서울시립대학교
UNIVERSITY OF SEOUL

SMART MOBILITY DESIGN

라운지 트랙 자율주행

중간 보고서

00조

박상윤 강OO 권OO 정OO

Contents

서울시립대학교
UNIVERSITY OF SEOUL

서울시립대학교
UNIVERSITY OF SEOUL

기존 계획

05/30 ~ 06/05	횡단보도 & 정지 구간 인식 구현
06/06 ~ 06/12	장애물 회피 & 터널 주행 구현
06/13 ~ 06/19	시간 단축 및 다양한 환경에서 안정적인 주행을 위한 튜닝

서울시립대학교
UNIVERSITY OF SEOUL

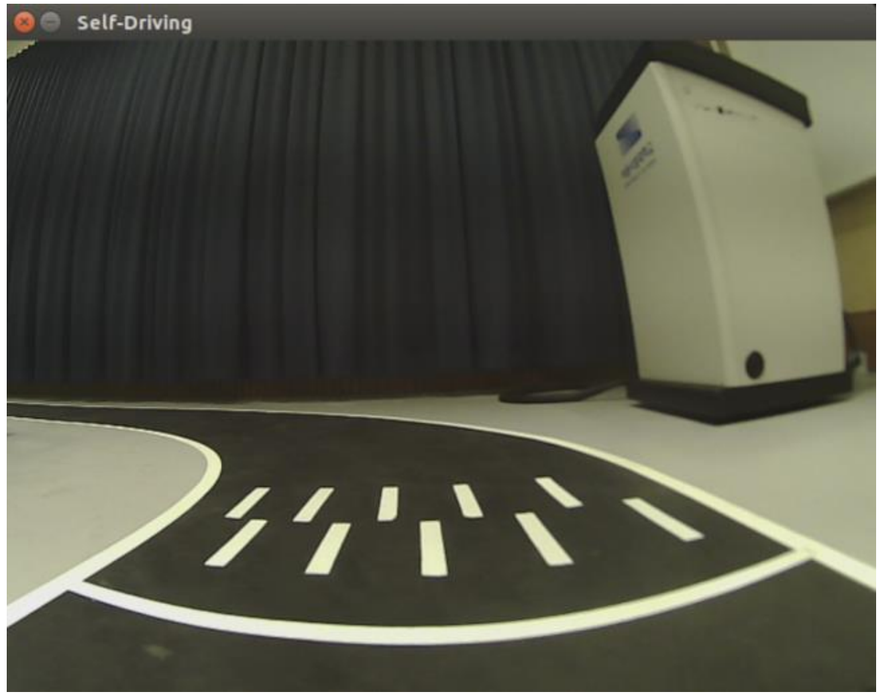
이미지 전처리

Calibration 후 Fish Eye 렌즈로 인한 왜곡 보정 - 횡단보도


```
Calibration successful!

('Camera matrix:\n', array([[ 344.46340212,    0.        ,  316.61958476],
       [   0.        ,  344.16639385,  221.98908518],
       [   0.        ,    0.        ,    1.        ]]))
('Distortion coefficients:\n', array([[-0.35127937,  0.18360865, -0.00103499, -0
.00391777, -0.05558141]]))
('New camera matrix:\n', array([[  62.4901123 ,    0.        ,  127.05087526],
       [   0.        ,  121.9288559 ,  156.50383536],
       [   0.        ,    0.        ,    1.        ]]))
('roi:\n', (0, 0, 0, 0))
```

Calibration 결과



횡단보도 보정 전

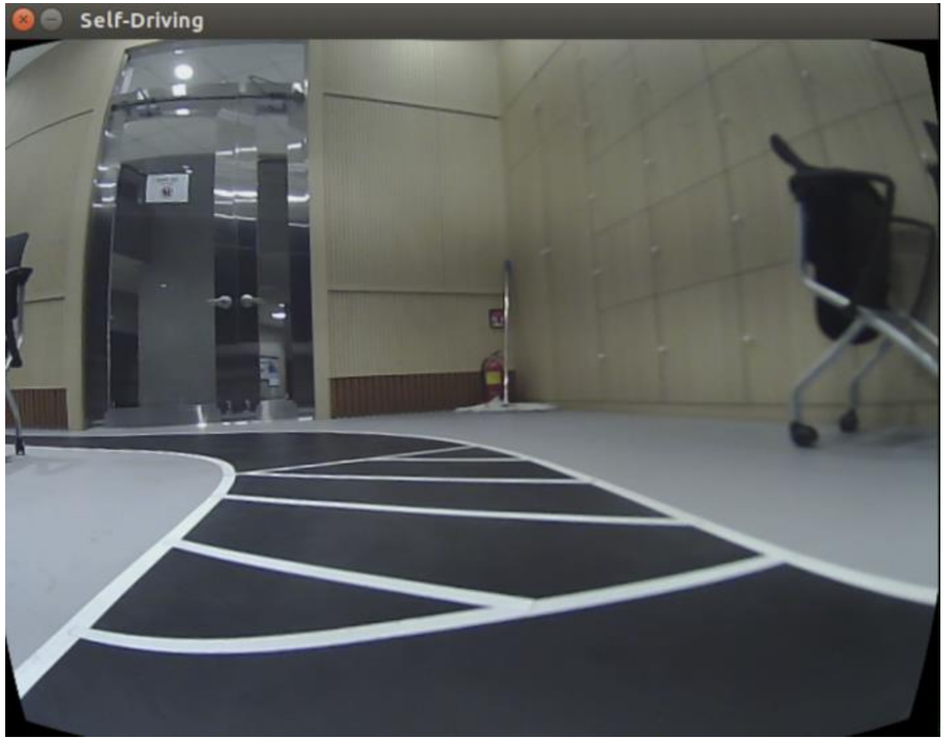
횡단보도 보정 후

서울시립대학교
UNIVERSITY OF SEOUL

이미지 전처리

Calibration 후 Fish Eye 렌즈로 인한 왜곡 보정 - 정지 구간

정지 구간 보정 전

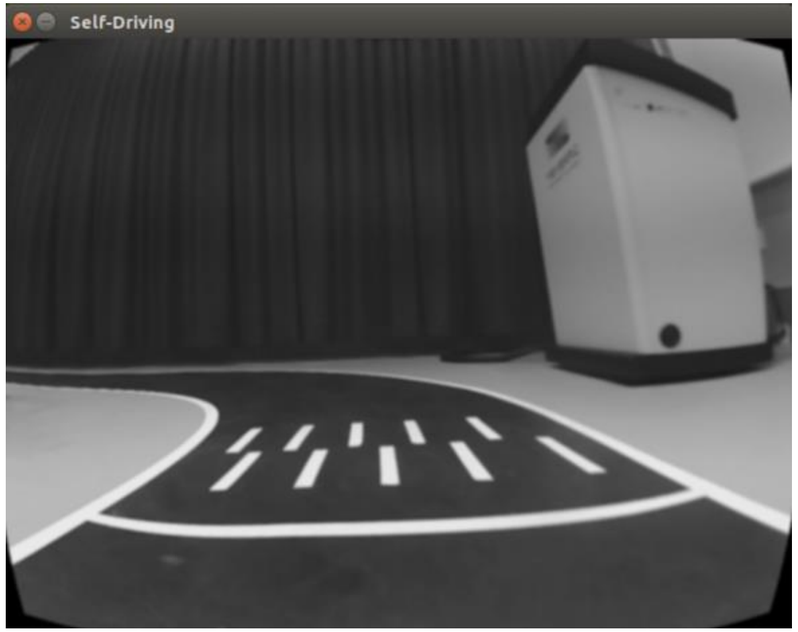


정지 구간 보정 후

서울시립대학교
UNIVERSITY OF SEOUL

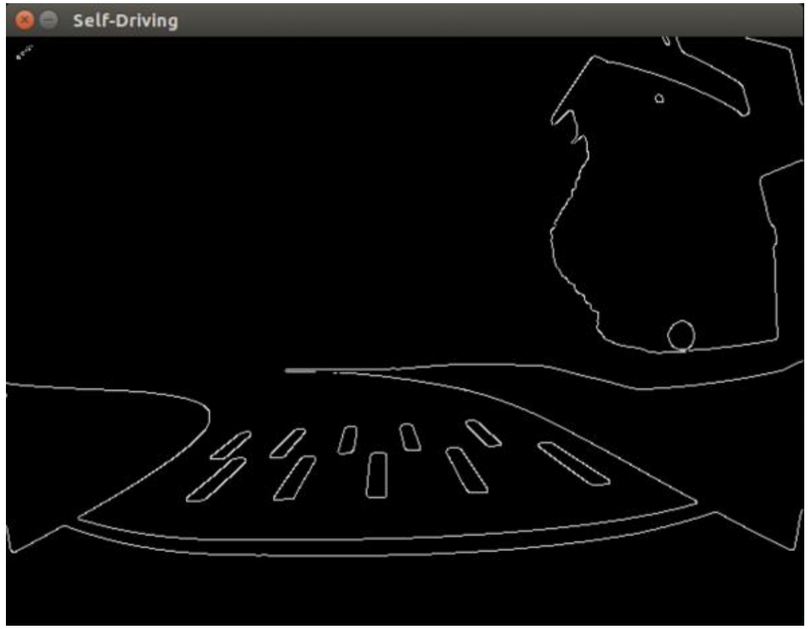
이미지 전처리

Gaussain Blur & 이진화 & Canny Edge Detection



Gray scale + Gaussian Blur

이진화



Canny Edge Detection

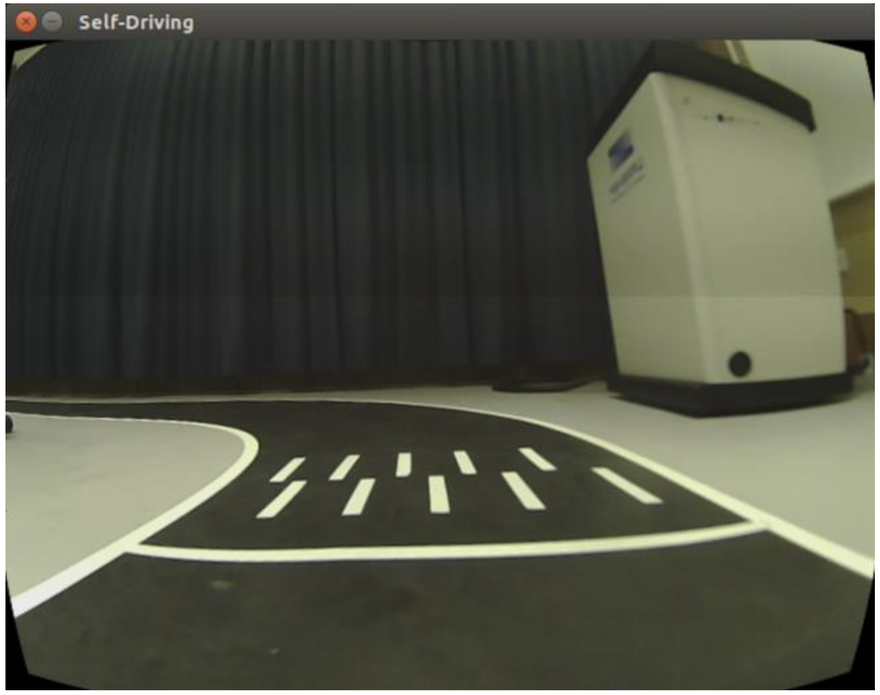
이미지 전처리

Bird Eye View(BEV)

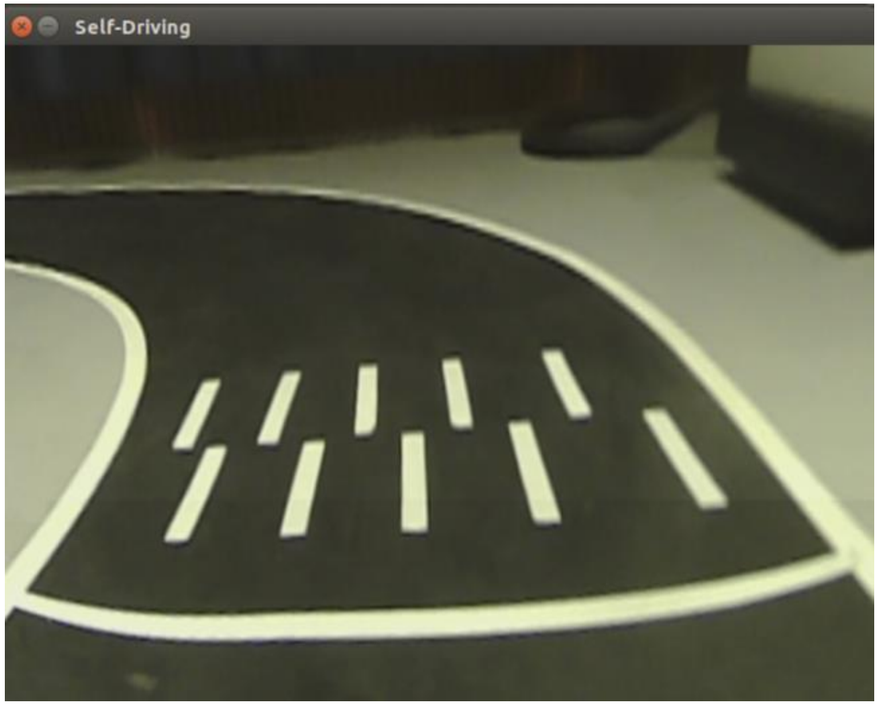
- 횡단보도 & 정지 구간에서만 적용

```python
# Bird Eye View
def bev(car):
    h, w = (480, 640)
    src = np.float32([
        [140, 240], # LEFT UP
        [500, 240], # RIGHT UP
        [40, 420],  # LEFT DOWN
        [600, 420]  # RIGHT DOWN
    ])

    dst = np.float32([
        [0, 0],
        [640, 0],
        [0, 480],
        [640, 480]
    ])
    M = cv2.getPerspectiveTransform(src, dst)
    car.bev_img = cv2.warpPerspective(car.img, M, (w, h))
    car.bev_edge_img = cv2.warpPerspective(car.edge_img, M, (w, h))
```



횡단보도 적용 전



횡단보도 적용 후

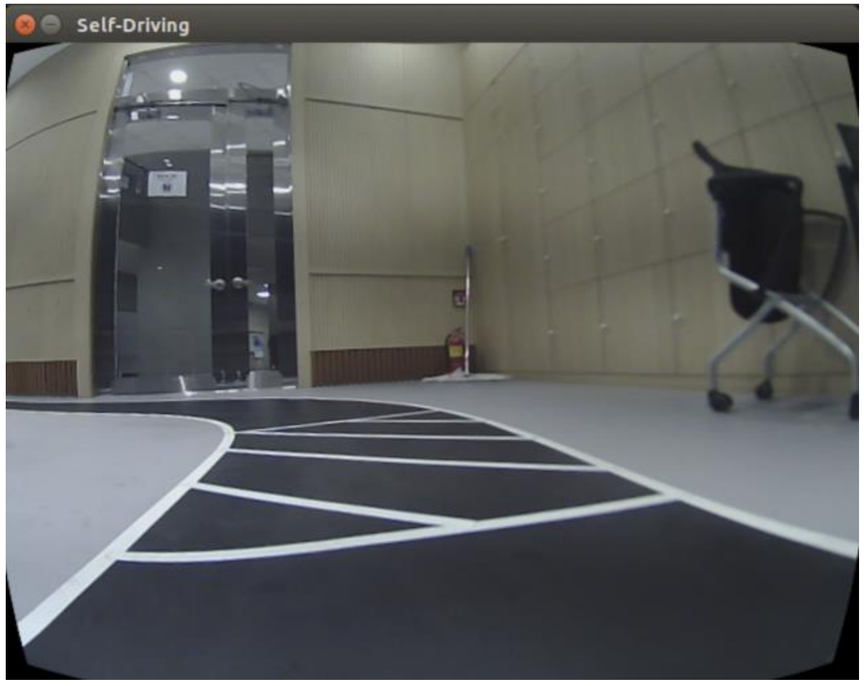
이미지 전처리

Bird Eye View(BEV)

- 횡단보도 & 정지 구간에서만 적용

```python
# Bird Eye View
def bev(car):
    h, w = (480, 640)
    src = np.float32([
        [140, 240], # LEFT UP
        [500, 240], # RIGHT UP
        [40, 420],  # LEFT DOWN
        [600, 420]  # RIGHT DOWN
    ])

    dst = np.float32([
        [0, 0],
        [640, 0],
        [0, 480],
        [640, 480]
    ])
    M = cv2.getPerspectiveTransform(src, dst)
    car.bev_img = cv2.warpPerspective(car.img, M, (w, h))
    car.bev_edge_img = cv2.warpPerspective(car.edge_img, M, (w, h))
```



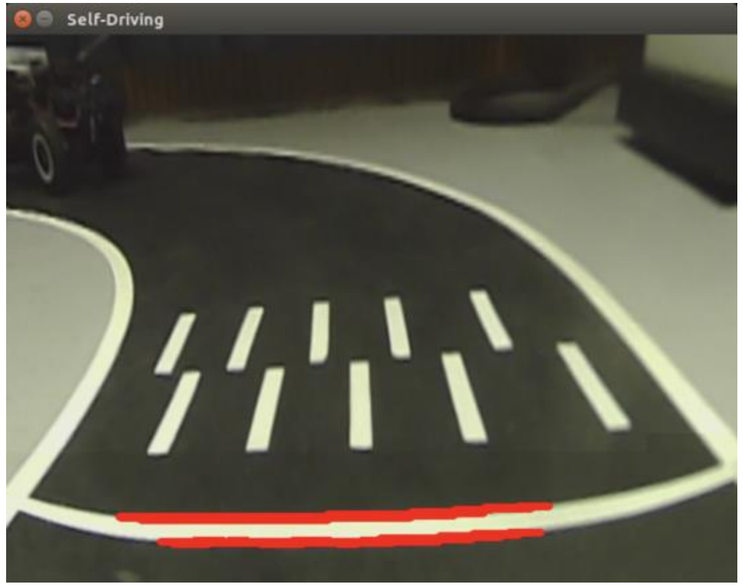
정지 구간 적용 전

정지 구간 적용 후

서울시립대학교
UNIVERSITY OF SEOUL

정지선

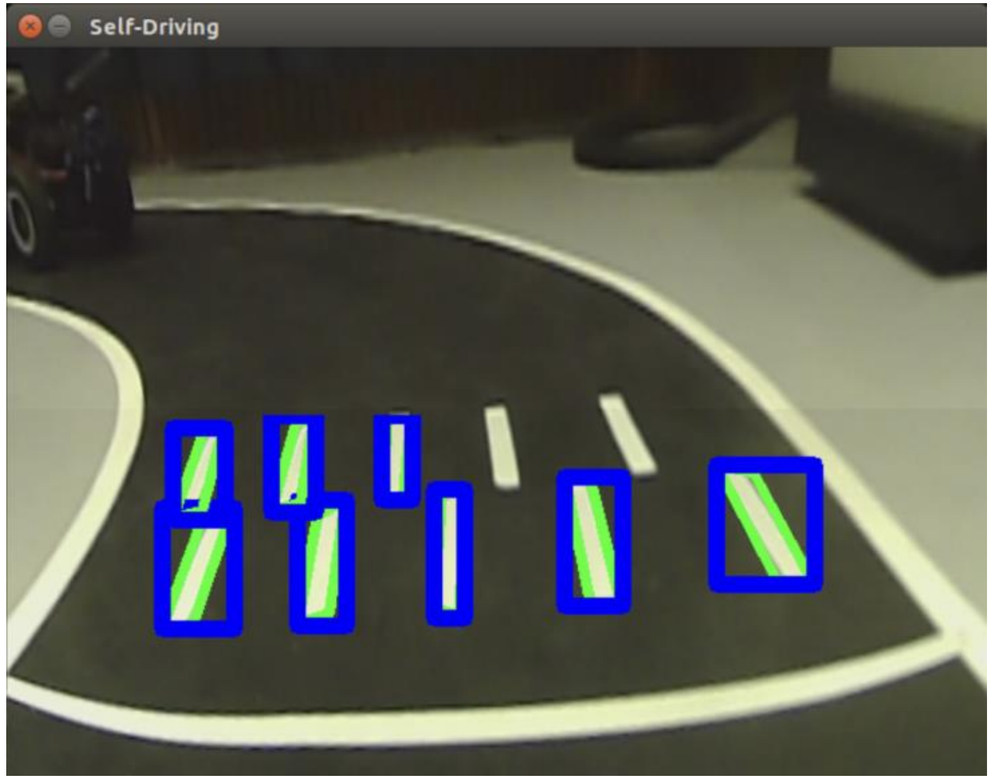
전처리한 이미지에서 Hough 변환 & 기울기로 filtering

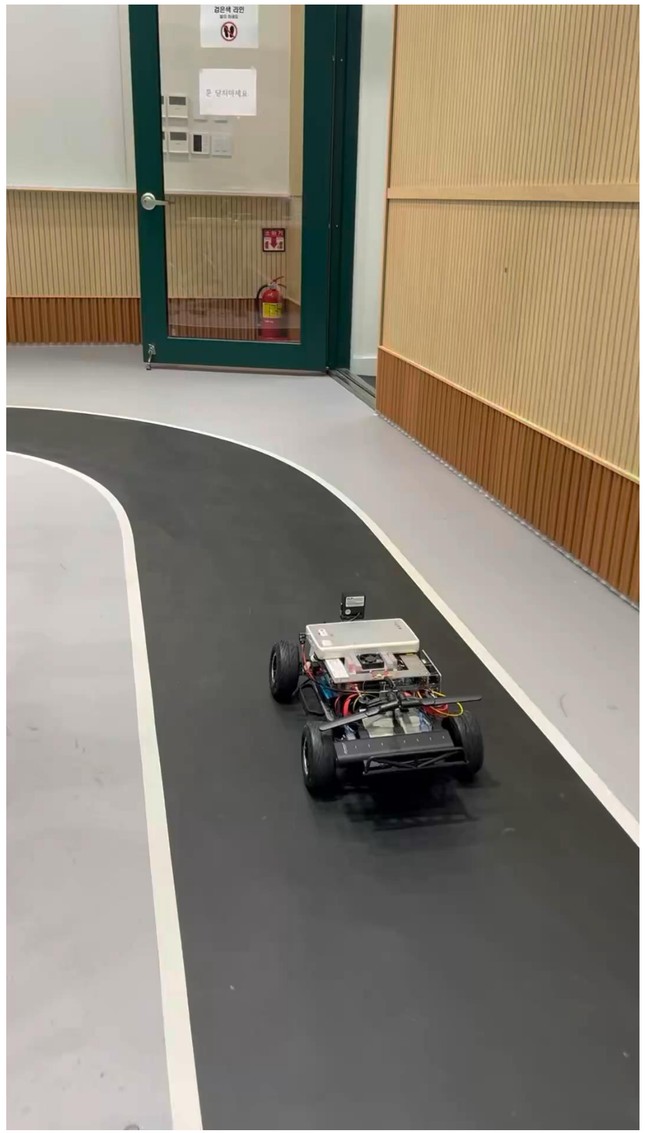


서울시립대학교
UNIVERSITY OF SEOUL

횡단보도

전처리한 이미지에서 사각형의 Contour(윤곽선)을 검출

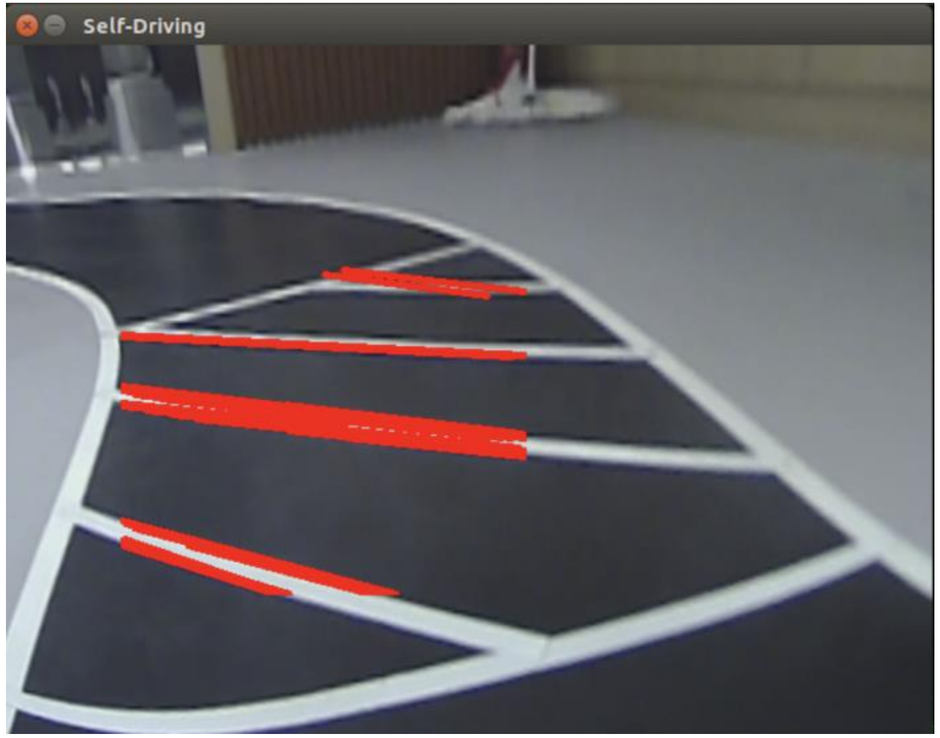


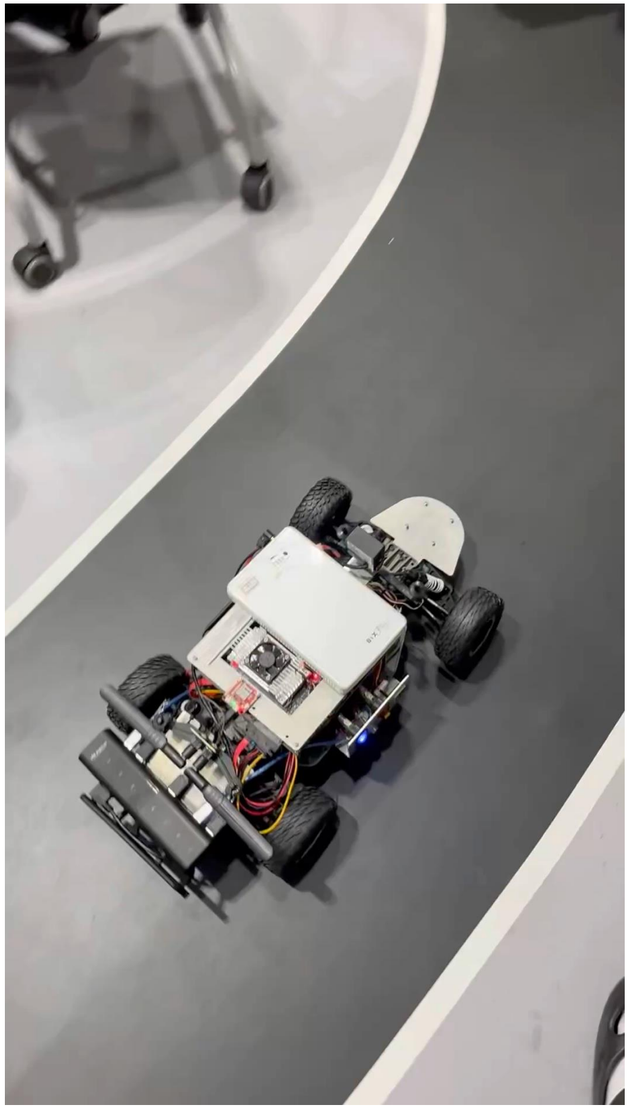


서울시립대학교
UNIVERSITY OF SEOUL

정지 구간

전처리한 이미지에서 대각선 검출 (Hough 변환)
& 새로운 정지선이 발견될 때까지 주행



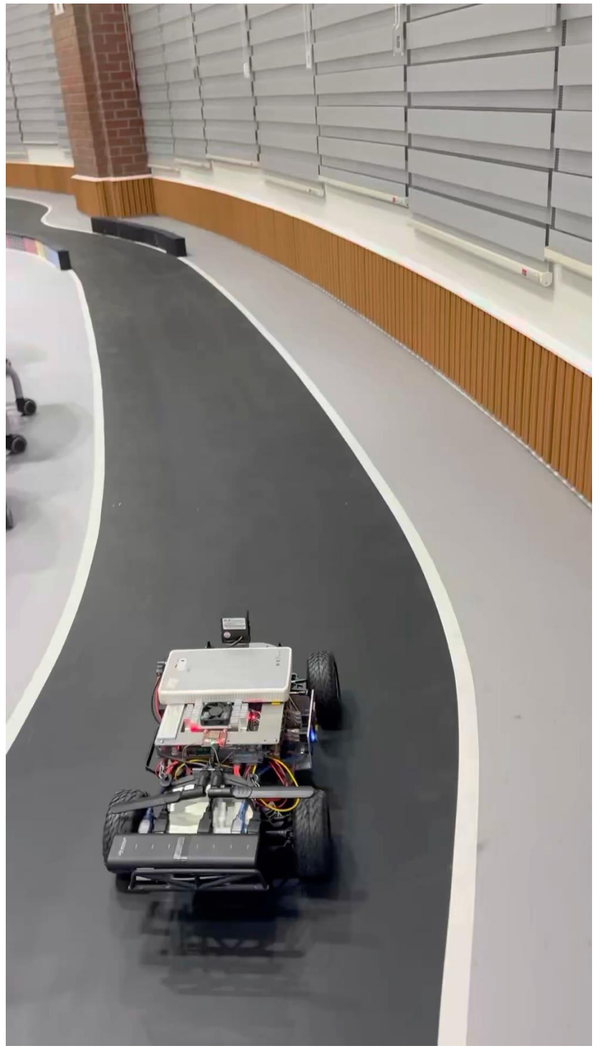


서울시립대학교
UNIVERSITY OF SEOUL

터널 주행

좌우 30~60도 범위의 각 거리 값 평균을 비교하여 조향 각도 계산

```
K = 30
while is_tunnel(car):
    angle = int((-car.tunnel_left + car.tunnel_right) * K)
    motor_control(car, angle=angle, speed=5)
```

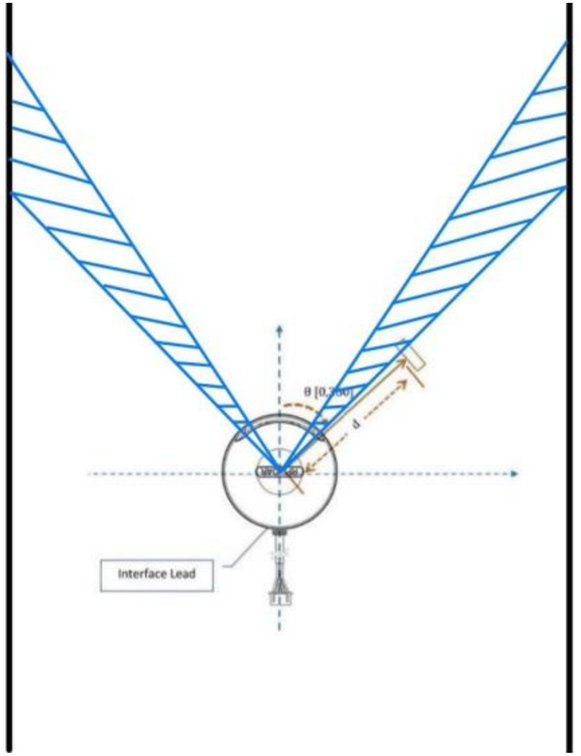


서울시립대학교
UNIVERSITY OF SEOUL

장애물 회피 주행

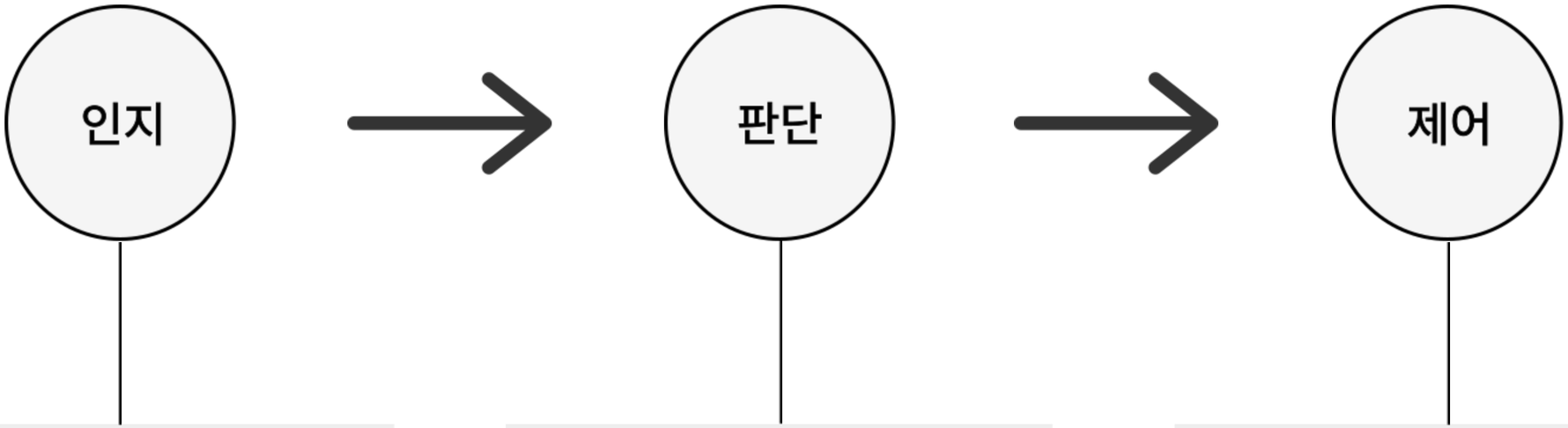
알고리즘

1. 좌/우 전방 장애물 인식
2. 장애물 위치에 따른 회피
3. 회피 후 차선 인식 주행



01 진행 현황

서울시립대학교
UNIVERSITY OF SEOUL



카메라, LiDAR 센서로 주변 환경 인지

✅ img_callback( )
✅ lidar_callback( )

횡단보도 및 정지 구간 판단
✅ is_crosswalk( )
✅ is_stopline( )

장애물 및 터널 구간 판단
❌ is_obstacle()
✅ is_tunnel()

판단에 따른 차량 제어

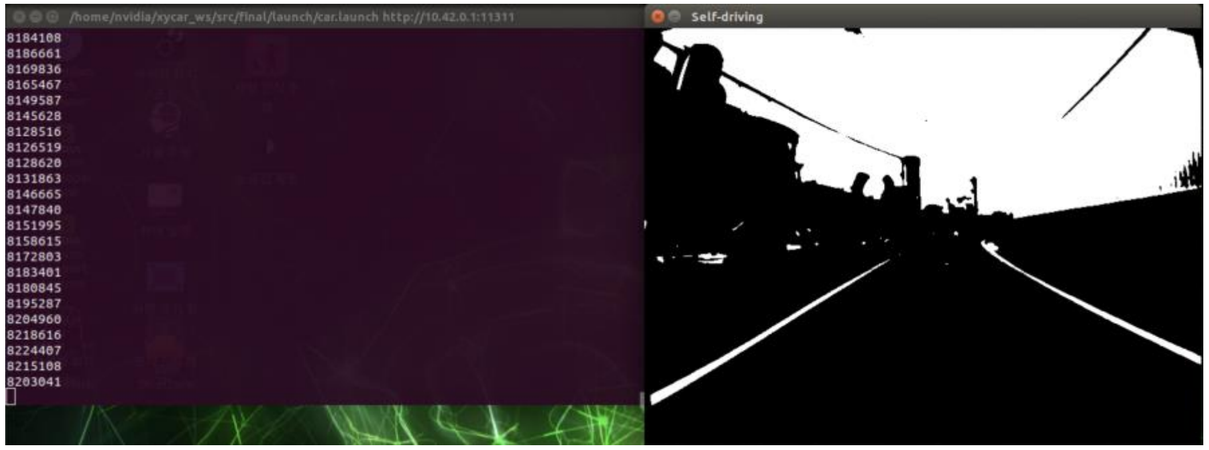
❌ avoid_obstacle( )
✅ wait_crosswalk( )
✅ tunnel_drive()
✅ finish_driving()

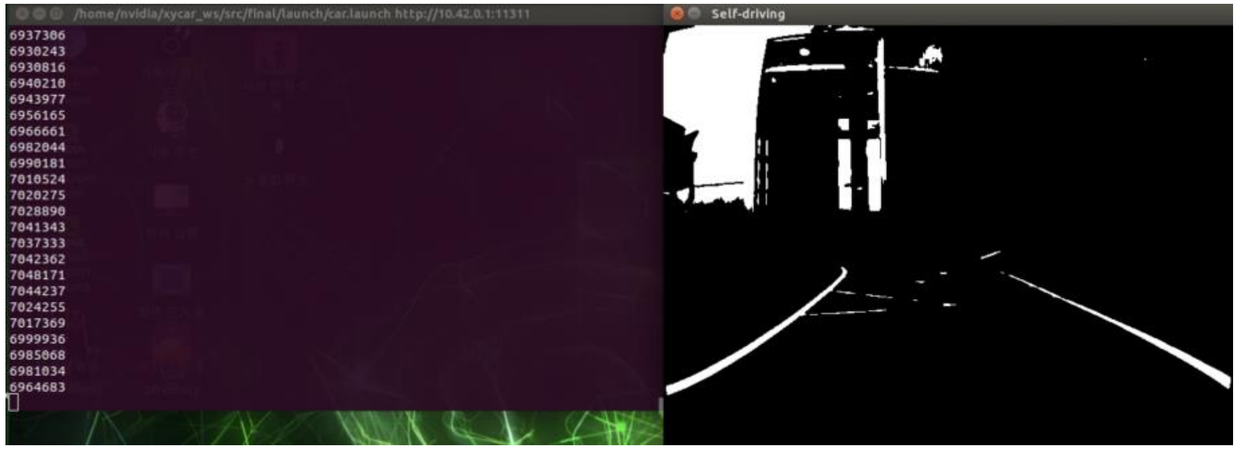
서울시립대학교
UNIVERSITY OF SEOUL

문제 - 조명 문제

```python
# Binary
BINARY_THRESHOLD = int(np.sum(self.blur_gray[300:480, :]) / 20000000 * 255)
self.binary = cv2.inRange(self.blur_gray, BINARY_THRESHOLD, 255)
```

문제	주변 밝기에 따라 detection 성능이 다름
해결 방법	Gray image의 ROI 내 pixel값(밝기) 합에 따라 이진화 기준을 다르게 설정





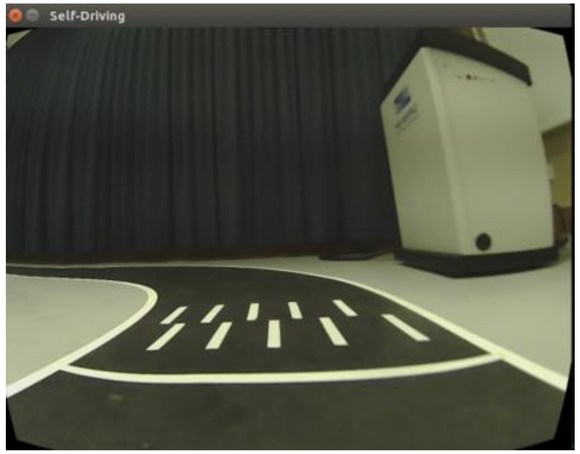
02 문제점 & 해결 방안

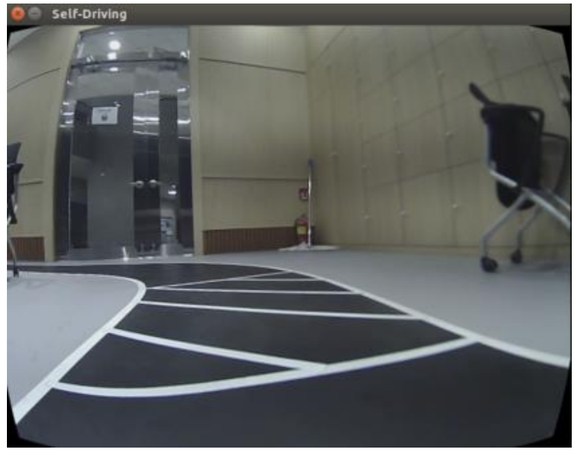
문제 - 카메라 왜곡

```python
# Calibration
camera_matrix = np.array([[352.15, 0, 315.03],
                          [0, 351.72, 216.93],
                          [0, 0, 1]])
dist_coeffs = np.array([-0.36, 0.2, 0.002, -0.002, 0])
new_camera_mtx, roi = cv2.getOptimalNewCameraMatrix(camera_matrix, dist_coeffs, (self.WIDTH, self.HEIGHT), 0.3, (self.WIDTH, self.HEIGHT))
self.img = cv2.undistort(self.img, camera_matrix, dist_coeffs, None, new_camera_mtx)
```

문제	Fish Eye 렌즈로 인한 왜곡
해결 방법	카메라 Calibration 후 왜곡 보정(undistort)

보정 전



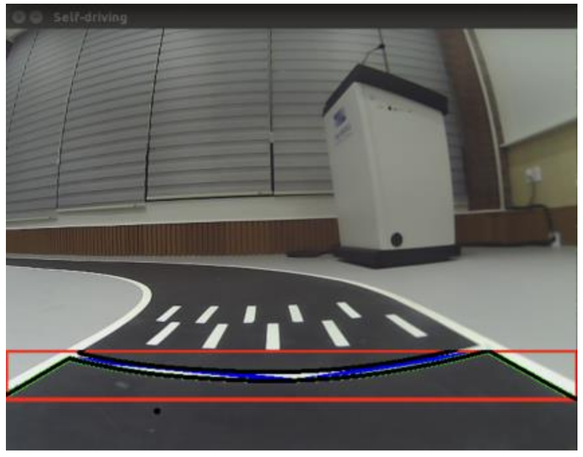


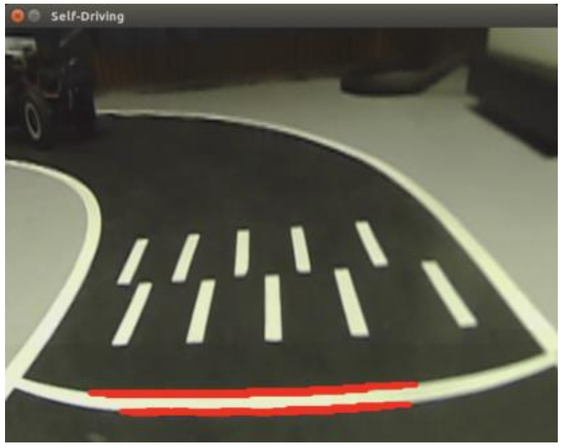
보정 후

서울시립대학교
UNIVERSITY OF SEOUL

문제 - 정지선 인식

처음 시도	Edge img에서 다각형 윤곽선(contour)를 검출
문제	정지선, 횡단보도, 정지 구간 인식 정확도가 낮음 이미지가 왜곡되어 정지선이 곡선으로 보임
해결 방법	Calibration 후 왜곡된 이미지를 보정(곡선 정지선 -> 직선 정지선) & Hough 변환

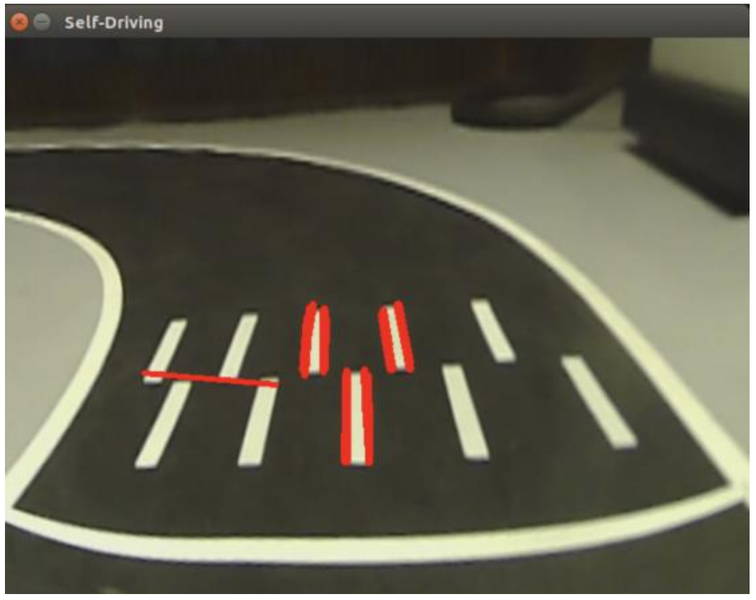


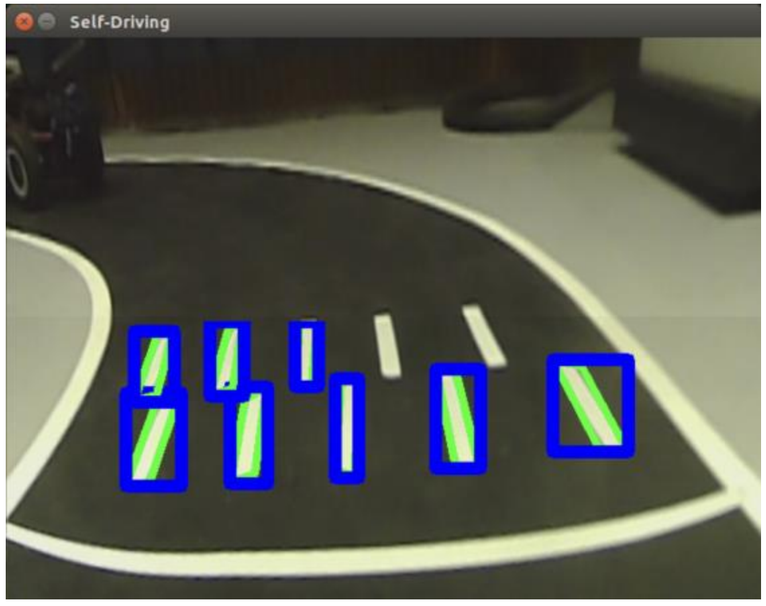


서울시립대학교
UNIVERSITY OF SEOUL

문제 - 횡단보도

처음 시도	횡단보도를 Hough 변환으로 검출
문제	정확한 인식 어려움 & 차선과 혼동 가능
해결 방법	사각형 contour를 검출하여 횡단보도 인식

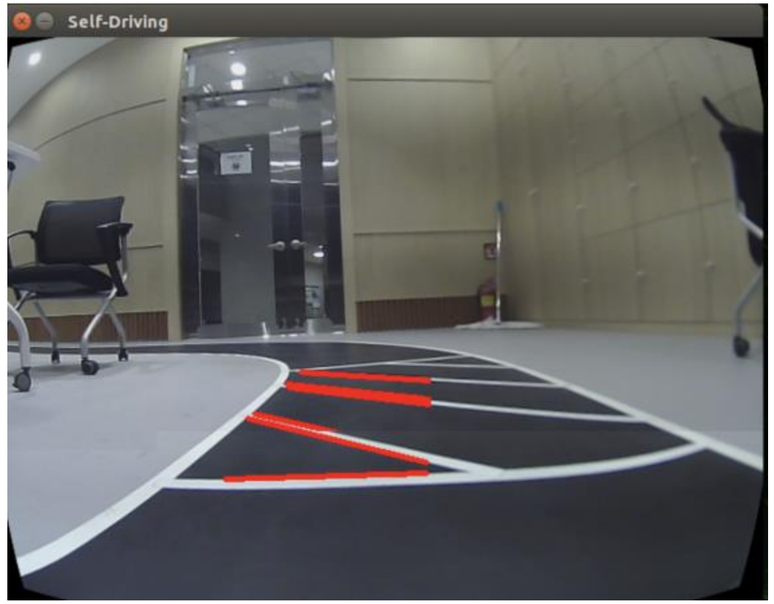


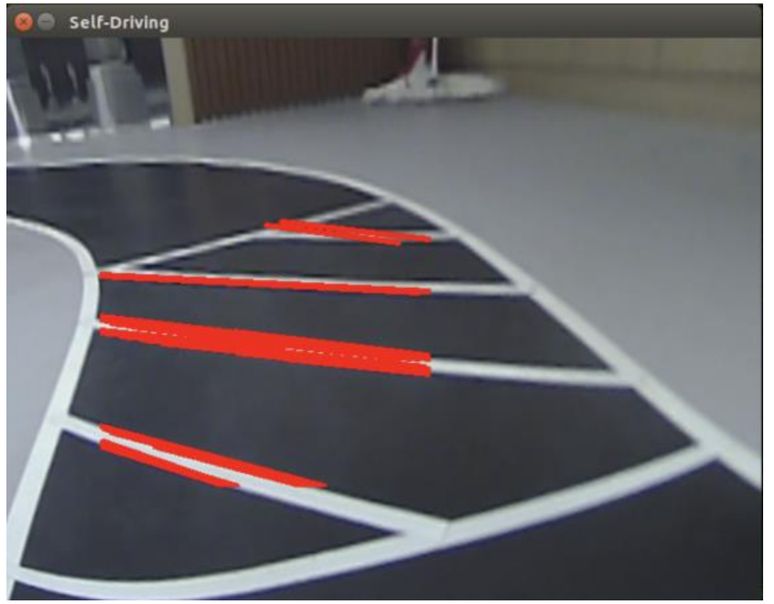


서울시립대학교
UNIVERSITY OF SEOUL

문제 - 정지 구간

처음 시도	BEV 없이 Hough 변환을 통해 대각선 검출
문제	정지선과 대각선 간 혼동
해결 방법	BEV를 적용하여 정지선과 대각선 간 기울기 차이를 키움





03 향후 계획

서울시립대학교
UNIVERSITY OF SEOUL

~ 06/13	장애물 회피 주행 구현
06/14 ~ 06/17	파인 튜닝 & 테스트
06/18 ~ 06/19	최종 보고서 작성

서울시립대학교
UNIVERSITY OF SEOUL

THANK YOU