Linear, Nonlinear cart pole simulation

소속	한서대학교 ACUS
팀원	항공기계학과 오승조 무인항공기학과 이순미
지도교수	이동진

개요

Ⅰ. 서론

1. 연구 목표
2. 연구 내용
3. 제어 흐름도

Ⅱ. 수학적 모델링

1. Nonlinear equation
2. Linear equation
3. State-space
4. ode45
5. Euler method
6. 4th RungeKutta method
7. ode45, Euler, RungeKutta 비교

Ⅲ. PID 제어

1. PID제어 정의
2. PID제어 적용

Ⅳ.Simulation

Ⅰ. 서론

1. 연구 목표

Cart pole 의 움직임을 동역학적으로 해석하여 미분방정식을 세우고 미분방정식을 수치적분 방법으로 적분, 적분한 값을 통해 Matlab을 이용하여 안정적인 PID제어기를 설계한다.

2. 연구 내용

●Cart pole의 움직임을 동역학적으로 해석한다.
- Linear equation
- Nonlinear equation

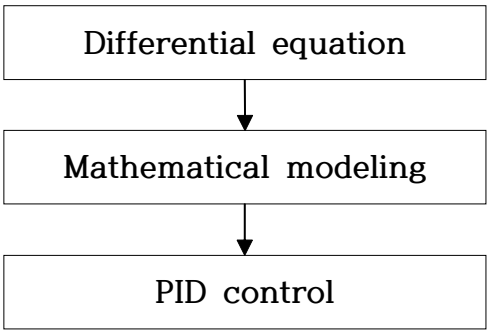
●미분방정식을 통해 수학적으로 모델링한다.
- State-space
- ode45
- Euler method
- 4th Rungekutta method

●수치해석을 통하여 얻은 값을 Matlab을 이용 PID제어를 한다.

●Cart pole의 움직임과 PID제어를 simulation을 통하여 점검한다.

3. 제어 흐름도

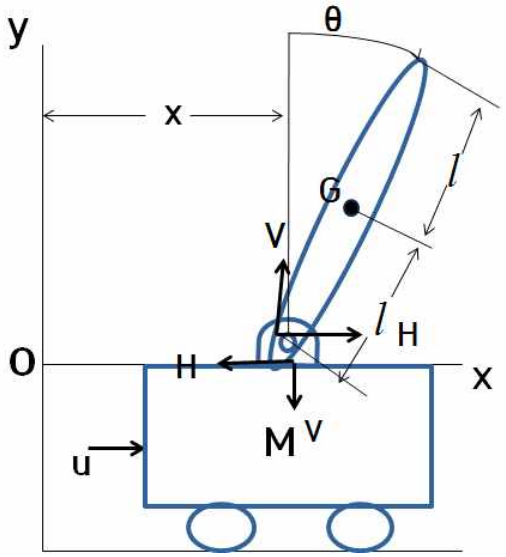
Cart pole 제어는 Closed-Loop system, 즉, output에 대한 feedback을 통하여 안정적인 제어를 하게 된다. 이러한 제어를 하기 위해서는 일련의 과정을 거치게 되는데 그 흐름은 아래와 같다.



Ⅱ.수학적 모델링

제어를 하기 위해서 control 하고자 하는 시스템을 수학적으로 표현하는 미분방정식을 구한다. 미분방정식을 수치적분 하여 PID제어에 적용할 수 있도록 한다.

1. NonLinear equation



위 그림은 Cart pole의 자유물체도 이다.

그림에서 중력이 작용하는 중력중심 G를 x과 y축에서의 좌표 값으로 나타내면

$$G_x = x + L\sin\theta$$

$$G_y = L\cos\theta$$

이다. pendulum의 회전운동을 방정식으로 나타내면

$$I\ddot{\theta} = Vl\sin\theta - Hl\cos\theta$$

이다. pendulum의 중력중심에 대해서 수평으로 작용하는 힘H와 수직으로 작용하는 힘V에 관한 식

$$H = m\frac{d^2}{dt^2}(x + l\sin\theta)$$

$$V = m\frac{d^2}{dt^2}(l\cos\theta) + mg$$

카트에 수평으로 작용하는 힘에 관한 식

$$M\frac{d^2x}{dt^2} = u - H$$

을 얻었다.

회전운동에 관한 방정식에 힘H와 V를 대입, 카트운동에 관한 방정식에 힘H를 대입하여 다음과 같은 식을 얻었다.

$$u = (M+m)\ddot{x} - mL\sin\theta\dot{\theta}^2 + mL\cos\theta\ddot{\theta}$$

$$(I_G + mL^2)\ddot{\theta} + mL\ddot{x}\cos\theta = mgL\sin\theta$$

두 식을 연립하여 식을 변형하여

$$\ddot{x} = \frac{-m^2L^2\,\mathrm{g}\cos\theta\sin\theta + (I+mL^2)mL\sin\theta\dot{\theta}^2}{(m+M)(I+mL^2) - m^2L^2\cos^2\theta} + \frac{I+mL^2}{(m+M)(I+mL^2) - m^2L^2\cos^2\theta}u$$

$$\ddot{\theta} = \frac{(m+M)mgl\sin\theta - m^2L^2\cos\theta\sin\theta\dot{\theta}^2}{(m+M)(I+mL^2) - m^2L^2\cos^2\theta} - \frac{mL\cos\theta}{(m+M)(I+mL^2) - m^2L^2\cos^2\theta}u$$

위와 같은 식을 얻었다.

2. Linear equation

비선형의 미분방정식을 선형화 하기위해 각도θ를 매우 작다고 가정하면, sinθ=θ, cosθ=1 이 된다. 따라서 위의 비선형 미분방정식은 다음과 같이 간략하게 표현할 수 있다.

$$\ddot{x} = \frac{m^2gl^2}{(M+m)(I+ml^2) + m^2l^2}\theta + \frac{I+ml^2}{(M+m)(I+ml^2) + m^2l^2}u$$

$$\ddot{\theta} = \frac{mgl(M+m)}{(M+m)(I+ml^2) + m^2l^2}\theta - \frac{ml}{(M+m)(I+ml^2) + m^2l^2}u$$

3. State-space

동역학적으로 얻은 미분방정식을 수학적으로 모델링하기 위해서 사용할 수 있는 표현은 크게 두 가지, 즉, 상태공간 표현과 전달함수로 나눌 수 있다. 이번 연구에서는 여러 개의 입력 값을 통해 여러 개의 출력 값을 얻는 데 용이한 상태공간(state-space)표현을 택하였다.

아래는 앞에서 얻은 미분방정식을 상태공간 표현으로 나타낸 것이다.

● Nonlinear equation

$$\begin{pmatrix}\dot{x}\\ \ddot{x}\\ \dot{\theta}\\ \ddot{\theta}\end{pmatrix}=\begin{pmatrix}0 & 1 & 0 & 0\\ 0 & 0 & 0 & \dfrac{(I+mL^2)mL\sin\theta\dot{\theta}}{(m+M)(I+mL^2)-m^2L^2\cos^2\theta}\\ 0 & 0 & 0 & 1\\ 0 & 0 & 0 & \dfrac{-m^2L^2\cos\theta\sin\theta\dot{\theta}}{(m+M)(I+mL^2)-m^2L^2\cos^2\theta}\end{pmatrix}\begin{pmatrix}x\\ \dot{x}\\ \theta\\ \dot{\theta}\end{pmatrix}$$

$$+\begin{pmatrix}0\\ \dfrac{I+mL^2}{(m+M)(I+mL^2)-m^2L^2\cos^2\theta}\\ 0\\ \dfrac{-mL\cos\theta}{(m+M)(I+mL^2)-m^2L^2\cos^2\theta}\end{pmatrix}u+\begin{pmatrix}0\\ \dfrac{-m^2L^2\,g\cos\theta\sin\theta}{(m+M)(I+mL^2)-m^2L^2\cos^2\theta}\\ 0\\ \dfrac{(m+M)mgL\sin\theta}{(m+M)(I+mL^2)-m^2L^2\cos^2\theta}\end{pmatrix}$$

$$y=\begin{pmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{pmatrix}\begin{pmatrix}x\\ \dot{x}\\ \theta\\ \dot{\theta}\end{pmatrix}+\begin{pmatrix}0\\ 0\\ 0\\ 0\end{pmatrix}u$$

●Linear equation

4. ode45

$$\begin{pmatrix}\dot{x}\\ \ddot{x}\\ \dot{\theta}\\ \ddot{\theta}\end{pmatrix}=\begin{pmatrix}0 & 1 & 0 & 0\\ 0 & 0 & \frac{m^2gl^2}{(M+m)(I+ml^2)+m^2l^2} & 0\\ 0 & 0 & 0 & 1\\ 0 & 0 & \frac{mgl(M+m)}{(M+m)(I+ml^2)+m^2l^2} & 0\end{pmatrix}\begin{pmatrix}x\\ \dot{x}\\ \theta\\ \dot{\theta}\end{pmatrix}+\begin{pmatrix}0\\ \frac{I+ml^2}{(M+m)(I+ml^2)+m^2l^2}\\ 0\\ -\frac{ml}{(M+m)(I+ml^2)+m^2l^2}\end{pmatrix}u$$

$$y=\begin{pmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{pmatrix}\begin{pmatrix}x\\ \dot{x}\\ \theta\\ \dot{\theta}\end{pmatrix}+\begin{pmatrix}0\\ 0\\ 0\\ 0\end{pmatrix}u$$

Matlab에서는 미분방정식을 수치적분 해주는 기능인 ode45를 제공하고 있다. ode45는 미분방정식의 함수를 입력받아 4,5차 RungeKutta 방법을 이용하여 적분을 하게 된다.

●Matlab에서의 ode45 사용

*미분방정식 함수(Nonlinear equation)

```
function da = nonlinear_fun(t,a);
h=0.001;
t=0:h:10;
%parameter
M=1.0;
m=0.3;
L=0.8;
I=0.004;
g=9.81;
u=0
%v=da(1), v_dot=da(2), w=da(3); w_dot=da(4)
%x=a(1). v=a(2), theta=a(3), w=a(4)
da=zeros(4,1);

da(1)=a(2);
da(3)=a(4);
da(2)=(12.5568*cos(a(3))*sin(a(3))-0.1546*sin(a(3))*a(4)^2)/(0.2548-0.0576*cos(a(3)).^2)
        +(0.1960)/(0.2548-0.0576*cos(a(3)).^2)*u;

da(4)=(3.0607*sin(a(3))- 0.24*cos(a(3))*sin(a(3))*a(4)^2)/(0.2548-0.0576*cos(a(3)).^2)
         +0.24*cos(a(3))/(0.2548-0.0576*cos(a(3)).^2)*u

end
```

*ode45함수 실행문

```
h=0.001;                %fixed step
t=0:h:10;

[t a]=ode45(@nonlinear_fun, t ,[0 0 10*pi/180 0]);
x=a(:,1);
v=a(:,2);
theta=a(:,3);
w=a(:,4);
```

*미분방정식 함수(Linear equation)

```
function da =ode_fun(t,a)
%parameter
M=1.0;
m=0.3;
L=0.8;
I=0.004;
g=9.81;
u=0;
q=((M+m)*(L+m*L^2)-m^2*L^2);

%v=da(1), v_dot=da(2), w=da(3); w_dot=da(4)
%x=a(1). v=a(2), theta=a(3), w=a(4)

da=zeros(4,1);

%da equation
da(1)=a(2);
da(2)=-m^2*L^2*g/q*a(3)+(I+m*L^2)/q*u;
da(3)=a(4);
da(4)=(M+m)*m*g*L/q*a(3)-m*L/q*u;

end
```

*함수 실행문

```
h=0.001;          %fixed step
t=0:h:10;

[t a]=ode45(@nonlinear_fun, t ,[0 0 10*pi/180
0]);
x=a(:,1);
v=a(:,2);
theta=a(:,3);
w=a(:,4);
```

5. Euler method

Matlab 내장기능인 ode45외에도 수치해석적으로 적분하는 방법에는 Euler, RungeKutta방법 등이 있다. Euler방법은 어느 한 점의 기울기와 step크기로 다음 step의 값을 구하는 방법이다.

●Matlab에서의 Euler method 사용

*입력 값을 받고 Euler방법을 통하여 적분을 해주는 함수

```
%Euler

function a= Euler_integrator(A,B,B2,C,X,u,h)

%Equation of state
func= A*X+B*u+B2;
y= C*X;

%Euler
ky = func
a=y+h*ky

end
```

*Euler 함수를 이용하여 미분방정식의 적분(Nonlinear equation)

```
h=0.001;
t=0:h:10;

M=1;
m=0.3;
L=0.8;
I=0.004;
u=0;
g=9.81;

a=zeros(4,length(t));
ky1=zeros(4,length(t));

a(1,1)=0;
a(2,1)=0;
a(3,1)=10*pi/180
a(4,1)=0;

for n=1:(length(t)-1);

%metrix of state
A=[0          1                        0                                        0;
    0          0              0     (0.047*sin(a(3,n))*a(4,n))/(0.2548-0.0576*cos(a(3,n)).^2)
             ;
    0          0                        0                                    1 ;
    0          0              0
(-0.0576*cos(a(3,n))*sin(a(3,n))*a(4,n))/(0.2548-0.0576*cos(a(3,n)).^2)     ;];
% 1 = x, 2=v, 3=theta, 4=w

X=[a(1,n)        ;
      a(2,n)       ;
      a(3,n)       ;
      a(4,n)       ;];

B=[0               ;
   0.1960/(0.2548-0.0576*cos(a(3,n)).^2);
   0               ;
   -0.24*cos(a(3,n))/(0.2548-0.0576*cos(a(3,n)).^2) ;];

B2=[0;
     -0.5651*cos(a(3,n))*sin(a(3,n))/(0.2548-0.0576*cos(a(3,n)).^2);
      0;
      3.0607*sin(a(3,n))/(0.2548-0.0576*cos(a(3,n)).^2  );];

C=[ 1      0      0      0;
      0      1      0      0;
      0      0      1      0;
      0      0      0      1;];


a=Euler_integrator(A,B,B2,C,X,u,h);

end
```

*Euler 함수를 이용하여 미분방정식의 적분(linear equation)

```
h=0.001;
t=0:h:10;

M=1;
m=0.3;
L=0.8;
I=0.004;
u=0
g=9.81;

a=zeros(4,length(t));
ky1=zeros(4,length(t));

a(1,1)=0;
a(2,1)=0;
a(3,1)=10*pi/180;
a(4,1)=0;
```

```
for n=1:(length(t)-1);
%metrix of state
A=[0          1                    0                    0;
   0          0          -m^2*L^2*g/q                   0;
   0          0                    0                    1;
   0          0          (M+m)*m*g*L/q                  0;];

% 1 = x, 2=v, 3=theta, 4=w
X=[a(1,n)       ;
     a(2,n)     ;
     a(3,n)     ;
     a(4,n)     ;];

B=[0            ;
  (I+m*L^2)/q   ;
  0             ;
  -m*L/q        ;];

B2=[0;
     0;
     0;
     0;];

C=[ 1     0     0     0;
     0     1     0     0;
     0     0     1     0;
     0     0     0     1;];


a=Euler_integrator(A,B,B2,C,X,u,h);

end
```

6. RungeKutta method

Euler방법 보다 더 정확한 수치적분을 하기 위해 고안된 방법으로써 한 점의 기울기와 step 크기로 다음 값을 구하는 Euler방법과는 달리 연구에서 사용된 4차 RungeKutta방법은 네 개의 점의 기울기를 이용하여 더욱 정확한 값을 구해낸다.

●Matlab에서의 RungeKutta method 사용

*입력 값을 받고 RungeKutta방법을 통하여 적분을 해주는 함수

```
%RK4

function a= RK4_integrator(A,B,B2,C,X,u,h)

%Equation of state
func= A*X+B*u+B2;
y= C*X;

%RK4

ky1 = func
ky2 = func+0.5*h*ky1;
ky3 = func+0.5*h*ky2;
ky4 = func+h*ky3;

a=y+h*(ky1+2*ky2+2*ky3+ky4)/6;


end
```

*RungeKutta 함수를 이용하여 미분방정식의 적분(Nonlinear equation)

```
h=0.001;
t=0:h:10;

M=1;
m=0.3;
L=0.8;
I=0.004;
u=0;
g=9.81;

a=zeros(4,length(t));
ky1=zeros(4,length(t));

a(1,1)=0;
a(2,1)=0;
a(3,1)=10*pi/180
a(4,1)=0;

for  n=1:(length(t)-1);

%metrix  of  state
A=[0          1                          0                                        0;
    0          0              0
(0.047*sin(a(3,n))*a(4,n))/(0.2548-0.0576*cos(a(3,n)).^2)                    ;
    0          0                          0                                  1 ;
    0          0              0
(-0.0576*cos(a(3,n))*sin(a(3,n))*a(4,n))/(0.2548-0.0576*cos(a(3,n)).^2)     ;];

 %  1  =  x,  2=v,  3=theta,  4=w

X=[a(1,n)        ;
    a(2,n)        ;
    a(3,n)        ;
    a(4,n)        ;];

B=[0               ;
   0.1960/(0.2548-0.0576*cos(a(3,n)).^2);
   0               ;
   -0.24*cos(a(3,n))/(0.2548-0.0576*cos(a(3,n)).^2)  ;];

B2=[0;
    -0.5651*cos(a(3,n))*sin(a(3,n))/(0.2548-0.0576*cos(a(3,n)).^2);
     0;
     3.0607*sin(a(3,n))/(0.2548-0.0576*cos(a(3,n)).^2  );];

C=[  1      0      0      0;
     0      1      0      0;
     0      0      1      0;
     0      0      0      1;];


a=RK4_integrator(A,B,B2,C,X,u,h);

end
```

*Rungekutta 함수를 이용하여 미분방정식의 적분(Linear equation)

```
h=0.001;
t=0:h:10;

M=1;
m=0.3;
L=0.8;
I=0.004;
u=0
g=9.81;

a=zeros(4,length(t));
ky1=zeros(4,length(t));

a(1,1)=0;
a(2,1)=0;
a(3,1)=10*pi/180
a(4,1)=0;

for  n=1:(length(t)-1);

%metrix  of  state
A=[0          1                    0                    0;
    0          0              -m^2*L^2*g/q               0;
    0          0                    0                    1;
    0          0              (M+m)*m*g*L/q              0;];

%  1  =  x,  2=v,  3=theta,  4=w

X=[a(1,n)       ;
      a(2,n)       ;
      a(3,n)       ;
      a(4,n)       ;];

B=[0              ;
   (I+m*L^2)/q  ;
   0              ;
   -m*L/q         ;];

B2=[0;
      0;
      0;
      0;];


C=[  1      0      0      0;
      0      1      0      0;
      0      0      1      0;
      0      0      0      1;];


a=RK4_integrator(A,B,B2,C,X,u,h);

end
```

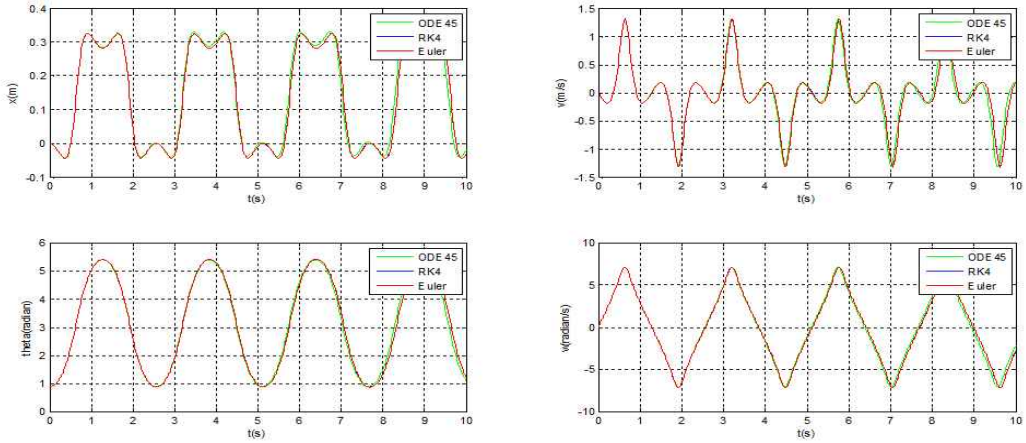
7. ode45, Euler, RungeKutta 비교

Euler방법은 ode45와 RungeKutta방법보다 정확도가 떨어진다. 하지만 step의 크기가 작을수록 3개의 오차는 감소한다.

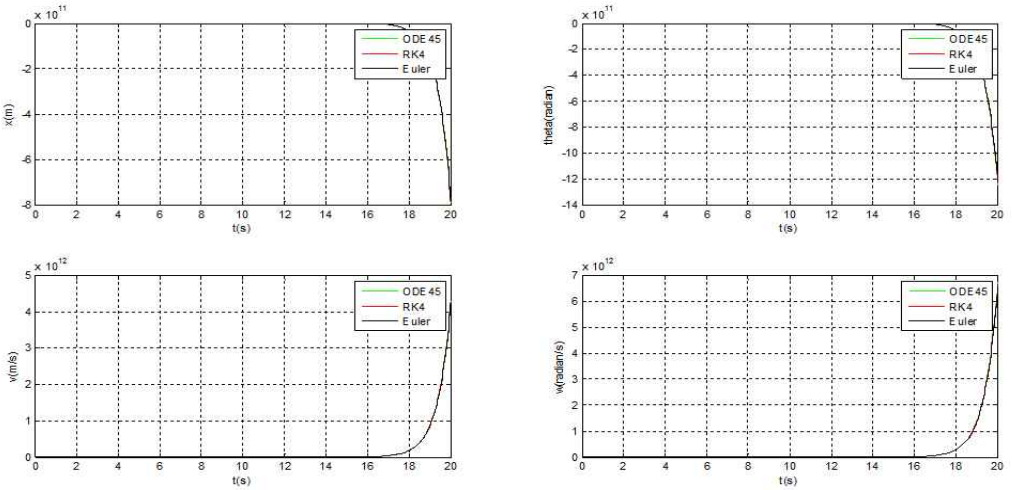
●Nonlinear equation 비교

아래의 그래프는 stepsize t가 0.0001일 때의 그래프이다.

ode45와 4차 RungeKutta 방법이 Euler방법에 비해 상대적으로 정확한 값을 얻어내지만 stepsize가 작을수록 3개의 그래프의 오차는 감소한다.



●Linear equation 비교



Ⅲ. PID 제어

1. PID제어 정의

자동 제어 알고리즘 중의 한 가지 방법으로, 제어 대상의 출력 값과 설정 값과의 오차를 이용하여 제어 값을 계산해내는 방식이다.

PID제어를 통한 조작 량은 아래와 같다.

$$u(t) = K_p e(t) + K_i \int e(t)dt + K_d \frac{de}{dt}$$

K_p : Proportional gain

K_i : Integral gain

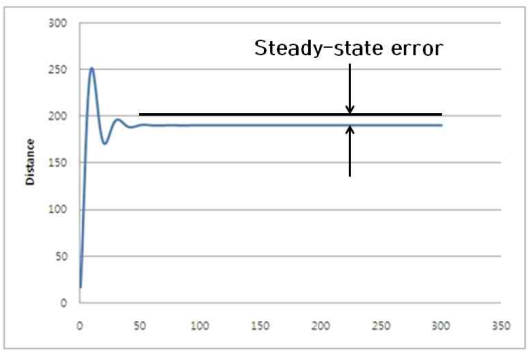
K_d : Derivative gain

●P control

오차(목표 값-실제 값)에 비례해서 제어 량을 조절하는 제어 방식으로 비례 제어라고 한다. 목표 값에 도달하는 시간을 줄일 수 있는 장점이 있다.

●PI control

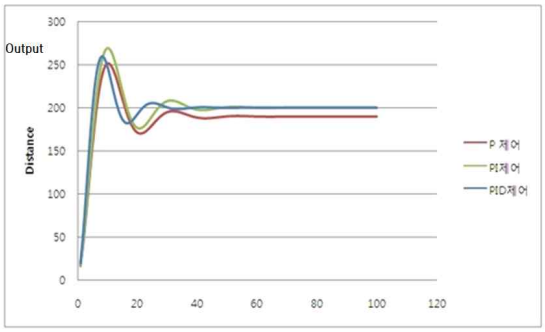
P(비례) 제어가 제어할 수 없는 정상 상태 오차는 그 크기가 너무 작기 때문인데 이를 잔류편차라 한다. 적분 제어는 이런 잔류편차를 시간단위로 적분하여 일정크기의 오차 값으로 읽은 다음, 조작 량을 증가시켜주는 방식이다.
이와 같이 비례 제어와 적분 제어를 합한 방식을 PI제어라 부른다.



●PID control

PI의 제어에 미분제어를 더할 경우 보다 빠르게 목표 값에 도달할 수 있다. 미분제어란 오차

의 변화량을 측정함으로서 조작 량의 조절하는 방식이다. 즉 오차의 변화의 반대방향으로 제어가 이루어지게 된다. 외란이 발생할 경우에도 신속히 대응할 수 있게 한다.



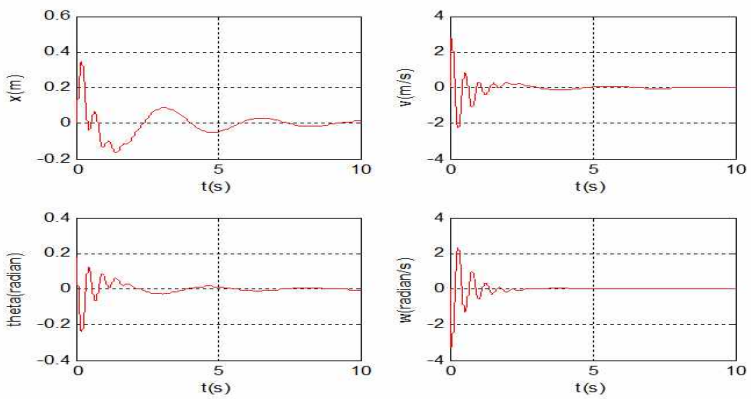
2. PID제어 적용

●Nonlinear equation

```
h=0.001;
t=0:h:10;

M=1;
m=0.3;
L=0.8;
I=0.004;
u=0;
g=9.81;

a=zeros(4,length(t));
ky1=zeros(4,length(t));

a(1,1)=0;
a(2,1)=0;
a(3,1)=10*pi/180
a(4,1)=0;

% Control Loop parameter
aim=0;
kp1=-50;                                    %-0.1,0,0.1     -200,0,-10
ki1=0;
kd1=-10;

kp2=-200;
ki2=0;
kd2=-12;

integral_x=0;
integral_th=0;
prev_err_x=0;
prev_err_th=0;
```

```
for n=1:(length(t)-1);

%metrix of state
A=[0          1                          0                                        0;
    0          0              0
(0.047*sin(a(3,n))*a(4,n))/(0.2548-0.0576*cos(a(3,n)).^2)                    ;
    0          0                          0                                  1 ;
    0          0              0
(-0.0576*cos(a(3,n))*sin(a(3,n))*a(4,n))/(0.2548-0.0576*cos(a(3,n)).^2)     ;];

 % 1 = x, 2=v, 3=theta, 4=w

X=[a(1,n)        ;
    a(2,n)        ;
    a(3,n)        ;
    a(4,n)        ;];

B=[0                ;
   0.1960/(0.2548-0.0576*cos(a(3,n)).^2);
   0                ;
   -0.24*cos(a(3,n))/(0.2548-0.0576*cos(a(3,n)).^2) ;];

B2=[0;
    -0.5651*cos(a(3,n))*sin(a(3,n))/(0.2548-0.0576*cos(a(3,n)).^2);
     0;
     3.0607*sin(a(3,n))/(0.2548-0.0576*cos(a(3,n)).^2 );];

C=[ 1     0     0     0;
     0     1     0     0;
     0     0     1     0;
     0     0     0     1;];


a=RK4_integrator(A,B,B2,C,X,u,h);     %또는 a= Euler_integrator(A,B,B2,C,X,u,h);

%Cotrol loop
err_x=aim-a(1,n);
err_th=aim-a(3,n);                                    %update error
kp_term=kp1*err_x+kp2*err_th;

integral_x= integral_x+err_x*h ;
integral_th= integral_th+err_th*h    ;                %integral term
ki_term=ki1*integral_x+ki2*integral_th;

derivative_x =  (err_x-prev_err_x)/h;
derivative_th = (err_th-prev_err_th)/h;              % derivative term
kd_term=kd1*derivative_x+kd2*derivative_th;

prev_err_x=err_x;
prev_err_th=err_th;

u =kp_term+ki_term+kd_term;                  % action of control

end
```

●Nonlinear equation 그래프

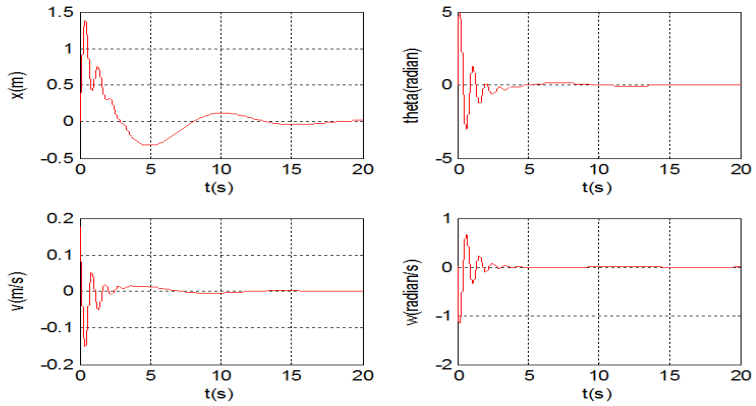


●Linear equation

```
h=0.001;
t=0:h:10;

M=1;
m=0.3;
L=0.8;
I=0.004;
u=0
g=9.81;

a=zeros(4,length(t));
ky1=zeros(4,length(t));

a(1,1)=0;
a(2,1)=0;
a(3,1)=10*pi/180
a(4,1)=0;

%Control Loop parameter
aim=0;
kp1=10;
ki1=0.01;
kd1=-0.05;
kp2=300;
ki2=100;
kd2=15;
integral_x=0;
integral_th=0;
prev_err_x=0;
prev_err_th=0;

for n=1:(length(t)-1);

%metrix of state
A=[0          1                    0                    0;
   0          0          -m^2*L^2*g/q                   0;
   0          0                    0                    1;
   0          0          (M+m)*m*g*L/q                  0;];

% 1 = x, 2=v, 3=theta, 4=w

X=[a(1,n)       ;
     a(2,n)       ;
     a(3,n)       ;
     a(4,n)       ;];

B=[0             ;
  (I+m*L^2)/q    ;
  0              ;
  -m*L/q         ;];

B2=[0;
     0;
     0;
     0;];

C=[ 1     0     0     0;
     0     1     0     0;
     0     0     1     0;
     0     0     0     1;];


a=RK4_integrator(A,B,B2,C,X,u,h);       %또는 a= Euler_integrator(A,B,B2,C,X,u,h);
```

```
%Control loop
err_x=aim-a(1,n);
err_th=aim-a(3,n);
kp_term=kp1*err_x+kp2*err_th;

integral_x= integral_x+err_x*h ;
integral_th= integral_th+err_th*h
ki_term=ki1*integral_x+ki2*integral_th;

derivative_x =  (err_x-prev_err_x)/h;
derivative_th = (err_th-prev_err_th)/h;
kd_term=kd1*derivative_x+kd2*derivative_th;

prev_err_x=err_x;
prev_err_th=err_th;

u =-(kp_term+ki_term+kd_term);

end
```

●Linear equation 그래프



Ⅳ.Simulation

동영상 파일첨부