Programming Assignments 1 & 2 (circle one)
600.445/645 Fall 2016

Score Sheet (hand in with report)

Name 1	
Email	
Other contact information (optional)	
Name 2	
Email	
Other contact information (optional)	
Signature (required)	I (we) have followed the rules in completing this assignment ______________________________ ______________________________

Grade Factor		
Program (40)		
Design and overall program structure	20	
Reusability and modularity	10	
Clarity of documentation and programming	10	
Results (20)		
Correctness and completeness	20	
Report (40)		
Description of formulation and algorithmic approach	15	
Overview of program	10	
Discussion of validation approach	5	
Discussion of results	10	
TOTAL	100	

CIS PA 1

MENGZE XU, LIUJIANG YAN

Summary

This report is for programming assignment 1 basic transformations and pivot calibration, containing following parts,

* ∗ Mathematical Approach and Algorithm
* ∗ Programming Structure
* ∗ Validation and Results
* ∗ Discussion
* ∗ Summary of Unknown Data

The folder contains following subfolders and Matlab files,

* ∗ PA1 - Essential Matlab functions
  - – registration.m: 3D point set to 3D point set registration
  - – c_expected_value.m: compute C expected value by composition rule
  - – em_pivot_calibration.m: determine EM pivot position relative to EM tracker
  - – opt_pivot_calibration.m: determine OPT pivot position relative to EM tracker
* ∗ PA12-StudentData - Data for PA1
* ∗ parse - Matlab functions for parsing data files
  - – parseCALBODY.m
  - – parseCALREADINGS.m
  - – parseEMPIVOT.m
  - – parseOPTPIVOT.m
  - – parseOUTPUT_1.m
* ∗ pa1output - Output data derived by PA1
* ∗ pa1driver.m - driver script for PA1 and leads to pa1output files
* ∗ pa1validation.m - Matlab script for validation and error analysis

Mathematical Approach and Algorithm

(a) 3D point set to 3D point set registration

The goal of 3D point set to 3D point set registration is to find the best estimation of the mapping from one to another.

$$F_{AB}P_B = P_A$$

Here we rewrite the expression in Homogeneous representation,

$$\begin{bmatrix} R & p \\ 0 & 1 \end{bmatrix} \begin{bmatrix} P_B \\ 1 \end{bmatrix} = \begin{bmatrix} P_A \\ 1 \end{bmatrix}$$

In order to perform least square, we need to rewrite the expression into form $At = b$, where t refers to a column of the elements of transformation F.

$$\begin{bmatrix} B_{1x} & B_{1y} & B_{1z} & 1 & ... & ... \\ ... & B_{1x} & B_{1y} & B_{1z} & 1 & ... \\ ... & ... & B_{1x} & B_{1y} & B_{1z} & 1 \\ ... & & & & & \end{bmatrix} \begin{bmatrix} R_{11} \\ R_{12} \\ R_{13} \\ p_1 \\ R_{21} \\ R_{22} \\ R_{23} \\ p_2 \\ R_{31} \\ R_{32} \\ R_{33} \\ p_3 \end{bmatrix} = \begin{bmatrix} A_{1x} \\ A_{1y} \\ A_{1z} \\ ... \end{bmatrix}$$

$$A_{3n\times 12}t_{12\times 1} = b_{3n\times 1}$$

The Matlab function solves registration problem for $Fp = P$ is shown following.

```
function Fform = registration(P, p)
    % 3D point set to 3D point set registration
    % Fa = A -> solve linear system At=b
    A = [];
    b = [];
    for i=1:length(p)
        subA = [p(i,1) p(i,2) p(i,3) 1 0 0 0 0 0 0 0 0;
                0 0 0 0 p(i,1) p(i,2) p(i,3) 1 0 0 0 0;
                0 0 0 0 0 0 0 0 p(i,1) p(i,2) p(i,3) 1];
        subb = [P(i,1); P(i,2); P(i,3)];
        A = [A; subA];
        b = [b; subb];
    end
    % solve the least square
    h = linsolve(A,b);
    Ffrom = reshape(h,4,3);
    Fform = [Ffrom'; 0 0 0 1];
end
```

(b) Pivot Calibration

The goal of pivot Calibration is to determine the pivot position, given a set of markers'

position with different orientation. The general idea is to define a local coordinate in the probe and then, for each orientation, find the transformation between the position relative to tracker base frame and the position relative to local coordinate. By rewriting the expression, the pivot value can be solved by performing least square.
Firstly, we need to define a local coordinate. Here we use the midpoint of the first data frame as the local coordinate's origin,

$$G_0 = \frac{1}{N_G} \sum G_{1i}$$

and then translate each data frame to the local coordinate,

$$g_i = G_i - G_0$$

For each data frame, we perform registration method derived above to find the transformation of the data frame from local coordinate to EM track base coordinate. Take kth frame for example, the transformation should be,

$$G_{k,i} = F_k g_{k,i}$$

So that in this frame, the pivot value can be presented as

$$P_{pivot} = R_k t_{pivot} + p_k$$

In order to perform least square, we write the expression as,

$$\begin{bmatrix} R_k & -I \end{bmatrix} \begin{bmatrix} t_{pivot} \\ P_{pivot} \end{bmatrix} = -p_k$$

Combining all data frame, we get

$$\begin{bmatrix} R_1 & -I \\ R_2 & -I \\ \cdots & \end{bmatrix} \begin{bmatrix} t_{pivot} \\ P_{pivot} \end{bmatrix} = \begin{bmatrix} -p_1 \\ -p_2 \\ \cdots \end{bmatrix}$$

To solve the linear system derived above, we could get pivot value, where t_{pivot} stands for the pivot value relative to the local coordinate and P_{pivot} stands for the pivot value relative to the EM tracker base.
The matlab script will be shown in EM pivot calibration part and Optical pivot calibration part, since the procedure is different between these two process so that a general function is not feasible.
(c) Compute expected value $C_i^{expected}$ for C_i
To compute the expected value $C_i^{expected}$ for C_i we need to perform composition rule as below,

$$C_i^{expected} = F_{AD}^{-1} F_{AC} c_i$$

In order to solve the expression above, we need to perform registration for transformation from optical tracker to EM tracker and from optical tracker to calibration object, where the transformation stands for,

$$F_{AC} a_i = A_i$$

$$F_{AD} d_i = D_i$$

The Matlab function to compute $C_i^{expected}$ for C_i is shown here,

```
function c_expected = c_expected_value(D, d, A, a, c)
    % perform 3D point set to 3D point set registration
    F_AC = registration(A, a);
    F_AD = registration(D, d);
    % perform composition rule
    F_DC = F_AD\F_AC;
    c(:,4) = 1;
    % write c in homogeneous representation and compute C expected
    c_expected = F_DC*c';
    c_expected = c_expected(1:3,:)';
end
```

(d) EM pivot calibration
The pivot calibration for EM tracker is almost the same as the general pivot calibration process shown above.
Also, we record the first frame relative to the local coordinate for future use (PA2: compute the locations of fiducial points with respect to EM tracker base).

```
function [pivot, g_1st_frame] = em_pivot_calibration(Ng, Nframes, M)

    % use the first frame calibration data to define a local coordinate
    G_init = M(1:Ng,:);
    midpoint = sum(G_init)/length(G_init);

    % the position of EM markers in the first frame to local cooridnate
    for i=1:Ng
        g_1st_frame(i,:) = G_init(i,:)-midpoint;
    end

    % predefine for linear system solving
    A=[];
    b=[];

    % loop the data frames
    for i=1:Nframes
        % get the calibration data of each frame
        startpoint = (i-1)*Ng + 1;
        endpoint = i*Ng;
        G = M(startpoint:endpoint,:);

        % transform to the local coordinate and get g
        for j=1:3
            g(:,j) = G(:,j) - midpoint(:,j);
        end

        % compute the transformation between EM tracker frame and local
        % coordinate for each data frame
        F = registration(G, g);
        R = F(1:3,1:3);
        p = F(1:3,4);

        % construct the matrix for least square problem
        subA = [R, -eye(3)];
        A = [A; subA];
        b = [b; -p];
    end

    % solve the linear system and get the pivot value
    pivot = linsolve(A,b);

end
```

(e) Optical pivot calibration
In order to get the pivot value of optical probe with respect to EM tracker, we need to transform the position of the optical markers with respect to optical tracker base to EM tracker base, slightly different from the process presented above as general pivot calibration. In each frame, we need to transform the position of H to EM tracker base,

$$F_{DA}H_{opt} = H_{EM}$$

Where we could perform registration for F_{DA},

$$F_{DA}D = d$$

So before we derive the local coordinate, and before each time we compute the transformation from local coordinate to EM tracker base, we need to transform the position of optical markers' position to EM tracker with the derived transformation matrix F_{DA} for each data frame.

```
function pivot = opt_pivot_calibration(d, ND, NH, Nframes, M)
    % use the first frame calibration data to define the local coordinate
    % first get the transformation FD by 3D point registration
    D = M(1:ND,:);
    FD = registration(D,d);

    % get the first frame calibration data
    % and transform the data with respect to EM tracker frame
    H_init = M(ND+1:ND+NH,:);
    H_init(:,4) = 1;
    H_EM_init = FD\H_init';
    H_EM_init = H_EM_init';

    % get the midpoint as the origin of the local coordinate
    midpoint = sum(H_EM_init)/length(H_EM_init);

    % predefine for linear system solving
    A=[];
    b=[];

    % loop the data frames
    for i = 1:Nframes

        % read the coordinate D and H respcetively
        D_start = (i-1)*(ND+NH)+1;
        D_end = (i-1)*(ND+NH)+ND;
        H_start = (i-1)*(ND+NH)+ND+1;
        H_end = i*(ND+NH);

        D = M(D_start:D_end,:);
        H = M(H_start:H_end,:);

        % compute the transformation for each data frame
        FD = registration(D,d);

        % transform H with respect to EM tracker frame
        H(:,4) = 1;
        H_EM = (FD\H')';
        H_EM = H_EM(:,1:3);

        % transform to the local coordinate and get h
        for j=1:3
            h_EM(:,j) = H_EM(:,j) - midpoint(j);
        end

        % compute the transformation between EM tracker frame and local
        % coordinate for each data frame
        F = registration(H_EM, h_EM);
        R = F(1:3,1:3);
        p = F(1:3,4);

        % construct the matrix for least square problem
        subA = [R,-eye(3)];
        A = [A; subA];
        b = [b; -p];
    end

    % solve the linear system and get the pivot value
    pivot = linsolve(A,b);
end
```

Programming Structure

(a) Phase Functions

Before we go through the structure of the programming structure, first of all, we explain how we phase the data file and get what we want.

Take calbody for example, we return the number of the markers or data frame, as well as the raw data matrix following.

Other functions to parse other data files are almost in the same structure with difference in the return.

```
function [Nd, Na, Nc, M] = parseCALBODY(filepath)
    FID = fopen(filepath);
    file = fgetl(FID);
    % get the first line
    header = textscan(file,'%f%f%f%s','delimiter',',');

    % extract the numerical data
    Nd = header{1,1};
    Na = header{1,2};
    Nc = header{1,3};
    % get the raw data
    M = csvread(filepath, 1, 0);
end
```

(b) Programming Structure
The main script for programming assignment 1 is **pa1driver.m**.
The first part of it is to add some essential paths and clear the workspace.

```
% programming assignment 1 driver
clc
clear
format compact

addpath('PA1');
addpath('PA2');
addpath('PA12-StudentData');
addpath('parse');
```

Then, we loop each debug or unknown data files. We specify the type of files we are reading.

```
for char = 'a':'k'
    % specify the file type
    if ismember(char, 'a':'g')
        type = 'debug';
    elseif ismember(char, 'h':'k')
        type = 'unknown';
    end
% the main functions are running inside here
end
```

We read the position of optical markers in EM tracker base relative to EM tracker base stored as d, the position of EM markers in calibration object relative to calibration object base as c and the position of optical markers in calibration object relative to calibration object base as a.

```
% read positions of EM tracker and Optical trakcer
% that describes the calibration object
[Nd, Na, Nc, M] = parseCALBODY...
    (strcat('pa1-',type,'-',char,'-calbody.txt'));
d = M(1:Nd,:);
```

```
a = M(Nd+1:Nd+Na,:);
c = M(Nd+Na+1:end,:);
```

Then we read the position of optical markers in EM tracker base relative to optical tracker base as D, the position of optical markers in calibration object relative to optical tracker base as A and the position of EM trackers in calibration object relative to EM tracker base as C.
With given data above, we perform the function shown above to compute the expected value $C_i^{expected}$ for C_i for each data frame.

```
% compute the expected value C_expected for C
[ND, NA, NC, NFrames, M] = parseCALREADINGS...
    (strcat('pa1-',type,'-',char,'-calreadings.txt'));
C_e = [];
for i=1:NFrames
    start = (i-1)*(ND+NA+NC)+1;
    D = M(start:start+ND-1,:);
    A = M(start+ND:start+ND+NA-1,:);
    C = M(start+ND+NA:start+ND+NA+NC-1,:);
    C_expected = c_expected_value(D, d, A, a, c);
    C_e = [C_e; C_expected];
end
```

Perform pivot calibration for EM probe, return the pivot value of EM probe with respect to EM tracker base.

```
% pivot calibration for EM tracker
[Ng, Nframes, M] = parseEMPIVOT...
    (strcat('pa1-',type,'-',char,'-empivot.txt'));
EMpivot = em_pivot_calibration(Ng, Nframes, M);
```

Perform pivot calibration fro optical probe, return the pivot value of optical probe with respect to EM tracker base.

```
% pivot calibration for optical tracker
[ND, NH, Nframes, M] = parseOPTPIVOT...
    (strcat('pa1-',type,'-',char,'-optpivot.txt'));
OPTpivot = opt_pivot_calibration(d, ND, NH, Nframes, M);
```

The last part is to output the results in certain form.

```
% output the result
output = zeros(3 + NC*NFrames,3);
% number of EM markers on calibration object
output(1,1) = NC;
% number of data frames
output(1,2) = NFrames;
% Estimated post position with EM probe pivot calibration
output(2,:) = EMpivot(4:6)';
```

```
% Estimated post position with optical probe pivot calibration
output(3,:) = OPTpivot(4:6)';
% expected value for C
output(4:end,:) = C_e;

% define output file name
output_filename = strcat('pa1output/pa1-',type,'-',char,'-output1.txt');
csvwrite(output_filename, output);
```

Validation and Results

With the output result data and given debug output data, we are able to perform some error analysis.
Here, first of all, we present the structure of our validation Matlab script.

```
clc;
clear;
format compact;
% add essential file path
addpath('PA1');
addpath('PA2');
addpath('PA12-StudentData');
addpath('parse');

% record the error between computed and output data
EM_pivot_error = [];
OPT_pivot_error = [];
C_expected_error = [];

count = 0;
for char = 'a':'g'
    count = count + 1;

    original_filename = strcat('pa1-',type,'-',char,'-output1.txt');
    computed_filename = strcat('pa1output/pa1-',type,'-',char,'-output1.txt');

    % get the data
    m_original = csvread(original_filename, 1, 0);
    m_computed = csvread(computed_filename, 1, 0);

    % get the error of each part of data
    EM_pivot_error(count,:) = m_original(1,:)-m_computed(1,:);
    OPT_pivot_error(count,:) = m_original(2,:)-m_computed(2,:);
    C_expected_error(:,:,count) = m_original(3:end,:)-m_computed(3:end,:);
end

% compute the norm of the error term
% use max() or mean() command to get the statistical data
EM_pivot_error_norm = sum(abs(EM_pivot_error).^2,2).^(1/2);
OPT_pivot_error_norm = sum(abs(OPT_pivot_error).^2,2).^(1/2);
C_expected_error_norm = sum(abs(C_expected_error).^2,2).^(1/2);
```

For the EM probe pivot calibration part, the error between the computed results and the given output data are displayed below.

EM	x	y	z	$\|\|\Delta r\|\|$
a	0	0	0	0
b	0.01	-0.01	0	0.01
c	0.01	-0.41	0.13	0.43
d	0	0	-0.01	0.01
e	-1.14	-0.41	2.48	2.76
f	1.52	0.5	-1.22	2.01
g	0.37	-1.32	0.46	1.44

For the Optical probe pivot calibration part, the error between the computed results and the given output data are displayed below.

opt	x	y	z	$\|\|\Delta r\|\|$
a	-0.01	0.01	0	0.01
b	0	-0.01	0	0.01
c	0.01	0	0.01	0.01
d	0	0	0.01	0.01
e	0	0.01	0.01	0.01
f	0.01	0.01	0	0.01
g	0	0	-0.01	0.01

For the expected value $C_i^{expected}$ for C_i part, we present the mean and maximum error term here.

mean	x	y	z	$\|\|\Delta r\|\|$
a	0	0	0	0
b	-0.03	0.03	0.03	0.5
c	0.03	0.03	0.05	0.4
d	0	0	0	0
e	0.11	0.01	-0.26	1.7
f	-0.05	-0.05	0.05	1.6
g	-0.11	0.03	-0.03	1.5

max	x	y	z	$\|\|\Delta r\|\|$
a	0.01	0.01	0.01	0.01
b	0.5	0.49	0.5	0.75
c	0.62	0.72	0.85	0.92
d	0.03	0.01	0.03	0.03
e	3.39	2.72	2.31	4.1
f	3.28	2.84	2.54	4.5
g	3.48	2.23	2.54	3.5

Discussion

(a) registration

In this programming assignment, to solve for transformation matrix,

$$F_{AB}P_B = P_A$$

we rewrite the expression and use least square in following form.

$$\begin{bmatrix} B_{1x} & B_{1y} & B_{1z} & 1 & ... & ... \\ ... & B_{1x} & B_{1y} & B_{1z} & 1 & ... \\ ... & ... & B_{1x} & B_{1y} & B_{1z} & 1 \\ ... \end{bmatrix} \begin{bmatrix} R_{11} \\ R_{12} \\ R_{13} \\ p_1 \\ R_{21} \\ R_{22} \\ R_{23} \\ p_2 \\ R_{31} \\ R_{32} \\ R_{33} \\ p_3 \end{bmatrix} = \begin{bmatrix} A_{1x} \\ A_{1y} \\ A_{1z} \\ ... \end{bmatrix}$$

$$A_{3n\times 12}t_{12\times 1} = b_{3n\times 1}$$

The accuracy of the registration process will be dependent on the residual of our least square method. Here we use the linsolve(A,b) function for the least square solution, with its default residual criteria.
Also, more pairs of data points we have, more accurate the registration process will be.
(b) the expected value $C_i^{expected}$ for C_i
As shown above, we apply composition rule to compute $C_i^{expected}$

$$C_i^{expected} = F_{AD}^{-1}F_{AC}c_i$$

where,

$$F_{AC}a_i = A_i$$
$$F_{AD}d_i = D_i$$

Since both F_{AC} and F_{AD} are dependent of the optical tracker base, we need to consider the jiggle term, which should be a random term influencing the translation term. Here we name them as Δp_{AC} and Δp_{AD}. Then we correct the expression for $C_i^{expected}$ as,

$$\begin{aligned}(C_i^{expected})^* &= R_{AD}^T(R_{AC}c_i + p_{AC} + \Delta p_{AC} - p_{AD} - \Delta p_{AD}) \\ &= C_i^{expected} + R_{AD}^T(\Delta p_{AC} - \Delta p_{AD})\end{aligned}$$

Since the jiggle term is the same for any simultaneous measurement,

$$\Delta p_{AC} = \Delta p_{AD}$$

then,

$$(C_i^{expected})^* = C_i^{expected}$$

The jiggle term has no influence on the computation for the expected value $C_i^{expected}$ for C_i.
From the result data we can see, in file d, where there is no EM tracker noise or EM tracker distortion but only optical tracker base jiggle, the error term is almost zero, which verifies our mathematical reasoning.
However, from the file b and file c, we could have a sight about the influence about EM tracker distortion and noise.
(c) EM tracker base pivot calibration
We need to consider the noise term Δp_n and distortion term ΔF_d, so that

$$G_k = R_k\Delta R_d g_k + \Delta p_n + \Delta p_d$$

which can not be cancelled during pivot calibration. Such results are consistent with the error term derived in the verification process. The case which only have noise shows a deviation less than 1 mm and the case with distortion has a error about several millimeter.
(d) optical tracker base pivot calibration
Here as how we computed the expected value, we need to consider the optical tracker jiggle term as Δp, so each registration should be rewritten as,

$$H_k = R_k \Delta h_k + p_k + \Delta p$$

and then transform to EM tracker base,

$$H_{k,EM} = R_{AD}^T (R_k h_k + p_k + \Delta p - p_{AD} - \Delta p) = R_{AD}^T (R_k h_k + p_k - p_{AD})$$

The jiggle term is cancelled during the composition rule, so that has no influence on pivot calibration. Also from the result between our computed data and the debug output files, the error term is near to zero in any case, so the mathematical approach as well as the algorithmic process for this problem can be verified.

Summary of Unknown Data

EM Pivot Calibration

EM	x	y	z
h	185.39	230.85	218.73
i	189.95	204	210.53
j	193.12	193.86	199.09
k	201.44	186.97	201.18

Optical Pivot Calibration

OPT	x	y	z
h	394.39	390.28	201.3
i	395.27	404.87	207.25
j	390.08	397.45	195.17
k	398.18	409.93	194.19

Expected value for C_i

Due to the huge amount of data, please refer to the result output files for unknown data.