**CSE252a – Computer Vision – Assignment 1**
Instructor: Ben Ochoa
Due: Thursday, October 23, 11:59 PM

## Instructions:

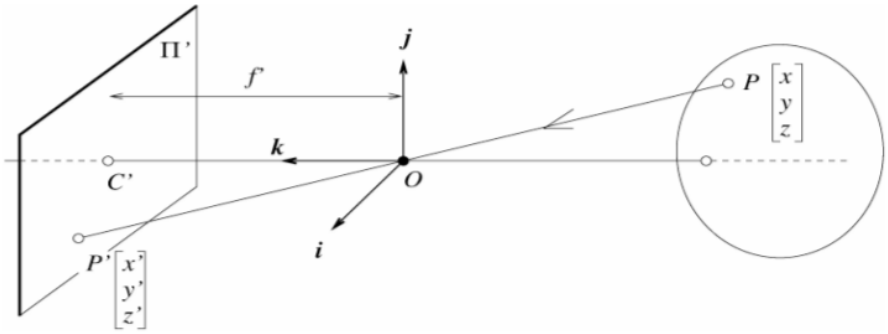
- Submit your assignment electronically by email to iskwak+252a@cs.ucsd.edu with the subject line *CSE252 Assignment 1*. The email should have one file attached. Name this file: CSE_252_hw01_lastname_studentid.zip. The contents of the file should be:
  1. A pdf file with your writeup. This should have all code attached in the appendix. Name this file: CSE_252_hw01_lastname.pdf.
  2. All of your source code in a folder called code.
- No physical hand-in for this assignment.
- In general, code does not have to be efficient. Focus on clarity, correctness and function here, and we can worry about speed in another course.

# Perspective Projection [2 pts]

Consider a perspective projection where a point

$$P = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

is projected onto an image plane $\Pi'$ represented by $k = f' > 0$ as shown in the following figure.




The first, second, and third coordinate axes are denoted by i, j, and k respectively. Consider the projection of a ray in the world coordinate system

$$Q = \begin{bmatrix} 4 \\ 2 \\ 0 \end{bmatrix} + t \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}$$

## Part A

What is the projection of the point when $t = -1$?

### Part B

What is the projected point's coordinates when $t = -\infty$? Note, to get full credit you cannot take the limit as $t$ goes to $-\infty$.

## Thin Lens Equation [2 pts]

An illuminated arrow forms a real inverted image of itself at a distance w = 50cm, measured along the optic axis of a convex thin lens (see Figure 1). The image is just half the size of the object.

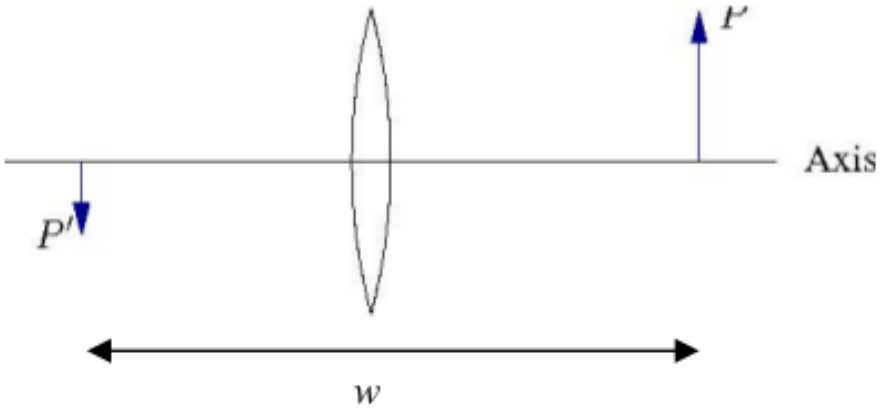



Figure 1: Problem 3 Setup

A. How far from the object must the lens be placed?

B. What is the focal length of the lens?

C. Suppose the arrow moves $x$ cm to the right while the lens and image plane remain fixed. This will result in an out of focus image; what is the radius of the corresponding blur circle formed from the tip of the arrow on the image plane assuming the diameter of the lens is $d$?

## Affine Projection [2pts]

Consider an affine camera and a line in 3D space. Consider three points (A, B, and C) on that line, and the image of those three points (a, b and c). Now consider the distance between a and b and the distance between a and c. Show that the ratio of the distance is independent of the direction of the line.

## Image formation and rigid body transformations [10 points]

This problem has written and coding components.

In this problem we will practice rigid body transformations and image formations through the projective and affine camera model. The goal will be to 'photograph' the following four points given by ${}^{A}P_1 = (-1, -0.5, 2)^T$, ${}^{A}P_2 = (1, -0.5, 2)^T$, ${}^{A}P_3 = (1, 0.5, 2)^T$, ${}^{A}P_4 = (-1, 0.5, 2)^T$ in world

coordinates. To do this we will need two matrices. Recall, first, the following formula for rigid body transformation

$$^{B}P = {}^{B}_{A}R \; {}^{A}P + {}^{B}O_{A} \tag{1}$$

where $^{B}P$ is the point coordinate in the target ($B$) coordinate system, $^{A}P$ is the point coordinates in the source ($A$) coordinate system, ${}^{B}_{A}R$ is the rotation matrix from $A$ to $B$, and $^{B}O_{A}$ is the origin of coordinate system $A$ expressed in the $B$ coordinates. The rotation and translation can be combined into a single $4 \times 4$ *extrinsic parameter* matrix, $Pe$, so that $^{B}P = Pe * {}^{A}P$. Once transformed, the points can be photographed using the *intrinsic camera* matrix, $Pi$ which is a $3 \times 4$ Once these are found, the image of a point, $^{A}P$, can be calculated as $Pi * Pe * {}^{A}P$. We will consider four different settings of focal length, viewing angles and camera positions below.

For each part calculate or provide the following:

- the extrinsic transformation matrix.
- intrinsic camera matrix under the perspective camera assumption.
- intrinsic camera matrix under the affine camera assumption. In particular, around what point do you do the taylor series expansion?
- The actual points around which you did the taylor series expansion for the affine camera models.
- How did you arrive at these points?
- The affine projection is an approximation. Plot what happens when using a poor choice for the taylor series expansion, for example try $(0, 5, 2)$.
- Calculate the image of the four vertices and plot using the supplied plotsquare.m/plotsquare.py function (see e.g. output in figure 2).

1. [**No rigid body transformation**]. Focal length = 1. The optical axis of the camera is aligned with the z-axis.
2. [**Translation**] $^{B}O_{A} = (0, 0, 1)^{T}$. The optical axis of the camera is aligned with the z-axis.
3. [**Translation and rotation**]. Focal length = 1. ${}^{B}_{A}R$ encodes a 20 degrees around the z-axis and then 40 degrees around the y-axis. $^{B}O_{A} = (0, 0, 1)^{T}$.
4. [**Translation and rotation, long distance**]. Focal length = 5. ${}^{B}_{A}R$ encodes a 20 degrees around the z-axis and then 40 degrees around the y-axis. $^{B}O_{A} = (0, 0, 10)^{T}$.

Note: we will not use a full intrinsic camera matrix (e.g. that maps centimeters to pixels, and defines the coordinates of the center of the image), but only parameterize this with $f$, the focal length. In other words: the only parameter in the intrinsic camera matrix under the perspective assumption is $f$, and the only ones under the affine assumption are: $f, x_0, y_0, z_0$, where $x_0, y_0, z_0$ is the center of the taylor series expansion.

# Image warping and merging [10 pts]

This is a programming assignment, which should be done in Matlab or python. All data necessary for this assignment is available on the course web page.

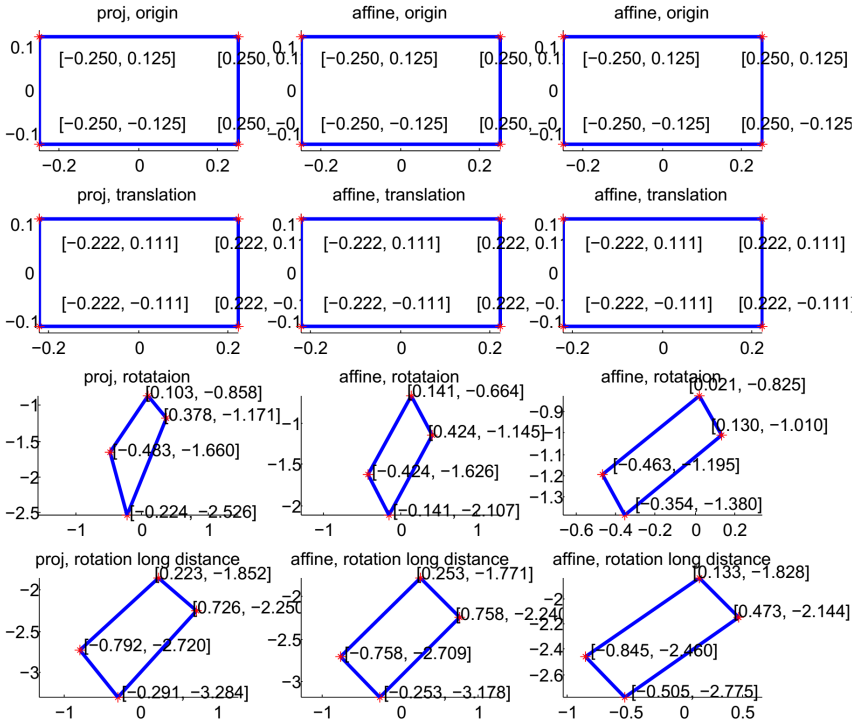



Figure 2: Example output for image formation problem. Note: the angles and offsets used to generate these plots are different from those in the problem statement, it's just to illustrate how to report your results.

## Introduction

In this assignment, we consider a vision application in which components of the scene are replaced by components from another image scene.

Optical character recognition, OCR, is one computer vision's more successful applications. However OCR can struggle with text that is distorted by imaging conditions. In order to help improve OCR, some people will "rectify" the image. An example is shown in Fig 3. Reading signs from Google street view images can also benefit from techniques such as this.

This kind of rectification can be accomplished by finding a mapping from points on one plane to points another plane. In Fig 3, $P_1, P_2, P_3, P_4$ were mapped to $(0,0), (1,0), (1,1), (0,1)$. To solve this section of the homework, you will begin by deriving the transformation that maps one image onto another in the planar scene case. Then you will write a program that implements this transformation and uses it to rectify ads from a stadium.

To begin, we consider the projection of planes in images. Imagine two cameras $C_1$ and $C_2$ looking at a plane $\pi$ in the world. Consider a point $P$ on the plane $\pi$ and its projections $p = (u1, v1, 1)^\top$ in image 1 and $q = (u2, v2, 1)^\top$ in image 2.

**Fact 1** *There exists a unique (up to scale) 3x3 matrix $H$ such that, for any point $P$:*

$$q \equiv Hp$$

*($\equiv$ denotes equality in homogeneous coordinates, meaning that the left and right hand side are proportional.) Note that $H$ only depends on the plane and the projection matrices of the two cameras.*

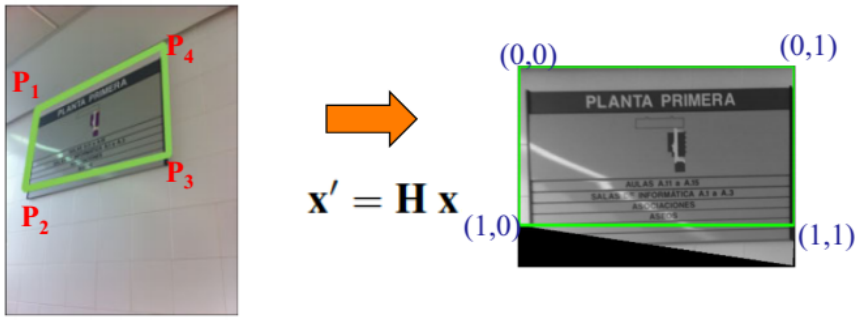



Figure 3: Input image (left) and target (right) for image mapping problem

The interesting thing about this result is that by using H we can compute the image of P that would be seen in camera C2 from the image of the point in camera C1 without knowing its three-dimensional location. Such an H is a projective transformation of the plane, also referred to as a homography.

## Problem definition

Write files computeH.m and warp.m that can be used in the following skeleton code. warp takes as inputs the original image, corners of an ad in the image, and finally, $H$, the homography. Note that the homography should map points from the gallery image to the sign image, that way you will avoid problems with aliasing and sub-sampling effects.

```
I1 = imread('stadium.jpg');
% get points from the image
figure(10)
imshow(I1)
% select points on the image, preferably the corners of an ad.
points = ginput(4);

figure(1)
subplot(1,2,1);
imshow(I1);

new_points = [...]; % choose your own set of points to warp your ad too

H = computeH(points, new_points);

% warp will return just the ad rectified. This will require cropping the ad out of the
% stadium image.
warped_img = warp(I1, points, H);

subplot(1,2,2);
imshow(warped_img);
```

## Report

Run the skeleton code and include the output image in your report. Attach source code for computeH.m and warp.m in the appendix.

## For fun (no credit or hand-in)

Try running an off the shelf OCR engine on the original picture and your rectified image!

Good luck!