
ONECLICK

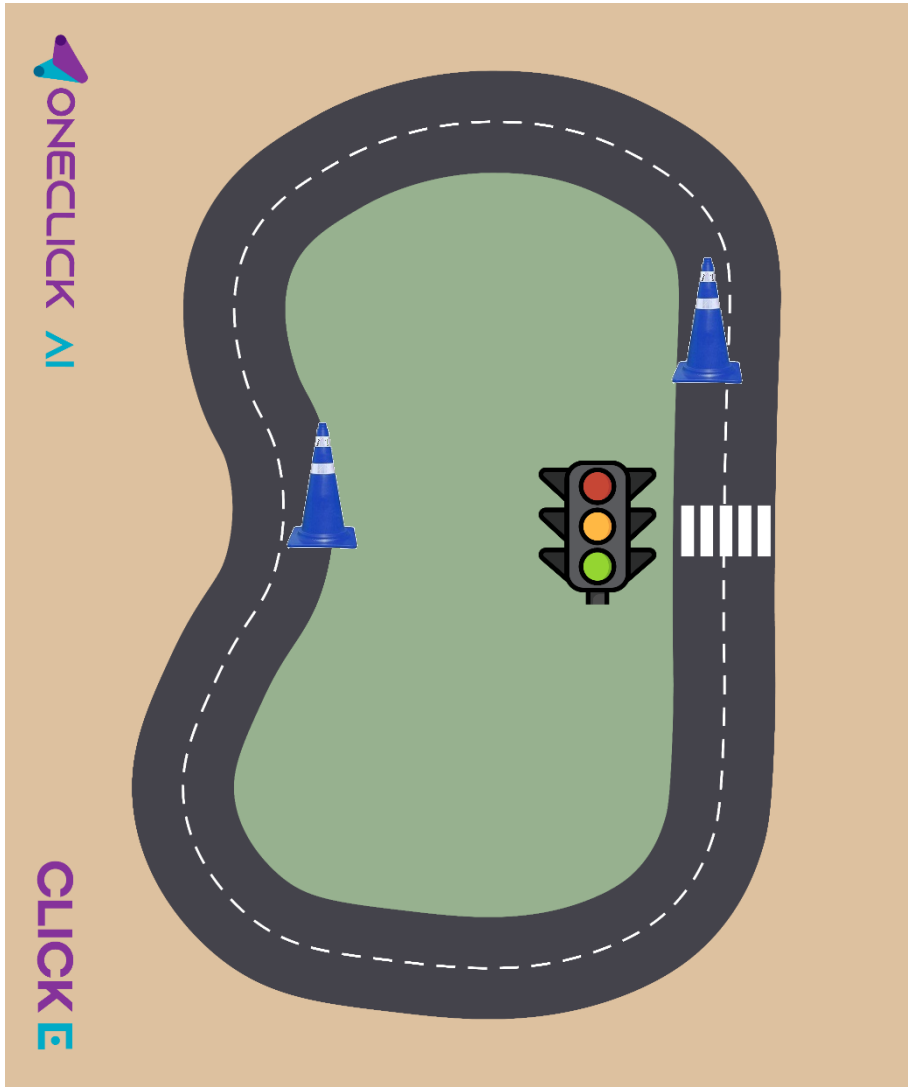
1차 평가 기준

1차 평가 기준

- 3바퀴 주행 (시계방향)
- 2번 기회 중 높은 점수
- 처음 두 바퀴는 신호등 초록색인 상태에서 주행
- 마지막 한 바퀴는 신호등 빨간색으로 변경(멈춰야함)

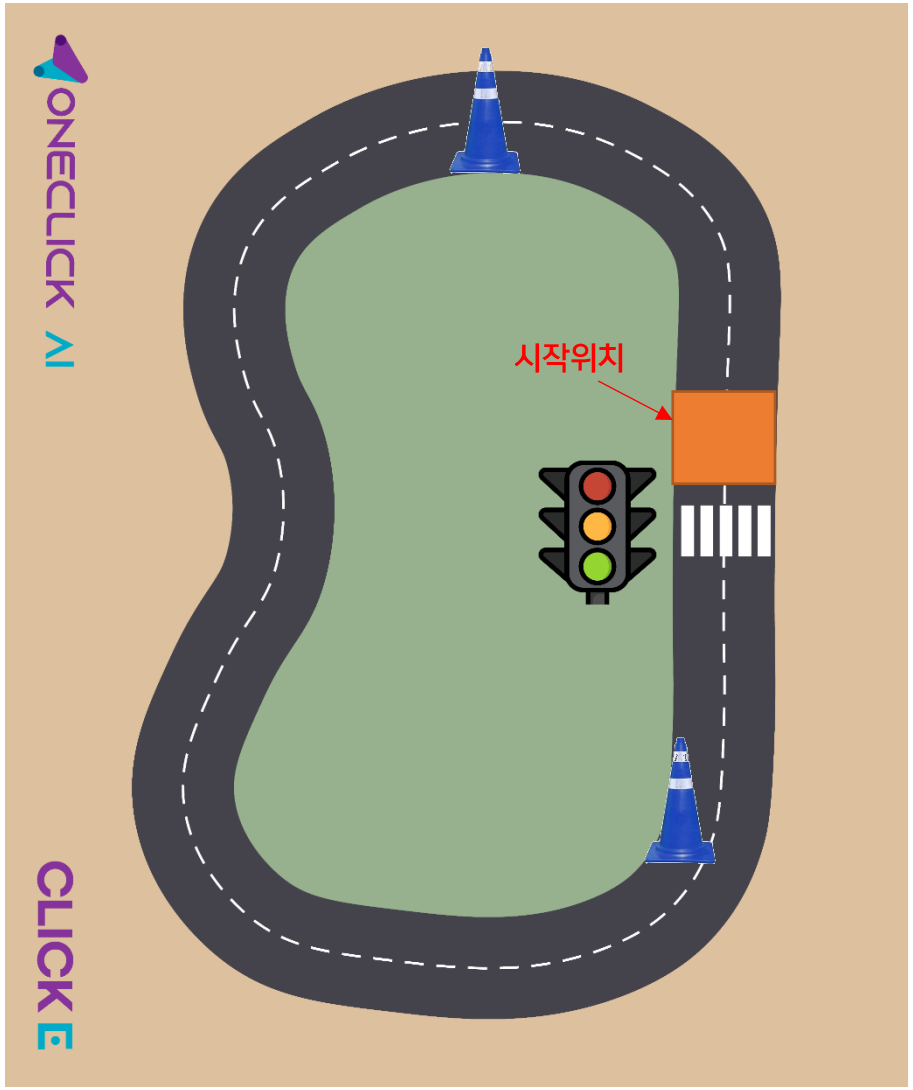
- 네 바퀴 모두 빠져나가면 감점 (-50)
- 초록불에서 멈추면 감점 (-10)
- 빨간불에서 직진하면 감점 (-20)
- 빨간불에서 횡단보도 밟을 시 감점 (-5)
- 장애물 충돌시 감점 (-10)

- 점수에 따라 토너먼트 조 배치에 유리하도록 진행

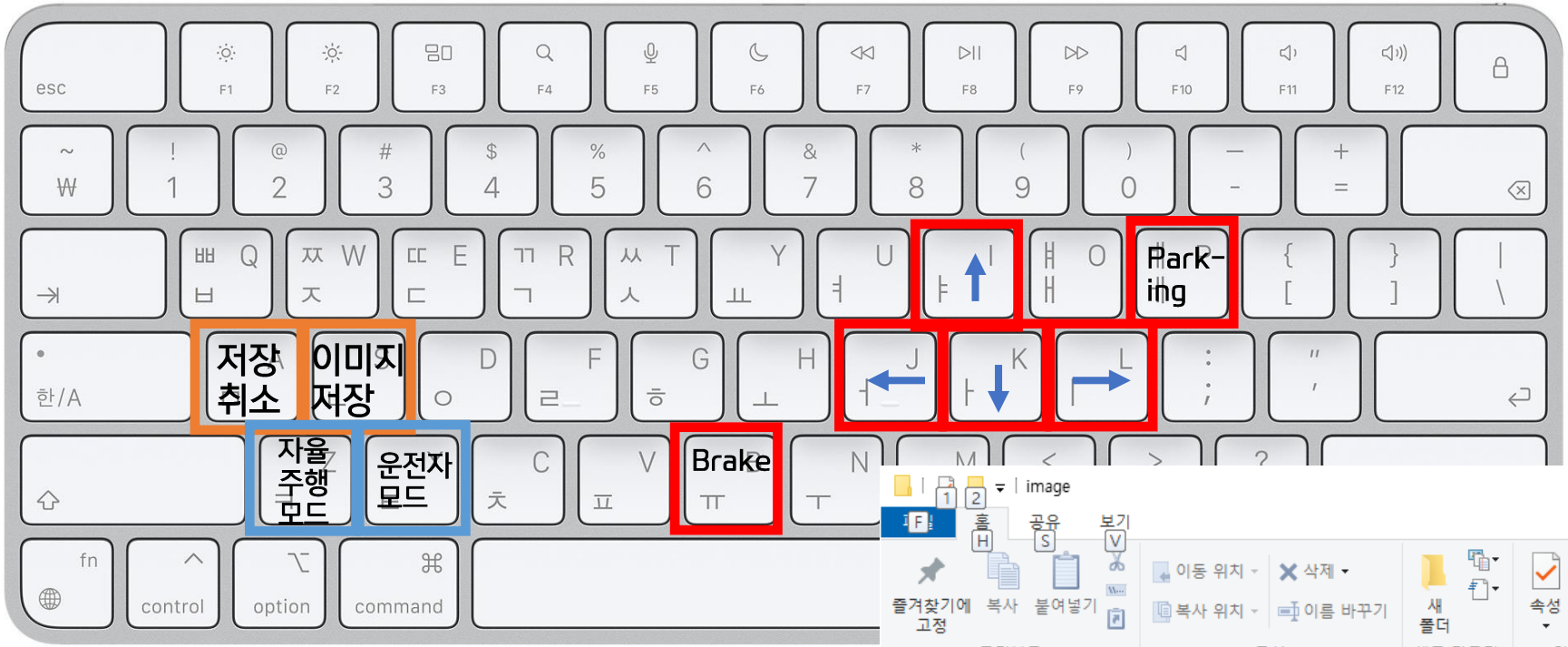


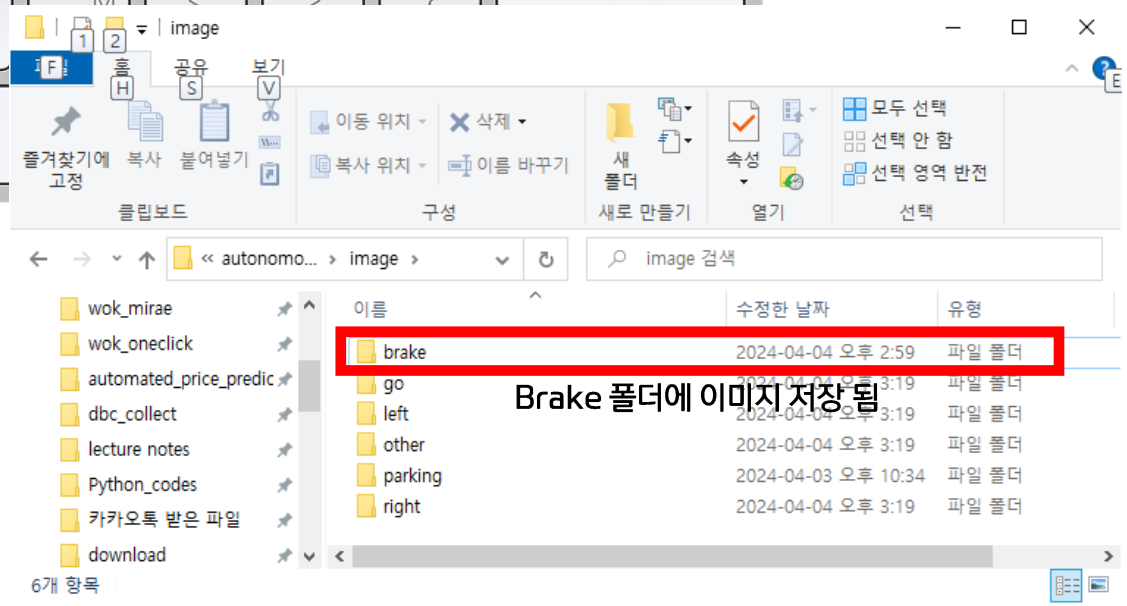
1차 평가 기준

- 준비:
  - 시작위치에서 자유롭게 배치
- 시작:
  - 신호등 초록색 유지
  - z키 눌러서 자율주행모드 진입
- 진행:
  - 3바퀴 주행,
  - 마지막 바퀴에서 신호등 빨간색으로 변경
  - 장애물은 2개 랜덤하게 배치 (인코스에 배치할 예정)



신호등은 B 버튼 활용





Brake 폴더에 이미지 저장 됨

브레이크 추가

1. 지금 model.py는 class 3개만 학습함 (직진, 좌회전 우회전)

2. model.py를 수정하여 brake class도 추가

3. brake 데이터는 키보드에서 "b" 키를 눌러 취득 (이미지 저장 시)

코드 수정

1. 이미지 파일을 불러옴

현재는 class가 3개만 있음
"go", "left", "right"
각각 라벨은 0, 1, 2

같은 방식으로 brake가 추가 되어야 함
brake는 라벨 값 3을 갖도록 추가

data_path = … + "brake"
labels.append(3)

```
10  # 이미지 경로와 라벨을 저장할 리스트 생성
11  data = []
12  labels = []
13  test = []
14
15  # 이미지 데이터를 읽어오고 라벨을 저장: 직진
16  data_path = ".." + os.sep + "image" + os.sep + "go"
17  for img in os.listdir(data_path):
18      image = cv2.imread(os.path.join(data_path, img))
19      data.append(image)
20      test.append(img)
21      labels.append(0)  # 라벨 0
22
23  # 이미지 데이터를 읽어오고 라벨을 저장: 좌회전
24  data_path = ".." + os.sep + "image" + os.sep + "left"
25  for img in os.listdir(data_path):
26      image = cv2.imread(os.path.join(data_path, img))
27      data.append(image)
28      test.append(img)
29      labels.append(1)  # 라벨 1
30
31  # 이미지 데이터를 읽어오고 라벨을 저장: 우회전
32  data_path = ".." + os.sep + "image" + os.sep + "right"
33  for img in os.listdir(data_path):
34      image = cv2.imread(os.path.join(data_path, img))
35      data.append(image)
36      test.append(img)
37      labels.append(2)  # 라벨 2
```

코드 수정

3. 학습할 모델을 정의하고 학습
   모델: Convolutional Neural Network

```
51  # Convolutional Neural Network Model
52  model = tf.keras.Sequential()
53  model.add(layers.Conv2D(64, (3, 3), activation='relu', input_shape=(64, 64, 3)))
54  model.add(layers.MaxPooling2D((2, 2)))
55  model.add(layers.Conv2D(128, (3, 3), activation='relu'))
56  model.add(layers.MaxPooling2D((2, 2)))
57  model.add(layers.Conv2D(64, (3, 3), activation='relu'))
58  model.add(layers.MaxPooling2D((2, 2)))
59  model.add(layers.Conv2D(32, (3, 3), activation='relu'))
60  model.add(layers.Flatten())
61  model.add(layers.Dense(64, activation='relu'))
62  model.add(layers.Dropout(0.5))
63  model.add(layers.Dense(3, activation='softmax'))
64
65  # 모델 학습
66  model.compile(loss='sparse_categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
67  history = model.fit(X_train, Y_train, validation_data=(X_valid, Y_valid), epochs=30, batch_size = 128)
```

3→4로 변경
이제 class가 4개로 바뀜