Яндекс

УрФУ
Институт радиоэлектроники и информационных технологий — РТФ

Машинное зрение

и библиотека OpenCV

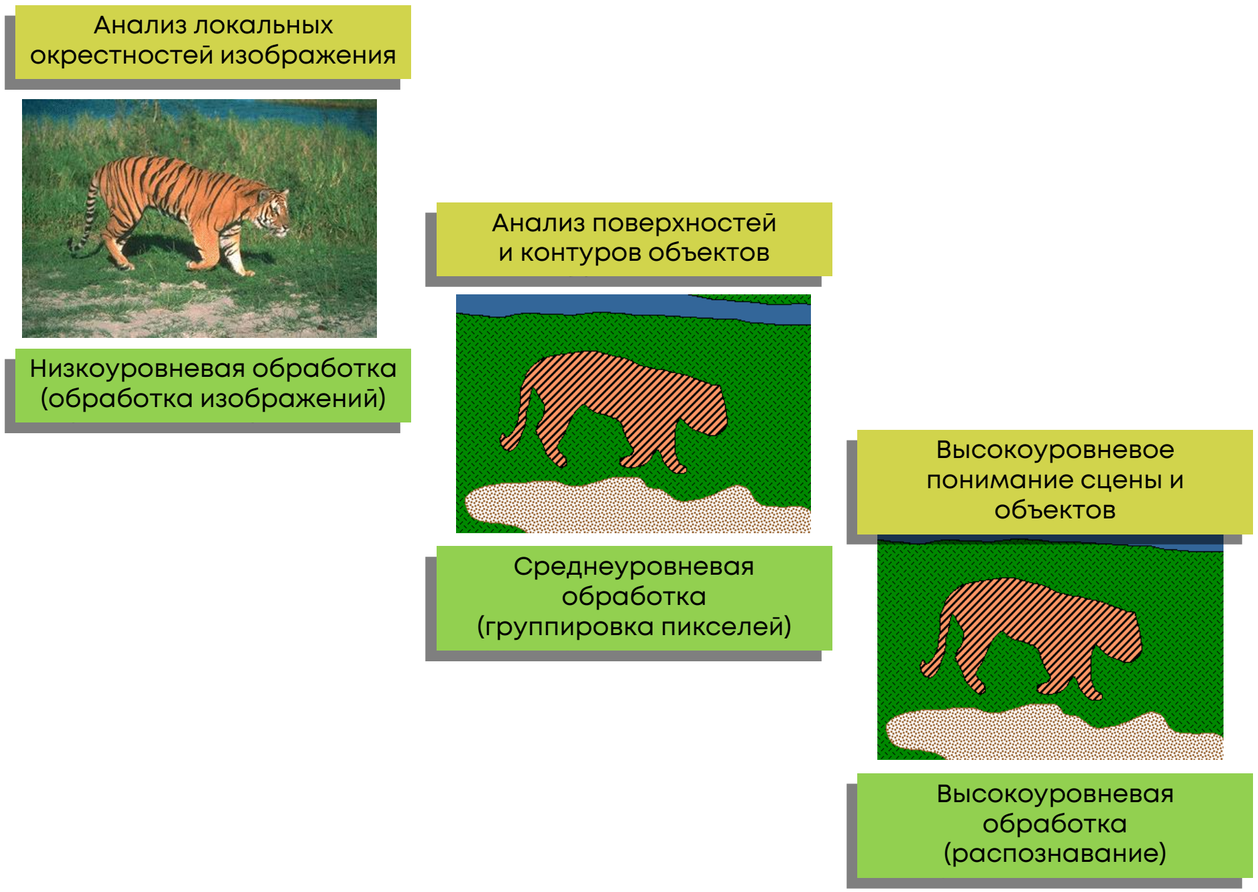
Яндекс

УрФУ
Институт радиоэлектроники и информационных технологий — РТФ

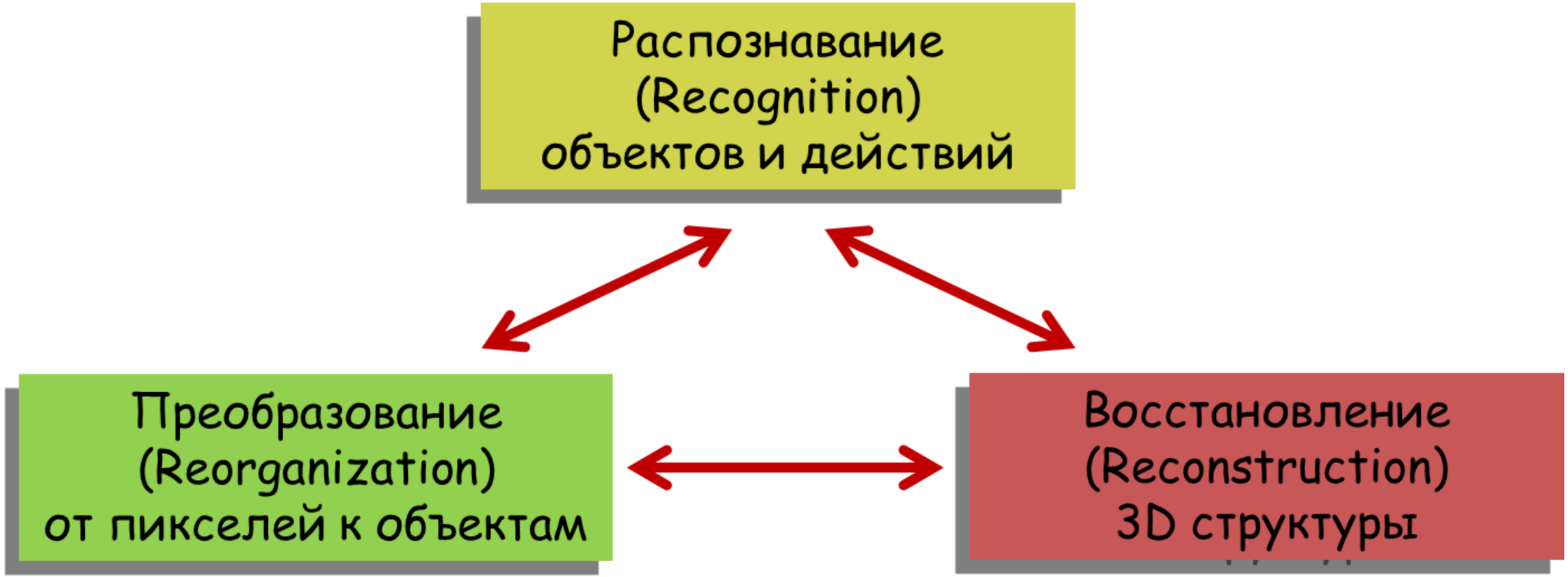
Уровни и методы компьютерного зрения

Уровни обработки данных:

- **Низкий уровень** (попиксельная обработка) – на входе изображение на выходе изображение (фильтрация простых шумов, гистограммная обработка);
- **Средний уровень** (сегментирование) – на входе изображение на выходе векторная информация, описание в виде элементов и связей между ними (сегментация);
- **Высокий уровень** (распознавание) – на входе описание в терминах элементов изображения на выходе семантическая интерпретация в терминах видимой сцены объектного мира.



Компьютерное зрение в современной трактовке (Д. Малик)

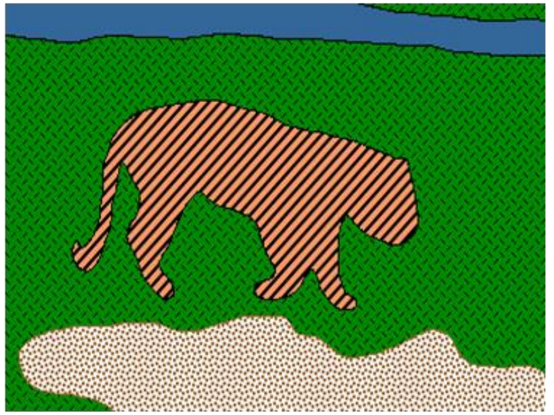


Изображения

Полутоновое

Многоспектральное

Бинарное



Маркированное

Яндекс

УрФУ
Институт радиоэлектроники и информационных технологий — РТФ

Поиск координат красного кубика: основные этапы

Этап 0: Подготовить видео

Этап 1: Низкоуровневая обработка изображения, бинаризация, фильтрация шумов

Этап 2: Средне уровневая обработка, выделение контуров, сегментация, фильтрация по геометрическим характеристикам

Этап 3: Вычисление координат центра красного кубика

Яндекс

УрФУ
Институт радиоэлектроники и информационных технологий — РТФ

Этап 0: Запись видео в файл

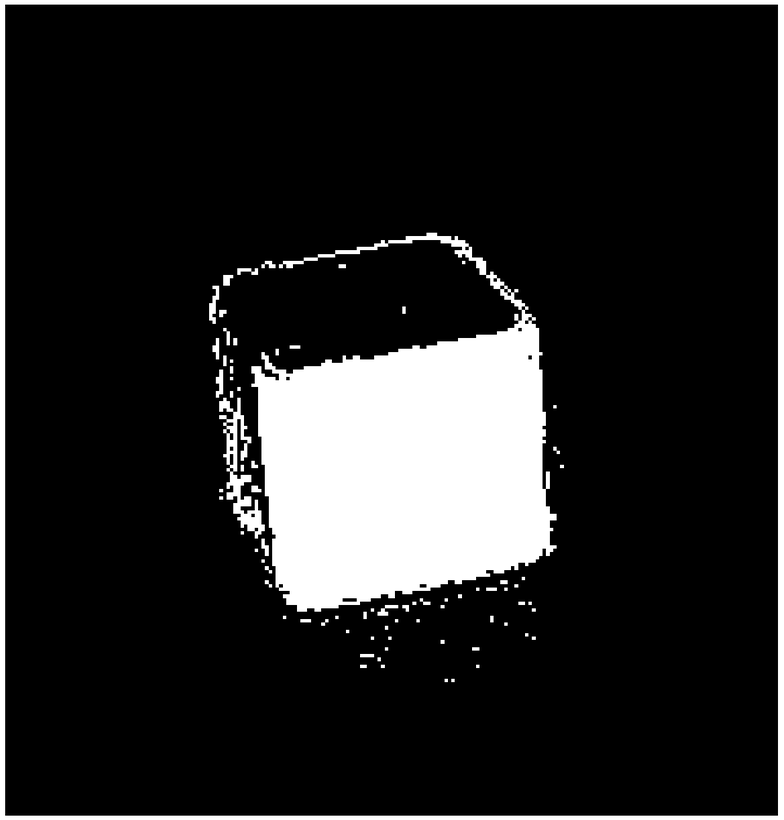
```python
import cv2
import numpy as np

cap = cv2.VideoCapture('http://10.32.21.66:8080/?action=stream')
fourcc = cv2.VideoWriter.fourcc(*'MP4V')
out = cv2.VideoWriter('robotCam_tst.avi', fourcc, 20.0, (640, 480))

while True:
    ret, frame = cap.read()
    #gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    out.write(frame)
    cv2.imshow( winname: 'video feed', frame)
    #cv2.imshow('gray feed', gray)

    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

cap.release()
out.release()
cv2.destroyAllWindows()
```

Яндекс

УрФУ
Институт радиоэлектроники и информационных технологий — РТФ

Этап 1: Бинаризация

```
mask = cv2.inRange(frame, lowerb: (minb, ming, minr), upperb: (maxb, maxg, maxr))
```



Яндекс

УрФУ
Институт радиоэлектроники и информационных технологий — РТФ

Бинарные изображения

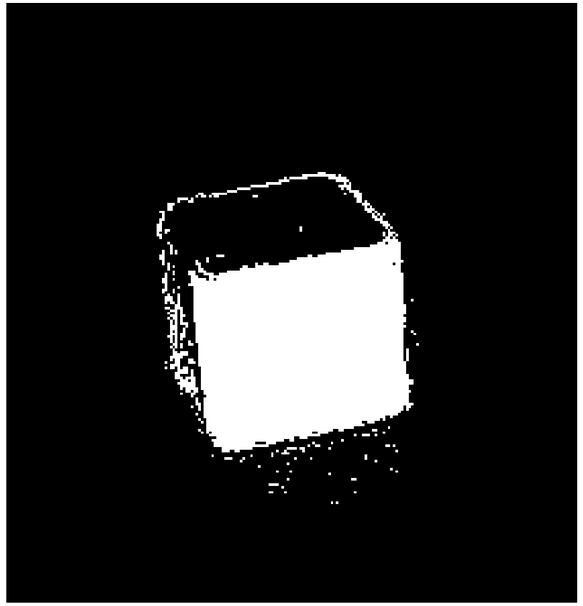
Яндекс

УрФУ
Институт радиоэлектроники и информационных технологий — РТФ

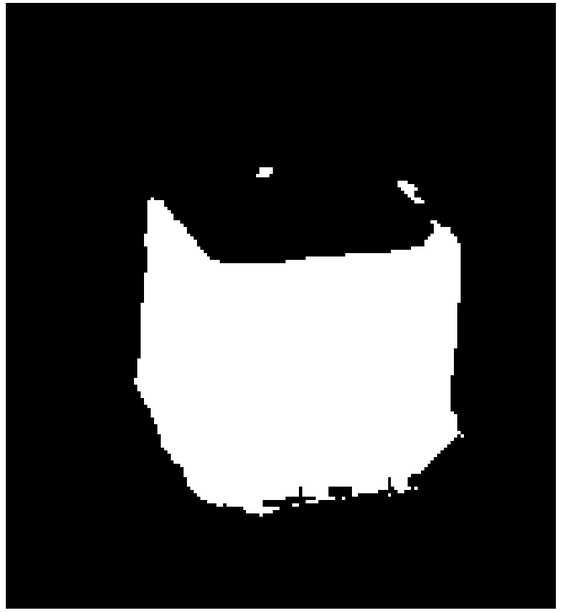
Этап 1: Фильтрация шумов. Размытие (Blur)

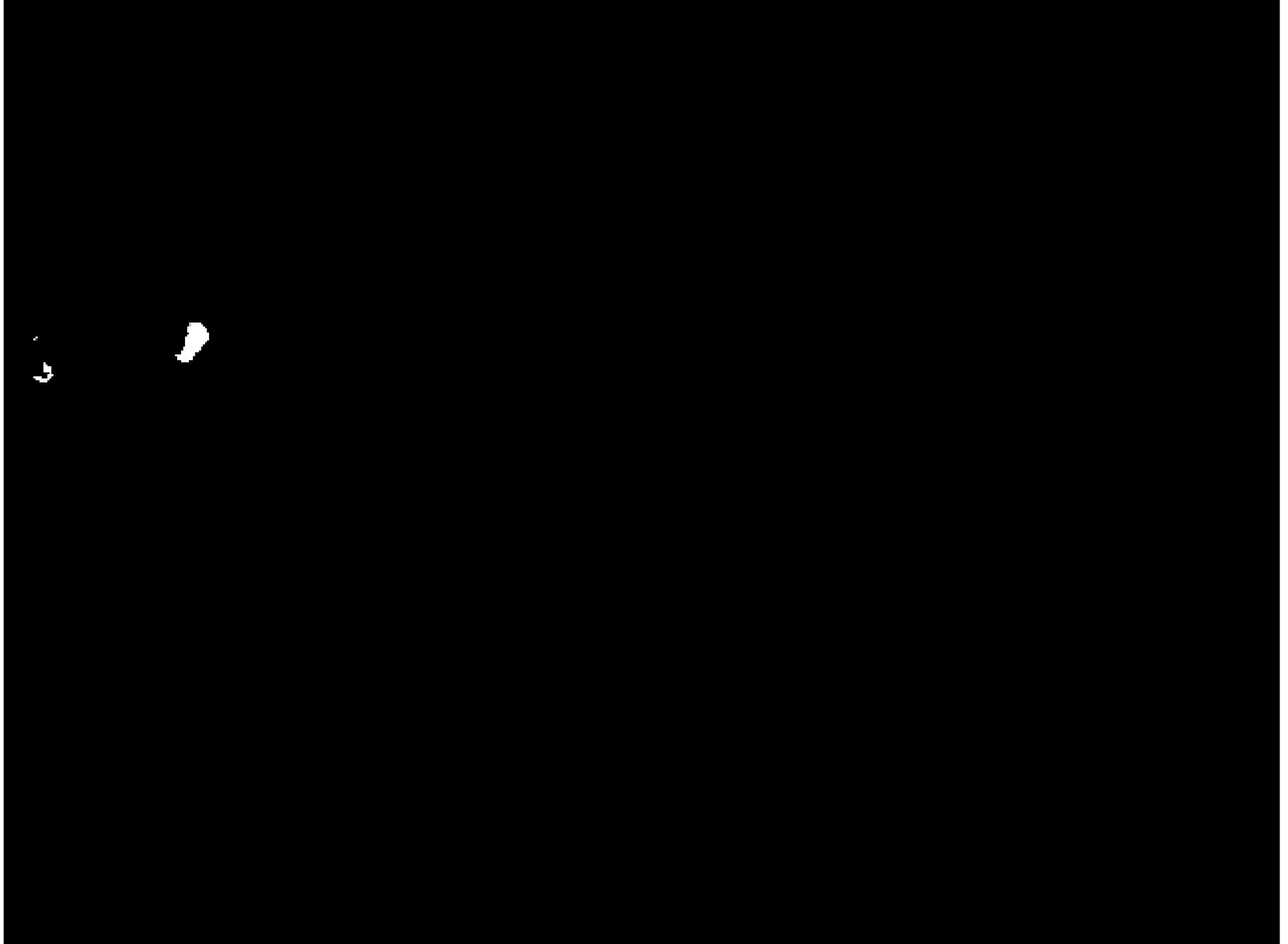
```
frame = cv2.medianBlur(frame, 1+blurVal*2)
mask = cv2.inRange(frame,  lowerb: (minb, ming, minr),  upperb: (maxb, maxg, maxr))
```

До



После





Яндекс

УрФУ
Институт радиоэлектроники и информационных технологий — РТФ

Принцип работы медианного размытия

0	0	0	0	0	0
0	90	90	90	90	0
0	90	90	90	90	0
0	90	0	90	90	0
0	90	90	90	90	0
0	0	0	0	0	0

Фрагмент исходного изображения

1/9*

1	1	1
1	1	1
1	1	1

Пример ядра медианного фильтра

X	X	X	40	40	40
X	40	60	40	40	40
X	50	80	40	40	40
30	30	30	80	60	X
30	30	30	50	40	X
30	30	30	X	X	X

Фрагмент обраб. изображения

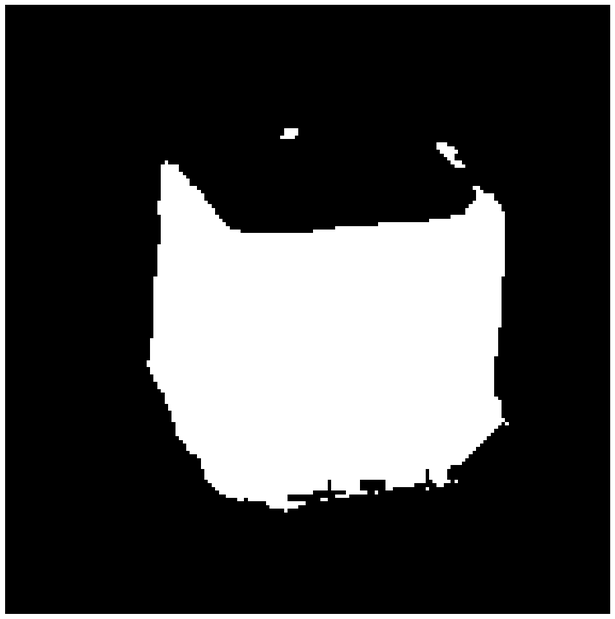
Яндекс

УрФУ
Институт радиоэлектроники и информационных технологий — РТФ

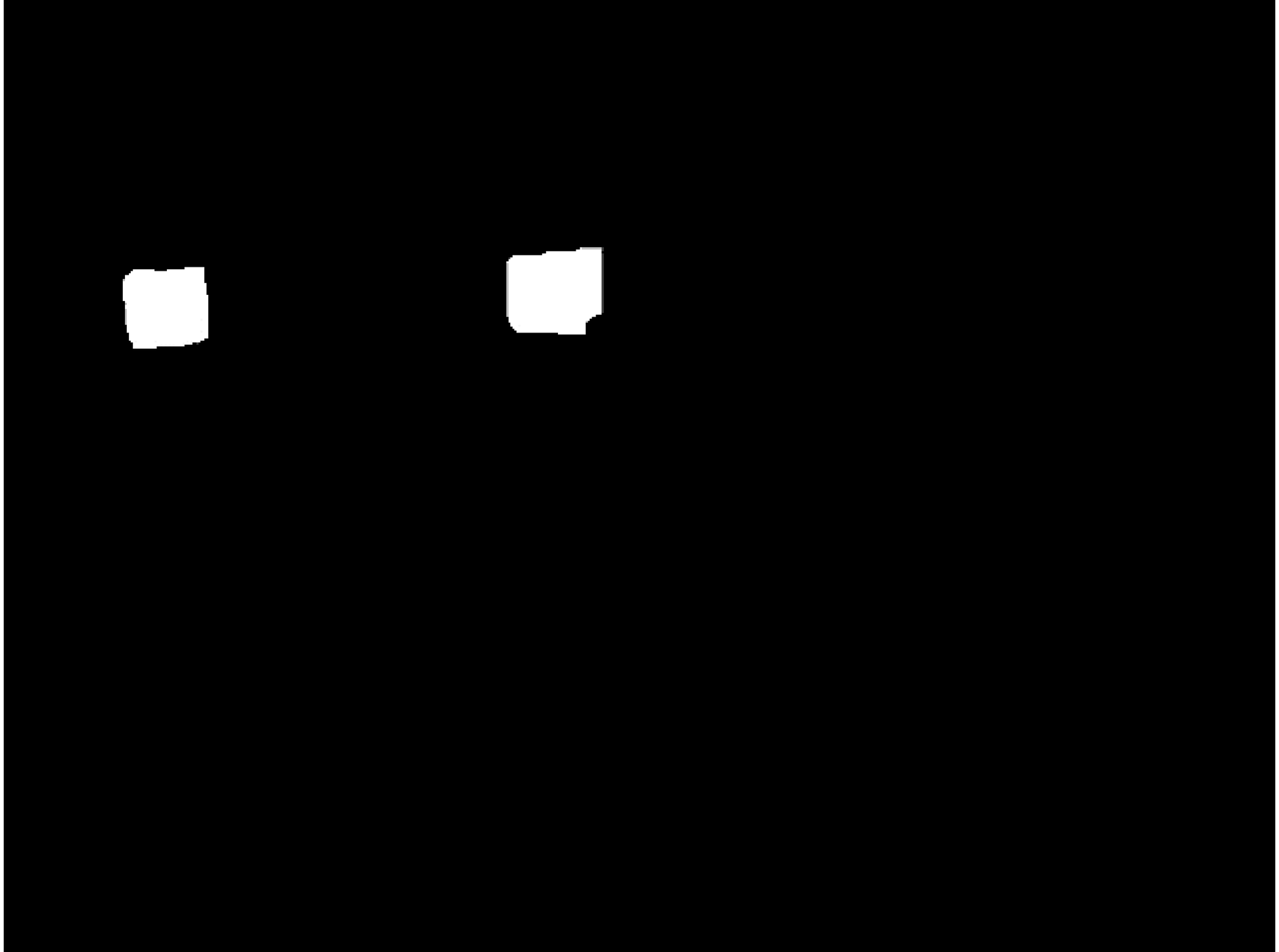
Этап 1: Улучшение через морфологические операции

```
frame = cv2.medianBlur(frame, 1+blurVal*2)
mask = cv2.inRange(frame, lowerb: (minb, ming, minr), upperb: (maxb, maxg, maxr))
kernel = cv2.getStructuringElement(cv2.MORPH_RECT, ksize: (10, 10))
mask = cv2.morphologyEx(mask, cv2.MORPH_OPEN, kernel, iterations=1)
```

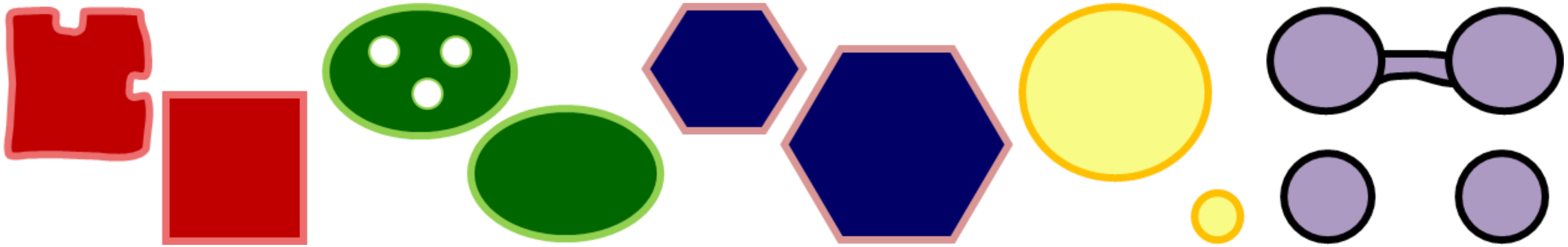
До



После



Морфология бинарных изображений



Наращивание

Эрозия

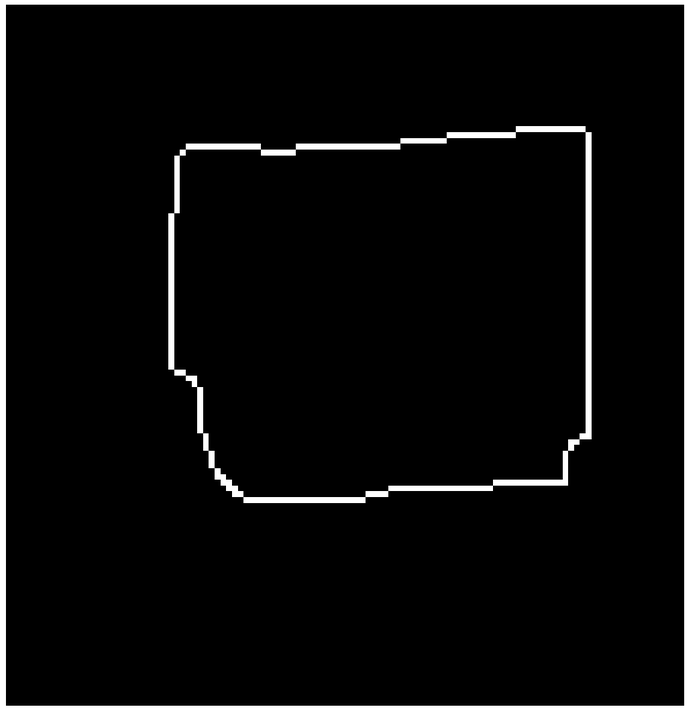
Замыкание

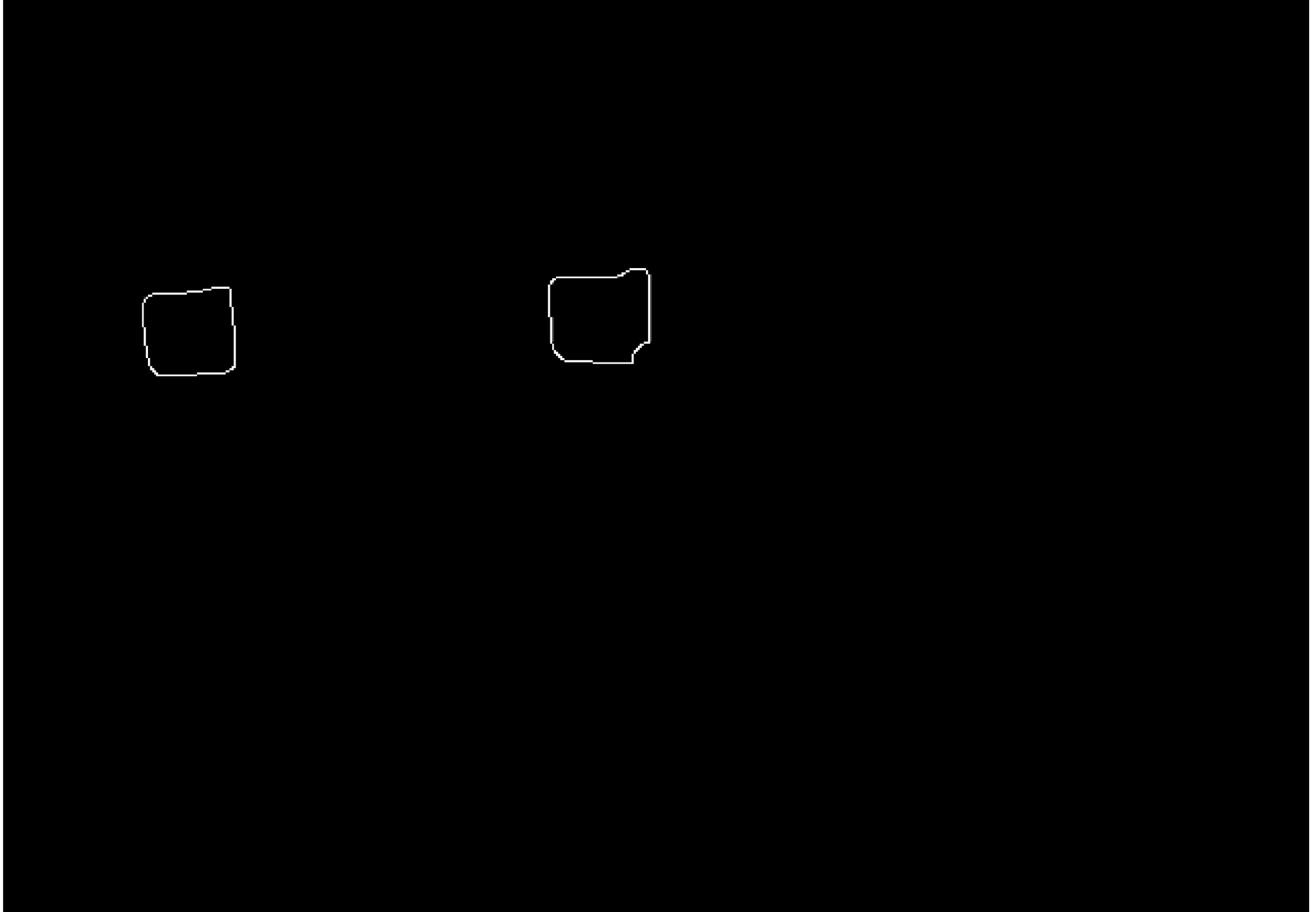
Размыкание

Этап 2: Обнаружение краев, контуров объекта

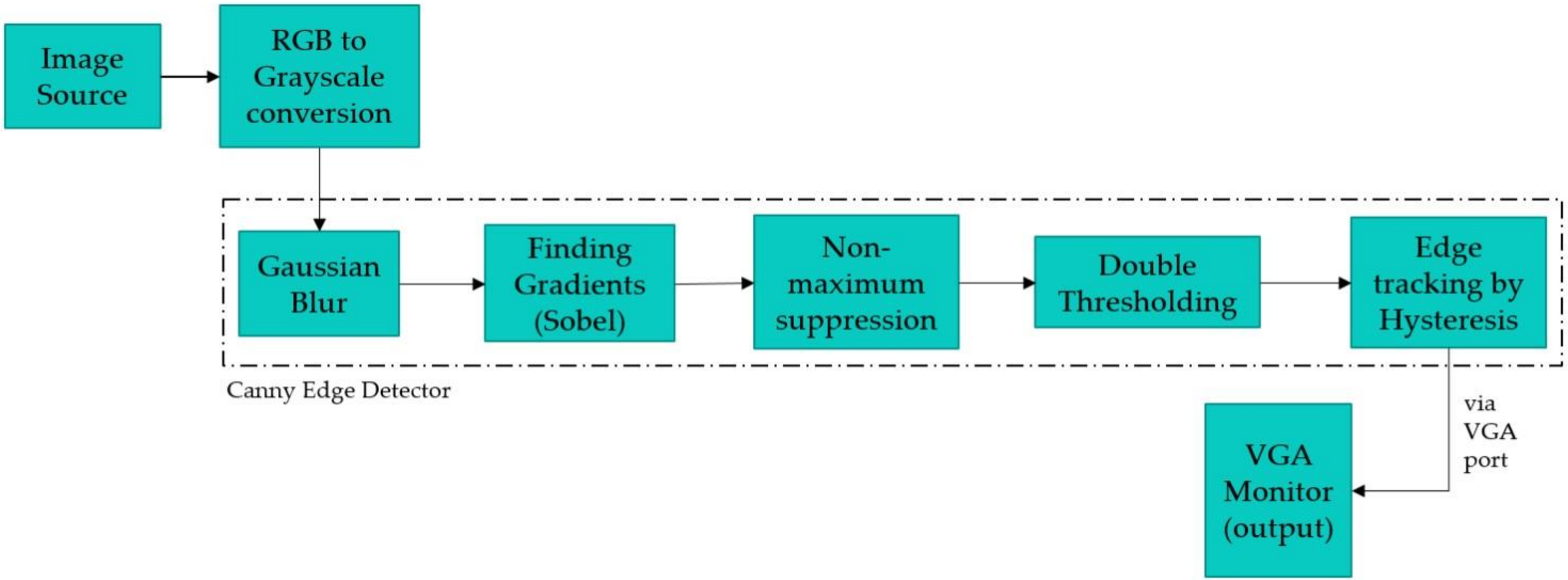
До

После





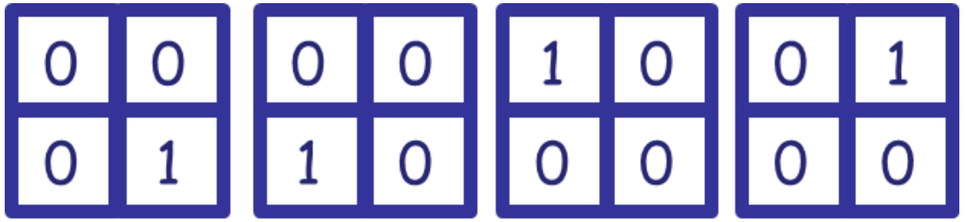
Функция обнаружения краёв Canny



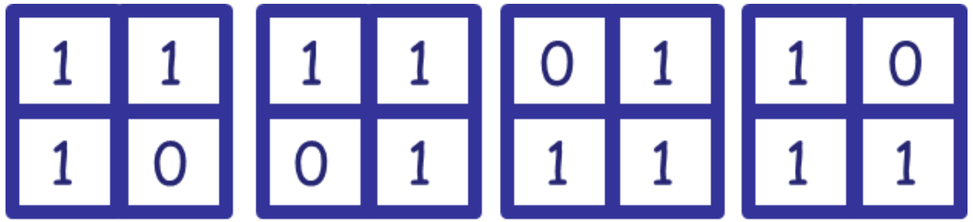
Яндекс

УрФУ
Институт радиоэлектроники и информационных технологий — РТФ

Пример алгоритма поиска углов



Маски, описывающие внешние углы



и внутренние углы

Этап 2: Трассировка и анализ контуров с findContours()

```
contours, hierarchy = cv2.findContours(edged, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_NONE)
contours = sorted(contours, key=cv2.contourArea, )
for i in range(len(contours)):
    if  cv2.contourArea(contours[i]) > 2000  :
        cv2.drawContours(frame, contours[i], -1,  color: (0, 255, 0),  thickness: 3)
```

Яндекс

УрФУ
Институт радиоэлектроники и информационных технологий — РТФ

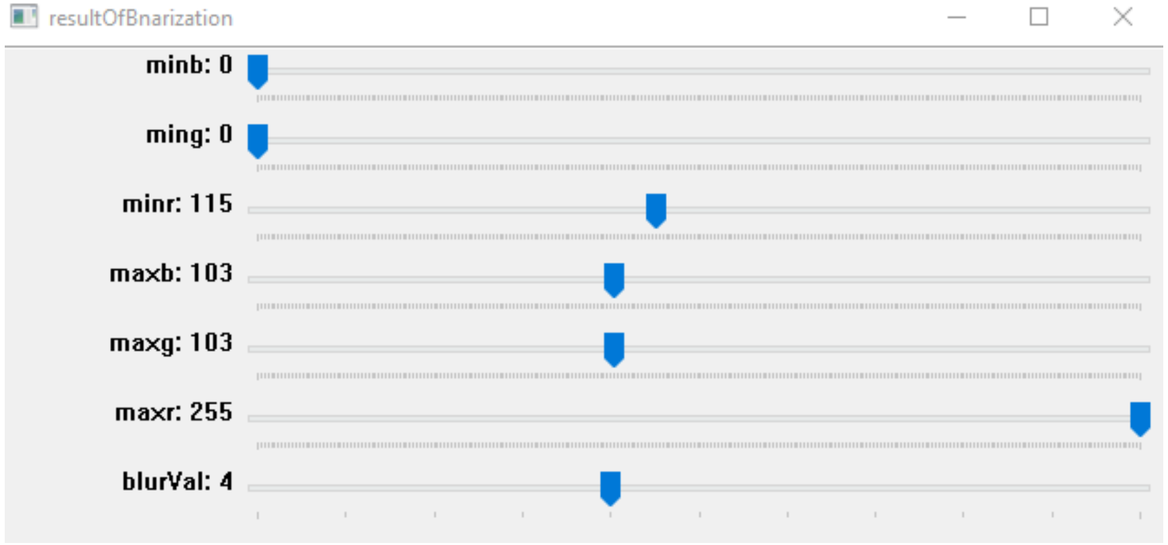
Этап 3: Вычисление координат с moments()

```
if  cv2.contourArea(contours[i]) > 2000  :
  cv2.drawContours(frame, contours[i], -1, color: (0, 255, 0), thickness: 3)
  M = cv2.moments(contours[i])
  if M['m00'] != 0:
    cx = int(M['m10'] / M['m00'])
    cy = int(M['m01'] / M['m00'])
  cv2.putText(frame, str(i) + '_(' + str(cx) + ':' + str(cy) + ')', org: (cx + 10, cy - 40),
              cv2.FONT_HERSHEY_SIMPLEX, fontScale: 1, color: (255, 255, 255), thickness: 1)
```

Яндекс

УрФУ
Институт радиоэлектроники и информационных технологий — РТФ

Полезные функции: слайдеры для настройки параметров




```
cv2.createTrackbar( trackbarName: 'minb', windowName: 'resultOfBnarization', value: 0, count: 255, nothing)
cv2.createTrackbar( trackbarName: 'ming', windowName: 'resultOfBnarization', value: 0, count: 255, nothing)

while(True):
  _, frame = videoCapture.read()
  minb = cv2.getTrackbarPos( trackbarname: 'minb', winname: 'resultOfBnarization')
  ming = cv2.getTrackbarPos( trackbarname: 'ming', winname: 'resultOfBnarization')
```

Яндекс

УрФУ
Институт радиоэлектроники и информационных технологий — РТФ

Характеристики камеры

Модель	TC-C32XN I3/E/Y/2.8MM/V5.1
Тип камеры	IP
Цвет изображения	цветная
Тип конструкции камеры	купольная
Исполнение	уличная
Тип крепления	потолочная
Тип матрицы	CMOS
Размер матрицы	1/2.9"
Разрешение камеры	2 Мп
Фокусное расстояние	2.8 мм

Характеристики съемки

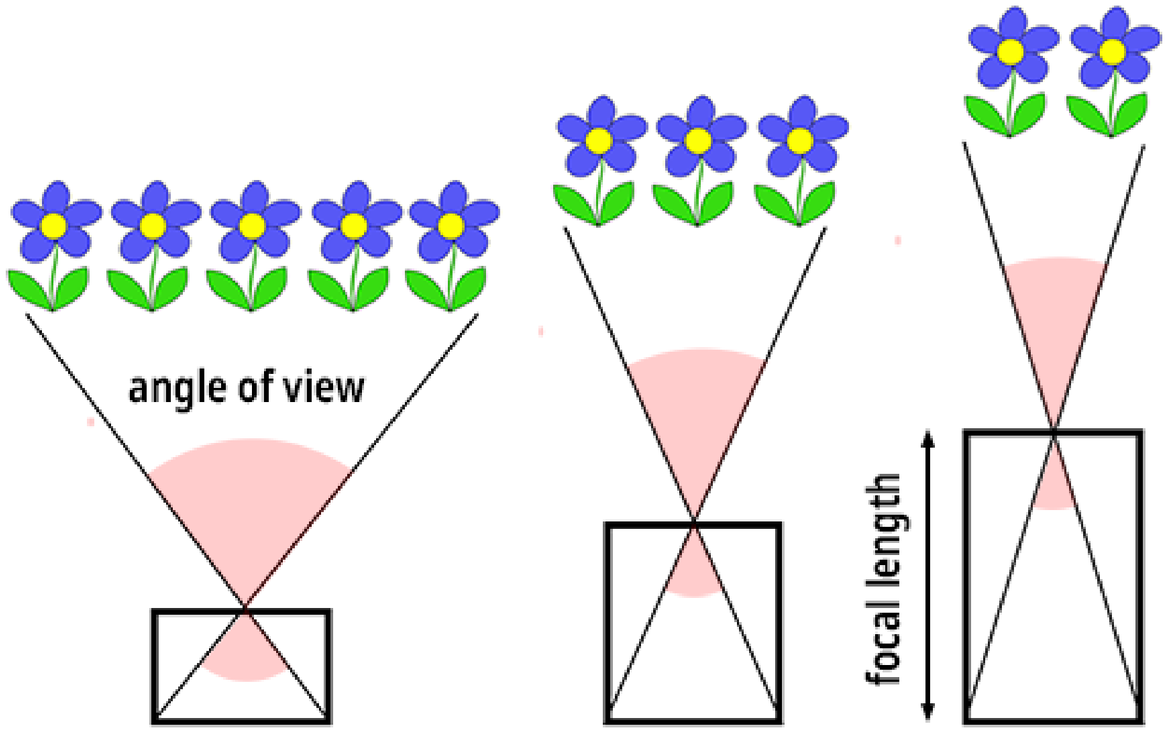
Разрешение	1920 x 1080
Угол обзора, макс	102.8 °
Максимальное разрешение видеозаписи	1080p
Скорость передачи видео	25 кадр/с

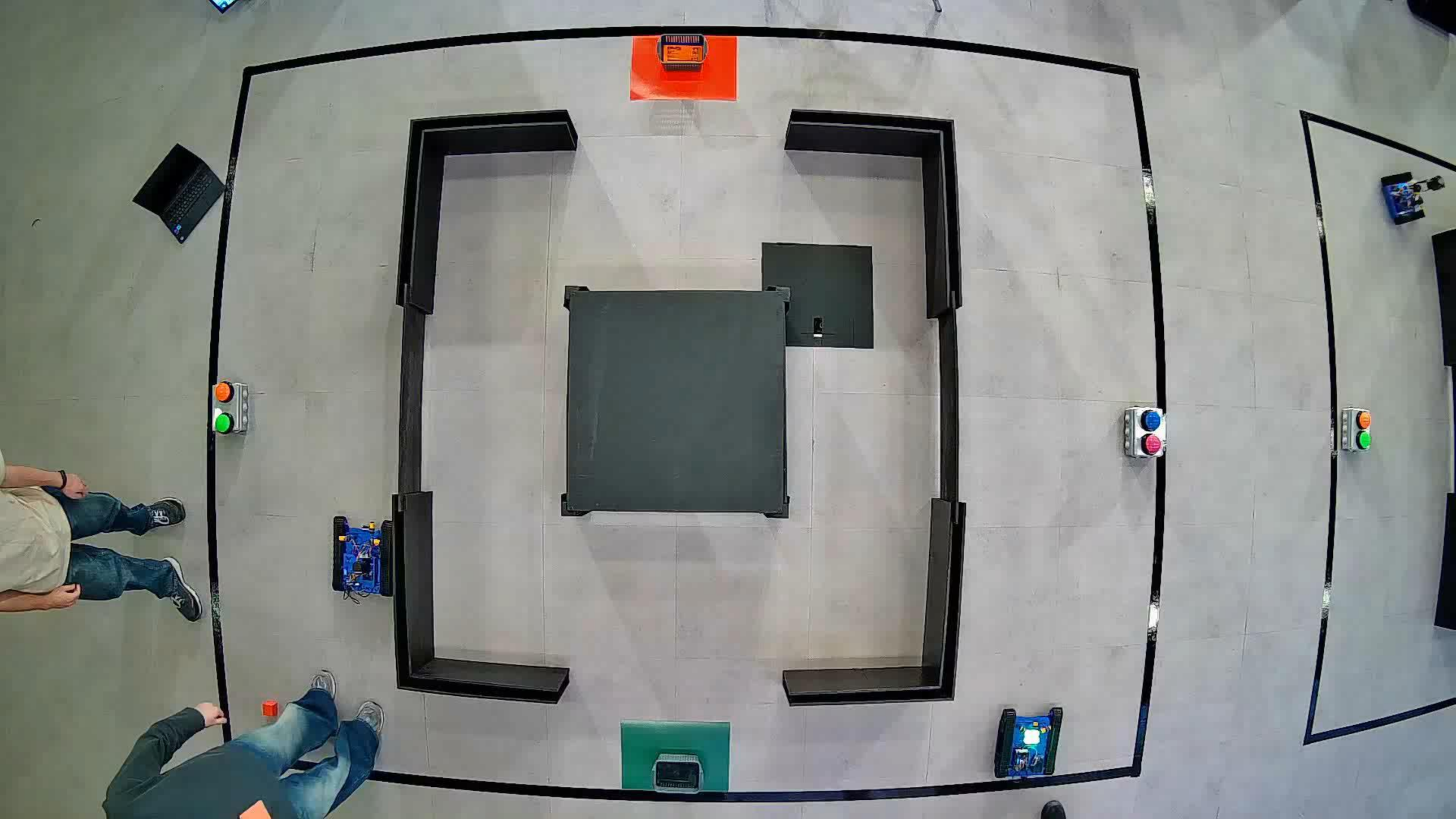
Яндекс

УрФУ
Институт радиоэлектроники и информационных технологий — РТФ

Фокусное расстояние

Размер матрицы	1/2.9"	Угол обзора, макс	102.8 °
Разрешение камеры	2 Мп		
Фокусное расстояние	2.8 мм		

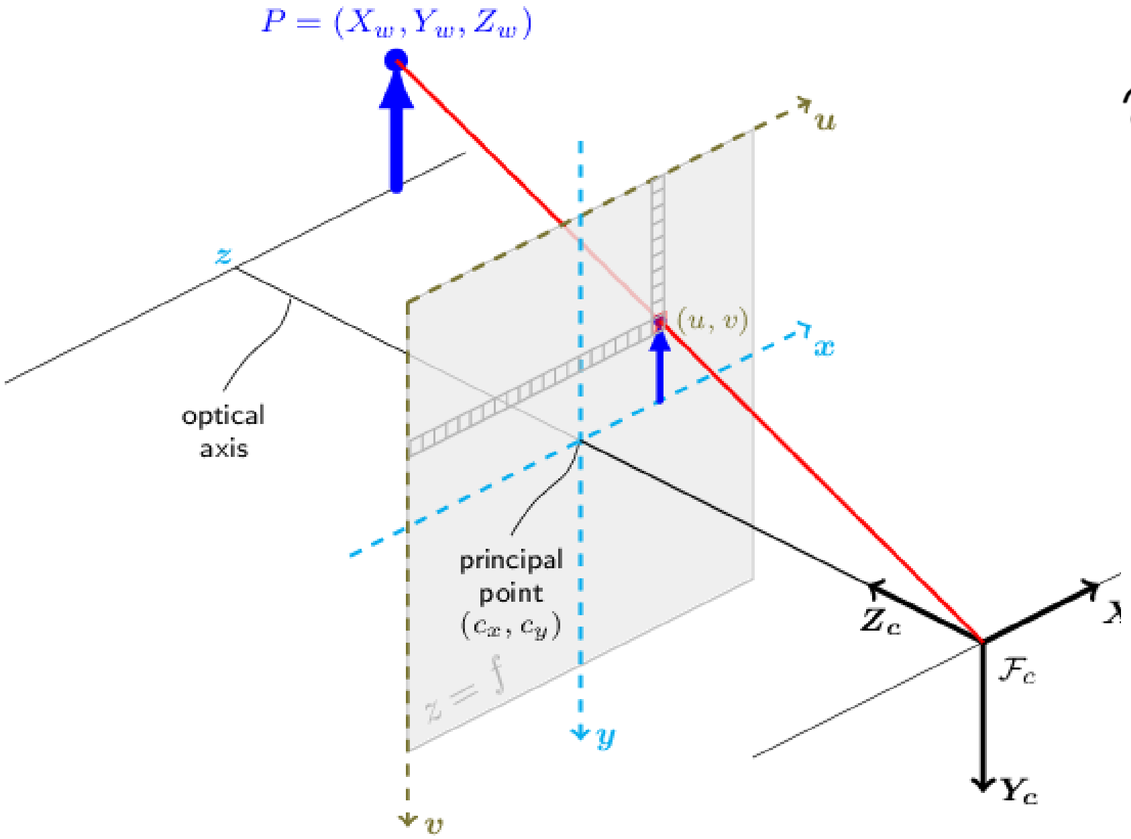




Яндекс

УрФУ
Институт радиоэлектроники и информационных технологий – РТФ

Модель камеры в OpenCV



$$u = \frac{f_x x}{z} + c_x, v = \frac{f_y y}{z} + c_y$$

$$camera\ matrix = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

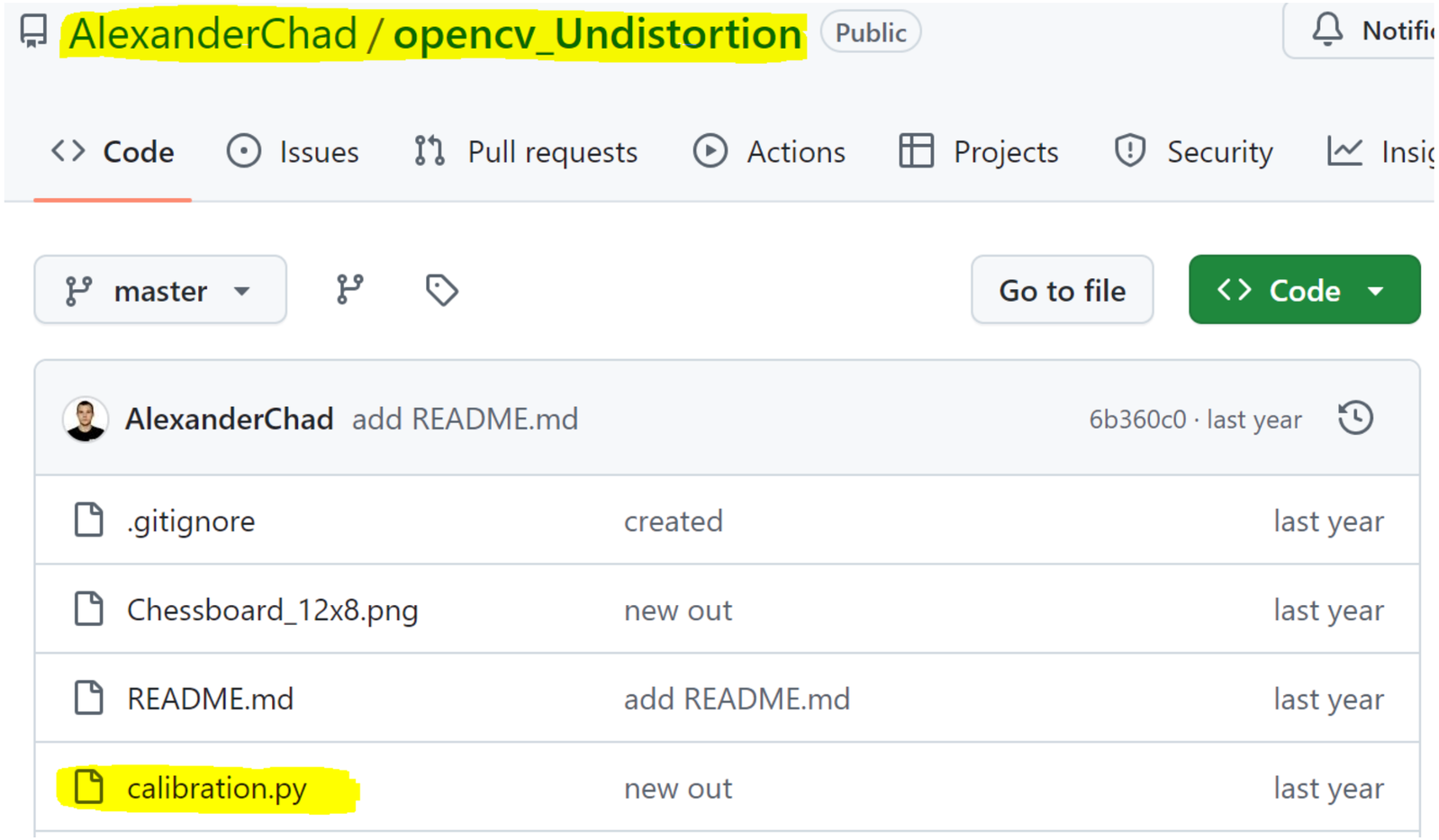
$$Distortion\ coefficients = (k_1 \quad k_2 \quad p_1 \quad p_2 \quad k_3)$$

Калибровочная доска

Яндекс

УрФУ
Институт радиоэлектроники и информационных технологий — РТФ

Калибруем камеру



Яндекс

УрФУ
Институт радиоэлектроники и информационных технологий — РТФ

Калибруем камеру

```
chessboardSize = (9, 6)

# termination criteria
criteria = (cv2.TERM_CRITERIA_EPS + cv2.TERM_CRITERIA_MAX_ITER, 30, 0.001)

# prepare object points, like (0,0,0), (1,0,0), (2,0,0) ....,(6,5,0)
objp = np.zeros( shape: (chessboardSize[0] * chessboardSize[1], 3), np.float32)
objp[:, :2] = np.mgrid[0:chessboardSize[0],
                       0:chessboardSize[1]].T.reshape(-1, 2)

img_names = glob('.\calib\*.png')
```

Яндекс

УрФУ
Институт радиоэлектроники и информационных технологий — РТФ

Калибруем камеру

```
processing 0.0%, image: .\calib\1.png... ok
processing 6.25%, image: .\calib\vlcsnap-2024-10-14-23h32m05s293.png... ignore
processing 12.5%, image: .\calib\vlcsnap-2024-10-14-23h32m17s924.png... ok
processing 18.75%, image: .\calib\vlcsnap-2024-10-14-23h33m11s770.png... ignore
processing 25.0%, image: .\calib\vlcsnap-2024-10-14-23h33m20s808.png... ignore
processing 31.25%, image: .\calib\vlcsnap-2024-10-14-23h33m29s803.png... ok
processing 37.5%, image: .\calib\vlcsnap-2024-10-14-23h33m43s135.png... ok
processing 43.75%, image: .\calib\vlcsnap-2024-10-14-23h34m08s422.png... ok
processing 50.0%, image: .\calib\vlcsnap-2024-10-14-23h34m16s918.png... ignore
processing 56.25%, image: .\calib\vlcsnap-2024-10-14-23h34m24s731.png... ok
```

$$(k_1 \quad k_2 \quad p_1 \quad p_2 \quad k_3)$$

```
RMS: 0.7685980218277246

camera_matrix = np.array([[1.18271958e+03, 0.00000000e+00, 9.27035425e+02],
       [0.00000000e+00, 1.18623664e+03, 6.09522211e+02],
       [0.00000000e+00, 0.00000000e+00, 1.00000000e+00]])
dist_coefs = np. array([-0.39994649,  0.22596037,  0.0006536 ,  0.00137182, -0.08901383])
```

$$\begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

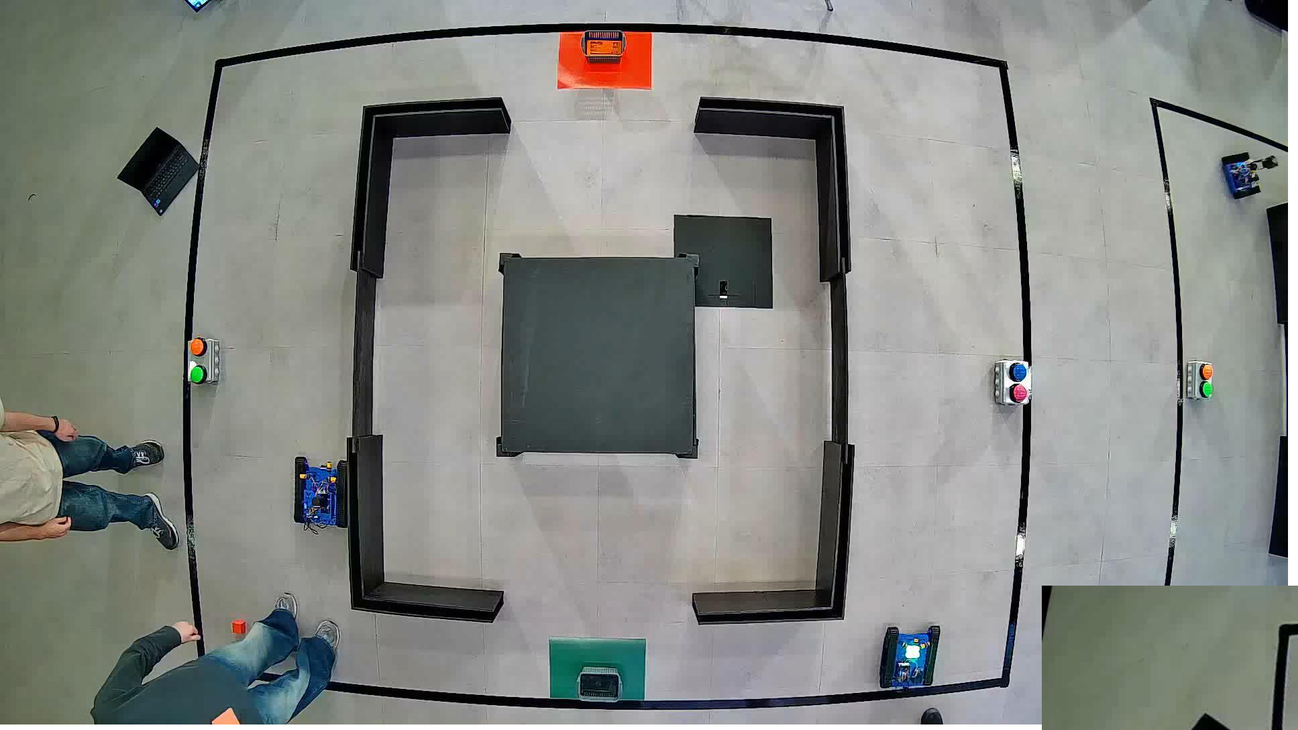
Яндекс

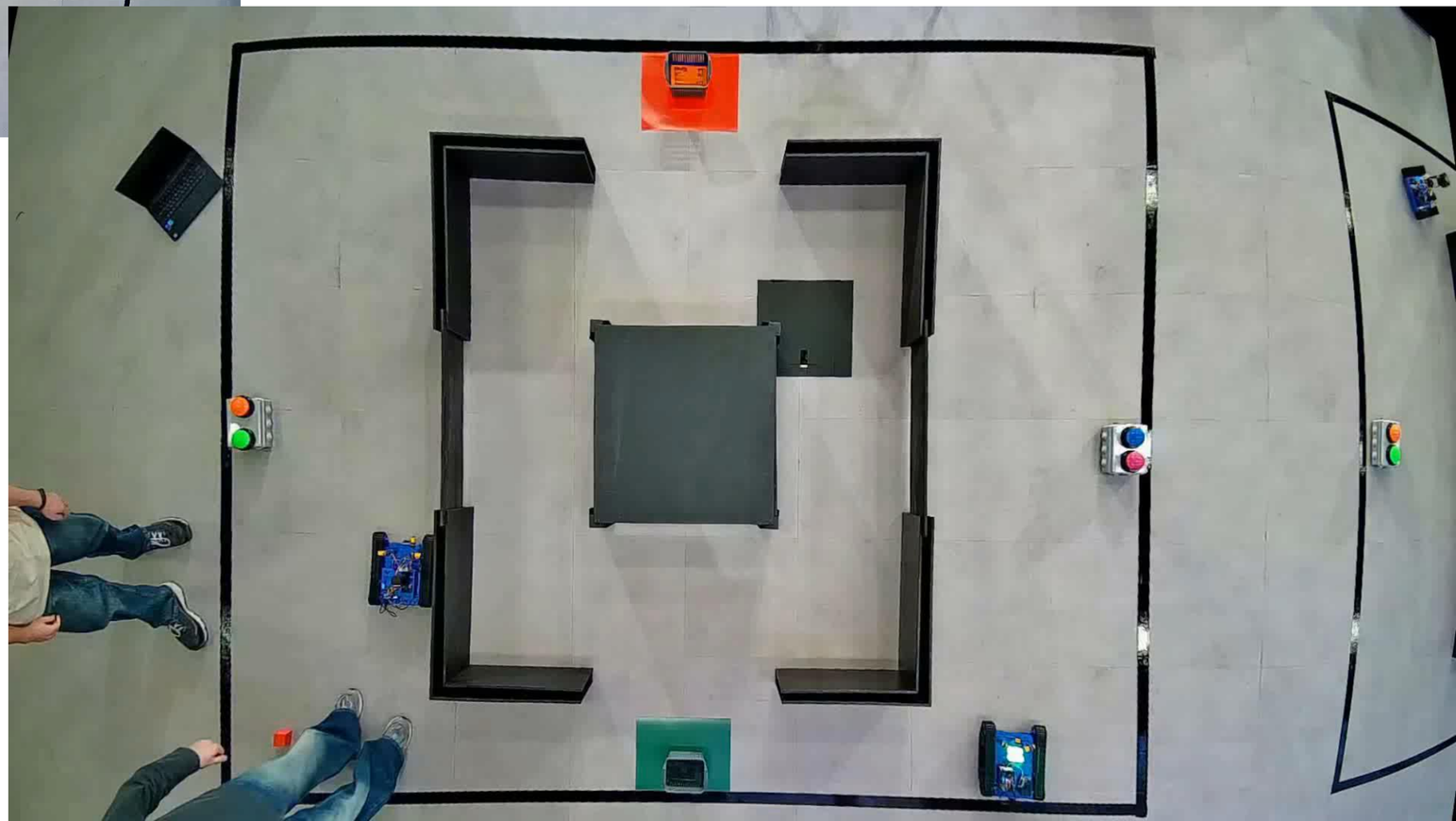
УрФУ
Институт радиоэлектроники и информационных технологий — РТФ

Применяем калибровку к видеопотоку

```
camera_matrix = np.array( object: [[1182.719, 0, 927.03],
                           [0, 1186.236, 609.52],
                           [0, 0, 1]], dtype=np.float32)

dist_coeffs = np.array( object: [-0.5, 0.3, 0, 0, 0], dtype=np.float32)

# Загрузить видео
video_input_file = 'Right_1.avi'
```

```
# Исправление искажения
frame_undistorted = cv2.undistort(frame, camera_matrix, dist_coeffs,  dst: None, new_camera_matrix)
# Опционально: обрежьте до ROI, если необходимо
x, y, w, h = roi
frame_undistorted = frame_undistorted[y:y+h, x:x+w]
# Изменение размера обратно к исходному (если требуется)
frame_undistorted = cv2.resize(frame_undistorted,  dsize: (960, 540))
# Запись в выходной файл
cv2.imshow( winname: "FishEye", frame_undistorted)  # Показываем обработанный кадр в окне
cv2.waitKey(1)
```





Яндекс

УрФУ
Институт радиоэлектроники и информационных технологий — РТФ

Спасибо за внимание!

Пелевин Владимир

к.пед.н., доцент, УрФУ
инженер, ООО Микроэлектроника и Робототехника
v.n.pelevin@mirrobotics.ru

Яндекс

УрФУ
Институт радиоэлектроники и информационных технологий — РТФ

Понравилось занятие?

Сканируй QR код и оставляй свой отзыв