



[首页](#)
[最新文章](#)
[IT 职场](#)
[前端](#)
[后端](#)
[移动端](#)
[数据库](#)
[运维](#)
[其他技术](#)

- 导航条 - ▼

[伯乐在线](#) > [首页](#) > [所有文章](#) > [Python](#) > 用 Python 和 OpenCV 检测图片上的条形码

用 Python 和 OpenCV 检测图片上的条形码

2014/11/28 • [Python](#), [书籍与教程](#), [开发](#) • [2 评论](#) • [OpenCV](#), [Python](#)

分享到： 35 本文由 [伯乐在线](#) - [Halal](#) 翻译, [黄利民](#) 校稿。未经许可, 禁止转载!

英文出处: [Adrian Rosebrock](#)。欢迎加入[翻译组](#)。

更新: 这篇文章的介绍看起来有点“离题”, 某些方面是因为在写文章之前, 我刚看完《南方公园黑色星期五》, 所以我肯定在僵尸购物者、黑色星期五的混乱和《权利的游戏》中得到一些灵感。

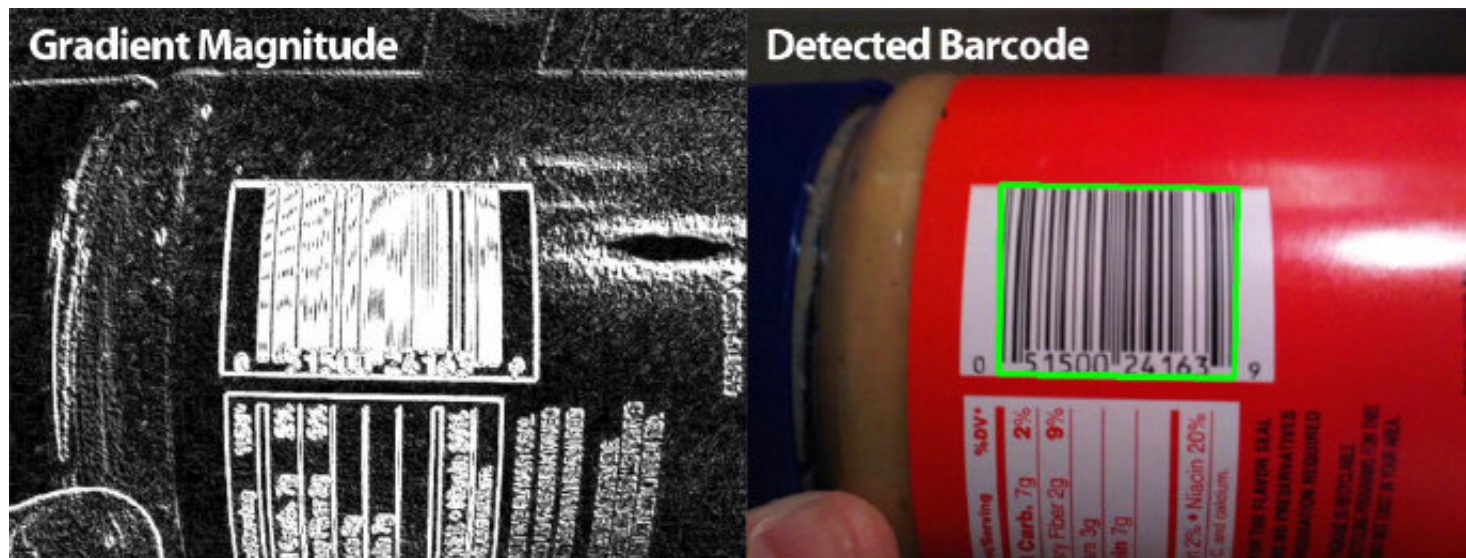
黑色星期五要来了。

疯狂的消费者成群结队, 中西部的中年女性蜂拥而出, 露出没有牙齿的嗜血牙龈, 直奔当地沃尔玛75%折扣的最新一季的《权利的游戏》。

感恩节之夜, 他们将在沃尔玛门外排起长队, 团结在一起, 用他们的双手和头部, 击打紧锁的大门, 直到身体鲜血淋漓, 就像《惊变28天》中的僵尸一样, 只不过不是为了肉身, 他们渴望小小的消费寄托, 他们的战争呐喊着折扣, 销售额将会上升到极点, 他们雷鸣般的脚

步造成整个大平原的地震。

当然，媒体也无济于事，他们将危言耸听每一个小场景。从冻伤的家庭在寒风中露营整晚，到瞒珊老太在大门打开后被蜂拥而入的低价抢购人群踩踏，就像侏罗纪公园中似鸡龙的蹂躏。这所有的一切只是因为她想为9岁的孙女蒂米买到最新的光晕游戏，而蒂米的父母，在去年的这个时候离世了，就在沃尔玛，在这黑色星期五。



我不得不问，所有的这些混乱值得么？

见鬼，当然不。

我在这个黑色星期五时的购物都是在网上完成的，就像用一杯咖啡和少量泰诺（Tylenol）护理宿醉一样。

但是如果你决定外出到现实世界勇敢地低价抢购，你会想先下载本文附带的源码。

想象一下你会觉得多么愚蠢，排队，等待结账，只是为了扫描一下最新一季的《权利的游戏》上的条形码，然后查明它便宜了5美元。

接下来，我将展示给你怎样仅仅通过Python和OpenCV，来检测图片中的条形码。

用 Python 和 OpenCV 检测图片上的的条形码

这篇博文的目的是应用计算机视觉和图像处理技术，展示一个条形码检测的基本实现。我所实现的算法本质上基于[StackOverflow 上的这个问题](https://stackoverflow.com/questions/25128152/barcode-detection-with-opencv)，浏览代码之后，我提供了一些对原始算法的更新和改进。

首先需要留意的是，这个算法并不是对所有条形码有效，但会给你基本的关于应用什么类型的技术的直觉。

假设我们要检测下图中的条形码：



图1：包含条形码的示例图片

现在让我们开始写点代码，新建一个文件，命名为detect_barcode.py，打开并编码：

```
1 | 1 # import the necessary packages
2 | 2 import numpy as np
```

Python

```
3 import argparse
4 import cv2
5
6 # construct the argument parse and parse the arguments
7 ap = argparse.ArgumentParser()
8 ap.add_argument("-i", "--image", required = True, help = "path to the image file")
9 args = vars(ap.parse_args())
```

我们首先做的是导入所需的软件包，我们将使用NumPy做数值计算，argparse用来解析命令行参数，cv2是OpenCV的绑定。

然后我们设置命令行参数，我们这里需要一个简单的选择，- image是指包含条形码的待检测图像文件的路径。

现在开始真正的图像处理：

```
1 11 # load the image and convert it to grayscale
2 12 image = cv2.imread(args["image"])
3 13 gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
4 14
5 15 # compute the Scharr gradient magnitude representation of the images
6 16 # in both the x and y direction
7 17 gradX = cv2.Sobel(gray, ddepth = cv2.CV_32F, dx = 1, dy = 0, ksize = -1)
8 18 gradY = cv2.Sobel(gray, ddepth = cv2.CV_32F, dx = 0, dy = 1, ksize = -1)
9 19
10 20 # subtract the y-gradient from the x-gradient
11 21 gradient = cv2.subtract(gradX, gradY)
12 22 gradient = cv2.convertScaleAbs(gradient)
```

Python

12~13行：从磁盘载入图像并转换为灰度图。

17~18行：使用Scharr操作（指定使用ksize = -1）构造灰度图在水平和竖直方向上的梯度幅值表示。

21~22行：Scharr操作之后，我们从x-gradient中减去y-gradient，通过这一步减法操作，最终得到包含高水平梯度和低竖直梯度的图像区域。

上面的gradient表示的原始图像看起来是这样的：



图:2: 条形码图像的梯度表示

注意条形码区域是怎样通过梯度操作检测出来的。下一步将通过去噪仅关注条形码区域。

```
1 24 # blur and threshold the image
2 25 blurred = cv2.blur(gradient, (9, 9))
3 26 (_, thresh) = cv2.threshold(blurred, 225, 255, cv2.THRESH_BINARY)
```

25行：我们要做的第一件事是使用9*9的内核对梯度图进行平均模糊，这将有助于平滑梯度表征的图形中的高频噪声。

26行：然后我们将模糊化后的图形进行二值化，梯度图中任何小于等于255的像素设为0（黑色），其余设为255（白色）。

模糊并二值化后的输出看起来是这个样子：



图3：二值化梯度图以此获得长方形条形码区域的粗略近似

然而，如你所见，在上面的二值化图像中，条形码的竖杠之间存在缝隙，为了消除这些缝隙，并使我们的算法更容易检测到条形码中的“斑点”状区域，我们需要进行一些基本的形态学操作：

```
1 28 # construct a closing kernel and apply it to the thresholded image
2 29 kernel = cv2.getStructuringElement(cv2.MORPH_RECT, (21, 7))
3 30 closed = cv2.morphologyEx(thresh, cv2.MORPH_CLOSE, kernel)
```

29行：我们首先使用cv2.getStructuringElement构造一个长方形内核。这个内核的宽度大于长度，因此我们可以消除条形码中垂直条之间的缝隙。

30行：这里进行形态学操作，将上一步得到的内核应用到我们的二值图中，以此来消除竖杠间的缝隙。

现在，你可以看到这些缝隙相比上面的二值化图像基本已经消除：



图4：使用形态学中的闭运算消除条形码竖条之间的缝隙

当然，现在图像中还有一些小斑点，不属于真正条形码的一部分，但是可能影响我们的轮廓检测。

让我们来消除这些小斑点：

```
1 32 # perform a series of erosions and dilations
2 33 closed = cv2.erode(closed, None, iterations = 4)
3 34 closed = cv2.dilate(closed, None, iterations = 4)
```

我们这里所做的是首先进行4次腐蚀（erosion），然后进行4次膨胀（dilation）。腐蚀操作将会腐蚀图像中白色像素，以此来消除小斑点，而膨胀操作将使剩余的白色像素扩张并重新增长回去。

如果小斑点在腐蚀操作中被移除，那么在膨胀操作中就不会再出现。

经过我们这一系列的腐蚀和膨胀操作，可以看到我们已经成功地移除小斑点并得到条形码区域。

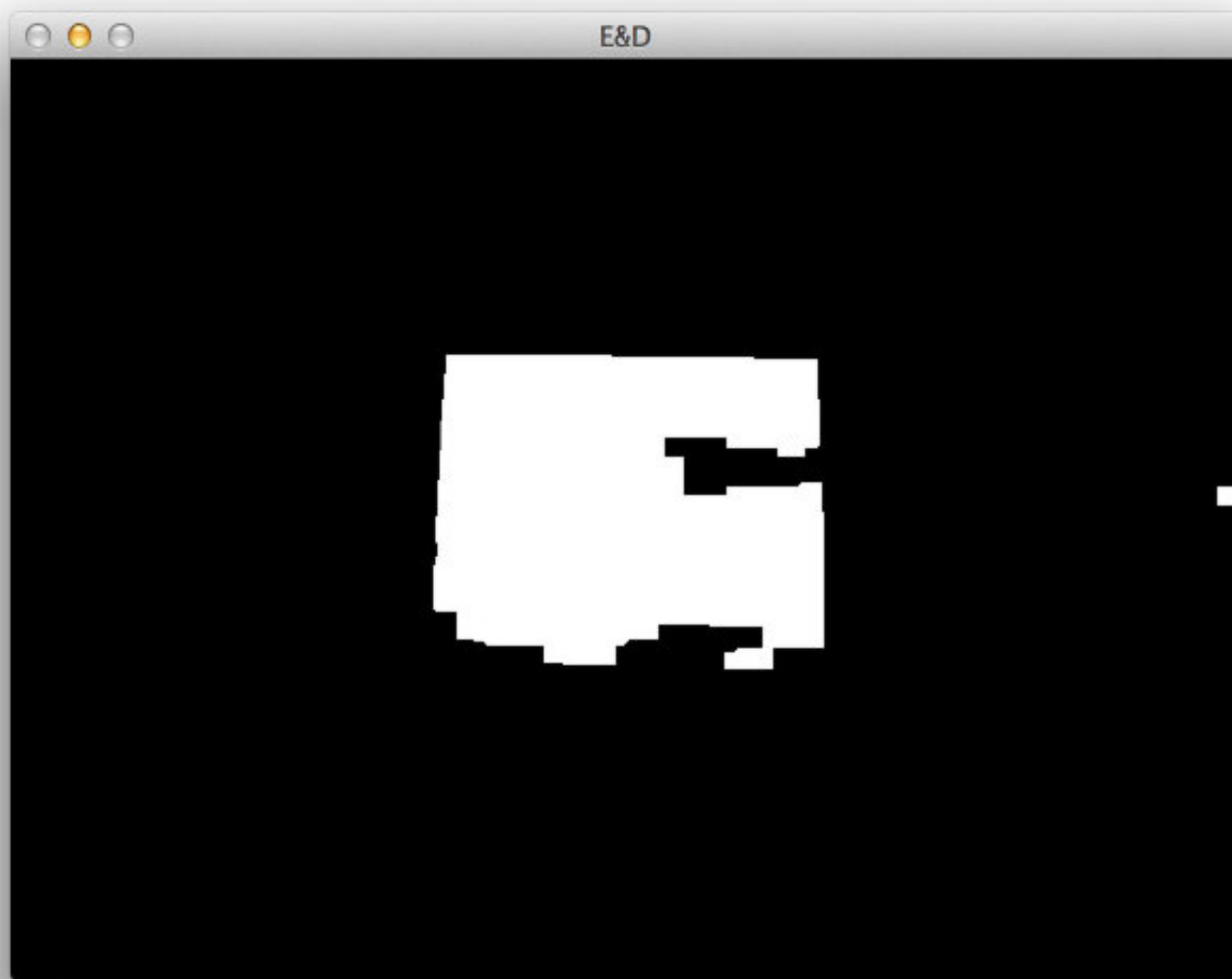


图5：应用一系列的腐蚀和膨胀来移除不相关的小斑点

最后，让我们找到图像中条形码的轮廓：

```
1 36 # find the contours in the thresholded image, then sort the contours
2 37 # by their area, keeping only the largest one
3 38 (cnts, _) = cv2.findContours(closed.copy(), cv2.RETR_EXTERNAL,
4 39 cv2.CHAIN_APPROX_SIMPLE)
5 40 c = sorted(cnts, key = cv2.contourArea, reverse = True)[0]
6 41
7 42 # compute the rotated bounding box of the largest contour
8 43 rect = cv2.minAreaRect(c)
9 44 box = np.int0(cv2.cv.BoxPoints(rect))
10 45
11 46 # draw a bounding box around the detected barcode and display the
12 47 # image
13 48 cv2.drawContours(image, [box], -1, (0, 255, 0), 3)
14 49 cv2.imshow("Image", image)
15 50 cv2.waitKey(0)
```

38~40行：幸运的是这一部分比较容易，我们简单地找到图像中的最大轮廓，如果我们正确完成了图像处理步骤，这里应该对应于条形码区域。

43~44行：然后我们为最大轮廓确定最小边框

48~50行：最后显示检测到的条形码

正如你在下面的图片中所见，我们已经成功检测到了条形码：



图6：成功检测到示例图像中的条形码

下一部分，我们将尝试更多图像。

成功的条形码检测

要跟随这些结果，请使用文章下面的表单去下载本文的源码以及随带的图片。

一旦有了代码和图像，打开一个终端来执行下面的命令：

Python

```
1 | $ python detect_barcode.py --image images/barcode_02.jpg
```



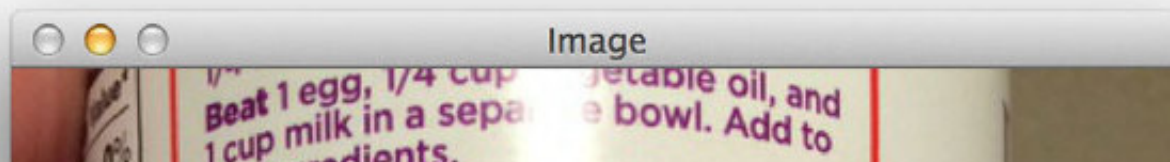


图7：使用OpenCV检测图像中的一个条形码

检测椰油瓶子上的条形码没有问题。

让我们试下另外一张图片：

```
1 | $ python detect_barcode.py --image images/barcode_03.jpg
```



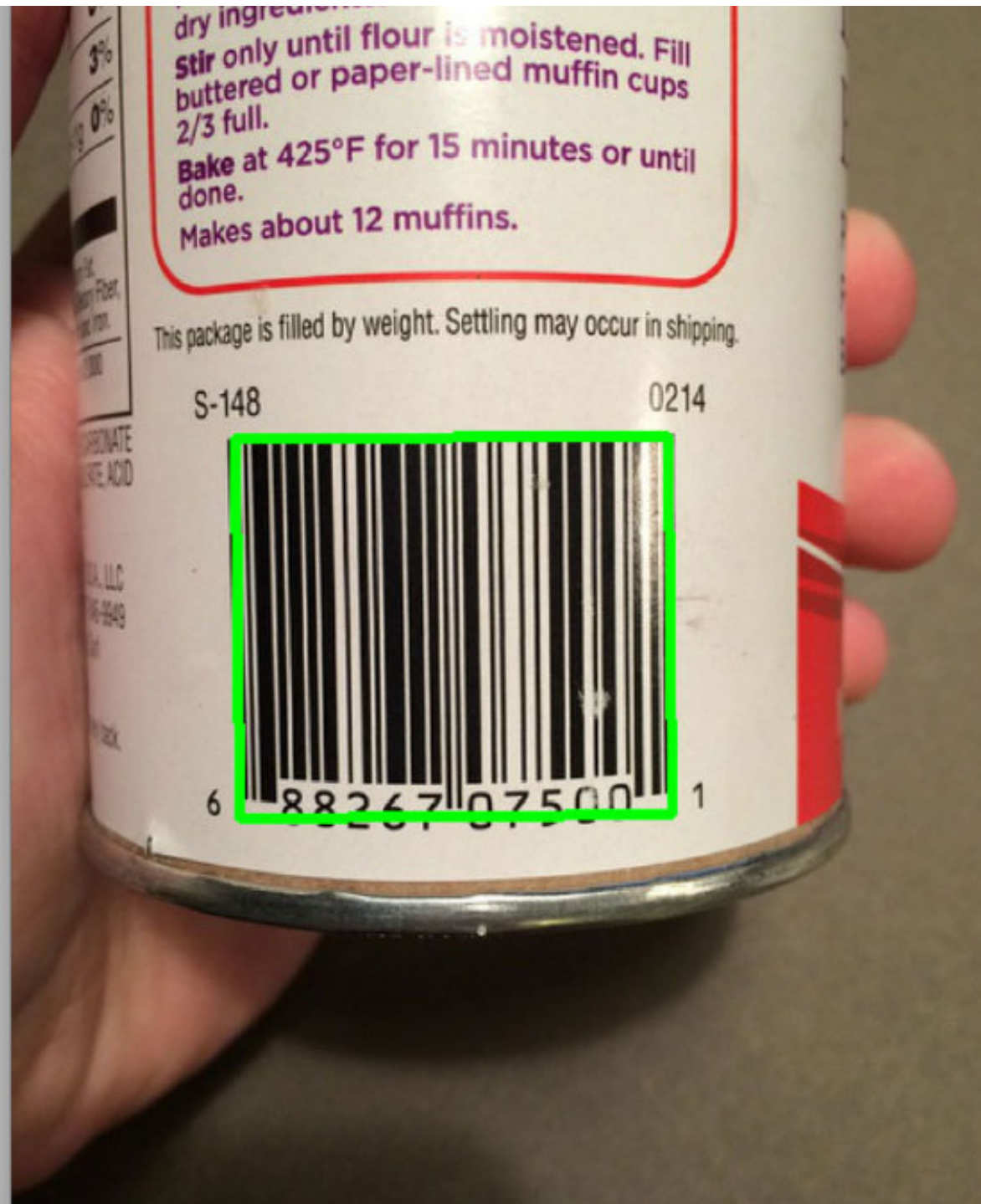


图8：使用计算机视觉检测图像中的一个条形码

我们同样能够在上面的图片中找到条形码。

关于食品的条形码检测已经足够了，书本上的条形码怎么样呢：

Python

```
1 | $ python detect_barcode.py --image images/barcode_04.jpg
```



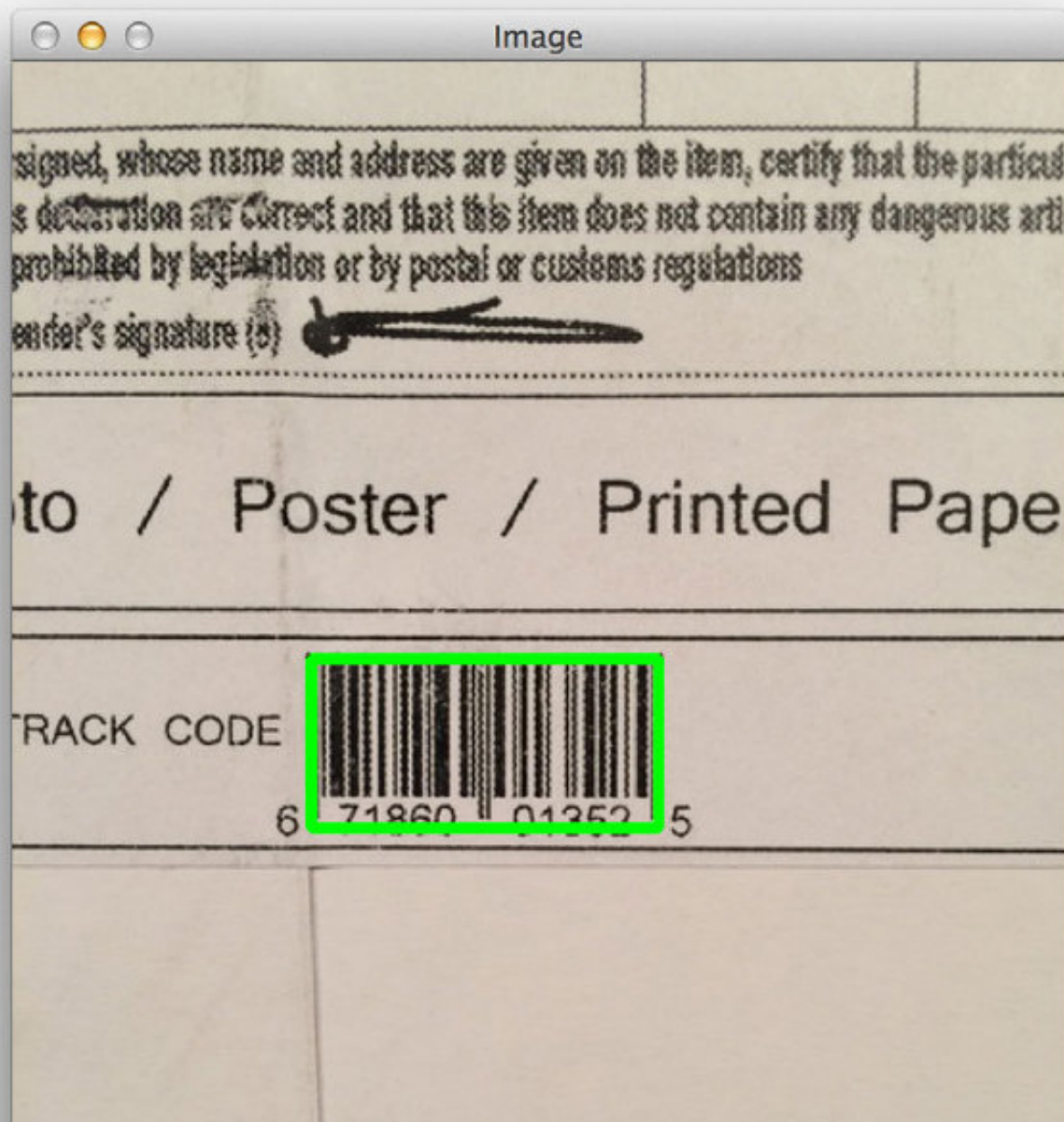


图9：使用Python和OpenCV检测书本上的条形码

没问题，再次通过。

那包裹上的跟踪码呢？


```
1 $ python detect_barcode.py --image images/barcode_05.jpg
```



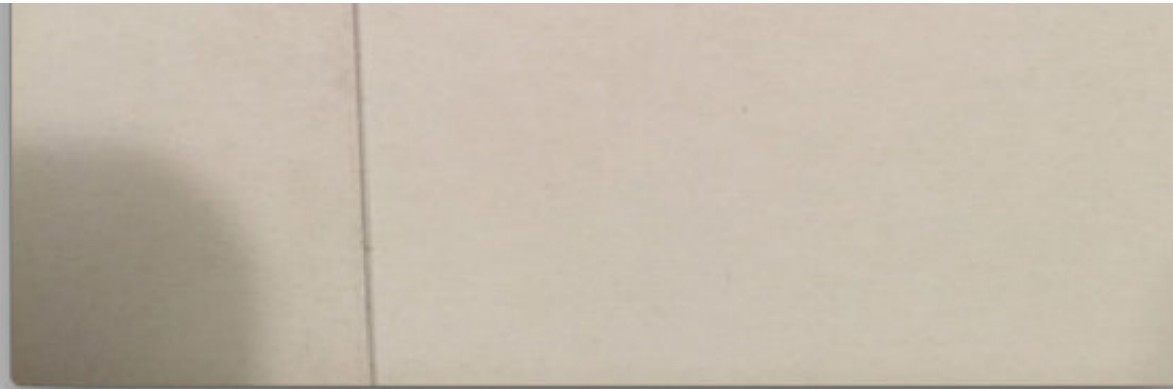
[首页](#) [资讯](#) [文章](#) [资源](#) [小组](#) [❤ 相亲](#)[频道](#)[🔑 登录](#)[👤 注册](#)[?](#)

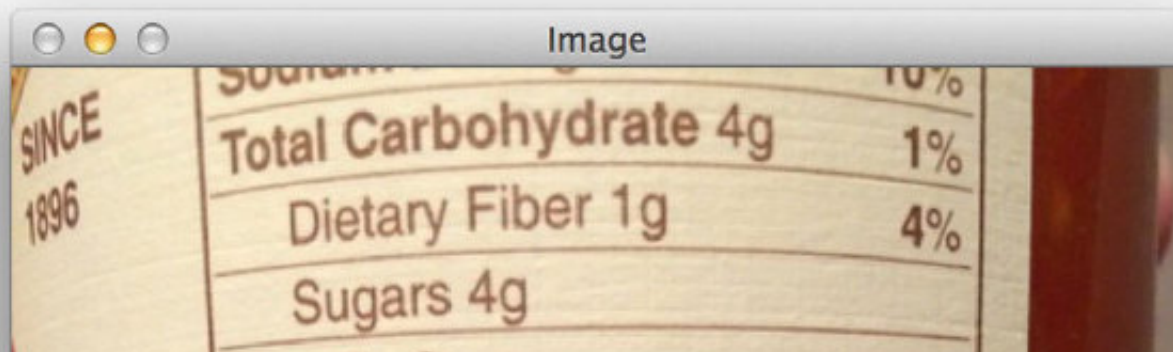
图10：使用计算机视觉和图像处理检测包裹上的条形码

我们的算法再次成功检测到条形码。

最后，我们再尝试一张图片，这个是最爱的意大利面酱—饶氏自制伏特加酱（Rao's Homemade Vodka Sauce）：

Python

```
1 | $ python detect_barcode.py --image images/barcode_06.jpg
```



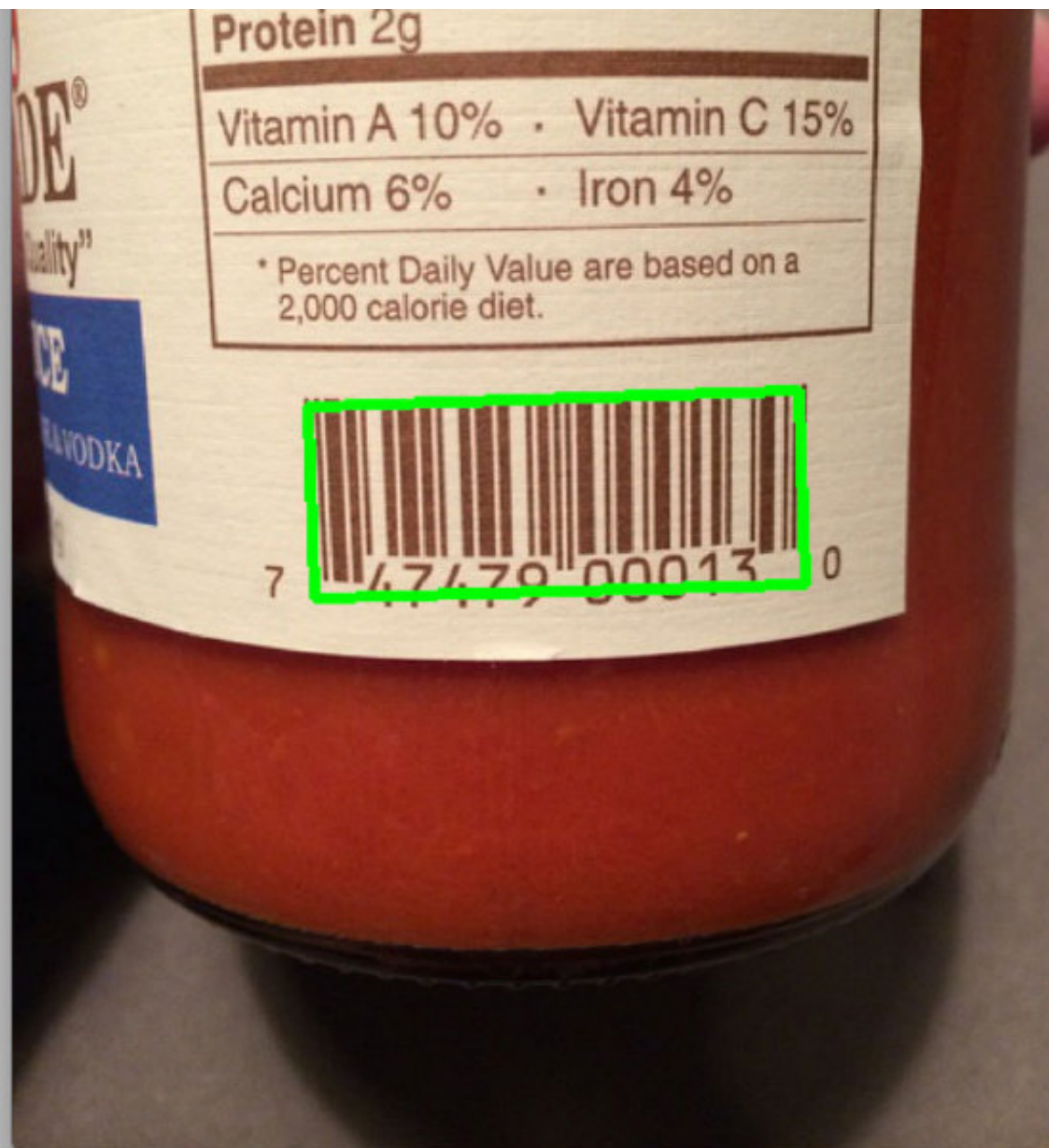


图11：使用Python和Opencv很容易检测条形码

我们的算法又一次检测到条形码！

总结

这篇博文中，我们回顾了使用计算机视觉技术检测图像中条形码的必要步骤，使用Python编程语言和OpenCV库实现了我们的算法。

算法概要如下：

1. 计算x方向和y方向上的Scharrr梯度幅值表示
2. 将x-gradient减去y-gradient来显示条形码区域
3. 模糊并二值化图像
4. 对二值化图像应用闭运算内核
5. 进行系列的腐蚀、膨胀
6. 找到图像中的最大轮廓，大概便是条形码

需要注意的是，该方法做了关于图像梯度表示的假设，因此只对水平条形码有效。

如果你想实现一个更加鲁棒的条形码检测算法，你需要考虑图像的方向，或者更好的，应用机器学习技术如Haar级联或者HOG + Linear SVM去扫描图像条形码区域。

源码下载：<http://pan.baidu.com/s/1jGMfcBs>



赞



8 收藏



2 评论

关于作者：Halal



w h o c a r e s



个人主页



我的文章



12



相关文章

- [用 Linux、Python 和树莓派酿制啤酒](#)
- [为什么 Python 增长如此之快?](#) • [🔗 2](#)
- [150 多个 ML、NLP 和 Python 相关的教程](#)
- [27 个机器学习、数学、Python 速查表](#) • [🔗 2](#)
- [如何做到老板来了就自动切换桌面（含源码）](#) • [🔗 18](#)

可能感兴趣的话题

- [一个程序员在大城市，还是选择离家近的好城市呢](#) • [🔗 1](#)
- [怎样判断一个公司有没有发展前景?](#) • [🔗 16](#)
- [该如何学习机器学习以及一些算法?](#)
- [ecstore 商城架构](#)
- [iOS APP 上架被拒，求助](#)
- [Linux 内核书籍书单](#)

[登录后评论](#)[新用户注册](#)[直接登录](#)

最新评论

伟伯 ([🎓 1](#))

2015/01/21



只能检测条形码区域，不能扫描条形码内容。

赞 回复



QINMS

2015/04/22

写得不错，对软件解条码的处理技术知之甚少，或者说是图像处理技术知之甚少。

赞 回复

- [本周热门文章](#)
- [本月热门文章](#)
- [热门标签](#)

0 [Nginx 教程（1）：基本概念](#)

1 [分析了 7500w+ GitHub 代码仓库...](#)

2 [做 ML 有关的工作，需要哪些技能？](#)

3 [一个 IT 青年北漂四年的感悟](#)

4 [免费知识哪里来？Arxiv 使用指南](#)

5 [聊聊分布式事务，再说说解决方案](#)

6 [堆和堆的应用：堆排序和优先队列](#)

7 [3 个可以替代 Emacs 和 Vim ...](#)

8 [揭秘 Quora 的 Web 服务器架构](#)

9 [采访 Node.js 之父 Ryan Dahl](#)



业界热点资讯

[更多 »](#)

[比尔·盖茨要开课了，计算机工作原理系列课程](#)

2 天前 • 43 • [🗨️ 1](#)



[英媒：中国掀科技人才争夺战 薪资水平直追美国硅谷](#)

4 天前 • 5 • [🗨️ 1](#)



[Facebook 开源了物体检测研究平台 Detectron](#)

01/23 • 20 • [🗨️ 3](#)



[Linus Torvalds 称英特尔的 Spectre 补丁是“垃圾”](#)

01/24 • 9 • [🗨️ 3](#)



[谷歌桌面系统将从 Ubuntu 迁移到基于 Debian 的 gLi...](#)

01/22 • 13 • 1



精选工具资源

更多资源 »



[mlpack: 一个C++机器学习库](#)
[C++](#), [机器学习](#)



[Whitewidow: SQL 漏洞自动扫描工具](#)
[数据库](#) • 2



[Caffe: 一个深度学习框架](#)
[机器学习](#) • 3



[静态代码分析工具清单：公司篇](#)

[静态代码分析](#)



[HotswapAgent：支持无限次重定义运行时类与资源](#)

[开发流程增强工具](#)

关于伯乐在线博客

在这个信息爆炸的时代，人们已然被大量、快速并且简短的信息所包围。然而，我们相信：过多“快餐”式的阅读只会令人“虚胖”，缺乏实质的内涵。伯乐在线内容团队正试图以我们微薄的力量，把优秀的原创文章和译文分享给读者，为“快餐”添加一些“营养”元素。

快速链接

[网站使用指南](#) »

[问题反馈与求助](#) »

[加入我们](#) »

[网站积分规则](#) »

[网站声望规则](#) »

关注我们

新浪微博：[@伯乐在线官方微博](#)

RSS：[订阅地址](#)

推荐微信号



程序猿的那些事



UI设计达人



极客范

合作联系

Email: bd@jobbole.com

QQ: 2302462408 (加好友请注明来意)

更多频道

- [小组](#) - 好的话题、有启发的回复、值得信赖的圈子
 - [头条](#) - 分享和发现有价值的内容与观点
 - [相亲](#) - 为IT单身男女服务的征婚传播平台
 - [资源](#) - 优秀的工具资源导航
 - [翻译](#) - 翻译传播优秀的外文文章
 - [文章](#) - 国内外的精选文章
 - [设计](#) - UI, 网页, 交互和用户体验
 - [iOS](#) - 专注iOS技术分享
 - [安卓](#) - 专注Android技术分享
 - [前端](#) - JavaScript, HTML5, CSS
 - [Java](#) - 专注Java技术分享
 - [Python](#) - 专注Python技术分享
-

© 2018 伯乐在线

[文章](#)

[小组](#)

[相亲](#)

[加入我们](#)

[📢 反馈](#)

[沪ICP备14046347号-1](#)

