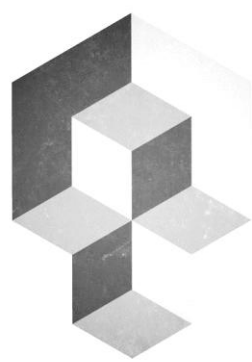
Deep Racing 개요



*** 시뮬레이션(ex. Angle Setting in Action Space) 실물 차량이 필요한 경우는 제외.**

Course2 - Deep Racer Component

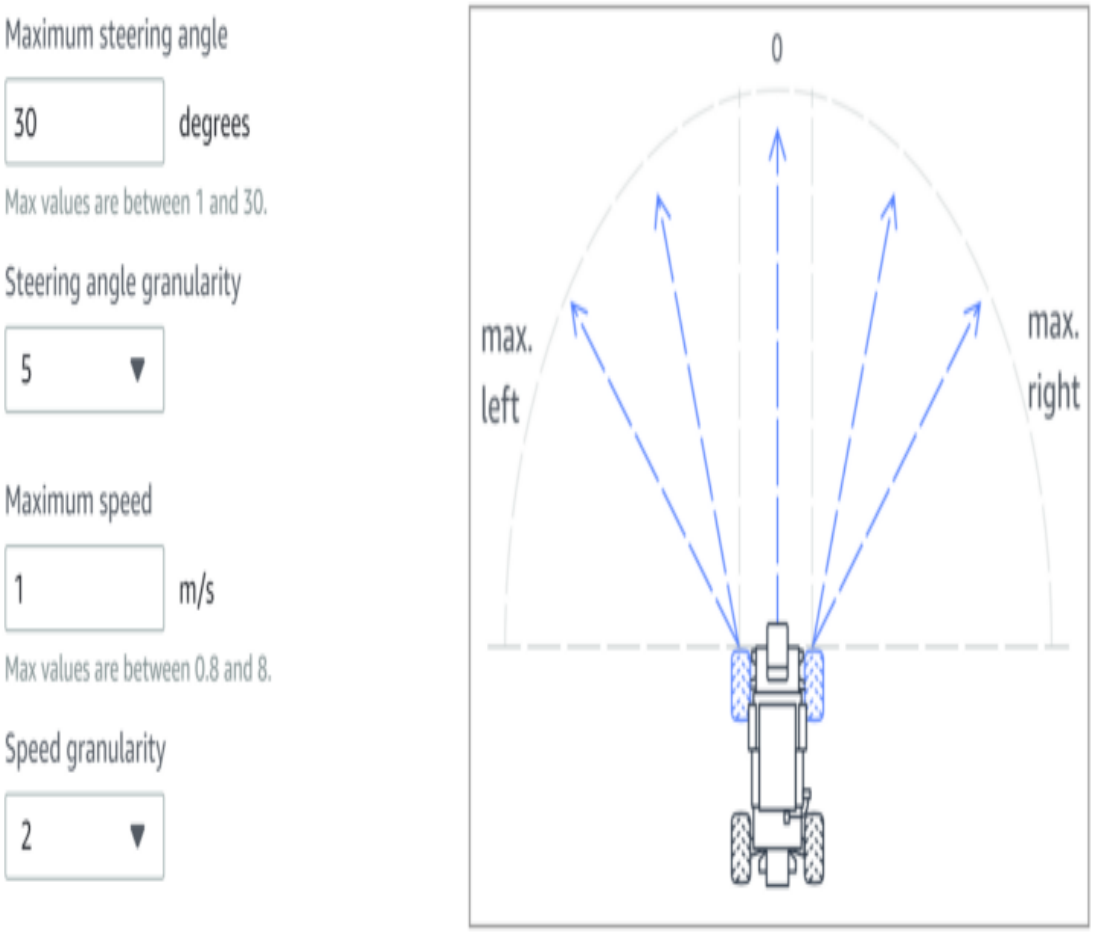
1. 4MP Camera
2. Car Chassis
3. Battery
4. Power Bank
5. Battery Charger
6. Full Size HDMI PORT (Side)
7. Micro USB PORT(Side)
8. Micro USB-C PORT(Side)

* Lidar and Camera (New Model)
*15 pictures in 1second → STEP

Course2 - Deep Racer Component

1. 32gigs of storage(= 32GB)
2. 4GB RAM
3. LED

Course2 – Setting in Action space



1. Maximum_steering_angle
→ 0~30 degree

2. Steering_granularity

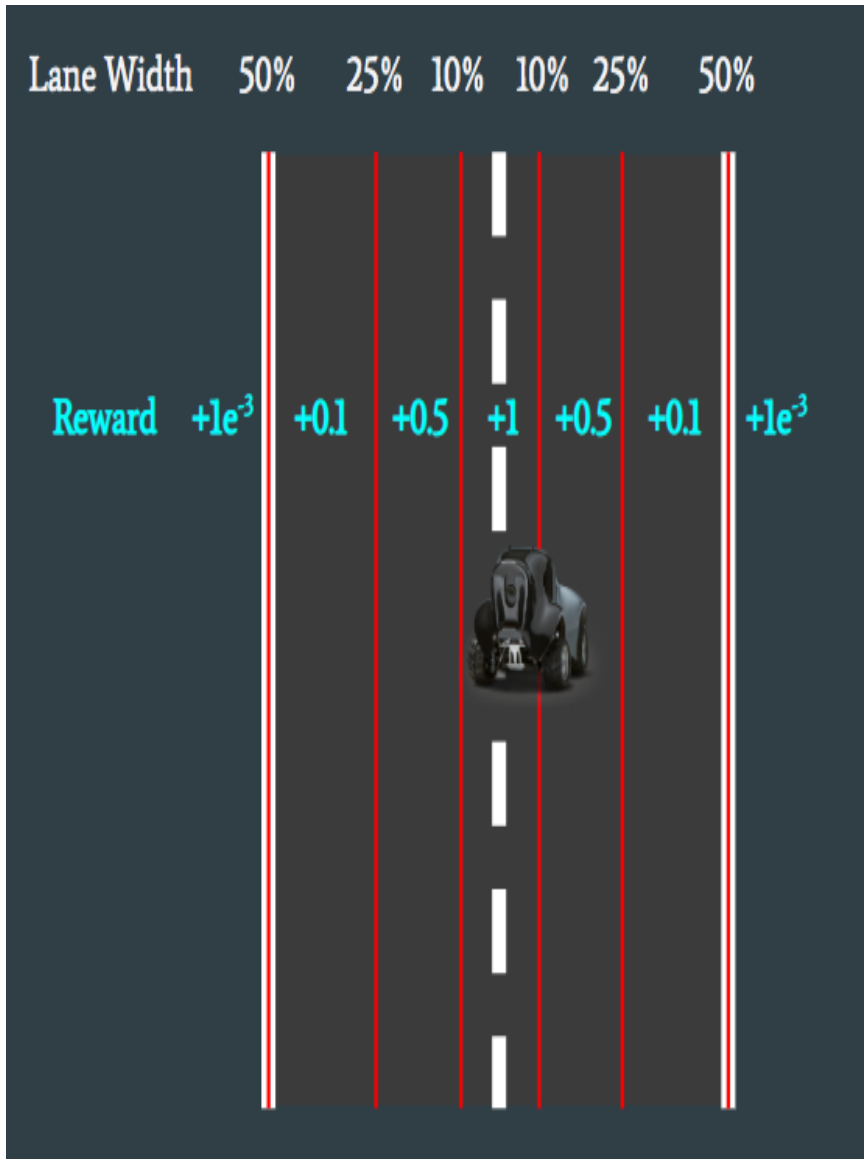
3. Maximum_speed
→ 0.8m/sec ~ 8.0m/sec

4. Speed_granularity
→ 1,2,3 (select)

* **Action Number** = steering_granularity * speed_granularity

Deep Racer Assembly, Steering Calibration, Throttle Calibration
https://classroom.udacity.com/courses/ud014/lessons/336b89e3-f4e8-4ec5-951c-dd67da63a089/concepts/50f3e7f8-778c-47d6-b424-8d3a4abd6074

Course3 – Reward Function – distance from Center




```python
def reward_function(params):
    '''
    Example of rewarding the agent to follow center line
    '''

    # Read input parameters
    track_width = params['track_width']
    distance_from_center = params['distance_from_center']

    # Calculate 3 markers that are at varying distances away from the center line
    marker_1 = 0.1 * track_width
    marker_2 = 0.25 * track_width
    marker_3 = 0.5 * track_width

    # Give higher reward if the car is closer to center line and vice versa
    if distance_from_center <= marker_1:
        reward = 1.0
    elif distance_from_center <= marker_2:
        reward = 0.5
    elif distance_from_center <= marker_3:
        reward = 0.1
    else:
        reward = 1e-3  # likely crashed/ close to off track

    return float(reward
```

Course3 – Reward Function - Example

Track Width = 100m
Distance_from_Center = 23m

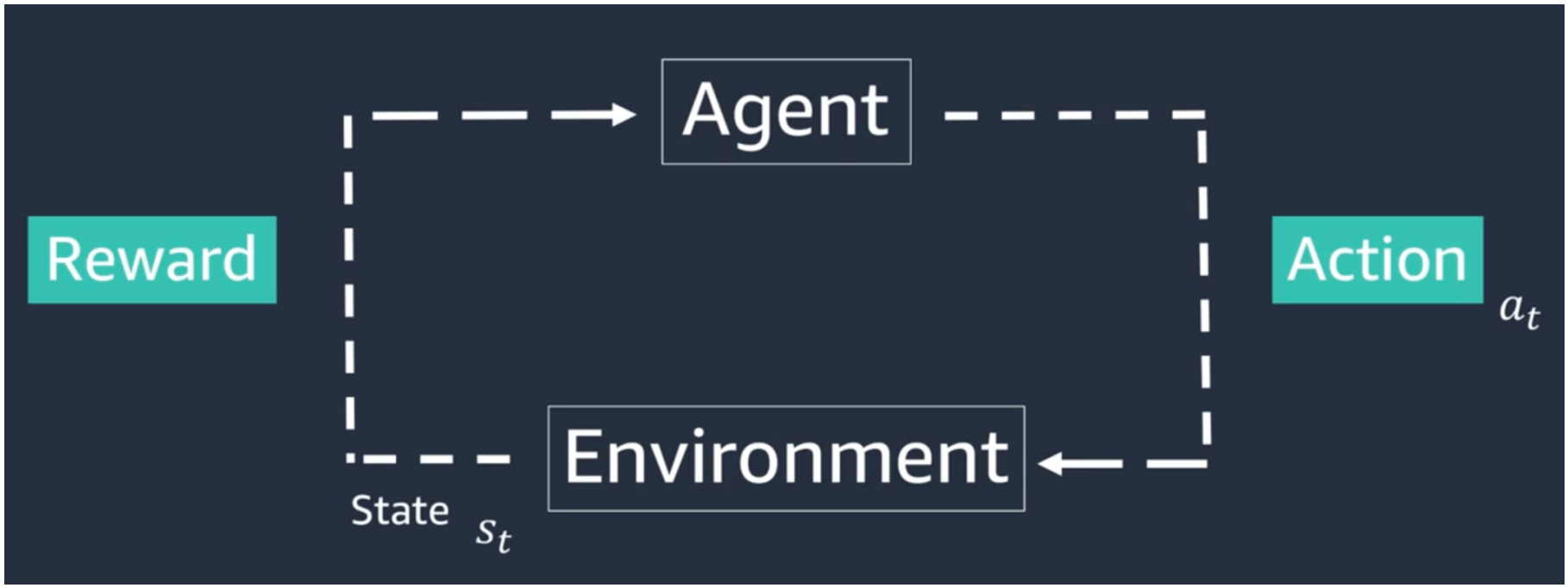
```
marker_1 = 0.1 * track_width
marker_2 = 0.25 * track_width
marker_3 = 0.5 * track_width
```

marker_1 = 10m
marker_2 = 25m
marker_3 = 50m

```
elif distance_from_center <= marker_2:
    reward = 0.5
```

* In Track Reward Function, Not Zig-Zag Reward Function

Course4 - Reinforcement learning



1. Agent
2. Action
3. Environment
4. State
5. Reward

1. Deep Racer
2 . Driving behavior(Turn, Accelerate)
3. Track
4. Point on the track at a given time
5. how far from the center of the track
   (In basic Reward Function)

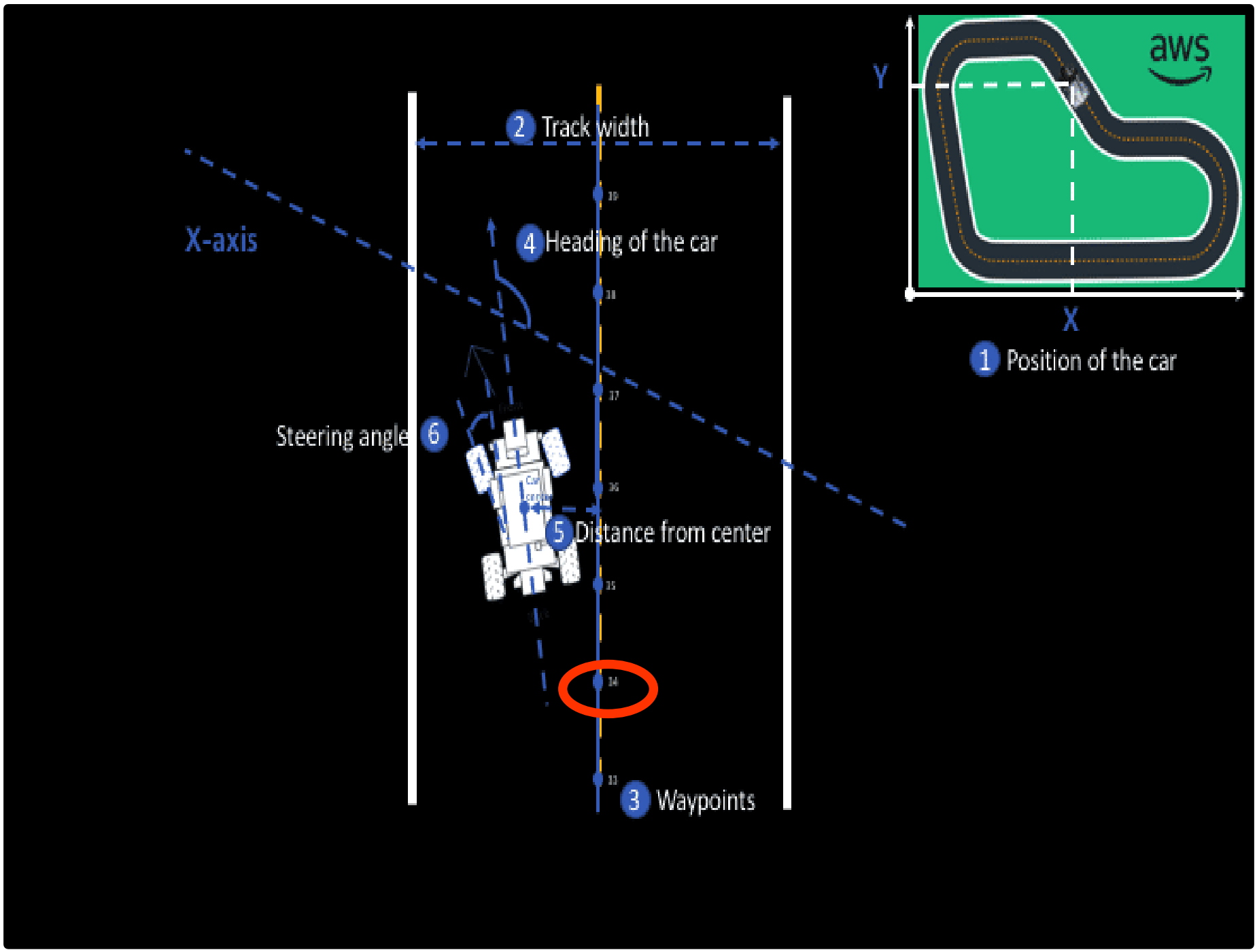
Course4 - Reinforcement learning

* Step(tuple) → state, action, reward, new state
* Episodes → 2types(Finish or not) , Set of steps
* Experience buffer → Set of steps in several episodes.
* Policy
* Value Function

Course5 – parameter

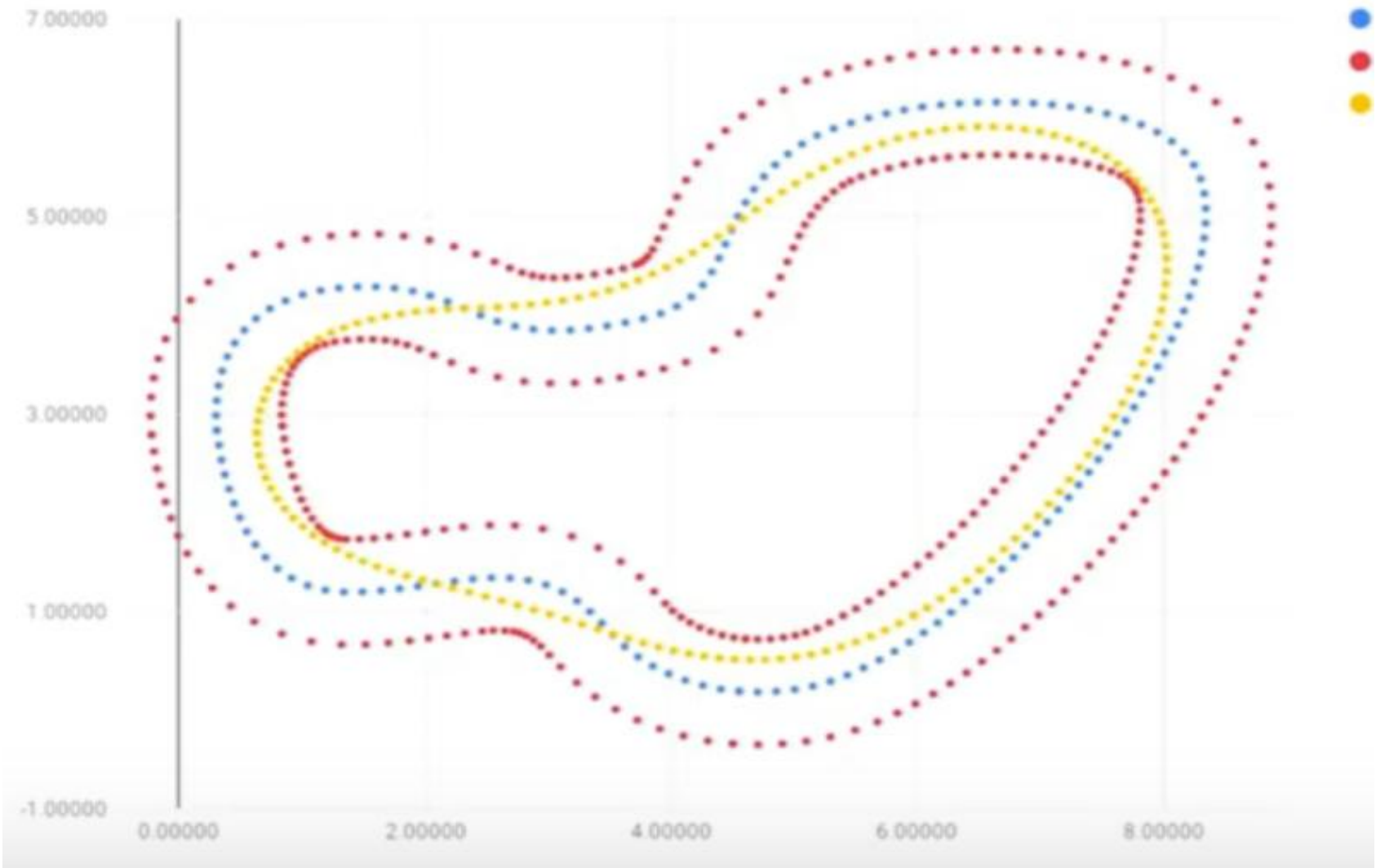
1. all_wheels_on_track (Bool)
2. x,y (float)
3. distacne_from_center(float)
4. is_left_of_center(Bool)
5. is_reversed(Bool)
6. Heading (float) → -180 ~ 180 degree
7. progress(float) → 0 ~ 100
8. Steps (intger)
9. Speed(float)
10. Steering angle(float) → + : going left , - : going right
11. track_width(float) → unit in meter
12. **waypoints(List)**
13. **closet waypoints(int)**

Course5 – parameter



https://classroom.udacity.com/courses/ud014/lessons/0afdbee8-cf7a-4966-b8ed-f860f3d68cb6/concepts/4208eda1-1094-4e66-94d2-9450a7afc6f9

Course5 – Waypoint



Course5 – Hyperparameter

* Batch Size
* Number of epochs
* LEARNING RATE
* EXPLORATION
* ENTROPY
* DISCOUNT FACTOR
* LOSS TYPE
* EPISODES

Course5 – Hyperparameter

Hyperparameter	Value
Gradient descent batch size	64
Entropy	0.01
Discount factor	0.888
Loss type	Huber
Learning rate	0.0003
Number of experience episodes between each policy-updating iteration	20
Number of epochs	10

* Course6
 ROS(Deep Racing을 위해 필수적으로 공부해야할 부분은 아닐 것으로 보임),
DeepRacer를 직접 Track에서 돌리는 방법을 설명.(실물이 있어야 한다고 판단)

Conclusion

,

1. 시뮬레이터 사용 여부 판단 필요.
2. 반복적인 시뮬레이션이 요구됨.
3. 어떤 식으로 Reward를 주는 것이 좋을 지 생각.
4. Waypoint의 활용방안 찾기
5. HyperParameter는 초기의 설정을 유지하는 것이 좋음.
   만약, 미세한 조정이 필요하다고 느껴질 시 , 그 때 수정.