ГУАП

КАФЕДРА № 44

ОТЧЕТ

ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| доцент, канд. техн. наук |  |  |  | Н. В. Соловьёв |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №1 |
| РАЗРАБОТКА WINDOWS-ПРИЛОЖЕНИЯ |
| по курсу: ЦИФРОВАЯ ОБРАБОТКА ИЗОБРАЖЕНИЯ |
|  |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. № | 4841 |  |  |  |  |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2021

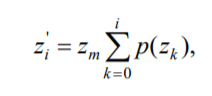
1. **Индивидуальное задание**

Эквализация и получение бимодальной гистограммы яркости.

1. **Теоретические положения**

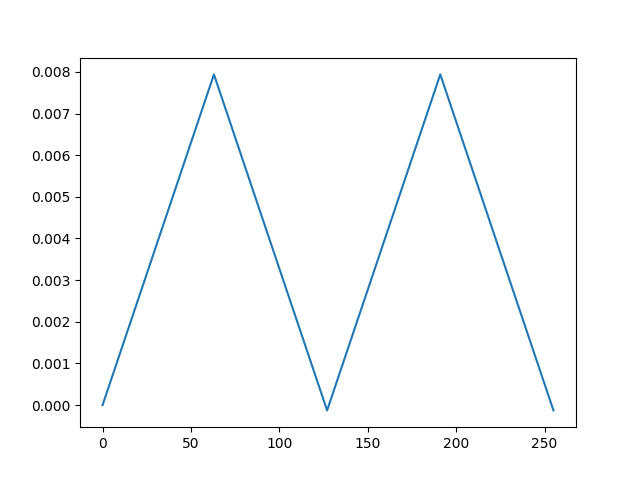
Локальное увеличение контрастности фрагмента изображения можно получить, используя гистограмму яркости некоторой окрестности преобразуемого пикселя (локальную гистограмму). После получения локальной гистограммы, выполняется требуемое преобразование шкалы яркости, например, эквализация согласно (3.8), и по его результату определяется яркость центрального пикселя окрестности. Далее центр окрестности смещается на следующий пиксель и процедура повторяется. В данном случае на каждом шаге не требуется заполнять массив соответствия яркостей, т.к. необходимо найти только один элемент этого массива, соответствующий яркости преобразуемого пикселя.

Увеличение различимости фрагментов изображения можно получить и путем эквализации или выравнивания гистограммы, т.е. такого преобразования шкалы яркости, при котором гистограмма результирующего изображения будет приближаться по форме к равномерной. Метод основан на предположении, что наибольшая контрастность достигается на изображении, гистограмма которого представляет равномерное распределение пикселей по яркостям на всем диапазоне (0 … 255). Преобразование шкалы яркости имеет вид

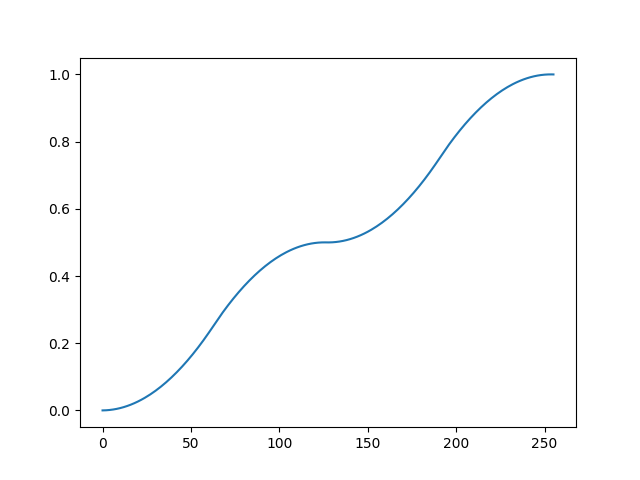


где z'i – значение элемента преобразованной шкалы яркости, соответствующее iой яркости исходной шкалы, p(zk) – нормализованная гистограмма яркости исходного изображения (i = 0 … 255).

Для получения требуемой гистограммы изображения была построена гистограмма следующего вида



Далее была получена кумулятивная гистограмма



Затем, итерируясь по массиву изображения, яркость исходного изображения заменялась на яркость, соответствующую ближайшему значению на кумулятивной бимодальной гистограмме.

1. **Процесс выполнения работы**

Язык разработки – python, фреймворк – Qt, версия – 6.2.3

Фреймворк Qt позволяет создавать десктоп-приложения, используя готовые модули для отображения объектов в окне программы.

Программа осуществляет поиск по названию среди изображений, доступных для просмотра. По нажатию кнопки система начинает процесс обработки изображения. После всех математических вычислений приложение выводит обработанное изображение в python-окне.

1. Вычисляется гистограмма исходного изображения
2. Гистограмма исходного изображения нормализуется
3. Вычисляется кумулятивная гистограмма
4. Вычисляется новое значение яркости

Первый пик бимодальной гистограммы находится на точке 63, второй пик находится на точке 191. Ноль между двумя пиками находится на точке 127. Приращение всегда одинаковое и составляет 1/64.

Был использован специальный модуль opencv для того, чтобы упростить задачу загрузки изображения в память в виде массива, вывода изображения на экран и сохранения получившегося изображения в виде файла формата .jpg

Код программы и результат выполнения представлены ниже.

1. **Код программы**

from cv2 import imwrite, imread, imshow, waitKey, IMREAD\_GRAYSCALE  
import Plots  
  
  
# Загрузка изображения  
def load\_image(text):  
 text += '.jpg'  
 image = imread(f'{text}', IMREAD\_GRAYSCALE)  
 return image  
  
  
# Эквализация изображения  
def equalize\_image(image, text, cum\_hist):  
  
 result\_image = image.copy()  
  
 for i in range(image.shape[0]):  
 for j in range(image.shape[1]):  
 result\_image[i][j] = 255 \* cum\_hist[result\_image[i][j]]  
  
 imshow('biba', result\_image)  
 waitKey(0)  
 imwrite(f'{text}-equalization.jpg', result\_image)  
  
 return result\_image  
  
  
# Нахождение ближайшего элемента  
def get\_nearest(x, list\_of\_x):  
 for i in range(1, len(list\_of\_x)):  
 if list\_of\_x[i] > x:  
 return i - 1  
  
  
# Обработка изображения, бимодальная гистограмма  
def make\_bimodal\_hist\_image(image, text, hist\_cum, hist\_cum\_bim):  
 result\_image = image.copy()  
  
 for i in range(image.shape[0]):  
 for j in range(image.shape[1]):  
 result\_image[i][j] = get\_nearest(hist\_cum[image[i][j]], hist\_cum\_bim)  
  
 imshow('result-bimodal', result\_image)  
 waitKey(0)  
 imwrite(f'{text}-bimodal.jpg', result\_image)  
  
 return result\_image  
  
  
class MainWindow(QMainWindow):  
  
 def \_\_init\_\_(self):  
 super(MainWindow, self).\_\_init\_\_()  
  
 self.button\_is\_checked = True  
  
 self.setWindowTitle("Matrix 3x3 income/result")  
  
 self.label = QLabel()  
  
 self.input = QLineEdit()  
 self.input.setObjectName("host")  
 self.input.setText("host")  
 self.input.setMaxLength(10)  
 self.input.textEdited.connect(self.label.setText)  
  
 self.button3 = QPushButton("Calculate")  
 self.button3.clicked.connect(self.calculate)  
  
 layout = QVBoxLayout()  
 layout.addWidget(self.input)  
 layout.addWidget(self.label)  
 layout.addWidget(self.button3)  
  
 container = QWidget()  
 container.setLayout(layout)  
  
 # Устанавливаем центральный виджет Window  
 self.setCentralWidget(container)  
  
 def set\_text(self, text):  
 self.label.setText(text)  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 var\_text = 'tiger'  
 biba\_image = load\_image(var\_text)  
  
 # Эквализация  
 image\_hist = Plots.make\_hist(biba\_image)  
 Plots.draw\_plot(image\_hist)  
 image\_cum\_hist = Plots.make\_cum\_hist(image\_hist)  
 Plots.draw\_plot(image\_cum\_hist)  
 # equalized\_image = equalize\_image(biba\_image, var\_text, image\_cum\_hist)  
 # equalized\_image\_hist = Plots.make\_hist(equalized\_image)  
 # Plots.draw\_plot(equalized\_image\_hist)  
  
 # Получение требуемой гистограммы  
 bimodal\_hist = Plots.make\_bimodal\_hist()  
 Plots.draw\_plot(bimodal\_hist)  
 bimodal\_cum\_hist = Plots.make\_cum\_hist(bimodal\_hist)  
 Plots.draw\_plot(bimodal\_cum\_hist)  
 bimodal\_image = make\_bimodal\_hist\_image(biba\_image, var\_text, image\_cum\_hist, bimodal\_cum\_hist)  
 bimodal\_hist\_image = Plots.make\_hist(bimodal\_image)  
 Plots.draw\_plot(bimodal\_hist\_image)

1. **Результаты выполнения программы**

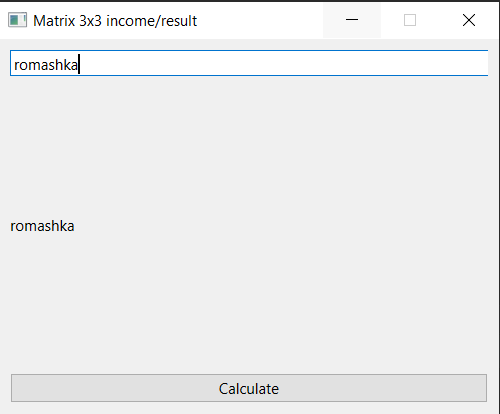
****

Рисунок 1 – графический интерфейс для работы с приложением



Рисунок 2 – исходное изображение 1

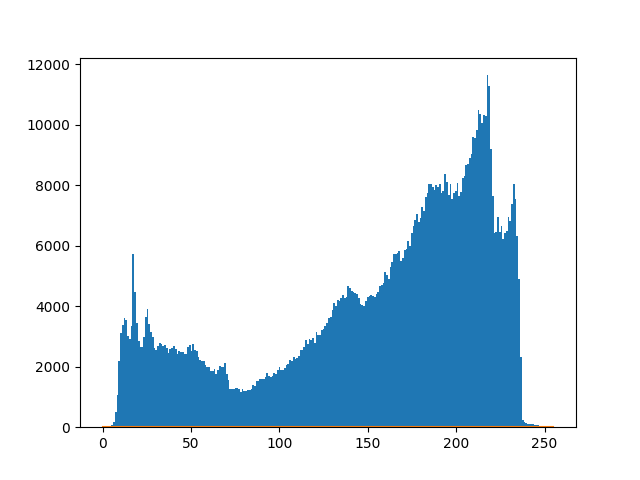


Рисунок 3 – гистограмма исходного изображения 1



Рисунок 4 – эквализированное изображение 1

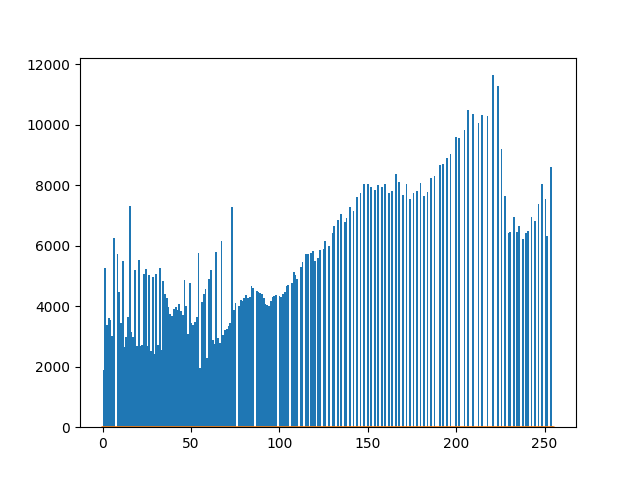


Рисунок 4 – гистограмма обработанного изображения 1



Рисунок 5 – изображение 1 с бимодальной гистограммой

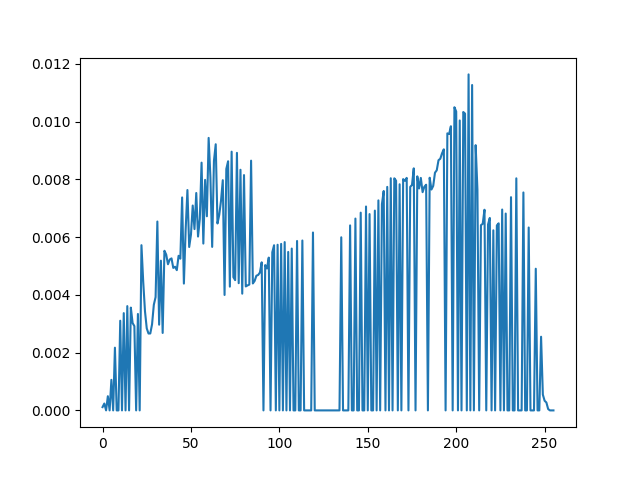


Рисунок 6 – бимодальная гистограмма изображения 1



Рисунок 7 – исходное изображение 2

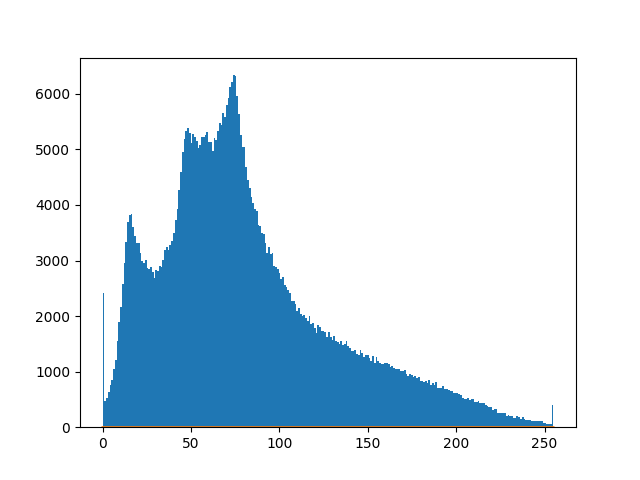


Рисунок 8 – гистограмма обработанного изображения 2



Рисунок 9 – эквализированное изображение 2

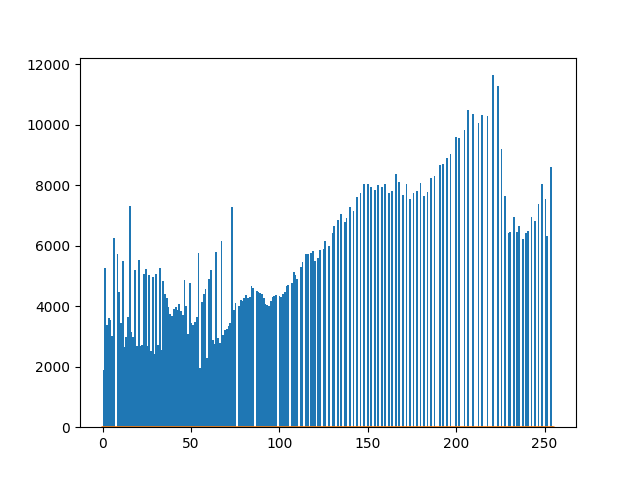


Рисунок 10 – эквализированная гистограмма изображения 2



Рисунок 11 – изображение 2 с требуемой гистограммой

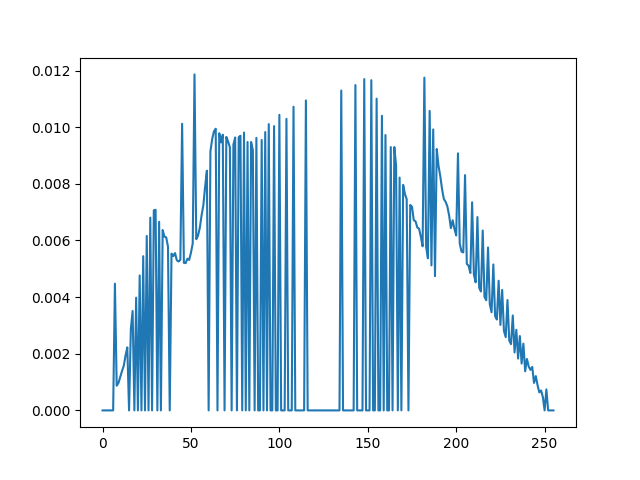


Рисунок 12 – бимодальная гистограмма изображения 2

1. **Вывод**

Выполнена лабораторная работа “Разработка Windows-приложения”, получены навыки эквализации и получения требуемой гистограммы изображения.