4th International Conference on Sensors Engineering and Electronics Instrumentation Advances (SEIA' 2018), 19-21 September 2018, Amsterdam, The Netherlands

Camera and IMU Frame Alignment in AR/VR Systems for Different Gyroscope Specifications

Zhanlue Zhao and Bryan Cook
CEVA Technologies, Inc., 15245 Shady Grove Road, Rockville, MD, 20850, USA
Tel.: + 1-240-386-0600
E-mail: Zhanlue.Zhao@ceva-dsp.com, Bryan.Cook@ceva-dsp.com

Summary: A common method for creating a 6 degree of freedom (DoF) device is to track a single marker with a camera to measure linear position, and use an IMU to measure angular position (orientation). This is often used for AR and VR controllers and head tracking. One key challenge is aligning the coordinate frames of the camera and the IMU, and keeping them aligned over time. This paper presents a method of frame alignment that doesn't require user intervention, and properly handles integration errors from the gyroscope as well as the very large errors encountered when performing double integration on the linear acceleration of the IMU's accelerometer to compute linear position. The paper also compares two options for solving for the alignment, either a recursive least-squares (RLS) approach or as a solution to Wahba's problem.

Keywords: AR/VR systems, Recursive Least squares, Wahba's problem, IMU, 6 DoF.

1. Introduction

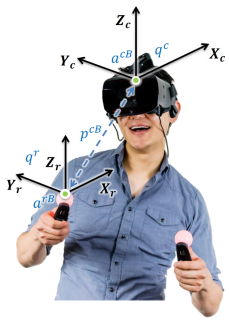
VR/AR systems are proliferating recently. In a VR/AR system with a remote that has 6 degrees of freedom, the information of the 3D linear position and 3D angular position of the remote is necessary to integrate it into the virtual world, where the remote could be portrayed as e.g., a sword or gun. A frequent solution is to estimate the remote's angular position (orientation) by an inertial measurement unit (IMU) consisting of accelerometer, gyroscope and, optionally, a magnetometer; and to estimate the remote's linear position by tracking one or more markers with a fixed camera (outside-in) or a head-mounted camera (inside-out) [1]. For this to work, the IMU measurement frame and the camera measurement frames need to be aligned. Common methods of alignment include the user performing an explicit calibration when the remotes are in a known orientation, or using a magnetometer in both the headset and the controller. When using the magnetometer, magnetic interference in the environment will often make it difficult to maintain good alignment. And if not using the magnetometer, the IMU integration errors will build up to cause the alignment to drift. To correct such misalignment, users need to frequently tare or recalibrate to realign the remote frame to the camera frame [2]. This paper describes mechanisms to keep the two frames aligned without the need of a magnetometer or user tares.

2. Problem Analysis

2.1. Sensor Frames and Sensor Measurements

In an inside-out camera system, the camera is located on the head-mounted device (HMD) [1]. It tracks the linear position of the remote through image processing. Usually, there are markers on the remote to facilitate tracking. The remote position in the camera frame is derived by the marker size and location in the image [3]. Note the camera frame and the HMD frame are assumed aligned already and used interchangeably in this paper. The IMU in the remote measures its orientation.

The sensor frames and sensor measurements are illustrated in **Fig. 1**



a^{rB}	the remote acceleration in its body frame
q^r	the remote orientation in the Earth frame
q^c	the camera orientation in the Earth frame
a^{cB}	the camera acceleration in its body frame
p^{cB}	remote linear position in camera body frame

Fig. 1. Measurements by the IMUs and camera system

The camera orientation q^c and acceleration a^{cB} provide the attitude for virtual world rendering. The remote acceleration a^{rB}, remote linear position p^{cB}, and orientation q^r provde the attitude of the remote in the virtual world. The errors in q^r and q^c due to gyro errors cause the frame misalignment.

2.2. Remote Position Measured by Camera and IMU

The position change in Earth frame measured by the IMU and camera system are:

$$p_t^{iU} = \iint q_t^r \otimes a_t^{rB}\, dtdt \tag{1}$$

$$p_t^{cU} = q_t^c \otimes p_t^{cB} - q_0^c \otimes p_0^{cB} + \iint q_t^c \otimes a_t^{cB}\, dtdt \tag{2}$$

where p_0^{cB} and q_0^c are the initial linear and angular position which are constant; $\otimes$ stands for the quaternion rotation operation which rotates the

measurements in the body frame into the Earth frame. The term $\iint q_t^c \otimes a_t^{cB}\, dtdt$ is to compensate the position changes in the camera frame caused by the head movement. Ideally, p_t^{iU} and p_t^{cU} should be the same. However, with consumer-grade MEMS sensors, performing double integration on the accelerometer typically has so much error as to be unusable due to the non-zero offset in acceleration.

2.3. Make p_t^{iU} and p_t^{cU} Comparable

To control these errors, a high-pass (DC-blocking) filter is used. One possible DC-block filter is the one-pole recursive filter specified by the difference equation:

$$y(n) = [x(n) - x(n-1)]\frac{(1+\alpha)}{2} + \alpha y(n-1) \quad (3)$$

The frequency responses with the pole a = 0.5, 0.8, 0.95 are shown in **Fig. 2**

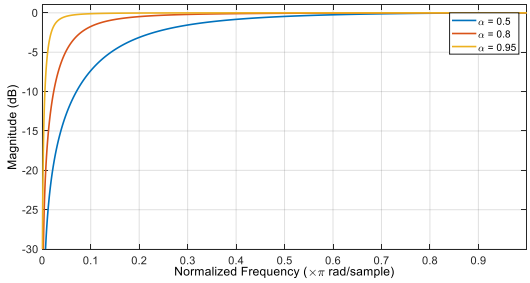


Fig. 2. DC-blocking filter frequency responses

The DC-blocking filter is applied to the accelerometer, to help cancel the effects of gravity, and after each integration operation to avoid error build-up. [5]

$$p_t^{iU\_filt} = f_{DCB}\left(\int\left(f_{DCB}\left(\int f_{DCB}(q_t^r \otimes a_t^{rB})\, dt\right)\right)dt\right) \quad (4)$$

where $f_{DCB}(*)$ where stands for the DC-blocking filter.

After applying these filters, the resulting signal is not very useful for estimating the true linear position of the device, as it only retains any change in position. However, if we apply the same filters to p_t^{cU}:

$$p_t^{cU\_filt} = f_{DCB}\left(f_{DCB}\left(f_{DCB}(p_t^{cU})\right)\right) \quad (5)$$

then we now have an equivalent estimate to the change in position, that makes p^{iU_filt} and p^{cU_filt} comparable.

3. Two Formulations to Align the Two Frames

Now that there are comparable signals, we can formulate a solution for the frame alignment:

$$p_k^{cU\_filt} = p_k^{iU\_filt} A + v_k \quad (6)$$

where A is an arbitrary 3x3 matrix when solving with a RLS formulation, but is constrained to be an orthogonal 3x3 rotation matrix when formulated as a Wahba problem, and v_k is a noise term.

The solutions to the above formulations can be found at [6] and [8]. The Wahba's problem solution directly provides the rotation matrix to align the two frames. For RLS solution, once A is obtained, the result can either be applied directly, or alternatively can be reduced to just a rotation matrix by performing a matrix polar decomposition. The detailed solution are provided in the following subsections.

3.1. Solution to Wahba's Problem

To align $p_k^{iU\_filt}$ with $p_k^{cU\_filt}$ using only a rotation transformation, a Wahba's problem can be formulated that seeks the rotation matrix A to minimize the cost function

$$J(A) = \sum_{k=1}^{N} \left\| p_k^{cU\_filt} - p_k^{iU\_filt} A \right\|^2 \quad (7)$$

where A is a rotation matrix to align $p_k^{iU\_filt}$ with $p_k^{cU\_filt}$ of N samples measured by IMU and camera.

For the Wahba problem formulation, the solution can be found using a singular decomposition as described in [6]:

1) Compute a matrix Φ as follows:

$$\Phi = \sum_{k=1}^{N} a_k p_k^{cU_{filt}} \left(p_k^{iU_{filt}}\right)^T \quad (8)$$

where a_k are the weights for the column vectors $p_k^{cU\_filt}$ and $p_k^{iU\_filt}$ at time k; $(\cdot)^T$ is the vector transpose.

2) Obtain the singular value decomposition of Φ

$$\Phi = SVD^T \quad (9)$$

3) The rotation matrix is

$$A = SMD^T \quad (10)$$

where $M = diag([1 \quad 1 \quad det(S)det(D)])$, $det(S)$is the determinant of the matrix S, and A is the rotation matrix to align the two frames.

3.2. Recursive Least Squares Solution (RLS)

3.2.1. Weighted RLS

Alternatively, the standard least squares solution is

$$\hat{A} = arg\min_A \sum_{k=1}^{N} \left\| p_k^{cU\_filt} - p_k^{iU\_filt} A \right\|^2 \quad (11)$$

A can be found by the RLS method. In this application, a weighted RLS serves the purpose better which gives less weights to the older measurements, i.e.,

$$\hat{A} = arg\min_A \sum_{k=1}^{N} \lambda^{N-k} \left\| p_k^{cU\_filt} - p_k^{iU\_filt} A \right\|^2 \quad (12)$$

where λ is a forget factor and $0 < \lambda < 1$.

To simplify the notations, let $z_k = p_k^{cU\_filt}$ and $x_k = p_k^{iU\_filt}$. The weighted RLS solution for A at kth observation is (see Appendix for derivation):

$$K_k = \frac{\lambda^{-1} P_{k-1} x_k'}{1 + \lambda^{-1} x_k P_{k-1} x_k'} \quad (13)$$

$$\mathrm{P_k} = \lambda^{-1}\mathrm{P_{k-1}} - \lambda^{-1}\mathrm{K_k x_k P_{k-1}} \quad (14)$$

$$\hat{A}_k = \hat{A}_{k-1} - P_k x_k{}' x_k \hat{A}_{k-1} + P_k x_k{}' z_k \quad (15)$$

When initializing the algorithm, $\hat{A}_0$ is initialized as zero vector or matrix; P_0 is initialized as a diagonal matrix δI. I is the identity matrix. δ is a large positive value.

The fitting error Ψ_k can be recursively computed as

$$\Psi_k = \frac{\lambda(k-1)\Psi_{k-1} + \left(z_k - x_k \hat{A}_{k-1}\right)\left(z_k - x_k \hat{A}_{k-1}\right)'}{k} \quad (16)$$

3.2.2. Matrix Polar Decomposition

Once A is obtained, the rotation matrix can be derived by the matrix polar decomposition [7]. The polar decomposition of a square matrix is a matrix decomposition of the form

$$A = \Sigma P \quad (17)$$

where $\mathbf{\Sigma}$ is a rotation matrix and P is a positive-semidefinite symmetric matrix. To compute the polar decomposition, assume that the singular value decomposition of A is

$$A = UDV' \quad (18)$$

Then

$$\Sigma = UV' \quad (19)$$

$$P = VDV' \quad (20)$$

The rotation matrix Σ can be used to correct the misalignment of the remote orientation. The P matrix scales the space along the orthogonal axes spanned by V's column vectors.

If P is quite different from the Identify matrix, it could indicate the two measurements in the two frames are too different to be aligned well by rotation only.

4. Two Solution Comparison

Comparing the linear model fitting formulation with Wahba's problem formulation, the former takes into account not only the frame misalignment but also the scale difference between the two measurements. When the scale matrix is close to an identity matrix, the computed rotation matrix will be more credible. This provides a good indicator for assessing the reliability of the computed rotation matrix.

To facilitate comparing the performance of these two methods, the following simulated scenarios are designed. The measurements from a camera system are used. The same measurements are manually transformed to the measurements in remote frame from the IMU. Meanwhile different noise profiles are added to each scenario. If we assume the measurements from the camera are X, then the corresponding IMU measurements are

$$Y = XA + v \quad (21)$$

where v is the noise with a Gaussian distribution $N(0, \Omega)$. Ω is the covariance of the noise. Here X is from a camera system as shown in **Fig. 3**.

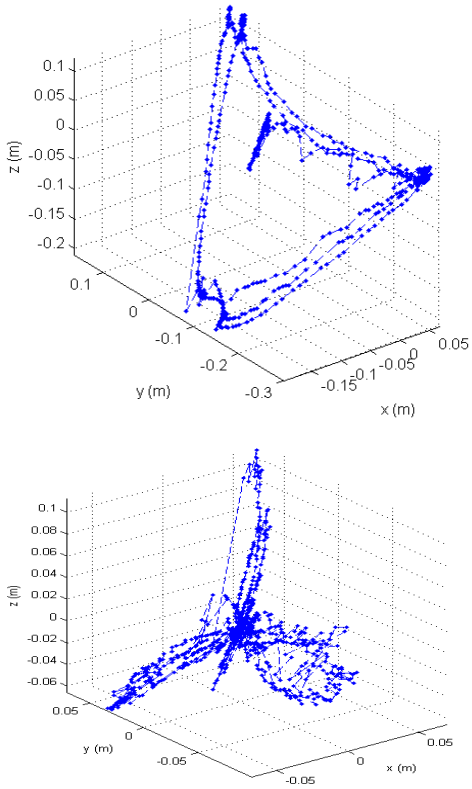


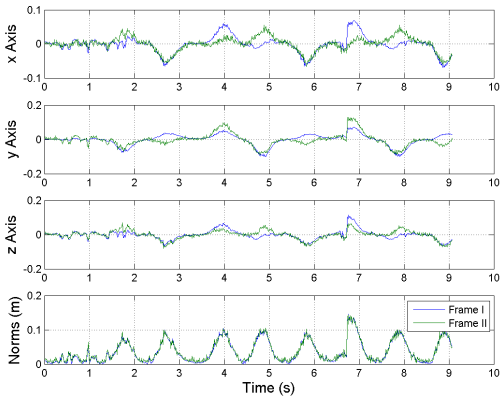
Fig. 3. The trajectory before and after DC-Blocking filter in Camera Frame

Provided X and Y, the rotation matrix will be estimated by the above two methods and their performance will be compared in the following four cases. In each case, first, the difference between X and Y is illustrated; second, the computed Euler angle and the truth are compared; third, the difference between Y and the rotated X is illustrated, which shows how well the rotation can align the measurements. The closer the computed Euler angle to the truth, the better the algorithm performs. The faster the computed Euler angle converges to the truth, the better the algorithm performs.

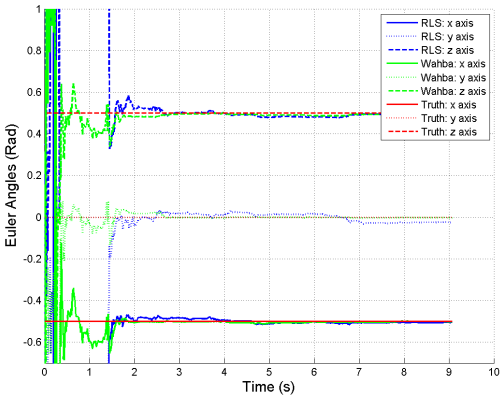
Case 1: A is a rotation matrix. Noise level Ω is relatively small. Let A = euler2dcm([-0.5,0,0.5]), where euler2dcm(*) converts rotation angles of yaw, pitch, roll to a direction cosine matrix. Ω = I*0.005. I is an Identity matrix.

For case 1, as shown in **Fig. 4**, these two methods converge to the true Euler angles at similar speeds. If the difference in measurements from two sensor systems is mainly due to frame alignment and the noise level is low, both methods worked well and

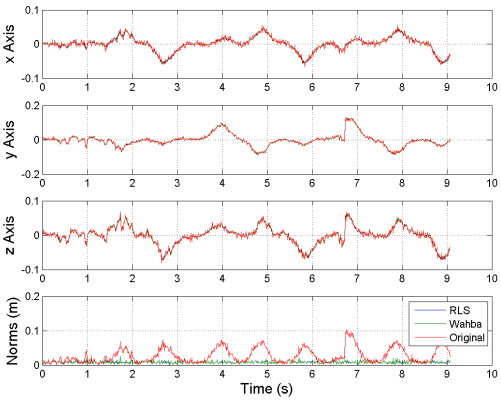
demonstrated similar performance.



(a) Measurements Before Alignment



(b) Computed Euler Angles and the True Values

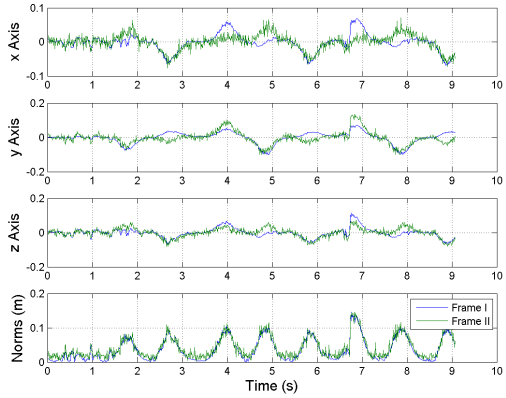


(c) Measurements After Alignment

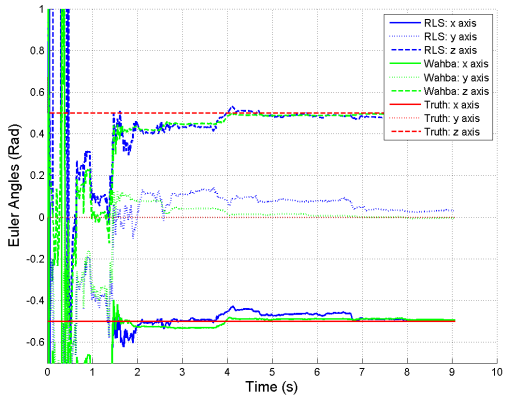
Fig. 4. Comparison in Case 1

Case 2: A is a rotation matrix. Noise level Ω is relatively large. Let $A = euler2dcm([-0.5,0,0.5])$, $\Omega =$ I*0.01.

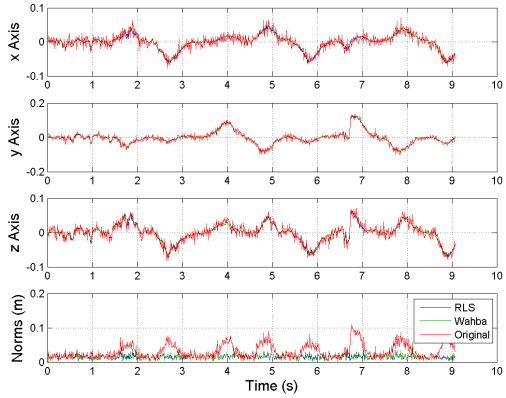
For case 2, as is shown in **Fig. 5,** the solution to Wahba's problem converges much faster and the computed Euler angles are closer to the true values. If the measurements from two sensor systems are different due to frame misalignment and measurement noise, the solution to Wahba's problem is more robust and converges faster.



(a) Measurements Before Alignment



(b) Computed Euler Angles and the True Values

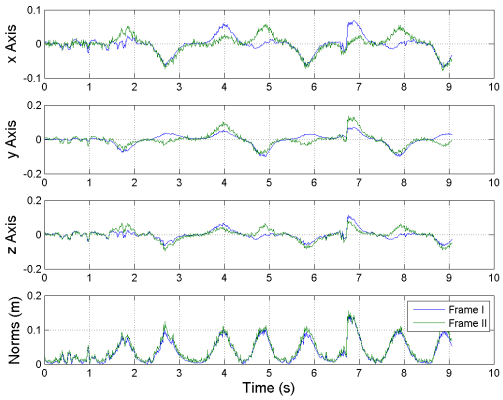


(c) Measurements After Alignment

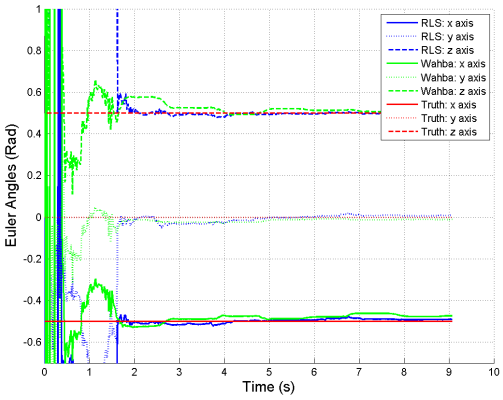
Fig. 5. Comparison in Case 2

Case 3: A is a rotation matrix multiplied by a scale matrix. Noise level Ω is relatively small. Let $A = euler2dcm([-0.5,0,0.5]) * diag([1.2\ 1\ 1.2])$, where $diag(*)$ converts the vector into a diagonal matrix. $\Omega = I * 0.005$

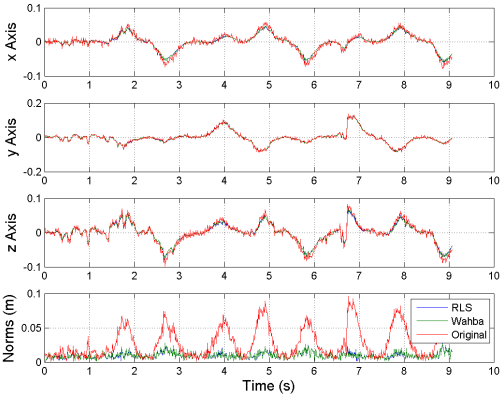
For case 3, as is shown in **Fig. 6**, the RLS method converges much faster and the computed Euler angles are closer to the true values. If two measurements from two sensor systems are different due to frame misalignment and scale mismatch with relatively small measurement noise, RLS method is more robust and converges faster.



(a) Measurements Before Alignment

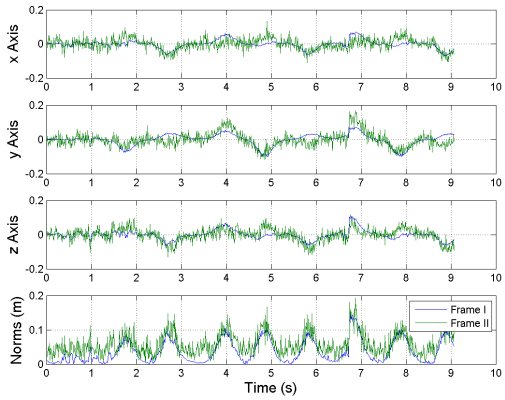


(b) Computed Euler Angles and the True Values

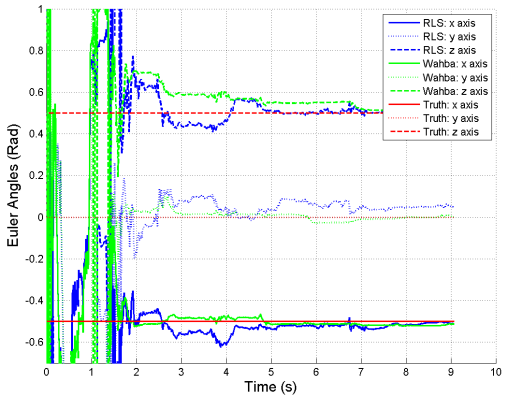


(c) Measurements After Alignment

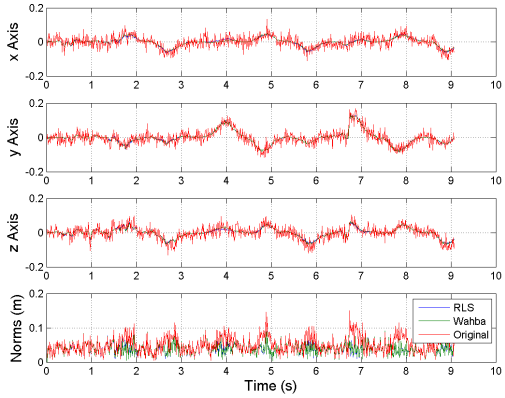
Fig. 6. Comparison in Case 3



(a) Measurements Before Alignment



(b) Computed Euler Angles and the True Values



(c) Measurements After Alignment

Fig. 7. Comparison in Case 4

Case 4: A is a rotation matrix multiplied by a scale matrix. $\boldsymbol{\Omega}$ is relatively large. $\boldsymbol{\Omega = I * 0.025}$. Let A = euler2dcm([-0.5,0,0.5])*diag([1.2 1 1.2]).

When the scale difference exists and measurement noise is large, the relative performance of the two methods will depend on which one is more dominant. For case 4, as is shown in **Fig. 7** the noise level is dominant so the solution to Wahba's problem converges a little bit faster.

5. Conclusions

This paper shows a method for aligning a camera and an IMU even when there are very large integration errors. When the measurement noise is low, both RLS and the solution to Wahba's problem perform well. When the gyro scale error is prominent, the RLS method is preferred. Otherwise, the solution to Wahba's problem should be used.

Appendix

The recursive least squares with matrix parameter *A* is derived as follows, which is very similar to the derivation in [8] except where *A* is a vector.

Assume the measurement model is

$$z = xA + v \tag{22}$$

where A is a n × m matrix, x is 1× m vector. $\boldsymbol{v}$ is the noise. The standard least squares solution that minimizes the cost function

$$[Z_n - X_n A]'[Z_n - X_n A] \tag{23}$$

is

$$\hat{A}_n = (X_n{}'X_n)^{-1}X_n{}'Z_n \tag{24}$$

where

$$X_n = \begin{bmatrix} x_1 \\ \vdots \\ x_{n-1} \\ x_n \end{bmatrix}, Z_n = \begin{bmatrix} z_1 \\ \vdots \\ z_{n-1} \\ z_n \end{bmatrix} \tag{25}$$

denoting

$$\begin{aligned} \Phi_n &= X_n{}'X_n \\ \psi_n &= X_n{}'Z_n \end{aligned} \tag{26}$$

Then

$$\begin{aligned} \Phi_n &= [X_{n-1}{}' \quad x_n{}'] \begin{bmatrix} X_{n-1} \\ x_n \end{bmatrix} = \Phi_{n-1} + x_n{}'x_n \\ \psi_n &= [X_{n-1}{}' \quad x_n{}'] \begin{bmatrix} Z_{n-1} \\ z_n \end{bmatrix} = \psi_{n-1} + x_n{}'z_n \end{aligned} \tag{27}$$

Equation (24) becomes

$$\hat{A}_n = \Phi_n^{-1}\psi_n \tag{28}$$

The matrix inversion formula: if B and C are m × m positive definite matrices, D is a n × n matrix, and E is a m × n matrix which are related by

$$B = C^{-1} + ED^{-1}E' \tag{29}$$

Then

$$B^{-1} = C - CE(D + E'CE)^{-1}E'C \tag{30}$$

Apply the matrix inversion formula to (27), we obtain

$$\Phi_n^{-1} = \Phi_{n-1}^{-1} - \frac{\Phi_{n-1}^{-1}x_n{}'x_n\Phi_{n-1}^{-1}}{1 + x_n\Phi_{n-1}^{-1}x_n{}'} \tag{31}$$

Denoting

$$P_n = \Phi_n^{-1} \tag{32}$$

and

$$K_n = P_{n-1}x_n'/(1 + x_nP_{n-1}x_n{}') \tag{33}$$

From (31), (32) and (33) we obtain

$$P_n = P_{n-1} - K_nx_nP_{n-1} \tag{34}$$

From (34), we obtain

$$\begin{aligned} K_n(1 + x_nP_{n-1}x_n') &= P_{n-1}x_n' \\ K_n &= P_{n-1}x_n' - K_nx_nP_{n-1}x_n' \end{aligned} \tag{35}$$

From (34), by the right multiplication of x_n', we obtain

$$P_nx_n{}' = P_{n-1}x_n' - K_nx_nP_{n-1}x_n{}' \tag{36}$$

Thus

$$K_n = P_nx_n{}' \tag{37}$$

Then the recursive solution of (30) can be derived as

$$\begin{aligned} \hat{A}_n &= \Phi_n^{-1}\psi_n && (38)\\ &= P_n\psi_n && (39)\\ &= P_n(\psi_{n-1} + x_n{}'z_n) && (40)\\ &= P_n(\Phi_{n-1}\hat{A}_{n-1} + x_n{}'z_n) && (41)\\ &= \hat{A}_{n-1} - P_nx_n{}'x_n\hat{A}_{n-1} + P_nx_n{}'z_n && (42)\\ &= \hat{A}_{n-1} + P_nx_n'\left(z_n - x_n\hat{A}_{n-1}\right) && (43)\\ &= \hat{A}_{n-1} + K_n\left(z_n - x_n\hat{A}_{n-1}\right) && (44) \end{aligned}$$

The fitting error Ψ_k can be computed:

$$\Psi_k = \frac{(k-1)\Psi_{k-1}}{k} + \frac{(z_k - x_k\hat{A}_{k-1})(z_k - x_k\hat{A}_{k-1})'}{k} \tag{45}$$

Note: this solution works if A is a matrix or vector.

For the exponentially weighted RLS, the older measurements are weighted less, i.e.,

$$\begin{aligned} \Phi_n &= \lambda\Phi_{n-1} + x_n{}'x_n && (46)\\ \psi_n &= \lambda\psi_{n-1} + x_n{}'z_n && (47) \end{aligned}$$

where λ ($0 < \lambda < 1$) is the forget factor. The least squares solution is

$$\hat{A}_n = \Phi_n^{-1}\psi_n \tag{48}$$

Following the same procedure, the corresponding terms can be found as below:

$$\begin{aligned} K_n &= \frac{\lambda^{-1}P_{n-1}x_n'}{1+\lambda^{-1}x_kP_{n-1}x_n'} && (49)\\ P_n &= \lambda^{-1}P_{n-1} - \lambda^{-1}K_nx_nP_{n-1} && (50)\\ \hat{A}_n &= \hat{A}_{n-1} - P_nx_n{}'x_n\hat{A}_{n-1} + P_nx_n{}'z_n && (51) \end{aligned}$$

The fitting error Ψ_n can be recursively computed as

$$\Psi_n = \frac{\lambda(n-1)\Psi_{n-1} + (z_n - x_n\hat{A}_{n-1})(z_n - x_n\hat{A}_{n-1})'}{n} \tag{52}$$

Acknowledgements

The authors would like to thank Ravikumar Pragada and Dr. Chuck Gritton for their input.

References

[1]. H. Langley, Inside-out v Outside-in: How VR tracking works and how it's going to change, (https://www.wareable.com/vr/inside-out-vs-outside-in-vr-tracking-343), May 2017.

[2]. J. Karner, How to fix drift problems with Daydream View, (https://www.vrheads.com/how-fix-drift-problems-daydream-view), Nov 2017.

[3]. G. Korak und G. Kucera, Optical Tracking System, *International Journal of Electronics and Telecommunications*, vol. 61, no. 2, 2015, pp. 165–170.

[4]. Y. K. Thong, M. S. Woolfson, J. A. Crowe, B. R. Hayes-Gill, and D. A. Jones, Numerical double integration of acceleration measurements in noise, *Measurement*, vol. 36, no. 1, 2004, pp. 73–92.

[5]. H. B. Gilbert, O. Ozkan, M. K. Malley. Long-term double integration of acceleration for position sensing and frequency domain system identification. *International Conference on Advanced Intelligent Mechatronics*, Montréal, Canada, 2010.

[6]. F. L. Markley, D. Mortari, Quaternion Attitude Estimation Using Vector Observations, *Journal of the Astronautical Sciences,* 2000, 48(2):359-380.

[7]. G. Buffington, Polar Decomposition of a Matrix, (http://buzzard.ups.edu/courses/2014spring/420projects/math420-UPS-spring-2014-buffington-polar-decomposition.pdf).

[8]. I. Tabus, Recursive Least Squares Estimation. (http://www.cs.tut.fi/~tabus/course/ASP/LectureNew10.pdf).