Princeton University Computer Science

# COS429: Computer Vision
# Assignment 0: Setup

*Due: N/A, Feb 5th is a good target*

**Guidelines:**

1. You may complete assignments either individually or in pairs, whichever you find most conducive to your learning. While you may discuss questions at a high level with other students, all written work and code must be solely the work of you and your partner.
2. Homework is to be submitted via Gradescope. If you work with a partner, only one submission needs to be made. You can submit as often as you need (we suggest doing a trial submission well in advance of the deadline). However, only the last submission is graded, and submissions made after the deadline may incur a late penalty depending on your remaining free late days. You must label all the pages of every question on Gradescope when prompted.
3. Consulting course staff (e.g. TAs, UCAs) and/or other students from this course is allowed. Please refer to our collaboration policy for guidance on what is and isn't acceptable. **List your collaborators at the top of your submission.** You should also **cite all sources** you used per question if they aren't a required or optional course reading.
4. Submissions must be legible. Typesetting your submissions in LaTeX is encouraged. If you've never used LaTeX you can refer to the short guide here: `http://bit.ly/WorkingWithLaTeX`. You may submit a scan or photograph of hand-written work, but ensure the contrast is sufficient for reading.

## 1. Installing Python 3.8 through Miniconda

The first thing that you need to do is set up the Python environment. In this class, we will be using the `conda` package management system. If you currently have Python installed, but not through `conda`, we **highly** recommend you still go through this tutorial, since this makes environment and package management very simple. If you already have Anaconda or Miniconda installed, skip ahead to step 2.

(1) **Install Miniconda**. The installation guide for Miniconda for each operating system can be found here: `https://conda.io/projects/conda/en/latest/user-guide/install/index.html`
   - Make sure you install Miniconda, rather than Anaconda. There's no problem if you install Anaconda instead, but that comes with far more tools than this course will need, making installation more cumbersome.

- Follow the regular installation instructions for your operating system. These instructions have a step to verify installer hashes for security purposes, but this step is not strictly necessary.

(2) **Create a virtual environment for this class** To keep dependencies and packages consistent between all members of the class, we are going to create a virtual environment. Think of an environment as a separate installation of python. Packages installed in an environment stay localized to it, so by creating an environment purely for this class, we reduce the risk of having packages from other classes interfering with our dependencies. You can read up on virtual environments here: `https://docs.conda.io/projects/conda/en/latest/user-guide/tasks/manage-environments.html`, but the TL;DR for creating an environment with Python 3.8 and name cos429 is as follows:

- Open up terminal (if you're on MacOS) or the Anaconda Prompt.
- To create an environment with python 3.8 installed, type
  `conda create --name cos429 python=3.8`
- To activate the environment, type
  `conda activate cos429`
- To deactivate the environment, type
  `conda deactivate`

Note: You'll only create the environment once, but any time you want to do work for this class, from writing code to installing new packages, always activate the environment first. De-activate the environment when you're not doing cos429 work.

(3) **Installing necessary packages.** Luckily, installing packages using conda management system is a breeze. For now, you're going to install some packages that will definitely be needed in the class, but if additional packages will be needed later on, the TAs will also provide the proper command to be typed into terminal/anaconda prompt. Make sure that you have your cos429 environment activated (conda activate cos429), and then type the following lines one at a time:

- `conda install -c anaconda numpy`
  - `numpy` allows for matrix computations.
- `conda install -c anaconda opencv`
  - `opencv` is an image processing library that, in conjunction with `numpy`, lets us work with image files.
- `conda install -c conda-forge matplotlib`
  - `matplotlib` gives us tools to plot and visualize data.
- `conda install -c anaconda scikit-image`
  - `scikit-image` gives us pre-written computer vision toolkits

Let's check to make sure everything installed properly. Open a python shell (type python in terminal or Anaconda Prompt), and try the following lines:

- `import numpy`
- `import cv2`

- `import matplotlib.pyplot as plt`
- `import skimage`

If none of these gave you errors, then the packages were successfully installed. (You can exit python shell with `exit()`). Note: It is possible that you won't be able to install OpenCV with Python 3.8. If that's the case, switch to Python 3.7 instead. You can do this by first removing the existing cos429 environment with `conda remove --name cos429 --all`, then restarting step 1.b while specifying `python=3.7` when creating the environment.

Finally, you should install Jupyter Notebook as a tool to help code and debug. Read about Jupyter here: `https://jupyter.org/`. To do this, do the following:

- `conda install -c conda-forge jupyterlab`
- `conda install -c anaconda ipykernel`
- `python -m ipykernel install --user --name=cos429`

To start a Jupyter Notebook, simply call `jupyter lab` from the terminal. When working on the assignment, make sure you are opening the ipynb files using the `cos429` kernel!

**If you have difficulties during any of these installation steps, please ask for help either through EdStem or come to Office Hours.**

## 2. Getting familiar with Python

Note: This assignment, and all subsequent ones, assume basic familiarity with Python. If you have no experience with the language, we encourage you to check the resources provided.

To familiarize yourself with how `OpenCV` and `Numpy` interact with each other in Python, go through the following exercises in a python shell or a Jupyter Notebook. Remember to first import the packages first (`import cv2, import numpy as np`) before attempting these. Work through the following tasks using an image of your choice:

(1) **Read a color image into a variable.**
- ***Hint 1***: A basic cv2 tutorial can be found here: `https://docs.opencv.org/3.0-beta/doc/py_tutorials/py_gui/py_image_display/py_image_display.html` Pay attention to flags! You want to read a color image.
- ***Hint 2***: If your image isn't in the same directory as where you launched your shell, make sure you also provide directory path in the image name.
- ***Hint 3***: Note that the color channel order is BLUE, GREEN, RED, rather than the standard RGB. The image is read in as a numpy matrix.

(2) **Display the image using Matplotlib with the following code.**

```
plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
plt.xticks([]), plt.yticks([])
plt.show()
```

(3) **Convert the image to grayscale**
- ***Hint***: Look up `cv2.cvtColor`.

(4) **Find the height and width of the image.**
- ***Hint***: Since the image is a `numpy` matrix, we can use the `.shape()` property of a matrix. Which dimension corresponds to height?

(5) **Extract the middle 100 pixels from the greyscale image.**
- ***Hint 1***: Extracting a part of a matrix is done by
  `matrix2 = matrix1[row_min:row_max,col_min:col_max]`
  Note that the first dimension is row (or y), and second dimension is column (or x). Just ":" will grab all values in that dimension.
- ***Hint 2***: Since the middle 100 pixels is also an image, you can display these pixels as well using OpenCV to make sure you did it properly.

(6) **Write a pair of nested "for" loops to set a grid of every 10th pixel horizontally and every 20th pixel vertically to 0.**

(7) **Flip the image vertically. Then create and display a new image that has the original and the flipped image side-by-side.**
- ***Hint 1***: Look up `cv2.flip()`
- ***Hint 2***: Create a new numpy matrix with twice the number of columns, and dtype `np.uint8`. Put the original image on the left side, and the flipped image on the right side.

(8) **Write the combined image back to a new file.**
- ***Hint***: Look up `cv2.imwrite()`