*※보고서 제목 양식 예 : 이름\_20240403*

|  |
| --- |
| **2024 ALTIS SW 개인 활동 보고서** |

|  |  |
| --- | --- |
| **활동 개요** | 가속도 자이로 값으로 각도 추정하는 방법 |
| **일시** | 2024년 04월 13일 15시 |
| **작성자** | 문시경 |
| **활동 내용** | 가속도 자이로 값을 활용하여 각도를 추정하는 방법에 대하여 알아보고 사용가능한 방법을 찾아 실험해본다.  찾아본 방법들과 선택 : 칼만 필터와 상보 필터를 활용하는 방법, 자이로 값으로부터 각속도를 적분하여 각도를 추정하는 방법, 가속도 센서의 측정값에서 중력 가속도를 제거하여 기기의 자세를 추정하는 방법 등을 찾아보았고, 이들 중 칼만 필터를 활용하는 방법을 선택하여 실험해 보기로 한다.  선택 이유 : 칼만 필터는 시스템의 상태를 추정할 때 과거와 현재의 측정값을 결합하여 보다 정확한 추정이 가능하며, 측정 오차나 외부 요인에도 덜 민감하여 변수의 발생에도 유용한 대처가 가능하다. 또한 실시간으로 데이터를 처리하고 필터링할 수 있어 위성이나 로켓에 사용한다고 가정할 때 매우 유용할 것이라 판단하여 선택하게 되었다.  사용코드 :  #include<Wire.h>  #definempu\_add0x68//mpu6050 address  classkalman {  public :  doublegetkalman(doubleacc, doublegyro, doubledt){  //project the state ahead  angle += dt \* (gyro - bias);  //Project the error covariance ahead  P[0][0] += dt \* (dt \* P[1][1] - P[0][1] - P[1][0] + Q\_angle);  P[0][1] -= dt \* P[1][1] ;  P[1][0] -= dt \* P[1][1] ;  P[1][1] += Q\_gyro \* dt ;  //Compute the Kalman gain  doubleS = P[0][0] + R\_measure ;  K[0] = P[0][0] / S ;  K[1] = P[1][0] / S ;  //Update estimate with measurement z  doubley = acc - angle ;  angle += K[0] \* y ;  bias += K[1] \* y ;  //Update the error covariance  doubleP\_temp[2] = {P[0][0], P[0][1]};  P[0][0] -= K[0] \* P\_temp[0] ;  P[0][1] -= K[0] \* P\_temp[1] ;  P[1][0] -= K[1] \* P\_temp[0] ;  P[1][1] -= K[1] \* P\_temp[1] ;  returnangle ;  };  voidinit(doubleangle, doublegyro, doublemeasure){  Q\_angle = angle ;  Q\_gyro = gyro ;  R\_measure = measure ;  angle = 0;  bias = 0;  P[0][0] = 0;  P[0][1] = 0;  P[1][0] = 0;  P[1][1] = 0;  };  doublegetvar(intnum){  switch(num){  case0:  returnQ\_angle ;  break;  case1:  returnQ\_gyro ;  break;  case2:  returnR\_measure ;  break;  }  };  private :  doubleQ\_angle, Q\_gyro, R\_measure ;  doubleangle, bias ;  doubleP[2][2], K[2] ;  };  kalman kal ;  longac\_x, ac\_y, ac\_z, gy\_x, gy\_y, gy\_z ;  doubledeg, dgy\_y ;  doubledt ;  uint32\_tpasttime ;  voidsetup(){  // put your setup code here, to run once:  Serial.begin(9600);  Wire.begin();  Wire.beginTransmission(mpu\_add);  Wire.write(0x6B);  Wire.write(0);  Wire.endTransmission(true);  kal.init(0.001, 0.003, 0.03);//init kalman filter  Serial.println();  Serial.print("parameter");  Serial.print("\t");  Serial.print(kal.getvar(0), 4);  Serial.print("\t");  Serial.print(kal.getvar(1), 4);  Serial.print("\t");  Serial.println(kal.getvar(2), 4);  }  voidloop(){  // put your main code here, to run repeatedly:  Wire.beginTransmission(mpu\_add);//get acc data  Wire.write(0x3B);  Wire.endTransmission(false);  Wire.requestFrom(mpu\_add, 6, true);  ac\_x = Wire.read()<< 8| Wire.read();  ac\_y = Wire.read()<< 8| Wire.read();  ac\_z = Wire.read()<< 8| Wire.read();  Wire.beginTransmission(mpu\_add);//get gyro data  Wire.write(0x43);  Wire.endTransmission(false);  Wire.requestFrom(mpu\_add, 6, true);  gy\_x = Wire.read()<< 8| Wire.read();  gy\_y = Wire.read()<< 8| Wire.read();  gy\_z = Wire.read()<< 8| Wire.read();  deg = atan2(ac\_x, ac\_z)\* 180/ PI ;//acc data to degree data  dgy\_y = gy\_y / 131.;//gyro output to  dt = (double)(micros()- pasttime)/ 1000000;  pasttime = micros();//convert output to understandable data  doubleval = kal.getkalman(deg, dgy\_y, dt);//get kalman data  Serial.print("kalman degree");  Serial.print("\t");  Serial.println(val);  }      각도가 잘 받아지는 모습이다. 위치를 이동시켜도 실시간으로 각도의 변화가 눈에 보이며 딜레이도 크게 보이지 않았다. |
| **활동 사진** | 활동 내용 마지막의 사진 참조. |
| **활동 결과** | 칼만 필터를 사용하여 mpu6050으로 실시간 각도 변화를 측정하는 방법을 생각하여 실습을 해보았고, 생각한대로 잘 된 것을 확인할 수 있었다. 센서와 컴퓨터 간의 거리가 먼 로켓이나 위성의 경우에는 각도를 받아오는 도중에 딜레이가 생길 수 있겠지만, 거리가 가까운 상황만을 한정하였을 때는 값을 수월하게 받아오는 것을 알 수 있었다. |
| **계획** | 칼만 필터 이외의 방법을 실험해 보고, 가장 좋은 방법을 선택하여 먼 거리에서 값을 받아오는 경우에도 큰 문제가 없는지 확인할 것. |

2024년 04월 13일

작성자 : 문시경 (인)