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**

**Отчёт**

**по лабораторной работе №4**

**Дисциплина: АЛГОРИТМЫ ЦИФРОВОЙ ОБРАБОТКИ МУЛЬТИМЕДИА**

**Тема: «выявление движения на видео»**

Работу выполнил \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Д. Ш. Пивоварова

Направление подготовки 02.03.02 Фундаментальная информатика и

информационные технологии

Направленность (профиль) Математическое и программное обеспечение

компьютерных технологий

Преподаватель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А. С. Жук

**Цель работы**

Цель работы заключается в решении задачи детекции движения на видео.

**Ход работы**

Работа будет осуществляться с помощью средств языка Python 3.10 и IDE PyCharm2022.1.2 с учебной лицензией. Для работы необходимо установить библиотеку opencv.

Необходимо прочитать исходное видео и записать новое, состоящее только из тех кадров, где произошло движение.

Алгоритм определение кадров с движением показан на рисунке 1.

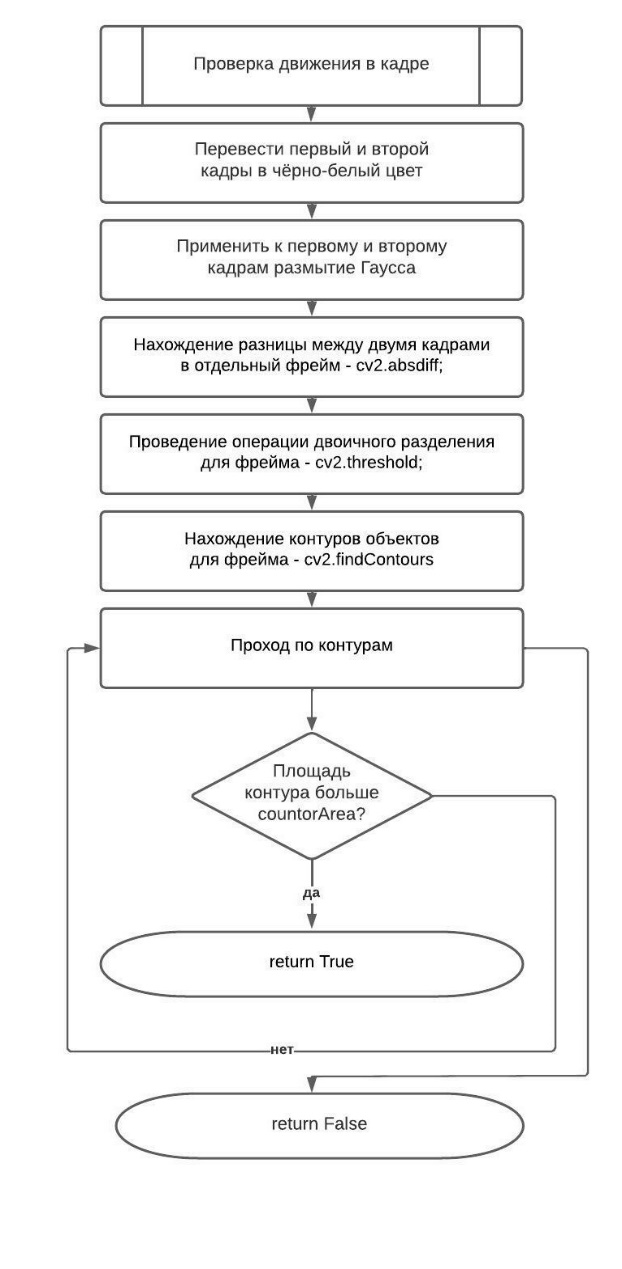


Рисунок 1 — Блок-схема алгоритма определения движения в кадре

Для реализации данного алгоритма понадобились следующие функции (и аргументы функций) библиотеки OpenCV:

cv2.cvtColor(frame, cv2.COLOR\_BGR2GRAY) — используется для преобразования изображения из одного цветового пространства в другое;

cv2.COLOR\_BGR2GRAY — преобразование между RGB/BGR и оттенками серого, значения пикселей находятся в диапазоне от 0 (чёрный цвет) до 255 (белый цвет);

cv2.GaussianBlur (frame, (5, 5), 1.3) — выполняет размытие по Гауссу. Применение размытия по Гауссу помогает удалить некоторые высокочастотные края изображения, которые нас не интересуют, и позволяет получить более «чистую» сегментацию;

(5, 5) — ширина и высота ядра;

1.3 — стандартное отклонение;

cv2.absdiff (frame1, frame2) — вычисляет абсолютную разницу между двумя массивами, frame1 и frame2;

cv2.threshold (frame, 30, 255, cv2.THRESH\_BINARY)[1] — применяет пороговое значение фиксированного уровня к каждому элементу массива. Функция возвращает кортеж из двух значений: первое является пороговым значением. В случае простого порогового значения это значение тривиально, поскольку мы вручную указали порог — 30. Второе возвращаемое значение — это само пороговое изображение;

frame — изображение в градациях серого;

30 — пороговое значение, которое используется для классификации значений пикселей;

255 — это максимальное значение, которое присваивается значениям пикселей, превышающим пороговое значение;

cv2.THRESH\_BINARY — определяет логику классификации. В данном случае функция threshold возвращает изображение, в котором все пиксели, которые темнее (меньше) 30 заменены на 0, а все, которые ярче (больше) 30 — на 255. Механизм работы cv2.THRESH\_BINARY показан на рисунке 2.

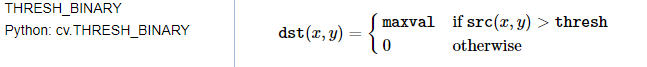


Рисунок 2 — Проверка оператора if для cv2.THRESH\_BINARY

cv2.findContours(thresh\_frame.copy(), cv2.RETR\_EXTERNAL, cv2.CHAIN\_APPROX\_SIMPLE) — применяет пороговое значение фиксированного уровня к каждому элементу массива. В OpenCV поиск контуров — это поиск белых объектов на черным фоне. Результат функции — это список всех контуров на изображении. Каждый отдельный контур представляет собой массив Numpy из (x,y) координат граничных точек объекта;

cv2.RETR\_EXTERNAL — извлекает все контуры и реконструирует полную иерархию вложенных контуров;

cv2.CHAIN\_APPROX\_SIMPLE — метод аппроксимации контура. Если передать [cv2.CHAIN\_APPROX\_NONE](https://docs.opencv.org/4.x/d3/dc0/group__imgproc__shape.html#gga4303f45752694956374734a03c54d5ffaf7d9a3582d021d5dadcb0e37201a62f8) , все граничные точки сохранятся. Но нет необходимости во всех точках: например, найден контур прямой линии, чтобы представить эту линию, нужны только две конечные точки этой линии. Это и делает [cv2.CHAIN\_APPROX\_SIMPLE](https://docs.opencv.org/4.x/d3/dc0/group__imgproc__shape.html#gga4303f45752694956374734a03c54d5ffa5f2883048e654999209f88ba04c302f5) . Он убирает все лишние точки и сжимает контур, тем самым экономя память;

cv2.contourArea (contours) — вычисляет площадь контуров;

Далее на рисунках 3–6 показан алгоритм сравнения двух первых кадров исходного видео:

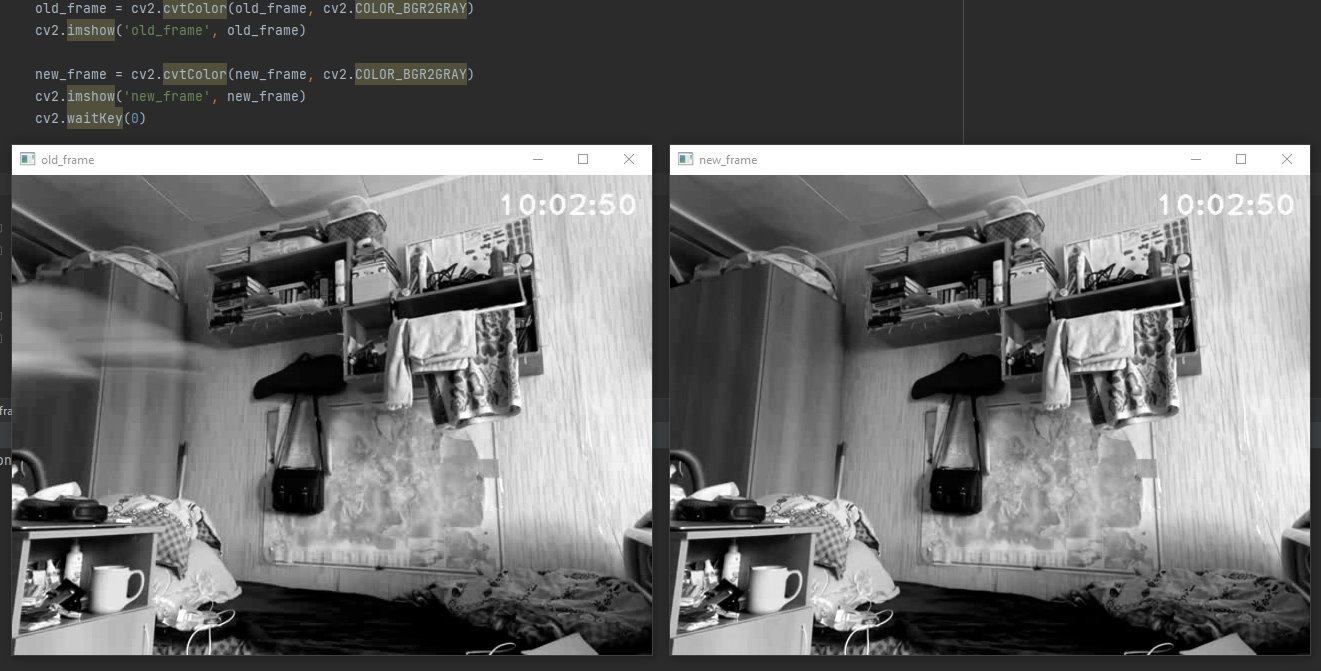


Рисунок 3 — Применение функции cv2.cvtColor

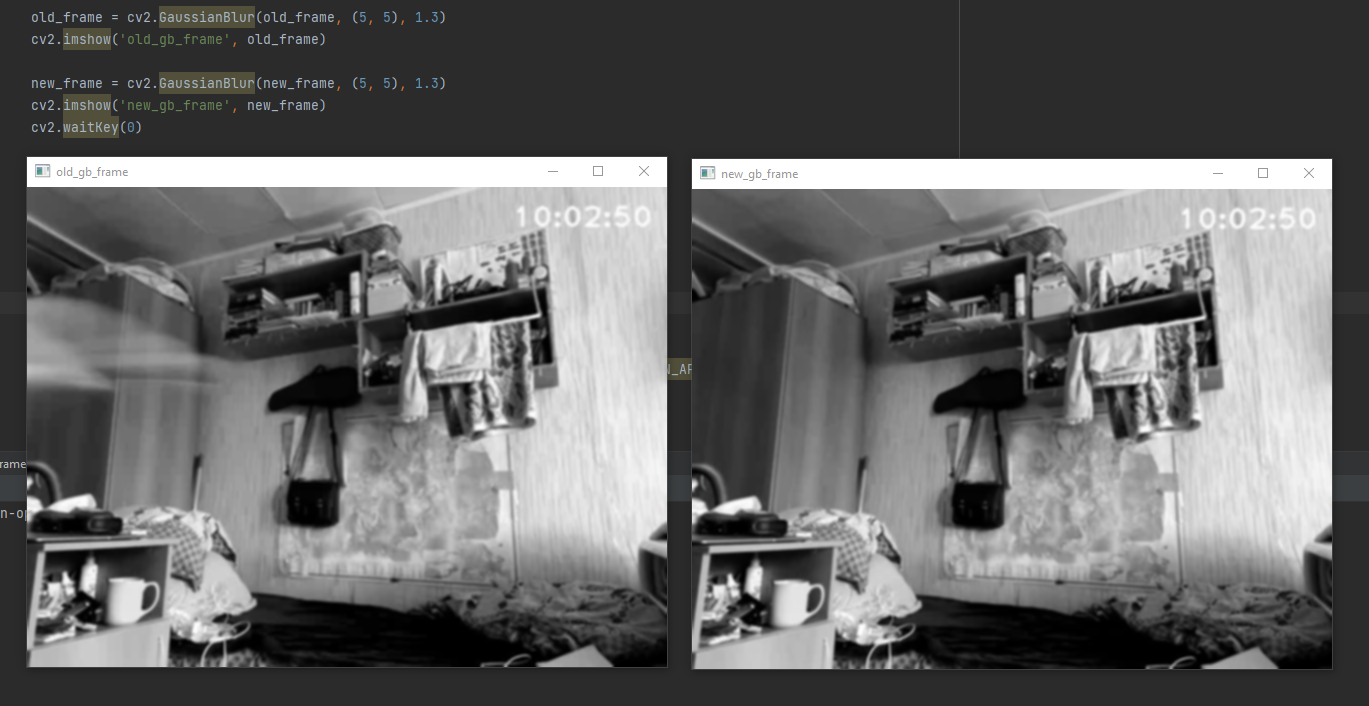


Рисунок 4 — Применение функции cv2.GaussianBlur

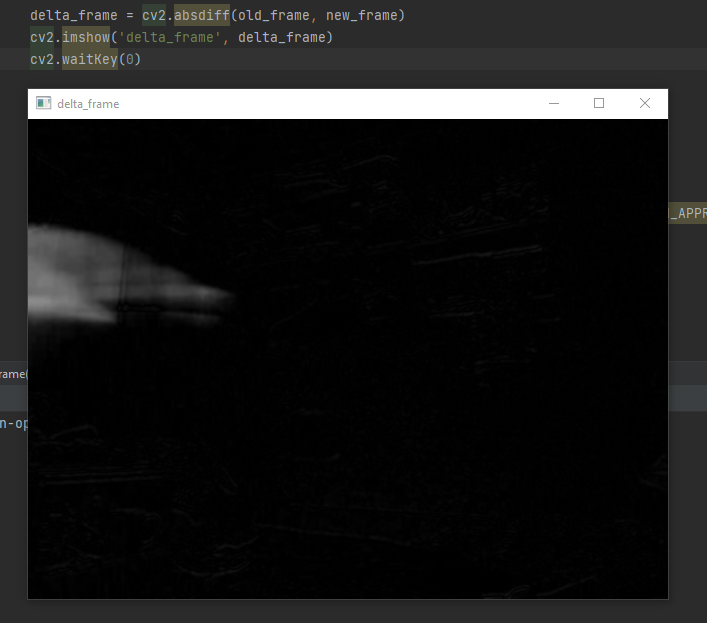


Рисунок 5 — Применение функции cv2.absdiff

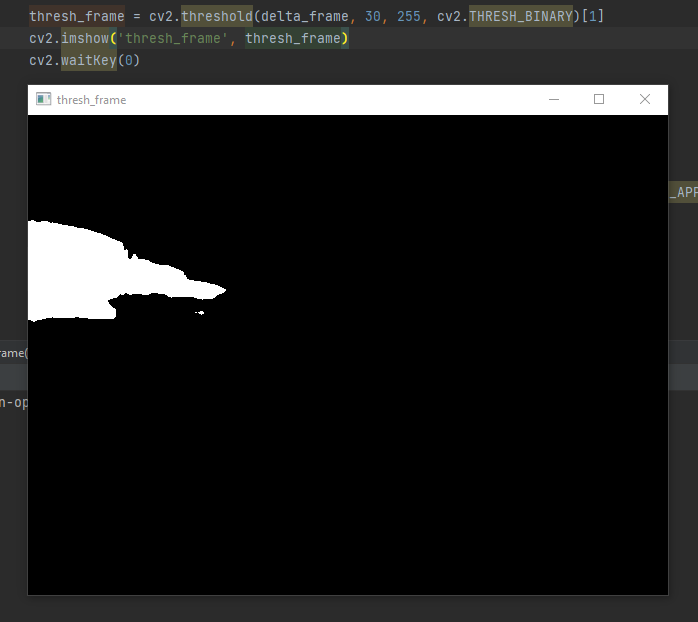


Рисунок 6 — Применение функции cv2.threshold

Если провести эксперементы с границей разделения для метода threshold и площадью минимального объекта, то можно сделать следующие выводы:

С помощью границы разделения для метода threshold можно отлавливать и незначительные изменения (пороговое значение — 30), а затем с площадью минимального объекта вычленять из них — более заметные. С другой стороны, если мы изначально не планируем использовать кадры с минимальным движением, то можно выбрать пороговое значение выше, например, 100. Для данного видео это критично — практически не фиксируется движения рук, поэтому стоит оставить его в диапазоне от 30 до 50, а площадь минимального объекта — от 1000 до 1500 (при 2500 уже не фиксируется движение рук справа).

**Вывод**

Были освоены функции библиотеки opencv для решения задачи детекции движения на видео.