PA1. Shape detection report - tanukiShapes

010-5136-6003, tlsgusghq@kaist.ac.kr, 신현호(Hyeonho Shin), 20203344, Electrical Engineering

# Abstract

My answer to PA1, tanukiShapes is detecting shape with features such as area, perimeter, and the number of vertices. It does not need more library except for numpy, opencv. Also extremely fast because it exploits on Rule-based method only.

One more thing. For convenience in scoring, I added TA code which can operate my classifier off-the-shelf. There is no trick. It modified some arguments because my classifier does not need some arguments. Mine is not learning-based.

# Tested Environment

* Python 3.7.9
* OpenCV 3.4.2
* Numpy 1.19.2

# How to use

Same as main.py you gave us.

## When checking accuracy with training images (you gave)

1. Pose the main.py with ‘./shapes’ folder which has images.
2. In terminal, type ‘python main.py’

## When checking accuracy with test images (you only have)

1. Pose the main.py with ‘./../ForTA’ folder which has images.
2. Un-comment the lines under the line, “ Pre-made TA code which I made for convenience.”
3. In terminal, type ‘python main.py’

# Design

## Summary

My shape classifier can be decomposed into two steps. In the first step, it extracts features, such as area, periment, number of vertices. In Classify step, using them, it classifies shapes based on pre-determined rule. Let me explain details in next sections.

## Preprocess

In this step, features are extracted from images. The extracted features are area, periment of shape in image and the number of vertices.

The number of vertices is calculated using approximation. When an image comes, it finds contours, approximates to a polygon. And return the number of the polygon’s vertices.

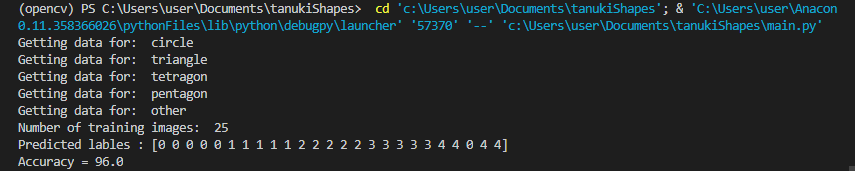
When finding area and periment(=de\_stretch()), we sum image and canny-edged image. Also, to make them scale- and roation-invariant, it extracts ROI only. In other words, it finds and extracts the rectangle with miminum area by minAreaRect(). And then for removal of stretching effect, it resizes the weight-height ratio to 1:1. For removal of resizing noises, make it down-scaled and Noise-filtered(=bilateralFilter()) in prior.

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |

Figure Original, Extracted ROI, Resized & filtered, Cannyed

## Classify

# Result in train images



# Conclusion

# References

1. 옥수별, Oct-14-2015, “[11편] 이미지 변환 – 리사이징, 이동, 회전, 원근효과”, <https://m.blog.naver.com/samsjang/220504966397>
2. OpenCV document, “Contour Features”, https://docs.opencv.org/3.4/dd/d49/tutorial\_py\_contour\_features.html
3. Sergio Canu, Sep-25-2018, “Simple shape detection – Opencv with Python 3”, https://pysource.com/2018/09/25/simple-shape-detection-opencv-with-python-3/
4. Jrosebr1, “imutils”, <https://github.com/jrosebr1/imutils/blob/master/imutils/convenience.py>