0. Introduction

1) Numerical Derivative

수치 미분(Numerical Derivative)는 수치해석 기법 중 하나이다. 본래 미분에 대한 정의는 연속함수에서 x와 y의 극한에서 변화량이다. 하지만 디지털 시그널은 특정 주기로 를 또는 등 특정 점에서 표현된다. 또한 Digital signal의 특징은 양자화(Quantization)으로 연속함수가 아닌 컴퓨터가 인식할 수 있는 수로 처리된다. 따라서 이러한 디지털 신호의 변화량에 대해서는 유한 차분법을 사용하여 data points의 미분 상태를 추정한다. 이 때 데이터의 차를 구하고 데이터와 데이터 간 사이 Sampling time으로 나누어 미분을 수치적으로 근사하는 기법이 주로 사용된다. 에서 기인할 수 있듯이 2-point central difference는

로 나오게 된다.

2-point backward 기법은 실시간에서 반영할 수 있는 기법으로 현재 값과 바로 전 값에 대한 차이를 구하고 그 두 값의 차이가 발생하기까지 걸린 시간으로 나누어 수치 미분을 하는데 그때의 식은 으로 표현할 수 있다.

2) Median Filter

Median의 정의는 주어진 표본을 순서대로 정렬하였을 때 배열 가장 중앙에 있는 값을 의미한다.

Median Filter는 디지털 신호 처리 기법이며 주로 이미지나 신호의 노이즈를 처리할 때 쓰인다. 배열에서 해당 인덱스 주위에 있는 배열까지 Filter window가 적용되고 그 중에 중간 값을 해당 값을 현 해당 값에 할당하는 방식으로 작동되며 필터가 움직이면서 그 마다 해당하는 중간 값을 해당 인덱스에 할당하는 방식이다. 이 방식을 사용하는 이점은 그래프의 형상을 보존하며 노이즈를 제거할 수 있다는 것이 장점이다. 또한 극대점 또는 극소점을 줄여 폭이 작은 그래프를 나타낼 수 있다.

3) Derivative using Pulse Frequency

요약) 엔코더의 각속도를 산출해내는 방법 3가지에 대해 조사하고 자신의 언어로 요약하세요.

첫번째 방법은 M-method로 단위시간 당 몇 번의 Pulse frequency가 나오는지 측정하여 각속도로 변환한다. 단위시간 T 동안 몇 개의 파형이 나오는지 count하는 알고리즘이 갖춰져야 하며 총 한바퀴 당 Pulse의 개수에 대해 파악하고 있어야 계산을 할 수 있다.  
표준 각속도 공식인 에 기반하여 엔코더의 각속도 공식을 로변형해 쓸 수 있다. 이때 는 해당 시간동안 발생한 Pulse와 한 바퀴 당 Pulse개수의 비율을 라디안으로 바꾸는 공식이며 해당 시간 T로 나눠주어 각속도 공식을 만족하는 식을 세운다. 현재 수업에서 사용하는 엔코더는 한바퀴 당 24개의 Pulse가 나오고 만약 1초 당 24번의 파형이 측정됬다면 그 시간의 각도 변화율은 이 된다. 다만 이 방식은 T가 짧고 그 사이에 알맞은 크기의 Pulse개수가 측정되는 고속 주행에서 유리한 기법이다.

두번째 방법인 T-method는 Pulse와 Pulse 사이의 시간을 측정하여 각속도를 측정하는 기법이다.T-method는 별도의 시간을 재는 알고리즘과 한 펄스 당 각도를 아는 상태에서 펄스와 펄스 사이의 간격을 세고 펄스 당 각도를 그 간격으로 나눠 각속도를 산출하는 방식이다. 이는 시간 Counter 보다 더 빠른 주파수를 잡지 못한다는 단점이 있기 때문에 저속에서 많이 사용된다. T-method는 로 이때 는 한 펄스 당 몇 라디안인지, T는 펄스와 펄스 사이 걸린 시간이다.

마지막 방법인 MT- method는 M-method에서 발생할 수 있는 문제와 T-method에서 발생할 수 있는 문제를 보완하기 위한 방법으로 MT-method로 M-method의 단위의 low frequency문제와 T-method의 카운터의 frequency 사양을 보완하기 위해 두 기법을 동시에 사용하는 기법이다.   
M-method의 와 이후에 첫 Pulse가 측정되기까지 걸리는 시간()을 T-method의 카운터로 측정해 포괄적인 와 총 Pulse 수를 결합하여 엔코더의 각속도를 측정하는 방법이다. 이 기법은 M-method와 T-method를 결합하여 결과의 정확도와 총 펄스당 걸리는 시간을 비교적 정밀적으로 해석할 수 있다.

도표이(가) 표시된 사진

자동 생성된 설명

그림 MT-method

1. Rotational angular velocity using potentiometer

1) 시간에 대한 각도 그래프를 실시간으로 그려보세요.

- Sample rate: 1000[Hz]  
- Duration time: 20[s]

텍스트, 편지이(가) 표시된 사진

자동 생성된 설명텍스트이(가) 표시된 사진

자동 생성된 설명

그림 선형회귀식 그림 선형회귀식에 사용된 전압과 각도

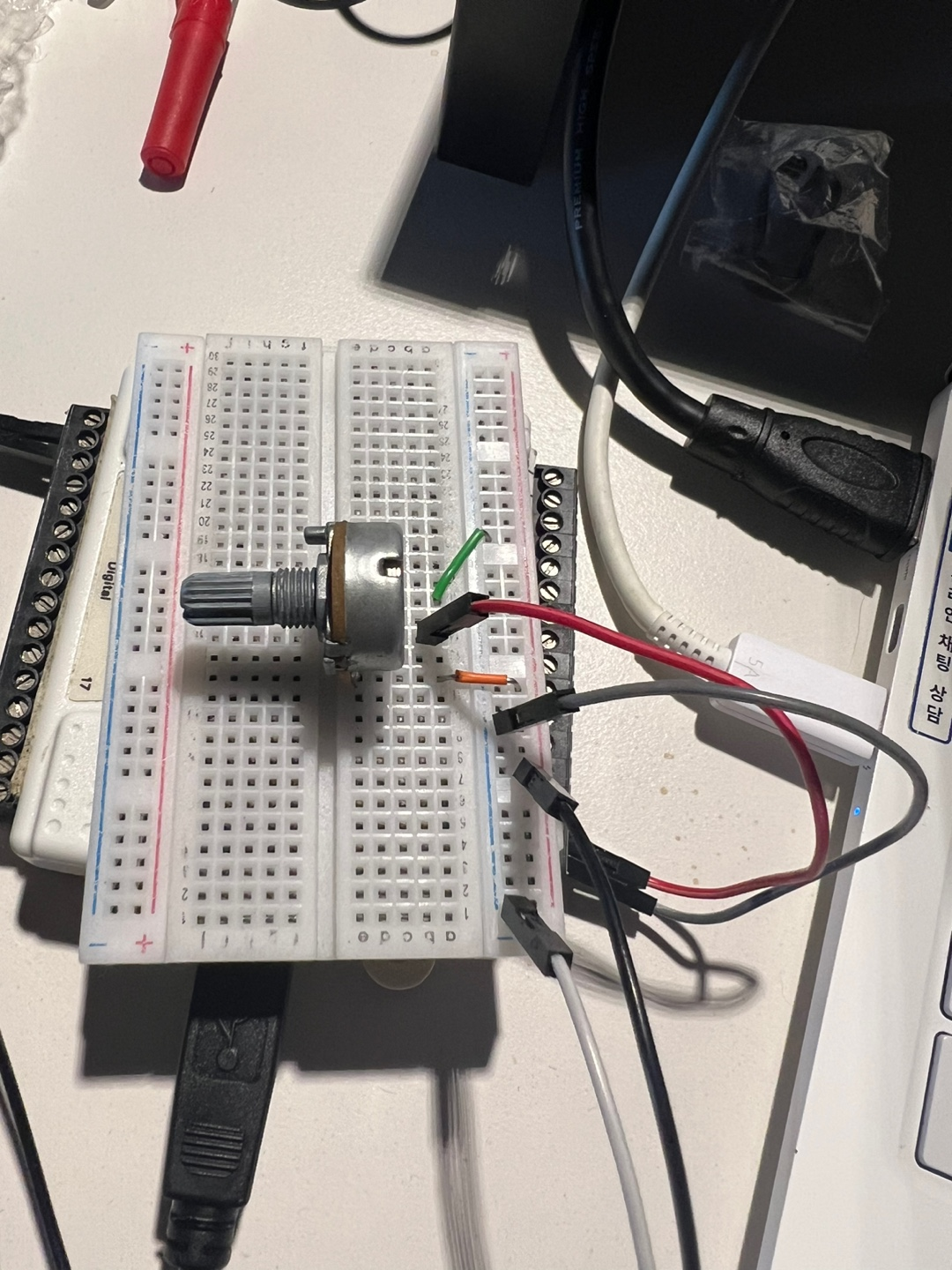


그림 가변저항을 연결한 회로

|  |  |
| --- | --- |
| 코드 | 설명 + 주석 |
| Coeff1 = 50.4005105991401; %Curve fit P1 계수  Coeff2 = -130.389997882329; %Curve fit P2 계수  if isempty(data\_stack)  data\_stack = event.Data;  time\_stack = event.TimeStamps;  angle\_stack= Coeff1.\*data\_stack + Coeff2;  else  data\_stack = [data\_stack; event.Data]; %Voltage  time\_stack = [time\_stack; event.TimeStamps]; %Times  angle\_stack= Coeff1.\*data\_stack +Coeff2;  end  plot(time\_stack,angle\_stack);  xlabel('Time [sec]');  ylabel('Angle [degree]');  title('Potentiometer degree'); | 가변저항의 전압과 각도의 관계는 일정한 간격을 두고 각도기로 가변저항의 각도를 잰 후, 그 각도일 때 가변저항에 걸린 전압의 관계를 선형회귀식으로 모델링하여 나타낸다.  Data\_stack에는 전압이 나타나므로 전압에서 각도로 바꿀 수 있는 선형회귀 계수를 대입하여 각도를 산출하는 알고리즘이다. |
| 결과 | 설명 |
|  | 처음 가변저항의 각도가 약 90도에서 시작하여 그 전후로 왔다갔다하는 것이 관찰되었다. |

2) 수치 미분을 통해 각속도 그래프를 실시간으로 그려보세요.

답)

|  |  |
| --- | --- |
| 코드 | 설명 + 주석 |
| Coeff1 = 50.4005105991401; %Curve fit P1 계수  Coeff2 = -130.389997882329; %Curve fit P2 계수  N = 9; %The filter size should not exceed the length of event.Data  dt= 1/1000  if isempty(data\_stack)  data\_stack = event.Data; %Voltage  time\_stack = event.TimeStamps;  angle\_stack= Coeff1.\* data\_stack + Coeff2; %Angle  AVel\_stack=zeros(length(event.Data),1); %Angular Velocity  for i=2:length(event.Data)  AVel\_stack(i,1)= (angle\_stack(i,1)-angle\_stack(i-1,1)) /dt; %Angular Velocity  end  else  data\_stack = [data\_stack; event.Data]; %Voltage  time\_stack = [time\_stack; event.TimeStamps]; %Times  angle\_stack= Coeff1.\*data\_stack +Coeff2; %Angle length(angle\_stack)==length(data\_stack)  k=zeros(length(event.Data),1); %Temporary Angular velocity  k(1,1)= (Coeff1\*event.Data(1,1)+Coeff2 - angle\_stack(length(angle\_stack)-length(event.Data),1))/dt;  for i=2:length(event.Data)  k(i,1)= (angle\_stack(length(angle\_stack) -length(event.Data) +i ,1)- angle\_stack(length(angle\_stack) -length(event.Data) +i-1 ,1)) /dt;  end  AVel\_stack=[AVel\_stack; k]; %Angular Velocity  end  plot(time\_stack,AVel\_stack);  xlabel('Time [sec]');  ylabel('Angular Velocity [deg/s]');  title('Angular velocity of Potentiometer (Unfiltered)'); | 먼저 전압으로 각도를 얻는 선형회귀식으로 각도를 구한다. 전압으로 산출된 각도의 변화율은 2-point backward기법을 쓴다. 이때 바로 전 값이 필요하기 때문에 첫번째는 0, 두번째부터 수치 미분 값이 들어가게 된다. |
| 결과 | 설명 |
|  | 파형이 크게 튀는 부분이 있어서 Scale이 크게 관찰되었다. 필터를 거치지 않았기 때문에 노이즈가 큰 것을 확인할 수 있었다. |

3) Median Filter를 사용하여 잡음을 제거해보세요.  
- Filter size: 5, 10, 15, 20

|  |  |
| --- | --- |
| 코드 | 설명 + 주석 |
| global data\_stack time\_stack angle\_stack AVel\_stack mAngVel    Coeff1 = 50.4005105991401; %Curve fit P1 계수  Coeff2 = -130.389997882329; %Curve fit P2 계수  N = 9; %The filter size should not exceed the length of event.Data  dt= 1/1000  if isempty(data\_stack)  data\_stack = event.Data; %Voltage  time\_stack = event.TimeStamps;  angle\_stack= Coeff1.\* data\_stack + Coeff2; %Angle  AVel\_stack=zeros(length(event.Data),1); %Angular Velocity  for i=2:length(event.Data)  AVel\_stack(i,1)= (angle\_stack(i,1)-angle\_stack(i-1,1)) /dt; %Angular Velocity  end  if length(event.Data)<N  for i=1:length(event.Data)  mAngVel(i,1)= AVel\_stack(i,1);  end  else  for i=1:N-1  mAngVel(i,1)= AVel\_stack(i,1); %Median Filtered Angular Velocity  end  end  for i=N:length(event.Data)  temp=AVel\_stack((i-N+1):i,1); %Temporary Angular Velocity  tempsort=sort(temp); %Sorting angular velocity Array  mAngVel(i,1)= tempsort(ceil(N/2)); %Extracting Median  end  else  data\_stack = [data\_stack; event.Data]; %Voltage  time\_stack = [time\_stack; event.TimeStamps]; %Times  angle\_stack= Coeff1.\*data\_stack +Coeff2; %Angle length(angle\_stack)==length(data\_stack)  k=zeros(length(event.Data),1); %Temporary Angular velocity  k(1,1)= (Coeff1\*event.Data(1,1)+Coeff2 - angle\_stack(length(angle\_stack)-length(event.Data),1))/dt;  for i=2:length(event.Data)  k(i,1)= (angle\_stack(length(angle\_stack) -length(event.Data) +i ,1)- angle\_stack(length(angle\_stack) -length(event.Data) +i-1 ,1)) /dt;  end  AVel\_stack=[AVel\_stack; k]; %Angular Velocity  for i=length(AVel\_stack)-length(event.Data)+1:length(AVel\_stack)  temp=(AVel\_stack(i-N+1:i,1)); %Temporary Angular Velocity Array  tempsort= sort(temp); %Sorting Array  mAngVel(i,1) = tempsort(ceil(N/2),1); %Extracting Median  end  end  plot(time\_stack, mAngVel);  xlabel('Time [sec]');  ylabel('Angular velocity [degree/s]');  title('Filtered Angular velocity of potentiometer'); | 먼저 전압과 각도의 관계식으로 각도를 구한다. 그 각도를 토대로 각속도를 구한다. 그 각속도에 관한 Median filter를 현 코드에서 설계하였다. 먼저 Filter size보다 작은 Index일 경우에는 필터를 씌우지 않고 그대로 각속도 값을 반영한다. 그 이후에 받는 각속도 인덱스는 각속도 Raw-data의 현 인덱스부터 앞의 Index들을 크기로 Sorting 시키는 배열을 만들어서 그 가운데 값을 현 Median-Filter 인덱스(mAngVel)에 할당한다. Median Filter을 사용하는 범위는 필터 사이즈에 해당하는 인덱스부터 마지막 인덱스까지 이다. |
| 결과 | 설명 |
|  | 본래 필터를 거치기 전의 극대점 또는 극소점 같은 노이즈가 없어지고 편차가 확연히 줄어든 것을 확인할 수 있었다. |