Math 16-811 - HW1 - resubmit

Xiang Zhi Tan

October 12, 2015

2.b

Let $L = I, A' = A$

L	A´	Operations
$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ \frac{1}{2} & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 4 & 8 & 0 & 0 \\ 0 & -4 & -2 & 0 \\ 0 & 4 & -1 & 0 \\ 0 & -2 & 0 & 2 \\ 0 & 0 & 2 & -1 \end{pmatrix}$	$R_2 = R_2 - \frac{1}{2}R_1$
$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ \frac{1}{2} & 1 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 4 & 8 & 0 & 0 \\ 0 & -4 & -2 & 0 \\ 0 & 0 & -3 & 0 \\ 0 & 0 & 1 & 2 \\ 0 & 0 & 2 & -1 \end{pmatrix}$	$R_3 = R_3 + R_2$ $R_4 = R_4 - \frac{1}{2}R_2$
$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ \frac{1}{2} & 1 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 \\ 0 & \frac{1}{2} & -\frac{1}{3} & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 4 & 8 & 0 & 0 \\ 0 & -4 & -2 & 0 \\ 0 & 0 & -3 & 0 \\ 0 & 0 & 0 & 2 \\ 0 & 0 & 2 & -1 \end{pmatrix}$	$R_4 = R_4 + \frac{1}{3}R_3$
$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ \frac{1}{2} & 1 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 \\ 0 & \frac{1}{2} & -\frac{1}{3} & 1 & 0 \\ 0 & 0 & -\frac{2}{3} & -\frac{1}{2} & 1 \end{pmatrix}$	$\begin{pmatrix} 4 & 8 & 0 & 0 \\ 0 & -4 & -2 & 0 \\ 0 & 0 & -3 & 0 \\ 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 0 \end{pmatrix}$	$R_5 = R_5 + \frac{2}{3}R_3$ $R_5 = R_5 + \frac{1}{2}R_4$

Therefore:

$$P = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

$$L = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ \frac{1}{2} & 1 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 \\ 0 & \frac{1}{2} & -\frac{1}{3} & 1 & 0 \\ 0 & 0 & -\frac{2}{3} & -\frac{1}{2} & 1 \end{pmatrix}$$

$$D = \begin{pmatrix} 4 & 0 & 0 & 0 \\ 0 & -4 & 0 & 0 \\ 0 & 0 & -3 & 0 \\ 0 & 0 & 0 & 2 \end{pmatrix}$$

$$U = \begin{pmatrix} 1 & 2 & 0 & 0 \\ 0 & 1 & \frac{1}{2} & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

SVD Decomposition

$$u = \begin{pmatrix} -0.9005 & -0.0155 & 0.3562 & -0.1617 & -0.1895 \\ -0.0891 & -0.7311 & 0.1847 & 0.5290 & 0.3789 \\ -0.3787 & 0.0340 & -0.7352 & -0.1479 & 0.5413 \\ 0.1942 & -0.3642 & 0.2836 & -0.8023 & 0.3248 \\ 0.0078 & 0.5757 & 0.4670 & 0.1686 & 0.6496 \end{pmatrix}$$

$$s = \begin{pmatrix} 9.8844 & 0 & 0 & 0 \\ 0 & 3.3506 & 0 & 0 \\ 0 & 0 & 2.3134 & 0 \\ 0 & 0 & 0 & 1.9288 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

$$v = \begin{pmatrix} -0.3824 & -0.4549 & 0.7755 & 0.2132 \\ -0.9214 & 0.2210 & -0.2847 & -0.1456 \\ 0.0579 & 0.7699 & 0.5619 & -0.2970 \\ 0.0385 & -0.3892 & 0.0434 & -0.9193 \end{pmatrix}$$

1 Question 5

I received help from Devin Schwab for this question. The solution was based on the work by Arun(1988) titled Least-Squares Fitting of Two 3-D Point Sets(link: http://162.105.204.96/teachers/yaoy/Fall2011/arun.pdf).

This problem asked to find the rotation and translation function that transforms p_i to q_i for where $i = 1...n$. The equation of function can be written out

as following:
$$q_i = Rp_i + t$$

Where R is the rotation matrix and t is the translation vector.

Since we assume that our measurement of q_i contains error or noise, we are trying to find the rotation matrix and translation vector with the least error. The error function of the equation is as following:

$$E = \sum_{i=1}^{N} ||q_i - (Rp_i + T)||^2$$

Each point are first rotated than moved by the translation matrix. Because rotation are done with a fixed centroid, the centroid of all the points should be the same and did not change by the rotation. Therefore, we could calculate the centroid for both p_i and q_i and move them to the origin point on the plan. This will make our new points only be affected by rotation instead of both rotation and translation. Here's the equation for the operations. First, we calculate the centroid

$$p_c = \frac{1}{n} \sum_{i=1}^{n} p$$

$$q_c = \frac{1}{n} \sum_{i=1}^{n} q$$

we then subtract the centroid from each points to create a set of points:

$$p'_i = p_i - p_c$$
$$q'_i = q_i - q_c$$

Our error function is now updated to the following:

$$E = \sum_{i=1}^{N} ||(q'_i - Rp'_i)||^2$$

By the theorem in which $u\dot{u} = ||u||^2$ where $||u||$ is the length of the vector and

identity of $AB = (B^T A^T)^T$. We can rewrite the equation as:

$$E = \sum_{i=1}^{N} (q_i' - Rp_i') \bullet (q_i' - Rp_i')$$

$$= \sum_{i=1}^{N} (q_i' - Rp_i')^T (q_i' - Rp_i')$$

$$= \sum_{i=1}^{N} (q_i')^T (q_i') - (q_i')^T (Rp_i') - (Rp_i')^T (q_i') + (Rp_i')^T (Rp_i')$$

$$= \sum_{i=1}^{N} (q_i')^T (q_i') - (q_i')^T (Rp_i') - (q_i')^T (Rq_i') + (R)^T (p_i')^T (Rp_i')$$

$$= \sum_{i=1}^{N} (q_i')^T (q_i') + (p_i')^T (p_i') - 2(q_i')^T (Rp_i')$$

Because we are trying to minimize the error, the only term that is important would be the negative term of $2(q_i')^T (Rp_i')$. where we hope to maximize it. This leads to this equation

$$F = \sum_{i=1}^{N} (q_i')^T (Rp_i')$$

Because $(q_i')^T$ is a 1 x 3 vector and RP_i' is a 3 x 1 vector. F is the inner product of $(q_i')^T$ and RP_i'. Because the inner product is the trace of the outer product of the 2 vectors. The equation is rewritten as:

$$F = Trace(\sum_{i=1}^{N} Rp_i' q_i'^T)$$

We will define a 3x3 matrix called H that is $\sum_{i=1}^{N} p_i' q' i^T$. This will lead the equation to be:

$$F = Trace(RH)$$

We can decompose H using SVD into $U\Sigma V^T$. Let $R = BX$ and VU^T. Therefore, $XH = VU^T U\Sigma V^T$ which is equal to $V\Sigma V^T$. Because B is orthonormal(Rotation matrix are orthonormal), the following lemma can be use:

Lemma: For any positive definite matrix AA^T and any orthonormal matrix B,

$$Trace(AA^T) \geq Trace(BAA^T)$$

Which means $Trace(XH) \geq Trace(BXH)$. H is fixed because it is generated from the points given, therefore only by maximizing X can we maximize F.From the previous equation and substituting $H = U\Sigma V^T$, we can find R as $R = X = VU^T$

After finding R, we can find the translation matrix by applying rotation to the centroids and find the difference between then.

$$T = p' - Rp$$

```matlab
function [R,T] = pointSolver(i,o)
%UNTITLED Summary of this function goes here
%   Detailed explanation goes here

curMat = eye(3);

%find the center of the problem
cI = [0,0,0]';
cO = [0,0,0]';
for k=1:length(i)
    cI = cI + (1/length(i)) * i{k};
    cO = cO + (1/length(o)) * o{k};
end
p = [];
q = [];
for k=1:length(i)
    p = [p i{k} - cI];
    q = [q o{k} - cO];
end
%create the H matrix
H = q * p';
[U,S,V] = svd(H);
R = U * V';
T = cO - R*cI
end
```