제어시스템개론 기말 Report

융합전자공학부 2018006671 오우진

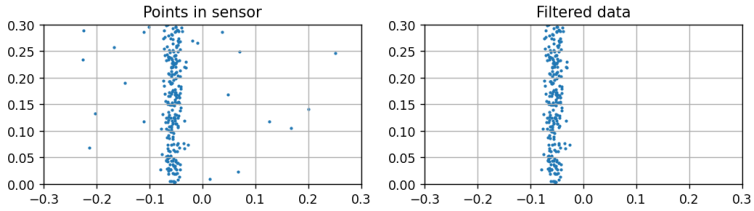
1. 서론

제공된 dynamics 파일을 기준으로, controller을 설계하여 시뮬레이션 상에서 라인을 따라 빠르게 출발점에서 목적지로 도달하도록 한다. 주어진 트랙과 임의의 트랙 모두에서 성공적으로 완주하고, 가능한 빠르게 완주하는 controller를 구현하도록 한다. T_DEL은 0.01로 고정이고, 로봇의 초기 위치는 트랙 위에서 트랙을 똑바로 바라보는 방향으로 설정한다. 로봇은 이전 중간 프로젝트와 마찬가지로, unicycle model로 가정한다.

2. 방법론

2.1 이상치(노이즈) 제거

주어진 상황에서는 트랙을 센서를 통해 관측하는 과정에서 노이즈가 발생한다고 가정하고 있다. 정확한 트랙의 경로를 알기 위해서는 이러한 노이즈를 제거해야 한다. 이를 위해, 좌표들의 x좌표를 기준으로 'Tukey Fences'를 적용하였다. 처음에는 z-scores 기법을 적용하였지만, 실험을 통해 'Tukey Fences' 기법이 다양한 상황에서 노이즈를 더 많이 제거하여 이 기법을 사용하였다.



<그림.1 노이즈 제거 전후 트랙 데이터>

2.2 각도(각속도) 제어

위에서 이상치를 제거한 데이터를 기반으로, 트랙 데이터를 가장 정확하게 나타내는 직선을 구한다. 이 직선의 기울기와 위치(y=0.15일때의 x좌표)를 기준으로 제어를 한다.

- 직선 구하기

주어진 데이터를 기반으로 직선을 구한다. 이를 위해 y<0.05인 점들의 평균 좌표, y>0.25인 점들의 평균 좌표를 구한다. 구한 두 점을 기준으로 1차 방정식을 구한다. 해당 범위에 점이 없는 경우도 발생하여, 그런 경우에는 주어진 데이터의 y좌표 중앙값을 구한 후, 이 점을 기준으로 상하로 나누어 점을 2개 구하여 직선을 계산한다.

처음에는 linear regression을 통해 1차 방정식을 구하였지만, 노이즈에 의해 실제 트랙과 다른 직선을 예측하는 경우가 발생하였다. 또한, 코드를 실행하는데 오래 걸려 더 가벼운 앞서 말한 방식을 적용하였다.

- 중간값 기준 제어(y=0.15인 x좌표)

$$error = 0 - x\_median$$

위에서 구한 직선에서, 중간값인 y=0.15일때의 x좌표 값(x_median)을 구한 후, x좌표가 x=0에서 얼마나 멀리 떨어져 있는지에 따라 각도를 제어한다. error를 위 식과 같이 정의하였다. P제어를 적용하여, error가 크다는 것은 트랙이 로봇의 중심으로부터 멀리 떨어져 있는 것을 의미하므로, error가 클 때 각도 제어를 더 많이 하게 된다.

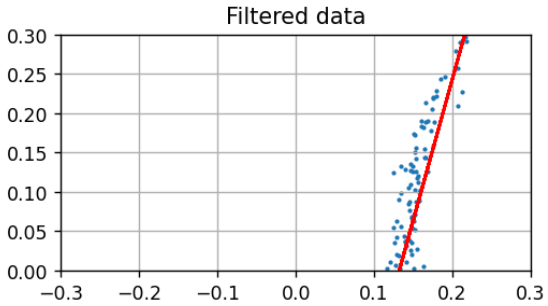
추가적으로, D제어를 적용하였다. D제어를 통해 error 값이 급격히 변화할 경우, 그 변화율에 비례하여 제어를 하여 error가 급격히 변화하는 것을 억제하였다. P-gain은 110,

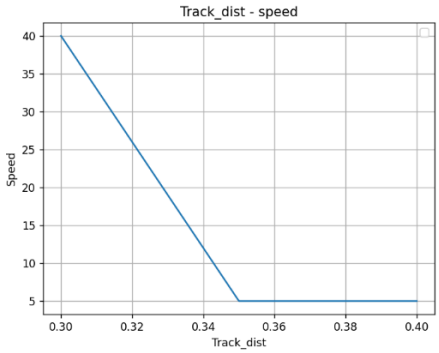
D-gain은 0.4를 사용하였다.

- 기울기 기반 제어

기존에 PD제어만 적용한 경우, 트랙의 실제 기울어진 각도는 고려하지 않고, 오직 트랙이 로봇의 중심으로부터 벗어난 거리만 고려하여 제어를 하였다. 이 경우, 트랙을 따라가기 위해서는 높은 P-gain을 설정해야 했고, 높은 P-gain에 의해 각도의 변화가 컸다. 트랙을 일정하게 따라가지 못하고 진동하는 문제가 발생하였다.

이를 보완하기 위해 트랙의 기울어진 각도도 고려하여 제어하였다. 우선, 위에서 구한 직선을 통해 기울기를 구한다. 구한 기울기를 통해, 트랙과 로봇의 중앙(x=0)이 평행하기 위해 필요한 각도만큼 제어를 해준다. 이를 통해, 로봇이 트랙으로부터 멀어지는 상황에서 기존 P제어만 적용한 상황보다 낮은 P-gain을 이용하면서도 각도 제어를 크게 할 수 있다. 만약 트랙으로부터 가까워지는 상황의 경우, 기존보다 각도 제어를 작게 하여, 로봇이 트랙에 가까워지는 상황에서는 현재 각도를 비슷하게 유지할 수 있다. T_del = 0.01이므로, 기울기의 gain은 100을 사용하여 트랙과 로봇의 중심 사이의 각도 크기만큼 반영되도록 하였다.





<그림.2-1 노이즈 제거된 데이터를 기반으로 트랙을 1차 함수로 예측, 그림.2-2 트랙 길이-속력 그래프 >

2.3 속력 제어

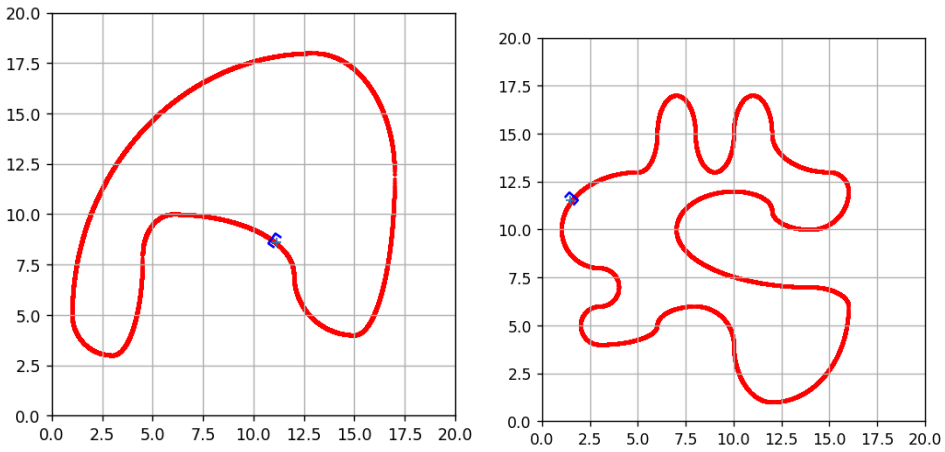
속력은 현재 보이는 트랙의 길이와 반비례하게 제어하였다. 트랙과 로봇의 중심이 평행일 경우, 트랙의 길이는 0.3이다. 만약 트랙이 기울었다면, 그 길이는 더 커진다. 이를 기반으로, 현재 보이는 트랙의 길이가 0.3에 가까울수록 속력을 크게 하고, 길이가 클수록 속력이 낮아지도록 제어하였다. 실험을 통해 트랙의 길이가 0.34를 넘는 경우가 거의 없음을 관측하였다. 이를 기반으로, 길이가 0.35 이상일 경우 매우 낮은 속력인 5에 수렴하도록 하였다. 트랙의 길이와 속도 사이의 관계를 그림.2-2을 통해 나타내었다. 트랙의 길이는 0.3/sin(theta) (theta는 x축과 직선 사이 각도)를 통해 구하였다.

트랙을 더 빠르게 완주하기 위해, 트랙이 로봇의 중심과 가까울 경우, 속력이 큰 값을 가지도록 제어하였다. 직선의 기울기, 로봇의 중심과 직선 사이의 거리를 고려하여 로봇의 중심과 충분히 가깝고, 기울기가 로봇의 중심과 평행하다고 판단되면 속력이 50이 되도록 하였다. 추가적으로, 만약 위와 같은 상황에서, 이전 속력이 50 이상일 경우, 속력을 2만큼 증가시켰다. 이를 통해 트랙의 직선 부분에서 빠르게 주행하도록 하였다.

2.4 예외사항 처리

현재 사용중인 트랙은 점들의 집합으로 되어있다. 간혹, 점들이 적은 구간이 존재하는데, 이런 경우 위의 방식대로 이상치를 제거하거나 직선을 예측하면, 기존 트랙과 다른 결과가 나온다. 이를 방지하기 위해, 센서로 읽어온 데이터의 개수가 10개 미만이면, 속도는 35로 고정하고, x좌표들의 중앙값을 기준으로 P제어를 통해 각도를 조율한다.

3. 결과 분석



<그림.3 제공된 트랙, 제작한 트랙>

3.1 성능 평가

주어진 트랙에 대해 출발점으로 돌아오는데 걸리는 for 문 반복 횟수를 기준으로 성능 평가를 진행하였다. 기존 트랙을 완주하는데 총 139 번 for 문을 반복하였다. 최고 속도를 조정하여 더 빠르게 완주할 수 있으나, 임의의 트랙에서 안정적으로 작동한다는 것을 보장하지 못해 속도를 더 이상 늘릴 수 없었다. 완주하는 동안 총 49 번은 속력이 50 이상인 상태로 주행하였다. 이를 적용하지 않으면 완주하는데 153 번 소요되었다.

기존 트랙보다 곡선이 심한 트랙에서도 성능이 안정적으로 나오는지 확인하기 위해 그림.3 과 같은 트랙을 제작하였다. 기존보다 직선 구간이 적고, 곡선의 휘는 정도도 기존보다 심하다. 실험 결과, 제작한 트랙에서도 성공적으로 완주하였다. 이를 통해 곡선이 심한 트랙에서도 어느정도 안정적으로 작동 가능한 것을 확인했다. 완주하는데 총 222 번 for 문을 반복하였다.

완주 이후에도 트랙을 잘 따라가는지 확인하기 위해 for 문을 10000 번 반복하여 트랙을 벗어나는지 확인하였다. 그 결과, 두 트랙 모두에서 트랙을 벗어나는 일은 관측되지 않았다.

3.2 노이즈가 없는 경우

로봇이 트랙을 센서로 관측하는 과정에서 노이즈가 발생하지 않는다고 가정하고 실험을 진행하였다. 그 결과, 기존 제공된 트랙을 완주하는데 134 번 소요되었고, 제작한 트랙은 221 번 소요되었다. 기존 노이즈가 있는 경우와 큰 차이가 발생하지 않았다. 이를 통해 제시된 controller 가 노이즈에도 강인하다는 것을 확인하였다.

3.3 dynamics 변경

현재 사용중인 dynamics 의 경우, controller 에서 받아온 속력과 각속도를 기준으로 제어하고 있다. 이때, 속력에 따라 이동한 후, 각도를 조정하고 있다. 이 순서를 조정하여 새로운 dynamics 를 설계하고 이에 대한 성능을 평가하였다. 새 dynamics 의 경우, 각도 조율을 한 이후 이동하도록 하였다. 이 방법으로 기존 트랙을 133 번 만에 완주하였다.

4. 결론

설계한 controller 가 노이즈에도 강인하고, 곡선이 심한 트랙에서도 완주가 가능함을 확인하였다. 기존 PD 제어에 트랙과 로봇의 중심 사이의 각도를 통한 제어를 추가하여, 기존보다 트랙을 잘 따라가도록 하였다.