*※보고서 제목 양식 예 : 이름\_20240403*

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

|  |  |
| --- | --- |
| **활동 개요** | 로켓의 각도 추정 방법에 대한 조사 |
| **일시** | 2024년 04월 13일 18시 |
| **작성자** | 노은지 |
| **활동 내용** | **로켓의 각도 추정 방법에 대해 조사**  텍스트, 스크린샷, 폰트, 문서이(가) 표시된 사진  자동 생성된 설명  도입  1. 자세 제어의 필요성: 비행체가 안정적으로 경로를 따라가려면 제어가 필요하다.  2. 유도 방식: 프로그램 유도(컴퓨터의 메모리에 기억시킴), 전파유도(전파 또는 로켓에서 전송되는 신호와 지상 레이더의 관측 자료를 사용하여 계획된 궤도로 원격 조정하는 방법), 관성항법 시스템(로켓 내부에 탑재된 자이로와 가속도계를 사용하여 위치를 계산하고 경로를 스스로 유도한다.)  3. 제어 방법: 로켓에 부착된 날개에서 발생하는 공력을 이용  급격한 선회나 기동이 필요하여 공력보다 더 큰 조종력이나 고도가 높아 공기가 희박한 경우는 추력의 방향을 제어한다.  Imu(+지자계): 자세제어 / GPS, 가속센서 : 3D위치와 속도제어  압력 고도계와 라이더: 고도계  현재 자세 결정🡪 현재 값과 제어값 값 비교🡪추정  4. 목표 : 센서를 이용할 때, 다양한 노이즈가 들어오게 되는데 이 노이즈를 제거하여 깔끔한 데이터를 받아 계산하는 방법을 찾는 것  텍스트, 폰트, 번호, 라인이(가) 표시된 사진  자동 생성된 설명  위키피티아에 따르면 지금은 곱셈 확장 칼만 필터(MEKF)를 많이 쓰는 듯 하다..  참고: <https://www.youtube.com/watch?v=dmAZV65Q4PY> – 수학적 개념  1. 센서  IMU(Inertia Measerment Unit) 센서:관선을 측정하여 최종적으로 구하고자 하는 값은 물체가 기울어진 각도를 정확하게 측정하는 것. + 각속도계, 가속도계, 지자기센서로 구성된 센서를 뜻한다.  각속도\*시간= 회전 각도  MEMS 자이로 센서는 코리올리의 힘을 측정하여 전기적 신호로 변환한다.  참고: <https://m.blog.naver.com/ccaa09/220704142932>  [가속도 센서를 이용한 각도 측정은 정지된 상태에서 물체가 움직이면 기울기가 나타나지 않을 수 있다. 따라서 로켓에서는 자이로 센서를 이용한다.](https://wadadada.tistory.com/entry/%EA%B0%80%EC%86%8D%EB%8F%84%EC%84%BC%EC%84%9C%EC%9D%98-%EC%9B%90%EB%A6%AC-%EA%B0%81%EB%8F%84-%EA%B2%80%EC%B6%9C%EB%B0%A9%EB%B2%95%EC%A4%91%EB%A0%A5%EA%B0%80%EC%86%8D%EB%8F%84%EC%84%BC%EC%84%9C-%EC%9E%90%EC%9D%B4%EB%A1%9C%EC%84%BC%EC%84%9C)  • MPU6050: 드론에서 회전, 기울기 등을 측정 하여 모터를 정확히 제어하는 담당’  가속도 3축 + 자이로 2축+ 온도1 축을 가진다.  I2C통신을 한다.  가속도 센서: 가속도 자체를 측정. 중력 가속도가 3축으로 얼만큼의 영향을 주었는가를 통해 기울기 확인  자이로 센서: 각속도를 측정. 가속도에 비해 안정적이나 각도를 계산하는 적분에서 누적 오차가 발생한다.  온도 센서: 자이로 센서가 온도에 민감하기 때문에 달린다.  온도 보정법-https://m.blog.naver.com/pg365/223019520416  • ICM- 20948: MPU6050과 같이 자세제어에 사용된다.  3축 각속도 + 3축 각속도 + 3축 지자계  SPI, I2C통신  MPU6050을 통한 각도 계산은 아래 링크에서 참조할 수 있다. (각도 계산 코드 포함)  <https://blog.naver.com/codingbird/221766900497>  <https://www.nano-i.com/index.php?mid=gyro&document_srl=1276643&listStyle=viewer> - mcu칩에서 출력되는 원시 데이터 확인 및 계산 과정 확인  MPU6050에 많은 노이즈가 섞이므로 필터링이 필요하다.  텍스트, 스크린샷, 그래프, 라인이(가) 표시된 사진  자동 생성된 설명  텍스트, 스크린샷, 도표, 번호이(가) 표시된 사진  자동 생성된 설명출처: https://chigun.tistory.com/21    MPU6050 회로도  텍스트, 스크린샷, 번호, 폰트이(가) 표시된 사진  자동 생성된 설명  MPU6050 pin map  datasheet  각도 변환 표  출처: https://int-i.github.io/cpp/2023-06-24/mpu6050-complementary-filter/  MEMS(Micro – Elctro Mechanical System)방식: 미세 전자 기계 시스템. 센서의 제조 방식이다. 압저항, 정전용량,열전달, 압전, 홀 효과, 서보 방식이 이에 해당된다.  텍스트, 폰트, 번호, 스크린샷이(가) 표시된 사진  자동 생성된 설명  **참고**  <https://m.blog.naver.com/iotsensor/221629271678> - MEMS의 공정  <https://blog.onepredict.ai/63a5e7e6-c76f-4c9f-ab34-22ee9ef09d83> - 가속도계의 종류와 설명  <https://mj-thump-thump-story.tistory.com/entry/IMU-Arduino%EC%97%90%EC%84%9C-MPU-9250-%EC%82%AC%EC%9A%A9%ED%95%98%EB%8A%94-%EB%B0%A9%EB%B2%95> - IMU 입문  <https://swiftcam.tistory.com/95> -- IMU센서와 MEMS에 대한 이해  <https://m.blog.naver.com/proscw/221276680031> - MPU 6050 코드포함  <https://blog.naver.com/eduino/221081288204> - MPU6050 기울기 센서 원리  <https://swiftcam.tistory.com/159> -센서 측정, 노이즈 심하다  [https://mokhwasomssi.tistory.com/m/27 - ICM20948](https://mokhwasomssi.tistory.com/m/27%20-%20ICM20948)참고  2. 필터  **2-1. 칼만필터**  Drawing  칼만필터 흐름도  출처: https://gaussian37.github.io/ad-ose-lkf\_basic/  새로운 측정 데이터를 과거의 데이터와 함께 재귀적 연산을 통해 데이터에 포함된 노이즈를 제거시키는 역할로, 새 결과를 추정한다.  텍스트, 폰트, 라인, 스크린샷이(가) 표시된 사진  자동 생성된 설명  출처: <https://pajamacoder.tistory.com/8>  **참고**  <https://gaussian37.github.io/ad-ose-lkf_basic/> - 칼만필터 수학적 모델  <https://blog.naver.com/msnayana/80106682874> – 칼만필터 이론  <https://limitsinx.tistory.com/74> - 칼만필터 코드 분석,  <https://soohwan-justin.tistory.com/43> - 칼만필터 그림 참고  **2-2. 상보필터**  텍스트, 도표, 스크린샷, 평면도이(가) 표시된 사진  자동 생성된 설명  출처: https://yjhtpi.tistory.com/352  텍스트, 도표, 라인, 폰트이(가) 표시된 사진  자동 생성된 설명  출처: <https://swiftcam.tistory.com/163>  가속도 센서 특성상 고주파 영역의 노이즈를 제거하고자 low pass filter를 쓴다.  자이로 센서는 저주파 영역에서 drift현상이 발생해 high pass filter를 쓴다.  **참고**  <https://yjhtpi.tistory.com/352> - 상보필터 코드  <https://alnova2.tistory.com/1085> – 코드  <https://m.blog.naver.com/ysahn2k/221385063966> -상보필터 테스트  <https://fouranswer.tistory.com/84> - 상보필터 수학적 개념  **2-3. 베이지안 필터**  텍스트, 스크린샷, 폰트, 라인이(가) 표시된 사진  자동 생성된 설명  출처: <https://pasus.tistory.com/145> - 베이즈 필터와 마르코프 시퀀스  이산 시간 확률 동적 시스템의 상태 변수를 추정하는 확률론적 방법으로 추정 알고리즘의 기반이 됨  1. 측정값과 입력 값이 주어질 때, 현재 상태를 추정하는 알고리즘이다.  2. 베이즈 룰을 이용하여 센서 입력값을 현재 상태에 반영  **참고**  <https://gaussian37.github.io/autodrive-ose-bayes_filter/> -베이즈 필터 개념, 이해하기 좋은자료  **2-4. 파티클 필터**  도표, 라인, 그림이(가) 표시된 사진  자동 생성된 설명  출처: https://ryanclaire.blogspot.com/2020/06/particle-filter-principle.html  라인, 폰트, 스크린샷이(가) 표시된 사진  자동 생성된 설명  참고: <https://soohwan-justin.tistory.com/48>  입자를 기반으로 한 칼만 필터의 한 종류  칼만 필터와 다르게 상태 변수가 가우시안 분포를 따르지 않아도 적용이 가능한 비선형 필터이다.  1. 하나 이상의 솔루션 모델링 가능  2. resampling step 단계 존재  - 파티클 필터를 이용한 로켓 고장 검출  로켓의 자세제어에 사용은 어려울 것 같다!  **참고**  <https://velog.io/@soup1997/Particle-Filter> - 파티클 필터 개념  <https://limitsinx.tistory.com/80> - 비선형 칼만 필터  <https://unknownpgr.com/posts/particle-filter/index.html> - 파티클 필터 수학적 개념  <https://alida.tistory.com/62>  **2-5. 노치 필터**  그래프, 라인, 도표, 텍스트이(가) 표시된 사진  자동 생성된 설명  출처: https://blog.naver.com/matthias91song/221532526087  좁은 주파수 대역의 신호를 차단  **참고**  <https://medialink.tistory.com/105> - 노치필터 개념  <https://patents.google.com/patent/KR20080103228A/ko> -노치필터로 고유진동수를 구하여 신호 필터링  3. 코드  **칼만필터**  #include <Wire.h> //라이브러리 불러오기  #define mpu\_add 0x68  //mpu6050 address  class kalman {    public :      double getkalman(double acc, double gyro, double dt) { //15자리 숫자        //project the state ahead        angle += dt \* (gyro - bias) ; //각도 계산        //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        double S = P[0][0] + R\_measure ;        K[0] = P[0][0] / S ;        K[1] = P[1][0] / S ;        //Update estimate with measurement z        double y = acc - angle ;        angle += K[0] \* y ;        bias += K[1] \* y ;        //Update the error covariance        double P\_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] ;        return angle ;      } ;      void init(double angle, double gyro, double measure) {        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 ;      } ;      double getvar(int num) {        switch (num) {          case 0 :            return Q\_angle ;            break ;          case 1 :            return Q\_gyro ;            break ;          case 2 :            return R\_measure ;            break ;        }      } ;    private :      double Q\_angle, Q\_gyro, R\_measure ;      double angle, bias ;      double P[2][2], K[2] ;  } ;  kalman kal ;  long ac\_x, ac\_y, ac\_z, gy\_x, gy\_y, gy\_z ; // 정수를 저장하는 변수  double deg, dgy\_y ;  double dt ;  uint32\_t pasttime ;  void setup() {    // 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) ;  }  void loop() {    // 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    double val = kal.getkalman(deg, dgy\_y, dt) ;  //get kalman data    Serial.print("kalman degree") ;    Serial.print("\t") ;    Serial.println(val) ;  }  출처:<https://m.blog.naver.com/roboholic84/220421114302>  **상보필터**  #include<Wire.h>  const int MPU\_ADDR = 0x68; // I2C통신을 위한 MPU6050의 주소  int16\_t AcX, AcY, AcZ, Tmp, GyX, GyY, GyZ; // 가속도(Acceleration)와 자이로(Gyro)  double angleAcX, angleAcY, angleAcZ;  double angleGyX, angleGyY, angleGyZ;  double angleFiX, angleFiY, angleFiZ;  const double RADIAN\_TO\_DEGREE = 180 / 3.14159;  const double DEG\_PER\_SEC = 32767 / 250; // 1초에 회전하는 각도  const double ALPHA = 1 / (1 + 0.04);  // GyX, GyY, GyZ 값의 범위 : -32768 ~ +32767 (16비트 정수범위)  unsigned long now = 0; // 현재 시간 저장용 변수  unsigned long past = 0; // 이전 시간 저장용 변수  double dt = 0; // 한 사이클 동안 걸린 시간 변수  double averAcX, averAcY, averAcZ;  double averGyX, averGyY, averGyZ;  void setup() {  initSensor();  Serial.begin(115200);  caliSensor(); // 초기 센서 캘리브레이션 함수 호출  past = millis(); // past에 현재 시간 저장  }  void loop() {  getData();  getDT();  angleAcX = atan(AcY / sqrt(pow(AcX, 2) + pow(AcZ, 2)));  angleAcX \*= RADIAN\_TO\_DEGREE;  angleAcY = atan(-AcX / sqrt(pow(AcY, 2) + pow(AcZ, 2)));  angleAcY \*= RADIAN\_TO\_DEGREE;  // 가속도 센서로는 Z축 회전각 계산 불가함.    // 가속도 현재 값에서 초기평균값을 빼서 센서값에 대한 보정  angleGyX += ((GyX - averGyX) / DEG\_PER\_SEC) \* dt; //각속도로 변환  angleGyY += ((GyY - averGyY) / DEG\_PER\_SEC) \* dt;  angleGyZ += ((GyZ - averGyZ) / DEG\_PER\_SEC) \* dt;  // 상보필터 처리를 위한 임시각도 저장  double angleTmpX = angleFiX + angleGyX \* dt;  double angleTmpY = angleFiY + angleGyY \* dt;  double angleTmpZ = angleFiZ + angleGyZ \* dt;  // (상보필터 값 처리) 임시 각도에 0.96가속도 센서로 얻어진 각도 0.04의 비중을 두어 현재 각도를 구함.  angleFiX = ALPHA \* angleTmpX + (1.0 - ALPHA) \* angleAcX;  angleFiY = ALPHA \* angleTmpY + (1.0 - ALPHA) \* angleAcY;  angleFiZ = angleGyZ; // Z축은 자이로 센서만을 이용하열 구함.  Serial.print("AngleAcX:");  Serial.print(angleAcX);  Serial.print("\t FilteredX:");  Serial.print(angleFiX);  Serial.print("\t AngleAcY:");  Serial.print(angleAcY);  Serial.print("\t FilteredY:");  Serial.println(angleFiY);  Serial.print("\t AngleAcZ:");  Serial.print(angleGyZ);  Serial.print("\t FilteredZ:");  Serial.println(angleFiZ);  // Serial.print("Angle Gyro X:");  // Serial.print(angleGyX);  // Serial.print("\t\t Angle Gyro y:");  // Serial.print(angleGyY);  // Serial.print("\t\t Angle Gyro Z:");  // Serial.println(angleGyZ);  // delay(20);  }  void initSensor() {  Wire.begin();  Wire.beginTransmission(MPU\_ADDR); // I2C 통신용 어드레스(주소)  Wire.write(0x6B); // MPU6050과 통신을 시작하기 위해서는 0x6B번지에  Wire.write(0);  Wire.endTransmission(true);  }  void getData() {  Wire.beginTransmission(MPU\_ADDR);  Wire.write(0x3B); // AcX 레지스터 위치(주소)를 지칭합니다  Wire.endTransmission(false);  Wire.requestFrom(MPU\_ADDR, 14, true); // AcX 주소 이후의 14byte의 데이터를 요청  AcX = Wire.read() << 8 | Wire.read(); //두 개의 나뉘어진 바이트를 하나로 이어 붙여서 각 변수에 저장  AcY = Wire.read() << 8 | Wire.read();  AcZ = Wire.read() << 8 | Wire.read();  Tmp = Wire.read() << 8 | Wire.read();  GyX = Wire.read() << 8 | Wire.read();  GyY = Wire.read() << 8 | Wire.read();  GyZ = Wire.read() << 8 | Wire.read();  }  // loop 한 사이클동안 걸리는 시간을 알기위한 함수  void getDT() {  now = millis();  dt = (now - past) / 1000.0;  past = now;  }  // 센서의 초기값을 10회 정도 평균값으로 구하여 저장하는 함수  void caliSensor() {  double sumAcX = 0 , sumAcY = 0, sumAcZ = 0;  double sumGyX = 0 , sumGyY = 0, sumGyZ = 0;  getData();  for (int i=0;i<10;i++) {  getData();  sumAcX+=AcX; sumAcY+=AcY; sumAcZ+=AcZ;  sumGyX+=GyX; sumGyY+=GyY; sumGyZ+=GyZ;  delay(50);  }  averAcX=sumAcX/10; averAcY=sumAcY/10; averAcZ=sumAcY/10;  averGyX=sumGyX/10; averGyY=sumGyY/10; averGyZ=sumGyZ/10;  }  출처:<https://yjhtpi.tistory.com/352> |
| **활동 사진** | - |
| **활동 결과** | 필터와 센싱하는 과정에 대해서 이해하고 자세를 제어하는 방법에 대해 이해  MPU6050, ICM20948,IMU,상보필터,칼만필터,베이지안 필터,파티클 필터,노치필터 |
| **계획** | 1. 필터와 로켓에 대해 조사하고 적용 방법에 대해 자세히 생각하기  2. 센서 값 받아들여 와서 조사된 내용 적용하기 |

2024년 04월 13일

작성자 : 노은지 (인)

