PNI APPLICATION NOTE:

TRAX2 Absolute Rotation Tracking

Table of Contents

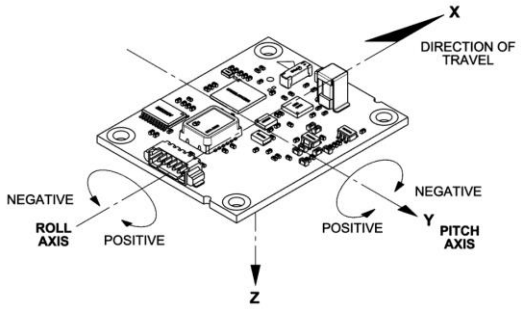
1 PURPOSE

It is sometimes desirable to track heading, pitch, and roll changes beyond common 360, +/-90, and +/-180 extremes implicit in these measurements. This document will outline the math and code needed to create a measurement that uses the quaternion output from an TRAX2 unit and produces the total rotation about the body x, y, and z axes.

2 DEFINITIONS

2.1 FRAME DEFINITIONS

The sensor body frames of a TRAX2 unit are as follows.



Note that this is for the standard (North East Down) configuration. Changing body frame configuration will change the frame definitions. This general orientation setup will apply to any given TRAX2 board once the front is known.

2.2 QUATERNION DEFINITIONS

PNI quaternions are defined as follows.

$$q = \begin{bmatrix} q(1) \\ q(2) \\ q(3) \\ q(4) \end{bmatrix} = \begin{bmatrix} q_x \\ q_y \\ q_z \\ q_w \end{bmatrix}$$

x, y, and z are the vector components and w is the scalar component helping to denote direction. The vector components are the important parts of the quaternion, containing the majority of the information, while the scalar point primarily exists to resolve singularity issues. A quaternion describes the relationship between two frames, in this case a body frame (from above) and an inertial frame (for PNI this is the north-east-down frame). The quaternions output from TRAX2 units describe the rotation from the body frame to the inertial frame. A quaternion inverse would describe the opposite rotation, moving from the inertial to body frame. The definition of quaternion inverse is:

$$q^{-1} = \begin{bmatrix} -q(1) \\ -q(2) \\ -q(3) \\ q(4) \end{bmatrix} = \begin{bmatrix} -q_x \\ -q_y \\ -q_z \\ q_w \end{bmatrix}$$

Similarly, quaternion multiplication can be defined to describe the effect of performing one rotation and then a second rotation. The definition of quaternion multiplication (matrixwise) is as follows.

$$q_1 \otimes q_2 = \begin{bmatrix} q_1(4) & q_1(3) & -q_1(2) & q_1(1) \\ -q_1(3) & q_1(4) & q_1(1) & q_1(2) \\ q_1(2) & -q_1(1) & q_1(4) & q_1(3) \\ -q_1(1) & -q_1(2) & -q_1(3) & q_1(4) \end{bmatrix} \begin{bmatrix} q_2(1) \\ q_2(2) \\ q_2(3) \\ q_2(4) \end{bmatrix}$$

Similarly, by using these two combinations, we can effectively create a quaternion "subtraction" to determine the difference between two quaternions. The angular difference between two quaternions can be defined as follows.

$$q_1 - q_2 = \delta q = q_1 \otimes q_2^{-1}$$

3 ABSOLUTE ANGLE CALCULATIONS

A propagation quaternion *qp* can be defined which updates the attitude quaternion *q(t)* from the previous *q(t-Δt)* based on the rate sensor inputs:

$$q(t) = \begin{bmatrix} q1 \\ q2 \\ q3 \\ q4 \end{bmatrix}(t) = qp \otimes q(t - \Delta t) = \begin{bmatrix} qp1 \\ qp2 \\ qp3 \\ qp4 \end{bmatrix} \otimes \begin{bmatrix} q1 \\ q2 \\ q3 \\ q4 \end{bmatrix}(t - \Delta t)$$

A prolonged differential equation solution would eventually lead to the following equation.

$$\Delta = \begin{pmatrix} \delta x \\ \delta y \\ \delta z \\ \delta^2 \end{pmatrix} = \begin{pmatrix} \omega x \cdot \Delta t/2 \\ \omega y \cdot \Delta t/2 \\ \omega z \cdot \Delta t/2 \\ \left(\omega x^2 + \omega y^2 + \omega z^2\right) \cdot (\Delta t/2)^2 \end{pmatrix}$$

ω represents the angular rate of the system, and if integrated this would just lead to the delta-angle vector. We are just looking to solve the right side of the equation, and are only interested in the first three values. Therefore from a set of sequential quaternions we find δq and from δq we can find the total angle traversed over a Δt.

$$\theta_{Total} = \sum 2\delta q$$

Or piecewise at point k

$$\delta q_k = q_k \otimes q_{k-1}^{-1}$$

$$\theta_k = \theta_{k-1} + 2\delta q_k$$

4 IMPLEMENTATION AND SAMPLE CODE

4.1 SAMPLE CODE

How this is specifically implemented in PNI's code interface is shown below. Note that due to internal quaternion math some values might be inverted from the standard implementation presented above. When using quaternion output from TRAX2 units, use of the code below is recommended.

Main function:

```
function [sumxyz, prevq] = q2qunroll(q, prevq, sumxyz)
% function to compute the total angle rotated about each body axis.
% inputs/outputs:
% q: current quaternion
% prevq: the quaternion at previous function call
% sumxyz: current sum of angles, user should reset to [0;0;0] when
appropriate. Units in radians.

dq = qmult_eml(qinv(q),prevq)'; % subtract current quaternion from
previous point's quaternion
```

```
propw = -2*dq(1:3); % generated a propogation angle based on small angle
approximation
sumxyz = sumxyz + propw; % sum the angles about x, y, z
prevq = q; % save the current quaternion, now the "previous" quaternion
```

Math functions:

```
function qo = qinv(qi)

eml.inline('never');

qo = qi;
qo(1:3) = -qo(1:3);
end


function q_out=qmult_eml(q1,q2)

q_out = [q1(1) q1(2) q1(3) q1(4)]*...
      [q2(4) -q2(3)  q2(2) -q2(1)
      q2(3)  q2(4) -q2(1) -q2(2)
      -q2(2)  q2(1)  q2(4) -q2(3)
      q2(1)  q2(2)  q2(3)  q2(4)];
end
```

4.2 NOTABLE ASSUMPTIONS

Any quaternion set can be used for these types of calculations. However, this assumes a **small angle approximation**. Therefore, it is highly recommended that quaternion outputs occur at a rate of at least 10hz; the faster the better. It is also recommended that the system not be rotated at rates higher than 50dps. Slower update rates or faster motions are more likely to cause motion blur or random walk, that could over longer periods result in errors in this calculation.

Because yaw/pitch/roll is a 3-2-1 Euler angle definition, the first angle output from the code above represents roll, the second pitch, and the third yaw. This is different than the hpr output from TRAX2 systems.

If it is desired to use this measurement to detect if a system has rotated too far, simply observe when the system has passed through a certain boundary, say +/-720, and then perform corrections from there. Resetting sumxyz above will effectively reset the reference of the system and should be used on startup.