Got angle and angular velocity(Complementary filter algorithm)

The tutorial introduces the use of complementary filtering algorithm to calculate the attitude angle.

```
The MPU6050 module can detect three-axis acceleration, three-axis gyroscope
motion data and temperature data.
```

Complementary filtering algorithm

The complementary filtering algorithm adds two highly complementary signals in a certain weighted ratio to achieve the effect of filtering and fusion.

By combining the data of the accelerometer and gyroscope, the complementary filtering algorithm can make a more accurate estimate of the attitude over the entire frequency range. The low-frequency compensation of the accelerometer can effectively eliminate the integral error of the gyroscope, and the high-frequency filtering of the gyroscope can eliminate the fluctuation of the accelerometer, thereby improving the accuracy and stability of the attitude estimation.

Implementation ideas

Accelerometers are mainly used for compensation of low-frequency signals because their output is relatively stable in the long run, but they are easily affected by high-frequency noise; gyroscopes are mainly used for filtering high-frequency signals because their output is relatively accurate in a short time, but there are problems such as integral drift.

Comparison item	Accelerometer	Gyroscope
High-frequency vibration noise	Sensitive	Insensitive
Low-frequency attitude drift	No drift	Will drift

Comparison item	Accelerometer	Gyroscope
High-frequency interference resistance	No	Yes
Low-frequency interference resistance	Yes	No

> **Implementation formula**

$$angle = K_1 * angle + (1 - K_1) * (angle + gyro_m * d_t)$$

Implementation code

```
float Complementary_Filter_x(float angle_m, float gyro_m)
{
static float angle;
float K1 =0.02;
angle = K1 * angle_m+ (1-K1) * (angle + gyro_m * dt);
return angle;
}

float Complementary_Filter_y(float angle_m, float gyro_m)
{
static float angle;
float K1 =0.02;
angle = K1 * angle_m+ (1-K1) * (angle + gyro_m * dt);
return angle;
}
```

Software Code

Balance Car PID Control Basics: 08-13 Tutorial only provides one project file.

```
Product Supporting Materials Source Code Path: Attachment → Source Code Summary →
3.PID_Course → 08-13.Balanced_Car_PID
```