Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

**(ФГБОУ ВО «КубГУ»)**

**Факультет компьютерных технологий и прикладной математики**

**Кафедра вычислительных технологий**

**ЛАБОРАТОРНАЯ РАБОТА №5**

**Дисциплина: Алгоритмы цифровой обработки мультимедиа**

Работу выполнил: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Романов В.В.

Направление подготовки: 02.03.02 Фундаментальная информатика и информационные технологии

Преподаватель:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Крамаренко А.А.

**Тема:** Выявление движения на видео.

**Ход работы:**

1. Начать чтение из файла, прочитать первый кадр, перевести в черно-белый цвет и применить размытие Гаусса;
2. Подготовить файл для записи;
3. Начать цикл, который завершиться по окончании файла, внутри этого цикла:

* скопировать старый кадр;
* прочитать новый кадр, перевести в черно-белый цвет, применить размытие Гаусса;
* если чтение неуспешно, остановить цикл;
* найти разницу между двумя кадрами в отдельный фрейм (frame\_diff) - cv2.absdiff;
* провести операцию двоичного разделения для фрейма (frame\_diff) - cv2.threshold;
* найти контуры объектов для фрейма (frame\_diff) - cv2.findContours;
* пройтись по контурам объектов для фрейма (frame\_diff) и найти контур площадью большей, чем наперед заданный параметр -cv2.contourArea;
* если такой контур найден, значит было движение, записать кадр в файл;
* отобразить видео.

Подготовим кадр для дальнейшей обработки: переведем в ЧБ формат и применим Гауссовское размытие. Запустим цикл, где будут происходить все преобразования и вычислим разницу между искомым изображением и новым с помощью функции cv2.absdiff(). Теперь бинаризируем её, используя функцию cv2.threshold(), превращая пиксели, превышающие порог delta\_tresh, в белый цвет, а остальные – в черные. Дальше найдем контуры (cv2.findContours()), используя пороговое значение. Наконец, будем проходиться по каждому контуру и сравнить его площадь с общей площадью кадра. Если она больше или равна значению min\_area, то записываем текущий кадр в выходное видео, иначе продолжаем проход. На рисунке 1 изображен процесс обработки видео программой.

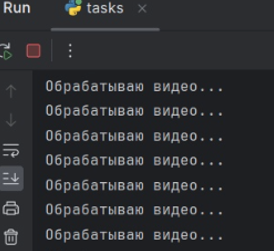


Рисунок 1 – Обработка видео циклом.

Были проведены тесты, которые показали более оптимальные значения параметров для выбранного видео. По завершении программы выводятся 5 видеофайлов, скриншот которых продемонстрирован на рисунке 2.

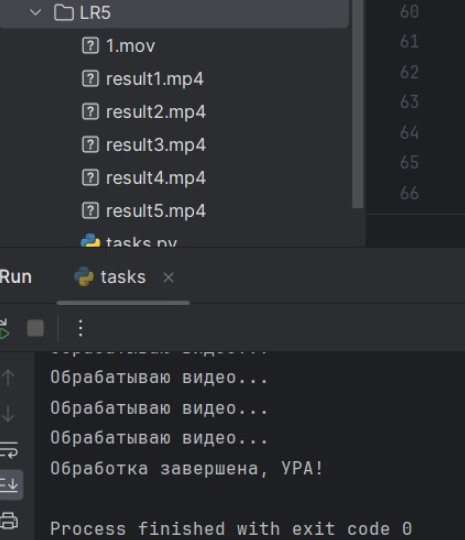


Рисунок 2 – Скриншот полученных видеофайлов.

**Тестирование №1:**

Размер ядра = 3

Стандартное отклонение = 50

Пороговое значение = 60

Минимальная площадь = 20

**Тестирование №2:**

Размер ядра = 3

Стандартное отклонение = 50

Пороговое значение = 40

Минимальная площадь = 5

**Тестирование №3:**

Размер ядра = 5

Стандартное отклонение = 50

Пороговое значение = 70

Минимальная площадь = 30

**Тестирование №4:**

Размер ядра = 5

Стандартное отклонение = 30

Пороговое значение = 50

Минимальная площадь = 15

**Тестирование №5:**

Размер ядра = 11

Стандартное отклонение = 70

Пороговое значение = 60

Минимальная площадь = 20

**Вывод:** в результате тестирования с разными входными параметрами, более оптимальные значения были продемонстрированы при тестировании №5 за счет того, что размер ядра и стандартное отклонение были больше, чем на других тестах.

**Листинг программы**

**Файл tasks.py**

import cv2

i = 0

def main(kernel\_size, standard\_deviation, delta\_tresh, min\_area):

global i

i += 1

video = cv2.VideoCapture('1.mov', cv2.CAP\_ANY)

ret, frame = video.read()

img = cv2.cvtColor(frame, cv2.COLOR\_BGR2GRAY)

img = cv2.GaussianBlur(img, (kernel\_size, kernel\_size), standard\_deviation)

w = int(video.get(cv2.CAP\_PROP\_FRAME\_WIDTH))

h = int(video.get(cv2.CAP\_PROP\_FRAME\_HEIGHT))

fourcc = cv2.VideoWriter\_fourcc(\*'mp4v')

video\_writer = cv2.VideoWriter('result' + str(i) + '.mp4', fourcc, 25, (w, h))

while True:

print('Обрабатываю видео...')

old\_img = img.copy()

is\_ok, frame = video.read()

if not is\_ok:

break

img = cv2.cvtColor(frame, cv2.COLOR\_BGR2GRAY)

img = cv2.GaussianBlur(img, (kernel\_size, kernel\_size), standard\_deviation)

frame\_diff = cv2.absdiff(img, old\_img)

thresh = cv2.threshold(frame\_diff, delta\_tresh, 255, cv2.THRESH\_BINARY)[1]

(contors, hierarchy) = cv2.findContours(thresh, cv2.RETR\_EXTERNAL, cv2.CHAIN\_APPROX\_SIMPLE)

for contr in contors:

area = cv2.contourArea(contr)

if area < min\_area:

continue

video\_writer.write(frame)

video\_writer.release()

print('Обработка завершена, УРА!')

kernel\_size = 3

standard\_deviation = 50

delta\_tresh = 60

min\_area = 20

main(kernel\_size, standard\_deviation, delta\_tresh, min\_area)

kernel\_size = 3

standard\_deviation = 50

delta\_tresh = 40

min\_area = 5

main(kernel\_size, standard\_deviation, delta\_tresh, min\_area)

kernel\_size = 5

standard\_deviation = 50

delta\_tresh = 70

min\_area = 30

main(kernel\_size, standard\_deviation, delta\_tresh, min\_area)

kernel\_size = 5

standard\_deviation = 30

delta\_tresh = 50

min\_area = 15

main(kernel\_size, standard\_deviation, delta\_tresh, min\_area)

kernel\_size = 11

standard\_deviation = 70

delta\_tresh = 60

min\_area = 20

main(kernel\_size, standard\_deviation, delta\_tresh, min\_area)