Homework 1

September 19, 2024

Due Date: October 3 by 23:59:59

1 Short questions (15 points)

1.1 Homogeneous coordination (10 points)

In the Cartesian coordinate system, a point in N-dimensional space is represented by N coordinates as $P = (x_1, x_2, \ldots, x_N)$. In homogeneous coordinates, the same point is represented with $N+1$ coordinates as $P' = (x_1, x_2, \ldots, x_N, w)$, where w is a non-zero real number representing the weight or scale of the point. For simplicity, P' can be rewritten as $(x'_1, x'_2, \ldots, x'_N, 1)$.

Please:

- **Derive the mathematical relationship between x_i and x'_i. (3 points)**
- **Explain at least two advantages of using homogeneous coordinates, and why they are adopted in computational photography. (7 points)**

1.2 Dolly zoom (5 points)

A dolly zoom (also known as a Hitchcock shot or Vertigo shot) is an in-camera effect that appears to undermine normal visual perception. This technique can produce striking effects, where the background appears to expand rapidly in size and detail, overwhelming the foreground, or where the foreground swells to dominate its surroundings, depending on how the dolly zoom is performed. Figure 1 illustrates a classic example of this effect. **Please explain the mechanics behind how this effect is achieved.**

(Hint: Consider camera movement and changes in the field of view (FoV).)

2 Camera parameters from the image (25 points)

Multiple sets of parallel structures on the spatial horizontal plane can provide geometric information about the camera or scene, such as camera pose and object heights.

Figure 1: An example of dolly zoom.

Figure 2 is a picture with 4096×3072 pixels of our campus. We can establish a right-handed world coordinate system where the ground is the $z = 0$ plane, the edge of the roadside lawn is the x-axis, and the edge below the flowerbed serves as the y-axis. All measurements are in meters (m). The height of the first flowerbed (red rectangle) is 0.76 meters, and the camera's coordinates are $(13.4, 4.5)$ in the x and y directions. Additionally, the edge point A of the hexagonal flowerbed has coordinates $(0, 3.74, 0.76)$ in the world coordinate system.

Figure 2: A figure of our campus is provided. Your task is to calculate the camera's height and focal length using the information from this image.

Please:

- **Find the vanishing line and calculate the camera height H. (10 points)**
- **Write down the intrinsic parameter matrix and derive the camera focal length f. (15 points)**

Please draw the lines and leave your calculation process explicitly.

(Hint: For question 1, you can refer to Figure 3. For question 2, express the focal length f in terms of the intrinsic matrix elements (as well as R and T) and image dimensions.)

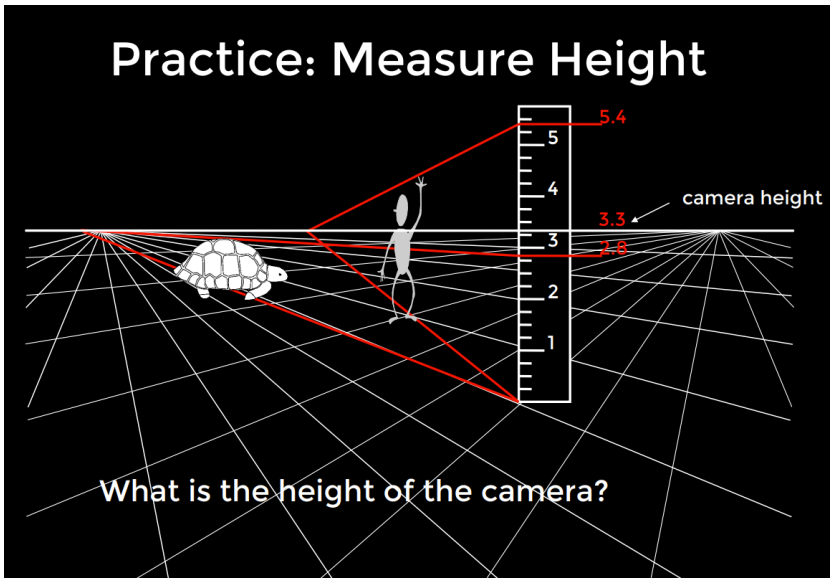

Figure 3: A visual explanation for question 1. The vanishing line indicates relative camera height, where you may use a reference object to compute the actual height.

3 Image filtering and subsampling (60 points)

This assignment is designed to help you understand and implement key image processing algorithms using Python and NumPy. You will implement six core functions, each building on the previous ones (except for functions 3 and 5). The task requires you to write the code from scratch without using pre-built libraries (like OpenCV). The functions you will implement are:

1. `cross_correlation_2d`
2. `convolve_2`
3. `gaussian_blur_kernal_2d`
4. `low_pass`
5. `image_subsampling`
6. `gaussian_pyramid`

3.1 Image Filtering. (20 points)

Image filtering (or convolution) is a fundamental image processing tool. Refer to lecture materials, technical blogs or chapter 3.2 of Szeliski to learn more about image filtering (specifically Gaussian filtering). You are asked to write down your function from scratch for this assignment without using pre-implemented libraries. More specifically, you will implement `cross_correlation_2d`, followed by `convolve_2d`, which would use `cross_correlation_2d`.

3.2 Gaussian Blur. (10 points)

Blurring an image can be done in several ways, such as by computing an unweighted average of neighboring pixels. Gaussian blur, however, uses a weighted average where nearby pixels have a higher influence. You will implement the function `gaussian_blur_kernel_2d`, which generates a Gaussian kernel of specified size and standard deviation. This kernel will then be passed to `convolve_2d` to produce a blurred version of the image.

3.3 Low Pass Filters. (10 points)

Low pass filters are used to remove high-frequency details from an image, making it appear smoother. By applying a Gaussian blur, you can create a low pass filter. Implement the `low_pass` function, which uses the Gaussian blur technique to filter out fine details from the image.

3.4 Image subsampling/downsampling. (10 points)

Subsampling reduces the resolution of an image by discarding rows and columns. Given an image of dimensions $M \times N$, subsampling by a factor of s will result in a new image with dimensions $(M/s) \times (N/s)$. The factor s should be a common divisor of both M and N. In this problem, you are asked to implement a simple downsampling technique: discard every alternate row and column to produce an image that is half the size of the original. This is equivalent to keeping either all even or all odd rows and columns.

3.5 Gaussian pyramid. (20 points)

A Gaussian pyramid is a multi-scale representation of an image, where each level is progressively blurred and downsampled. At each level of the pyramid, the image is smoothed using a Gaussian filter, and then downscaled to reduce its resolution. The

pixels in the downsampled image represent local averages of pixel neighborhoods from the previous level. In this task, you are required to create a Gaussian pyramid with 4 levels: the original image, followed by images at half, quarter, and one-eighth resolutions. You can adjust a parameter to control the amount of high-frequency content removed from the image at each level.

Requirements:

1. Please complete this assignment independently. Avoid using generative AI tools for coding. If you do use AI, document the AI-assisted code and provide a detailed process analysis in your report. Grading will be based on this analysis. If AI-generated code is found and fails to run, that portion will receive zero points.

2. Do not use libraries such as Numpy, Scipy, or OpenCV to perform the core image processing tasks directly. However, you may use OpenCV, PIL, or matplotlib solely for reading and saving images. Basic matrix operations like `np.shape`, `np.zeros`, `np.flip`, `np.transpose`, `np.exp` etc. are allowed. For efficient code, leverage Numpy vectorization and reduce nested loops. You are also encouraged to call your own functions where necessary.

3. We provide two images for the Gaussian pyramid task, but you are encouraged to test your implementation with additional images of your choice.

4. Prepare a report with your answers for questions 1 and 2. For question 3, create a Python script named `gaussian_pyramid.py` to implement the required functions. Your script should read an image, apply the Gaussian pyramid transformations, and **save** the resulting images at 1/2, 1/4, and 1/8 resolutions as separate files. Verify that all image paths (both input and output) in `gaussian_pyramid.py` are relative paths, so the script runs correctly and produces the Gaussian pyramid images.

5. Organize your submission by placing the `gaussian_pyramid.py` script, all image files, and your report into a single folder. Zip the folder and name it "YOURNAME_YOURID_HW1".