Lowell Neal

Oct 14th 2012

***Metal Detector Magic* - Explained**

I’m mostly reminding myself in writing this, but if it is useful to Jeff also, then all the better.

The purpose of metal detector magic is to create a circle of deadband rather than a square of deadband. The confusing part of the code comes as a result of dealing with the 0-360 wrap effect of headings.

Figure 1 provides an example case to demonstrate the purpose and necessity of the code.

In this example the target is 10degrees above, at a bearing of 350. This is represented by the red dot.

Please note, this figure is not created to scale.

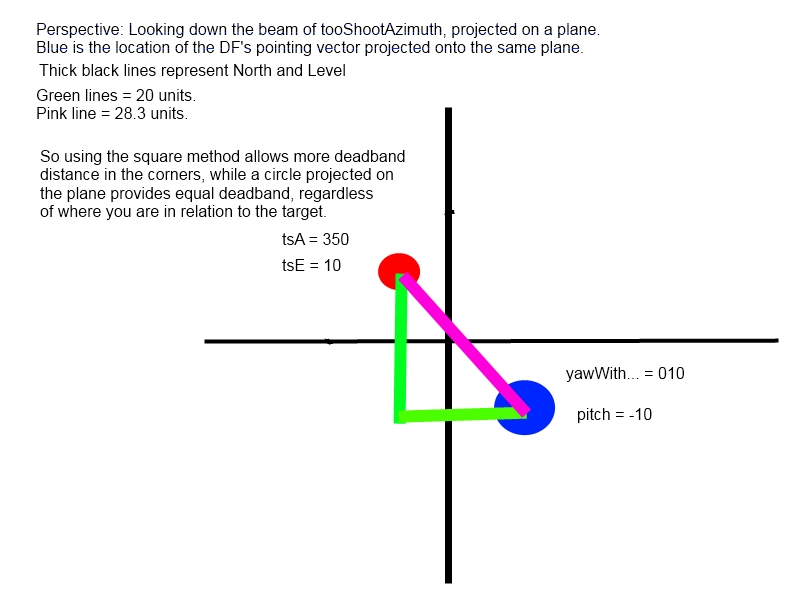


Figure 1 (not to scale): An example case for code explination

The pink leg is 28.2 units, found using equation1 (some think Pythagoras wrote it a^2 + b^2 = c^2, but I’m pretty sure he was using different colored green sticks, so this is closer to the original).

Equation 2 is the code performing the same operation as shown in equation 1.

Now keep reading, and we’ll work backwards to explain all of the code.

eq(1)

*Pink = sqrt(VerticalGreen^2 + HorizontalGreen^2)*

eq(2)

*double magnitudeOfDeltas\_mrad = System.Math.Pow(azimuthDelta\_mrad \* azimuthDelta\_mrad + elevationDelta\_mrad \* elevationDelta\_mrad, 0.5);*

To feed equation2, we must calculate azimuthDelta\_mrad and elevationDelta\_mrad. However, note that azimuth is reported in 0 to 360 deg, while elevation is reported in -90 to 90 deg.

Calculating elevationDelta is straightforward.

eq(3)

double elevationDelta\_mrad = toShootElevationAngle\_rad \* 1000 - positionMe.Pitch\_mrad;

Regardless of which value is above the other, the magnitude of the answer to equation 3 will yield the actual magnitude of the delta between the two vectors. There is no concern for the sign of the answer to equation 3 because elevationDelta will be squared when it is put into equation 2.

Calculating azimuthDelta is not as simple. While both vectors are within 180 degrees of each other moving clockwise around the circle, it is as simple as subtracting.

Example: tsA = 350, and yaw = 180. Subtracting provides either 170, or -170, both of which are true, and would yield accurate results in equation 2 for this example.

However, in the case of 350 and 10, this method would suggest an answer of 340, which is not possible since 180deg is the maximum delta for two vectors in a plane.

Thus, we must be more clever. **Instead of only subtracting the two values, “subtract” going one way around the circle, and the subtract going the otherway around the circle, and take the minimum of the two.** This is the purpose of equation 4.

eq(4)

double azimuthDelta\_mrad = exMath.Min((2 \* exMath.PI \* 1000 - maxOfAzimuths) + minOfAzimuths, maxOfAzimuths - minOfAzimuths);

The second option for the exMath.Min function to select from, is:

*maxAzimuth – minAzimuth*

For our example it is 350 – 10 = 340. This is subtracting from 350 -> 10, counterclockwise around the circle.

The first option for the exMath.Min function to select from is:

*360 – maxAzimuth + minAzimuth.*

For our example this is 360 – 350 + 10 = 10 + 10 = 20.

The 360 – max measures the angle from North to max.

This value is then added to the Min, which is just a measure of North to that angle itself.

That concludes, Metal detector magic explained… hopefully future Lowell and Jeff won’t be confused by this code after reading that explanation… but who knows.