教程| 摄影爱好者玩编程：利用Python和OpenCV打造专业级长时曝光摄影图

机器之心
百家号 | 10-08 13:22

选自pyimagesearch

机器之心编译

参与：乾树、蒋思源

在本文中，我们将学习如何使用 OpenCV 和图像处理技术来模拟长时曝光图像。为了模拟长时曝光，我们采用了对一组图像取平均值的帧平均法。机器之心对该教程进行了简要的介绍。

机器之心
百家号 | 最近更新：10-08 13:22

简介: 专业的人工智能媒体和产业服务平台

作者最新文章

从大间隔分类器到核函数：全面理解支持向量机

资源| 博士生开源深度学习C++库DLL：快速构建卷积受限玻尔兹曼机

教程 | 如何判断LSTM模型中的过拟合与欠拟合

相关文章

Python最全模块！看这位神级程序员整理出来…
国王的荣耀 10-08

月薪三万Python学习路线

长时曝光是摄影师最喜欢的摄影技术之一，运用长时曝光技术可以拍出展示时光流逝的图片，而这是传统技术难以企及的。我们经常使用这种技术表达流光夜景或柔顺的流水。优秀的长时曝光作品是摄影师对快门速度、光圈大小和 ISO 感光度的完美把控，那么我们如何使用 Python 和 OpenCV 库来实现这种长时曝光的效果呢？

使用长时曝光技术后，水流变得如丝般光滑，夜空中的星星也随着地球的旋转留下一道光线轨迹，车头/尾灯成为了一束光带。

长时曝光技术效果酷炫，但为了拍到这种类型的镜头，我们需要学习一些系统方法：把相机放在三脚架上，应用各种滤镜，计算曝光值等。更不用说，我们需要成为一名熟练的摄影师！

作为一名计算机视觉研究员和程序员，本文作者知道很多关于图像处理的知识。虽然他是个菜鸟摄影师，但有一种通过应用多帧图像平均法来模拟长时

PHP基础学习4.深入循环之for循环

三个没劲 10-02

python中应用广泛的装饰器要如何学习？

潮白家 10-07

干货｜一张图告诉你:人生苦短,请用Python!

东方头条 10-04

而且由于视频实际上是一系列的图像，我们可以通过计算视频中的所有帧的平均值来实现长时曝光效果。如此得到的是令人惊叹的长时曝光效果。

用 OpenCV 和 Python 实现长时曝光效果

这篇文章分为三部分。在本文的第一部分，我们将讨论如何通过帧平均法来模拟长时间曝光效果。随后我们将编写为输入视频创建长时曝光效果的 Python 和 OpenCV 代码。最后，我们将在一些样例视频上使用我们的代码，以创建酷炫的长时曝光图像。

通过多帧图像平均法模拟长时曝光效果

通过平均数模拟长时曝光的想法由来已久。事实上，如果我们去浏览热门的摄影网站，就会找到有关如何使用相机和三脚架手动实现这类效果的教程。

我们今天的目标是简单地实现这种方法，所以我们使用 Python 和 OpenCV 自动为输入视频创建长时曝光效果的图像。给定一个输入视频，我们将计算所有帧的平均值（加权平均）以创建长时曝光效果。

注意：我们也可以使用多个连续图像创建这种长时曝光效果，但是由于视频的实质是一系列图像，因此使用视频演示此技术更容易。在将此技术应用到自定义图像时，请牢记这一点。我们看到，代码并不复杂，并且在应用于使用三脚架捕获的视频时（不要抖动相机）效果很好。

OpenCV 实现模拟长时曝光效果

我们首先创建一个名为 long_exposure.py 的新文件，然后插入以下代码：

argparse.ArgumentParser()ap.add_argument("-v", "--video", required=True,help="path to input video file")ap.add_argument("-o", "--output", required=True,help="path to output'long exposure'")args = vars(ap.parse_args())

2-4 行导入软件包，因此我们需要预先安装 Imutils 和 opencv。如果你没有安装 imutils 模块，可以通过 pip 安装：

$ pip install --upgrade imutils

如果你的电脑没有安装配置 OpenCV，那么请自行搜索 OpenCV 3 的安装教程，并选择适合你系统的安装方式。

我们在 7-12 行解析命令行参数。

--video : 视频文件目录路径

--output : 输出「长时曝光」图像的路径+文件名

接下来执行一些初始化步骤：

initialize the Red, Green, and Blue channel averages, along with# the total number of frames read from the file(rAvg, gAvg, bAvg) = (None, None, None)total = 0 # open a pointer to the video fileprint("[INFO] opening video file pointer...")stream = cv2.VideoCapture(args["video"])print("[INFO] computing frame averages (this will take awhile)...")

对于本教程，我们正在使用包含所有帧的视频文件，因此有必要在 21 行创建一个捕获视频流的文件指针。

现在我们进入计算平均值的循环语句中：

loop over frames from the video file streamwhile True:# grab the frame from the file stream(grabbed, frame) = stream.read() # if the frame was not grabbed, then we have # reached the end of the sfile if not grabbed: break # otherwise, split the frmae into its respective channels(B, G, R) = cv2.split(frame.astype("float"))

在循环语句中，我们将从流中捕获帧（27 行），并将帧各自分解到对应的 BGR 通道变量（35 行）。请注意循环语句退出条件 ：如果未从视频文件流的末尾抓取帧，我们将退出循环（31 行和 32 行）。

我们将在循环语句的其它部分执行平均值计算：

if the frame averages are None, initialize them if rAvg is None:rAvg = RbAvg = BgAvg = G # otherwise, compute the weighted average between the history of# frames and the current frames else:rAvg = ((total * rAvg) + (1 * R)) / (total + 1.0)gAvg = ((total * gAvg) + (1 * G)) / (total + 1.0)bAvg = ((total * bAvg) + (1 * B)) / (total + 1.0) # increment the total number of frames read thus fartotal += 1

如果这是第一次迭代，我们在第 38-41 行上将 RGB 的初始平均值设置为抓取的第一帧的通道值（if 语句仅在第一次迭代时执行此操作）。

以浮点型总帧数（我们将分母总数加一，因为生成的是一个新帧）。我们将计算结果存储在相应的 RGB 通道平均值数组中。

最后，我们增加总帧数，以便能够保持运行时平均值（第 51 行）。一旦我们遍历完视频文件中的所有帧，我们就可以将（平均）通道值合并成一个新图像并将其写入磁盘：

merge the RGB averages together and write the output image to diskavg = cv2.merge([bAvg, gAvg, rAvg]).astype("uint8")cv2.imwrite(args["output"], avg) # do a bit of cleanup on the file pointerstream.release()

在 54 行，我们使用 cv2.merge 函数，同时指定了列表中的每个图像的通道平均值。因为这些数组包含浮点数（它们是所有帧的平均值），所以我们需要使用 astype("uint8") 函数将像素值转换为 [0-255] 的整数。

我们使用命令行参数 path + filename 在随后的第 55 行中将 avg 图像写入磁盘。我们也可以通过 cv2.imshow 函数将图像显示在屏幕上，但是由于这会花费大量的 CPU 资源来处理视频文件，所以我们只是将图像保存到磁盘以便进一步查看。

该脚本的最后一步是通过释放视频流指针（58 行）来清空内存。

长时曝光效果与 OpenCV 实现对比

我们通过处理三个示例视频来测试脚本效果。请注意，每个视频均由安装在稳定性良好的三脚架上的相机拍摄。

验结果，请参考我提供的原始视频的链接。

我们的第一个示例是 15 秒钟的水冲石头的视频，下面的视频中包含了一个样本帧：

视频地址：https://videohive.net/item/mountain-river-water-and-stones-01/16602591

Figure 1: Sample frames of river water rushing over rocks.

图 1 ：河水冲击石头的样本帧

我们只需执行以下命令以实现长时曝光效果。

averages (this will take awhile)...
real2m1.710suser0m50.959ssys0m40.207s

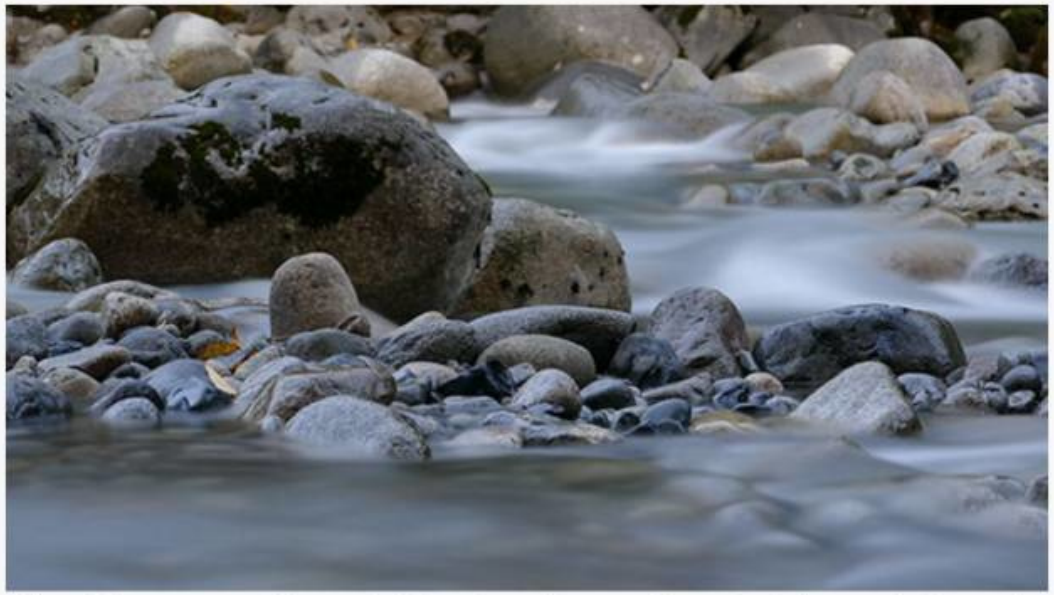

Figure 2: Long exposure of 15 seconds of river water rushing over rocks, constructed by averaging frames with Python and OpenCV.

图2:通过 Python 和 OpenCV 运用平均帧法实现的 15 秒的河水长时曝光效果图。

注意水是如何由平均法处理而得到丝滑的效果。我们继续河流的第二个例子，再次得到一幅蒙太奇效果图如下：

Figure 3: Sample frames of another river.

图 3： 另一条河流的样本帧

以下命令用于生成长时曝光效果图：

$ time python long_exposure.py --video videos/river_02.mov --output river_02.png [INFO] opening video file pointer...[INFO] computing frame averages (this will take awhile)... real0m57.881suser0m27.207ssys0m21.792s

Figure 4: Long exposure of a second river which makes the water look silky smooth (created using OpenCV).

图 4：第二条河流的丝滑的长时曝光效果图（由 OpenCV 创建）

注意静止的岩石是如何保持原状，但是湍急的河水被平均化为连续的图片，从而模拟出长时曝光效果。

最后一个例子是我最喜欢的，因为水的颜色令人赞叹，它使水和森林交相辉映：

Figure 5: Sample frames of a rushing river through a forest

图 5：激流穿越森林的样本帧

当用 OpenCV 产生长时曝光效果时，它会给你一种超现实的梦幻般的感觉：

$ time python long_exposure.py --video videos/river_03.mov --output river_03.png [INFO] opening video file pointer...[INFO] computing frame averages (this will take awhile)... real3m17.212suser1m11.826ssys0m56.707s

Figure 6: A dream-like long exposure of a river through a forest created using Python and OpenCV.

图 6：通过使用 Python 和 OpenCV 创建的梦幻般的长时曝光效果图。

才外，我们还可以考虑通过有规律的间隔从输入，从视频中对帧进行采样而不是对所有帧取平均值来构造不同的输出。

总结

在本文中，我们学习了如何使用 OpenCV 和图像处理技术来模拟长时曝光图像。为了模拟长时曝光，我们采用了对一组图像取平均值的帧平均法。我们假设输入图像/视频是使用固定的相机拍摄的（否则产生的输出图像会失真）。虽然这并非真正的「长时曝光」，但是效果上是极其（视觉上）相似

原 文 链 接 ： https://www.pyimagesearch.com/2017/08/14/long-exposure-with-opencv-and-python/

本文为机器之心编译， 转载请联系本公众号获得授权。

加入机器之心（全职记者/实习生）：hr@jiqizhixin.com

投稿或寻求报道：content@jiqizhixin.com

广告&商务合作：bd@jiqizhixin.com

> 本文仅代表作者观点，不代表百度立场。系作者授权百家号发表，未经许可不得转载。