1 Szenario

Es soll behandelt werden, wie die Lageregelung eines Objektes mit Hilfe von Gyroskopen bewerkstelligt werden kann. Gyroskope(Gyros) messen in jedem Schritt die Änderung der Winkelgeschwindigkeit ω. Es ist ein leichtes zu behaupten, dass die Orientierung in Form von Yaw, Pitch und Roll Winkeln wie folgt errechnet werden kann:

$$Roll = \int \omega_x dt \approx \sum \omega_x \cdot \Delta t$$
$$Pitch = \int \omega_y dt \approx \sum \omega_y \cdot \Delta t$$
$$Yaw = \int \omega_z dt \approx \sum \omega_z \cdot \Delta t$$

Dies wäre viel zu ungenau, würde es in der Praxis eingesetzt werden.

2 Idee

Um ein einigermaßen akzeptables Ergebnis zu erhalten, geht man wie folgt vor:

1. Initialisierung durch ein Lagequaternion q=[1,0,0,0]
2. Auslesen der Sensorwerte x,y,z
3. Errechnen des Quaternions aus x,y,z
4. Anpassung des Lagequaternions
5. Ermitteln von Yaw,Pitch,Roll aus dem angepassten Quaternion
6. Ermitteln der Lageänderung

3 Implementierung

Es soll hier ein Zyklus gezeigt werden, natürlich muss dies in der Realität durchgehend ausgeführt werden, um ein genaues Ergebnis zu erhalten.

```
Quaternion Lage(1,0,0,0);  // step 1
double yaw, pitch,roll =0;
double x,double y, double z;  // step 2
YPR y(x,y,z);
Quaternion deltaLage =y.toQuaternion().normalize();
// step 3
//normalisiertes Quaternion == Rotation
Quaternion aktLage =Lage.qMult(deltaLage).normalize();
// step 4
y = aktLage.toYPR();  // step 5
double deltaYaw =(y.yaw - yaw)*1000/SAMPLETIME;
double deltaPitch =(y.pitch - pitch)*1000/SAMPLETIME;
double deltaRoll = (y.roll - roll)*1000/SAMPLETIME;
//Step 6
//Aktuelle Werte für nächsten durchlauf Speichern
yaw =y.yaw;
pitch = y.pitch;
roll = y.roll;
// Quaternion updaten
Lage = aktLage;
```

Es muss hier noch gesagt werden, dass die Einheitenumrechnung, die hier in Step 6 passiert natürlich an das verwendete Model angepasst werden muss. Des weiteren stellt sich beim Gyroskop ein Drift ein, der im Idealfall durch einen KalmanFilter reduziert werden sollte.