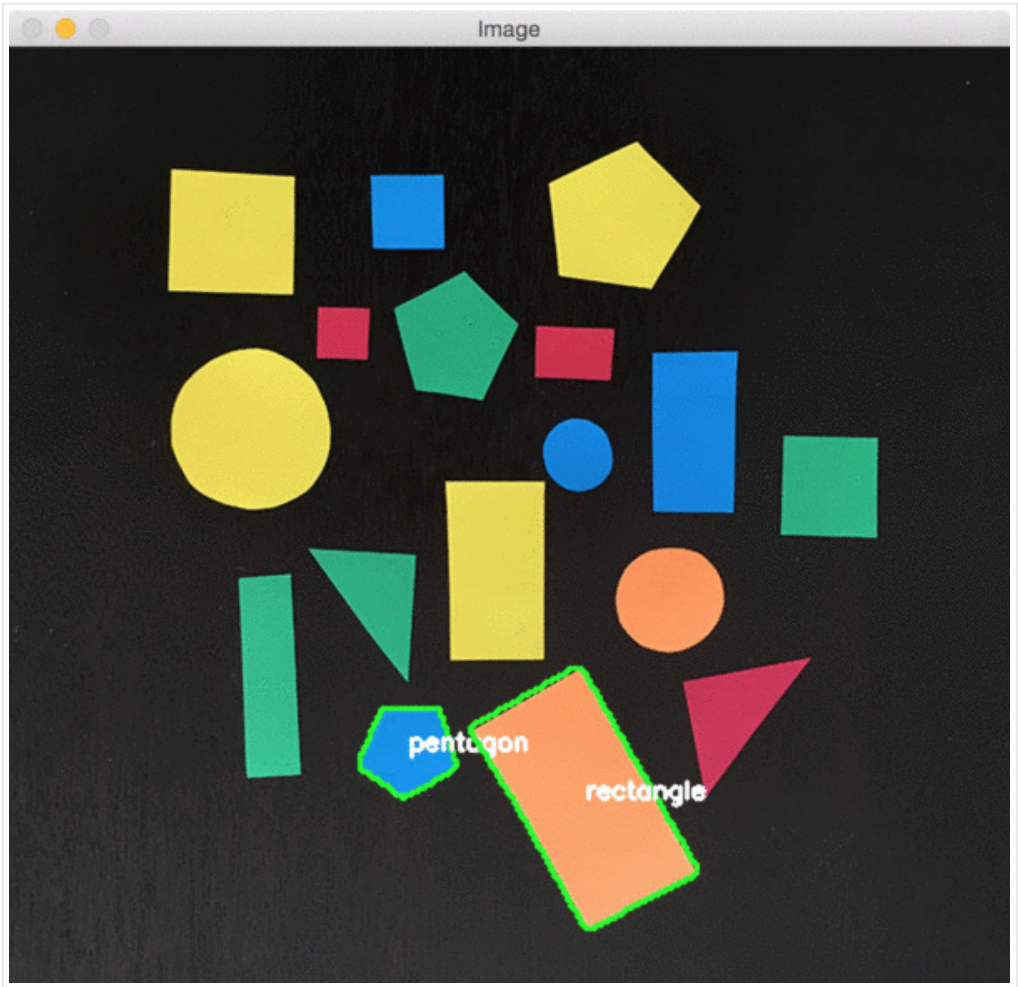
pyimagesearch
be awesome at building image search engines

OpenCV shape detection

by **Adrian Rosebrock** on February 8, 2016 in **Image Processing**, **OpenCV 3**, **Tutorials**

53



This tutorial is the second post in our three part series on *shape detection and analysis*.

Last week we learned how to compute the center of a contour using OpenCV.

Today, we are going to leverage contour properties to actually *label* and *identify* shapes in an image, just like in the figure at the top of this post.

Looking for the source code to this
Jump right to the downloads secti

Free 21-day crash course on computer vision & image search engines

OpenCV shape detection

Before we get started with this tutorial, let's quickly review our project structure:

```
OpenCV shape detection                                    Shell
1 |--- pyimagesearch
2 |    |--- __init__.py
3 |    |--- shapedetector.py
4 |--- detect_shapes.py
5 |--- shapes_and_colors.png
```

As you can see, we have defined a `pyimagesearch` module. Inside this module we have `shapedetector.py` which will store our implementa

Finally, we have the `detect_shapes.py` driver script

shapes, and then perform shape detection and ident

Before we get started, make sure you have the imutil

OpenCV convenience functions that we'll be using la

```
OpenCV shape detection                                    l
1 $ pip install imutils
```

Defining our shape detector

The first step in building our shape detector is to write

logic.

Let's go ahead and define our `ShapeDetector` . Ope

following code:

```
shapedetector.py                                          Python
 1 # import the necessary packages
 2 import cv2
 3
 4 class ShapeDetector:
 5     def __init__(self):
 6         pass
 7
 8     def detect(self, c):
 9         # initialize the shape name and approximate the contour
10         shape = "unidentified"
11         peri = cv2.arcLength(c, True)
12         approx = cv2.approxPolyDP(c, 0.04 * peri, True)
```

Line 4 starts the definition of our `ShapeDetector` class. We'll skip the `__init__` constructor here since nothing needs to be initialized.

We then have our `detect` method on **Line 8** which requires only a single argument, `c` , the contour (i.e., outline) of the shape we are trying to identify.

In order to perform shape detection, we'll be using *contour approximation*.

As the name suggests, contour approximation is an a

with a reduced set of points — thus the term *approxi*

This algorithm is commonly known as the Ramer-Douglas-Peucker algorithm, or simply the split-and-merge algorithm.

Contour approximation is predicated on the assumption that a curve can be approximated by a *series of short line segments*. This leads to a resulting approximated curve that consists of a subset of points that were defined by the original cruve.

Contour approximation is actually already implemented in OpenCV via the `cv2.approxPolyDP` method.

In order to perform contour approximation, we first compute the perimeter of the contour (**Line 11**), followed by constructing the actual contour approximation (**Line 12**).

Common values for the second parameter to `cv2.ap` e original contour perimeter.

Note: *Interested in a more in-depth look at contour a PyImageSearch Gurus course where I discuss comp as contours and connected-component analysis in deta*

Given our approximated contour, we can move on to

OpenCV shape detection

```
14          # if the shape is a triangle, it
15          if len(approx) == 3:
16              shape = "triangle"
17
18          # if the shape has 4 vertices, it
19          # a rectangle
20          elif len(approx) == 4:
21              # compute the bounding box of
22              # bounding box to compute the
23              (x, y, w, h) = cv2.boundingRe
24              ar = w / float(h)
25
26              # a square will have an aspect ratio that is approximately
27              # equal to one, otherwise, the shape is a rectangle
28              shape = "square" if ar >= 0.95 and ar <= 1.05 else "rectangle"
29
30          # if the shape is a pentagon, it will have 5 vertices
31          elif len(approx) == 5:
32              shape = "pentagon"
33
34          # otherwise, we assume the shape is a circle
35          else:
36              shape = "circle"
37
38          # return the name of the shape
39          return shape
```

It's important to understand that a contour consists of a *list of* ***vertices***. We can check the number of entries in this list to determine the shape of an object.

For example, if the approximated contour has *three vertices*, then it must be a triangle (**Lines 15 and 16**).

If a contour has *four vertices*, then it must be either a *square* or a *rectangle* (**Line 20**). To determine which, we compute the aspect ratio of the shape, which is s by the height (**Lines 23 and 24**). If the aspect ratio is ~ have approximately equal length). Otherwise, the sha

If a contour has *five vertices*, we can label it as a *pentagon* (**Line 31 and 32**).

Otherwise, by process of elimination (in context of this example, of course), we can make the assumption that the shape we are examining is a *circle* (**Lines 35 and 36**).

Finally, we return the identified shape to the calling method.

Shape detection with OpenCV

Now that our `ShapeDetector` class has been defined, let's create the `detect_shapes.py` driver script:

OpenCV shape detection — Python

```
# import the necessary packages
from pyimagesearch.shapedetector import S
import argparse
import imutils
import cv2

# construct the argument parse and parse
ap = argparse.ArgumentParser()
ap.add_argument("-i", "--image", required
    help="path to the input image")
args = vars(ap.parse_args())
```

We start off on **Lines 2-5** by importing our required p
implementation of the `ShapeDetector` class from th

Lines 8-11 handle parsing our command line argum
which is the path to where the image we want to pro

Next up, let's pre-process our image:

OpenCV shape detection

```
# load the image and resize it to a small
# the shapes can be approximated better
image = cv2.imread(args["image"])
resized = imutils.resize(image, width=300)
ratio = image.shape[0] / float(resized.shape[0])

# convert the resized image to grayscale, blur it slightly,
# and threshold it
gray = cv2.cvtColor(resized, cv2.COLOR_BGR2GRAY)
blurred = cv2.GaussianBlur(gray, (5, 5), 0)
thresh = cv2.threshold(blurred, 60, 255, cv2.THRESH_BINARY)[1]

# find contours in the thresholded image and initialize the
# shape detector
cnts = cv2.findContours(thresh.copy(), cv2.RETR_EXTERNAL,
    cv2.CHAIN_APPROX_SIMPLE)
cnts = cnts[0] if imutils.is_cv2() else cnts[1]
sd = ShapeDetector()
```

First, we load our image from disk on **Line 15** and resize it on **Line 16**. We then keep track of the `ratio` of the old height to the new resized height on **Line 17** — we'll find out exactly why we do this later in the tutorial.

From there, **Lines 21-23** handle converting the resized image to grayscale, smoothing it to reduce high frequency noise, and finally thresholding it to reveal th

Free 21-day crash course on computer vision & image search engines

After thresholding, our image should look like this:

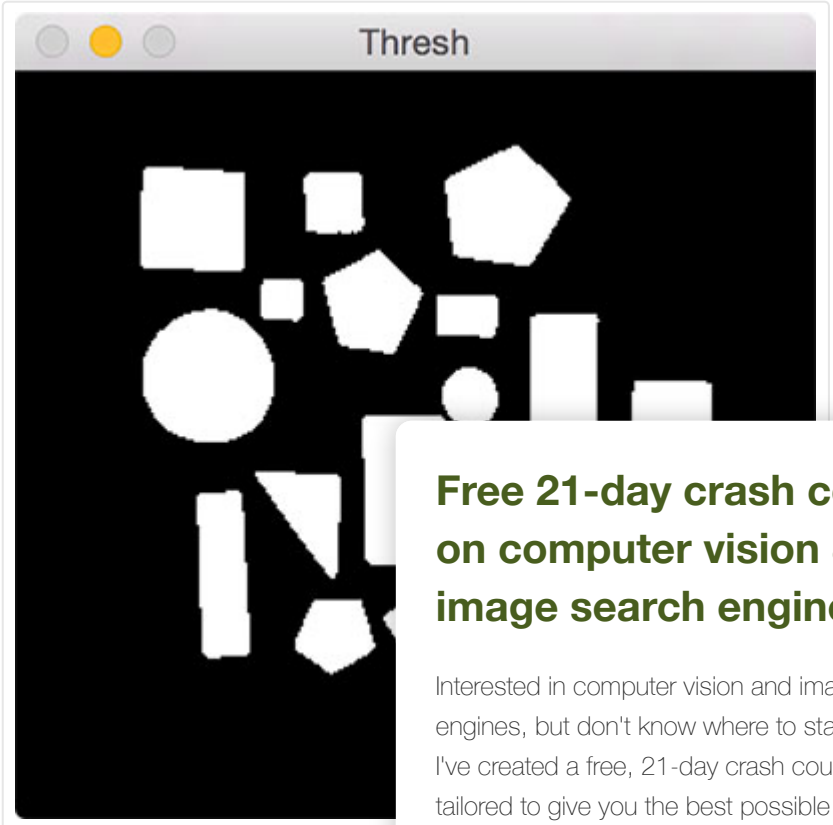


Figure 1: Thresholding re

Notice how our image has been *binarized* — the sha
background.

Lastly, we find contours in our binary image, handle g
`cv2.findContours` based on our OpenCV version,
30).

The last step is to identify each of the contours:

OpenCV shape detection — Python

```python
# loop over the contours
for c in cnts:
    # compute the center of the contour, then detect the name of the
    # shape using only the contour
    M = cv2.moments(c)
    cX = int((M["m10"] / M["m00"]) * ratio)
    cY = int((M["m01"] / M["m00"]) * ratio)
    shape = sd.detect(c)

    # multiply the contour (x, y)-coordinates by the resize ratio,
    # then draw the contours and the name of the shape on the image
    c = c.astype("float")
    c *= ratio
    c = c.astype("int")
    cv2.drawContours(image, [c], -1, (0, 255, 0), 2)
    cv2.putText(image, shape, (cX, cY), cv2.FONT_HERSHEY_SIMPLEX,
        0.5, (255, 255, 255), 2)

    # show the output image
    cv2.imshow("Image", image)
    cv2.waitKey(0)
```

On **Line 33** we start looping over each of the individual contours. For each of them, we compute the center of the contour, followed by performing shape detection and labeling.

Since we are processing the contours extracted from the *resized image* (rather than the original image), we need to multiply the contours and center *(x, y)*-coordinates by our resize `ratio` (**Lines 43-45**). This will give us the correct *(x, y)*-coordinates for both the contours and centroid of the original image.

Lastly, we draw the contours and the labeled shape on our image (**Lines 44-48**), followed by displaying our results (**Lines 51 and 52**).

To see our shape detector in action, just execute the following command:

```
OpenCV shape detection
1 $ python detect_shapes.py --image shapes_a
```

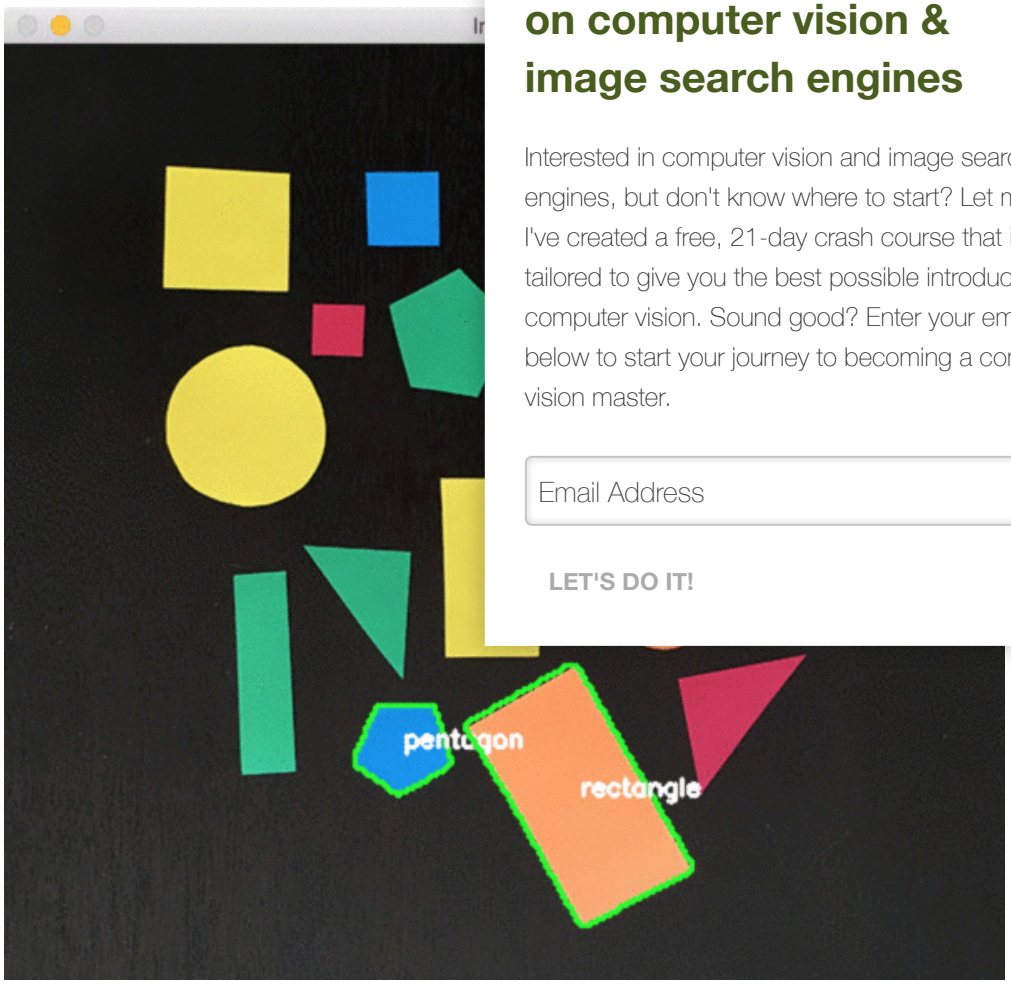



Figure 2: Performing shape detection with OpenCV.

As you can see from the animation above, our script loops over each of the shapes individually, performs shape detection on each one, and then draws the name of the shape on the object.

Summary

In today's post blog, we learned how to perform shape detection with OpenCV and Python.

To accomplish this, we leveraged *contour approxima* a curve to a more simple *approximated* version.

Then, based on this contour approximation, we examined the number of vertices each shape has. Given the vertex count, we were able to accurately label each of the shapes.

This lesson is part of a three part series on shape detection and analysis. Last week we covered how to compute the center of a contour. Today we covered shape detection with OpenCV. And next week we'll discuss how to ***label the actual color of a shape*** using color channel statistics.

Be sure to enter your email address in the form below to be notified when the next post goes live — ***you won't want to miss it!***

Downloads:

If you would like to download the
email address in the form below
you a **FREE 11-page Resourc**
Engines, including **exclusive te**
If so, enter your email address a

Email address:

Your email address

DOWNLOAD THE CODE!

Resource Guide (it's totally free).

Enter your email address below to get my **free 11-page Image Search Engine Resource Guide PDF**. Uncover **exclusive techniques** that I don't publish on this blog and start building image search engines of your own!

Your email address

DOWNLOAD THE GUIDE!

approximate contours, **contour properties**, **contours**, **find contours**, **shapes**

‹ OpenCV center of contour

Determining object color with OpenCV ›

121 Responses to *OpenCV shape de*

leena February 9, 2016 at 5:59 am #

REPLY

Why it is scanning and labeling from bottom to top?

How to to scan and label top to bottom?

Adrian Rosebrock February 9, 2016 at 3:54 pm #

REPLY

That is how the `cv2.findContours` method is implemented. If you would like to sort contours, see this post.

leena February 17, 2016 at 4:57 am #

Thanks Adrian. It worked and I am

Please help me in identifying lines connected
connected with line/arrow

with regards.

leena February 9, 2016 at 6:11 am #

I have done the same with

shape factor= area / (peri * peri)

if shapefactor >= 0.06 and shapefactor = 0.0484 and
shapefactor = 0.95 and ar <= 1.05 else "rectangle"

Adrian Rosebrock February 9, 2016 at 3:55 pm #

REPLY

Is there a particular reason you are taking the ratio of the area to the perimeter squared? It seems to make the rule more complicated.

leena February 10, 2016 at 11:11 pm #

REPLY

Actually I do not know the reason, just it got solved my problem, so I took it. You or somebody can help me understanding this and the better solution .
Thanks

bitflip June 21, 2016 at 11:30 am #

Free 21-day crash course on computer vision & image search engines

Given you have following triangle:

```
1 +--------+
2   \      /
3     \  /
4       +
```

The bounding rect of it would have aspect-ratio about 1:1.
So, better take the area() of the contour and compare it to width*height of the bounding rect.
when the error is too hight –> rectangle.

Vincent February 18, 2016 at 3:41 pm #

Hi Adrian,

First and foremost, thank you for this excellent tutorial

I have used the logic here to detect red triangles in a
class to identify only triangles. I am able to successfu
http://imgur.com/6Z9CnBA

I've noticed that sometimes a very messy contour will
http://imgur.com/4a06psM

What would be a good way to tweak this?
http://pastie.org/10727912
http://pastie.org/10727915

Adrian Rosebrock February 18, 2016 a

Keep in mind that the code is only as good as the images that you put into it. The code detailed in this post assumes simple shapes that can be recognized utilizing contour properties. For more advanced shapes, or shapes that have substantial variances in how they appear (such as noisy contours), you might need to train your own custom object detector.

Anyway, the reason sometimes even messy contours get classified differently is due to the contour approximation. Play around with the percentage used in `cv2.approxPolyDP` and you'll be able to see the differences.

Peng March 4, 2016 at 7:49 pm #

REPLY

Hi Adrian,

Thank you for making this. A little feedback on the image file.

I notice that if using a .jpg file as the source, the moment(cnt) will not get a correct value.

It report an error :

cntX = int(M[“m10”] / M[“m00”])
ZeroDivisionError: float division by zero

Any ideas on this?

Thanks

Adrian Rosebrock March 6, 2016 at 9:20 am #

REPLY

Version version of OpenCV and Python are you using?

In either case, you can resolve the issue by doing:

```
1 if M["m00"] > 0:
2     # ... continue to process the co
```

This `if` statement will take care of the divide by z

Alternatively, you can add a tiny value to the mom

```
cX = (M["c10"] / (M["m00"] + 1e-7)
```


saurabh November 18, 2017 at 4:27 ar

can you please elaborate this .. i di

==========
cX = int((M[“m10”] / M[“m00”] + 1e-7) * ratio)
ZeroDivisionError: float division by zero
==========
showing this error

Adrian Rosebrock November 18, 2017 at 8:07 am #

REPLY

My original comment is missing a parenthesis:

```
M["c10"] / (M["m00"] + 1e-7)
```

Notice how the addition is done before the divide.

Euan March 10, 2016 at 10:00 pm #

REPLY

Hi Adrian,

Firstly thanks for a great tutorial and site. I’m a mechanical engineer and noob to openCV, python and linux and have managed to get openCV 3.0 and python2.7

https://www.pyimagesearch.com/2015/06/22/install

Free 21-day crash course on computer vision & image search engines

In order to get this code running on my setup, I needed to modify “float” on line 17 to “int” as it was causing cast problems on line 43 “c *=ratio”. I believe this is probably due to an update of how python works from how it worked when you wrote this tutorial. Is this the case?

Adrian Rosebrock March 13, 2016 at 10:29 am #

REPLY

Interesting, Brandon mentioned this issue in a comment above. Which version of OpenCV and NumPy are you using?

brandon March 11, 2016 at 4:20 pm #

Adrian, great stuff. I’ve learned a lot from you working through this post for now, and I’m getting the

```
1   File "detect_shapes.py", line 46, in
2     c *= ratio
3 TypeError: Cannot cast ufunc multiply ou
```

Adrian Rosebrock March 13, 2016 at 1

I personally haven’t seen this error mess NumPy versions you are using?

Brandon March 14, 2016 at 12:28 pm #

REPLY

numpy is 1.10.4, and it happens with both OpenCV 2.4.12 and 3.1.0 (in a virtualenv, thanks to another of your tutorials) under Python 2.7 on OS X 10.11.3

The workaround was simply to adjust data types pre and post multiplication:

c=c.astype(np.float_)
c *= ratio
c=c.astype(np.int32)

It works now.

Adrian Rosebrock March 14, 2016 at 3:15 pm #

REPLY

Thanks for the tip Brandon! I’ll be sure to dive into this more. I’m using NumPy 1.9.3, so perhaps this is an issue with NumPy 1.10.X.

Ahmed Abdeldaim March 24, 2016 at 10:30 am #

REPLY

Great work Mr. Adrian
but is there a way to make the selection more softer, for example by reduce point size ??
or this is the best result??

Adrian Rosebrock March 24, 2016 at 5:10 pm #

REPLY

Absolutely, you just need to apply *contour approximation* first. I detail contour approximation in a few blog posts, but I would start with this one.

Ahmed Abdeldaim March 26, 201

Thanks for your help.

darshan March 26, 2016 at 2:09 am #

how to install imutils module
I used pip install imutils I'm getting error

Adrian Rosebrock March 27, 2016 at 9

Please see the "OpenCV shape detection" section of this blog post. You just need to use `pip`

```
$ pip install imutils
```

firoz khan April 23, 2016 at 10:42 am #

REPLY

hi adrian it is only detcting one pentagon and nothing else

Adrian Rosebrock April 25, 2016 at 2:09 pm #

REPLY

Make sure you click on the window and press any key on your keyboard — this will advance the script. Right now a keypress is required after each detection.

Diego Fernando Barrios April 29, 2016 at

Good afthernoon!

Thanks very much for this tutotrial, you´re doing a great and util work.

Friend, I have a problem with contour detection, when I change the image, the project don't work (I'm using a black background, I take the image from USB camera). I not have problem with the image path.

The python scripts should recognize (9 "nine" rectangles") but just one is recognized

Sorry for the writing, my english is not so good.

I would like that you can help me. I'm working in my work grade.

Thanks very much!

Adrian Rosebrock April 30, 2016 at 3:5

Depending on your image, this could b
sure that after thresholding your 9 rectangles hav
contour approximation and see how many points
tweak the `cv2.approxPolyDP` parameters.

itai May 8, 2016 at 4:24 am #

Hey Adrian,
I was wondering why did you use the Ramer-Douglas
of using the convex hull ?

Thanks

Adrian Rosebrock May 8, 2016 at 8:12 am #

REPLY

The contour approximation algorithm and Convex Hull algorithm are used for two separate purposes. As the name implies, contour approximation is used to reduce the number of points along a contour by "simplifying" the contour based on a percentage of the perimeter. Your resulting contour approximation is this a simplification of the shape by utilizing points that are *already* part of the shape.

The convex hull on the other hand is the smallest convex set that contains all points along the contour — it is, by definition, not a simplification of the contour shape and the resulting convex hull actually contains points that are not part of the original shape.

In this case, I used contour approximation because I wanted to reduce the number of (x, y)-coordinates that comprise the contour, while ensuring that all points in the *resulting approximation* were also part of the *original shape*.

Armin June 30, 2016 at 2:56 pm #

REPLY

Hello Adrian

thanks for tutorial

I want to show detected shapes in seperate windows(each shape on each window), what should I do? also tried cropping them (using ROI) but I didn't able to work it out.

tnx

Adrian Rosebrock July 1, 2016 at 2:59 pm #

REPLY

Hey Armin — you're on the right track. You should be applying array slicing to extract the ROI, then using the `cv2.imshow` function on each ROI. An example of ROI slicing can be found in this blog post as well as Practical Python and OpenCV.

Alex Hopper July 24, 2016 at 5:04 pm #

Hello,
I'm new openCV-Python user.. I have a question abou
I have a database containing pre-processed images
these images. Is there a best way to start?

https://www.pyimagesearch.com/2016/02/08/openc

Can I use it to start?

Thanks.

Adrian Rosebrock July 27, 2016 at 2:3

What type of images are you working with? You mentioned they were pre-processed by the Kinect. Are they depth images? RGB images?

Neal July 27, 2016 at 8:58 am #

REPLY

hi

I'm looking for advice in shape detection. I want to use a camera to detect different kinds of shapes on a microcontroller. What would be the best method to approach this?

Your help will be much appreciated.
Thanks

Adrian Rosebrock July 27, 2016 at 1:55 pm #

REPLY

Hey Neal — to start, you need to segm
you have any example images that you're workin

Neal August 2, 2016 at 8:23 am #

REPLY

well i'm going to be using different shape cut outs of wooden blocks as my different objects.

Adrian Rosebrock August 2, 2016 at 2:57 pm #

REPLY

I would use a similar approach as detailed in this blog post. Cut out your wooden blocks and place the camera such that i
edge detection to find the blocks. And fr
technique to label the shapes.

Anupam September 4, 2016 at 3:40 am #

Can you please help me out with detecting

Adrian Rosebrock September 5, 2016

For overlapping shapes, I would sugges

Leena October 6, 2016 at 8:25 am #

How can we use the shapedetector to classify polygon as rectangle/diamond(decision box)/ parallelogram….
please help

Poehe October 31, 2016 at 11:09 am #

REPLY

Hi Adrian, thank you so much for the tutorial, it's a great starting point for me to dive into OpenCV.

I noticed when processing images using your code that in pictures with a white background the engine also shows the contours/edges of the whole input picture as being a shape, while in images with a black background (as in your example) the engine ignores the outside contours of the whole input pic and only shows the contours of the objects within the input pic itself (which is the way it should work, I suppose).

Could you think of a solution that makes the engine not classify the picture edges as contours?

Your help will be much appreciated!

Cheers

Adrian Rosebrock November 1, 2016 at 8:59 am #

REPLY

So if I understand your question correctly, you are using a background that is lighter than the shapes themselves? And after thresholding your shapes appear as "black" on a "white" background? Am I understanding that correctly? If so, simply invert the threshold step to make the shapes "white" on a "black" background.

Megha Maheshwari November 21, 2016 at 12:42 am #

REPLY

Hi Adrian

How can we differentiate between rectangle and trape
corners and hence is either a square, rectangle or tra
check the width and height, but how do i differentiate

Adrian Rosebrock November 21, 2016

There are many ways to do this. I would
bounding box area). A perfect rectangle will have
much less than 1.0. You can also compute the a
would have near perfect 90 degree angles. Eithe

Mohamad November 23, 2016 at 8:12 am #

Hi Great Man. Mr. Adrian
I guess your offer in this tutorial use the webcam for raspberry pi. its Ok? other question is that for detect other shapes such as (H) similar or (L) similar or … How doing it? Is this method work with edge or point detection?

Adrian Rosebrock November 23, 2016 at 8:30 am #

REPLY

You can certainly use a webcam or Raspberry Pi camera module to perform shape detection. You would just need to read the frame from the camera and process it. I provide tutorials on how to access webcams here.

As for detecting an "H" or "L" you can do that using contour properties (extent, solidity, etc.), template matching, or image descriptors such as Histogram of Oriented Gradients. I would suggest taking a look at Practical Python and OpenCV along with the PyImageSearch Gurus for more advanced demonstrates of recognizing objects in images.

Luís Serrador November 25, 2016 at 5:48 am

Hi Adrian,

I tried two of your tutorials (this one and 'OpenCV center of contour') and when I execute the command to run the .py file my result is not the same as you show at the end of the tutorials. My final image only recognize the first shape/center, and doesn't recognize more shapes. Is there anything that could be wrong?

Adrian Rosebrock November 28, 2016 at 10:41 am #

REPLY

Hi Luís — what versions of OpenCV and Python are you using?

Luís Serrador November 28, 2016 a

Hi Adrian! I'm using OpenCV 3.0 ar

Adrian Rosebrock Novembe

Nope, the Python and OpenCV
the setup to confirm it wasn't an outlier s
executing the code. I assume you down
post rather than copying and pasting the
code as well.

Thomas November 30, 2017 at 7:46 am #

Hi Luis,

you should make an empty file of __init__.py , to solve you issue for not detect all the shape from the *.png file.

Jan December 4, 2016 at 1:08 am #

REPLY

Hi Adrian,
Thanks for the tutorial,
I did the same, but for certain cicrles the vertices were shown as 4 and hence were displayed as squares , can you suggest a way to increase the number of detected vertices in the picture.
Thanks

Adrian Rosebrock December 5, 2016 a

You'll want to play with the following line

```
approx = cv2.approxPolyDP(c, 0.04 * peri, True)
```

The smaller the value passed in for `peri`, the more vertices you'll obtain.

Preethi December 23, 2016 at 6:16 am #

REPLY

Hi Adrian,
From Shape detection i should detect circle alone. even though rectangle,square and etc present. In your Same example i need this modification

Adrian Rosebrock Decembe

For circle detection, take a look
determine a circle as well. A circle will ha
other shapes. You basically need to crea

Niel January 5, 2017 at 12:52 am #

Hi, Adrian i'm mechatronic students and no
can i create a detection square box in OpenCV for de
or to stop

Adrian Rosebrock January 7, 2017 at 9

Hey Niel — can you elaborate on what you mean by a "detection square box"? I'm not sure what you mean.

Chandu January 18, 2017 at 1:23 am #

REPLY

Hi Adrian

i'm getting an error

Traceback (most recent call last):
File "", line 2, in
from pyimagesearch.shapedetector import ShapeDetector
ImportError: No module named pyimagesearch.shapedetector

Adrian Rosebrock January 18, 2017 at

Hey Chandu — make sure you download the source code to this blog post using the "Downloads" section. It's likely that your project directory structure does not match mine (perhaps missing a `__init__.py` file. Please download my code and compare it to yours.

Arturo January 19, 2017 at 9:48 pm #

REPLY

Hi, i´m having a problem with the code, Sorry for my ignorance but where do I have to put the path of the image?
I can´t understand the lines 9-12 where you say that we have to place the path, the (– image) part

Adrian Rosebrock January 20, 2017 at

Hey Arturo, I suggest you read up on c
need to modify the code at all.

tal January 25, 2017 at 12:14 pm #

Hi, i´m having a problem with the code, i hav
ap.add_argument("-i", "–image", required=True, help=
and i'm getting this error.

usage: detect_shapes.py [-h] -i IMAGE
detect_shapes.py: error: argument -i/–image is requir
thanks alot

Adrian Rosebrock January 26, 2017 at 8:21 am #

REPLY

You ***do not*** have to modify the code at all. You just need to supply the `--image` switch to the Python script via command line argument:

```
$ python detect_shapes.py --image shapes_and_colors.png
```

Please read up on command line arguments before continuing.

BKumar March 2, 2017 at 2:07 am #

REPLY

Hey Adrian,

I was wondering about how to find out the number of different shapes in such an example. Like if there are 4 squares, 2 rectangles etc., how can you label them in the image as Square #1, Square #2 or something. Or display the number of instances of each shape in the image.

Thanks in advance.

Adrian Rosebrock March 2, 2017 at 6:41 am #

REPLY

I would use a Python built-in dictionary type and simply count the number of shapes as you loop over them. Your pseudocode might look something like:

```python
shapes = {}
for shape in detectedShapes:
    shapes[shape] = shapes.get(shape, 0) + 1
```

This would give you a dictionary of shape counts.

Milán Vincze March 5, 2017 at 1:42 pm #

Hello!

A get an error when I run the program:
ImportError: No module named 'imutils'

I installed imutils before. Could you please help?

Adrian Rosebrock March 6, 2017 at 3:

The first part of this blog post discusses

```
$ pip install imutils
```


Milán Vincze March 6, 2017 at 4:42 pm #

REPLY

Yes I know I ran that before I start the program. Maybe the problam is that the raspberry want to run it in python3. How can I run in python2 if I installed python2 and python 3 too?
I thought that is the problem because the installation put the imutils in python2.7 library.

Milán Vincze March 6, 2017 at 4:46 pm #

REPLY

I also copied the imutils folder to python3.1 folder

Milán Vincze March 6, 2017 at 4:49 pm #

Sorry, I copied the imutils to the shape detector folder and it worked, sorry to bother you.

Adrian Rosebrock March 8, 2017 at 1:15 pm #

REPLY

For what it's worth, you could also just install imutils for your Python 3 as well:

```
$ pip3 install imutils
```

Milán Vincze March 12, 2017 at 6:46 am #

REPLY

Hello! Adrian

First thank you for the exellent tutorial! I am interesting in a real-time shape detector with the picamera. Maybe there is a tutorial for it that you made?

Adrian Rosebrock March 13, 2017 at 1

I would suggest using the `cv2.Video` this blog post. My book, Practical Python and Op streams and video files — this would help you pc image.

Tyler April 16, 2017 at 1:13 pm #

Adrian can u suggest what change time using laptop camera feed ,I would be ve

Adrian Rosebrock April 19, 2017 at 1:06 pm #

REPLY

The comment you replied to has a link to a blog post + book that I recommend that you read so you can access your laptop webcam. Take the time to study the basics of OpenCV first, then it will be easy to implement this method for real-time applications.

PANJI March 14, 2017 at 3:48 pm #

REPLY

Sir i have problem error

usage: detect_shapes.py [-h] -i IMAGE
detect_shapes.py: error: argument -i/–image is required

anyone help me?

Adrian Rosebrock March 15, 2017 at 8

Please read the comments before you submit your own. I answered your question in reply to "Arturo" above.

Open The CV March 19, 2017 at 11:38 pm #

REPLY

Hi, how can I find the angle between a triangle and the horizontal axe ?

Adrian Rosebrock March 21, 2017 at 7:24 am #

REPLY

I'm not sure what you mean. Do you ha

Open The CV March 21, 2017 at 7:3

Problem solved with some trigonon

sandra May 8, 2017 at 12:44 pm #

Hello, how can i detect certain rectangle wh

Adrian Rosebrock May 8, 2017 at 12:4

Hi Sandra — I'm not sure I understand your question. It would be helpful if you had an example image of what you're working with.

Abu May 11, 2017 at 6:51 pm #

REPLY

Hello Adrain,

Awesome tutorial, thank you.

I was wondering if there is a way to detect rectangles or squares to exact approximately. Basically, I have an image, and it has shapes and text in it. I only want to detect the shapes and ignore all the text in the image. This tutorial really helped me. But, I am still detecting squares and rectangles in the text of the image. is there a way, I can completely ignore that?.

Thanks.

Adrian Rosebrock May 15, 2017 at 9:0

Free 21-day crash course on computer vision & image search engines

I would compute the solidity of the shape which is the area of the contour divided by the convex hull area. Text will have a lower solidity than a rectangle which should be equal to one.

Laura May 30, 2017 at 2:46 pm #

REPLY

Hello Adrian,

Thanks a lot for the awesome tutorial.

I am using the shape detection to get the coordinates from where rectangular elements are located in an image that I am getting from my phone camera. For some specific layouts the code works perfectly, but when I start to place elements in the same row, the c
different order.

However, when I try the same layout from an image I
correctly. When I compared both shape detection pro
are sorted is also changing. I tried altering the lightnin
something that is altering the order of how the elemer
are organised.

Image from camera: http://imgur.com/a/xHaGn

Digital image: http://imgur.com/a/NNsv5

Do you have any idea of what could be the problem?

Adrian Rosebrock May 31, 2017 at 1:0

The `cv2.findContours` function wil
your contours if you expect them to be in a given

Martin June 14, 2017 at 10:21 am #

REPLY

Hi Laura,
I am working on the same topic –> finding squares on picture taken by my phone camera.
Could you share your project with us?
Thanks

jandi May 30, 2017 at 7:43 pm #

REPLY

Thanks for this tutorial …
I have a question. In my image, I have a square and lozenge shape and I want to distinguish between them, how can I do that?

Adrian Rosebrock May 31, 2017 at 1:06 pm #

REPLY

I would suggest using either:

1. Contour properties, such as extent, solidity, and aspect ratio.
2. Features, such Hu Moments or Zernike Moments.

Thailynn June 19, 2017 at 12:35 pm #

REPLY

Hi Adrian,
Is there a way to save the results of the image classifi
squares and saving a new image with just the square
imagery.

Thank you very much for any insight you can offer!

Adrian Rosebrock June 20, 2017 at 10

Hi Thailynn — if you are trying to extract
box and extract it using array slicing. You can the
function. I cover this in more detail inside Practica

Cuningan July 23, 2017 at 12:55 pm #

Hello Adrian, i ahve bought you curse and r
I am using RPI with with OpenCV3.2 compiled like you described on you post.

My problem is that also, python2 and 3 can not find imutils, i have tryed to install it with pip but failed with a large crash…

Any idea about??

Adrian Rosebrock July 24, 2017 at 3:33 pm #

REPLY

Which tutorial did you use to install OpenCV? Also, what is the error you are getting when trying to install imutils? Without knowing the error, I cannot provide any suggestions.

disheet August 8, 2017 at 3:46 am #

REPLY

Hello Sir,

I am getting below error.Please give me the solution

File "/usr/local/lib/python2.7/dist-packages/imutils/convenience.py", line 69, in resize
(h, w) = image.shape[:2]
AttributeError: 'NoneType' object has no attribute 'shape'

disheet August 8, 2017 at 3:48 am #

REPLY

And i am using logitech c170 webcamera.

Adrian Rosebrock August 10, 2017 at 8

REPLY

It sounds like OpenCV cannot access y
resolve them) in detail inside this post.

disheet August 11, 2017 at 1:57 am #

Hello Sir,

How can i do shape detection using webcam?

Adrian Rosebrock August 14, 2017 at 1

You apply the exact same algorithm det
can access the frames of a video using this post

morejump September 10, 2017 at 12:49 am #

REPLY

Hi Adrian,
the rectange is shape which has 4 vertices, and also has three of them is 90 degree. cause your algorithm dose not work every time

Doson September 26, 2017 at 4:29 am #

REPLY

I don't think it is strict to recognize the sharp just by the number of the 'approx'.For example, a rhomboid have for 'approx' but it is neither a square or rectangle.I suggest that more judgement such as checking the degree in each 'approx' .If more than three degree was close to 90， maybe 91 or 89， we can believe it is a square or rectangle.

David October 11, 2017 at 5:08 am #

Just wanted to say that these tutorials have been amazing! Thanks for publishing these

Adrian Rosebrock October 13, 2017 at 8:57 am #

REPLY

Thanks David, I really appreciate that 🙂

KansaiRobot October 23, 2017 at 5:08 am #

REPLY

Very nice Blog, I have just discovered it. I am going to be browsing through it tomorrow but let me ask you a question. In this post you covered shape d
each other- therefore you can find the true center.

How about detecting shapes when one shape is touc

Adrian Rosebrock October 23, 2017 at

Take a look at the Watershed algorithm.

Enko October 27, 2017 at 5:56 am #

Hi Adrian,
Thank's for your excellent tutorial, I want to detect circ
detection part:
else:
# shape = "circle"
img = np.zeros(image.shape, image.dtype)
cv2.drawContours(img, [c], -1, (255, 255, 255), -1)
img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
circles = cv2.HoughCircles(img, cv2.HOUGH_GRADIENT, 1.2, 100)
if circles is not None:
shape = "circle"
but the "circles" is always None. Can you help me?

Adrian Rosebrock October 27, 2017 at 11:17 am #

REPLY

Hi Enko — see this blog post: Detecting Circles in Images using OpenCV and Hough Circles.

KansaiRobot October 29, 2017 at 8:15 pm #

REPLY

Part 1) I would like to ask if watershed can be effectively used for adjacent objects that are not circles. In particular tablet-like. (I tried the method and it does not work)

Adrian Rosebrock October 31, 2017 at 7:55 am #

REPLY

In general, yes, the watershed algorithm can be used for these types of objects provided you can obtain a reasonable segmentation.

KansaiRobot October 31, 2017 at 8:

REPLY

Yes, segmentation is not a problem
segmentation. I can send you the original (no
black and white)

KansaiRobot October 29, 2017 at 8:16 pm #

An example of my problems can be found h
https://stackoverflow.com/questions/46107628/prob

I would like to hear your advice on how to tackle this p

Adrian Rosebrock October 31, 2017 at

Do you have the original example image

KansaiRobot October 31, 2017 at 8:28 pm #

REPLY

Thanks. I have some sample images obtained with backward ilumination. (Therefore basically dark shapes on white background). I can send some to you for the post you are considering.

Adrian Rosebrock November 2, 2017 at 2:36 pm #

REPLY

Sure, that would be cool to take a look at. Send me a note here or if you already know my email address you can send the images and reference this blog post.

KansaiRobot October 29, 2017 at 8:17 pm #

Free 21-day crash course on computer vision & image search engines

Part2) You seem to be using scikit watershed not opencv. Any reason for that?

Adrian Rosebrock October 31, 2017 at 7:55 am #

REPLY

At the time the watershed blog post was published the watershed + connected-component analysis functions in OpenCV were not as easy to use in OpenCV versus scikit-image. That has changed now and I would recommend using whichever one you are more comfortable with.

Kanwal November 21, 2017 at 9:14 am #

REPLY

Hello Sir,

Can you help how to solve the problem of occlusion.

Sameer November 26, 2017 at 5:14 am #

Hi Adrian,

When I run the program, only single object is being c

Adrian Rosebrock November 27, 2017

Make sure you click the active window
the detection.

Yadnyesh December 1, 2017 at 10:32 am #

REPLY

Hello Adrian,
I have slight problem
The code works fine with images which have a black background it detects all the shapes
the problem comes when an image has a white background.
It recognizes the frame of image as a rectangle and none of the shapes within it.
Can you help me with this?
Thank you in advance

Yadnyesh December 1, 2017 at 11:31 am #

REPLY

Sorry for this I read your reply for a comment
So all I have to do this
replace
thresh = cv2.threshold(blurred, 60, 255, cv2.THF

with
thresh = cv2.threshold(blurred, 255, 60, cv2.THRESH_BINARY)[1]

Leave a Reply

Name (required)

Email (will not be published) (r

Website

SUBMIT COMMENT

Free 21-day crash course on computer vision & image search engines

You're interested in deep learning and computer vision, *but y* **k will teach you all you need to know about deep learni**

CLICK HERE TO MASTER DEEP LEARNING

Free 21-day crash course on computer vision & image search engines

Interested in computer vision and image search engines, but don't know where to start? Let me help. I've created a free, 21-day crash course that is hand-tailored to give you the best possible introduction to computer vision. Sound good? Enter your email below to start your journey to becoming a computer vision master.

Email Address

LET'S DO IT!

You can detect faces in images & video.

Are you interested in **detecting faces in images & video?** But **tired of Googling for tutorials** that *never work?* Then let me help! I guarantee that my new book will turn you into a **face detection ninja** by the end of this weekend. Click here to give it a shot yourself.

CLICK HERE TO MASTER FACE DETECTION

PyImageSearch Gurus: NOW ENROLLING!

Free 21-day crash course on computer vision & image search engines

The PyImageSearch Gurus course is *now enrolling!* Inside the course you'll learn how to perform:

- Automatic License Plate Recognition (ANPR)
- Deep Learning
- Face Recognition
- *and much more!*

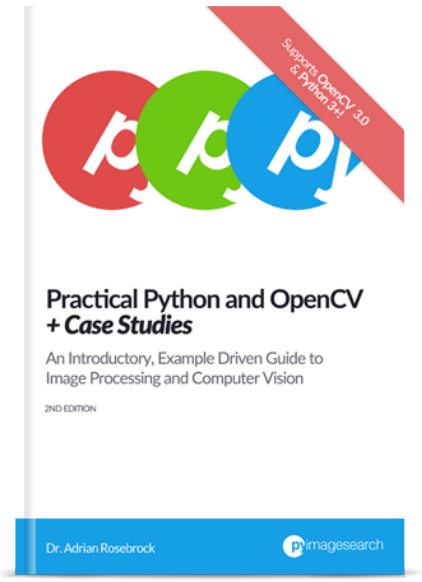
Click the button below to learn more about the cours

TAKE A TOUR & GET 10 (FREE) LESSONS

Hello! I'm Adrian Rosebrock.

I'm an entrepreneur and Ph.D who has l
Chic Engine. I'm here to share my tips, t

Learn computer vision in a single weekend.



Want to learn computer vision & OpenCV? I can teach you in a **single weekend**. I know. It sounds crazy, but it's no joke. My new book is your **guaranteed, quick-start guide** to becoming an OpenCV Ninja. So why not give it a try? Click here to become a computer vision ninja.

CLICK HERE TO BECOME AN OPENCV NINJA

Free 21-day crash course on computer vision & image search engines

Interested in computer vision and image search engines, but don't know where to start? Let me help. I've created a free, 21-day crash course that is hand-tailored to give you the best possible introduction to computer vision. Sound good? Enter your email below to start your journey to becoming a computer vision master.

Email Address

LET'S DO IT!

Free 21-day crash course on computer vision & image search engines

Subscribe via RSS

Never miss a post! Subscribe to the PyImageSearch RSS Feed and keep up to date with my image search engine tutorials, tips, and tricks

POPULAR

Install OpenCV and Python on your Raspberry Pi 2 and B+
FEBRUARY 23, 2015

Install guide: Raspberry Pi 3 + Raspbian Jessie + OpenCV 3
APRIL 18, 2016

Home surveillance and motion detection with the Raspb
JUNE 1, 2015

How to install OpenCV 3 on Raspbian Jessie
OCTOBER 26, 2015

Basic motion detection and tracking with Python and O
MAY 25, 2015

Ubuntu 16.04: How to install OpenCV
OCTOBER 24, 2016

Accessing the Raspberry Pi Camera with OpenCV and P
MARCH 30, 2015

Free 21-day crash course on computer vision & image search engines

Interested in computer vision and image search engines, but don't know where to start? Let me help. I've created a free, 21-day crash course that is hand-tailored to give you the best possible introduction to computer vision. Sound good? Enter your email below to start your journey to becoming a computer vision master.

Email Address

LET'S DO IT!

Search

Search...

Find me on **Twitter**, **Facebook**, **Google+**, and **LinkedIn**.

Free 21-day crash course on computer vision & image search engines