National Research University Higher School of Economics

Graduate School of Business

Programme «Big Data Systems»

**Project Report**

Defect detection with computer vision

Performed by student:

Victor E. Kozlovskiy .

(Name)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*(Signature)*

**Project supervisor:**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*(position, full name)*

*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

*(grade)*

*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

*(Date) (signature)*

**Moscow 2021**

## Defect detection with computer vision

Initial requirements:

* Web-camera
* Python
* OpenCV library

Two basic libraries are used in this work to create computer-vision script: NumPy and OpenCV.

**NumPy** is the fundamental package for scientific computing in Python. It is a Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much more.

**OpenCV** (Open Source Computer Vision Library) is an open-source library that includes several hundreds of computer vision algorithms. Originally developed by Intel. The library is cross-platform and free for use under the open-source Apache 2 License. Starting with 2011, OpenCV features GPU acceleration for real-time operations.

**IMutils** is also required to make basic image processing functions such as translation, rotation, resizing, skeletonization, displaying Matplotlib images, sorting contours, detecting edges etc.

All needed packages are installed with pip manager:

pip install numpy

pip install opencv-python

pip install imutils

### Task 1

Dynamically recognize arrangement of simple objects in relation to their initial position. Indicate when a book is rotated at angle greater than some specified value.

To solve the problem the following approach is proposed:

1. Recognize objects using basic library operators
2. Recognize angle of object’s tilt
3. Capture video stream from web-camera
4. Create graphical output indicating expected changes

First we read image (figure 1) from static file, convert it to grayscale and blur to get rid of high-frequenced noises (figure 2):

gray = cv.cvtColor(image, cv.COLOR\_BGR2GRAY)

gray = cv.GaussianBlur(gray, (15, 15), 0)

cv.imwrite("output/gray.jpg", gray)



Figure 1. Original static image



Figure 2. Blurred image

Sobel operator (figure 3) showed more recognizable edges than Laplace operator (figure 4).



Figure 3. Result of Sobel operator



Figure 4. Result of Laplace operator

Get edged image using Canny operator:

edged = cv.Canny(gray, 10, 250)

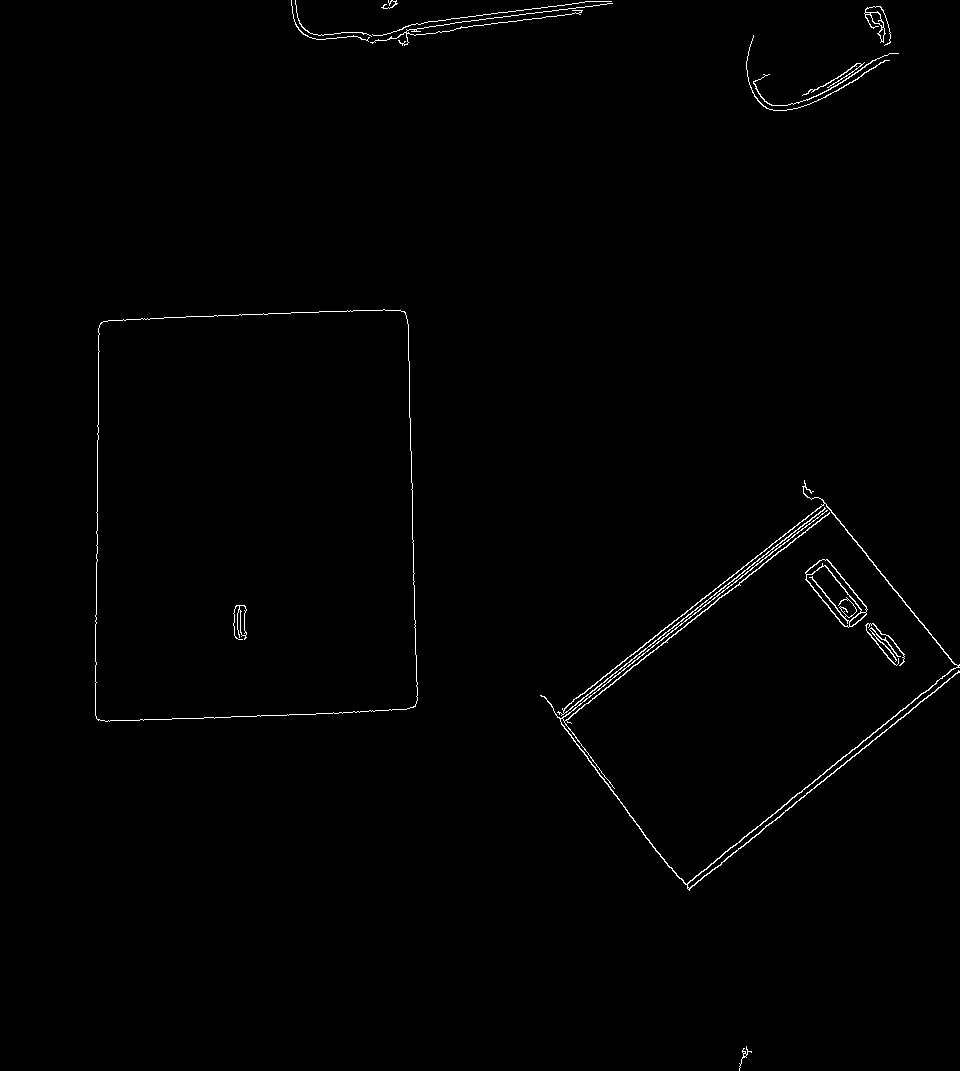


Figure 5. Result of Canny operator

Make obtained contours closed:

kernel = cv.getStructuringElement(cv.MORPH\_RECT, (7, 7))

closed = cv.morphologyEx(edged, cv.MORPH\_CLOSE, kernel)

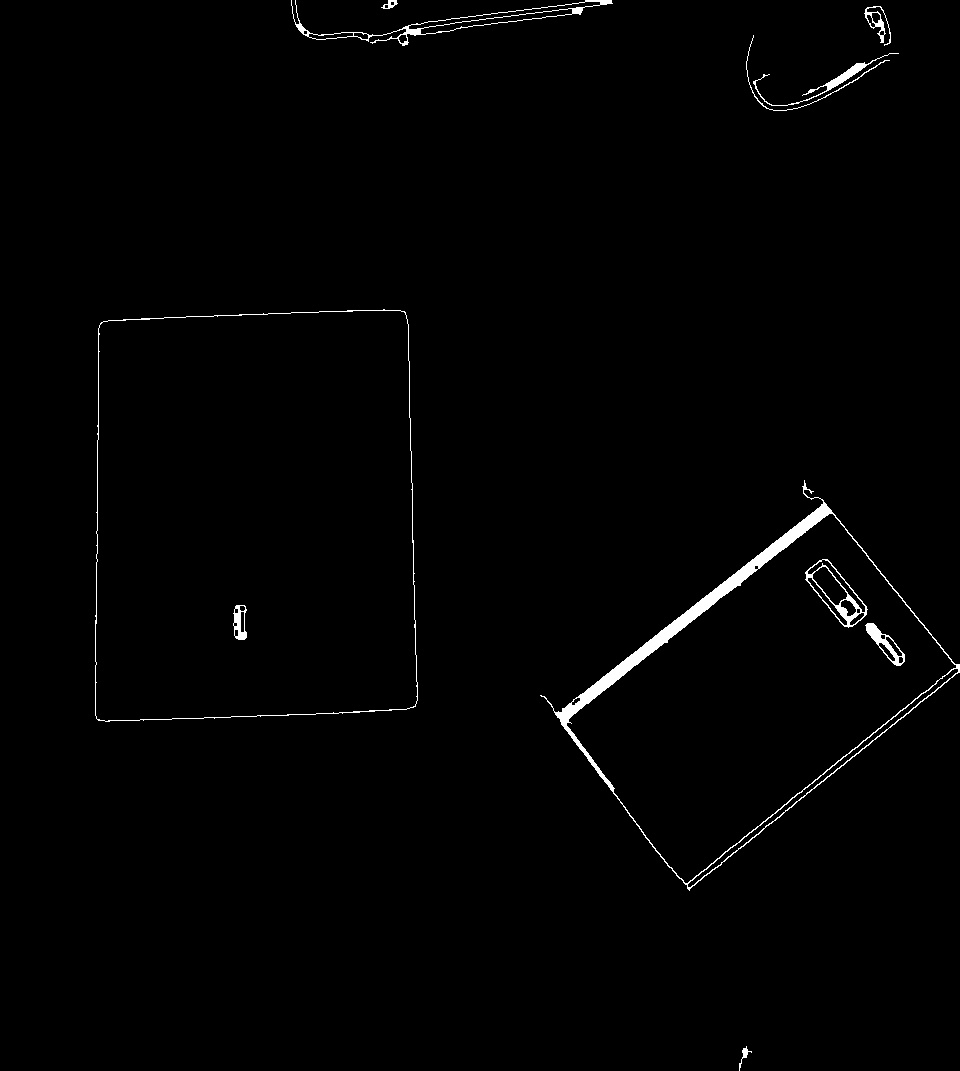


Figure 6

Find separate contours with IMutils:

cnts = cv.findContours(closed.copy(), cv.RETR\_EXTERNAL, cv.CHAIN\_APPROX\_SIMPLE)

cnts = imutils.grab\_contours(cnts)

Then approximate each of found contours to closed polygon, approximate central line for ones that have 4 vertices, and calculate its tilt angle.

for c in cnts:

    peri = cv.arcLength(c, True)

    approx = cv.approxPolyDP(c, 0.1 \* peri, True)

    if (len(approx) in [4]):

        cv.drawContours(image, [approx], -1, (0, 255, 0), 4)

        rows,cols = image.shape[:2]

        [vx,vy,x,y] = cv.fitLine(c, cv.DIST\_L2, 0, 0.0001, 0.0001)

        lefty = int((-x\*vy/vx) + y)

        righty = int(((cols-x)\*vy/vx)+y)

        line\_start = (cols-1,righty)

        line\_end = (0,lefty)

        cv.line(image, line\_start, line\_end, (0,255,0), 2)

        angle\_pi = (righty - lefty)/(cols-1)

        angle = int(np.arctan(angle\_pi)\*180/np.pi)

        text\_start = (int(x[0]),int(y[0]))

        rect = np.array( [[[text\_start[0]-10,text\_start[1]],[text\_start[0]+50,text\_start[1]],[text\_start[0]+50,text\_start[1]-30],[text\_start[0]-10,text\_start[1]-30]]], dtype=np.int32 )

        cv.fillPoly(image, rect, (255, 255, 255), cv.LINE\_4)

        cv.putText(image, str(angle), text\_start, cv.FONT\_HERSHEY\_COMPLEX\_SMALL, 1, (0,0,255))

Central line and value of its angle are put to output image using OpenCV (figure 7).

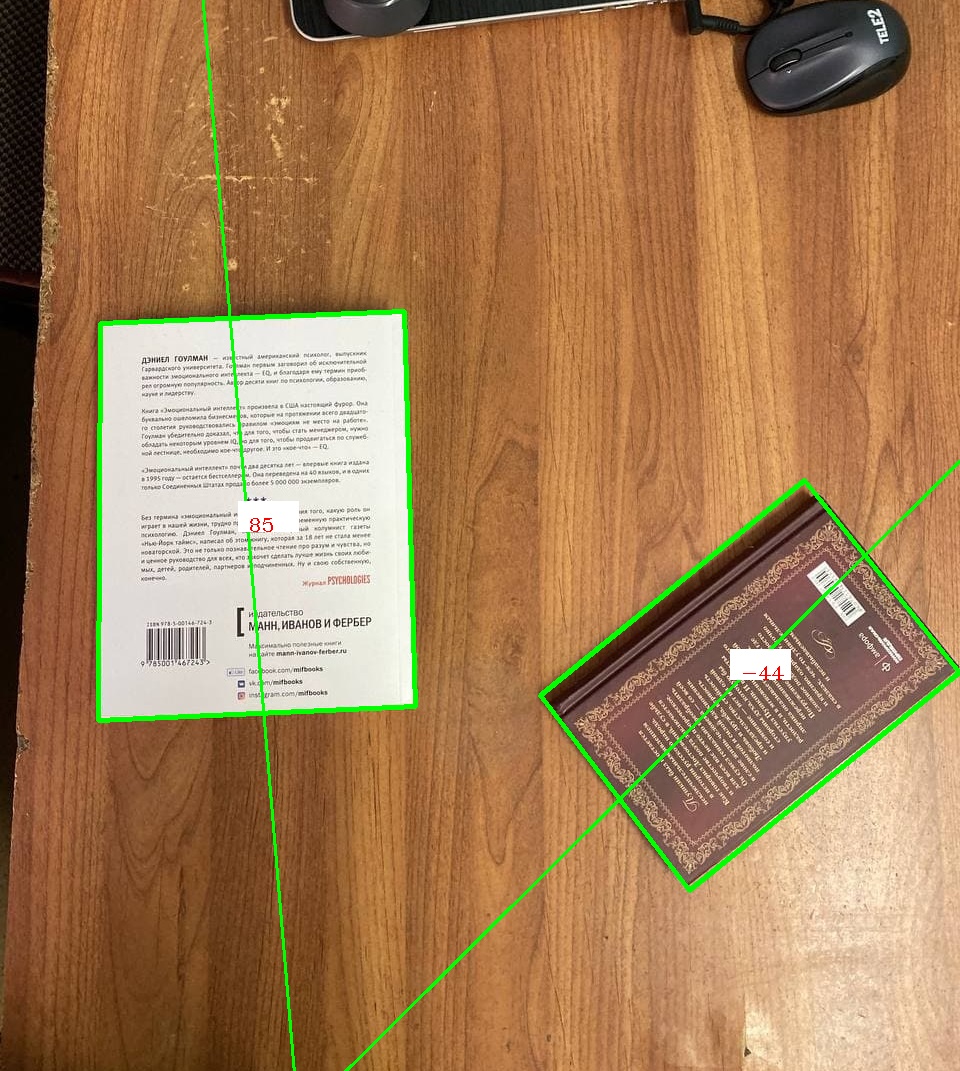


Figure 7. Output image

Video stream from web-camera is handled with OpenCV’s VideoCapture function:

cap = cv.VideoCapture(0)

while True:

    ret, image = cap.read()

Quality of detection is worse for dynamic video-stream ­– redundant objects are being recognized and indicated (figure 8).

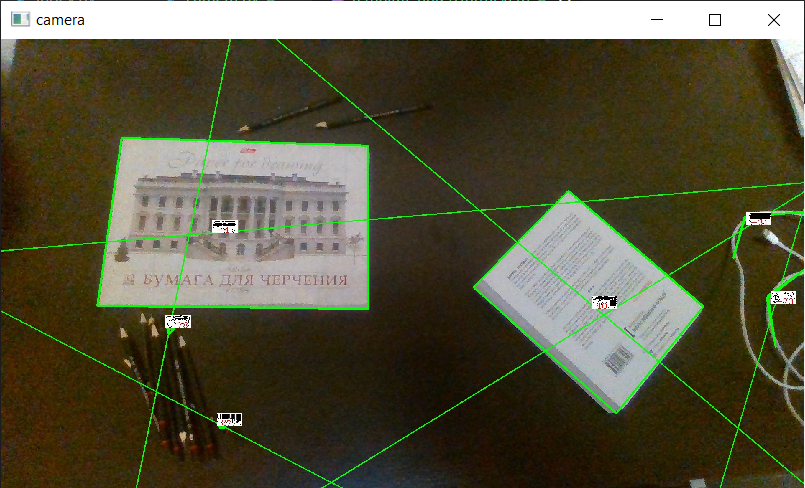


Figure 8. Web-camera output image processed

### Task 2

Dynamically detect the appearance of some object based on image of its outline. Find location of a detail using its flat vector image.

To solve the problem the following approach is proposed:

1. Recognize contours on input image using basic library operators
2. Convert vector image of detail to a bitmap
3. Detect contours with the strongest matching with the detail’s bitmap
4. Create graphical output indicating found matchings

**Assessment sheet**

Defect detection with computer vision

Consultation project

01.02.2021 – 06.06.2021

|  |  |  |  |
| --- | --- | --- | --- |
| **Project supervisor**  Full name, position | |  | |
| **Student[[1]](#footnote-1)**: | | **Victor E. Kozlovskiy** | |
| Master’s Programme | | **Big Data Systems** | |
| Group № | | **MBD202** | |
| Components of the final grade[[2]](#footnote-2) | | Grade on a 10-point scale | | Notes (if necessary) | |
| **О пр** – Grade for the project result (product) | |  | |  | |
| **О сп** Grade for the methods and technologies used | |  | |  | |
| **О р** Grade for the implementation of the project’s work | |  | |  | |
| **О к** Grade for the developed competencies | |  | |  | |
| **О гр** Grade for the student's individual contribution to group work | |  | |  | |
| **О ком** Grade for team work | |  | |  | |
| **О з** Grade for the presentation, project defense | |  | |  | |
| **О вз** Grade given by other project participants (peers evaluation) | |  | |  | |
| **О с** Student self-assessment | |  | |  | |
| Grade calculation (formula with weight coefficient) | |  | | | |
| **Final grade for the project** | |  | | Project supervisor’s signature | |
| **Number of credits** | | 3 | |

Date\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. For group projects, an assessment sheet is filled out for each group member [↑](#footnote-ref-1)
2. Only necessary components are used, if some component is not used in the assessment, then a dash is put in the corresponding line [↑](#footnote-ref-2)