**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра САУ**

**отчет**

**по лабораторной работе № 4**

**по дисциплине «Техническое зрение»**

**Тема: Пороговые фильтры**

|  |  |  |
| --- | --- | --- |
| Студентки гр. 6491 |  | Ким А.В.  Фещенко М.В. |
| Преподаватель |  | Моклева К.А. |

Санкт-Петербург

2020

*Цель работы:* изучить принцип применения пороговых фильтров для обработки изображений

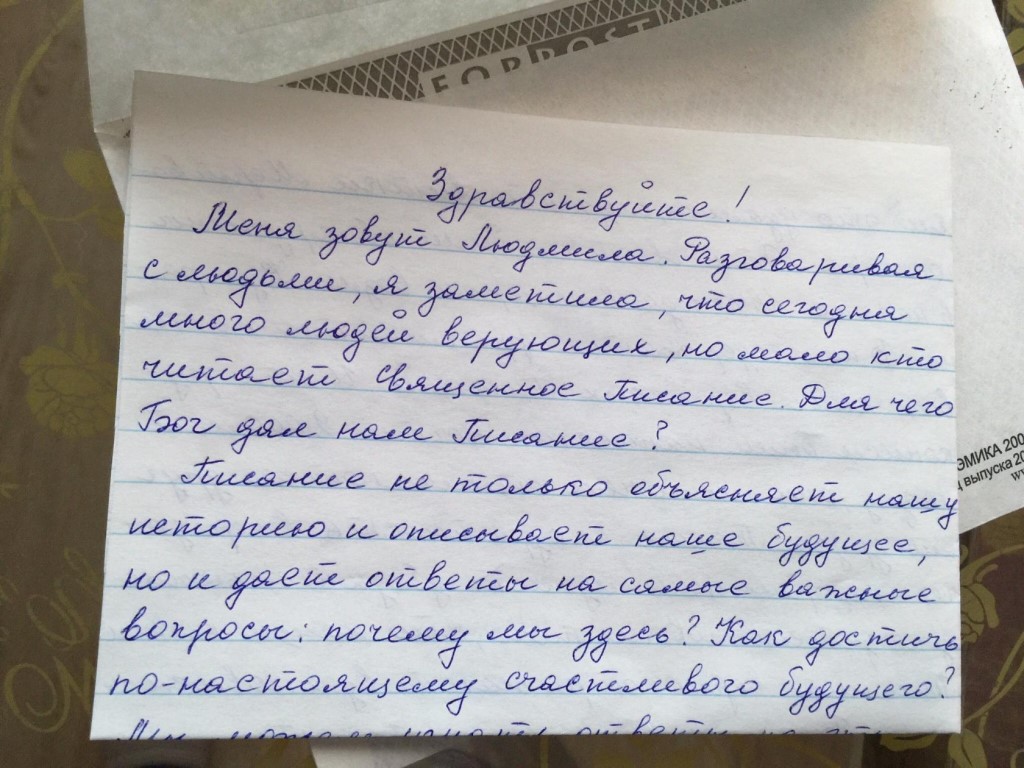
*Задание:*

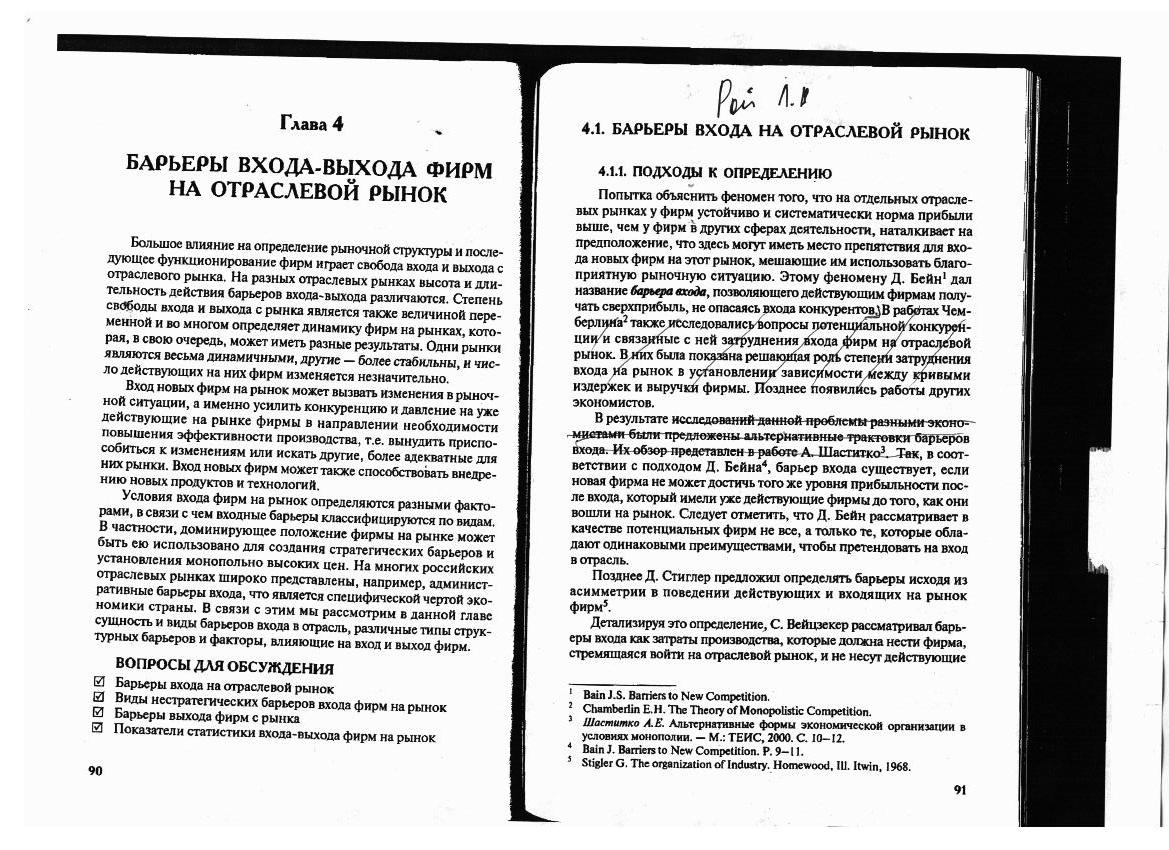
1. Напишите свою “легкую” реализацию cv2.threshold() только для варианта THRESH\_BINARY. Функция должна принимать значение threshold. Пусть maxVal по умолчанию всегда будет 255.
2. Примените на практике бинаризацию всех типов для следующих изображений (нескольких каждого типа):
3. фото дорожной разметки белого цвета;
4. изображение отсканированного текстового документа;
5. фото написанного от руки или напечатанного текста.

**Изображения для работы**

****

****

****

****

**Код программы**

import cv2

import numpy as np

import math

a = input ("Ввведите порог:")

thresh = int(a)

bsize = input ("Ввведите blocksize:")

b = int(bsize)

const = input ("Ввведите константу С:")

c = int(const)

img = cv2.imread ('tulip.jpg',0)

cv2.imshow ('pic00',img)

ret ,new\_img = cv2.threshold (img, tresh, 255, cv2.THRESH\_BINARY)

cv2.imshow('pic01',new\_img)

cv2.imwrite('binary.jpg',new\_img)

def function(img, thresh):

for i in range(len(img)):

for j in range(len(img[i])):

if img[i][j] > thresh:

img[i][j] = 255

else:

img[i][j] = 0

return img

imgself = cv2.imread ('tulip.jpg',0)

imgself = function (img, thresh)

cv2.imshow('pic01self',imgself)

cv2.imwrite('binaryself.jpg',imgself)

img1 = cv2.imread ('road.jpg',0)

cv2.imshow('pic10',img1)

ret ,new\_img1 = cv2.threshold( img1, thresh, 255, cv2.THRESH\_BINARY)

cv2.imshow ('pic1',new\_img1)

cv2.imwrite ('binary1.jpg',new\_img1)

img2 = cv2.imread ('scan.jpg',0)

cv2.imshow ('pic20',img2)

ret ,new\_img2 = cv2.threshold (img2, thresh, 255, cv2.THRESH\_BINARY)

cv2.imshow ('pic2',new\_img2)

cv2.imwrite ('binary2.jpg',new\_img2)

img3 = cv2.imread ('pismo.jpg',0)

cv2.imshow ('pic30',img3)

ret ,new\_img3 = cv2.threshold(img3, thresh, 255, cv2.THRESH\_BINARY)

cv2.imshow ('pic3',new\_img3)

cv2.imwrite ('binary3.jpg',new\_img3)

ret ,new\_img4 = cv2.threshold( img1, thresh, 255, cv2.THRESH\_OTSU)

cv2.imshow ('pic4',new\_img4)

cv2.imwrite ('binary4.jpg',new\_img4)

ret ,new\_img5 = cv2.threshold (img2, thresh, 255, cv2.THRESH\_OTSU)

cv2.imshow ('pic5',new\_img5)

cv2.imwrite ('binary5.jpg',new\_img5)

ret ,new\_img6 = cv2.threshold(img3, thresh, 255, cv2.THRESH\_OTSU)

cv2.imshow ('pic6',new\_img6)

cv2.imwrite ('binary6.jpg',new\_img6)

new\_img7 = cv2.adaptiveThreshold(

img1, 255, cv2.ADAPTIVE\_THRESH\_GAUSSIAN\_C,

cv2.THRESH\_BINARY, b, c)

cv2.imshow ('pic7',new\_img7)

cv2.imwrite ('binary7.jpg',new\_img7)

new\_img8 = cv2.adaptiveThreshold(

img2, 255, cv2.ADAPTIVE\_THRESH\_GAUSSIAN\_C,

cv2.THRESH\_BINARY, b, c)

cv2.imshow ('pic8',new\_img8)

cv2.imwrite ('binary8.jpg',new\_img8)

new\_img9 = cv2.adaptiveThreshold(

img3, 255, cv2.ADAPTIVE\_THRESH\_GAUSSIAN\_C,

cv2.THRESH\_BINARY, b, c)

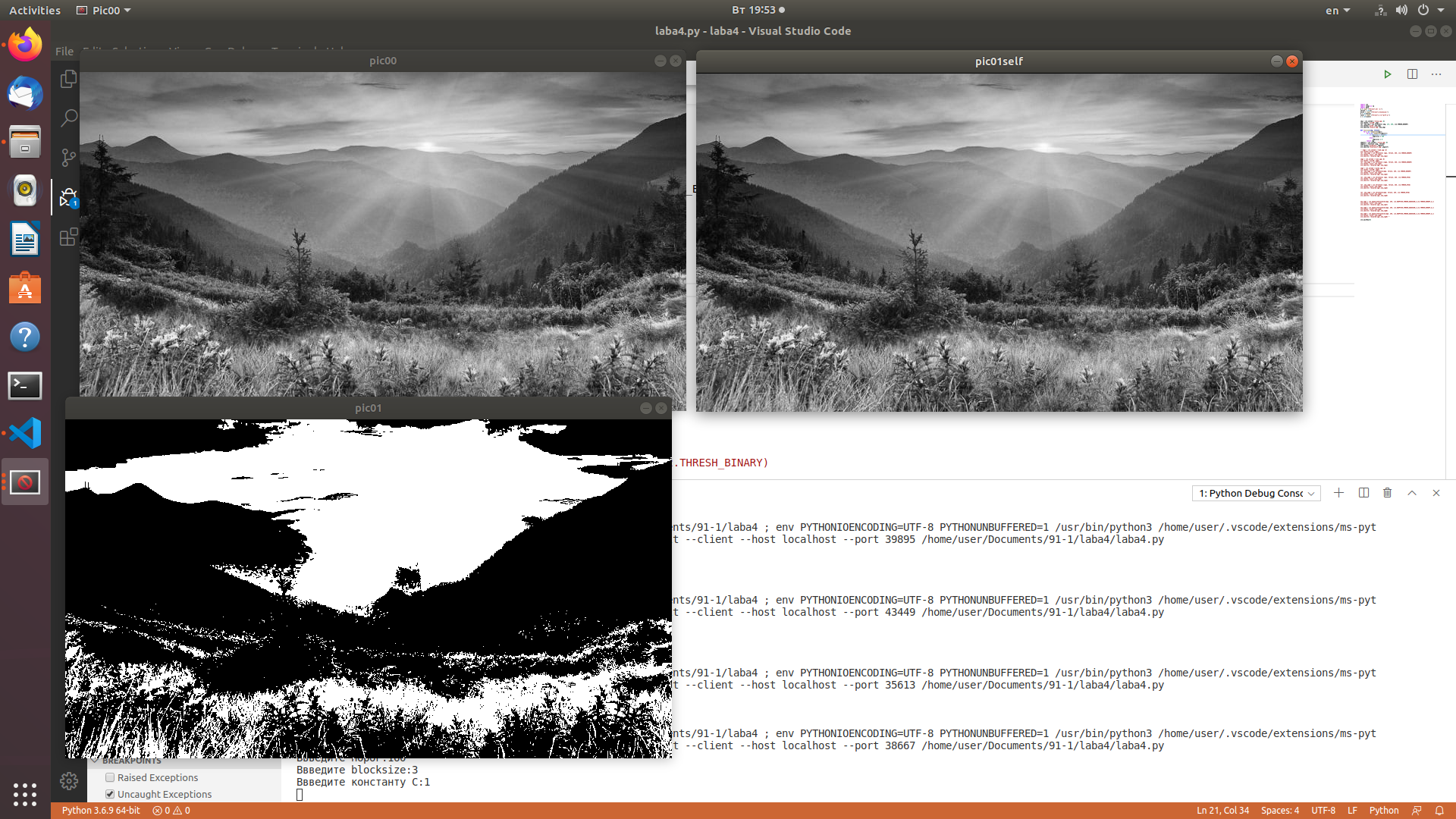
cv2.imshow ('pic9',new\_img9)

cv2.imwrite ('binary9.jpg',new\_img9)

cv2.waitKey(0)

**Результаты**

**1)**

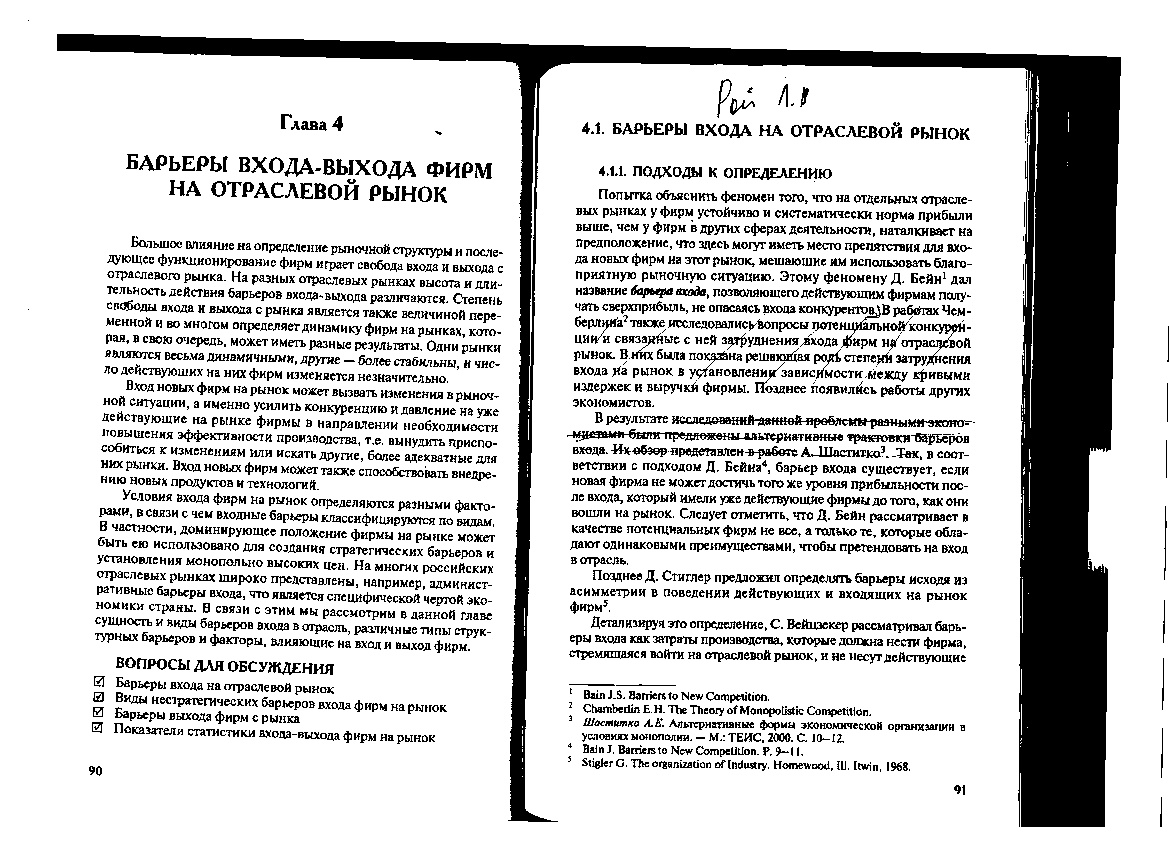
****

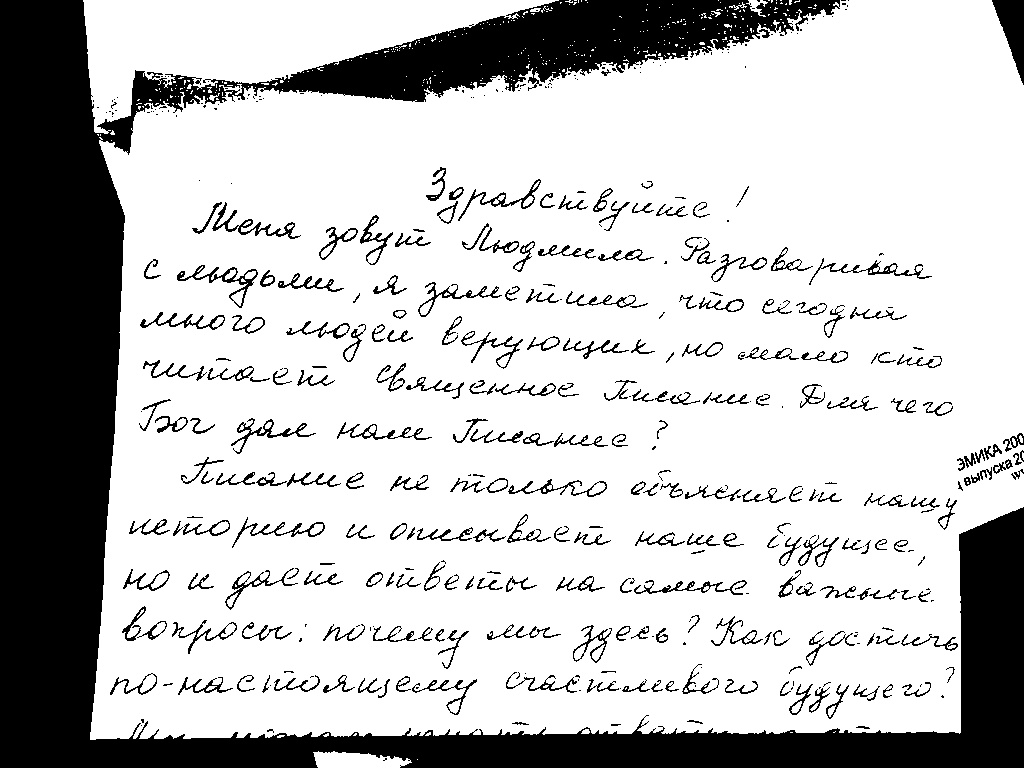
Из полученных результатов видно, что ручное задание фильтрации изображения сильно отличается от встроенного фильтра.

**2)** Параметры: порог – 127, blocksize – 5, константа С – 7

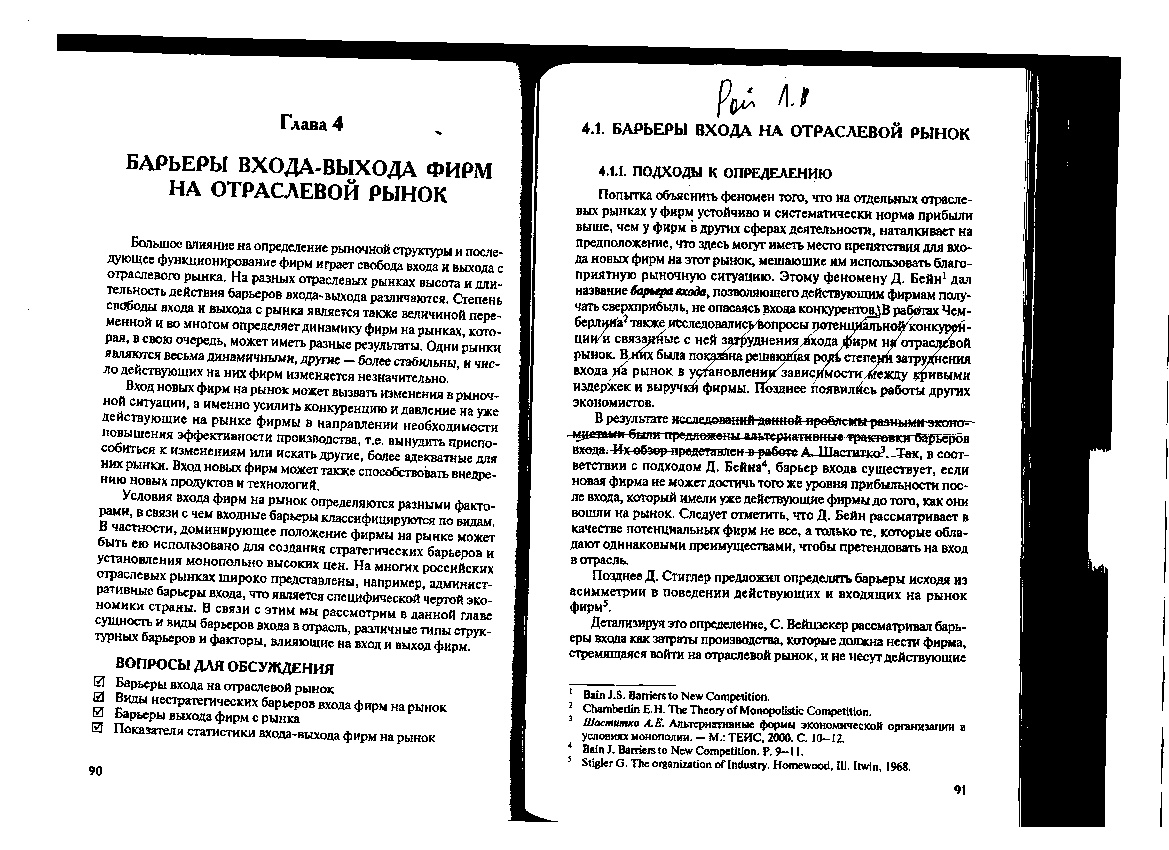


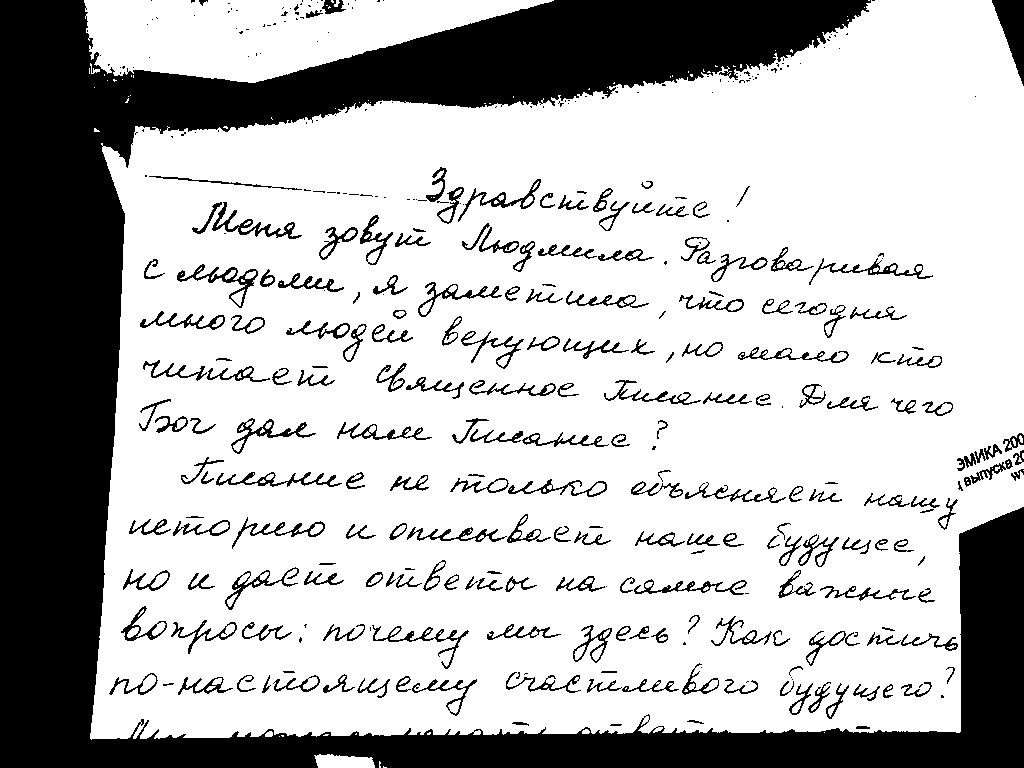




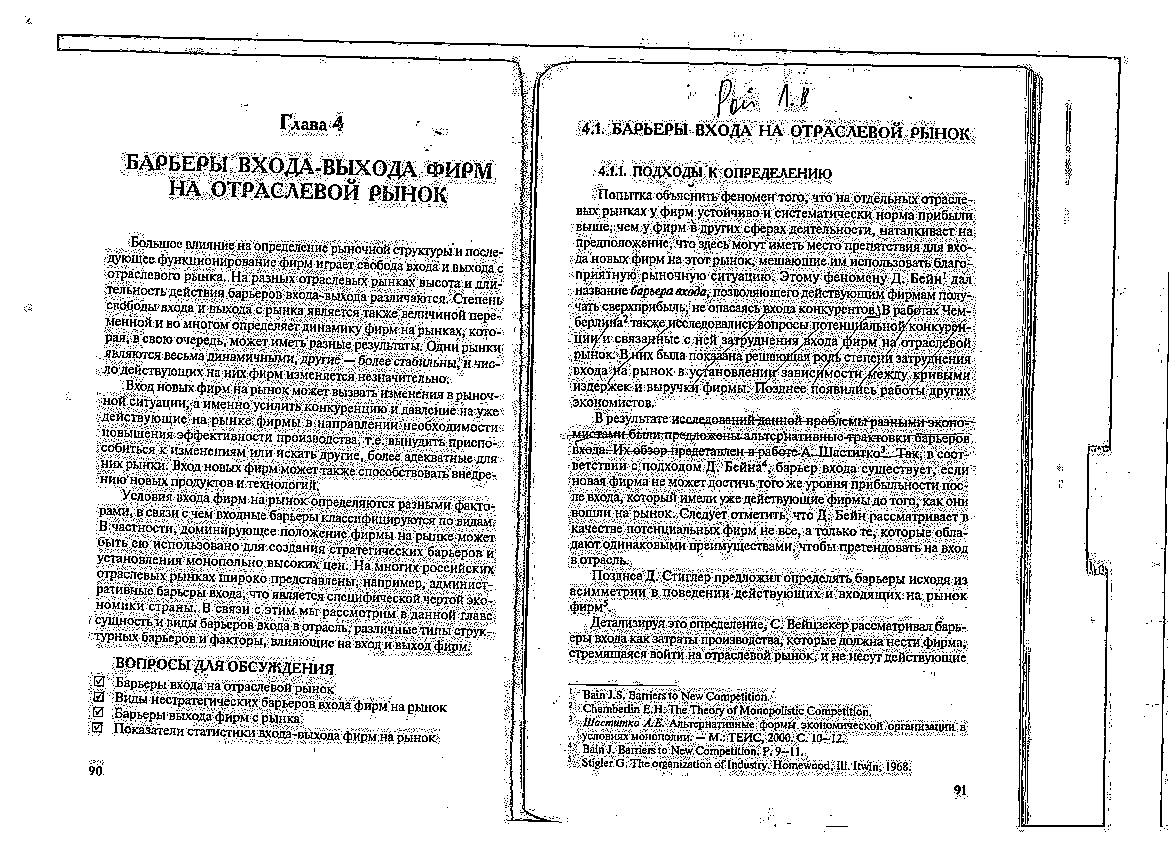


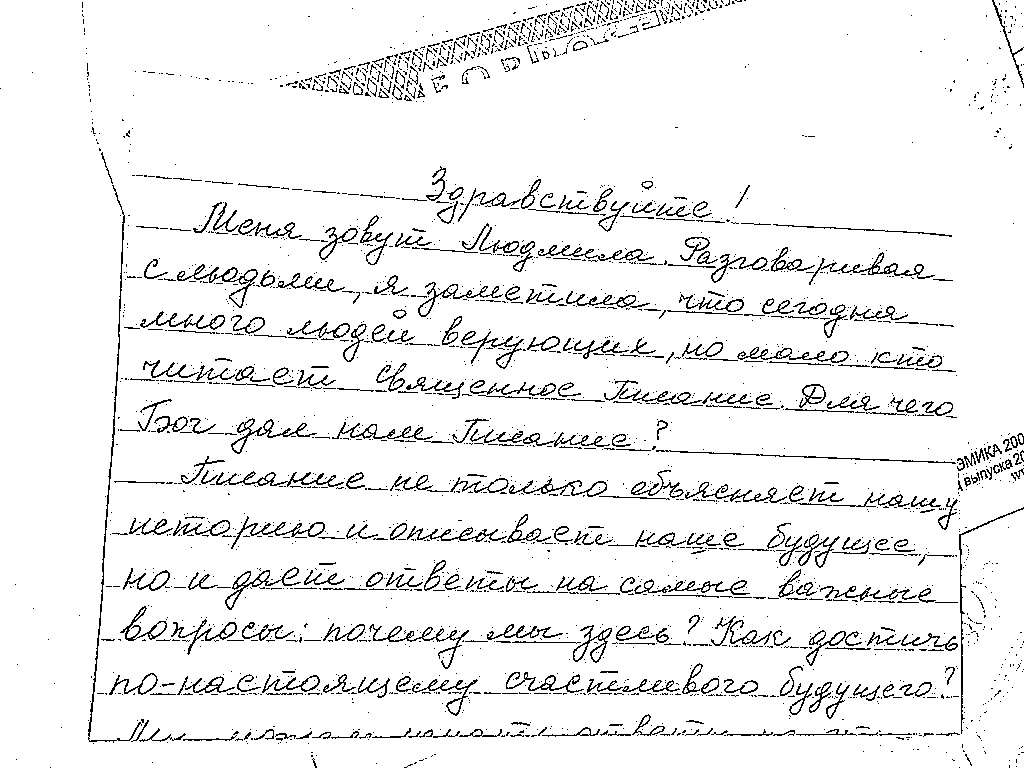












Из полученных результатов видно, что для каждого из видов изображений требуется определенный фильтр для того, чтобы была возможность получить необходимую информацию. Например, фильтр Оцу отлично подходит для распознавания текста с фотографии и дорожной разметки на дороге; для отсканированного документа бинарный фильтр хорошо отредактировал изображение.

*Вывод:* в данной лабораторной работе былиизучены принципы применения пороговых фильтров для обработки изображений