
ONECLICK

대회 평가 기준

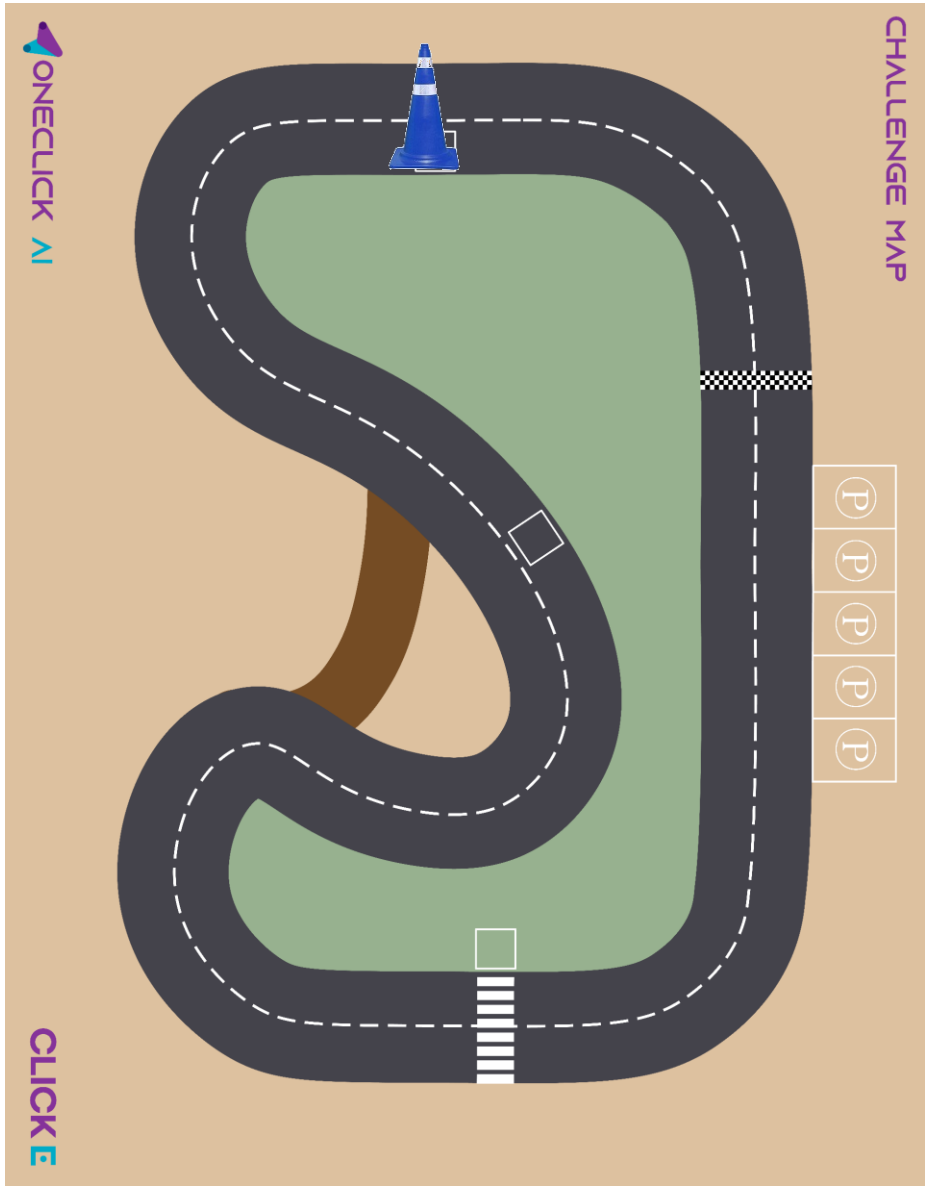
대회 평가 기준

- 레이싱 2팀 대결
- 3판 2선승
- 시계방향 2바퀴 주행
- 장애물 1개 (감점은 없음)

- 한 차량은 앞에서 시작하고 다른 차량은 뒤에서 시작, 즉 시작점이 다름!
- 2바퀴를 주행하고 시작점을 먼저 지나가야 승리!

- 아웃필드(갈색 구간)는 4바퀴 모두 밖으로 나가도 됨
- 단, 인필드(초록색인 구간)은 최대 2바퀴만 나갈 수 있음
- 지름길 활용 가능

- 차량이 경기장 밖으로 나가서 다시 돌아오지 못할 경우 탈락



키보드



소프트웨어 변경

모터 속도 변경 가능 (config.py)

GO_OUTPUT: 직진 속도 (%)

TURN_OUTPUT: 좌회전 또는 우회전시 더 천천히 회전하는 바퀴 속도 (%)

예를들어 좌회전 할때

왼쪽 바퀴는 속도 10, 오른쪽 바퀴는 속도 30

→ 더 잘 회전할 수 있음

우회전 할때

왼쪽 바퀴는 속도 30, 오른쪽 바퀴는 속도 10

→ 더 잘 회전할 수 있음

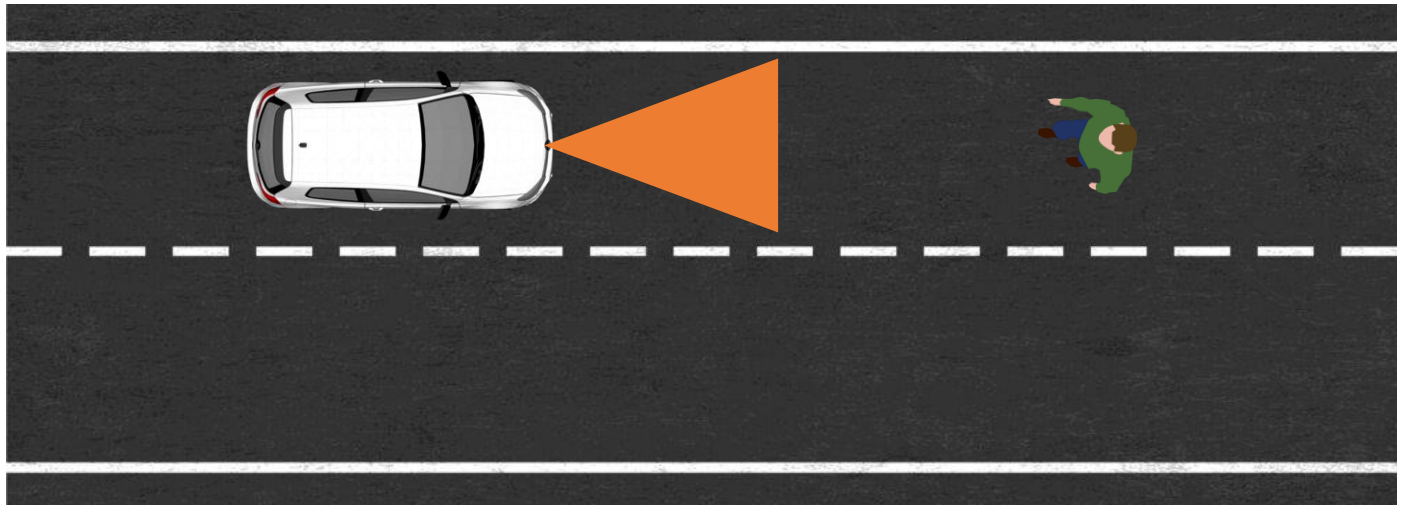
```
# config.py
…
# ==================== 서보 모터 각도 설정 ====================
NEUTRAL_DEG = 83    # 중립 (직진)
LEFT_DEG = 60       # 좌회전
RIGHT_DEG = 105     # 우회전

# ==================== 모터 속도 설정 ====================
GO_OUTPUT = 20      # 직진 속도 (PWM 듀티 사이클 0-100)
TURN_OUTPUT = 20    # 회전 속도 (PWM 듀티 사이클 0-100)
```

GO_OUTPUT, TURN_OUTPUT을 마음대로 변경!
(최대 100)

제어 전략

고속 자율주행은 제어적으로 어려운 문제

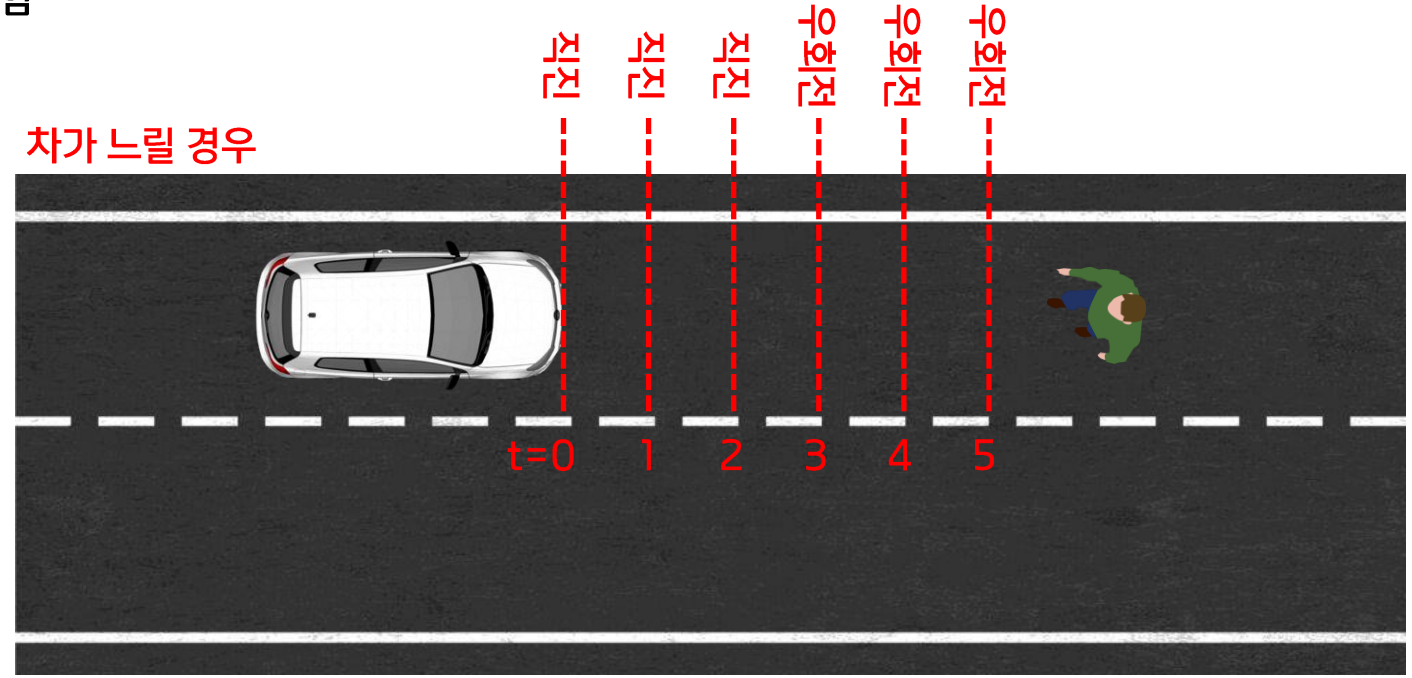
- 차량 속도가 느린 경우, 제어 속도가 중요한 요소가 아님
- 단, 차량이 빠르면 빠를수록 제어속도도 빨라져야 함
- 또한, 모델도 더 정확해야 함



제어 전략

고속 자율주행은 제어적으로 어려운 문제

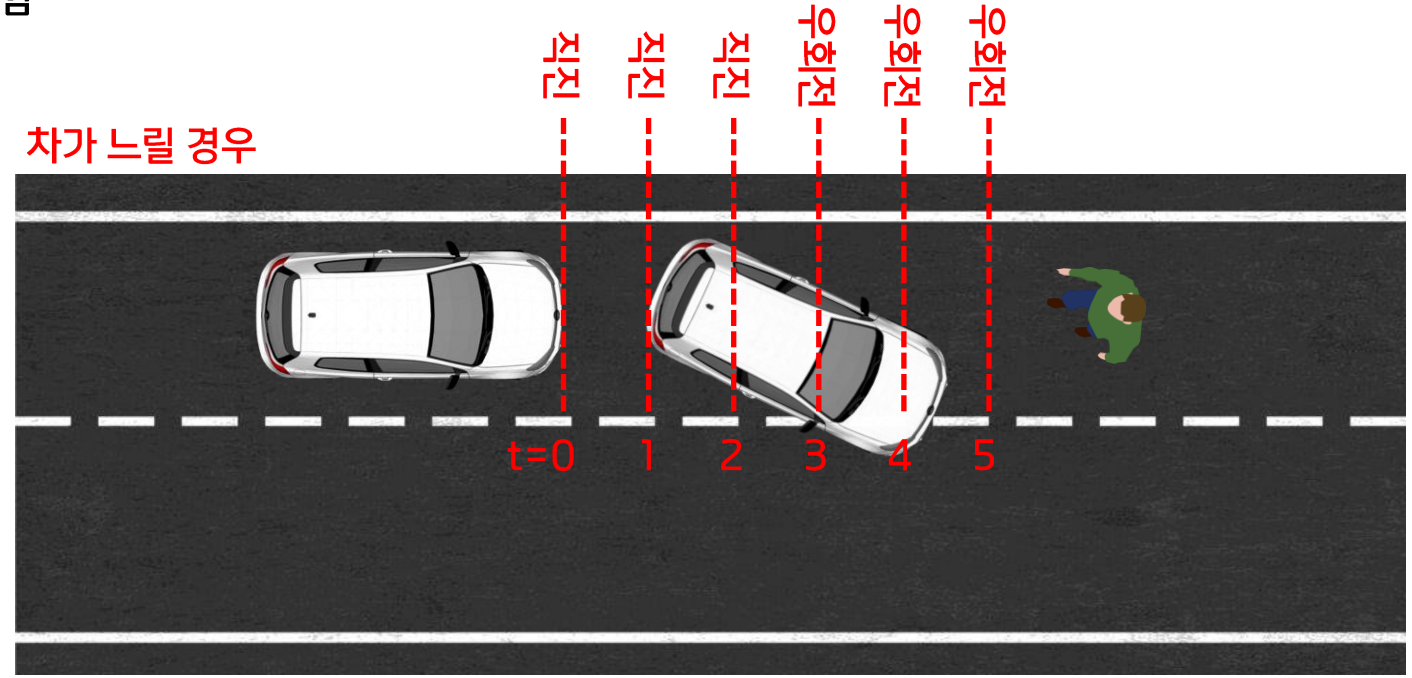
- 차량 속도가 느린 경우, 제어 속도가 중요한 요소가 아님
- 단, 차량이 빠르면 빠를수록 제어속도도 빨라져야 함
- 또한, 모델도 더 정확해야 함



제어 전략

고속 자율주행은 제어적으로 어려운 문제

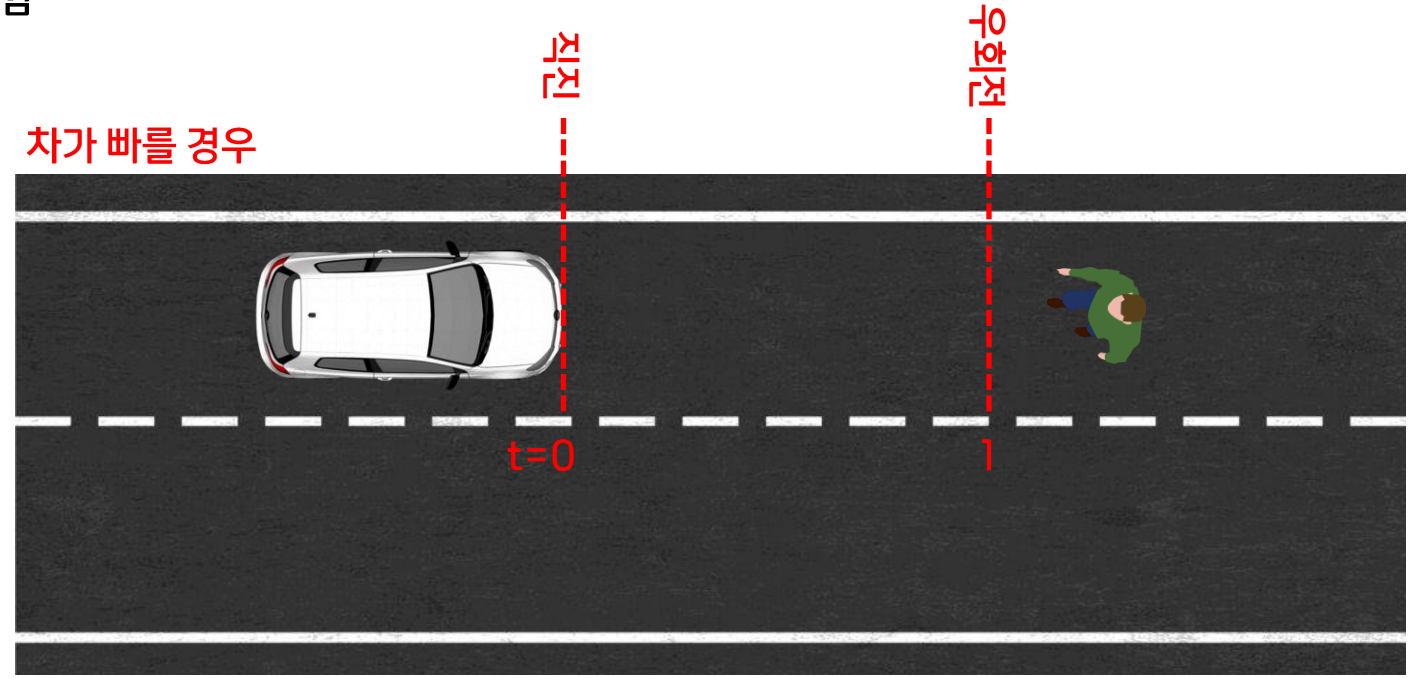
- 차량 속도가 느린 경우, 제어 속도가 중요한 요소가 아님
- 단, 차량이 빠르면 빠를수록 제어속도도 빨라져야 함
- 또한, 모델도 더 정확해야 함



제어 전략

고속 자율주행은 제어적으로 어려운 문제

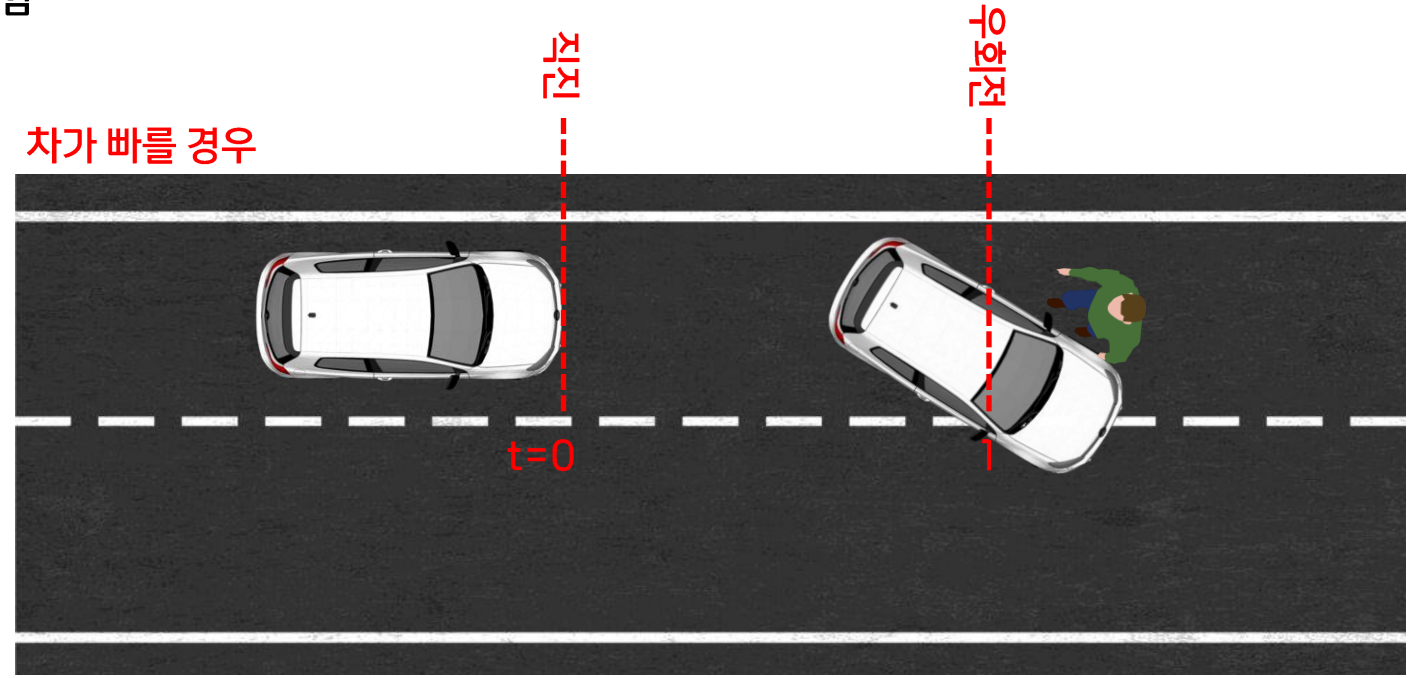
- 차량 속도가 느린 경우, 제어 속도가 중요한 요소가 아님
- 단, 차량이 빠르면 빠를수록 제어속도도 빨라져야 함
- 또한, 모델도 더 정확해야 함



제어 전략

고속 자율주행은 제어적으로 어려운 문제

- 차량 속도가 느린 경우, 제어 속도가 중요한 요소가 아님
- 단, 차량이 빠르면 빠를수록 제어속도도 빨라져야 함
- 또한, 모델도 더 정확해야 함



제어 전략

고속 자율주행은 제어적으로 어려운 문제

- 차량 속도가 느린 경우, 제어 속도가 중요한 요소가 아님
- 단, 차량이 빠르면 빠를수록 제어속도도 빨라져야 함
- 또한, 모델도 더 정확해야 함

제어속도에 영향을 미치는 요소

- 기본 동작을 위한 연산 (i.e. 이미지 crop, resize)
- 제어 모델 연산 (Tensorflow: model.predict)
- 카메라 (30fps)
- DC모터, 서보모터 반응속도
- 제어용 통신

} 최소한 100ms 안에 돌아야 함

하지만 차량 속도가 2배 늘어난다면?

제어 전략

```
def capture_img(self):              # camera_capture.py
    …
    try:
…
…
            # 화면에 띄울 이미지 선정 (show_image)
            frame = cv2.resize(frame, (512, 512))
            self.show_image = display_frame
            cv2.imshow('Frame', self.show_image)
            cv2.waitKey(40)
…

def autonomous_control_loop():      # main.py
    t0 = 0
    predicted_action = None

    while True:
        time.sleep(CONTROL_LOOP_SLEEP)
        # 자율주행 모드
        if key.manual == 0:
            …
```

FPS

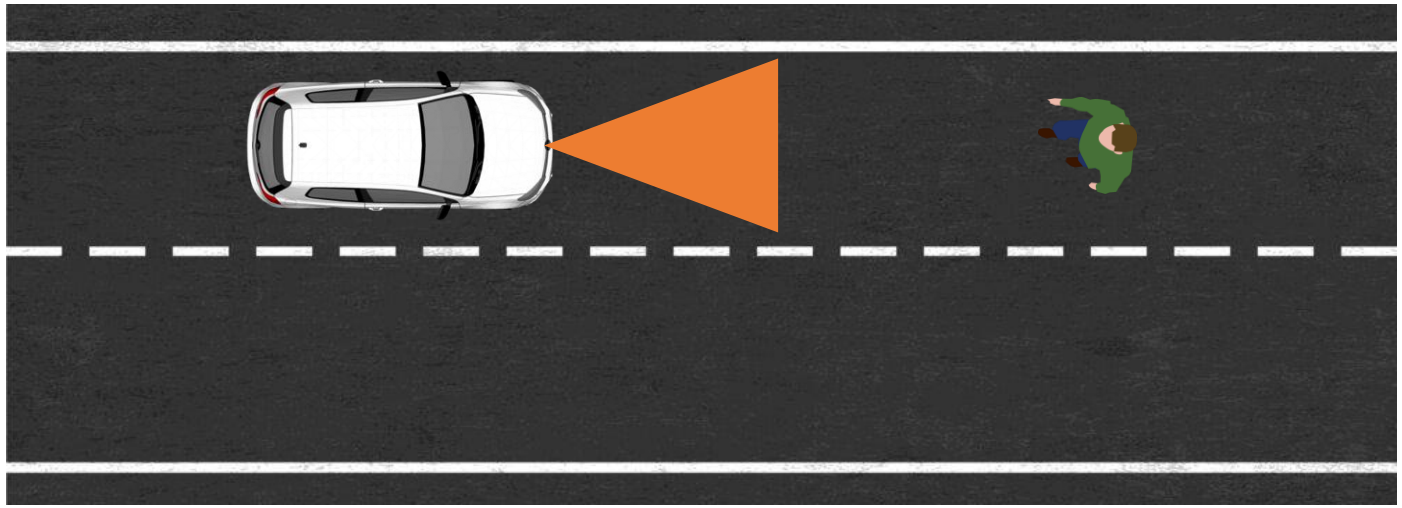
Loop Delay

제어 전략

고속 자율주행은 제어적으로 어려운 문제

- 차량 속도가 느린 경우, 제어 속도가 중요한 요소가 아님
- 단, 차량이 빠르면 빠를수록 제어속도도 빨라져야 함
- 또한, 모델도 더 정확해야 함

차가 느릴 경우

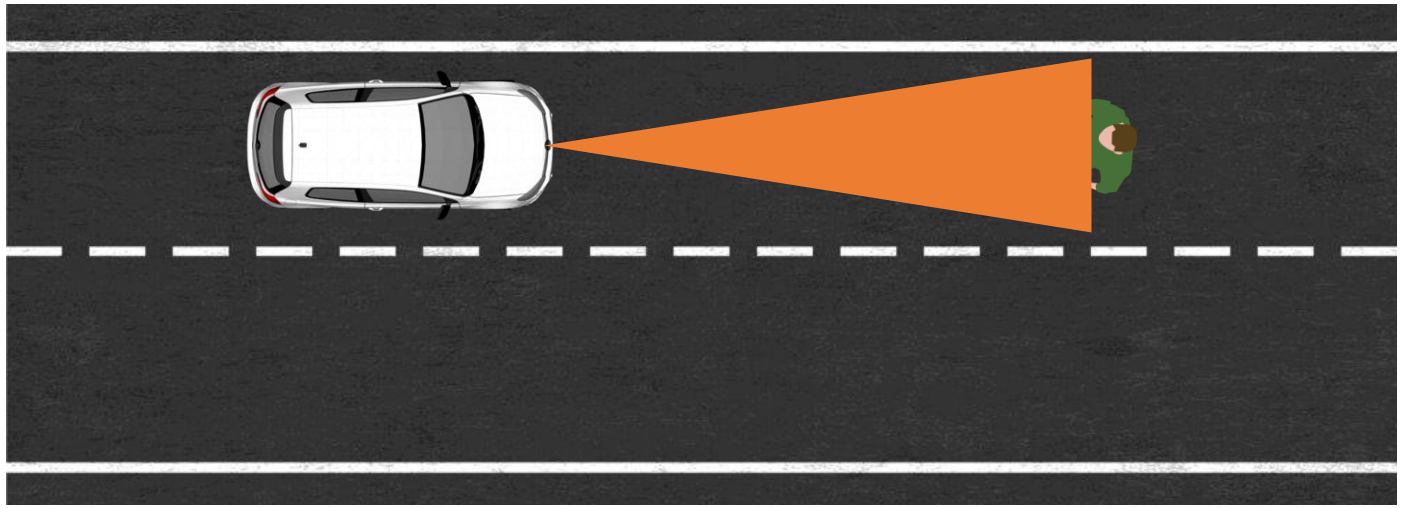


제어 전략

고속 자율주행은 제어적으로 어려운 문제

- 차량 속도가 느린 경우, 제어 속도가 중요한 요소가 아님
- 단, 차량이 빠르면 빠를수록 제어속도도 빨라져야 함
- 또한, 모델도 더 정확해야 함

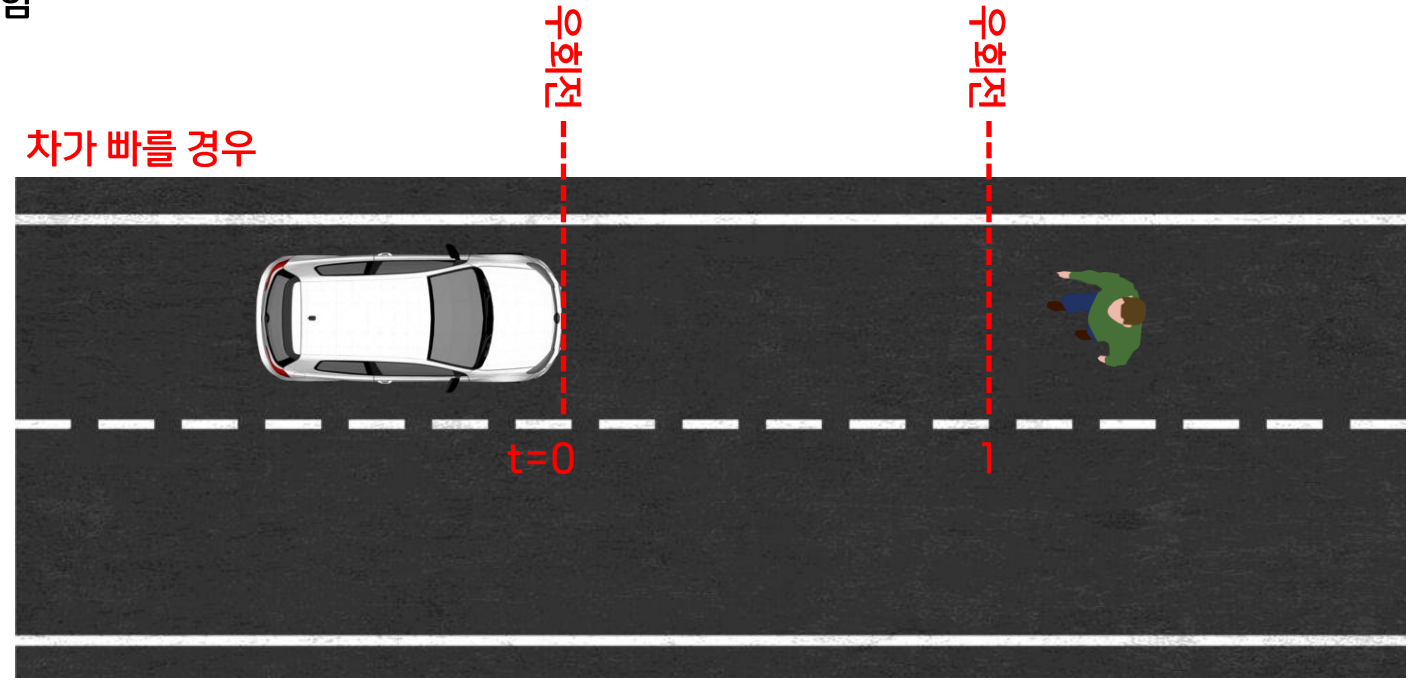
차가 빠를 경우



제어 전략

고속 자율주행은 제어적으로 어려운 문제

- 차량 속도가 느린 경우, 제어 속도가 중요한 요소가 아님
- 단, 차량이 빠르면 빠를수록 제어속도도 빨라져야 함
- 또한, 모델도 더 정확해야 함

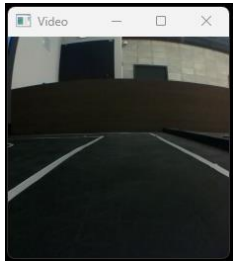


모델 학습 – 학습 전략 수립

- '양질의 데이터'를 '많이' 쌓는 것이 중요
- 최소 1000장 이상 필요
- 도로 중앙으로 정확히 주행하기 위해 직선 코스에서 좌우 조향을 많이 하면 오히려 학습에 불리
  - 도로 내 정상 주행이지만 정중앙이 아닌 경우와 확실한 좌/우 커브코스를 동일한 라벨로 학습하게 됨
  - Q. 정중앙 주행을 위해선 어떤 알고리즘이 필요할까?

- 저장할 이미지 기준 수립
  - 카메라 각도 및 높이 조절
  - 3인칭(도로 내 차량 위치 파악) 주행 vs 1인칭(수신 비디오 영상) 주행





- 이미지의 어느 부분을 사용할 것인지?
- 비율 변경 시 camera_capture.py 수정

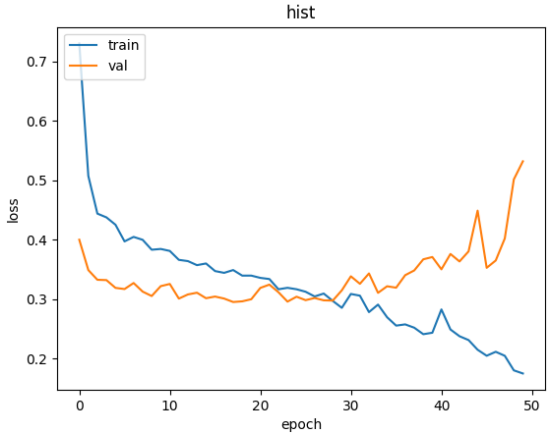
모델 학습 – 학습 전략 수립

model.py

```
model.compile(loss='sparse_categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
history = model.fit(X_train, Y_train, validation_data=(X_valid, Y_valid), epochs=10, batch_size = 128)
```

- 얼마만큼 학습을 반복할 것인지? : epoch, batch 조절
  - Ex) 총 반복 횟수를 늘리고 싶다 → epoch 증가(학습소요시간도 함께 증가)
  - Ex) parameter 1회 업데이트에 필요한 데이터양을 늘리고 싶다 → batch 조절

- 무조건 데이터를 많이, 여러 번 한다고 성능이 좋아지는 것은 아님
  - Accuracy와 loss값의 변화에 유의
  - Validation accuracy가 0.8이상 되어야 사용 가능



Validation 데이터의 loss와 정확도

```
Epoch 16/50
49/49 [==============================] - 37s 755ms/step - loss: 0.3475 - accuracy: 0.8535 - val_loss: 0.3046 - val_accuracy: 0.8667
Epoch 17/50
49/49 [==============================] - 25s 501ms/step - loss: 0.3445 - accuracy: 0.8494 - val_loss: 0.3013 - val_accuracy: 0.8767
Epoch 18/50
49/49 [==============================] - 25s 510ms/step - loss: 0.3492 - accuracy: 0.8564 - val_loss: 0.2955 - val_accuracy: 0.8833
Epoch 19/50
49/49 [==============================] - 38s 790ms/step - loss: 0.3398 - accuracy: 0.8581 - val_loss: 0.2966 - val_accuracy: 0.8700
Epoch 20/50
37/49 [=====================>........] - ETA: 10s - loss: 0.3430 - accuracy: 0.8443
```