CS231A: Computer Vision,
From 3D Reconstruction to Recognition Homework #1
(Winter 2023) Due: Friday, January 27th

Bryan Mbanefo

On to the problems!

1 Projective Geometry Problems [20 points]

a) Prove that parallel lines in the world reference system are still parallel in the camera reference system.

Answer:
Rigid transformations do not change the angle between lines, therefore if the lines are parallel in the world reference system, they will remain parallel in the camera reference system.

Consider 2 parallel lines k and l

By using equation of parallel lines

$$(k1 - k2) \times (l1 - l2) = 0$$

Where k1 and k2 are the slopes of the 2 lines and l1 and l2 are the y-intercepts.

Translation by vector p:

$$(k1 + p - k2 - p) \times (l1 + p - l2 - p) = (k1 - k2) \times (l1 - l2) = 0$$

Rotation by R:

$$\begin{aligned}(Rk1 - Rk2) \times (Rl1 - Rl2) &= R(k1 - k2) \times R(l1 - l2) \\ &= R\big((k1 - k2) \times (l1 - l2)\big) = 0\end{aligned}$$

Therefore lines are parallel even with rigid transformations.

b) Prove that under any affine transformation, the ratio of parallel line segments is invariant, but the ratio of non-parallel line segments is not invariant.

Answer:
Affine transformations preserve collinearity and the ratios of distances between points. So parallelism is preserved between 2 line segments and also the ratio of their lengths.

Consider 2 parallel line segments A and B:

$$K = \frac{A}{B}\ ; \quad where\ K\ is\ the\ ratio\ of\ line\ segments$$

With affine transformation new segments A' and B' will also be parallel therefore:

$$K' = \frac{A'}{B'} = \frac{A}{B} = K$$

Therefore the ratio of line segments is invariant under any affine transformation.

For non-parallel line segments A', B' will no longer be parallel and K' != K so non-parallel line segments are not invariant.

For example consider a 2-D plane, If we have an affine transformation that stretches by a factor of 3 in the y-direction. We see that a line segment from the origin to (1,0) remains the same, but a segment from the origin to (0,1) is stretched by a length of 3.

c) Consider a unit square pqrs in the world reference system where p, q, r, and s are points. Will the same square in the camera reference system always have unit area? Prove or provide a counterexample. Similarly, will the unit square be preserved or not under an affine transformation?

Answer:
Unit square has an area of 1 in the world reference system.

In the camera reference system this may or may not be the case depending on Intrinsic and Extrinsic parameters of the camera such as the focal length. The square may be affected by perspective distortion and show up more like a parallelogram in the camera reference system.

For affine transformations, the area will be preserved if the transformation matrix has a determinant of 1 since the affine transformation will preserve the ratio of parallel lines.

Given a transformation matrix T:

$$\| (Tq - Tp) \times (Ts - Tp) \| = (detT) \| (q - p) \times (s - p) \|$$
$$= 1 \| (q - p) \times (s - p) \| = 1$$

d) You have explored whether these three properties (rotational and translational invariance, and ratio of parallel line segments) hold for affine transformations. Do these properties hold under any projective transformation? Justify briefly in one or two sentences (no proof needed).

Answer:

No. Projective transformations maps lines to lines, points to points but does not preserve parallelism as parallel lines in the world reference system will intersect in the image at a vanishing point. Therefore area, ratio of line segments are not preserved.

2 Affine Camera Calibration (35 points):

a) Given correspondences for the calibrating grid, solve for the camera parameters using Eq. 2. Note that each measurement (xi , yi) ↔ (Xi , Yi , Zi) yields two linear equations for the 8 unknown camera parameters. Given N corner measurements, we have 2N equations and 8 unknowns. Using the given corner correspondences as inputs, complete the method compute camera matrix(). You will construct a linear system of equations and solve for the camera parameters to minimize the least-squares error. After doing so, you will return the 3 × 4 affine camera matrix composed of these computed camera parameters. **Explain your approach and include the camera matrix that you compute in the written report. [15 points for code + 5 for write-up]**

Answer:

Approach:

- By expanding out the original linear system, we can reformulate the question by placing all the affine camera matrix element unknowns into a single x vector (the A matrix and b vector will be known), then use least squares to find a best fit solution.
- The question defines the linear system matrix we need to solve for the affine camera
- Since the 3D and 2D points are known we can write out a linear system of equations in matrix form and solve for x. Our equation will be Ax = b where A and b are defined as the following matrices

$$A = \begin{bmatrix} X_1 & Y_1 & Z_1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & X_1 & Y_1 & Z_1 & 1 \\ X_2 & Y_2 & Z_2 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & X_2 & Y_2 & Z_2 & 1 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \end{bmatrix}$$

$$b = \begin{bmatrix} x_1 \\ y_1 \\ x_2 \\ y_2 \\ \dots \end{bmatrix}$$

- The x matrix is the list of unknowns that we can then solve using least squares.
- Computed camera matrix is as follows:

$$\begin{bmatrix} 5.31276507e-01 & -1.80886074e-02 & 1.20509667e-01 & 1.29720641e+02 \\ 4.84975447e-02 & 5.36366401e-01 & -1.02675222e-01 & 4.43879607e+01 \\ 0.00000000e+00 & 0.00000000e+00 & 0.00000000e+00 & 1.00000000e+00 \end{bmatrix}$$

b) Compute the RMS error for the camera matrix that you found in part (a). What are some possible sources of error when computing the error between the given corners 3 and calculated points using the camera matrix. [5 points for code + 5 points for write-up]

Answer:

RMS Error: 0.99383048327985

There are various sources of error that can occur such as:

- Improper camera calibration which will lead to the camera matrix inaccurately representing the intrinsic and extrinsic properties of the camera.
- Noise in the image can lead to inaccuracy of the detected corners.
- Image distortion can also lead to inaccuracies.

c) Could you calibrate the matrix with only one checkerboard image? Explain briefly in one or two sentences. [5 points]

Answer:

You can calibrate the matrix with one checkerboard image but it will not be enough to fully and accurately calibrate the camera. In order to accurately estimate the intrinsic and extrinsic properties of the camera, you need more images taken from different positions.

3 Single View Geometry (45 points)

a) See code
b) See code
c) Is it possible to compute the camera intrinsic matrix for any set of vanishing points? Similarly, is three vanishing points the minimum required to compute the intrinsic camera matrix?

Answer:

- Yes it is possible to compute the camera matrix for any set of vanishing points. It requires two sets of parallel lines and their corresponding vanishing points.
- Yes the minimum number of vanishing points required is 3. Each set of parallel lines in 3D space will have vanishing points. The intersections of these 3 vanishing points is always the image center and once that is known then computing the intrinsic matrix is possible. This is an approximate center and may not always be the real center of the image in real world scenarios but it is good enough to get an approximation for the intrinsic camera matrix.

d) The method used to obtain vanishing points is approximate and prone to noise. Discuss what possible sources of noise could be, and select a pair of points from the image that are a good example of this source of error:

Answer:

- A good source of error will be image distortion which makes the intersection of 3 vanishing points not always be the center of the image.
- Limited accuracy of the 3D world lines. The 3D lines from the world reference system may not be accurate leading to errors.

Vanishing points:

- Floor vanishing point and box vanishing point are the same [4545.256281407035, 50.85427135678392] which may not be an accurate representation in the 3D space.

e) Fill out the method compute angle between planes() and include a brief description of your solution and your computed angle in your report.

Answer:

- Using the vanishing point provided in the function, we need to compute the vanishing lines l_1 and l_2 that belong to the 2 planes.
- Then we compute the omega matrix from the given camera matrix and take the inverse.

- Using the computed parameters we can find the angle by solving the equation

$$\cos(\theta)\frac{l_1^T w^{-1} l_2}{\sqrt{l_1^T w^{-1} l_1}\sqrt{l_2^T w^{-1} l_2}}$$

- Ensure to convert from radians to degrees as numpy gives the answer in radians

  Angle between floor and box: $89.81938858899667^{\circ}$

f) Fill out the method compute rotation matrix between cameras() and submit a brief description of your approach and your results in the written report.

Answer:

- Normalize the vanishing points by applying the inverse of the intrinsic camera matrix and dividing by the norm.
- Calculate rotation matrix by taking the dot product of the transpose of the normalized vanishing points of the second camera with the inverse transpose of the normalized vanishing points of the first camera.
- This will give you the rotation matrix R

$$R = \begin{bmatrix} 1.05220283 & 0.02418414 & -0.09924512 \\ -0.06187726 & 1.0221037 & 0.0105692 \\ 0.06406076 & -0.03169809 & 0.91794226 \end{bmatrix}$$

Angle around z-axis (pointing out of camera): -1.316672 degrees
Angle around y-axis (pointing vertically): -6.170270 degrees
Angle around x-axis (pointing horizontally): -0.659675 degrees