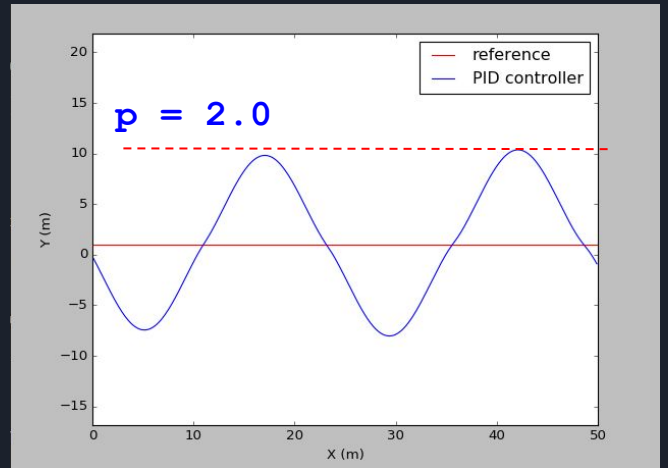
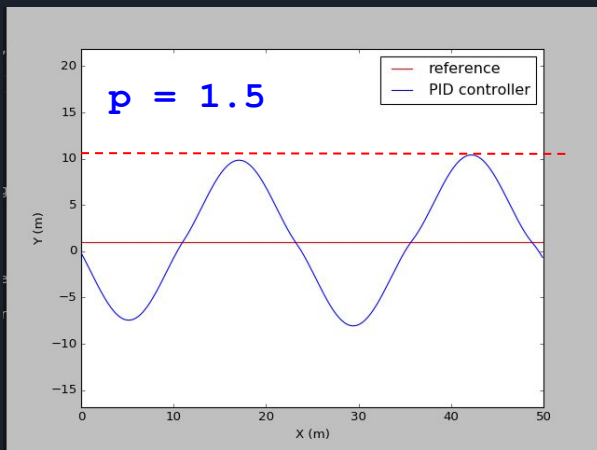
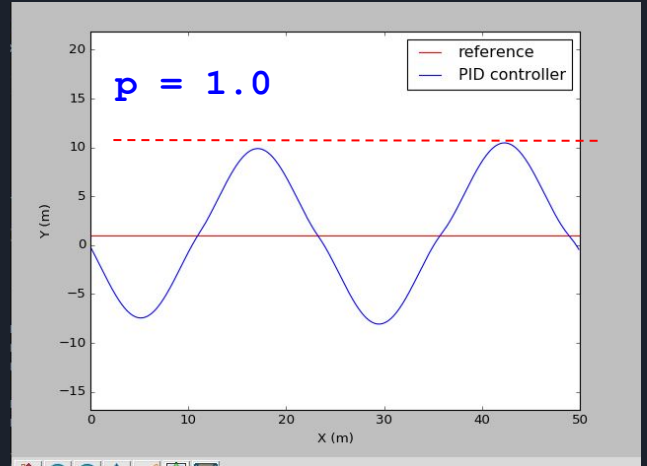
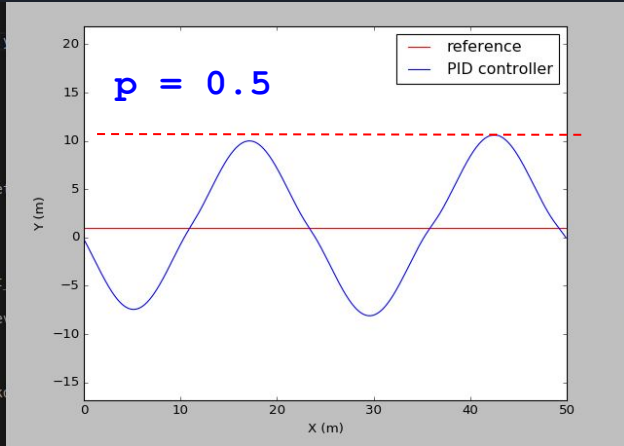




# PID 제어 프로그래밍과제

프로그래머스 자율주행 코스 1기  
조정민

# P 제어



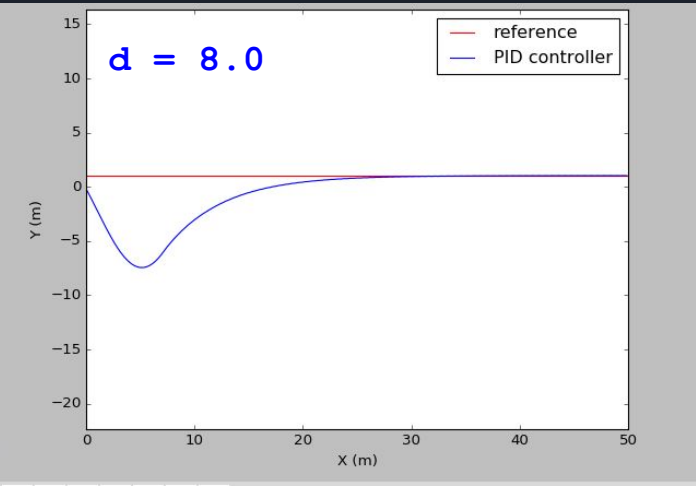
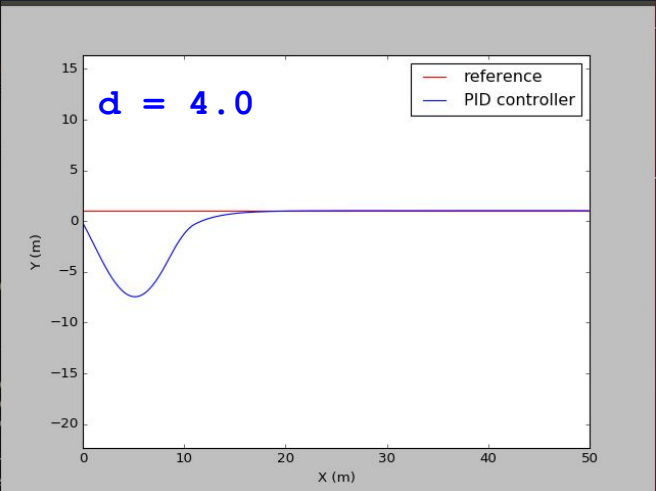
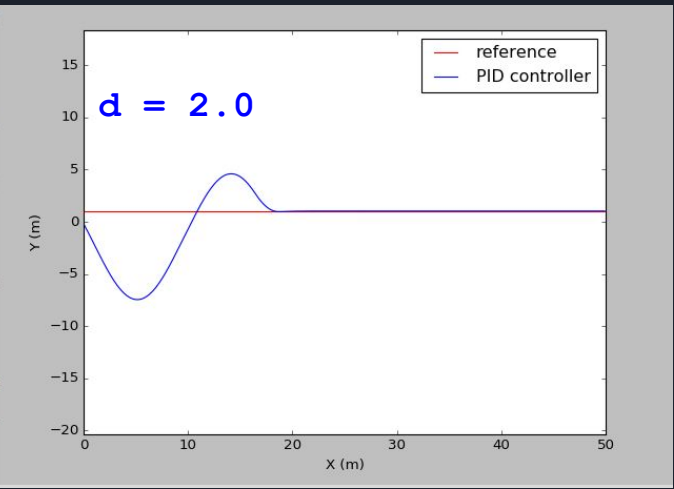
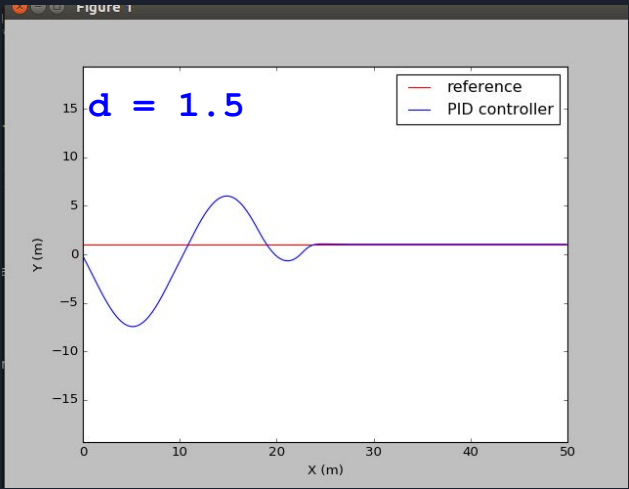
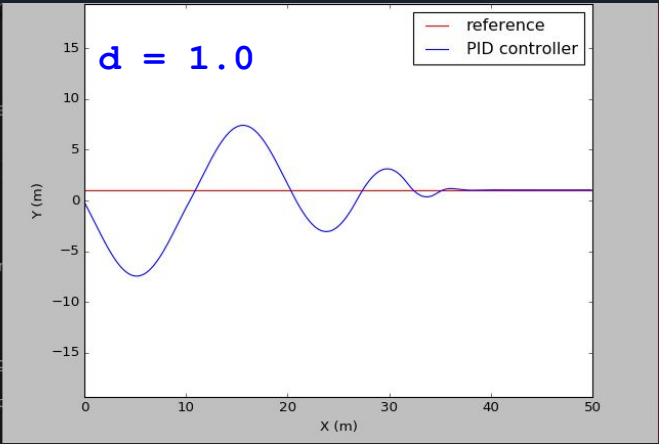
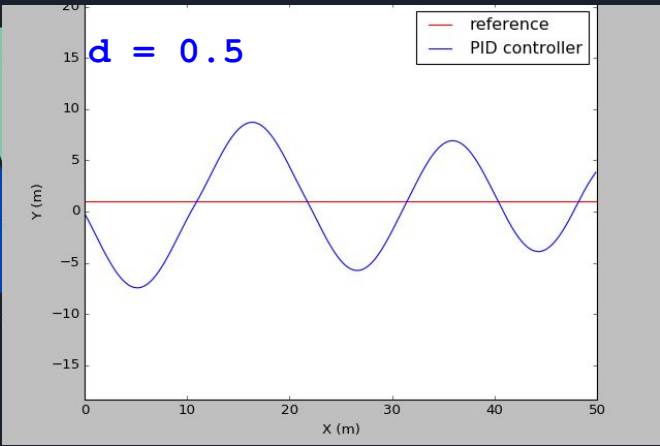
P gain값을 통해서 오차에 비례하는 값을 control output 값에 대입해서 진행한다.

이전, 이후 값에는 상관없이 output을 생산하기 때문에 해당 경로를 주행할 수는 있지만, 오차가 점점 나아지지 않고, 반복적인 형태를 띈다.

약간 그때그때마다 대처하는 느낌!

P gain값은 모두 비슷해보이지만, 1.5 값일때 가장 변동폭이 좁은것으로 보아 p값이 1.5 일때 최적이라고 판단했다.

# PD 제어



D control을 이용하면 에러가 줄어들고 있다는것을 컨트롤러한테 알려주기 때문에 Count Steer를 치는게 가능하다.

d값이 4.0 이상이되면 과도하게 부드러워? 지는 경향으로 인해 오히려 오차가 줄어드는 구간이 멀어진다.

# PID 제어

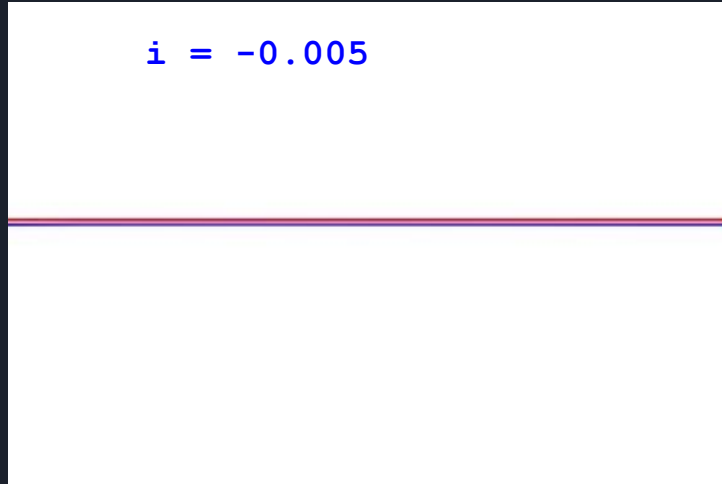
$i = 0.0$

A plot showing a horizontal line at the center of the y-axis, representing a constant value. The line is black, and the background is white.

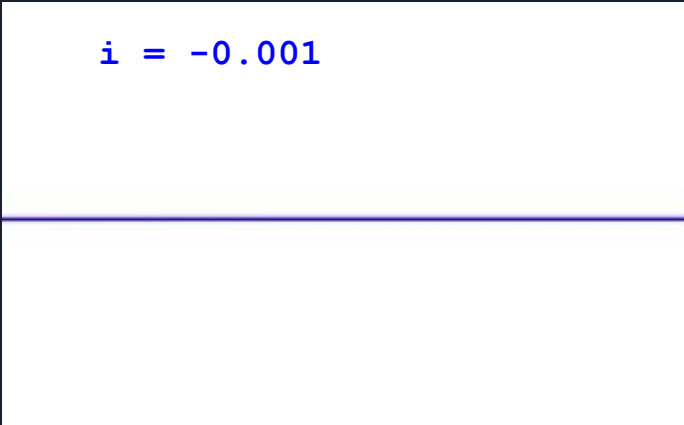
$i = 0.001$

A plot showing a horizontal line slightly above the center of the y-axis, representing a constant value. The line is black, and the background is white.

$i = -0.005$

A plot showing a horizontal line slightly below the center of the y-axis, representing a constant value. The line is black, and the background is white.

$i = -0.001$

A plot showing a horizontal line slightly below the center of the y-axis, representing a constant value. The line is black, and the background is white.

PD control을 이용 에러가 Constant하게 유지가 되었을때, integral을 하면 해당 오차를 줄일 수 있다.

i gain 값이 0.00 일때 실제 주행 값이 좀더 위로 위치하는것을 볼 수 있다.  
i gain 값이 0.001로 늘렸을때 좀더 위로 올라가는것을 확인하고 i 값을 -0.001로 결정하였다.

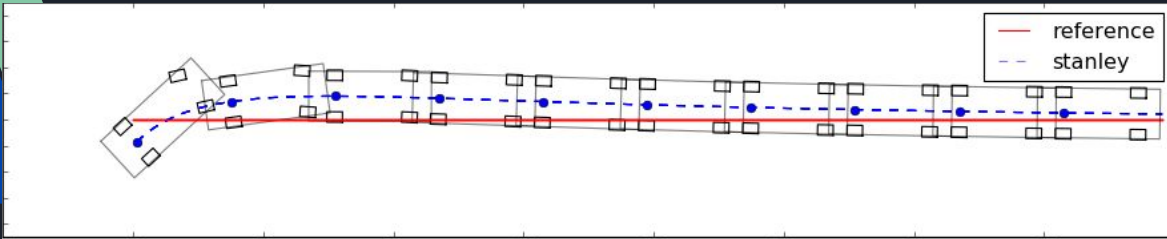


# stanley 프로그래밍과제

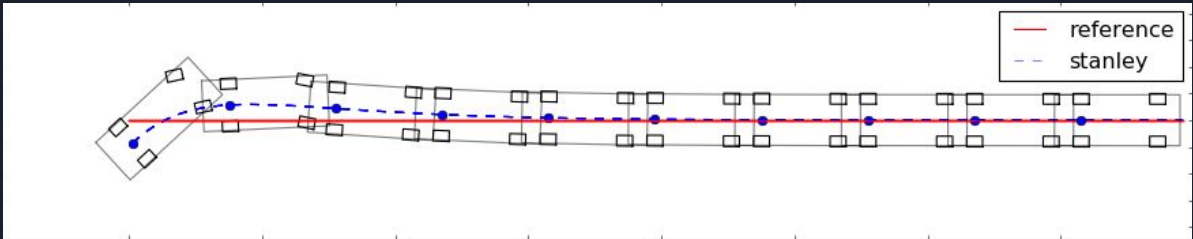
프로그래머스 자율주행 코스 1기  
조정민

# PID 제어

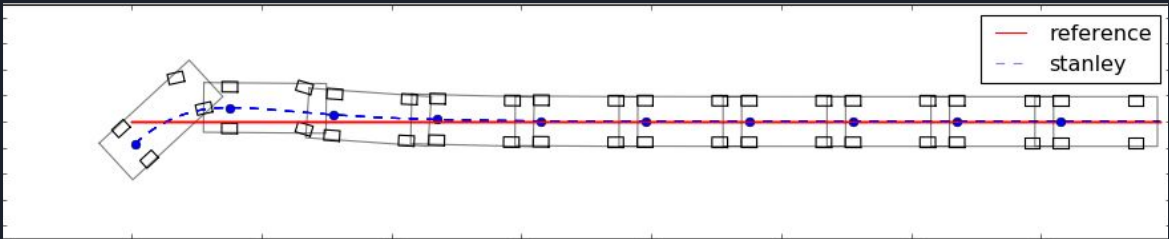
$k = 0.1$



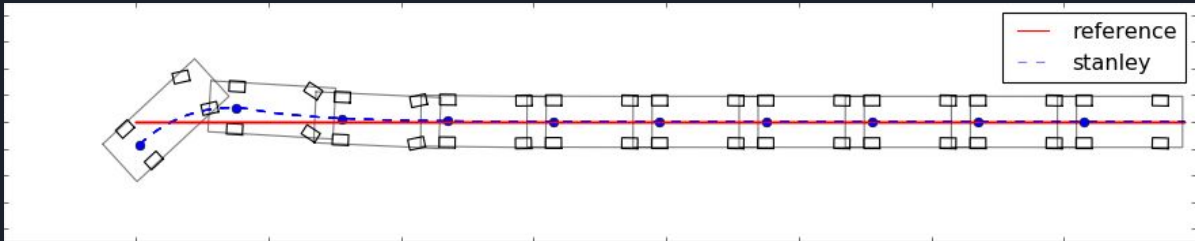
$k = 0.5$



$k = 1.0$



$k = 20.0$



CTE에 곱해지는  $k$  gain의 값을 조정하여 부드럽게 주행하게 할 수 있다.