МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ

ФЕДЕРАЦИИ

Федеральное государственное автономное

образовательное учреждение высшего образования

«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Кафедра инфокоммуникаций

«Процессы дискретизации и квантования изображения»

Отчет по лабораторной работе № 8 (2)

по дисциплине «Технологии распознавания образов»

Выполнил студент группы ПИЖ-б-о-21-1

Кучеренко С. Ю. « » 2023г.

Подпись студента_____

Работа защищена « »_____2023г.

Проверил Воронкин Р.А. _____

(подпись)

Ставрополь 2023

Цель работы: изучение функций, использующихся для моделирования процессов квантования и дискретизации изображения на языке Python.

Выполнение работы:

Задание 2.1.

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

In [37]:
```
import cv2
import numpy as np
```

In [38]:
```
image = cv2.imread('avto.jpg')
img = image.copy()
```

In [39]:
```
K = 10 # 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))
```

In [40]:
```
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)
```

In [41]:
```
img = cv2.resize(img, (w1, h1))
res = np.hstack((image, img))
```

In [ ]:
```
cv2.imshow("Img", res)
cv2.waitKey(0)
```

Рисунок 1 – Код программы

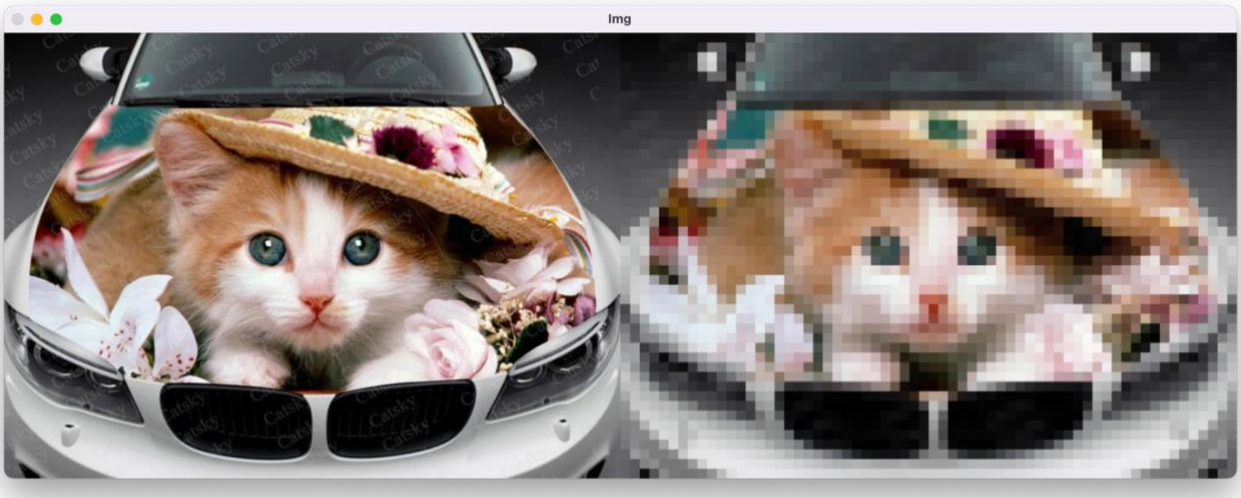


Рисунок 2 – Результат выполнения программы

Задание 2.2.

Проквантовать изображение, сократив число градаций до 4

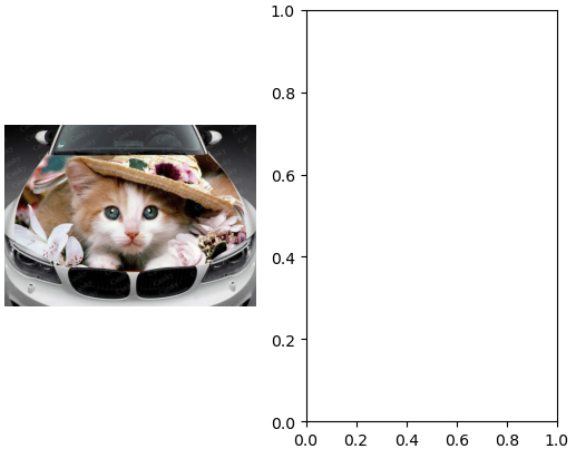
In [1]:
```
import cv2
import numpy as np
from matplotlib import pyplot as plt
```

In [6]:
```
plt.subplot(121)
img = cv2.imread('avto.jpg')

plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
plt.axis('off')
plt.subplot(122)

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

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



In [8]:
```
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))
```

In [*]:
```
cv2.imshow('Img2', res2)
cv2.waitKey(0)
```

Рисунок 3 – Код программы

Рисунок 4 – Результат выполнения программы

Самостоятельное задание

Загружаем изображение, которое оставляет только красный цвет, а остальную часть картинки делает серой.

In [39]:
```
import cv2
import numpy as np
from matplotlib import pyplot as plt
```

In [40]:
```
img = cv2.imread('cat.jpeg')
```

Определяем диапазон красного цвета в HSV и конвертируем изображение в цветовое пространство HSV

In [41]:
```
lower_red = np.array([0, 50, 50])
upper_red = np.array([10, 255, 255])

hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)
```

Выполняем бинаризацию изображения, оставляя только красные области, используя cv2.inRange() (создает бинарную маску изображения, где белые пиксели соответствуют красному цвету, а черные пиксели - остальным цветам). Применяем маску к изображению, оставляя только красный цвет.

In [46]:
```
mask = cv2.inRange(hsv, lower_red, upper_red)

red_only = cv2.bitwise_and(img, img, mask=mask)
```

Преобразуем оставшуюся часть изображения в серый цвет.

In [43]:
```
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
gray = cv2.cvtColor(gray, cv2.COLOR_GRAY2BGR)

gray_only = cv2.bitwise_and(gray, gray, mask = cv2.bitwise_not(mask))
```

Объединяем красный и серый цвета в одно изображение.

In [44]:
```
result = cv2.bitwise_or(red_only, gray_only)
```

Рисунок 5 – Код программы самостоятельного задания

```
In [50]: plt.subplot(221), plt.imshow(cv2.cvtColor(red_only, cv2.COLOR_BGR2RGB)), plt.title('Red
         plt.xticks([]), plt.yticks([])

         plt.subplot(222), plt.imshow(cv2.cvtColor(gray_only, cv2.COLOR_BGR2RGB)), plt.title('Gr
         plt.xticks([]), plt.yticks([])

         plt.subplot(223), plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB)), plt.title('Original
         plt.xticks([]), plt.yticks([])

         plt.subplot(224), plt.imshow(cv2.cvtColor(result, cv2.COLOR_BGR2RGB)), plt.title('Proce
         plt.xticks([]), plt.yticks([])
         plt.show()
```

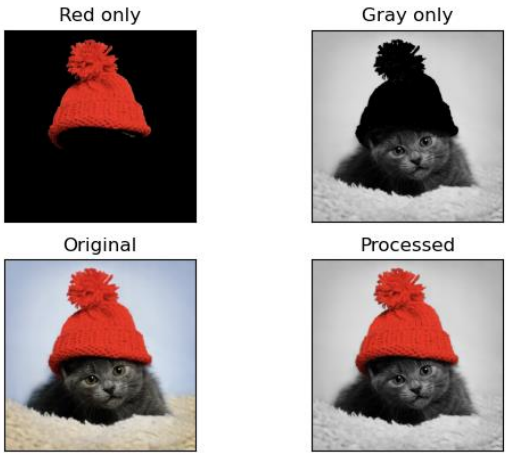


Рисунок 6 – Код программы самостоятельного задания

Вывод: при квантовании изображения уменьшается число градаций в сером изображении. Качество изображения становится хуже.