창업연계공학설계입문

AD프로젝트 보고서

조 번호	1	조원 이름 (학번)	윤현승 20191634 오홍석 20191627 강영환 20143025 안금장 20181637 이성연 20191640
주제	인접한 차선의 차량 유무에 따른 안전한 차선 변경		
동기	자율주행자동차가 실생활에 적용되기 위해 필요한 조건을 생각하였습니다 현재 도로에는 2차선 3차선 4차선등이 많은데 자신이 원하는 목적지를 가기 위해서는 길을 찾는 능력도 중하지만 그걸 수행할 능력도 필요하다고 생각되었습니다 목적지를 가기 위해서 차가 직진만 할 수는 없기에 우회전이나 좌회전을 위하여 차선을 변경하는 방법을 마련하고자하여 차선변경이란 주제를 가지고 AD Project를 수행하게 되었습니다		
목표	같이 주행하는 차나 장애물이 있어도 차선변경을 수월하게 할 수 있게 하는 것이 목표입니다. 예를들어 3차선에서 주행시 현재 차가 2차선에서 달리는 중이라 할 때 목적지에 가기 위하여 우회전을 하여야 한다고 합니다. 현재 교통법상 3차선인 도로에서 2차선에 있는 차는 우회전을 할 수 없고 3차선에서만 가능합니다. 그러므로 자율주행자동차에는 필요시 차선을 바꿀 수 있는 능력이 필요한데 그 능력을 구현하는 것입니다. 차선 변경을 시도하기 전에 주변 장애물을 초음파 센서로 미리 탐지하여 장애물이 없어 안전하다고 인식될 때에만 차선을 변경하도록 구현할 것입니다. 차선 변경을 완료한 후, 주변 자동차의 속도가 모두 일정한 값을 갖는다는 조건하에, 차선변경후 부딪히지 않도록 뒤에 차량의 속도에 맞춰 주행하도록 구현할 것입니다.		

방법	1.차선 변경	차선변경 알고리즘
		
		1.도로 상에서 차선변경을 하기 위해선 먼저 차선변경이 가능한 차선인지를 확인해야한다. 도로 준법상 흰색 점선에서만 차선 변경을 할 수 있으므로 차선을 변경하기 전에 주변 차선이 점선인지 인식하여 확인하는 코드를 구현하였다.
		
		`def conv_image(self, data):`
		`  self.cam_img = self.bridge.imgmsg_to_cv2(data, 'bgr8')`
		`  self.gray = cv2.cvtColor(self.cam_img, cv2.COLOR_BGR2GRAY)`
		
		`  self.Blur = cv2.GaussianBlur(self.gray, (5, 5), 0)`
		`  hsv = cv2.cvtColor(self.cam_img, cv2.COLOR_BGR2GRAY)`
		`  lower_white = np.array([0, 0, 70])`
		`  upper_white = np.array([131, 255, 255])`
		`  self.range = cv2.inRange(hsv, lower_white, upper_white)`
		
		먼저 cam이미지를 graysclae로 인식하여 GaussianBlur를 적용시켜 주었고, 이미지를 hsv로 변환시켜 3채널 이미지를 만들어 색, 채도, 명도의 하한값을 정하였다. inRange()메소드를 통해 흰색과 검은색 값 만을 가지는 이진화된 이미지를 만들어 self.range 변수에 저장해주었다.
		
		`r_dotarea = self.range[240: 340, 440: 640]`
		`l_dotarea = self.range[240: 340, 80: 280]`
		`if 50 < cv2.countNonZero(r_dotarea) < 300:`
		`  self.l_dot = True`
		`elif 50 < cv2.countNonZero(l_dotarea) < 300:`
		`  self.r_dot = True`
		
		`# Return positions of left and right lines detected.`
		`return self.left, self.right, self.l_dot, self.r_dot`
		
		점선 판단은 차선이 검출되는 ROI를 지정하여 점선과 실선을 비교하였을 때 점선이 상대적으로 흰색의 픽셀의 수가 적게 나오는데 이를 이용하였다. 직접 만든 주행조건에서 평균적으로 점선일 때 지정한 ROI에서 발견되는 흰색의 수를 관측하였다. 양쪽 차선의 점선 판단 이미지의 흰색 픽셀의 수가 이 관측 값과 근사하다면 차선이 점선으로 판단하여 self.l_dot과 self.r_dot을 True값으로 만들어 반환하였다.

2. 좌·우회전 차로 변경은 사전에 후방과 주위의 안전을 확인한 후, 서서히 진로 변경해야 한다. 따라서 좌회전시 왼쪽, 우회전시에는 오른쪽 각각 방향 마다 3개의 초음파 센서를 이용하여 안전을 확인하도록 구현하였다.

```
class ObstacleDetector:

    def __init__(self, topic):
        self.left = []
        self.mid = -1
        self.right = []
        self.back = -1
        rospy.Subscriber(topic, Int32MultiArray, self.read_distance)

    def read_distance(self, data):
        self.left.extend([data.data[0], data.data[7], data.data[3]])
        self.mid = data.data[1]
        self.back = data.data[4]
        self.right.extend([data.data[2], data.data[6], data.data[5]])

    def get_distance(self):
        return self.left, self.mid, self.right, self.back
```

ObstacleDetctor의 코드에서 왼쪽과 오른쪽을 보는 3개의 초음파 센서를 각각 읽어드려와 방향에 맞게 self.left와 self.right 리스트에 넣어주었다.

초음파 센서 값을 이용하기 위해 get_distance라는 getter 메소드의 반환하는 값들 중에 왼쪽과 오른쪽의 거리 정보는 리스트로 변경하여 주었고, 후방을 감지하는 센서도 추가하였다.

```
def steer(self, left, right, l_dot, r_dot, obs_l, obs_r):

    if left == -40 or right == 680:
        mid = (left + right) // 2
        angle = (mid - 320) // 1.8
    else:
        angle = 0

    if min(obs_l) > 40 and l_dot == True:
        angle = -25
    elif min(obs_r) > 40 and r_dot == True:
        angle = 25
    print("angle:", angle)
    return angle
```

Steer 함수에서 왼쪽 오른쪽 차선이 점선인지 확인하는 boolean값과 안전하게

		좌회전과 우회전을 할 수 있도록 위에서 만들어 주었던 초음파 센서 리스트를 인자값으로 받아오도록 추가하였다. If 문을 통하여 각 방향의 초음파 센서들 중 최소값이 안전하게 차선변경을 할 수 있을 최소의 거리를 충족 시키고 받아온 점선판단이 True를 충족시키는지 확인했을 경우에만 차선변경을 하도록 angle값을 설정해주었다.
		
		3. 진로 변경후에는 뒤차와의 충돌을 피하고자 후방 초음파 센서를 확인하고 소도를 뒤에서 접근하는 자동차에 맞게 변경하여야 한다. (자이카에서후방 안전 거리는 60cm로 설정)
		
		`def accelerate(self, obs_m, otherspeed, obs_back):`
		`  if obs_m < 50:`
		`  speed = 0`
		`  elif obs_back < 60:`
		`  speed = otherspeed`
		`  else:`
		`  speed = 20`
		
		`  print("speed : ", speed)`
		`  return speed`
		
		주행환경에서 다른 자동차의 속도를 일정한 값으로 정해주었다.
		
		Accelerate 함수에서 후방 충돌을 막기위해 위에서 가져왔던 후방 초음파 센서값과 주행 환경으로 정한 주변 차량들의 일정한 속도를 인자 값으로 추가하였다.
		
		이 상황에서 만약 차선변경 후 뒤의 자동차가 후방 안전거리로 정한 60cm이하일 때, speed를 미리 정한 다른자동차들과 같은 일정한 값으로 변환 시켜준다.

실행결과		1. 점선판단 결과: 주행환경을 미리 정한 명도의 하한값에 맞춘다는 전제하에 정상적으로 흰색의 픽셀값을 기준으로 점선을 판단하였다. 2. 초음파 센서를 통한 안전거리 판단: 초음파 센서값이 불안정 하긴 하지만 장애물이

	직접 가까이 있을때는 점선이어도 차선을 변경하지 않았고, 차선 변경후 뒤에서 차가 미리 정한 일정한 속도로 후방 안전거리 이하로 접근했을 때 속도를 뒤에 차에 맞게 바뀌어 평행한 속도로 주행한 것으로 보아, 각각 위치에 맞는 초음파 센서값을 제대로 받아들여 알맞게 처리하였다.