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 №1**

**Дисциплина: АЛГОРИТМЫ ЦИФРОВОЙ ОБРАБОТКИ МУЛЬТИМЕДИА**

**Тема: «Тестирование вывода изображений и видео на экран, запись в файл, Формат изображения HSV, определение цвета, построение надписей и доп изображений на рисунке»**

Работу выполнил \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ К. А. Корнилов

Направление подготовки 02.03.02 Фундаментальная информатика и

информационные технологии

Направленность (профиль) Математическое и программное обеспечение

компьютерных технологий

Преподаватель\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_А. А. Крамаренко

**Тема:** Тестирование вывода изображений и видео на экран, запись в файл, Формат изображения HSV, определение цвета, построение надписей и доп изображений на рисунке.

**Цель:** Изучить базовые принципы работы с библиотекой opencv.

**Задачи:**

**Ход работы:**

1. Была установлена библиотека opencv для python 3.12 с использованием менеджера пакетов pip.

Код: pip install opencv-python

1. Были выведены на экран изображения и протестировать три возможных расширения, три различных флага для создания окна и три различных флага для чтения изображения.

Для этого был написан цикл, перебирающий три файла, содержащий внутренний цикл, который перебирает флаги чтения изображения. На каждой итерации цикла изображение открывается поочередно с флагами окна cv2.WINDOW\_AUTOSIZE, cv2.WINDOW\_KEEPRATIO, cv2.WINDOW\_FULLSCREEN и флагами чтения изображения cv2.IMREAD\_GRAYSCALE, cv2.IMREAD\_REDUCED\_COLOR\_4, cv2.IMREAD\_UNCHANGED.

Код:

import cv2  
formats = ['jpeg','bmp','png']  
windowFlags = {cv2.WND\_PROP\_AUTOSIZE :cv2.WINDOW\_AUTOSIZE,cv2.WND\_PROP\_ASPECT\_RATIO:cv2.WINDOW\_KEEPRATIO,cv2.WND\_PROP\_FULLSCREEN:cv2.WINDOW\_FULLSCREEN}  
imageReadFlags = [cv2.IMREAD\_GRAYSCALE,cv2.IMREAD\_REDUCED\_COLOR\_4,cv2.IMREAD\_UNCHANGED ]  
for format in formats:  
 for imageReadFlag in imageReadFlags:  
 img = cv2.imread(f'img.{format}',imageReadFlag)  
cv2.namedWindow(f'window\_autosize\_{format}\_{imageReadFlag}',cv2.WINDOW\_AUTOSIZE)  
 cv2.imshow(f'window\_autosize\_{format}\_{imageReadFlag}', img)  
 cv2.waitKey(0)  
 cv2.destroyAllWindows()  
 # cv2.setWindowProperty("Display window",cv2.WND\_PROP\_FULLSCREEN,cv2.WINDOW\_FULLSCREEN)  
 cv2.namedWindow(f'window\_normal\_{format}\_{imageReadFlag}', cv2.WINDOW\_NORMAL)  
 cv2.imshow(f'window\_normal\_{format}\_{imageReadFlag}', img)  
 cv2.waitKey(0)  
 cv2.destroyAllWindows()  
  
 cv2.namedWindow(f'window\_full\_{format}\_{imageReadFlag}', cv2.WINDOW\_NORMAL) cv2.setWindowProperty(f'window\_full\_{format}\_{imageReadFlag}',cv2.WND\_PROP\_FULLSCREEN,cv2.WINDOW\_FULLSCREEN)  
 cv2.imshow(f'window\_full\_{format}\_{imageReadFlag}', img)  
 cv2.waitKey(0)  
 cv2.destroyAllWindows()



Рисунок 1 – Пример открытия изображения с флагам grayscale и всеми флагами открытия окна



Рисунок 2 – Пример открытия изображения в rgb формате

1. Было отображено видео в окне и рассмотрены методы класса VideoCapture. Предпринята попытка отобразить видео в разных форматах, в частности размеры и цветовая гамма.

Для этого была написана функция, принимающая путь до файла, размеры фрейма для отображения и цветовую гамму. Для чтения видео используется класс vidoeCapture. В теле цикла чтения из видеопотока видео считывается покадрово с помощью метода read(). Затем к полученному кадру применяется изменения размеров и цветовой гаммы. Если кадров больше нет, то работа цикла заканчивается.

Код:

import cv2  
  
def processVideo(path,frameSize=(1920,1080),color=cv2.COLOR\_BGR2GRAY):  
 cap = cv2.VideoCapture(path, cv2.CAP\_ANY)  
  
 if not cap.isOpened():  
 print("Error opening video stream or file")  
 exit()  
  
 while True:  
 ret, frame = cap.read()  
 if not(ret):  
 break  
 updateFrame = cv2.resize(cv2.cvtColor(frame, color), frameSize)  
 cv2.imshow('frame', updateFrame)  
 if cv2.waitKey(1) & 0xFF == 27:  
 break  
 cap.release()

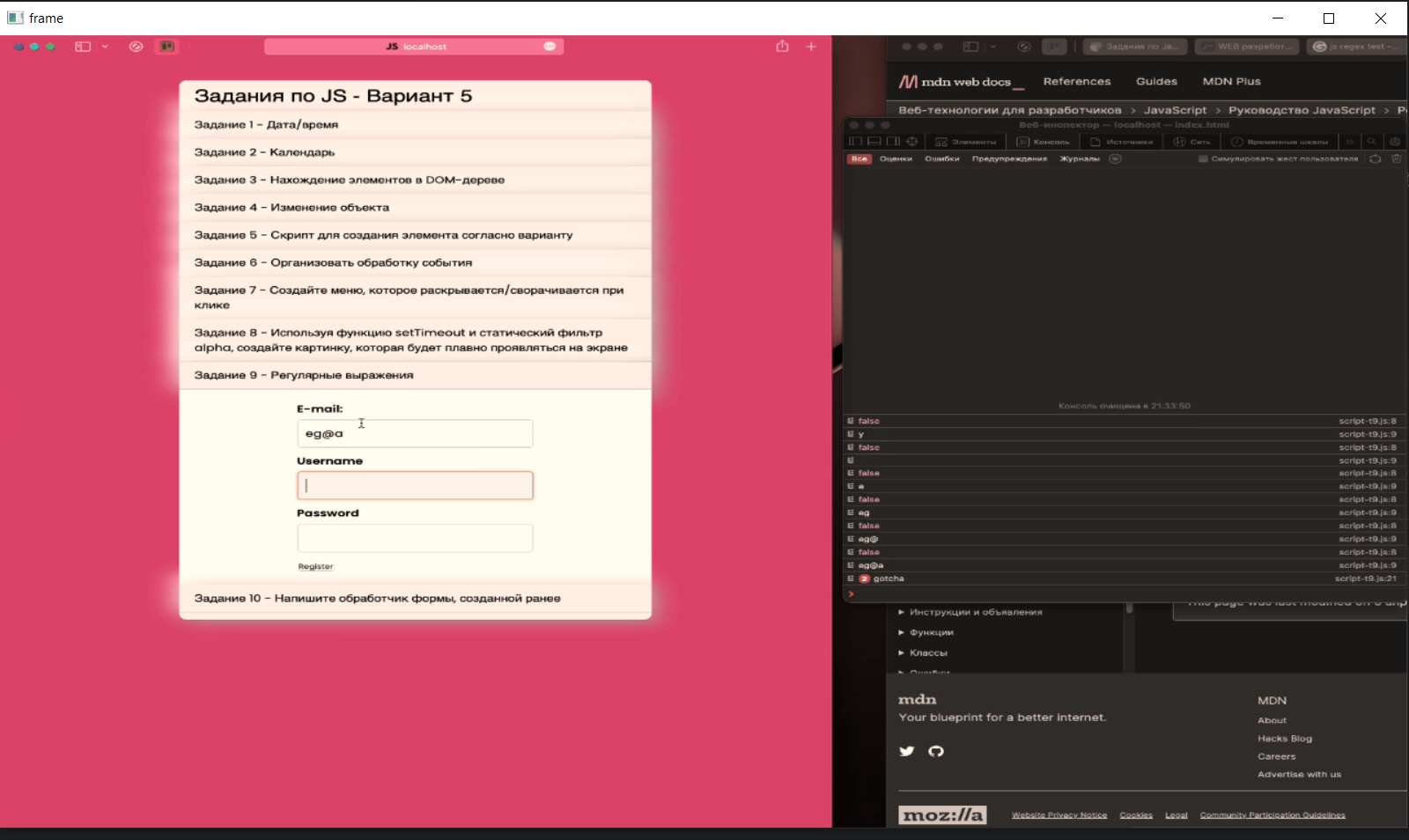


Рисунок 3 – Пример вывода видео в окне

1. Было записано видео из файла в другой файл. Для этого используется класс VideoWriter. Аналогично предыдущей задачи происходит получение кадра видео, затем с помощью класса VideoWriter(), принимающего путь до файла вывода, fourcc, количество фпс для записи и размер фрейма видео, происходит запись видео покадрово в файл с помощью метода write().

Код:

import cv2  
def readIPWriteTOFile():  
 cap = cv2.VideoCapture(r"video.mp4")  
 ok, img = cap.read()  
 w = int(cap.get(cv2.CAP\_PROP\_FRAME\_WIDTH))  
 h = int(cap.get(cv2.CAP\_PROP\_FRAME\_HEIGHT))  
 fourcc = cv2.VideoWriter\_fourcc(\*'mp4v')  
 video\_writer = cv2.VideoWriter("output.mp4", fourcc,fps=25, frameSize=(w, h))  
 while (True):  
 ok, img = cap.read()  
 if not(ok):  
 break  
 cv2.imshow('img', img)  
 video\_writer.write(img)  
 if cv2.waitKey(1) & 0xFF == ord('q'):  
 break  
 cap.release()  
 video\_writer.release()  
readIPWriteTOFile()

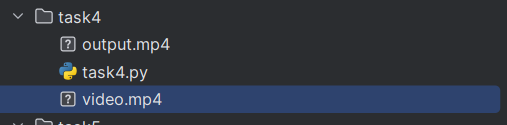


Рисунок 4 – Директория задачи после записи

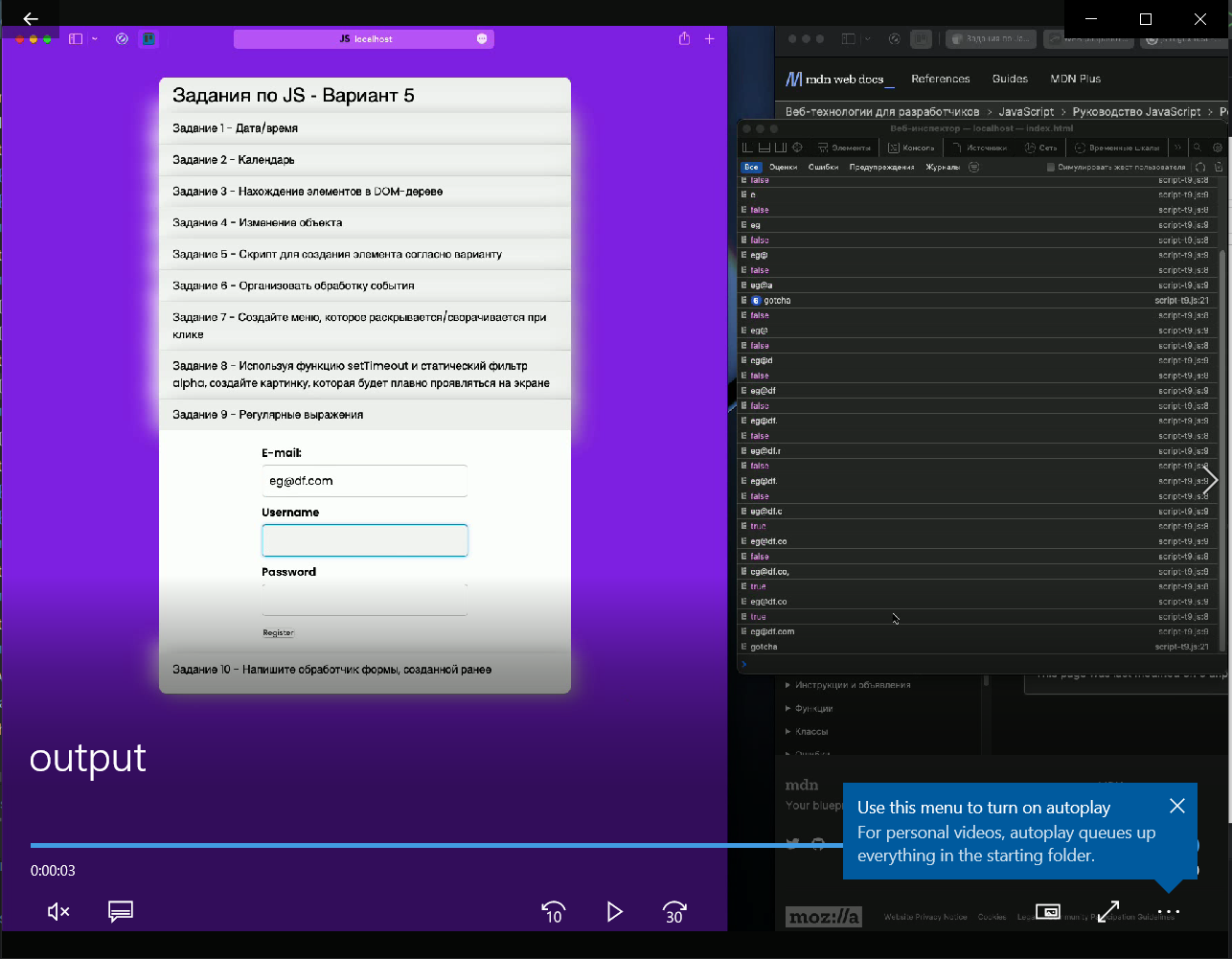


Рисунок 5 – Видео результат

1. Было прочитать изображение, переведено в формат HSV. Было выведено на экран два окна, в одном изображение в формате HSV, в другом – исходное изображение. Для этого к полученной с помощью метода imread() матрица изображения применяется конвертация цвета с помощью функции cv2.cvtColor(image,color).

Код:

import cv2  
  
def processToHsv(path):  
 img = cv2.imread(path)  
 updateImg = cv2.cvtColor(img, cv2.COLOR\_BGR2HSV)  
 print(updateImg)  
 print(img)  
 cv2.imshow('frame update', updateImg)  
 cv2.imshow('frame original', img)  
 cv2.waitKey(0)  
 cv2.destroyAllWindows()  
 img.release()  
  
processToHsv('red.png')

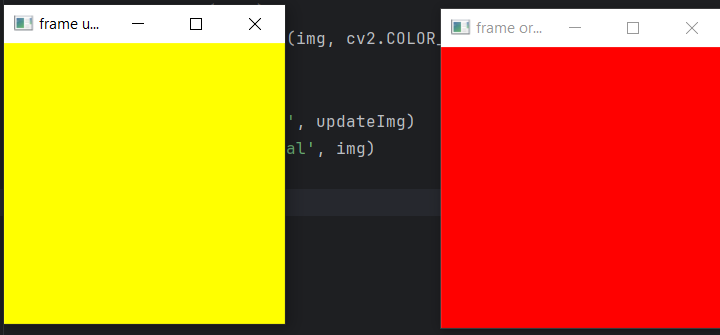


Рисунок 6 – Окна с оригинальным и переведенным изображениями

1. Было прочитано изображение с камеры. В центре на экране выведен крест. Информация с вебкамеры была записана видео в файл, Крест был залит одним из 3 цветов – красный, зеленый, синий по следующему правилу: НА ОСНОВАНИИ ФОРМАТА RGB определить, центральный пиксель ближе к какому из цветов красный, зеленый, синий и таким цветом заполнить крест.

Для этого VideoCapture() был передан id=0 для включения режима чтения с камеры. Затем размер изображения с камеры был установлен с помощью методов set класса VideoCapture(). Для записи видео используется класс VideoWriter() аналогично предыдущему заданию. Для отрисовки креста в центре происходит вычисления центрального пикселя с помощь поиска половины ширины и высоты фрейма видео. Затем используется функция cv2.rectangle() для отрисовки прямоугольника по заданным координатам.

Код:

def print\_cam(rectWidth,rectHeight,frameSize=(640,480)):  
 cap = cv2.VideoCapture(0)  
 cap.set(cv2.CAP\_PROP\_FRAME\_WIDTH,frameSize[0])  
 cap.set(cv2.CAP\_PROP\_FRAME\_HEIGHT,frameSize[1])  
  
 fourcc = cv2.VideoWriter\_fourcc(\*'mp4v')  
 videoWriter = cv2.VideoWriter("output.mp4", fourcc, fps=25, frameSize=(frameSize[0], frameSize[1]))  
 while True:  
 ret, frame = cap.read()  
  
  
 # Вычисляем центральные координаты  
 centerX = frameSize[0] // 2  
 centerY = frameSize[1] // 2  
  
 #Цвет центрального пикселя  
 centralPixel = frame[centerY, centerX]  
  
 #Определяем цвет  
 blue, green, red = centralPixel # BGR формат  
 if red > green and red > blue:  
 fillColor = (0, 0, 255) # Красный  
 elif green > red and green > blue:  
 fillColor = (0, 255, 0) # Зеленый  
 else:  
 fillColor = (255, 0, 0) # Синий  
  
 #Рисуем крест  
 cv2.rectangle(frame, (centerX - rectWidth // 2, centerY - rectHeight // 2),  
 (centerX + rectWidth // 2, centerY + rectHeight // 2), fillColor, -1)  
  
 cv2.rectangle(frame, (centerX - rectHeight // 2, centerY - int(rectWidth\*0.7)),  
 (centerX+rectHeight//2, centerY-rectHeight//2), fillColor, -1)  
 cv2.rectangle(frame, (centerX - rectHeight // 2, centerY+rectHeight//2),  
 (centerX + rectHeight // 2, centerY + int(rectWidth\*0.7)), fillColor, -1)  
  
  
 #gray = cv2.cvtColor(frame, cv2.COLOR\_BGR2GRAY)  
 cv2.imshow('frame',frame)  
 videoWriter.write(frame)  
 if cv2.waitKey(1)&0xFF==27:  
 break  
 cap.release()  
 videoWriter.release()  
 cv2.destroyAllWindows()

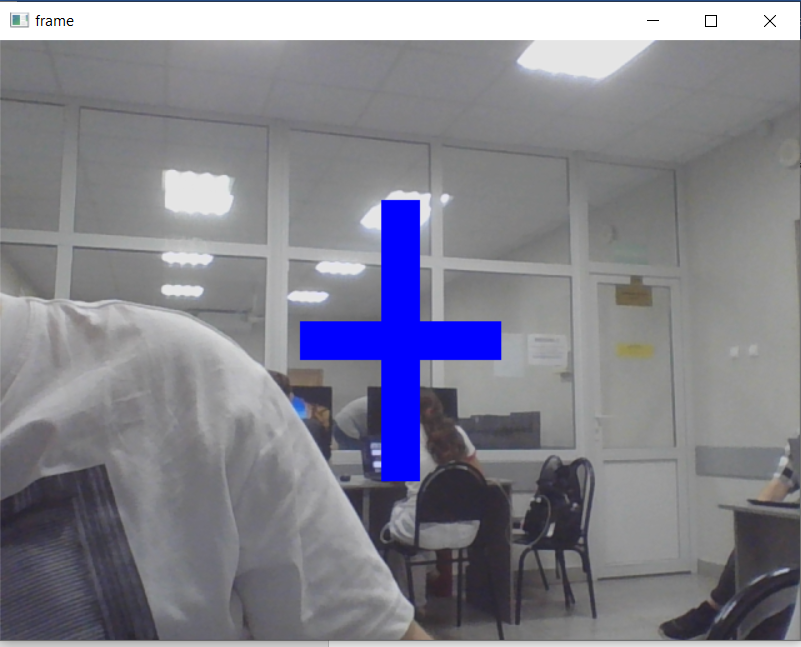


Рисунок 8 – Пример работы программы

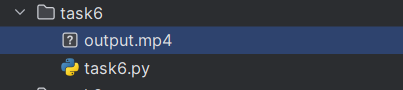


Рисунок 9 – Директория после работы программы

9. Подключен телефон, подключена его камера и выведено на экран видео с камеры. Для этого была использована функция из предыдущей задачи с единственным отличием: на вход конструктору VideoCapture() передается ip камеры.

Код:

import cv2  
def print\_cam(rectWidth,rectHeight,frameSize=(640,480)):  
 #ipwebcam  
 cap = cv2.VideoCapture("http://192.168.43.1:8080/video")  
 fourcc = cv2.VideoWriter\_fourcc(\*'mp4v')  
 videoWriter = cv2.VideoWriter("output.mp4", fourcc, fps=25, frameSize=(frameSize[0], frameSize[1]))  
 while True:  
 ret, frame = cap.read()  
 frame = cv2.resize(frame, frameSize)  
  
 # Вычисляем центральные координаты  
 centerX = frameSize[0] // 2  
 centerY = frameSize[1] // 2  
  
 #Цвет центрального пикселя  
 centralPixel = frame[centerY, centerX]  
  
 #Определяем цвет  
 blue, green, red = centralPixel # BGR формат  
 if red > green and red > blue:  
 fillColor = (0, 0, 255) # Красный  
 elif green > red and green > blue:  
 fillColor = (0, 255, 0) # Зеленый  
 else:  
 fillColor = (255, 0, 0) # Синий  
  
 #Рисуем крест  
 cv2.rectangle(frame, (centerX - rectWidth // 2, centerY - rectHeight // 2),  
 (centerX + rectWidth // 2, centerY + rectHeight // 2), fillColor, -1)  
  
 cv2.rectangle(frame, (centerX - rectHeight // 2, centerY - int(rectWidth\*0.7)),  
 (centerX+rectHeight//2, centerY-rectHeight//2), fillColor, -1)  
 cv2.rectangle(frame, (centerX - rectHeight // 2, centerY+rectHeight//2),  
 (centerX + rectHeight // 2, centerY + int(rectWidth\*0.7)), fillColor, -1)  
  
 #gray = cv2.cvtColor(frame, cv2.COLOR\_BGR2GRAY)  
 cv2.imshow('frame',frame)  
 videoWriter.write(frame)  
 if cv2.waitKey(1)&0xFF==27:  
 break  
 cap.release()  
 videoWriter.release()  
 cv2.destroyAllWindows()  
print\_cam(160,30)

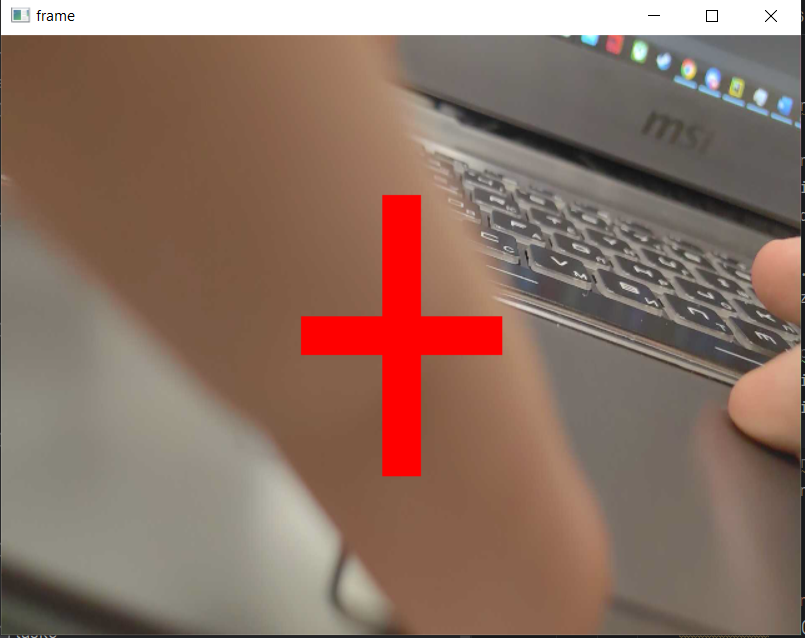


Рисунок 10 – Пример работы программы

**Вывод:** Были изучены основы библиотеки opencv.