CPE 428: Computer Vision
Homework 0: Background Subtraction
Instructor: Jonathan Ventura

For this warm-up lab, you need to set up a Python 3 environment with the OpenCV module installed. If you don't have Python 3 installed already, an easy way to install it is using Anaconda (http://anaconda.com) . To install OpenCV, you can use "pip install opencv-python."

The goal of this assignment is to give you a chance to get your environment set up and to teach you the basics of working with images and video in Python with OpenCV and Numpy. As an initial exercise, you will implement a simple "background subtraction" technique to separate foreground and background in a video.

You will submit your Python code and also a short report (Word Doc or PDF) explaining your solution and answering the questions given inline in the instructions below. Each step in the instruction is one point for a total of 15 points + 5 bonus points possible.

Part I:

1. Load 'frames/000000.jpg' and show it.
2. Print the shape of the Numpy array containing the image -- what do the sizes of the dimensions mean?
3. Print the image itself -- what do these numbers mean?
4. Convert the image to grayscale using and show it.
5. Save the grayscale image to a PNG.

Part II:

1. Set up a video capture using cv2.VideoCapture('frames/%06d.jpg').
2. Show each frame of the video.
3. Compute the average video frame by adding up all of the images (after converting to grayscale) and then dividing by the number of images. To get full points, use np.mean() instead of a for loop to compute the average. Convert the average image to 8-bit using .astype('uint8').
4. Show the average image, what we will call the "background" image. The cars have disappeared! Why?
5. Save the background image to a PNG.

Part III:

1. Load the images from parts I and II. (Pass the IMREAD_GRAYSCALE flag to ensure the images are read as grayscale.)
2. Compute the absolute difference between the background image and image from part I. To get full points, use the OpenCV function that computes the absolute difference. Show the result.
3. Threshold the absolute difference image to obtain a binary mask corresponding to the foreground pixels. To get full points, use the cv2.threshold() function. You need to manually determine the best threshold value.
4. Threshold the absolute difference image using Otsu's method (see the thresholding tutorial) and show the result.
5. How well does each technique work? What could be improved about the output?

Bonus:

Run the thresholding technique on each frame of the video and show the result as a movie. Detect a bounding box around each car and show the result as a movie.

Resources:

These OpenCV tutorials will be helpful for you:

- Getting Started with Images
- Getting Started with Video
- Image Thresholding
- Contours: Getting Started
- Contour Features
- Drawing Functions in OpenCV

Required libraries:

- imageio
- opencv
- numpy

Useful functions:

- cv2.imread() and cv2.imwrite() for reading and writing images
- cv2.cvtColor() to convert to grayscale
- cv2.imshow() to show images
- np.mean() to compute the mean
- cv2.threshold() for thresholding
- np.stack() to combine separate arrays into a multi-channel image
- cv2.waitKey() to wait for keypress if you want

Link to video:

- frames.zip

Deliverable:

- Python source file(s)
- Word Doc / PDF with your solution and answers to reflection questions

Deadline: Sunday, Oct 1, 11:59pm