Principal component analysis for the *UWB*-network calibration problem

Marcus Greiff[1]
22/02-2017

***Abstract*— This document describes a method of using principal component analysis in order to tune an *UWB* network. Created on behalf of Bitcraze AB.**

I. PROBLEM FORMULATION

An ultra-wideband (*UWB*) network typically consists of a set of N senders which communicate with a set of receivers using wide-band radio. In the considered case, the network is to be used for *UWB* localisation of small unmanned vehicles in $\mathbb{R}^3$. In order to localise the receivers in the room, the absolute positions, $\mathbf{s}_i \in \mathbb{R}^3$, of the $i \in [1, N]$ need to be known. The problem is then to identify all position $\mathbf{s}_i$ based solely on distance measurements $\hat{\mathbf{d}}_{ij} = ||\mathbf{s}_i - \mathbf{s}_j||_2 + \varepsilon_{ij}$ for some error term ε_{ij}. For this purpose we consider techniques of constrained principal component analysis (*PCA*) similar to that of [1] [2], complete with a Python implementation.

II. DEFINITIONS

In order to proceed with the method, we first define the operations $\mathbb{E}[\cdot]$ and $\mathbb{V}[\cdot]$ as the expectation value and variance of a random variable such that if $\mathbb{X} \sim \mathcal{N}(\mu, \sigma^2)$, $\mathbb{E}[\mathbb{X}] = \mu$ and $\mathbb{V}[\mathbb{X}] = \sigma^2$.

Given two vectors $\mathbf{u}, \mathbf{v} \in \mathbb{R}^3$, we define the skew symmetric tensor as

$$[\mathbf{u}]_\times = \begin{bmatrix} 0 & -u_3 & u_2 \\ u_3 & 0 & -u_1 \\ -u_2 & u_1 & 0 \end{bmatrix}, \tag{1}$$

satisfying $[u]_\times \mathbf{v} = \mathbf{u} \times \mathbf{v}$.

With this notation, the rotation of a vector $\mathbf{u}$ to an intended vector $\mathbf{v}$ may be done by the simple operation $\mathbf{v} = \mathbf{R}(\mathbf{u}, \mathbf{v})\mathbf{u}$, defined by

$$\theta_{\mathbf{u},\mathbf{v}} = \arccos\Big(\frac{\mathbf{u} \cdot \mathbf{v}}{||\mathbf{u}|| \cdot ||\mathbf{v}||}\Big), \tag{2}$$

and then

$$\mathbf{R}_{rot}(\mathbf{u}, \mathbf{v}) = \mathbf{I}_{3\times 3} - \sin(\theta_{\mathbf{u},\mathbf{v}})[\mathbf{u}]_\times + (1 - \cos(\theta_{\mathbf{u},\mathbf{v}}))[\mathbf{u}]_\times^2. \tag{3}$$

We also recall that reflection of a vector $\mathbf{u}$ in a hyperplane with normal $\mathbf{v}$ is given by

$$\mathbf{R}_{ref}(\mathbf{u}) = \mathbf{v} - 2\frac{\mathbf{u} \cdot \mathbf{v}}{||\mathbf{v}||_2^2}\mathbf{v}. \tag{4}$$

See [3] for useful explanations on the rotation operators.

[1]Marcus Greiff holds an Ms.C. in automatic control from Lund University
marcusgreiff.93@hotmail.com

III. OUTLIER-REJECTION AND DIAGNOSTICS

Due to the asymmetric radiation patterns of the antenna in the *UWB* system and the possibility of non-line-of-sight between senders, the error term ε_{ij} in the ranging need not be zero-mean nor normally distributed. The outlier measurements typically resulting in offsets of meters from the true ranging, and will have to be removed before proceeding with the *PCA*.

For this purpose, we consider assimilated data of ranges corresponding to a pair of senders d_{ij} independently. Consider a vector $\mathbf{d}_{ij}$ containing all measured ranges $\hat{d}_{ij}$ from sender i to sender j. By iteratively computing the mean expectation of $\mathbf{d}_{ij}$, we may remove all measurements satisfying

$$|\hat{d}_{ij} - \mathbb{E}[\mathbf{d}_{ij}]| > \alpha\sqrt{\mathbb{V}[\mathbf{d}_{ij}]}. \tag{5}$$

for some $\alpha > 2$. The Kolmogorov-Smirnov normality test is run on both the original and outlier compensated $\mathbf{d}_{ij}$ vectors, providing information on the sender positioning, hinting if they should be moved to increase performance.

IV. MULTI-DIMENSIONAL SCALING

The problem determining the sender positions in $P = 3$ dimensions is solved by first defining the matrix $\mathbf{S} = \begin{bmatrix} \mathbf{s}_1 & \cdots & \mathbf{s}_N \end{bmatrix} \in \mathbb{R}^{P\times N}$. The goal is to find these coordinates based solely on the distances between anchors, which, assuming $\mathbb{E}[\mathbf{d}_i] = 0$, may be written $\mathbf{d}_{ij} = ||\mathbf{s}_i - \mathbf{s}_j|| = (\mathbf{s}_i - \mathbf{s}_j)^T(\mathbf{s}_i - \mathbf{s}_j)$. We may then form the matrix $\mathbf{D} = [d_{ij}]$, and our goal is now to find $\mathbf{S}_i$ from $\mathbf{D}$.

Fist, we the define a centralising matrix $\mathbf{H} = \mathbf{I}_{N\times N} - \frac{1}{N}\mathbf{1}_N\mathbf{1}_N^T$. If defining $\mathbf{B} = [b_{ij}] = -\frac{1}{2}\mathbf{HDH}$, it follows that

$$b_{ij} = (\mathbf{s}_i - \sum_{n=1}^{N}\mathbf{s}_n)^T(\mathbf{s}_j - \sum_{n=1}^{N}\mathbf{s}_n). \tag{6}$$

Subjecting $\mathbf{B}$ to an Eigen-decomposition, we find that

$$\mathbf{B} = (\mathbf{HS})(\mathbf{HS})^T = \mathbf{V}^T\Lambda\mathbf{V} \tag{7}$$

where $\mathbf{V}$ contains the eigenvectors of $\mathbf{B}$ and $\Lambda = \text{diag}\{\lambda_1, \cdots, \lambda_N\}$ has the corresponding eigenvalues on its diagonal. The principal components of $\mathbf{B}$ with $P = 3$ will correspond to the three largest eigenvalues [1], as $\text{rank}(\mathbf{S}) = P = 3$ if $P < N$ and the positions $\mathbf{s}_i$ presumably span a space in $\mathbb{R}^3$. If denoting the P eigenvalues of Λ as $\tilde{\lambda}_1 > \cdots > \tilde{\lambda}_P$, and form $\tilde{\Lambda}_P = \text{diag}\{\tilde{\lambda}_1, \cdots, \tilde{\lambda}_P\} \in \mathbb{R}^{3\times 3}$ with the corresponding eigenvectors $\tilde{\mathbf{V}}_P \in \mathbb{R}^{N\times 3}$, we see that

$$(\mathbf{HS})(\mathbf{HS})^T = \mathbf{V}^T\Lambda\mathbf{V} = \tilde{\mathbf{V}}_P\tilde{\Lambda}_P\tilde{\mathbf{V}}_P^T = (\tilde{\mathbf{V}}_P\tilde{\Lambda}_P^{1/2})(\tilde{\mathbf{V}}_P\tilde{\Lambda}_P^{1/2})^T. \tag{8}$$

As $\mathbf{H}$ is an affine map, we have in $\tilde{\mathbf{S}} = \tilde{\mathbf{V}}_P \tilde{\Lambda}_P^{1/2}$ found our desired coordinates differing from the true coordinates $\mathbf{S}$ only in terms of translation and rotation.

V. Enforcing constraints

In order to determine the affine transformation from $\tilde{\mathbf{S}}$ to $\mathbf{S}$, some information regarding the true coordinates needs to be provided. Typically, this may be done by defining (i) a set of anchors which reside in a plane (ii) an axis and (iii) a point of origin.

A. Plane rotation

Here the user inputs three indices $\{a,b,c\}$ corresponding to points in $\mathbf{S}$ residing in a known plane with a normal $\mathbf{n}$. This could for instance be three anchors placed on the ground with the normal of the plane being $\mathbf{n} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T$. By computing the normal of this plane in $\tilde{\mathbf{S}}$ as $\tilde{\mathbf{n}} = (\tilde{\mathbf{s}}_a - \tilde{\mathbf{s}}_b) \times (\tilde{\mathbf{s}}_c - \tilde{\mathbf{s}}_b)$, the positions may be rotated to yield $\tilde{\mathbf{n}}_{current} || \mathbf{n}$ by the operation

$$\tilde{\mathbf{S}} := \mathbf{R}_{rot}(\tilde{\mathbf{n}}, \mathbf{n}) \tilde{\mathbf{S}}. \tag{9}$$

B. Axis rotation

Similar to the *Plane calibration*, the axis calibration takes two indices $\{a,b\}$ of points in $\mathbf{S}$ defining a known axis, $\mathbf{n}$, letting $\tilde{\mathbf{n}} = \tilde{\mathbf{s}}_a - \tilde{\mathbf{s}}_b$ and performing the operation in (9).

C. Axis mirroring

After two rotations, either with axis or planes, the rotational component of the affine map from $\tilde{\mathbf{S}}$ to $\mathbf{S}$ has been removed and the calibrated sender positions only differ from the true positions in terms of mirroring and translation. By providing indices $\{a,b\}$ of points in $\mathbf{S}$ defining, which should correspond to some vector $\mathbf{n}$ in $\mathbf{S}$. Letting $\tilde{\mathbf{n}} = \tilde{\mathbf{s}}_a - \tilde{\mathbf{s}}_b$, we then check if $\mathbf{n} \cdot \tilde{\mathbf{n}} < 0$. If the scalar product is negative, the system is mirrored according to

$$\tilde{\mathbf{S}} := \mathbf{R}_{ref}(\tilde{\mathbf{n}}) \tilde{\mathbf{S}}. \tag{10}$$

D. Origin translation

translation of an entire cluster of points by a translation of $\mathbf{p} \in \mathbb{R}^3$, accomplished by

$$\tilde{\mathbf{S}} = \tilde{\mathbf{S}} - (\mathbf{p} \otimes \mathbf{1}_N^T), \tag{11}$$

with the translated positions, now confirming with the true positions. Typically, $\mathbf{p}$ is chosen as one of the anchors, but it may not be necessary to chose it at all, depending on the intended application.

VI. Results

The above theory allows the *UWB* system to be calibrated, taking physical measurements as the input and returning the true anchor positions. Typically, the plane constraint may be useful if placing many anchors on the ground, as this allows for a more robust identification of the z-direction which is the most crucial direction when considering *UAV* applications.

Relying on simple arithmetic operations, the algorithm scales very well. For the considered problems of 6-10 senders, the computational time is measured in $[ms]$. But due to the polynomial scaling of the Eigen-decomposition algorithm being $O(N^3)$, the algorithm can calibrate systems of up to 400 senders in < 3 $[s]$ (see Figure 1).

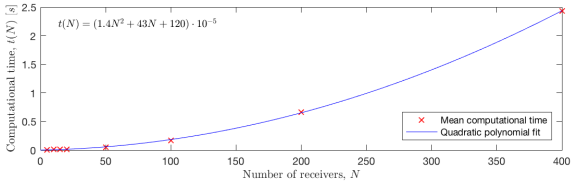

Fig. 1. Mean computational time of 50 runs as a function of the number of senders, N, indicating showing polynomial time scaling as expected.

With the constraint formulations, and assuming $\varepsilon_{ij} \sim \mathcal{N}(0,0)$ $\forall (i,j)$, problems can be solved down to numerical precision accuracy. In this example, a problem with 20 random anchors is solved with a residual of $||\mathbf{S} - \tilde{\mathbf{S}}||_2 \approx 10^{-14}$ (see Figure 2).

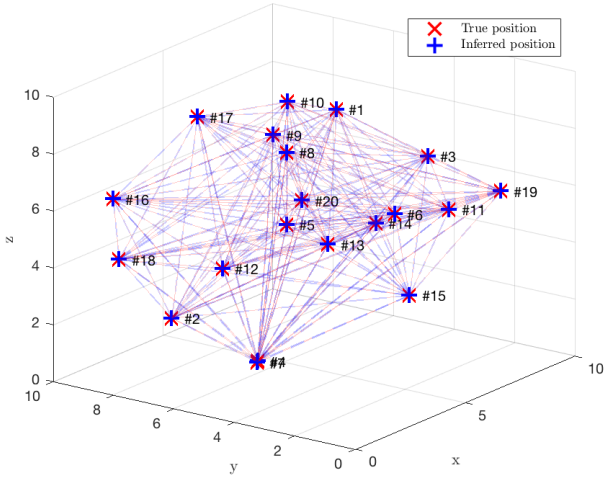

Fig. 2. The multidimensional scaling algorithm run on 20 random sender positions, with perfect knowledge of the distance matrix between the anchors with ε_{ij} $\mathcal{N}(0,0)$ $\forall (i,j)$. The resulting identification is correct with a residual $||\mathbf{S} - \tilde{\mathbf{S}}||_2 \approx 10^{-14}$ when using one plane rotation, one axis rotation, axis mirroring and coordinate translation.

When instead assuming noise in the measurements, with $\varepsilon_{ij} \sim \mathcal{N}(0.01, 0.01)$ $[m]$, the residual across all 20 anchors is typically $||\mathbf{S} - \tilde{\mathbf{S}}||_2 \approx 4.8 \cdot 10^{-2}$, with errors of $||\mathbf{s}_i - \tilde{\mathbf{s}}_i||_2 \approx 5.4 \cdot 10^{-3}$ $[m]$ for individual senders positions.

VII. Conclusion

In this short report, the mathematics for the *UWB* calibration using *PCA* is presented with simulated results referring to the Github repository for the Python code and experiments on the real-time platform. This report may be completed with references and experimental data in the near future, but for now, the document is simply intended to make sense of the Python implementation.

REFERENCES

[1] C. K. WilliaIns, "On a connection between kernel pca and metric multidimensional scaling," 2002.

[2] J. De Leeuw and W. J. Heiser, "Multidimensional scaling with restrictions on the configuration," *Multivariate analysis*, vol. 5, pp. 501–522, 1980.

[3] G. Gallego and A. Yezzi, "A compact formula for the derivative of a 3-d rotation in exponential coordinates," *Journal of Mathematical Imaging and Vision*, vol. 51, no. 3, pp. 378–384, 2015.