

로봇네비게이션 중간 프로젝트

조교 전형준

프로젝트 소개

International Conference on Intelligent Robots and Systems(IROS)는 최근 9th F1TENTH Autonomous Grand Prix라는 대회를 개최하였습니다. 해당 대회는 그림 1 과 같은 파이썬 시뮬레이션 환경에서 자율주행 레이싱을 진행합니다. 대회 우승을 위해서 레이싱카는 2D 라이다를 사용하여 트랙 2바퀴를 빠르게 완주해야 합니다. ([대회 사이트](#))

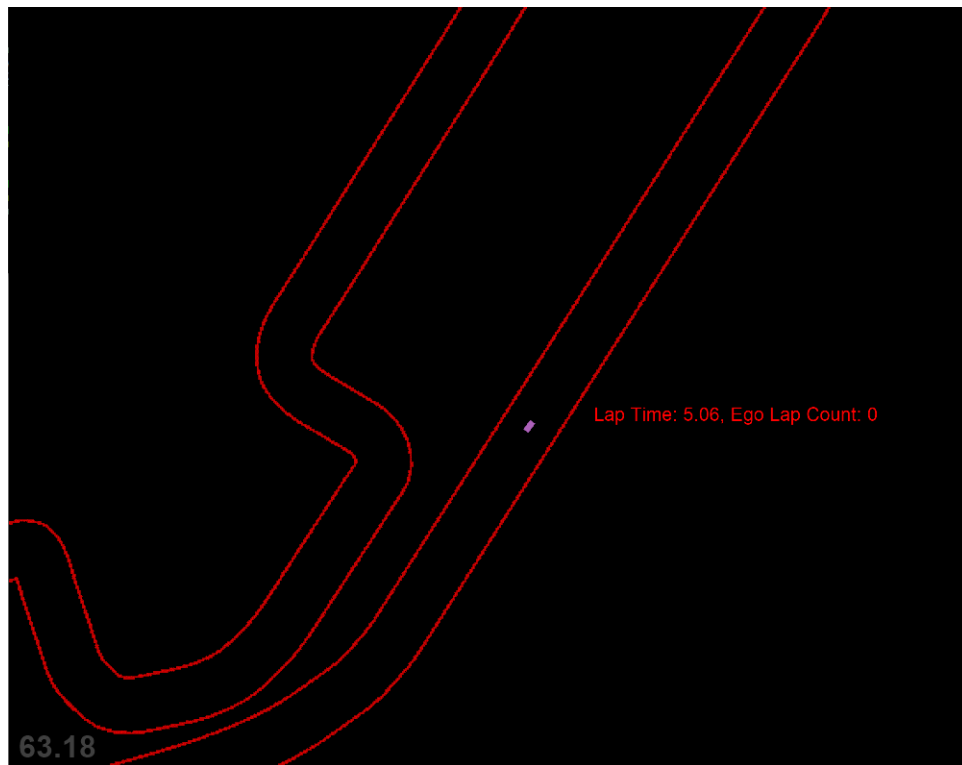


그림 1 F1tenth의 레이싱 시뮬레이션, 빨간색 선과 보라색 네모는 각각 레이싱 트랙과 레이싱카를 나타낸다.

이번 중간고사에서는 위의 레이스 경주를 하겠습니다. 실제 대회에서는 2개의 차량이 동시에 달려서 경주하고, 장애물이 있는 환경을 주행해야 합니다. 하지만 이번 대회에서는 장애물이 없는 환경에서 한 대의 레이싱카의 lap time으로 평가를 내리겠습니다.

이번 대회에서 좋은 성적을 거두어서, 내년에도 열리는 F1TENTH 대회에 도전해봅시다!

대회규칙

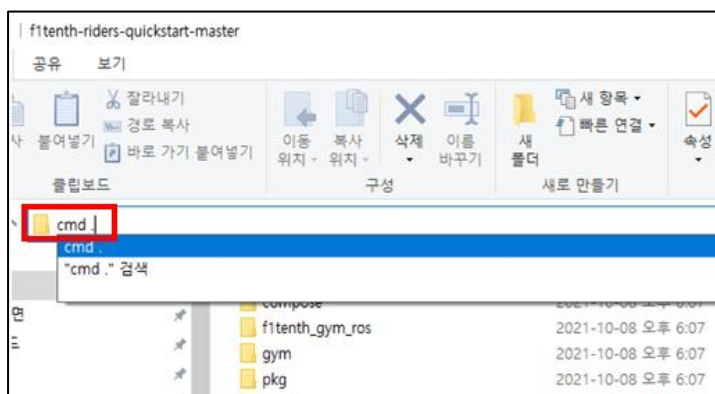
1. 대회는 개인전이며, 유사도가 높은 코드가 있을 시 감점이다.
2. 맵은 "SOCHI"와 "SILVERSTONE"을 사용한다.
3. 각각의 맵에 대해서, 트랙 2바퀴의 완주시간을 종합하여 순위를 결정한다.
4. 각 완주시간은 현실 시간 대신에 시뮬레이션의 lap time을 기준으로 평가한다.
5. 중간에 부딪히더라도 부딪힐 때의 이동거리와 lap time으로 순위를 나눈다.
6. 비정상적인 방식으로 주행 한 경우(역주행)는 점수 및 순위에 등록되지 않는다.
7. 라이더 데이터 이외에 정보를 이용하는 경우는 인정되지 않는다.
8. 맵을 직접적으로 이용하는 localization 등의 기법을 쓰는 것은 허용하지 않는다.
9. 제출 마감일은 10월 31일 23시 59분이다.

설치 튜토리얼

주의) 파이썬 및 pip 패키지가 깔려 있다는 가정으로 설명하겠습니다.

(윈도우 환경)

1. 다운로드 받은 압축파일을 풀고, f1tenth-riders-quickstart-master 폴더를 연다.
2. 아래의 그림과 같이 해당 경로에서 파일탐색기에 `cmd` . 을 입력하여 현재 경로에서, 명령프롬프트 창을 연다.



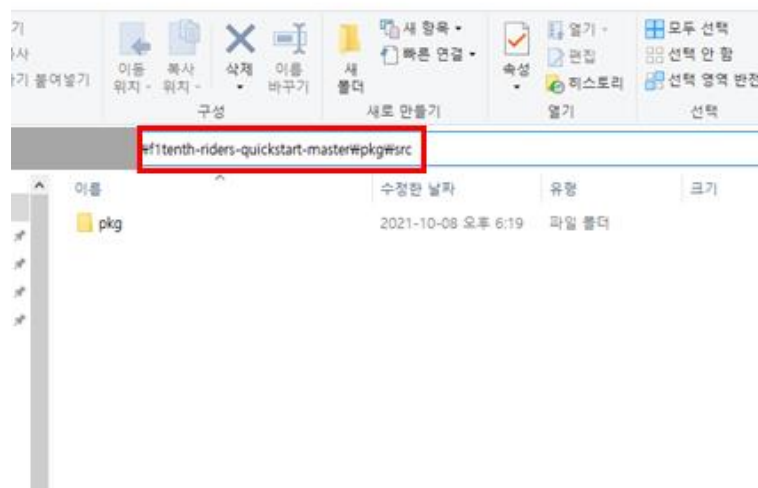
3. 프롬프트 창에 `pip install --user -e gym`를 입력하여 패키지를 설치한다.

```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19042.1237]
(c) Microsoft Corporation. All rights reserved.

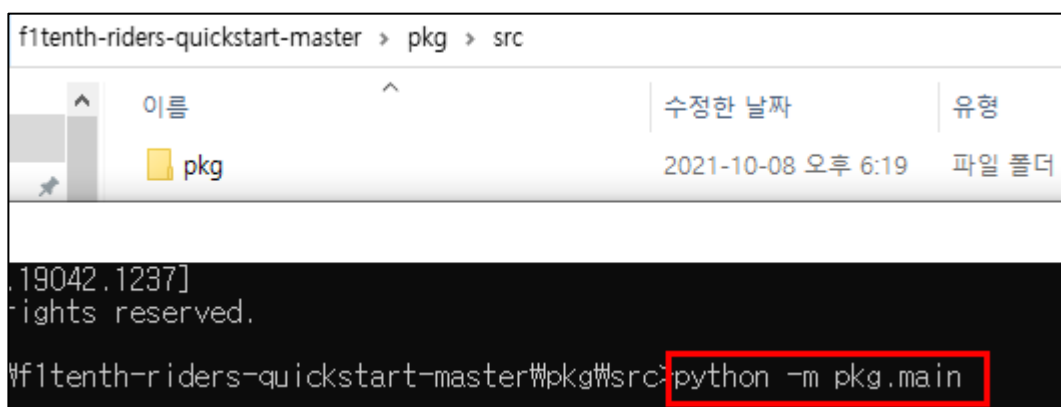
f1tenth-riders-quickstart-master>pip install --user -e gym
```

기본 실행 튜토리얼

1. 이번에는 현재경로(f1tenth-riders-quickstart-master)에서 'pkg' 폴더아래에 'src'폴더로 이동한다.



2. 다시한번 파일탐색기에 cmd . 을 입력하여 명령프롬프트 창을 켜서 python -m pkg.main 명령을 입력한다.



3. 기본 주행알고리즘으로 레이싱이 시작된다. 중지하고 싶다면 ESC버튼을 누르면 된다. (ESC종료 후에, 명령프롬프트 창에서 Ctrl+C 단축키로 완전히 코드를 종료해라. 종료가 안되면 명령프롬프트를 끌 것) 두바퀴를 다 돌거나 충돌하면 코드가 멈춘다.



한바퀴를 돌때마다, Ego Lap Count값이 올라간다.

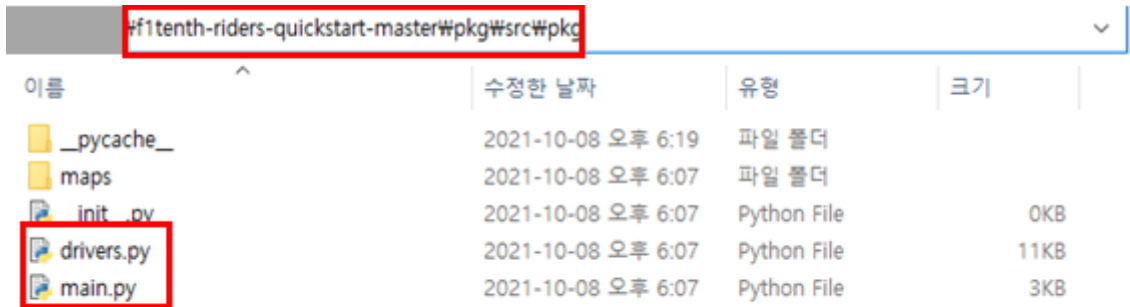


4. 2바퀴를 완주한다면 아래 그림의 빨간선으로 표시한 lap time이 나오면서 시뮬레이션이 종료된다. ※ Real elapsed time은 실제로 주행하는데 걸린 시간이지만, 이번 대회에서는 고려하지 않는다.

```
f1tenth-riders-quickstart-master\pkg\src>python -m pkg.main
Sim elapsed time: 179.49999999999998 Real elapsed time: 277.8576090335846
f1tenth-riders-quickstart-master\pkg\src>
```

코드 설명

1. 'f1tenth-riders-quickstart-master' 폴더에서 'pkg' - 'src' - 'pkg'폴더 아래로 이동한다. 그러면 아래의 사진과 같이 drivers.py와 main.py가 있다.



이름	수정한 날짜	유형	크기
__pycache__	2021-10-08 오후 6:19	파일 폴더	
maps	2021-10-08 오후 6:07	파일 폴더	
init.py	2021-10-08 오후 6:07	Python File	0KB
drivers.py	2021-10-08 오후 6:07	Python File	11KB
main.py	2021-10-08 오후 6:07	Python File	3KB

drivers.py는 주행알고리즘들이 담겨있는 파일이고, main.py는 주행알고리즘을 시뮬레이션에서 테스트하기 위한 파일이다.

2. drivers.py 파일안에는 아래와 같은 기본적으로 **process_observation**이라는 함수를 가진 Driver 클래스들이 있다. 아래의 코드는 그 중에서 **SimpleDriver**라는 클래스이다.

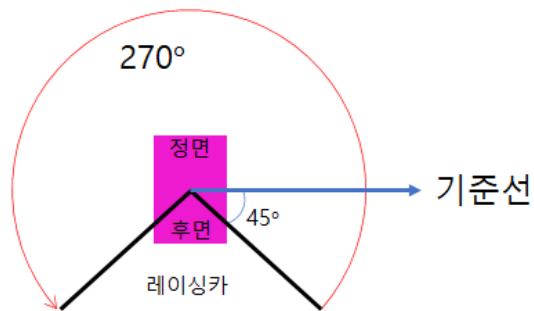
```
class SimpleDriver:
```

```
    def process_observation(self, ranges=None):
        speed = 5.0
        steering_angle = 0.0
        return speed, steering_angle
```

클래스 내에 **process_observation**함수는 2D 라이다의 센서값(**ranges**)을 받는다. 함수의 반환값은 차량의 목표 속도와 스티어링 각이다. (차량은 ackerman steering의 구조를 가진다.)

※ **ranges** 값 정의

- 2D라이다의 센서 값은 방위 별 거리값의 배열이며, 리스트 자료형이다.
- 거리값은 우측 방향의 기준선에 대해서 - 45도부터 225도까지 순서대로 리스트에 저장되어 있고, 0(m)에서 35(m)사이이다.
- 라이다 분해능은 0.25도 이며, 리스트의 크기는 1080개이다.



※ **steering_angle** 정의

`steering_angle` : an angle in the range $[-\pi/2, \pi/2]$, i.e. $[-90^\circ, 90^\circ]$ in radians, with 0° meaning straight ahead.

- ※ **SimpleDriver**이외에도 **GapFollower**, **AnotherDriver**, **DisparityExtender**라는 주행알고리즘을 가진 클래스가 있으니 참고바람.

3. main.py를 열어보면, 13번째 줄에 아래와 같은 라인이 있다.

```
-----  
from pkg.drivers import DisparityExtender as race_driver  
-----
```

여기서 **DisparityExtender**는 drivers.py에 정의된 클래스이다. 만약에 여기서 밑줄 친 부분을 다른 Driver 클래스 이름으로 바꾸면, 해당 클래스의 **process_observation** 함수에 의해서 레이싱 카가 주행한다.

또한 19번째 라인을 보면 아래와 같이 레이스트랙(맵)이 'SOCHI'로 되어있다.

RACETRACK= 'SOCHI'

이 레이스 트랙은 'SILVERSTONE'으로 바꿀 수 있다. 다양한 맵에 대해서 잘 작동하는 알고리즘을 구현하자.

나만의 custom driver만들기

1. drivers.py 아래에 위에서 설명한 process_observation라는 함수를 가진 **CustomDriver** 클래스를 만든다.

※여기서 process_observation함수의 구현은 임의로 하면 된다. 아래는 예시를 위해서, SimpleDriver코드를 넣었다.

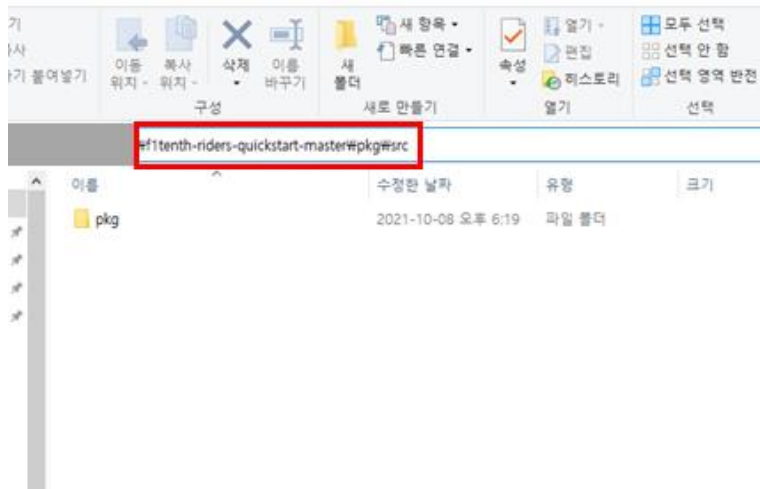
class CustomDriver:

```
def process_observation(self, ranges=None):  
    speed = 5.0  
    steering_angle = 0.0  
    return speed, steering_angle
```

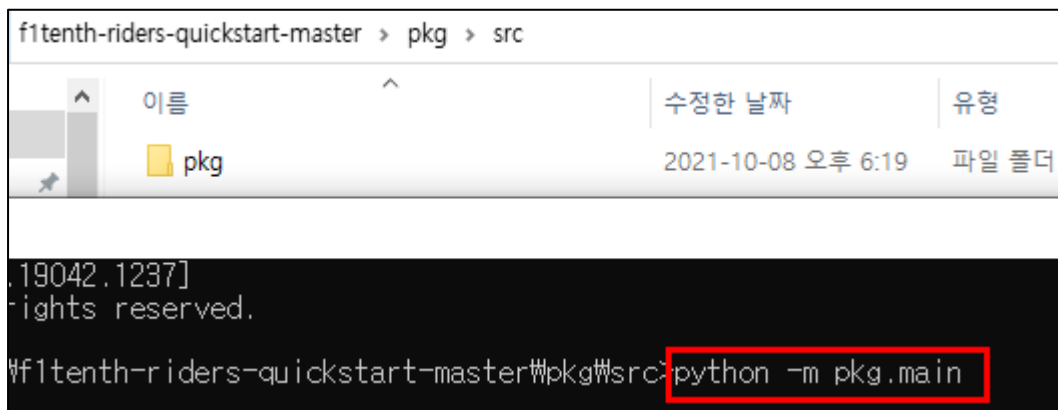
2. main.py를 열어서, 13번째 줄에 라인을 다음과 같이 바꾼다.
-

```
from pkg.drivers import CustomDriver as race_driver
```

3. f1tenth-riders-quickstart-master에서 'pkg' 폴더아래에 'src'폴더로 이동한다.



4. 파일탐색기에 cmd . 을 입력하여 명령프롬프트 창을 켜서 python -m pkg.main 명령을 입력한다.



5. 결과를 확인한다.

제출간 주의사항

1. 문의 및 코드 제출은 아래의 이메일로 한다.

jeonkw@kw.ac.kr

2. 기본적으로 제출해야 하는 파일은 drivers.py와 requirements.txt의 압축파일이다. drivers.py는 알고리즘 코드가 담겨있어야 하며, requirements.txt에는 추가 설치해줘야 하는 패키지 및 기타 사항을 아래와 같이 적어 둔다.

Ex) Pillow==8.3.2

3. 추가적으로 제출할 파일이나 환경설정이 있다면, 그에 따른 사용방법도 같이 보낼 것. (pdf형태)
4. drivers.py파일에 제출하는 주행알고리즘의 클래스 이름은 **CustomDriver**로 할 것
5. GPU를 사용하는 방식도 가능함.