

Рисунок 2 – Пример 2

Индивидуальное задание

Задание

Уменьшить количество цветов в изображении при помощи квантования методами:
а) Уменьшение градаций серого
б) Дизеринг Флойда-Стейнберга

In [1]:
```
import cv2
import numpy as np
from matplotlib import pyplot as plt
%matplotlib inline

def imgshow(image, conversion=cv2.COLOR_BGR2RGB):
    image = cv2.cvtColor(image, conversion)
    plt.imshow(image)
    plt.xticks([])
    plt.yticks([])
    plt.axis("off")
    plt.rcParams['figure.dpi'] = 400
    plt.show();

img = cv2.cvtColor(cv2.imread('pictures/bliss.jpg'), cv2.COLOR_BGR2RGB)
```

а) Уменьшение градаций серого квантованием

Процесс разбиения непрерывного динамического диапазона значений яркости на ряд дискретных уровней называется квантованием. Число уровней квантования равно $K = [A/\Delta A]$, где A определяет диапазон значений яркостей функции f (x, y), ΔA – величина кванта, для удобства полагаем, что ее значение равно единице.

In [2]:
```
img = cv2.cvtColor(img,cv2.COLOR_BGR2RGB)

Z = img.reshape((-1, 3))
Z = np.float32(Z)

crt = (cv2.TERM_CRITERIA_EPS + cv2.TERM_CRITERIA_MAX_ITER, 10, 1.0)

k = 4
ret, label, center = cv2.kmeans(Z, k, None, crt, 10, cv2.KMEANS_RANDOM_CENTERS)

center = np.uint8(center)
res = center[label.flatten()]
res2 = res.reshape((img.shape))

result = np.hstack((img, res2))
imgshow(result)
```

Рисунок 3 – Индивидуальное задание (1)

б) Алгоритм дизеринга Флойда-Стейнберга

Дизеринг (сглаживание) Флойда-Стейнберга - это метод уменьшения цветовой палитры изображения (например, для уменьшения размера его файла) при сохранении как можно большей воспринимаемой детализации. Для каждого пикселя в исходном изображении из ограниченной палитры выбирается ближайший к этому пикселю цвет, и любая "ошибка" (разница в значении цвета пикселя, оригинал - новый) распределяется по соседним пикселям следующим образом:

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & * & \frac{7}{16} \\ \frac{3}{16} & \frac{5}{16} & \frac{1}{16} \end{bmatrix}$$

, где * - рассматриваемый пиксель, а ошибка = фактическое значение пикселя - значение ближайшего цвета в палитре.

In [3]:
```
# Алгоритм Floyd-Steinberg
def dither(img):
    output = np.copy(img)
    output = output.astype('int')
    height, width = img.shape[0], img.shape[1]
    # Идем по пикселям (длина, ширина)
    for y in range(1, height-1):
        for x in range(1, width-1):
            # Рассматриваемый пиксель
            pixel = output[y][x]
            new_pixel = round(pixel/255) * 255
            # Считаем ошибку
            error = pixel - new_pixel
            output[y][x] = new_pixel
            # Заполняем соседние пиксели
            output[y    ][x + 1] += error * 7/16
            output[y + 1][x - 1] += error * 3/16
            output[y + 1][x    ] += error * 5/16
            output[y + 1][x + 1] += error * 1/16
    output = output.astype(dtype=np.uint8)
    return(output)

# Копируем изображение
img_original = img.copy()

# Отдельно берем каждый канал и прогоняем через алгоритм
blue = img[:,:,0]
blue = dither(blue)
green = img[:,:,1]
green = dither(green)
red = img[:,:,2]
red = dither(red)

# Совмещаем все каналы в одно изображение
img2 = cv2.merge((blue, green, red))

res = np.hstack((img_original, img2))
cv2.imwrite('pictures/dithered.jpg', img2)
imgshow(res)
imgshow(img_original)
imgshow(img2)
```

Рисунок 4 – Индивидуальное задание (2)

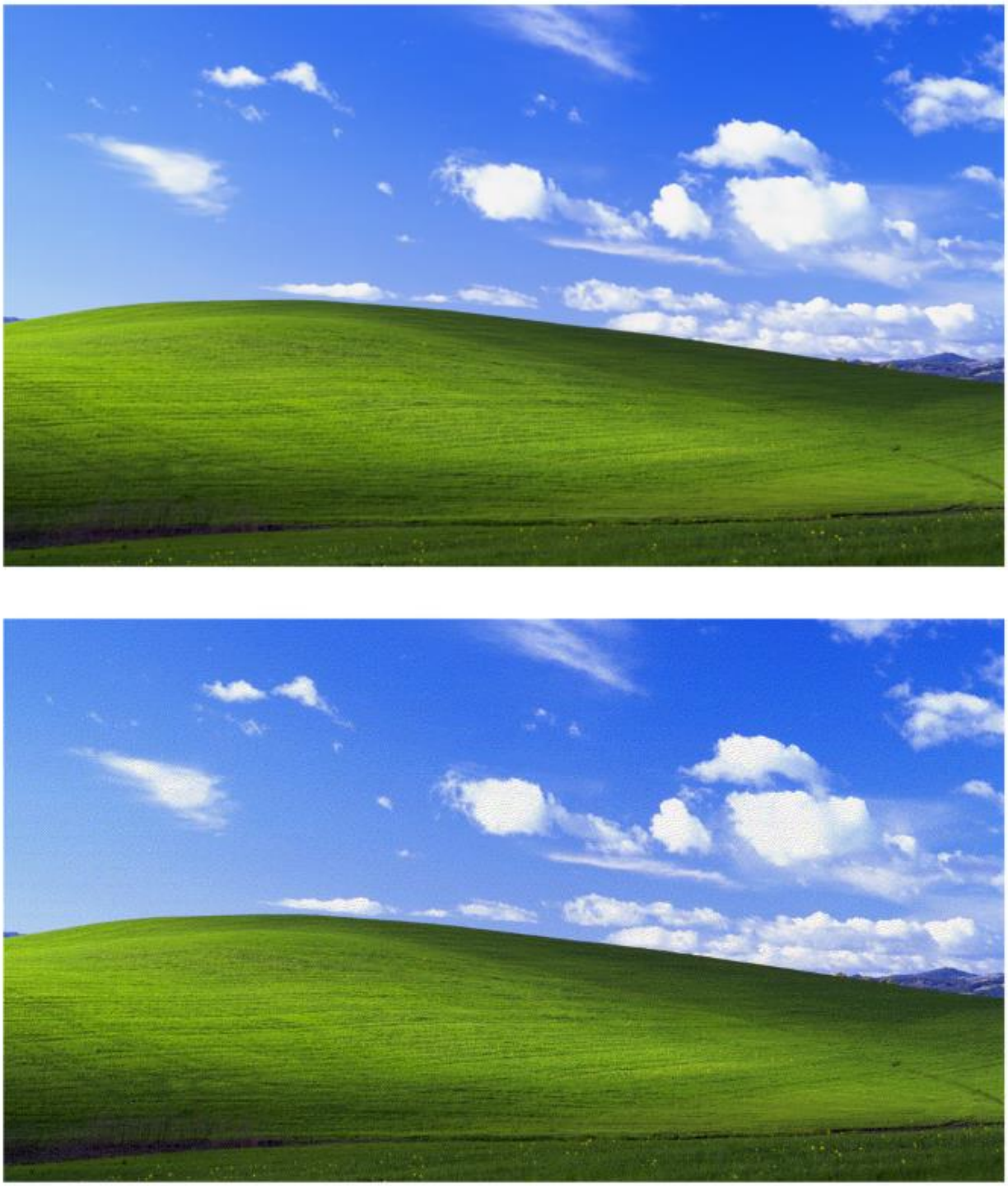

Рисунок 5 – Индивидуальное задание (2)

Вывод: В результате выполнения работы были изучены функции, использующиеся для моделирования процессов квантования и дискретизации изображения на языке Python.

1. Что такое интенсивность изображения?

Интенсивность изображения f (x, y) является функцией двух пространственных переменных x и y на ограниченной прямоугольной области.

2. Что такое квантование изображения?

Процесс разбиения непрерывного динамического диапазона значений яркости на ряд дискретных уровней называется квантованием.

3. Что происходит при квантовании изображения?

Уменьшается число градаций в сером изображении. Качество изображения становится хуже.

4. Чему равно число квантования?

$$K=[A/\Delta A]$$

5. Какая функция отвечает за квантование в OpenCV?

cv2.kmeans

6. Что такое cv.TERM_CRITERIA_EPS и cv.TERM_CRITERIA_MAX_ITER?

cv.TERM_CRITERIA_EPS - остановить итерацию алгоритма, если достигнута заданная точность (1.0)

cv.TERM_CRITERIA_MAX_ITER - останавливает алгоритм после указанного количества итераций (10)

7. Что такое дискретизация изображения?

Дискретизация – это преобразование непрерывных изображений в набор дискретных значений в форме кода.

8. Что произойдет с изображением после дискретизации?

Потеря степени детализации, качество изображения становится хуже, оно приобретает «ступенчатость».

9. Какой алгоритм дискретизации?

Алгоритм дискретизации: разбиваем три матрицы цветного изображения на отдельные блоки с шагом дискретизации К. В каждомблоке вычисляем среднее значение по каждому цвету в отдельности и полагаем, что внутри блока интенсивность равна вычисленному среднему значению.