# Homework #1

Name: Yunfeng Wei
CWID: 10394963
E-mail: ywei14@stevens.edu

## Problem #1:

(1) Figure (A) is taken with a perspective camera. Figure (B) is taken with an orthographic camera.

(2) The camera projection matrix for a perspective camera: $\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -\frac{1}{d} & 0 \end{bmatrix}$, d is the distance from the COP(Center of Projection) to the object.

The camera projection matrix for an orthographic camera: $\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$.

If we set d = -z, then the perspective projection will be $\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{1}{z} & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} => (x, y)$, and the orthographic matrix is still $\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} => (x, y)$. Obviously, in this situation, the perspective projection is equal to the orthographic projection.

(3) The following figure identifies a set of lines in picture A that are intersecting in a varnishing point with a finite coordinates.

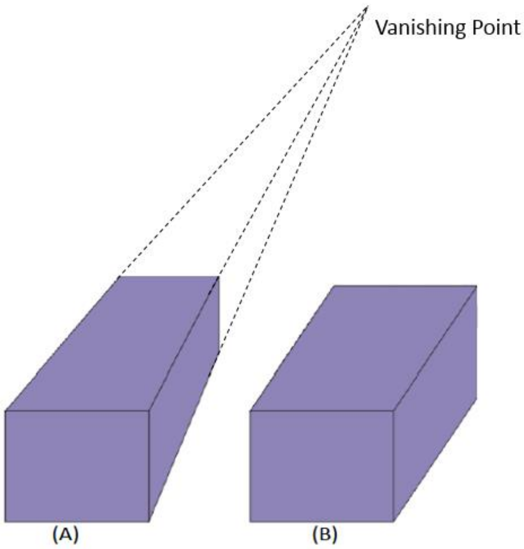


**Figure 1**: Two pictures taken by two different cameras.

## Problem #2:

(1) y1 = [1  1  2],  y2 = [2  3  1].

So in the Euclidean coordinates: y1 = $2\begin{bmatrix}\frac{1}{2} & \frac{1}{2} & 1\end{bmatrix} => \begin{bmatrix}\frac{1}{2} & \frac{1}{2}\end{bmatrix}$,

y2 = $1[2/1 \quad 3/1 \quad 1] => [2 \quad 3]$.

(2) l1 = $[2 \quad 3 \quad 3]$, l2 = $[1 \quad -1 \quad 2]$.

So the intersection of two lines is

**i. Solution 1:**

Homogenous coordinates:

$l1 \times l2 = [2 \quad 3 \quad 3] \times [1 \quad -1 \quad 2] =$

$[3 \times 2 - 3 \times -1 \quad 3 \times 1 - 2 \times 2 \quad 2 \times -1 - 3 \times 1] = [9 \quad -1 \quad -5]$

So the intersection point is $[9 \quad -1 \quad -5]$

**ii. Solution 2:**

Euclidean coordinates:

l1: 2x + 3y + 3 = 0

l2: x – y + 2 = 0

$\begin{cases} 2x + 3y + 3 = 0 \\ x - y + 2 = 0 \end{cases} => \begin{cases} x = -\frac{9}{5}, \\ y = \frac{1}{5} \end{cases};$

So the intersection point is $\begin{bmatrix}-\frac{9}{5} & \frac{1}{5}\end{bmatrix}$

## Problem 3:

(1) x1 = $[1 \quad 2 \quad 3 \quad 4] = 4[1/4 \quad 2/4 \quad 3/4 \quad 4/4] => \begin{bmatrix}\frac{1}{4} & \frac{1}{2} & \frac{3}{4}\end{bmatrix}$;

x2 = $[4 \quad 3 \quad 2 \quad 1] = 1\,[4/1 \quad 3/1 \quad 2/1 \quad 1/1] => [4 \quad 3 \quad 2]$;

So x1 represents $\begin{bmatrix}\frac{1}{4} & \frac{1}{2} & \frac{3}{4}\end{bmatrix}$, x2 represents $[4 \quad 3 \quad 2]$

(2) p1 = $[1 \quad -1 \quad 2 \quad -1]$ => x – y + 2z – 1 = 0

The normal vector is $[1 \quad -1 \quad 2]$ => $\begin{bmatrix}\frac{\sqrt{6}}{6} & -\frac{\sqrt{6}}{6} & \frac{\sqrt{6}}{3}\end{bmatrix}$

So the normal vector is $\begin{bmatrix}\frac{\sqrt{6}}{6} & -\frac{\sqrt{6}}{6} & \frac{\sqrt{6}}{3}\end{bmatrix}$

The distance to the origin is D = $\frac{|-1|}{\sqrt{1^2 + -1^2 + 2^2}} = \frac{\sqrt{6}}{6}$

## Problem 4:

(1) To build the regularized linear regression from the 256-dimensional data to the discrete labels, first assume the linear regression function:

$$f(x) = w^T x + b$$

Input x: a 256-dimensional vector.

Output y: response value

So formulate into the following problem:

$$min_{w \in R^d, b \in R} \sum_{i=1}^{n} ||f(x_i) - y_i||^2 + \theta ||w||^2$$

$$=> min_{w \in R^d} \sum_{i=1}^{n} ||w^T \alpha_i - \beta_i||^2 + \theta||w||^2$$

$$where\ b = -\frac{1}{n}\sum_{i=1}^{n}(w^T x_i - y_i)$$

Centered input and output: $\alpha_i, \beta_i, i = 1, \dots, n$

So we can decide the data matrix:

$$X = \begin{bmatrix} x_1{}^T - \frac{1}{n}\sum_{i=1}^{n} x_i{}^T \\ x_2{}^T - \frac{1}{n}\sum_{i=1}^{n} x_i{}^T \\ \dots \\ x_n{}^T - \frac{1}{n}\sum_{i=1}^{n} x_i{}^T \end{bmatrix} \in R^{n \times d}$$

Write the response vector

$$y = \begin{bmatrix} y_1 - \frac{1}{n}\sum_{i=1}^{n} y_i \\ y_2 - \frac{1}{n}\sum_{i=1}^{n} y_i \\ \dots \\ y_n - \frac{1}{n}\sum_{i=1}^{n} y_i \end{bmatrix} \in R^n$$

$So\ we\ can\ solve\ the\ linear\ regression\ problem\ as$:

$$min_{w \in R^d} \left|\left|Xw - y\right|\right|^2 + \theta\left|\left|w\right|\right|^2$$

Which has a unique global optimum $w^* = (X^T X + \theta I)^{-1} X^T y$

Because $\theta\left|\left|w\right|\right|^2$ is small, we can assume the linear regression problem as:

$$min_{w \in R^d} \left|\left|Xw - y\right|\right|^2$$

In this case, there are n = 4649 data samples, each $x_i$ has 256 dimensions.

So we can solve the w is a $256 \times 1$ value, b is 4.4917. The linear regression function is

$$f(x) = w^T x + 4.4917$$

Finally, use this function to predict the labels of the 4649 samples in the test dataset.

The answer is stored in the txt file called “testY.txt”.

**Following is the Matlab code:**

```
% Load the sample data
newData1 = load('linear_regression.mat', '-mat');

% Create new variables in the base workspace from those fields.
vars = fieldnames(newData1);
for i = 1:length(vars)
    assignin('base', vars{i}, newData1.(vars{i}));
end
% Load ended
```

```
% because the original trainY is 1 x 4649, transpose it to be a 4649 x 1 vector
trainY = trainY';

aveX = zeros(1, size(trainX, 2)); % Define a 1 * 256 vector called aveX
% Store the average value of X into the aveX
for i = 1 : size(trainX, 1) % sum all 4649 values
	aveX = aveX + trainX(i,:);
end
aveX = aveX / size(trainX, 1); % divide 4649 to calculate the average value

X = zeros(size(trainX, 1), size(trainX, 2)); % Define the data matrix
% Calculate the data matrix
for i = 1 : size(trainX, 1) % calculate each row
	X(i,:) = trainX(i,:) - aveX;
end

aveY = zeros(1, size(trainY, 2)); % Define a variable called aveY
% Store the average value of y into aveY
for i = 1:size(trainY, 1)
	aveY = aveY + trainY(i,:);
end
aveY = aveY / size(trainY, 1);

Y = zeros(size(trainY, 1), size(trainY, 2)); % Define the response vector called Y
% Calculate the response vector
for i = 1:size(trainY, 1)
	Y(i,:) = trainY(i,:) - aveY;
end

% Calculate the unique global optimum
% Store the value into w
w = inv(X' * X) * X' * Y;

% Calculate b and store it into b
b = zeros(1, 1);
for i = 1:size(trainY, 1)
	b = b + w' * trainX(i,:)' - trainY(i,:);
end
b = -b/size(trainY, 1);

%Test the data
testAns = zeros(size(testX, 1), 1);
for i = 1:size(testX, 1)
```

```
    testAns(i, 1) = w' * (testX(i, :))' + b;
end

testRes = uint8(testAns);
% For any value which is greater than 10, set to 10; any value which is
% smaller than 0, set to 0;
for i = 1:size(testX, 1)
    if testRes(i,1) >= 10
        testRes(i,1) = 10;
    elseif testRes(i,1) <= 0
        testRes(i,1) = 0;
    end
end

% Write the testY into 'testY.txt'
dlmwrite('testY.txt', (testRes), ';');
```