Johns Hopkins University
Dr. Alejandro Martin Gomez
Irene Kim, An Chi Chen
Yihao Liu, Muhammad Hadi

EN.601.454/654
Augmented Reality, Fall 2022
Assignment 2
Due: September 27th (Thu) 2022, 23:59

Submission Guidelines:

1. There are usually two types of questions: written and programming.
2. For written:
   - Scan your written solution or compile it as a pdf file
   - Submit the pdf to https://www.gradescope.com/ under *written* solution assignment box (Entry Code: **4V24DD**)
3. For programming:
   - Write a readme file to describe how to compile and run your codes on which platform *Codes that do not run will not be graded!*
   - Save the readme and all your source codes (no binary) as a zip file.
   - Submit the archive to https://www.gradescope.com/ under *programming* solution assignment box (Entry Code: **4V24DD**)
4. Some questions may be marked as ***(UG Optional)*** . They are voluntary for undergraduate students. If answered, they will be graded and will count as bonus points; however, the maximum score can get from one assignment remains 100%. E.g., if you get 45/40 in total, you will get 100% instead of 112.5%. **Graduate students must answer all questions.**

Assignment 2: Transformations and Parameter Estimation

In this assignment, you will work through some basic mathematics that may be useful for the entire class, which includes linear algebra, calculus, geometry, optimization etc. You are asked to review the related topics and finish the questions below.

1 (Written) Matrices & Transformations [20 points]

A matrix $A \in \mathbb{R}^{m \times n}$ is in the form:

$$A = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{bmatrix} \text{ where } a_{ij} \in \mathbb{R}$$

A linear mapping or linear transformation from one space $\mathbb{R}^n$ to another space $\mathbb{R}^m$, denoted as $f : \mathbb{R}^n \to \mathbb{R}^m$, can be expressed by a matrix $A \in \mathbb{R}^{m \times n}$ by

$$f(x) = Ax \text{ where } x \in \mathbb{R}^n \text{ is linearly mapped/transformed to } f(x) \in \mathbb{R}^m \text{ by } A$$

Most often, we will need to solve for A given some output $y \in \mathbb{R}^m$ and some input $x \in \mathbb{R}^n$, which you may see it many time as a formulation of systems of linear equations $y = Ax$.

For example, in augmented reality, if we have a desired location in the real world to place our virtual content, we will need to perform registration/calibration to find out the required transformation. For now, to get familiar with the operations required to complete this task, review your textbook and solve the following problems:

1.1 Show that matrix multiplication is not commutative, i.e. find two matrices $A, B \in \mathbb{R}^{2\times 2}$ such that $AB \neq BA$. (2 points)

1.2 However, in a special case, it is commutative. Consider $A, B \in \mathbb{R}^{n\times n}$ with $A^T = A$ and $B^T = B$ (symmetric matrices), show that if AB is also symmetric, then $AB = BA$. (2 points)

1.3 The matrix multiplication can be expressed in a summation form: e.g. for $X = UV$, the element x_{ij} can be expressed as $x_{ij} = \sum_{k=1}^{m} u_{ik}v_{kj}$, where $X \in \mathbb{R}^{n\times n}, U \in \mathbb{R}^{n\times m}, V \in \mathbb{R}^{m\times n}$. By using the summation form, for matrices $A, B \in \mathbb{R}^{n\times m}$, and $C \in \mathbb{R}^{m\times k}$, show that $(A + B)C = AC + BC$. (2 points)

1.4 Compute the rank and determinant of the following matrices if it exists and decide if the matrix is invertible or not. If so, compute the inverse. (6 points)

(a) $\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$ (1 point)

(b) $\begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix}$ (1 points)

(c) $\begin{bmatrix} 3 & 1 & 3 \\ 9 & 4 & 9 \\ 15 & 7 & 15 \end{bmatrix}$ (2 points)

(d) $\begin{bmatrix} 132 & 165 & 198 \\ 348 & 435 & 522 \\ 564 & 705 & 846 \end{bmatrix}$ (2 points)

1.5 Express the following 3D rotations in matrix form (6 points)

(a) Rotate around x-axis by 30 degrees, $\mathbf{R}_x$. (2 point)

(b) Rotate around y-axis by 45 degrees, $\mathbf{R}_y$. (2 point)

(c) Rotate around z-axis by 60 degrees, $\mathbf{R}_z$. (2 point)

1.6 Use the above matrices to show that $\mathbf{R}_x\mathbf{R}_y\mathbf{R}_z \neq \mathbf{R}_z\mathbf{R}_y\mathbf{R}_x$. (2 points)

2 (Written) Homogeneous Coordinates [15 points]

Consider the action of a matrix $M \in \mathbb{R}^{n\times n}$ on a vector $x \in \mathbb{R}^n$, followed by a translation $t \in \mathbb{R}^n$,

$$x \mapsto Mx + t.$$

This action can be expressed in terms of matrix multiplication by encoding the mapping *homogeneous coordinates*:

$$(M,\, t) \equiv \begin{bmatrix} M & t \\ 0^T & 1 \end{bmatrix} \in \mathbb{R}^{(n+1)\times(n+1)}$$
$$x \equiv \begin{bmatrix} x \\ 1 \end{bmatrix} \in \mathbb{R}^{n+1},$$

such that

$$\begin{bmatrix} M & t \\ 0^T & 1 \end{bmatrix} \begin{bmatrix} x \\ 1 \end{bmatrix} = \begin{bmatrix} Mx + t \\ 1 \end{bmatrix}.$$

Here we've taken a point $x \in \mathbb{R}^n$ and lifted it to $\mathbb{R}^{n+1}$ by assigning it a 'depth' of 1 in $n+1$ dimensional space such that $x \mapsto [x, 1]^T$. Similarly, expressing points $p \in \mathbb{R}^{n+1}$ such that $p = [x, z]^T$, where $x \in \mathbb{R}^n$ and $z \in \mathbb{R}$, we can project them down to $\mathbb{R}^n$ via the mapping

$$\begin{bmatrix} x \\ z \end{bmatrix} \mapsto x/z \in \mathbb{R}^n,$$

as long as $z \neq 0$.

(a) For some $\alpha \in \mathbb{R}$ and $t \in \mathbb{R}^n$, express the mapping given by

$$x \mapsto \alpha x + t, \quad \forall x \in \mathbb{R}^n \tag{1}$$

in homogeneous coordinates (*i.e.* as a matrix in $\mathbb{R}^{(n+1)\times(n+1)}$). (2 points)

(b) Given a point

$$x = \begin{bmatrix} 10 \\ 20 \\ 30 \end{bmatrix} \in \mathbb{R}^3$$

and a transformation

$$T = \begin{bmatrix} 20 & 0 & 5 & 0 \\ 0 & 20 & 5 & 0 \\ 0 & 0 & -10 & -100 \\ 0 & 0 & -1 & 0 \end{bmatrix} \in \mathbb{R}^{4\times4},$$

express x in homogeneous coordinates, apply T, and project the result back into $\mathbb{R}^3$. (3 points)

(c) Let R be an $n-$dimensional rotation matrix. Compute the inverse of

$$\begin{bmatrix} R & t \\ 0^T & 1 \end{bmatrix} \in \mathbb{R}^{(n+1)\times(n+1)}, \quad t \in \mathbb{R}^n.$$

(4 points)

(d) Consider a 3D point $\vec{P} = \begin{bmatrix} 4 \\ 0 \\ -2 \end{bmatrix}$ which is observed by two cameras with projection matrices:

$$\mathbf{P}_1 = \begin{bmatrix} 1 & 0 & 0 & -2 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad \text{and} \quad \mathbf{P}_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

where the image plane is assumed to be located at $z = 1$. Compute the corresponding projected points $\vec{p}_1$ and $\vec{p}_2$ in the image planes of the first and second camera, respectively. [Remember the projected points are in $\mathbb{R}^2$.] (6 points)

3 (Written) Parameter Estimation [10 points + 5 points]

Multiple views of a a planar object, such as a checkerboard, are often used in camera calibration to robustly estimate a camera's associated projection matrix, which depends on its often unknown internal dimensions. Any two images of the same planar object at different orientations are related by a homography, which defines the transformation between them and is determined by the camera's internal parameters. Given multiple views of the same planar object, we can estimate a number of homographies - all dependent on the camera's parameters. With enough views, we can solve for a unique set of parameters corresponding to the camera's internal dimensions. In this question, we will go through the homography estimation pipeline with a toy example as illustrated in Figure 1.

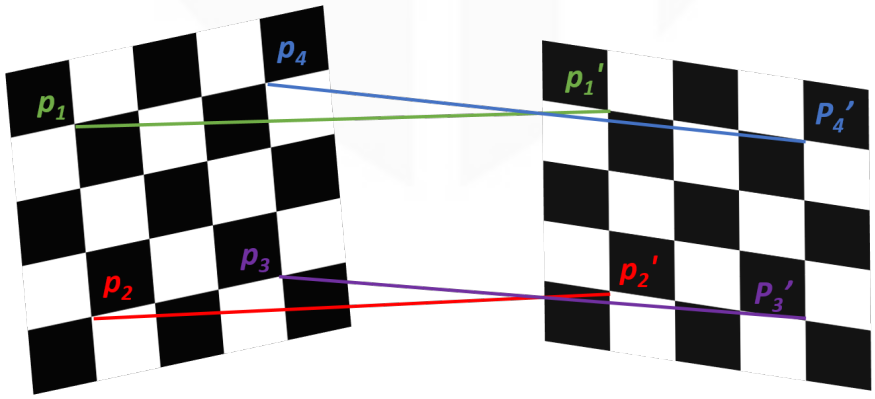

Figure 1: 4 point correspondence between two checkerboard images.

(a) What is the minimum number of point-to-point correspondence required for homography estimation? Why? (2 points)

(b) Let

$$\mathbf{H} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix}$$

be the homography between the two planar objects in Figure 1. Assume $p_1 = (0,0)^T$, $p_2 = (0,1)^T$, $p_3 = (1,1)^T$ and $p_4 = (1,0)^T$; and $p_1{}' = (1,1)^T$, $p_2{}' = (1,0)^T$, $p_3{}' = (2,0)^T$ and $p_4{}' = (2,1)^T$. Find $\mathbf{H}$ by fixing $h_{33} = 1$ and using the below equations. (4 points)

$$p_x{}' = \frac{h_{11}p_x + h_{12}p_y + h_{13}}{h_{31}p_x + h_{32}p_y + h_{33}} \tag{2}$$

$$p_y{}' = \frac{h_{21}p_x + h_{22}p_y + h_{23}}{h_{31}p_x + h_{32}p_y + h_{33}} \tag{3}$$

(c) ***(UG Optional)*** Why do equations (1) and (2) hold? (2 points)

(d) ***(UG Optional)*** Why can we fix $h_{33} = 1$? Is it always good to fix this for the estimation? Why or why not? What is the physical meaning when $h_{33} = 0$? (3 points)

(e) Four points can give us an unique solution, but it's unlikely the estimation will be robust due to noise and outliers. In general, all corner correspondences on the checkerboard are used for homography estimation, resulting in an overconstrained system. Suggest a suitable method for robust estimation in this paradigm, and explain your reasoning. (1 point)

(f) Will homography estimation work if we have four point-to-point correspondences between two views of a non-planar 3D shape? Why or why not? (2 points)

(g) Aside from camera calibration and image rectification, suggest another application for homography estimation.(1 point)

4 (Programming) Matrix Decomposition [10 points]

Matrix decomposition is a useful tool in solving systems of linear equations. In this question, use either Matlab or Python (with package numpy) to implement the following decomposition algorithms.

4.1 QR Decomposition (5 points)

Given a matrix $A \in \mathbb{R}^{m \times n}$, the reduced QR decomposition can be written as

$$A = QR = \begin{bmatrix} q_1 & q_2 & \cdots & q_n \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & \cdots & r_{1n} \\ 0 & r_{22} & \ddots & r_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & r_{nn} \end{bmatrix} \text{ where } Q \in \mathbb{R}^{m \times n} \text{ and } R \in \mathbb{R}^{n \times n}.$$

The columns of Q, q_i forms an orthonormal basis, which means $Q^TQ = I \in \mathbb{R}^{n\times n}$. The QR decomposition can be obtained by Gram-Schmid method, which is described as follows:

Denote the columns of A by $a_i \in \mathbb{R}^m$ and the projection of a vector v on another vector u by $P_u(v) = \frac{\langle v,u\rangle}{\langle u,u\rangle}u$. The orthonormal basis from the a_i can be obtained by

$$\begin{aligned}
u_1 &= a_1 & q_1 &= \frac{u_1}{||u_1||} \\
u_2 &= a_2 - P_{q_1}(a_2) & q_2 &= \frac{u_2}{||u_2||} \\
u_3 &= a_3 - P_{q_1}(a_3) - P_{q_2}(a_3) & q_3 &= \frac{u_3}{||u_3||} \\
&\vdots & & \\
u_n &= a_n - \sum_{i=1}^{n-1} P_{q_i}(a_n) & q_n &= \frac{u_n}{||u_n||}
\end{aligned}$$

Therefore,

$$a_i = u_i + \sum_{k=1}^{i-1}\langle a_i, q_k\rangle q_k = ||u_i||q_i + \sum_{k=1}^{i-1}\langle a_i, q_k\rangle q_k$$

This gives us:

$$A = \begin{bmatrix} q_1 & q_2 & \cdots & q_n \end{bmatrix} \begin{bmatrix} ||u_1|| & \langle a_2, q_1\rangle & \cdots & \langle a_n, q_1\rangle \\ 0 & ||u_2|| & \ddots & \langle a_n, q_2\rangle \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & ||u_n|| \end{bmatrix}$$

Implement this algorithm in the programming language and linear algebra package you choose, and compare your result with the QR decomposition provided by the package. Remember to write a readme to describe how to run your codes to check the results. (5 points)

4.2 LU Decomposition (5 points)

Given a non-singular matrix $A \in \mathbb{R}^{n\times n}$, it can be decomposed into lower L and upper U triangular matrices in $\mathbb{R}^{n\times n}$, i.e. $A = LU$. This is called LU decomposition, and it can be computed using a recursive algorithm. Assume

$$A = \begin{bmatrix} \alpha & u^T \\ v & A^* \end{bmatrix} = \begin{bmatrix} 1 & \\ w & L^* \end{bmatrix} \begin{bmatrix} \beta & x^T \\ & U^* \end{bmatrix} = LU$$

where $\alpha, \beta \in \mathbb{R}, u, v, w, x \in \mathbb{R}^{n-1}$, and $A^*, L^*, U^* \in \mathbb{R}^{(n-1)\times(n-1)}$. By comparison, we have

$$\begin{aligned}
\alpha &= \beta \\
u^T &= x^T \\
v &= \beta w \\
A^* &= wx^T + L^*U^*
\end{aligned}$$

Hence, we can find the LU decomposition by computing β, x, w and repeating it with the new matrix $A^* - wx^T \in \mathbb{R}^{(n-1)\times(n-1)}$ until A^* is a scalar.

Implement this algorithm in the programming language and linear algebra package you choose, and compare your result with the LU decomposition provided by the package. Remember to write a readme to describe how to run your codes to check the results. (5 points)

5 (Programming) Direct Linear Transformation [15 points]

(a) Given the 2D to 2D point correspondences of *problem 3.b*, determine the 2D homography matrix H such that $x_i^{'} = Hx_i$.

- For each correspondence $x_i \leftrightarrow x_i^{'}$ compute A_i (2 points).
- Assemble n 2-by-9 matrices A_i into a single $2n$-by-9 matrix A (1 point).
- Solve for h using $A^T A$ (5 points).
- Determine H from h (1 point).

(b) Using the matrix A computed in the previous step:

- Obtain SVD of A to find h (5 points).
- Determine H from h (1 point).

Implement this algorithm in the programming language and linear algebra package you choose. Remember to write a readme to describe how to run your codes to check the results.