EECS 504 Foundations of Computer Vision: PS1

Term: Winter 2026
Instructor: Jason J. Corso (jjcorso)
GSI: Haotian Qiao (qhaotian)
Due Date: 02/06 23:59 Eastern Time

Starter code: The starter code can be found at: Starter Code. We recommend editing and running your code in Google Colab, although you are welcome to use your local machine instead.

Submission Process:

1. To Gradescope: a `pdf` file as your write-up, including your answers to all the questions as well as the result images from the code. *Please do not handwrite.* The write-up must be electronic. You can use Word, Google Docs, LaTeX(try overleaf!), or any similar application.
2. To Canvas: The .ipynb file containing all the visualizations should be submitted to Canvas. Before submitting, please make sure to rename the file to EECS504_PS1_<your_uniquename>_<your_umid>.ipynb

Grading and Evaluation: The credit for each problem in this set is given in parentheses at the stated question (sub-question fraction of points is also given at the sub-questions). Partial credit will be given for both paper and python questions. The total credit for this assignment is 100 points.

Question and Discussion: You can either attend office hours or comment on Perusall (in which there is a PS1 discussion group, and this pdf is posted into the Library).

Constraints: This assignment may be discussed with other students in the course but must be written independently. Programming assignments should be written in python. Over-the-shoulder coding is strictly prohibited. You may not prompt or query AI agents for the direct solutions to any of the questions. **Web/Google-searching for background material is permitted. However, everything you need to solve these problems is presented in the course notes and background materials, which have been provided already.**

Goals: Test mathematical basics and deepen the understanding of image as functions material.

Problem 1 (15): Lambertian Model

Recall the Lambertian model of reflectance: $R(\mathrm{x}) = \rho \boldsymbol{\ell}(\mathrm{x})^{\mathrm{T}} \mathbf{n}(\mathrm{x})$, as shown in Fig. 1.

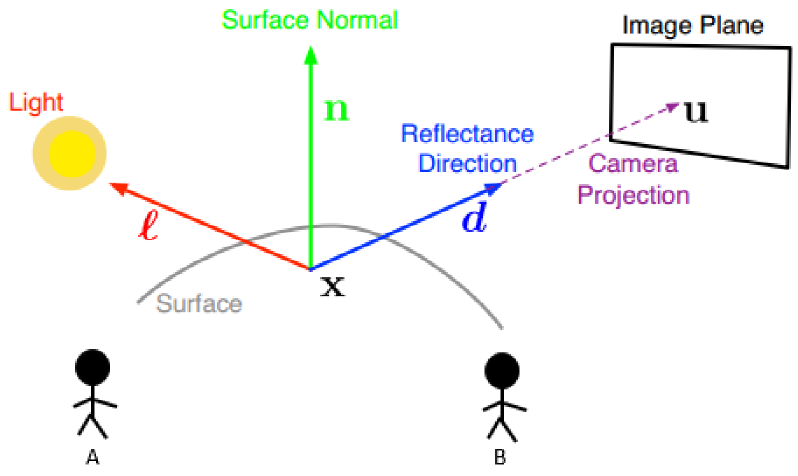

Figure 1: Lambertian model

1. (8) There are two people standing at location A and B. Who will observe stronger reflection at point x? Why?
2. (7) Tell one drawback of Lambertian model and name a specific object in the real world whose reflectance is not well modeled by the Lambertian model.

Problem 2 (15): Degenerate Line Parameterization

Let's go back to a simple least-squares problem: fitting a line. In the slope-intercept model, a line is formulated as $y = mx + b$.

Then we want to minimize the estimation error

$$E(m, b) = \sum_{i=1}^{n} (y_i - mx_i - b)^2. \tag{1}$$

The limitation of this formulation is: it is degenerate for exactly vertical lines. Propose a reformulation of the least-squares problem to overcome this limitation. The reformulation should operate directly on the sampled points and not rotate them to overcome the limitation.

Include the reformulated model, the least squares formulation to fit its parameters, the matrix formulation, and the answer to ”Can you solve it with the pseudo-inverse?” in your writeup.

Hint: You may need to consider alternative parameterization of the line.

Problem 3 (10): Rotation Invariance of DoG

We discussed the Difference of Gaussians operator. Show that this operator is invariant to rotations in the image plane. Concretely, for image $\mathbf{I}$ and Gaussian kernels $\boldsymbol{\kappa}_1$ and $\boldsymbol{\kappa}_2$ with respective parameters σ_1 and σ_2, consider the two convolved images $\mathbf{G}_1 = \boldsymbol{\kappa}_1 \otimes \mathbf{I}$ and $\mathbf{G}_2 = \boldsymbol{\kappa}_2 \otimes \mathbf{I}$. We know that

$$\mathbf{J} = \mathbf{G}_2 - \mathbf{G}_1 = \boldsymbol{\kappa}_2 \otimes \mathbf{I} - \boldsymbol{\kappa}_1 \otimes \mathbf{I} = (\boldsymbol{\kappa}_2 - \boldsymbol{\kappa}_1) \otimes \mathbf{I} \tag{2}$$

Show the operator $\boldsymbol{\kappa}_2 - \boldsymbol{\kappa}_1$ is rotationally invariant.

Problem 4 (20): Mapping Kernels, Cost and Separability

Kernel-based spatial range maps allow for operator generality but can be computationally expensive.

1. (5) For an image of size $n \times n$ and a kernel of size $2k+1 \times 2k+1$ what is the total number of operations in terms of multiplies and adds to map the kernel over the image (to convolve the image by the kernel) only apply it in *valid* locations where the kernel fits fully over the image lattice Λ.
2. (5) For certain convenient forms of a $2D$ kernel, such as one that samples a Gaussian weighting function of a given variance σ^2, this cost can be reduced by sequential operations of a $1D$ kernel. Show that a convolution by a 2D Gaussian kernel is equivalent to two sequential operations (one vertical and one horizontal) of the appropriate 1D Gaussian kernel. Specify the relationship between the 2D and 1D Gaussian kernel. (Hint: observe the linearity of convolution.)
3. (5) For this separable kernel of size $2k + 1 \times 1$ and an image of size $n \times n$, how many operations (multiplies and adds) are needed to achieve the equivalent operation as if we just applied the $2D$ kernel (i.e., nonseparable).
4. (5) Make an argument for or against whether or not we can leverage this separability when looking for step edges in images as zero crossings of the Laplacian operator output.

Problem 5 (20): Estimate homography with Linear Least Squares

We discussed the least-square estimation for affine transformation in the lecture. As you may know, affine transformation preserves parallelism, for example, a square could become a parallelogram after affine transformation but never be a trapezoid. Now we consider a more general case where only straight lines are preserved in the transformed space while other geometrical properties are not guaranteed. This type of transformation is called projective transformation, which is also named as homography. The transformation can be described mathematically as following:

$$w \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}, \tag{3}$$

where $p = [x, y, 1]$ and $p' = [x', y', 1]$ are homogeneous coordinates of the original and the transformed point respectively. w is a constant to make sure that the third element of p' is 1.

1. (6) Assume that we've known a list of pairs of points under homography, $\{(x_1, y_1), (x'_1, y'_1)\}, \{(x_2, y_2), (x'_2, y'_2)\}, ...,$ $\{(x_n, y_n), (x'_n, y'_n)\}$. We could use the similar strategy in affine transformation estimation: substitute pairs of points into (3) and reformulate it to the form of standard least-squares problem

$$\min \|Ax - y\|_2^2. \tag{4}$$

Write down what A, x and y are given the points above. Also include your derivation in the writeup.

2. (4) What is the solution to our estimation problem? How many pairs of points do you think are needed to ensure that the solution is unique? Write down your answer and include brief explanation in your writeup.

3. (10) Projective transformation is very common in our real world. When you take pictures of the same object from different viewing angles, the objects in the pictures are projections of the real one. Based on this, we can make useful applications. For example, American football games are played on a rectangular field marked with end lines, side lines and goal lines which are 10 yards inward from each end line, as shown in the Fig. 2. In the NFL broadcasts, virtual yellow lines are drawn on top of real scenes to create a better view for TV audience (see more in How the NFL's magic yellow line works). In this problem, you are expected to implement this based on the homography estimation you derived in previous questions.

   To simplify the problem, you will work on two images instead of a video. `img1.jpg` and `img2.jpg` are two screenshots captured from a football game and are provided to you. We use the first image as a baseline (though it is not a good one). The yellow line for the marker 33 has been highlighted, as shown in Fig. 3. You need to select a couple of correspondences in two images first. Then you estimate the transformation matrix and draw the corresponding line in the second picture. Include the result image in your report.

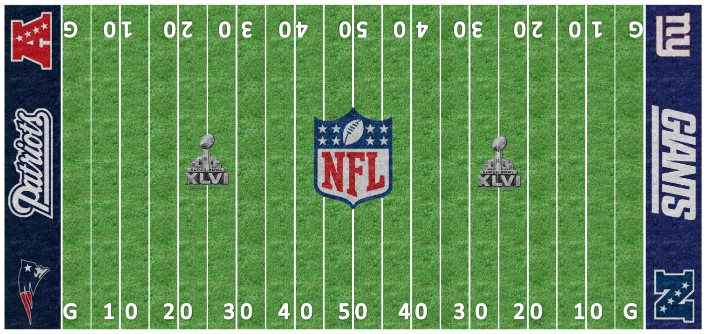

Figure 2: football field

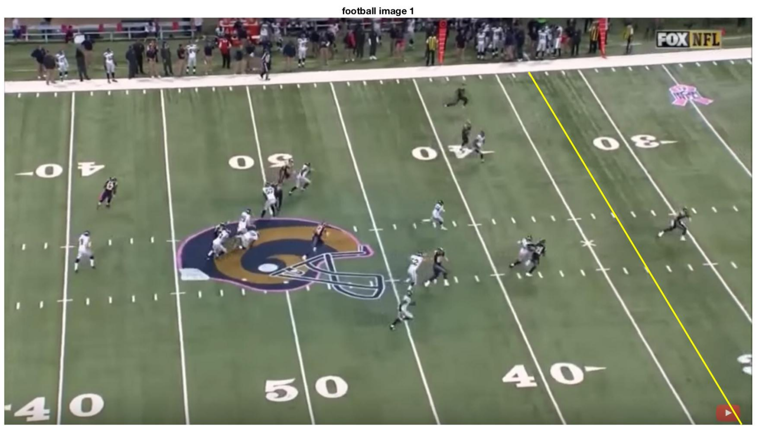

Figure 3: yellow line at marker 33

Problem 6 (20): Bayer Demosaicking

To digitalize a perfect color image, we overlay the Bayer pattern filter on top of the photo sensors to accept RGB separately. We could call this process mosaicking. Assume that we've got a digital image filtered by the Bayer pattern and want to demosaic it, i.e., reconstruct the original color image. One of the common ways to do this is to use bilinear interpolation. For a green pixel, two red neighbors can be interpolated to yield the red value of that pixel and two blue neighbors can be interpolated to yield the blue value. Similar way can be applied to red or blue pixels. The only difference is that for red or blue pixels you need to consider 4 neighbors instead of 2, as shown in the picture below. Note that for the pixels on the border, we only consider the neighboring pixels within the image domain.

You will have to write the function `demosaic` on your own. The image you will be working on is the Orion sculpture located in front of the University of Michigan Museum of Art (UMMA). The encoded image is a gray-scaled image in which the value of every pixel is the intensity of a single channel R, G or B. Fig. 5 shows an example of the filtered and the reconstructed image. By zooming in, you can see mosaicking effects in the gray-scaled image.

Include the reconstructed image in your report.

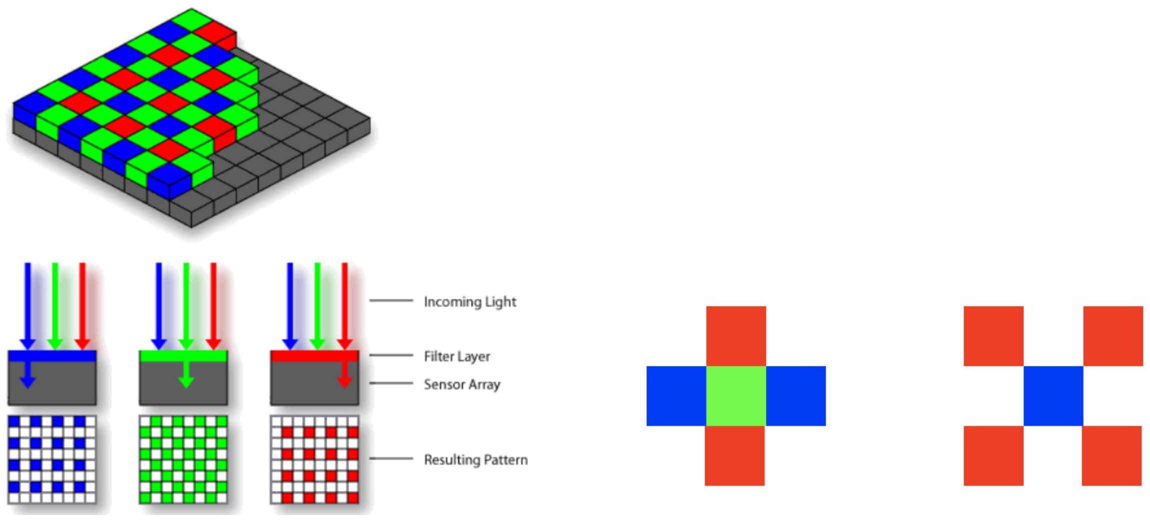

Figure 4: Bayer pattern (left) and neighboring pixels for interpolation (right)

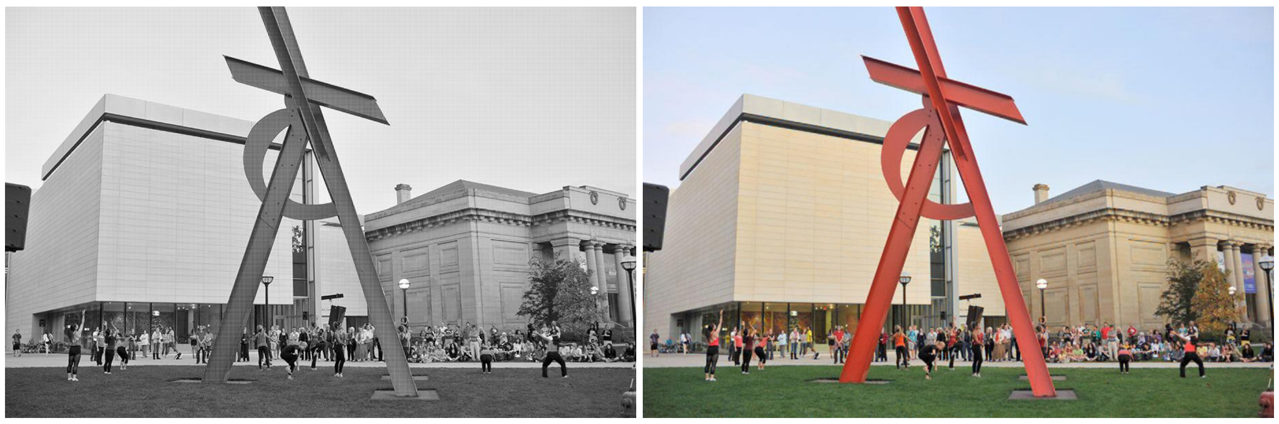

Figure 5: an example of Bayer encoded image (left) and reconstructed image (right)