# Programming Assignment #1; Due Date:10/01/2024

Submit your documented Python code via CANVAS. Also submit a **TYPED REPORT** that contains, concise discussion on your procedure/algorithm used (including any math derivations you did to develop the algorithm), any observations from the results and any reasoning necessary to explain your approach and results. The report should contain, results in the form of images that are arranged to show the input image and the output images. Try to minimize your page usage by displaying at least 3 images per row on a page. All your image displays must have an appropriate and relevant caption.

1. (25 points)

   The following is an example of an affine map

   $$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 1 \\ 1 & 3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 4 \\ 0 \end{bmatrix}$$

   Show that this affine map is a composition of a shear followed by a rotation (by an angle), then by a magnification (scaling) and then a translation. You are required to explicitly show the numerical values in the transformation matrices. Develop python code to apply each of these transformations, and then their composition to, (i) an image of a black square on a white background and show the resulting images. **Show the input square image followed by the output deformed square images after application of each of the transformations** namely, shear, rotation, magnification and translation respectively and then finally show the output image after applying the composition of the transformations. (ii) A picture of your face taken using your cellphone camera. Show the output image after the application of each of the transformations and the composition. *Caution: You will have to use some kind of an interpolation technique, e.g., bilinear interpolation, to get a meaningful image when applying transformations to an image, else, your output image will contain holes. Do not submit results without interpolation.*

2. **Software camera:** (50 points) A software camera is defined by the locations of the following four pixels in the image plane:

   - The pixel coordinates (in homogeneous representation) of the image point that corresponds to the world point at infinity along the world X direction: $(5, 100, 1)^T$.
   - Same as above except that now we have the pixel coordinates for the image point that corresponds to the world point at infinity along the world Y direction: $(400, 300, 1)^T$.
   - Same as above except that now we have the pixel coordinates for the image point that corresponds to the world point at infinity along the world Z direction: $(500, 490, 1)^T$
   - The pixel coordinates of the image point corresponding to the world origin: $(20, 20, 5)^T$.

   (a) If this is a finite projective camera, where is the camera center located in the world coordinate system?

   (b) Finally, using this software camera, take any one of your "old images" and reproject it through this camera to see what you get. Obviously, before you can carry out this reprojection, you will have to place your old 2-D image somewhere in front of this software camera. You can place your old image at a distance of two focal lengths from the camera center, with the principal axis passing through the center of your old image. Assume that the old image is placed parallel to the sensor plane (see figure 1 for illustration).

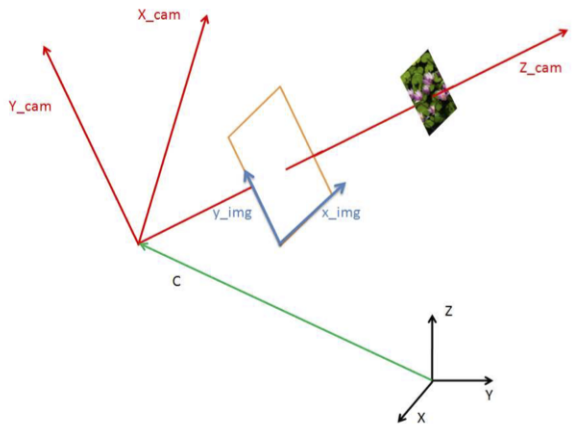


Figure 1: Camera model illustration for placement of the "old image."

- Clearly identify the steps you have taken to solve the problem in your own words.
- Your grade depends on the completeness and clarity of your work as well as the result.

**Hint:** use the fact that the homogeneous representation of the image point corresponding to a world point at infinity along the world X, Y and Z directions are the column vectors $\vec{p1}, \vec{p2}$ and $\vec{p3}$ of the camera matrix P. In addition, the $4^{th}$ column vector $\vec{p4}$ is the homogeneous representation of the image of world origin. This will give you the $(3, 4)$ camera projection matrix $P$.

3. (25 pts.) **Image Blending (25 pts):** In this problem, you will write code for blending two images (one of yourself and another of your friend) using Gaussian and Laplacian pyramids of the two images (you may read section 3.5.5 in the text book). You will first write code to set up the Gaussian and Laplacian pyramids, say with 3 or 4 levels starting with an image of size $(256, 256)$ or $(512, 512)$. Blending images basically involves reconstruction of either or both of the images using a blended Laplacian of the two. Blended Laplacian is simply a convex combination of the laplacians of the two given images at each level of the Laplacian pyramid. The weights in the convex combination are determined using a known mask function. Construct two different masks and achieve two different blendings of yourself and your friend. Be creative in defining your masks (see figure 3.43 in the textbook).