



Programming Assignments 1 & 2 601.455 and 601/655 Fall 2023

Please also indicate which section(s) you are in (one of each is OK)

Score Sheet

Name 1	Keerthana Thammana
Email	lthamma1@jhu.edu
Other contact information (optional)	
Name 2	Kiana Bronder
Email	kbronde1@jhu.edu
Other contact information (optional)	
Signature (required)	<p>I (we) have followed the rules in completing this assignment</p> <p>We both are in 601.455</p> <div style="text-align: center;">   </div>

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	

1 Overview of Mathematical Approach

We developed a math package for 3D points, rotations, and frame transformations. We also implemented Arun's registration method and pivot calibration, as covered in class.

2 Algorithms

2.1 Cartesian Math

Using a generic 3D point class from Github (cited in the code), we expanded on it to make it function as part of a frame. Each frame contains a 3D rotation matrix and 3D point, combining as follows:

$$\mathbf{F} = \begin{bmatrix} \mathbf{R} & \vec{p} \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & p_x \\ r_{21} & r_{22} & r_{23} & p_y \\ r_{31} & r_{32} & r_{33} & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The inverse of a frame was calculated according to the class slides and with the knowledge that $R^{-1} = R^T$ as:

$$\mathbf{F}^{-1} = \begin{bmatrix} \mathbf{R}^{-1} & -\mathbf{R}^{-1} \cdot \vec{p} \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R}^T & -\mathbf{R}^T \cdot \vec{p} \\ 0 & 1 \end{bmatrix}$$

Frame multiplication was implemented as:

$$\mathbf{F}_A \cdot \mathbf{F}_B = \begin{bmatrix} \mathbf{R}_A & \vec{p}_A \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{R}_B & \vec{p}_B \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R}_A \cdot \mathbf{R}_B & \mathbf{R}_A \cdot \vec{p}_B + \vec{p}_A \\ 0 & 1 \end{bmatrix}$$

Point set to point set registration was implemented as:

$$\vec{v}_A = \mathbf{F}_{AB} \cdot \vec{v}_B = \mathbf{R}_{AB} \cdot \vec{v}_B + \vec{p}_{AB}$$

Where v_A is some point v with respect to frame A , and v_B is that same point with respect to frame B . \mathbf{F}_{AB} is defined in Written Homework 1 as $\mathbf{F}_A^{-1} \cdot \mathbf{F}_B$ and implemented as such. Even if there are intermediary frames (e.g. \mathbf{F}_C between \mathbf{F}_A and \mathbf{F}_B), the above equation remains the same since it would be:

$$\vec{v}_A = \mathbf{F}_{AC} \cdot \mathbf{F}_{CB} \cdot \vec{v}_B = \mathbf{F}_A^{-1} \cdot \mathbf{F}_C \cdot \mathbf{F}_C^{-1} \cdot \mathbf{F}_B \cdot \vec{v}_B = \mathbf{F}_A^{-1} \cdot \mathbf{I} \cdot \mathbf{F}_B \cdot \vec{v}_B = \mathbf{F}_A^{-1} \cdot \mathbf{F}_B \cdot \vec{v}_B = \mathbf{F}_{AB} \cdot \vec{v}_B$$

2.2 Arun's Registration Method

We opted to implement a closed form solution for registration rather than an iterative method to have a quicker method. We used Arun's registration, which we describe below.

$$\bar{a} = \frac{1}{N} \sum_{i=1}^N \vec{a}_i \quad \tilde{a}_i = \vec{a}_i - \bar{a}$$

$$\bar{b} = \frac{1}{N} \sum_{i=1}^N \vec{b}_i \quad \tilde{b}_i = \vec{b}_i - \bar{b}$$

$$\mathbf{H} = \sum_i \begin{bmatrix} \tilde{a}_{i,x} \tilde{b}_{i,x} & \tilde{a}_{i,x} \tilde{b}_{i,y} & \tilde{a}_{i,x} \tilde{b}_{i,z} \\ \tilde{a}_{i,y} \tilde{b}_{i,x} & \tilde{a}_{i,y} \tilde{b}_{i,y} & \tilde{a}_{i,y} \tilde{b}_{i,z} \\ \tilde{a}_{i,z} \tilde{b}_{i,x} & \tilde{a}_{i,z} \tilde{b}_{i,y} & \tilde{a}_{i,z} \tilde{b}_{i,z} \end{bmatrix}$$

We then computed the singular value decomposition to get $\mathbf{H} = \mathbf{U}\mathbf{S}\mathbf{V}^t$, where the rotation matrix $\mathbf{R} = \mathbf{V}\mathbf{U}^t$. The translation was calculated using $T = \bar{b} - \mathbf{R}\bar{a}$. If the determinant of the rotation matrix is -1, the last column of \mathbf{V} was multiplied by -1 to make sure that the resulting rotation matrix has a determinant of 1. A determinant of -1 could be caused by having a very small rotation.

2.3 Pivot Calibration

For pivot calibration for the EM tracking data, we set up a least squares problem with 2 unknowns: p_{tip} and p_{dimple} , where p_{tip} is the translation of the tip from the probe, and p_{dimple} is the location of the pivot point as measured by the EM tracker.

We used the first frame of pivot calibration data to define a local probe coordinate system \vec{g}_j , and then translate the observations relative to this midpoint for every frame.

$$\vec{G}_0 = \frac{1}{N_G} \sum \vec{G}_j$$

$$\vec{g}_j = \vec{G}_j - \vec{G}_0$$

For each frame k of pivot data, we computed a transformation $F_G[k]$ using Arun's registration method between \vec{g}_j and \vec{G}_j , such that $\vec{G}_j = F_G[k]\vec{g}_j$. Thus, $p_{dimple} = F_G[k]p_{tip}$.

The pivot point's location p_{dimple} in EM tracker coordinates is $p_{dimple} = \mathbf{R}_i p_{tip} + t_i$, which can be rearranged to get:

$$\begin{bmatrix} \mathbf{R}_i & -I \end{bmatrix} \begin{bmatrix} p_{tip} \\ p_{dimple} \end{bmatrix} = -t_i$$

Combining data from all frames gives us:

$$\begin{bmatrix} \mathbf{R}_1 & -I \\ \mathbf{R}_2 & -I \\ \vdots & \vdots \\ \mathbf{R}_N & -I \end{bmatrix} \begin{bmatrix} p_{tip} \\ p_{dimple} \end{bmatrix} = \begin{bmatrix} -t_1 \\ -t_2 \\ \vdots \\ -t_N \end{bmatrix} \quad \leftarrow \mathbf{A}x = b$$

At first, we tried solving this equation with regular least squares, however, the error in our generated values was quite high. Instead, we solved this using singular value decomposition least squares, where

$$\mathbf{A}x = b \rightarrow \mathbf{U} \begin{bmatrix} \mathbf{S} \\ \mathbf{0} \end{bmatrix} \mathbf{V}^T x = \mathbf{b}$$

For pivot calibration of the optical tracking data, we used the same procedure as above, but calculated the transformation F_D before the pivot calibration for each frame to transform the optical tracker position into EM tracker coordinates.

3 Code Structure

We created files for the different parts of the project.

GenerateOutput.py is the main file that should be run on input data to get an output file.

This files includes functions to read the unknown dataset files and generate an output file, include pivot probe positions and expected C_i s.

FileIO.py includes functions to read the calbody, calreadings, empivot, optpivot, and output1 files.

For each type of file, the exact dimensions of the output arrays are specified in the python file, so that users can look at the description and understand how the array is formatted.

Point3d.py and Frame.py include classes to create 3d point objects, do transformations, save frame data, and other Cartesian math functions.

Each Point3d object stores the name of the frame it resides in in addition to its x, y, z coordinates. The base code for this class was taken off Github (cited in code) though we added several additional features (e.g., frame variable and error() function) as necessary for this scenario. Each Frame object stores its 3D rotation matrix \mathbf{R} and 3D position vector \vec{p} , as well as its neighboring Frames so that the whole system may be traversed from one Frame object.

Registration.py has an implementation of the Arun's registration, following the algorithm detailed above.

PivotCalibration.py and OpticalCalibration.py include functions to perform the EM tracker pivot calibration and optical tracker pivot calibration respectively.

testing.py is a script we used to debug our functions using the debug data sets and print error between our generated outputs and the expected outputs.

4 Verification of Code

To verify the Frame and Point3d class objects functioned as intended, we created random but plausible $\vec{p}_{start,A}$, \mathbf{F}_A , \mathbf{F}_B , variables and compared our output $\vec{p}_{start,B}$ to one generated on MATLAB.

$$\vec{p}_{start,A} = \begin{bmatrix} 0.5 \\ 0.5 \\ 0.5 \end{bmatrix}$$

$$\mathbf{F}_A = \begin{bmatrix} \mathbf{R}_A & \vec{p}_A \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 1.5 \\ 0 & 1 & 0 & 0.8 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{F}_B = \begin{bmatrix} \mathbf{R}_B & \vec{p}_B \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} -0.2309 & -0.9699 & 0.0772 & 0.2 \\ -0.7747 & 0.1353 & -0.6177 & 0.6 \\ 0.5887 & -0.2025 & -0.7826 & 1.3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\vec{p}_{start,B} = \mathbf{F}_{BA} \cdot \vec{p}_{start,A} = \mathbf{F}_B^{-1} \cdot \mathbf{F}_A \cdot \vec{p}_{start,A} =$$

We verified our Arun registration function by testing \mathbf{F}_D against the first frame's D coordinates (using debugging file "a") and confirmed that $\mathbf{F}_D^{-1} \cdot \vec{D}_0 = \vec{d}_0$ on MATLAB.

Refer to **Appendix** for screenshots of outputs.

To verify our pivot calibration works as expected, we calculated the average squared error between our generated C_{exp} and the actual C_{exp} in the output files for the debug sets a-g. The average error is shown below:

Set	Average Error between expected Cs (mm)
a	0.0050378
b	0.4697383
c	3.1186024
d	0.0147729
e	6.4527873
f	5.8078324
g	5.6959514

To verify our pivot calibration works as expected, we calculated the squared error $\sqrt{(g_x - c_x)^2 + (g_y - c_y)^2 + (g_z - c_z)^2}$ between our generated p_{dimple} , or g in the above equation, and the estimated post positions, c , in the output files for the debug sets a-g for both the EM probe and the optical probe. The error for both are shown below:

Set	Error in EM Pivot Positions (mm)
a	0.0055592
b	0.0061054
c	0.0594943
d	0.0056096
e	0.1979786
f	0.2582968
g	0.1295525

Set	Error in OPT Pivot Positions (mm)
a	0.0022392
b	0.0055302
c	0.0056206
d	0.0118627
e	0.0076360
f	0.0044818
g	0.0024433

Because of the low amount of error (≤ 1 mm) in both the EM and optical tracker pivot positions, we verified that our pivot calibration works as expected.

5 Results

5.1 Discussion

Looking at the errors from the debug datasets, there is significantly higher error in the expected Cs for datasets e, f, and g. We believe that this is due to the "OT jiggle", or the simulated random motion of the optical tracking about the simulated tripod. Arun's registration method may not be the best to handle the level of noise present in these datasets, so possibly switching to an iterative method that converges to an optima would have been better for the noisy datasets. Another potential solution is to remove outliers before using Arun's method. The pivot calibration datasets seem to not be too affected by noise because of the low error despite the use of Arun's method for some steps in the pivot calibration.

Surprisingly, the errors in OPT pivot position were consistently lower than EM pivot position despite the optical tracker moving with each frame, unlike the supposedly stationary EM tracker. The large error differences between P_{EM} and P_{opt} in data sets e, f, and g are likely due to the aforementioned "OT jiggle", resulting in some minute movement in the EM tracker position between frames that was accounted for by the optical calibration's implementation (since it works frame by frame) but not the pivot calibration implementation.

5.2 Unknown outputs

Our debugging results lead us to believe that our \vec{C}_i^{exp} values lie within an approximately 5 mm error margin, our P_{EM} within 0.3 mm error, and our P_{opt} within 0.015 mm error. For the text files, refer to the OUTPUT folder in the zip file. The results are also in the appendix section.

6 Partner Work

Both Kiana and Keerthana worked on all the python files as well as the report.

7 Appendix

7.1 Unknown Dataset Results

7.1.1 Dataset h

Params: 27, 8, pal-unknown-h-output3.txt
EM Pivot Position: 195.63, 189.78, 201.22
OPT Pivot Position: 409.19, 406.22, 207.82

$C_i^{expected}$ s:

211.56, 210.14, 211.61	458.56, 210.36, 443.41	333.70, 449.77, 452.99	451.27, 211.03, 448.33
211.33, 207.44, 336.58	461.87, 207.91, 568.34	331.89, 446.63, 577.94	453.08, 213.66, 573.29
211.10, 204.74, 461.55	465.17, 205.47, 693.27	330.08, 443.49, 702.88	454.88, 216.29, 698.24
213.55, 335.10, 214.32	457.74, 335.33, 445.88	333.26, 574.73, 456.12	454.75, 335.95, 445.65
213.32, 332.39, 339.29	461.04, 332.88, 570.81	331.45, 571.59, 581.07	456.55, 338.58, 570.61
213.09, 329.69, 464.26	464.35, 330.44, 695.74	329.65, 568.45, 706.02	458.35, 341.21, 695.57
215.53, 460.05, 217.02	456.91, 460.30, 448.34	332.83, 699.69, 459.26	458.23, 460.88, 442.97
215.30, 457.35, 341.99	460.22, 457.86, 573.28	331.02, 696.55, 584.20	460.03, 463.51, 567.93
215.07, 454.65, 466.96	463.52, 455.41, 698.21	329.21, 693.41, 709.15	461.83, 466.14, 692.89
336.55, 208.15, 211.80	210.15, 448.43, 209.53	458.68, 450.16, 454.81	576.21, 207.52, 446.60
336.32, 205.45, 336.77	206.78, 449.23, 334.49	456.87, 447.02, 579.76	578.02, 210.15, 571.56
336.09, 202.75, 461.74	203.42, 450.02, 459.44	455.07, 443.88, 704.70	579.82, 212.78, 696.52
338.53, 333.11, 214.50	206.70, 573.38, 208.65	458.25, 575.12, 457.94	579.69, 332.44, 443.92
338.30, 330.40, 339.47	203.34, 574.18, 333.60	456.44, 571.98, 582.89	581.49, 335.07, 568.88
338.07, 327.70, 464.44	199.98, 574.97, 458.55	454.63, 568.84, 707.84	583.29, 337.70, 693.84
340.52, 458.06, 217.21	203.26, 698.33, 207.76	457.81, 700.08, 461.07	583.17, 457.36, 441.24
340.29, 455.36, 342.18	199.89, 699.12, 332.72	456.01, 696.94, 586.02	584.97, 459.99, 566.20
340.06, 452.66, 467.15	196.53, 699.92, 457.67	454.20, 693.80, 710.97	586.77, 462.62, 691.16
461.53, 206.16, 211.99	335.06, 451.90, 212.87	448.78, 211.28, 210.79	701.15, 204.00, 444.87
461.30, 203.46, 336.96	331.69, 452.69, 337.83	447.44, 210.69, 335.78	702.95, 206.63, 569.83
461.07, 200.76, 461.93	328.33, 453.49, 462.78	446.11, 210.09, 460.78	704.76, 209.26, 694.79
463.52, 331.12, 214.69	331.61, 576.85, 211.99	446.89, 336.27, 211.37	704.63, 328.93, 442.19
463.29, 328.41, 339.66	328.25, 577.64, 336.94	445.55, 335.67, 336.36	706.43, 331.56, 567.15
463.05, 325.71, 464.63	324.88, 578.43, 461.89	444.22, 335.07, 461.35	708.23, 334.19, 692.11
465.50, 456.07, 217.40	328.17, 701.80, 211.11	445.00, 461.25, 211.95	708.10, 453.85, 439.51
465.27, 453.37, 342.37	324.80, 702.59, 336.06	443.67, 460.65, 336.94	709.91, 456.48, 564.47
465.04, 450.67, 467.34	321.44, 703.38, 461.01	442.33, 460.06, 461.93	711.71, 459.11, 689.43
208.65, 208.58, 449.99	459.96, 455.37, 216.21	573.75, 213.16, 212.14	451.24, 451.17, 210.68
211.96, 206.13, 574.92	456.60, 456.16, 341.17	572.42, 212.57, 337.13	452.05, 450.04, 335.67
215.27, 203.69, 699.85	453.24, 456.95, 466.12	571.09, 211.97, 462.12	452.85, 448.91, 460.66
207.83, 333.55, 452.45	456.52, 580.32, 215.33	571.87, 338.15, 212.71	448.58, 576.14, 211.83
211.14, 331.10, 577.39	453.15, 581.11, 340.28	570.53, 337.55, 337.70	449.38, 575.01, 336.82
214.44, 328.66, 702.32	449.79, 581.90, 465.23	569.20, 336.96, 462.69	450.19, 573.87, 461.81
207.00, 458.52, 454.92	453.07, 705.27, 214.45	569.98, 463.13, 213.29	445.91, 701.10, 212.97
210.31, 456.08, 579.85	449.71, 706.06, 339.40	568.64, 462.54, 338.28	446.72, 699.97, 337.97
213.62, 453.63, 704.79	446.35, 706.85, 464.35	567.31, 461.94, 463.27	447.53, 698.84, 462.96
333.61, 209.47, 446.70	208.71, 449.38, 451.17	698.73, 215.04, 213.48	576.21, 453.84, 209.90
336.91, 207.02, 571.63	206.90, 446.24, 576.12	697.40, 214.45, 338.47	577.02, 452.71, 334.89
340.22, 204.58, 696.56	205.09, 443.10, 701.07	696.06, 213.85, 463.46	577.82, 451.58, 459.88
332.78, 334.44, 449.17	208.28, 574.34, 454.30	696.84, 340.03, 214.05	573.55, 578.81, 211.04
336.09, 331.99, 574.10	206.47, 571.20, 579.25	695.51, 339.43, 339.05	574.35, 577.68, 336.04
339.39, 329.55, 699.03	204.66, 568.06, 704.20	694.18, 338.84, 464.04	575.16, 576.55, 461.03
331.96, 459.41, 451.63	207.84, 699.30, 457.44	694.96, 465.01, 214.63	570.88, 703.77, 212.19
335.26, 456.97, 576.57	206.03, 696.16, 582.39	693.62, 464.42, 339.62	571.69, 702.64, 337.18
338.57, 454.52, 701.50	204.23, 693.02, 707.33	692.29, 463.82, 464.61	572.49, 701.51, 462.18

701.18, 456.51, 209.11	445.73, 451.76, 575.27	567.27, 456.86, 703.54	696.76, 578.54, 454.18
701.99, 455.38, 334.11	442.34, 454.55, 700.19	571.82, 576.23, 450.84	693.36, 581.33, 579.11
702.79, 454.25, 459.10	446.89, 573.92, 447.50	568.43, 579.02, 575.76	689.97, 584.12, 704.03
698.51, 581.48, 210.26	443.49, 576.71, 572.42	565.03, 581.80, 700.69	694.52, 703.49, 451.33
699.32, 580.35, 335.25	440.10, 579.49, 697.34	569.59, 701.18, 447.99	691.13, 706.28, 576.26
700.13, 579.22, 460.25	444.65, 698.87, 444.65	566.19, 703.96, 572.91	687.73, 709.06, 701.18
695.85, 706.45, 211.41	441.26, 701.65, 569.57	562.80, 706.75, 697.84	
696.66, 705.31, 336.40	437.86, 704.44, 694.50	698.99, 453.59, 457.03	
697.46, 704.18, 461.39	574.06, 451.28, 453.69	695.60, 456.38, 581.95	
449.12, 448.97, 450.35	570.66, 454.07, 578.61	692.20, 459.17, 706.88	

7.1.2 Dataset i

Params: 27, 8, pal-unknown-i-output3.txt
EM Pivot Position: 189.32, 194.79, 195.32
OPT Pivot Position: 400.69, 391.14, 205.80

$C_i^{expected}$ s:

209.60, 211.48, 208.12	337.10, 207.87, 572.28	456.71, 455.59, 462.90	453.72, 335.80, 211.80
209.10, 211.19, 333.12	338.61, 204.22, 697.22	457.54, 578.03, 211.63	455.71, 334.41, 336.78
208.59, 210.90, 458.11	333.19, 336.45, 451.03	456.04, 579.30, 336.62	457.71, 333.03, 461.75
208.37, 336.47, 208.40	334.70, 332.79, 575.97	454.54, 580.57, 461.60	455.52, 460.78, 213.15
207.87, 336.19, 333.40	336.21, 329.14, 700.90	455.38, 703.01, 210.34	457.52, 459.39, 338.13
207.37, 335.90, 458.40	330.79, 461.37, 454.71	453.88, 704.27, 335.33	459.52, 458.01, 463.11
207.15, 461.47, 208.68	332.29, 457.71, 579.65	452.38, 705.54, 460.31	576.88, 209.03, 208.43
206.64, 461.18, 333.68	333.80, 454.06, 704.59	209.47, 451.64, 450.66	578.87, 207.65, 333.41
206.14, 460.89, 458.68	460.56, 213.97, 445.91	212.18, 450.82, 575.63	580.87, 206.27, 458.38
334.59, 212.70, 208.62	462.07, 210.32, 570.85	214.88, 449.99, 700.60	578.69, 334.01, 209.78
334.09, 212.42, 333.62	463.58, 206.66, 695.79	209.73, 576.64, 451.48	580.68, 332.63, 334.76
333.59, 212.13, 458.62	458.16, 338.89, 449.59	212.43, 575.81, 576.45	582.68, 331.24, 459.74
333.37, 337.70, 208.91	459.66, 335.24, 574.53	215.14, 574.99, 701.41	580.50, 458.99, 211.14
332.86, 337.41, 333.90	461.17, 331.58, 699.47	209.98, 701.63, 452.30	582.49, 457.61, 336.11
332.36, 337.12, 458.90	455.75, 463.81, 453.27	212.69, 700.81, 577.26	584.49, 456.22, 461.09
332.14, 462.69, 209.19	457.26, 460.16, 578.21	215.39, 699.99, 702.23	701.85, 207.24, 206.42
331.64, 462.40, 334.19	458.77, 456.51, 703.15	334.44, 451.40, 447.95	703.84, 205.86, 331.39
331.13, 462.12, 459.19	209.77, 448.70, 209.97	337.15, 450.58, 572.92	705.84, 204.48, 456.37
459.59, 213.93, 209.13	208.27, 449.96, 334.95	339.85, 449.76, 697.89	703.66, 332.22, 207.77
459.08, 213.64, 334.13	206.76, 451.23, 459.94	334.70, 576.40, 448.77	705.65, 330.84, 332.75
458.58, 213.35, 459.13	207.60, 573.67, 208.68	337.40, 575.58, 573.74	707.65, 329.46, 457.72
458.36, 338.92, 209.41	206.10, 574.94, 333.66	340.11, 574.75, 698.71	705.47, 457.20, 209.12
457.86, 338.64, 334.41	204.60, 576.21, 458.65	334.95, 701.40, 449.59	707.46, 455.82, 334.10
457.35, 338.35, 459.41	205.44, 698.65, 207.38	337.66, 700.57, 574.56	709.46, 454.44, 459.07
457.13, 463.92, 209.69	203.93, 699.91, 332.37	340.36, 699.75, 699.53	449.41, 210.08, 451.27
456.63, 463.63, 334.69	202.43, 701.18, 457.35	459.41, 451.17, 445.25	447.87, 211.31, 576.26
456.13, 463.34, 459.69	334.74, 450.88, 211.45	462.12, 450.34, 570.22	446.32, 212.54, 701.24
210.62, 209.08, 448.78	333.24, 452.15, 336.43	464.82, 449.52, 695.18	446.25, 335.04, 450.01
212.13, 205.42, 573.72	331.74, 453.41, 461.42	459.67, 576.16, 446.07	444.71, 336.27, 574.99
213.64, 201.77, 698.66	332.57, 575.85, 210.16	462.37, 575.34, 571.03	443.16, 337.49, 699.98
208.22, 334.00, 452.47	331.07, 577.12, 335.14	465.08, 574.52, 696.00	443.09, 459.99, 448.74
209.73, 330.35, 577.40	329.57, 578.39, 460.12	459.92, 701.16, 446.88	441.55, 461.22, 573.72
211.24, 326.69, 702.34	330.41, 700.83, 208.86	462.63, 700.34, 571.85	440.00, 462.45, 698.71
205.82, 458.92, 456.15	328.91, 702.09, 333.85	465.33, 699.51, 696.82	574.36, 213.26, 452.79
207.33, 455.27, 581.09	327.41, 703.36, 458.83	451.91, 210.82, 210.45	572.82, 214.49, 577.77
208.83, 451.62, 706.02	459.71, 453.06, 212.93	453.90, 209.43, 335.42	571.27, 215.72, 702.76
335.59, 211.52, 447.35	458.21, 454.33, 337.91	455.90, 208.05, 460.40	571.20, 338.21, 451.52

569.66, 339.44, 576.50	448.38, 574.64, 333.96	698.31, 578.77, 329.60	575.14, 575.73, 574.52
568.11, 340.67, 701.49	450.54, 575.84, 458.93	700.47, 579.97, 454.58	577.50, 573.12, 699.47
568.04, 463.17, 450.25	444.14, 698.41, 207.82	694.07, 702.54, 203.46	570.55, 703.29, 452.22
566.50, 464.40, 575.24	446.30, 699.61, 332.79	696.22, 703.74, 328.44	572.90, 700.68, 577.17
564.95, 465.62, 700.22	448.45, 700.82, 457.77	698.38, 704.94, 453.41	575.25, 698.07, 702.12
699.31, 216.44, 454.30	573.27, 450.52, 207.97	450.08, 451.09, 449.22	700.00, 455.68, 444.61
697.77, 217.66, 579.28	575.43, 451.72, 332.95	452.43, 448.48, 574.17	702.35, 453.07, 569.56
696.22, 218.89, 704.27	577.59, 452.93, 457.92	454.78, 445.87, 699.12	704.70, 450.46, 694.51
696.15, 341.39, 453.03	571.19, 575.50, 206.81	447.84, 576.04, 451.87	697.75, 580.63, 447.27
694.61, 342.62, 578.02	573.34, 576.70, 331.78	450.19, 573.43, 576.82	700.10, 578.02, 572.22
693.06, 343.85, 703.00	575.50, 577.90, 456.76	452.54, 570.82, 701.77	702.45, 575.41, 697.17
692.99, 466.34, 451.76	569.10, 700.48, 205.64	445.59, 700.99, 454.52	695.51, 705.58, 449.92
691.45, 467.57, 576.75	571.26, 701.68, 330.62	447.94, 698.38, 579.47	697.86, 702.97, 574.87
689.90, 468.80, 701.73	573.42, 702.88, 455.59	450.29, 695.77, 704.42	700.21, 700.36, 699.82
448.31, 448.46, 210.15	698.24, 452.59, 205.79	575.04, 453.38, 446.92	
450.46, 449.66, 335.12	700.39, 453.79, 330.77	577.39, 450.77, 571.87	
452.62, 450.86, 460.10	702.55, 454.99, 455.75	579.74, 448.16, 696.82	
446.22, 573.44, 208.98	696.15, 577.56, 204.63	572.79, 578.34, 449.57	

7.1.3 Dataset j

Params: 27, 8, pal-unknown-j-output3.txt
EM Pivot Position: 201.39, 207.10, 207.14
OPT Pivot Position: 394.80, 392.47, 201.56

$C_i^{expected}$ s:

208.96, 211.60, 210.83	208.20, 218.74, 700.38	208.50, 576.93, 333.76	217.80, 698.17, 455.61
208.34, 214.19, 335.80	208.16, 336.29, 446.79	211.06, 578.37, 458.72	219.78, 694.42, 580.54
207.72, 216.77, 460.77	206.79, 339.97, 571.73	203.09, 700.45, 207.41	221.76, 690.68, 705.47
210.86, 336.56, 208.25	205.42, 343.65, 696.67	205.65, 701.89, 332.38	336.76, 445.41, 446.15
210.24, 339.14, 333.23	205.39, 461.21, 443.08	208.21, 703.33, 457.34	338.74, 441.67, 571.08
209.62, 341.73, 458.20	204.02, 464.89, 568.02	333.73, 453.35, 207.58	340.72, 437.93, 696.01
212.76, 461.52, 205.68	202.65, 468.57, 692.96	336.29, 454.79, 332.55	339.75, 570.32, 449.85
212.14, 464.10, 330.65	335.90, 214.19, 451.79	338.85, 456.23, 457.52	341.73, 566.58, 574.77
211.52, 466.69, 455.62	334.53, 217.87, 576.73	330.88, 578.31, 206.20	343.71, 562.84, 699.70
333.94, 209.71, 211.49	333.16, 221.55, 701.66	333.44, 579.75, 331.17	342.75, 695.23, 453.54
333.32, 212.30, 336.46	333.13, 339.11, 448.08	336.00, 581.19, 456.13	344.73, 691.49, 578.47
332.71, 214.88, 461.43	331.76, 342.79, 573.02	328.03, 703.27, 204.82	346.71, 687.75, 703.40
335.84, 334.67, 208.91	330.39, 346.47, 697.95	330.59, 704.71, 329.79	461.71, 442.48, 444.08
335.22, 337.26, 333.88	330.35, 464.02, 444.37	333.15, 706.15, 454.75	463.69, 438.74, 569.01
334.61, 339.84, 458.86	328.98, 467.70, 569.31	458.67, 456.17, 204.99	465.67, 435.00, 693.94
337.74, 459.63, 206.34	327.61, 471.38, 694.24	461.23, 457.61, 329.96	464.70, 567.39, 447.78
337.13, 462.22, 331.31	460.86, 217.01, 453.07	463.79, 459.06, 454.93	466.68, 563.65, 572.71
336.51, 464.80, 456.28	459.49, 220.69, 578.01	455.82, 581.13, 203.61	468.66, 559.91, 697.63
458.93, 207.83, 212.14	458.12, 224.37, 702.95	458.38, 582.57, 328.58	467.70, 692.30, 451.47
458.31, 210.41, 337.12	458.09, 341.92, 449.36	460.94, 584.01, 453.54	469.68, 688.56, 576.40
457.69, 213.00, 462.09	456.72, 345.60, 574.30	452.97, 706.09, 202.23	471.66, 684.82, 701.33
460.83, 332.79, 209.57	455.35, 349.28, 699.24	455.53, 707.53, 327.20	450.13, 208.32, 211.08
460.21, 335.37, 334.54	455.31, 466.83, 445.66	458.09, 708.97, 452.16	448.82, 211.96, 336.02
459.59, 337.95, 459.51	453.94, 470.52, 570.59	211.81, 448.35, 448.22	447.50, 215.60, 460.96
462.73, 457.75, 207.00	452.57, 474.20, 695.53	213.79, 444.60, 573.15	450.76, 333.26, 207.44
462.11, 460.33, 331.97	208.79, 450.53, 210.18	215.77, 440.86, 698.08	449.45, 336.90, 332.38
461.49, 462.91, 456.94	211.35, 451.97, 335.14	214.81, 573.26, 451.92	448.14, 340.54, 457.33
210.94, 211.38, 450.50	213.91, 453.41, 460.11	216.79, 569.51, 576.84	451.40, 458.21, 203.81
209.57, 215.06, 575.44	205.94, 575.49, 208.79	218.77, 565.77, 701.77	450.08, 461.85, 328.75

448.77, 465.49, 453.69	455.16, 461.02, 567.99	443.92, 699.76, 215.91	455.13, 569.59, 700.62
575.12, 207.72, 212.41	458.41, 464.05, 692.91	445.38, 696.39, 340.86	455.89, 698.10, 452.40
573.81, 211.36, 337.35	575.55, 207.33, 445.94	446.84, 693.01, 465.80	456.50, 696.33, 577.39
572.49, 215.00, 462.29	578.79, 210.36, 570.86	573.52, 452.25, 207.71	457.12, 694.57, 702.37
575.76, 332.67, 208.78	582.04, 213.39, 695.78	574.98, 448.87, 332.66	576.90, 446.18, 448.25
574.44, 336.31, 333.72	576.21, 332.29, 442.89	576.44, 445.50, 457.60	577.51, 444.41, 573.23
573.13, 339.95, 458.66	579.45, 335.32, 567.81	571.21, 577.18, 211.11	578.12, 442.64, 698.22
576.39, 457.62, 205.15	582.70, 338.35, 692.73	572.66, 573.81, 336.06	578.89, 571.15, 450.00
575.07, 461.25, 330.09	576.87, 457.25, 439.84	574.12, 570.43, 461.01	579.50, 569.38, 574.99
573.76, 464.89, 455.03	580.11, 460.28, 564.76	568.89, 702.11, 214.52	580.11, 567.62, 699.98
700.11, 207.13, 213.74	583.36, 463.31, 689.68	570.35, 698.74, 339.46	580.87, 696.12, 451.76
698.80, 210.77, 338.68	700.50, 206.59, 442.71	571.81, 695.36, 464.41	581.49, 694.35, 576.75
697.48, 214.41, 463.62	703.75, 209.62, 567.63	698.49, 454.60, 206.32	582.10, 692.59, 701.73
700.75, 332.08, 210.11	707.00, 212.65, 692.55	699.95, 451.23, 331.26	701.88, 444.20, 447.61
699.43, 335.71, 335.05	701.16, 331.55, 439.66	701.41, 447.85, 456.21	702.49, 442.43, 572.59
698.12, 339.35, 459.99	704.41, 334.58, 564.58	696.18, 579.53, 209.72	703.11, 440.67, 697.58
701.38, 457.02, 206.48	707.66, 337.61, 689.50	697.64, 576.16, 334.66	703.87, 569.17, 449.36
700.07, 460.66, 331.42	701.82, 456.51, 436.61	699.09, 572.78, 459.61	704.48, 567.40, 574.35
698.75, 464.30, 456.36	705.07, 459.54, 561.53	693.86, 704.46, 213.12	705.09, 565.64, 699.34
450.59, 208.07, 449.17	708.32, 462.58, 686.45	695.32, 701.09, 338.06	705.86, 694.14, 451.12
453.84, 211.10, 574.09	448.55, 449.90, 209.11	696.78, 697.72, 463.01	706.47, 692.38, 576.11
457.08, 214.13, 699.01	450.01, 446.52, 334.06	451.92, 448.16, 448.89	707.08, 690.61, 701.09
451.25, 333.03, 446.12	451.47, 443.15, 459.00	452.53, 446.39, 573.87	
454.50, 336.06, 571.04	446.24, 574.83, 212.51	453.14, 444.62, 698.86	
457.75, 339.09, 695.96	447.69, 571.45, 337.46	453.90, 573.13, 450.64	
451.91, 457.99, 443.07	449.15, 568.08, 462.40	454.52, 571.36, 575.63	

7.1.4 Dataset k

Params: 27, 8, pal-unknown-k-output3.txt
EM Pivot Position: 206.04, 206.12, 193.14
OPT Pivot Postion: 400.62, 392.67, 203.25

$C_i^{expected}$ s:

208.20, 211.82, 211.86	458.19, 336.82, 209.67	326.59, 464.53, 443.66	333.58, 447.55, 211.44
210.17, 210.07, 336.83	460.16, 335.06, 334.65	328.80, 466.77, 568.62	333.17, 448.65, 336.44
212.14, 208.31, 461.81	462.14, 333.31, 459.62	331.01, 469.01, 693.59	332.76, 449.75, 461.43
208.23, 336.81, 213.61	458.22, 461.80, 211.43	458.24, 217.98, 445.75	335.56, 572.53, 210.34
210.20, 335.05, 338.59	460.19, 460.05, 336.40	460.46, 220.22, 570.71	335.15, 573.63, 335.34
212.17, 333.30, 463.56	462.16, 458.30, 461.37	462.67, 222.46, 695.67	334.74, 574.73, 460.33
208.25, 461.79, 215.37	208.37, 211.34, 450.30	454.88, 342.92, 443.57	337.55, 697.51, 209.25
210.22, 460.04, 340.34	210.59, 213.58, 575.26	457.10, 345.16, 568.53	337.14, 698.61, 334.24
212.19, 458.29, 465.31	212.80, 215.82, 700.22	459.31, 347.40, 693.49	336.73, 699.71, 459.24
333.19, 211.82, 209.89	205.01, 336.27, 448.12	451.52, 467.85, 441.39	458.56, 445.57, 211.87
335.16, 210.07, 334.86	207.23, 338.51, 573.08	453.73, 470.09, 566.35	458.15, 446.67, 336.86
337.13, 208.32, 459.84	209.44, 340.75, 698.04	455.95, 472.33, 691.31	457.75, 447.78, 461.86
333.21, 336.81, 211.64	201.65, 461.21, 445.94	208.60, 449.53, 211.01	460.55, 570.55, 210.77
335.18, 335.06, 336.62	203.86, 463.45, 570.90	208.19, 450.63, 336.01	460.14, 571.65, 335.77
337.15, 333.31, 461.59	206.08, 465.69, 695.86	207.78, 451.73, 461.00	459.73, 572.76, 460.76
333.23, 461.80, 213.40	333.31, 214.66, 448.02	210.58, 574.51, 209.92	462.53, 695.53, 209.67
335.20, 460.05, 338.37	335.52, 216.90, 572.99	210.17, 575.61, 334.91	462.12, 696.63, 334.67
337.17, 458.29, 463.34	337.73, 219.14, 697.95	209.76, 576.71, 459.91	461.71, 697.73, 459.66
458.17, 211.83, 207.92	329.95, 339.60, 445.84	212.56, 699.49, 208.82	211.97, 448.70, 450.39
460.14, 210.08, 332.89	332.16, 341.84, 570.81	212.15, 700.59, 333.82	211.94, 451.17, 575.36
462.11, 208.32, 457.87	334.37, 344.08, 695.77	211.74, 701.69, 458.81	211.92, 453.64, 700.34

212.36, 573.68, 447.92	575.02, 210.66, 336.03	576.49, 460.94, 700.55	694.22, 704.77, 214.90
212.34, 576.14, 572.90	575.23, 213.76, 460.99	700.84, 214.37, 446.85	692.51, 704.03, 339.88
212.31, 578.61, 697.87	577.94, 332.48, 207.96	702.38, 213.29, 571.84	690.80, 703.29, 464.87
212.76, 698.65, 445.45	578.16, 335.58, 332.93	703.92, 212.20, 696.83	450.02, 451.42, 449.31
212.73, 701.12, 570.43	578.37, 338.68, 457.89	699.61, 339.36, 447.95	446.32, 450.66, 574.26
212.70, 703.59, 695.40	581.08, 457.40, 204.86	701.16, 338.28, 572.94	442.62, 449.91, 699.20
336.97, 448.31, 450.42	581.29, 460.50, 329.82	702.70, 337.19, 697.92	449.20, 576.41, 450.05
336.94, 450.78, 575.40	581.51, 463.60, 454.78	698.39, 464.35, 449.05	445.49, 575.66, 574.99
336.92, 453.24, 700.37	699.76, 204.41, 210.93	699.93, 463.27, 574.04	441.79, 574.90, 699.93
337.36, 573.28, 447.95	699.98, 207.51, 335.89	701.47, 462.18, 699.02	448.37, 701.41, 450.78
337.34, 575.75, 572.93	700.19, 210.61, 460.85	448.86, 450.26, 210.03	444.66, 700.65, 575.72
337.31, 578.22, 697.91	702.90, 329.33, 207.82	447.15, 449.52, 335.02	440.96, 699.90, 700.66
337.76, 698.26, 445.49	703.12, 332.43, 332.79	445.44, 448.78, 460.01	574.97, 452.23, 453.02
337.73, 700.73, 570.46	703.33, 335.53, 457.75	446.57, 575.24, 210.74	571.26, 451.47, 577.97
337.70, 703.19, 695.44	706.04, 454.26, 204.72	444.86, 574.50, 335.73	567.56, 450.71, 702.91
461.97, 447.92, 450.46	706.26, 457.36, 329.68	443.15, 573.76, 460.72	574.14, 577.22, 453.76
461.94, 450.39, 575.43	706.47, 460.46, 454.64	444.29, 700.22, 211.45	570.43, 576.46, 578.70
461.92, 452.85, 700.41	450.87, 211.89, 449.92	442.58, 699.48, 336.44	566.73, 575.71, 703.64
462.36, 572.89, 447.99	452.41, 210.81, 574.90	440.87, 698.73, 461.43	573.31, 702.22, 454.49
462.34, 575.36, 572.97	453.96, 209.73, 699.89	573.83, 452.54, 211.76	569.61, 701.46, 579.43
462.31, 577.83, 697.94	449.64, 336.88, 451.02	572.12, 451.80, 336.74	565.90, 700.70, 704.37
462.75, 697.87, 445.52	451.19, 335.80, 576.00	570.41, 451.06, 461.73	699.91, 453.03, 456.73
462.73, 700.34, 570.50	452.73, 334.72, 700.99	571.54, 577.52, 212.47	696.21, 452.28, 581.68
462.70, 702.80, 695.47	448.42, 461.87, 452.12	569.83, 576.77, 337.45	692.50, 451.52, 706.62
449.84, 210.70, 211.21	449.96, 460.79, 577.10	568.12, 576.03, 462.44	699.08, 578.03, 457.46
450.06, 213.80, 336.17	451.50, 459.70, 702.09	569.25, 702.49, 213.18	695.38, 577.27, 582.41
450.27, 216.90, 461.13	575.85, 213.13, 448.39	567.54, 701.75, 338.16	691.67, 576.51, 707.35
452.98, 335.62, 208.11	577.40, 212.05, 573.37	565.84, 701.01, 463.15	698.25, 703.02, 458.20
453.19, 338.72, 333.07	578.94, 210.96, 698.36	698.79, 454.81, 213.48	694.55, 702.27, 583.14
453.41, 341.82, 458.03	574.63, 338.12, 449.48	697.08, 454.07, 338.47	690.84, 701.51, 708.08
456.12, 460.54, 205.00	576.17, 337.04, 574.47	695.37, 453.33, 463.45	
456.33, 463.64, 329.96	577.71, 335.95, 699.46	696.51, 579.79, 214.19	
456.55, 466.74, 454.92	573.40, 463.11, 450.58	694.80, 579.05, 339.18	
574.80, 207.56, 211.07	574.95, 462.03, 575.57	693.09, 578.31, 464.16	

7.2 Debugging Examples

```

In [27]: F_D = registrationArunMethod(d, D[0], "D")
...: print('D[0]:',D[0])
...: print('d:',d)
...: print('F_D:',F_D.R, F_D.p.coords)
D[0]: [[ 0.  0. -1500.]
 [ 0.  0. -1350.]
 [ 0. 150. -1500.]
 [ 0. 150. -1350.]
 [150.  0. -1500.]
 [150.  0. -1350.]
 [150. 150. -1500.]
 [150. 150. -1350.]]
d: [[ 0.  0.  0.]
 [ 0.  0. 150.]
 [ 0. 150.  0.]
 [ 0. 150. 150.]
 [150.  0.  0.]
 [150.  0. 150.]
 [150. 150.  0.]
 [150. 150. 150.]]
F_D: [[ 1.00000000e+00  3.49202726e-16  8.42200963e-17]
 [-1.93335660e-16  1.00000000e+00 -6.70448864e-17]
 [ 2.01213401e-17 -2.67928760e-16  1.00000000e+00]] [-1.42108547e-14  2.84217094e-14
-1.50000000e+03]

```

D_0 =

```

      0      0      -1500
      0      0      -1350
      0     150     -1500
      0     150     -1350
    150      0     -1500
    150      0     -1350
    150     150     -1500
    150     150     -1350

```

```
>> d = [ 0.  0.  0.; 0.  0. 150.; 0.
```

d =

```

      0      0      0
      0      0     150
      0     150      0
      0     150     150
    150      0      0
    150      0     150
    150     150      0
    150     150     150

```

```
>> d_0 = d(1,1:3)'
```

d_0 =

```

      0
      0
      0

```

```
>> invF_D = [R_D' -R_D'*p_D; 0 0 0 1]
```

invF_D =

1.0e+03 *

```

      0.0010  -0.0000  0.0000  0.0000
      0.0000  0.0010  -0.0000  -0.0000
      0.0000  -0.0000  0.0010  1.5000
           0           0           0  0.0010

```

```
>> invF_D * [D_0(1,1:3)';1]
```

ans =

```

      0.0000
     -0.0000
           0
      1.0000

```

ans == d_0 → verified

```

>> R_A = [1 0 0; 0 0 -1; 0 1 0]

R_A =

     1     0     0
     0     0    -1
     0     1     0

>> R_B = [-0.2309 -0.9699 0.0772; -0.7747 0.1353 -0.6177; 0.5887 -0.2025 -0.7826]

R_B =

    -0.2309    -0.9699     0.0772
    -0.7747     0.1353    -0.6177
     0.5887    -0.2025    -0.7826

>> p_A = [0; 1.5; 0.8]; p_B = [0.2; 0.6; 1.3];
>> F_A = [R_A p_A; 0 0 0 1]; F_B = [R_B p_B; 0 0 0 1];
>> p_start_A = [0.5; 0.5; 0.5];
>> inv_F_B = [R_B' -R_B'*p_B; 0 0 0 1];

>> inv_F_B * F_A * [p_start_A; 1]

ans =

    -0.3791
    -0.2369
    -0.2239
     1.0000

```

```

In [17]: R_A = np.array([[1.0000, 0, 0],[0, -0.00, -1.0000],[0, 1.0000, -0.0000]]); R_B =
np.array([[ -0.2309, -0.9699, 0.0772],[ -0.7747, 0.1353,-0.6177],[ 0.5887, -0.2025,
-0.7826]]); p_B = Point3d("B",0.2,0.6, 1.3); p_A = Point3d("A",0, 1.5000, 0.8000); p_start_A
= Point3d("A",0.5,0.5,0.5); frame_A = Frame("A",R_A,p_A); frame_B = Frame("B",R_B,p_B,
[frame_A]); frame_B.transformPoint(p_start_A)
Point: [-0.37915 -0.23685 -0.22392] in frame: B from A
Out[17]: <Point3d.Point3d at 0x7fe9210f7fa0>

```