

드론 Math/Physics 그리고 Algorithm

Subak.io

목차

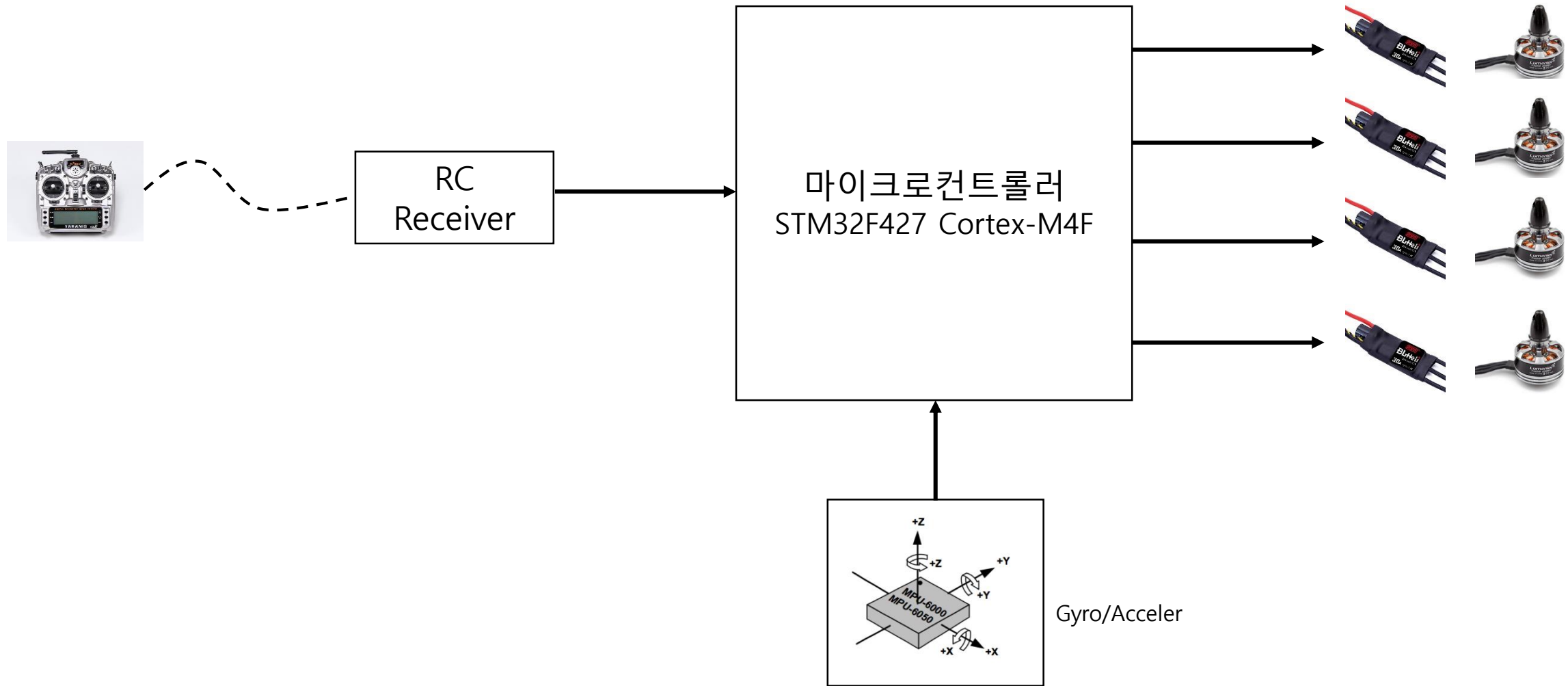
- Multirotors 특징
- QuadCopter 구조
- Roll, Pitch, Yaw
- 제어시스템
- Angle 제어
- Sensor Fusion
- Attitude 제어
- 그밖에 제어

Multirotors 특징

- 간단한 기구
 - N개 모터, N개 프로펠러
- 기계 장치 대신 motor speed로 제어
- 전자/컴퓨터 기반 제어 시스템 이용
- 좁은 공간 이착륙 가능

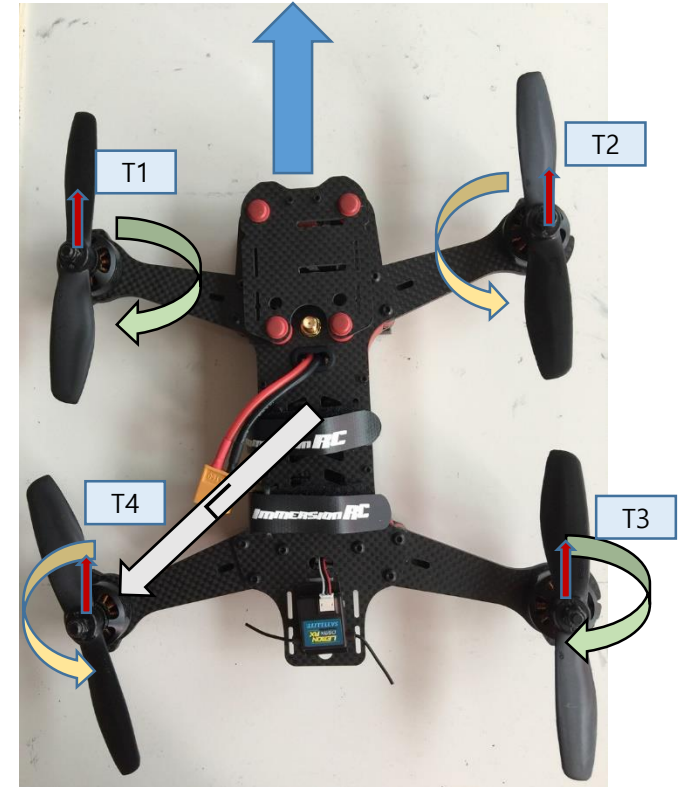
QuadCopter 구조

드론 시스템 구조 (Pixhawk)



기본 정의

- Weight
 $W = mg$ (m : mass)
- Rotation Speed
 $\omega_1, \omega_2, \omega_3, \omega_4$
- Force
각 Propeller에 가해지는 force : T_1, T_2, T_3, T_4
 $T_i \propto \omega_i^2$
- Moments
 - 각 Moment에 가해지는 moment : M_1, M_2, M_3, M_4
 - $M_i = L \times T_i$

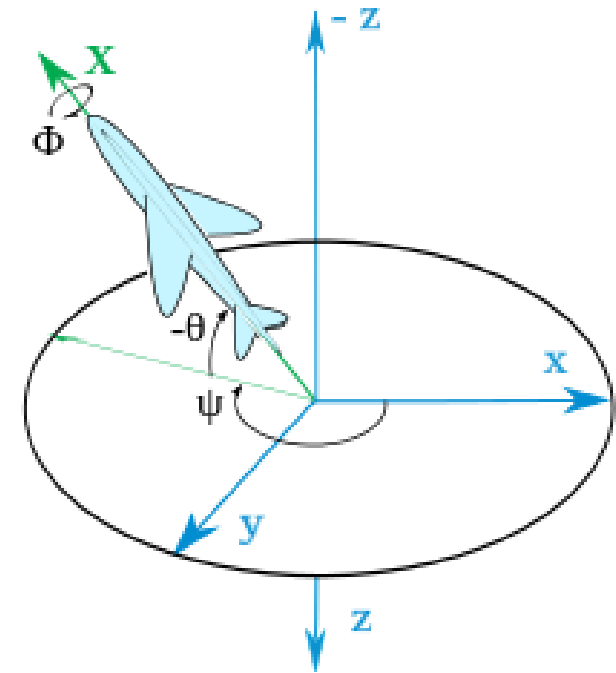


Hold 상태

- Force :
 - $\sum_{i=1}^4 T_i = -mg$
- Direction :
 - $T_{1,2,3,4} \parallel g$
- Moments :
 - $\sum_{i=1}^4 M_i = 0$
- Rotation Speed :
 - $(\omega_1 + \omega_3) - (\omega_2 + \omega_4) = 0$

레퍼런스 시스템

- 레퍼런스 시스템
 - 공간에서 위치를 정하기 위한 시스템
- 2가지 레퍼런스 시스템
 - Earth Frame
 - (x_E, y_E, z_E)
 - Body Frame
 - (x_B, y_B, z_B)



오일러 각

- 이 2가지 레퍼런스 시스템사이에서 3개 angle을 정의
 - Roll ϕ : $x_B \parallel x_E$
 - Pitch θ : $y_B \parallel y_E$
 - Yaw ψ : $z_B \parallel z_E$
- 이 3가지 각을 오일러 각이라고 부른다.

- 이 2가지 레퍼런스 시스템사이에서 3개 angle을 정의
 - Roll ϕ : $x_B \parallel x_E$
 - Pitch θ : $y_B \parallel y_E$
 - Yaw ψ : $z_B \parallel z_E$
- 이 3가지 각을 오일러 각이라고 부른다.

각속도

- 시간에 따른 각의 변화량
 - Roll Rate : $\dot{\phi}$
 - Pitch Rate : $\dot{\theta}$
 - Yaw Rate : $\dot{\psi}$

Drone 제어 분석

- Drone의 자세/이동에 따른 상태 분석

Hold 상태

- Force :
 - $\sum_{i=1}^4 T_i = -mg$
- Direction :
 - $T_{1,2,3,4} \parallel g$
- Moments :
 - $\sum_{i=1}^4 M_i = 0$
- Rotation Speed :
 - $(\omega_1 + \omega_3) - (\omega_2 + \omega_4) = 0$

$\phi = 0$	$\theta = 0$	$\psi = 0$
$\dot{\phi} = 0$	$\dot{\theta} = 0$	$\dot{\psi} = 0$

상하이동

- Force :

- $\sum_{i=1}^4 T_i \neq -mg$

- Direction :


- $T_{1,2,3,4} \parallel g$


- Moments :

- $\sum_{i=1}^4 M_i = 0$

- Rotation Speed :

- $(\omega_1 + \omega_3) - (\omega_2 + \omega_4) = 0$

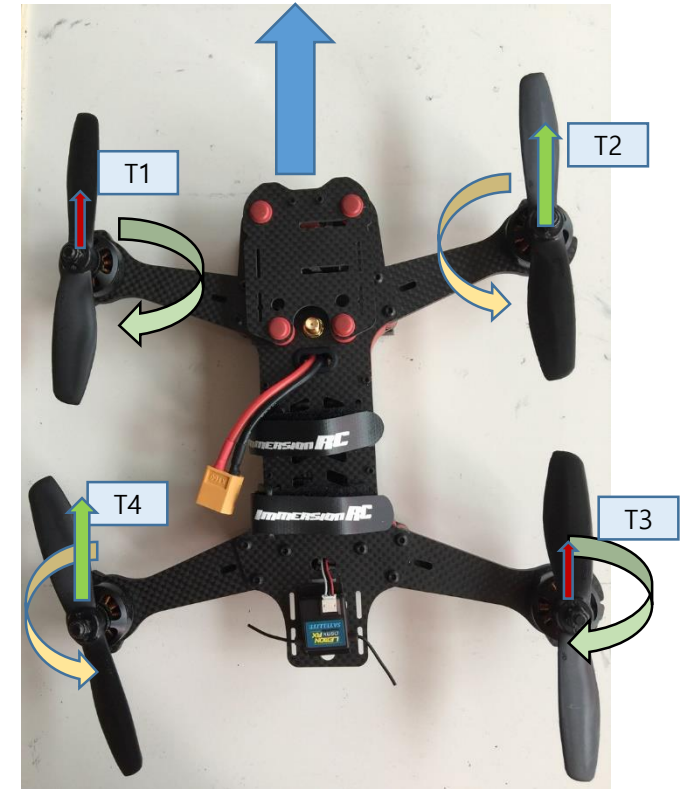

$$\sum_{i=1}^4 T_i > -mg$$


$$\sum_{i=1}^4 T_i < -mg$$

모든 프로펠러의 각속도를 높이거나 줄여서 (오일러 각과 rate는 여전히 0)

Yaw

- Force :
 - $\sum_{i=1}^4 T_i = -mg$
- Direction :
 - $T_{1,2,3,4} \parallel g$
- Moments :
 - $\sum_{i=1}^4 M_i = 0$
- Rotation Speed :
 - $(\omega_1 + \omega_3) - (\omega_2 + \omega_4) \neq 0$

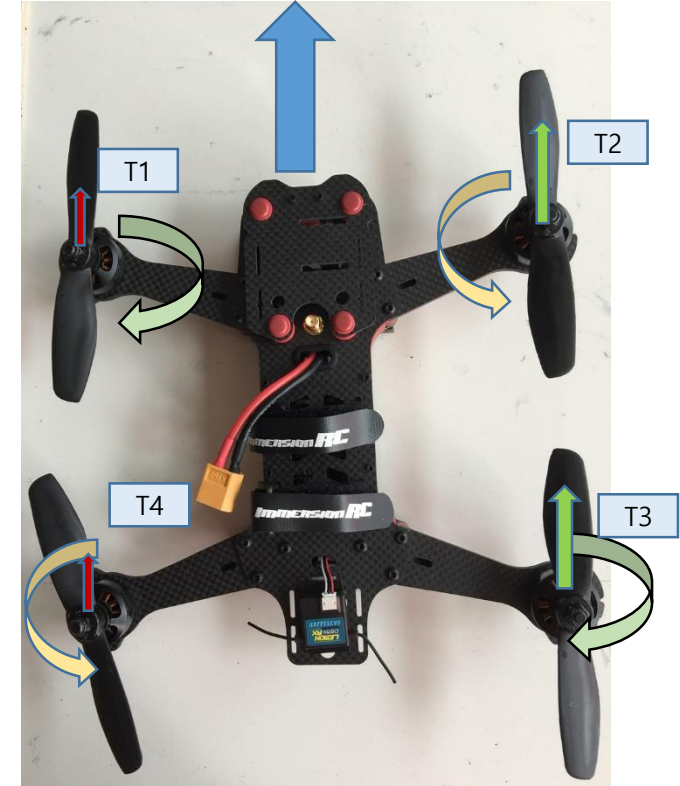


$$\dot{\psi} = k_Y((\omega_1 + \omega_3) - (\omega_2 + \omega_4))$$

$$\psi = \int \dot{\psi} dt$$

Roll

- **Direction :**
 - $T_{1,2,3,4}$ 와 g 는 평행이 아니다.
- **Moments :**
 - $\sum_{i=1}^4 M_i \neq 0$
- **Rotation Speed :**
 - $(\omega_1 + \omega_4) - (\omega_2 + \omega_3) \neq 0$

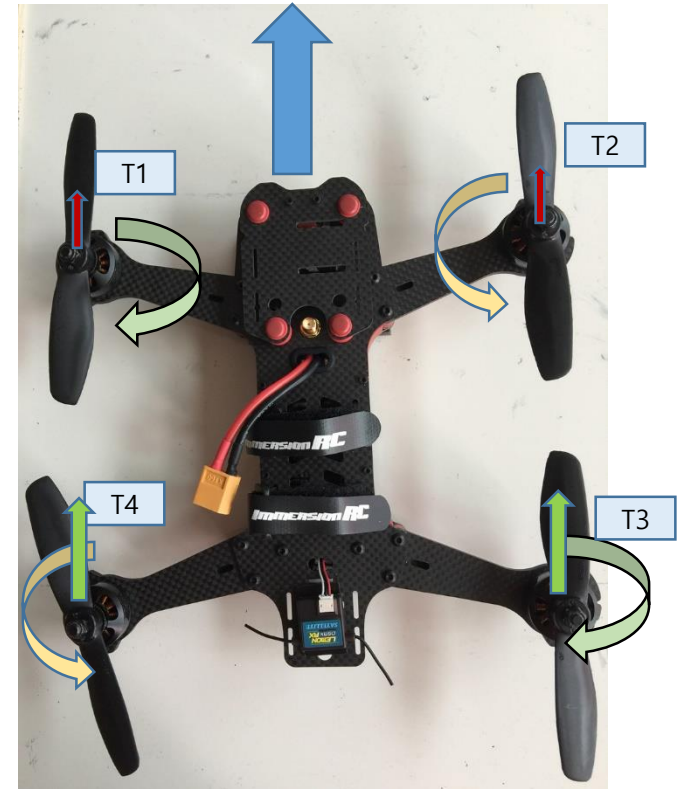


$$\dot{\phi} = k_R((\omega_1 + \omega_4) - (\omega_2 + \omega_3))$$

$$\phi = \int \dot{\phi} dt$$

Pitch

- **Direction :**
 - $T_{1,2,3,4}$ 와 g 는 평행이 아니다.
- **Moments :**
 - $\sum_{i=1}^4 M_i \neq 0$
- **Rotation Speed :**
 - $(\omega_1 + \omega_2) - (\omega_3 + \omega_4) \neq 0$



$$\dot{\theta} = k_P((\omega_1 + \omega_2) - (\omega_3 + \omega_4))$$

$$\theta = \int \dot{\theta} dt$$

운동 방정식

$$\begin{aligned}\dot{\phi} &= k((\omega_1 + \omega_4) - (\omega_2 + \omega_3)) = k\omega_1 - k\omega_2 - k\omega_3 + k\omega_4 \\ \dot{\theta} &= k((\omega_1 + \omega_2) - (\omega_3 + \omega_4)) = k\omega_1 + k\omega_2 - k\omega_3 - k\omega_4 \\ \dot{\psi} &= k((\omega_1 + \omega_3) - (\omega_2 + \omega_4)) = k\omega_1 - k\omega_2 + k\omega_3 - k\omega_4 \\ F &= k(\omega_1 + \omega_2 + \omega_3 + \omega_4) = k\omega_1 + k\omega_2 + k\omega_3 + k\omega_4\end{aligned}$$

$$\begin{pmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \\ F \end{pmatrix} = \begin{pmatrix} k & -k & -k & k \\ k & k & -k & -k \\ k & -k & k & -k \\ k & k & k & k \end{pmatrix} \begin{pmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \omega_4 \end{pmatrix}$$

k는 비례 상수

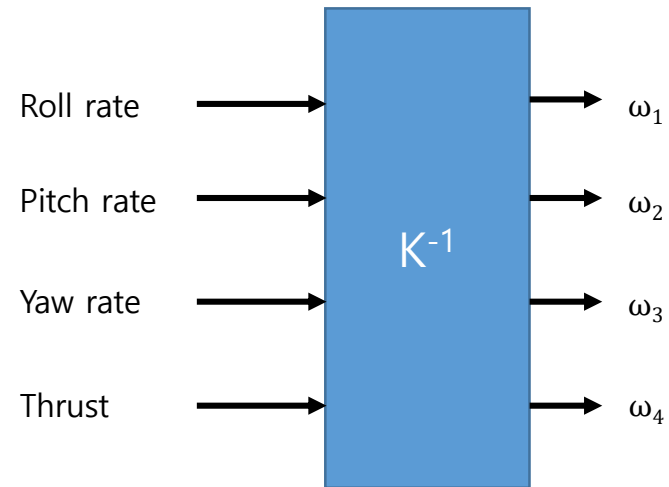
운동 방정식

$$\begin{pmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \\ F \end{pmatrix} = \begin{pmatrix} k & -k & -k & k \\ k & k & -k & -k \\ k & -k & k & -k \\ k & k & k & k \end{pmatrix} \begin{pmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \omega_4 \end{pmatrix} = K \begin{pmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \omega_4 \end{pmatrix}$$

여기서 각속도가 주어지면 Roll, Pitch, Yaw의 rate를 구할 수 있다.
여기서 rate가 곧 비행체의 rotation rate을 제어할 수 있다.

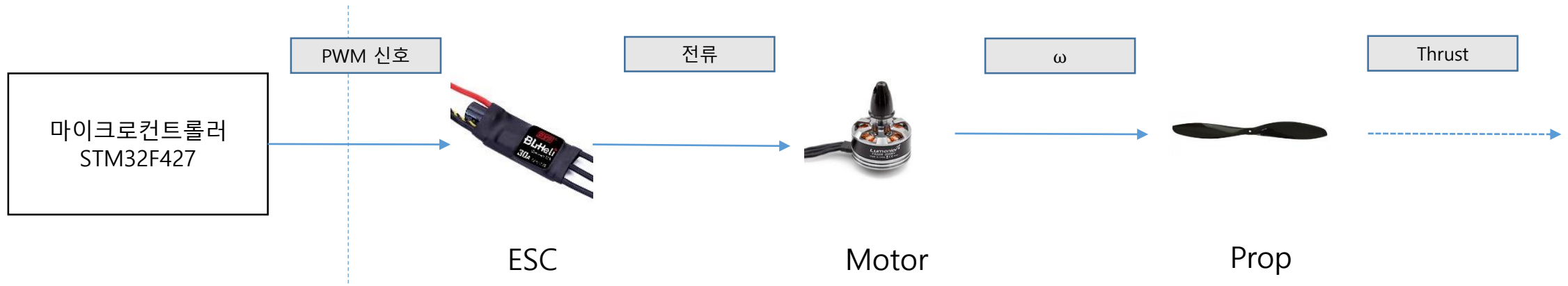
Roll, Pitch, Yaw Rate 및 Thrust

$$\begin{pmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \omega_4 \end{pmatrix} = K^{-1} \begin{pmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \\ F \end{pmatrix} = \begin{pmatrix} k & -k & -k & k \\ k & k & -k & -k \\ k & -k & k & -k \\ k & k & k & k \end{pmatrix} \begin{pmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \\ F \end{pmatrix}$$



제어 시스템은 왜 필요할까?

Rotation Rate 제어 과정



제어가 힘든 이유

- ESC, 모터, 프로펠러 4개가 완전히 동일하게 동작해야 한다!
 - 현실적으로 불가능!
- 같은 제조사의 동일 모델이더라도 각각의 편차가 발생
- 무게 중심도 매번 정확히 맞추기 어렵다.

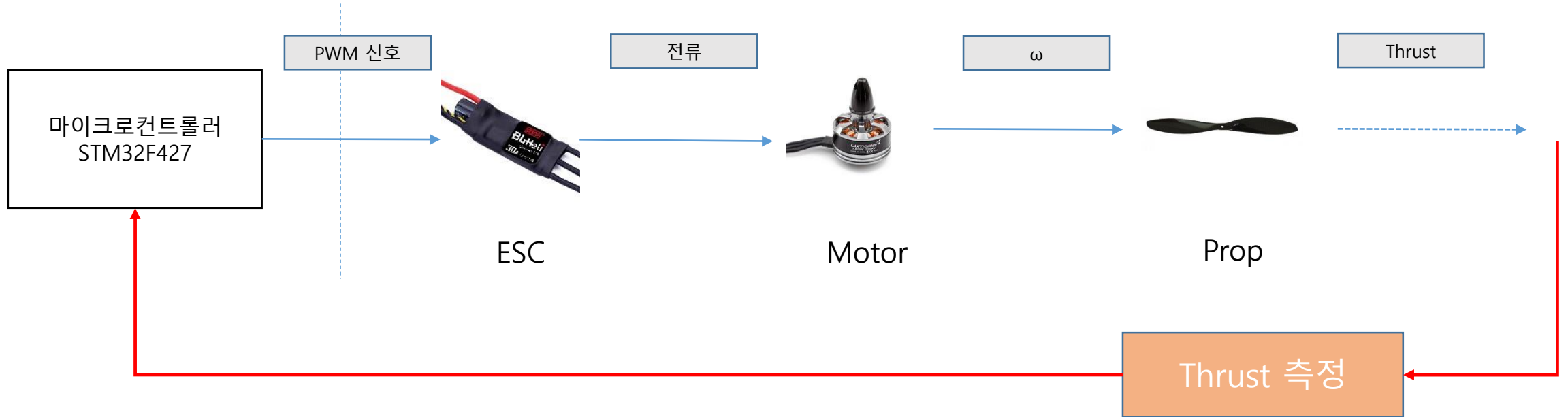
수학/물리학적 해결책

- 각 Arm의 ESC, 모터, 프로펠러를 하나의 chain으로 본다.
- 각 chain의 특성을 고려하여 동일하게 동작하는 함수 도출!
 - $T_i = f_i(PWM_i)$ $PWM_i = f_i^{-1}(T_i)$
- 문제점 :
 - 특성을 고려한 함수를 구하기 어렵다.
 - 부품중에 하나라도 바뀌면 과정을 반복해야 한다.
 - 비행시 발생하는 변수를 극복할 수 없다.(바람, 공기 밀도 등)

컴퓨터 공학 해결책

- 해결방법 :
 - 각 부품별 offset/gain 구하기
 - 원하는 동작에 맞는 값
- 문제점 :
 - Offset/gain 구하기 어렵다.
 - 부품중에 하나라도 바뀌면 과정을 반복해야 한다.
 - 비행시 발생하는 변수를 극복할 수 없다.(바람, 공기 밀도 등)

제어 시스템에서 해결책



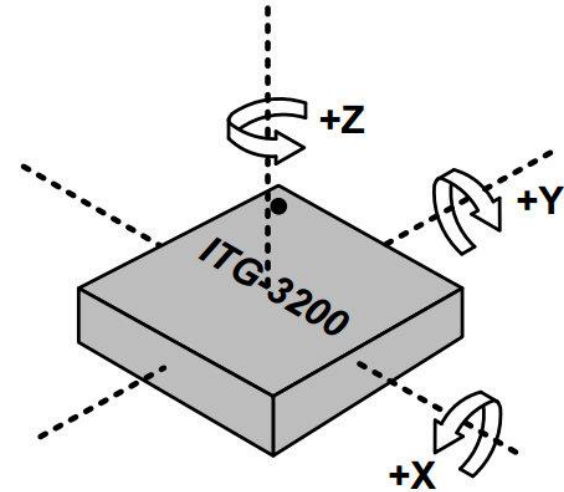
해결책 :

- Sensor를 변화 측정
- 목표값과 측정값 비교(error 값 도출)
- Error 값을 기초로 시스템 보정
- 반복

제어 시스템 목표/측정

측정 :

- 실제 각속도 측정
- Gyro 센서를 통해 3축 회전 속도 측정

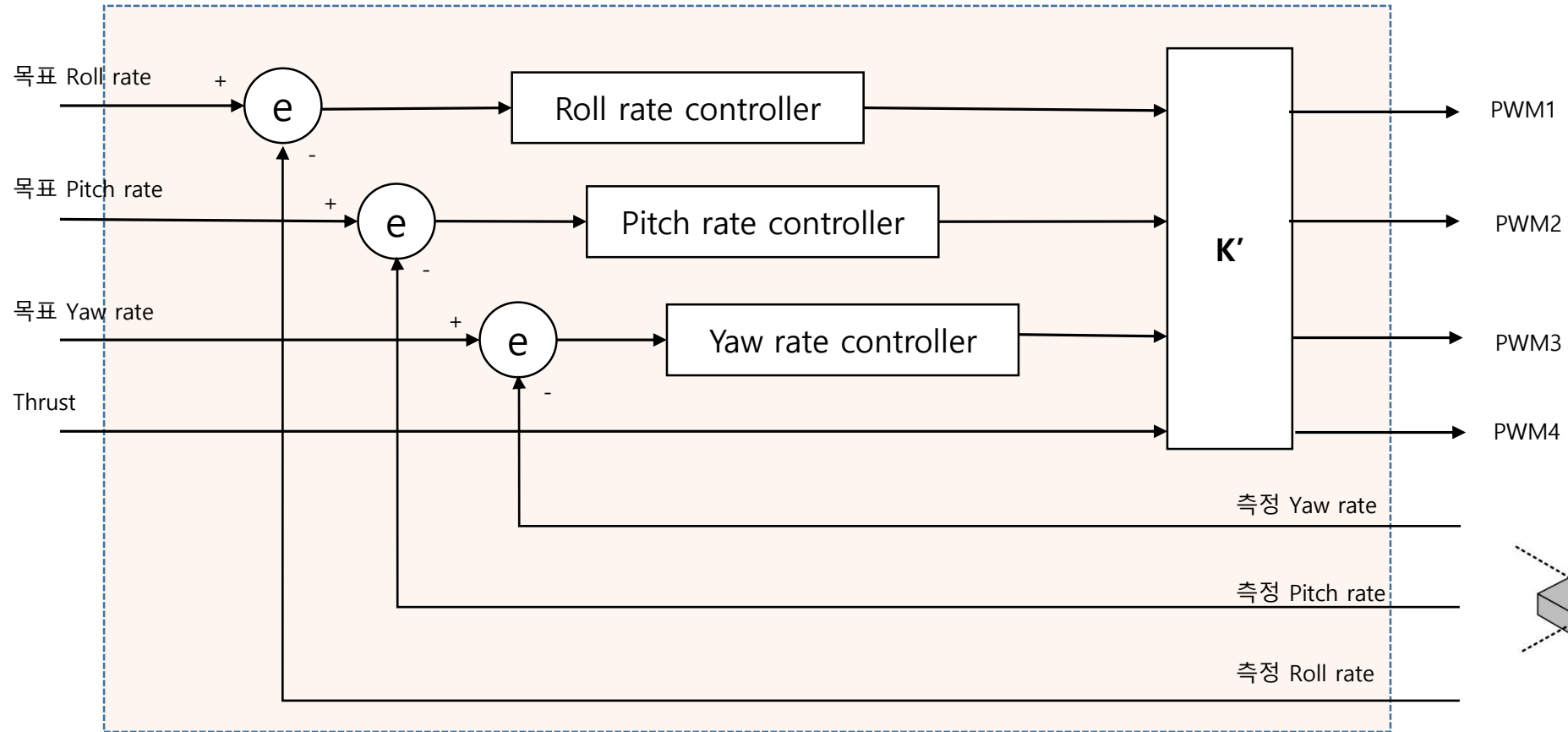


목표 :

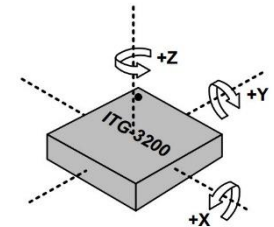
- 목표 각속도는 RC를 통해 설정



Feedback 제어 전체 구조



마이크로컴 내부 동작



알고리즘

While

$(\dot{\phi}_T, \dot{\theta}_T, \dot{\psi}_T, F) = \text{Read_RC_control}();$

$(\dot{\phi}_M, \dot{\theta}_M, \dot{\psi}_M) = \text{Read_gyro}();$

$e_{\dot{\phi}} = \dot{\phi}_T - \dot{\phi}_M; \quad e_{\dot{\theta}} = \dot{\theta}_T - \dot{\theta}_M; \quad e_{\dot{\psi}} = \dot{\psi}_T - \dot{\psi}_M;$

$C_{\dot{\phi}} = \text{roll_rate_controller}(e_{\dot{\phi}});$

$C_{\dot{\theta}} = \text{pitch_rate_controller}(e_{\dot{\theta}});$

$C_{\dot{\psi}} = \text{yaw_rate_controller}(e_{\dot{\psi}});$

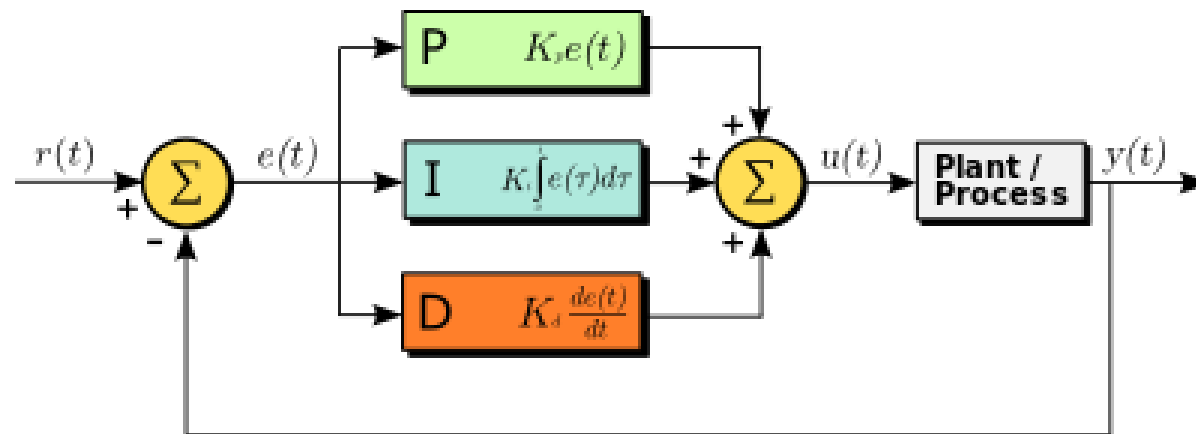
$(\text{pwm}_1, \text{pwm}_2, \text{pwm}_3, \text{pwm}_4)^T = K^{-1}(C_{\dot{\phi}T}, C_{\dot{\theta}T}, C_{\dot{\psi}T})^T;$

$\text{control_motor}(\text{pwm}_1, \text{pwm}_2, \text{pwm}_3, \text{pwm}_4);$

end

PID 제어

- 가장 일반적인 제어기 : Proportional-Integral-Derivative



K_p : short-term action

K_i : long-term action

K_d : error 경향에 기초한 action

출처 : wikipedia

드론 PID 제어 예

<https://www.youtube.com/watch?v=YNzqTGEI2xQ>

Rates vs. Angles

- 지금까지는 Rate에 대해서 다뤘다.
- 드론 제어에서 Rate와 Angle의 정확한 용어의 뜻을 이해해야 한다.

Angle 제어 (Roll, Pitch)



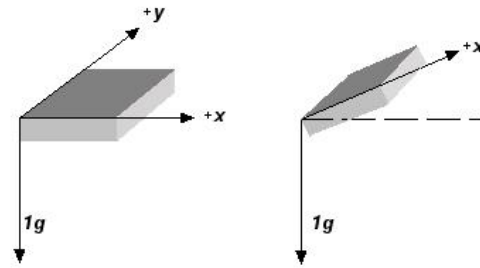
- 상용 드론의 기본 제어 방식
- Stick 이동
 - Center로부터 이동 거리가 angle이 된다.
 - Ex) stick이 가운데로 오면 0도 (균형상태)
- 지금까지 배운 rate 대신 angle을 제어하려면 어떻게 해야할까?

Angle 측정 (Gyro)

- 오일러 각 측정 센서
 - Gyroscope, Accelerometer, Magnetometer 사용
- Gyro
 - 3개 축에 대해서 각속도 측정
 - 얻은 각속도를 적분하면 Angle을 얻을 수 있다!
- 문제점
 - 물리적 오류 (오류 있는 값을 적분하는 경우 오차)
 - Offset 값에 의한 오차
 - 온도 등 외부 영향에 따른 오차
 - 신뢰할 수 없다.

Angle 측정 (Accelerometer)

- 3개 축 상에서 중력가속도로 각도를 측정하는 센서

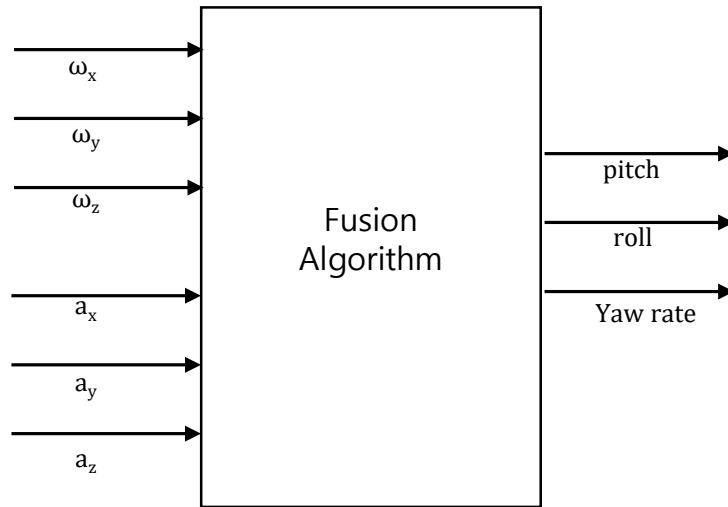
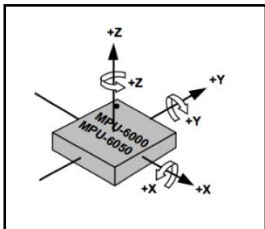


- 중력 g vector의 projection
- 센서가 움직이게 되면 해당 방향으로 acceleration이 발생
- \arctan 로 roll과 pitch를 구할 수 있다.
- 문제점
 - 고정된 상태에서 정확한 roll과 pitch를 각을 구할 수 있다.
 - 움직임이 발생하면 신뢰할 수 없다.

Angle 측정 (Gyro/Accelerometer)

- Gyro/Accelerometer 모두 정확한 angle을 구하지 못한다.
- 이유
 - 2개의 센서에서 각기 다른 방식으로 angle을 추출
 - 각 센서의 특징에 따른 다른 error 발생
 - 이 error로 인해 계산한 angle을 구할 수 없다!
- 해결책
 - Error를 보정
 - 신뢰할 수 있는 angle을 얻기 위한 방법
 - 2개 정보를 fusion!

Sensor Fusion



While

$(\dot{\phi}, \dot{\theta}, \dot{\psi}) = \text{Read_gyro}();$

$(a_x, a_y, a_z) = \text{Read_accel}();$

$(\phi, \theta, \psi) = (\phi, \theta, \psi) + \Delta T(\dot{\phi}, \dot{\theta}, \dot{\psi});$

$\hat{\phi} = \tan^{-1}(-a_y/-a_z);$

$\hat{\theta} = \tan^{-1}(-a_x/\sqrt{a_y^2 + a_z^2});$

$(\phi, \theta, \psi) = \text{fusion_filter}(\phi, \theta, \psi, \hat{\phi}, \hat{\theta});$

end

Sensor Fusion

- Filter 함수
 - Sensor Fusion의 핵심
 - Rotation 표현 방식 (DCM, Quaternion)
 - 상보 필터
 - Kalman Filter
- 기본 아이디어
 - error 함수 : $e(t) = \text{real}(t) - \text{estimated}(t)$
 - 시간이 지나면서 $e(t)$ 가 0이 되도록 설계!

3D에서 Rotation 표현하기

- DCM(Direction Cosine Matrice)
 - 참고자료
 - <http://www.control.aau.dk/~jan/undervisning/MechanicsI/ST8/ST8-MM2/ST8-MM1-view1.pdf>

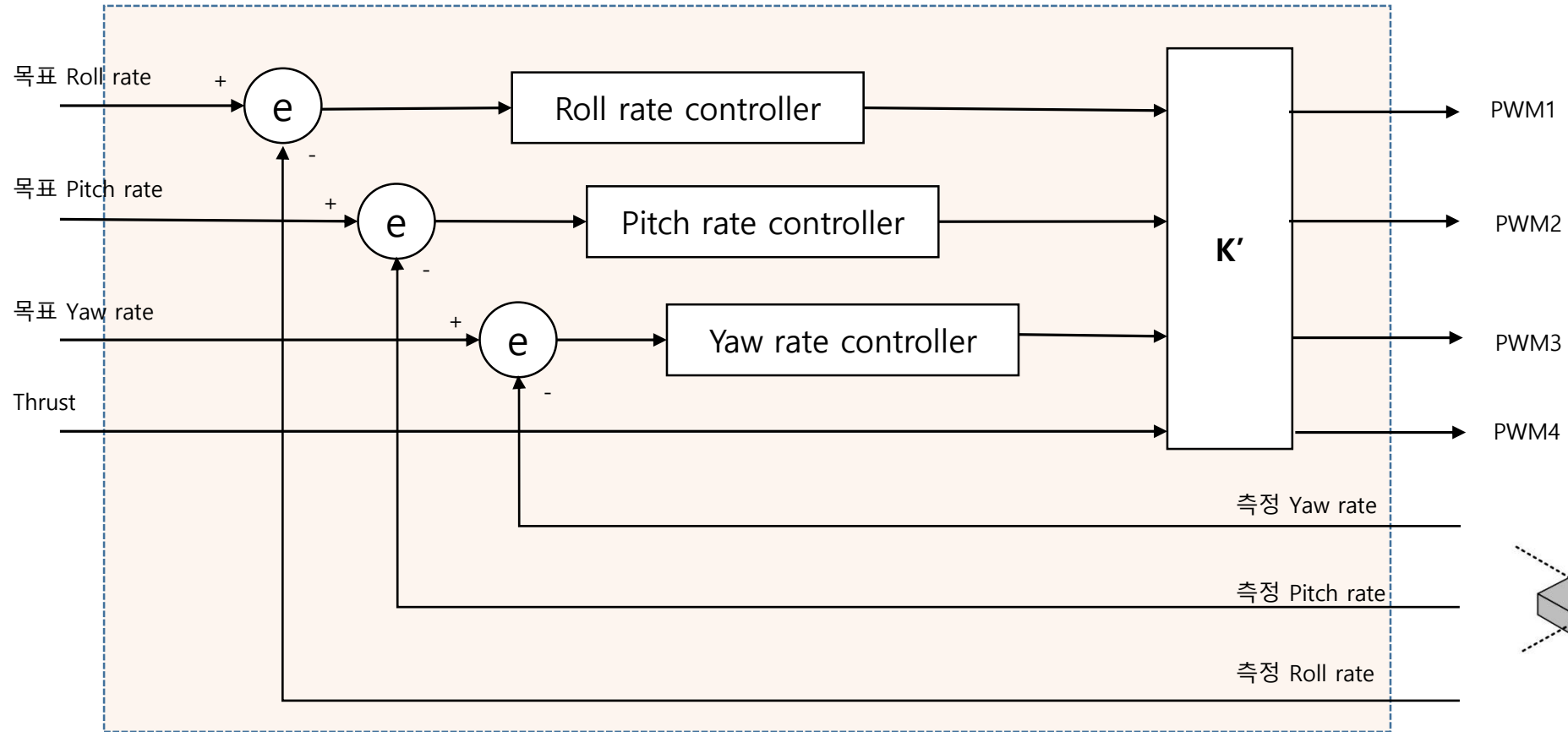
3D에서 Rotation 표현하기

- Quaternion
 - 3D를 복소수를 이용하여 표현
 - 1개 실수와 3개 허수 부분으로 표현
- 참고자료
 - <http://math.ucr.edu/~huerta/introquaternions.pdf>

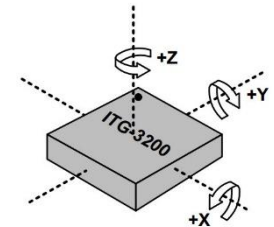
Attitude 제어하기

- Feedback 제어 이용
- RC로부터 목표 Attitude를 지정(T: Target)
 - $(\phi_T, \theta_T, \dot{\psi}_T)$
- 현재 비행체의 Attitude는 Sensor fusion으로 계산
 - $(\phi_M, \theta_M, \dot{\psi}_M)$
- Error(T와 M의 차이) 신호를 PID 제어기로 전달
- 여기서 나온 output이 실제 rate 제어를 위한 target rate가 된다.

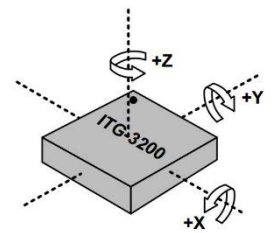
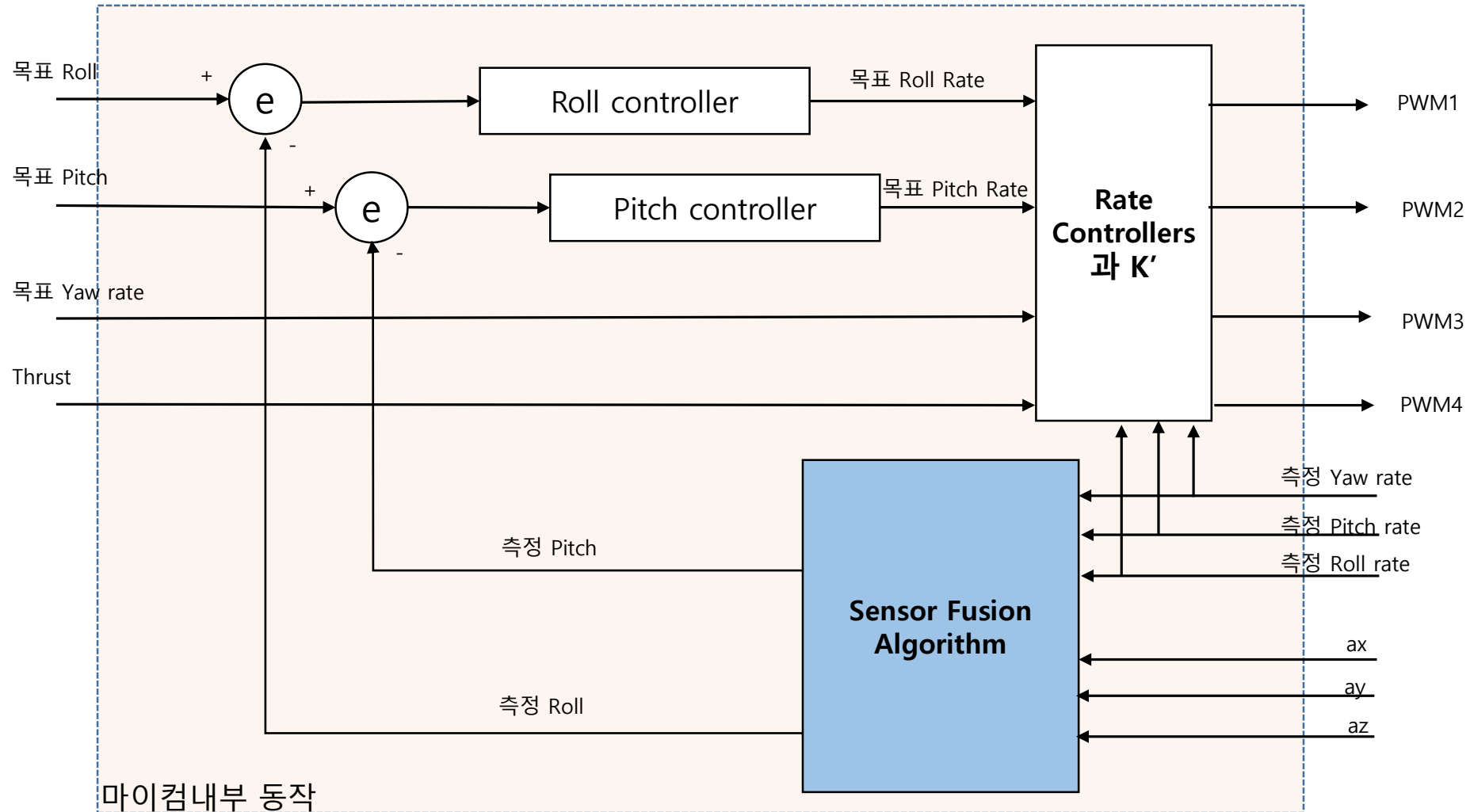
Rate Feedback 제어 전체 구조



마이크로컴 내부 동작



Attitude 제어 구조



Loop 제어

- Loop 종류
 - Rate control (inner)
 - Attitude control (outer)
- Attitude control에 따라 rate control이 결정
- Rate control로 원하는 target에 이르기 위해 시간이 필요
- Loop들은 각기 다른 sampling time을 가진다.
 - T_r = rate control sampling time
 - T_a = attitude control sampling time
 - $T_a \gg T_r$, $T_a = nT_r$ ($n > 1$)
- $T_r = 5\text{ms}$, $T_a = 50\text{ms}$

최종 알고리즘

While

```
( $\dot{\phi}_M, \dot{\theta}_M, \dot{\psi}_M$ ) = Read_gyro();  
( $a_x, a_y, a_z$ ) = Read_accel();  
( $\dot{\phi}_T, \dot{\theta}_T, \dot{\psi}_T, F$ ) = Read_RC_control();  
( $\phi_M, \theta_M$ ) = fusion_filter( $\dot{\phi}_M, \dot{\theta}_M, \dot{\psi}_M, a_x, a_y, a_z$ );  
if N loop 이 후:  
    ( $\phi_T, \theta_T, \dot{\psi}_T, F$ ) = Read_RC_control();  
     $\dot{\phi}_T$  = roll_controller( $\phi_M, \phi_T$ );  
     $\dot{\theta}_T$  = roll_controller( $\theta_M, \theta_T$ );  
end  
 $C_{\dot{\phi}}$  = roll_rate_controller( $\dot{\phi}_M, \dot{\phi}_T$ );  
 $C_{\dot{\theta}}$  = pitch_rate_controller( $\dot{\theta}_M, \dot{\theta}_T$ );  
 $C_{\dot{\psi}}$  = yaw_rate_controller( $\dot{\psi}_M, \dot{\psi}_T$ );  
(pwm1, pwm2, pwm3, pwm4)T = K-1( $C_{\dot{\phi}T}, C_{\dot{\theta}T}, C_{\dot{\psi}T}, F$ )T;  
control_motor(pwm1, pwm2, pwm3, pwm4);
```

end

Altitude와 GPS 제어

- 고도제어
- GPS 제어(위치)

Altitude 제어

- H_T : 타겟 고도
- H_M : 센서가 측정한 고도
- F : 타겟고도를 위한 thrust
- M : altitude 제어를 위한 sampling time

```
while
    ....
    if M loops 이후 :
         $H_M$  = read_altitude_sensor();
         $F$  = altitude_controller( $H_M$ ,  $H_T$ );
    end
    ....
end
```

GPS 제어

- $\text{Lat}_T, \text{Lon}_T$: 타겟 위치
- $\text{Lat}_M, \text{Lon}_M$: 센서가 측정한 위치
- ϕ_T, θ_T : 타겟 위치를 위한 pitch와 roll
- G : GPS제어 sampling time

```
while
    ....
    if G loops 이후 :
        ( $\text{Lat}_M, \text{Lon}_M$ ) = read_GPS();
         $\phi_T$  = GPS_lon_controller( $\text{Lon}_M, \text{Lon}_T$ );
         $\theta_T$  = GPS_lon_controller( $\text{Lat}_M, \text{Lat}_T$ );
    end
    ....
end
```