Drexel CS 435 Computational Photography: **Homework 1**

Instructions

- This homework is **due at 11:59:59 p.m. on Sunday Feb 2, 2025**.
- Please submit to BBLearn. The submission includes two parts:
    1. A `pdf` file as your write-up, including your plots and answers to all the questions and key choices you made.
       The write-up is preferred to be an electronic version and LaTeX is recommended but not mandatory (please use this template: `https://github.com/cvpr-org/author-kit/archive/refs/tags/CVPR2024-v2.zip`). You might like to combine several files to make a submission. Here is an example online link for combining multiple PDF files: `https://combinepdf.com/`.
    2. A `zip` file including all your code, and files specified in questions with Submit, all under the same directory. You can submit Python code in either `.py` or `.ipynb` format.

Python Environment

We are using Python 3.7 for this course. You can find references for the Python standard library here: `https://docs.python.org/3.7/library/index.html`. To make your life easier, we recommend you to install Anaconda 5.2 for Python 3.7.x from `https://www.anaconda.com/download/`. This is a Python package manager that includes most of the modules you need for this course.

We will make use of the following packages extensively in this course:

- Numpy (`https://numpy.org/doc/stable/user/quickstart.html`)
- OpenCV (`https://opencv.org/`)
- SciPy (`https://scipy.org/`)
- Matplotlib (`https://matplotlib.org/2.0.2/users/pyplot_tutorial.html`)

We will use the following packages for this homework and refer to some of them as:

- numpy as np
- matplotlib.pyplot as plt
- imageio (For gif generation only)

1 Camera Projection Matrix [60 pts]

Study the **Projection and Dolly Zoom** notebook (`dolly_zoom.py` or at[1]) and complete the following tasks:

1. Write a function `rotY()` which takes an angle theta (in radian) and outputs the 3D rotation matrix of rotating by theta about the y-axis (right-hand rule). You may refer to this Wikipedia entry: `https://en.wikipedia.org/wiki/Rotation_matrix#Basic_rotations`. After you are done, refer to the starter code to generate and **submit** `cube.gif` of a cube rotating around itself. **[10 pts]**

2. Similarly, write a function `rotX()` which rotates about the x-axis. Let $\theta = \pi/4$, consider the following two transformations:

   (a) `rotX(theta)`, followed by `rotY(theta)`

   (b) `rotY(theta)`, followed by `rotX(theta)`

   Using `renderCube()` in the same way, plot the resulting view of the cube from two transformations. Are 3D rotation matrices commutative? **[10 pts]**

3. Combine `rotX()` and `rotY()`, choose an appropriate order and a pair of parameters so that `renderCube()` draws a projection of the cube where one diagonal of the cube is projected to a single point, as shown in Figure 1 (left). **Report** the order and parameters you choose. **[20 pts]**

   Hint: The diagonal of the cube rotates together with the cube. When it projects to a single point in 2D, it is horizontal and perpendicular to the camera plane in 3D. You can either make a mathematical calculation or perform a numerical search.

4. Implement an orthographic camera by either adding a branch to function `projectLines()`, or refer to it and write a new one. Then plot the same rotated cube in the previous part with this orthographic camera. It should look like Figure 1 (right). **[20 pts]**

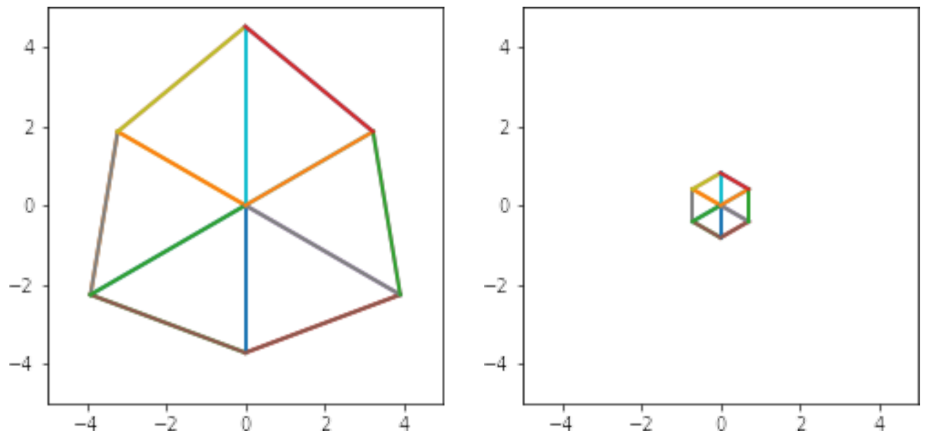

Figure 1: The diagonal of a cube projected to a single point

[1]https://colab.research.google.com/drive/1VgL6VI-xF09CLvhJQoz2VQyo8htXQ4dp

2 Color Spaces and Illuminance [40 pts]

The same color may look different under different lighting conditions. Images `indoor.png` and `outdoor.png` are two photos of a same Rubik's cube under different illuminances.

1. Load the images and plot their R, G, B channels separately as grayscale images using `plt.imshow()` (beware of normalization). Then convert them into LAB color space using `cv2.cvtColor()` and plot the three channels again. Include the plots in your report. **[10 pts]**

2. How do you know the illuminance change is better separated in LAB color space? **[10 pts]**

3. Choose two different lighting conditions and take two photos of a non-specular object. Try to make the same color look as different as possible (a large distance on AB plane in LAB space). Below is an example (Figure 2) of two photos of the same piece of paper, taken in the basement and by the window respectively.

   Submit the two images with names `im1.jpg` and `im2.jpg`, both cropped and scaled to 256x256. Under the same folder, also submit a file `info.txt` that contains two lines: Line 1 contains four integers x1, y1, x2, y2 where we will take a 32x32 patch around the coordinate on each image and compare colors. (You can use `plt.imshow()` and `plt.show()` to bring up a window where you can select pixel with coordinates.) Line 2 is a description of the lighting conditions that you chose. An example of this file is provided for you in the folder. **[20 pts]**

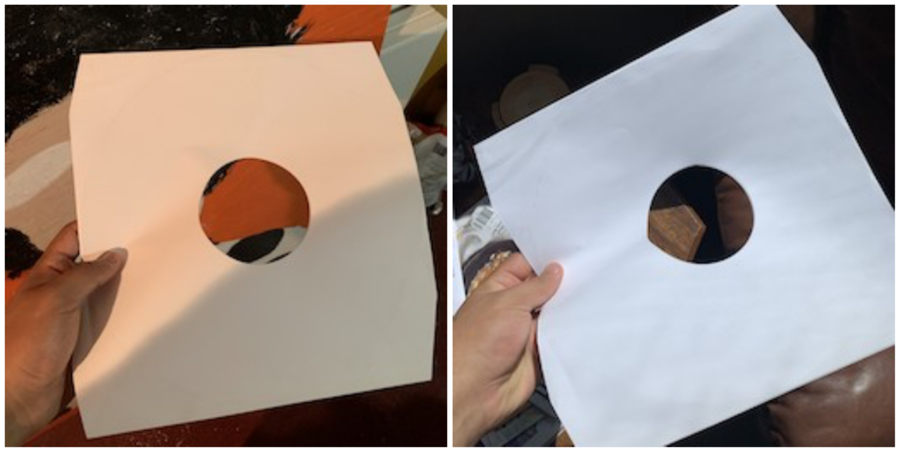

Figure 2: The same piece of paper in the basement and by the window