2019학년도 유레카프로젝트 조별 과제 최종 보고서

조번호	5 조				
조장/조원	성명	학과	학번	학년	비고
조 장	김신건	소프트웨어융합학과	20191564	01	
조 원	강경수	소프트웨어융합학과	20191545	01	
	고현성	소프트웨어융합학과	20191549	01	
	곽다윗	소프트웨어융합학과	20191550	01	
	김예리	소프트웨어융합학과	20191565	01	

해당 조는 국민대학교 소프트웨어학부 유레카프로젝트의 규정을 준수하여 성실하게 과제에 참여하였으며, 사실을 토대로 조별 최종 연구보고서를 작성하여 최종보고서를 제출합니다.

2019년 12월 17일

국민대학교 소프트웨어학부 유레카프로젝트

최종 보고서 목차

Ⅰ. 과제 수행 결과 보고서

2019학년도 유레카프로젝트 연구 보고서

공학적 접근 방식을 활용한 Xycar 자율 주행

국민대학교 소프트웨어융합대학
소프트웨어학부

5 조

2019 년 12월 17일

공학적 접근 방식을 활용한 Xycar 자율 주행

1. 서 론

1.1. 연구 배경

국민대학교 창업연계공학설계입문 강의와 유레카 프로젝트 강의를 통해서 아래 2가지의 목표를 달성하기 위해 이 연구를 진행하게 되었다.

첫째, 현실 제약이 따른 상황에서서의 시스템 설계를 하고 설계 -> 구현 -> 시험 -> 설계 수정의 과정을 통해 목표를 달성하면서 공학 설계에 대해 이해한다.

둘째, 임베디드 시스템 프로그래밍의 기초인 센서와 액츄에이터에 대해 이해하고 자율주행 알고리즘을 공부하면서 자율 주행 모형차 실습을 진행한다.

1.2 이론적 배경

1.2.1유레카 프로젝트 자율주행 경진대회 규칙

다음 규칙들은 유레카 프로젝트 강의에서 진행하는 <유레카 프로젝트 자율주행 경진대회>에 명시되어 있는 규칙이다.

첫째, Xycar의 트랙 완주 주행 시간은 5분으로 되어있으며, 정해진 시간 내에 완주 하지 못할 시 실격으로 처리가 된다.

둘째, 만약 트랙 완주 시간이 남는다면 그 시간동안 횟수에 상관없이 시도를 할 수 있다.

셋째, 주행 시 Xycar가 차선을 크게 이탈하게 된다면 실격 처리가 된다.

넷째, 대회 속 등수 나열은 어느 팀이 가장 빠른 Lap타임을 기록했는지를 기준으로 한다.

다섯째, Xycar 주행 시, 각 커브 구간에 노란색 포스트잇으로 붙여놓은 Check Point가 있고 이를 2번 밟으면 실격 처리가 된다.

여섯째, 각 팀에게 주어진 주행 시간은 5분으로 제한되어 있으며 주행 횟수에 제한은 없다.

일곱째, Check Point를 1번 밟으면 주행 완주 기록에 5초가 추가된다. Check Point를 2번 밟게 되면 실격 처리가 된다.

여덟째, 주행 시간대는 팀마다 다르지만 밤과 낮의 주행환경을 딱히 구분하지 않는다. 낮에 쓰던 주행환경을 밤에도 동일하게 적용한다.

2. 2장 연구의 목적, 내용, 방법

2.1. 연구의 목적

첫째, 국민대학교 7호관의 자율주행 스튜디오의 자율 주행 트랙을 성공적으로 완주한다.

둘째, 자율 주행 트랙을 완주하는 소프트웨어를 구성하면서 자율 주행 알고리즘, 차선 인식 알고리즘을 도출한다.

셋째, 창업연계공학설계입문의 실습 과제와 유레카 프로젝트의 자율 주행 경진 대회에서 도출한 알고리즘을 바탕으로 좋은 성과를 도출한다.

2.2. 연구 내용 및 방법

2.2.1. ROS

ROS(Robot operating System)은 오픈 소스 로봇 운영체제이다. ROS는 로봇 소프트웨어를 개발하는데 필요한 소프트웨어 프레임워크이다. ROS는 메타 운영체제이며 미들웨어이고 소프트웨어 모듈, 라이브러리 집합, 도구 집합으로 이루어져있다.

2.2.2. ROS 용어

2.2.2.1. 마스터(Master)

마스터는 Ros Core라고도 부르며, 마스터는 ROS의 노드들 사이의 모든 통신을 총괄 관리한다.

2.2.2.2. 노드(Node)

노드는 Ros에서 실행 가능한 최소의 단위, 프로세스 이며 Ros에서 발생하는 통신 주체이다. 하드웨어 장치와 소프트웨어 모듈에 대해서 각각 하나씩의 노드를 가지게 된다.

2.2.2.3. 토픽(Topic)

ROS 노드들이 관심을 가지고 있는 메시지 종류들을 말한다. 예를 들어, 센서 입력 데이터, 카메라 이미지, 액츄에이터 제어 명령 등이 있다.

2.2.2.4. 발행자(Publishers)

특정 토픽에 메시지를 발행 혹은 게시하는 노드를 말한다. 예를 들어, 각종 센서, 카메라, 모터 제어 알고리즘 등이 있다.

2.2.2.5. 구독자(Subscribers)

특정 토픽에 발행되는 메시지를 구독 혹은 수신하는 노드를 말한다. 예를 들어, 각종 액츄에이터 제어기, 데이터 시각화 도구 등이 있다.

2.2.2.6. 패키지(Packages)

하나 이상의 노드와 노드의 실행을 위한 정보등을 묶어 놓은 단위를 말한다. 노드, 라이브러리, 데이터, 설정 파일 등을 포함한다.

2.2.3. ROS 명령어

2.2.3.1. ROS 쉘 명령어

- roscd : 지정한 ros 패키지 디렉터리로 이동한다.
- rosls : ros 패키지 파일 목록을 확인한다.
- rosed: ros 패키지 파일을 편집한다.
- roscp : ros 패키지 파일을 복사한다.
- rospd : ros 디렉터리 인덱스에 디렉토리를 추가한다.

- rosd : ros 디렉터리 인덱스를 확인한다.

2.2.3.2. ROS 실행 명령어
- roscore : master + rosout + parameter server 를 합쳐 실행한다.
- rosrun : 패키지 노드를 실행한다.
- roslaunch : 패키지 노드를 여러 개를 실행한다.
- rosclean : ros 로그파일을 검사하거나 삭제한다.

2.2.3.3. ROS 정보 명령어
- rostopic : 토픽 정보를 확인한다.
- rosservice : 서비스 정보를 확인한다.
- rosnode : 노드 정보를 확인한다.
- rosparam : 파라미터 정보 확인, 수정한다.
- rosbag : 메시지를 기록하고 재생한다.
- rosmsg : 메시지 정보를 확인한다.
- rossrv : 서비스 정보를 확인한다.
- rosversion : 패키지 및 배포 릴리스 정보를 확인한다.

2.2.3.4. ROS catkin 명령어
- catkin_create_pkg
: catkin 빌드 시스템으로 패키지를 자동 생성한다.
- catkin_make
: catkin 빌드 시스템으로 빌드한다.
- catkin_eclipse
: 패키지를 eclipse에서 사용할 수 있게 변경한다.
- catkin_prepare_release
: changelog 정리 및 버전 태깅을 한다.
- catkin_init_workspace
: catkin 빌드 시스템의 작업 폴더를 초기화한다.
- catkin_find
: catkin 시스템 검색을 한다.

2.2.3.4. ROS package 명령어

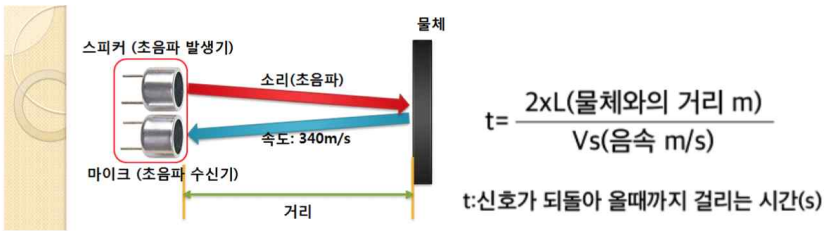
- rospack : 패키지와 관련된 정보를 본다.
- rosinstall : 추가 패키지를 설치한다.
- rosdep : 해당 패키지의 의존성 파일을 설치한다.
- roslocate : 패키지 정보 관련을 확인한다.
- rosmake : 패키지를 빌드한다. (구 시스템에서 사용되는 명령어 이다.)
- roscreate-pkg : 패키지를 자동 생성한다. (구 시스템에서 사용되는 명령어이다.)

2.2.4. 초음파 센서 제어

하드웨어의 입출력 제어를 하기위해서 폴링(polling)과 인터럽트(interrupt) 방식을 사용하는데, 초음파에는 폴링 방식으로 초음파의 값을 가져오게 된다. 폴링 방식은 프로세서 (processor) 가 입출력 장치에

게 데이터를 송신할 준비가 되었는지를 반복적으로 확인하고, 입출력 장치가 준비 되었으면 데이터를 입출력 장치에게 요청하여 프로세서가 수신받는 과정으로 진행된다.

Xycar에서 사용하는 초음파 센서의 모듈명은 HC-SR04로서 최대 4m, 최소 2cm까지 데이터를 받아올 수 있다. 초음파 센서에는 Vcc: 센서 회로의 전압 공급, GND: 회로의 그라운드에 연결, Trig: 센서를 동작시키기 위한 트리거 시그널(입력), Echo: 거리 측정 결과를 전달하기 위한 시그널 (출력)와 같은 4개의 핀이 있다. 초음파 센서의 동작과정은 첫째, 송신부에서 초음파를 발사한다. 둘째, 초음파가 물체에서 반사된다. 셋째, 반사된 초음파가 수신부에서 감지된다. 넷째, 송신과 수신 사이의 시간 간격을 기준으로 물체까지의 거리를 계산한다. 이와 같은 과정들로 이루어 진다.



초음파 센서를 이용해서 거리 측정을 하기 위해서 다음과 같은 공식이 사용된다. 초음파가 물체에 반사되어 센서에 인식되었을 때, 이 초음파의 발사 시점으로부터 수신 시점까지의 시간을 t 라고 둔다.

코드 안에서 적용을 시킬려고 한다면 다음과 같이 쓰면된다.

```
pulse_duration = pulse_end - pulse_start
distance = pulse_duration * 100 * 340.0 * 2
```

Xycar에서 초음파 센서의 값을 가지고 정차하는 프로그램을 작성할 때 초음파 센서의 튀는 값들 때문에 제대로 정차를 하지 못할 때가 있었다. 그래서 초음파 센서들의 값을 받아 필터로 가공한 데이터를 정차를 할 때 사용하기로 하였다.

먼저, 평균값 필터를 사용해 보았다. 최근 5개의 초음파 센서의 값을 리스트에 담아 값들의 평균을 정차를 할 때 사용 하는 방식이다. 필터가 없을 때 보다 정차의 정확도가 많이 올라왔지만, 4번의 1번 정도는 정차가 이상할 때가 있었다. 그래서 최근 5개의 초음파 센서의 값을 리스트에 담아 그중 2번 인덱스에 있는 값을 정차를 할 때 사용 하는 중간값 필터를 사용 해봤다. 하나의 값을 사용해서 정차를 판단해서 불확실 때가 발생하긴 했지만, 값이 안들어오는 경우(0)을 제외 하고 값을 정차에 활용하니 정확도가 많이 올라오는 것을 확인할 수 있었다.

2.2.5. 모터 제어

Xycar에서 모터를 제어하는 방식은 PWM(펄스 폭 변조: 아날로그 신호의 시간에 따른 변화를 디지털 신호를 이용하여 묘사하는 기법)를 사용한다. 속력을 제어 하기 위해서는 아두이노로부터 신호를 받은 ESC가 PWM 신호로 속력을 제어하고, 조향각을 제어하기 위해서는 아두이노가 PMW 신호를 만들어 조향 서보를 제어함으로써 조향각을 제어할 수 있다. 그리고 아두이노(하드웨어)를 제어할 때는 TX-2 보드에서 실행되는 ROS노드가 USB 시리얼 통신으로 아두이노에게 명령을 전달하여 아두이노를 제어할 수 있다.

모터를 제어하는 노드인 /xycar_B2_motor는 /xycar_motor_msg 토픽을 구독하고 있다. 모터를 제어하기 위해서는 새로운 발행노드를 만들어서 Int32MultiArray형식으로 0번째 인덱스에는 조향각을 1번째 인

덱스에는 속도를 넣고, /xycar_motor_msg 토픽에 데이터를 발행하면 Xycar의 모터를 제어 할 수 있다.

발행자가 메시지를 구독자에게 전달하면 아두이노는 모터 컨트롤러에게 전달하고 모터 컨트롤러가 PWM 신호를 만들어서 차를 움직인다.

발행노드 코드는 다음과 같다.

```
publisher = None

# 발행할 값 설정
def drive(Angle, Speed):
        global publisher
        drive_info = [Angle, Speed]
        drive_info = Int32MultiArray(data=drive_info)
        publisher.publish(drive_info)

# 발행노드 설정
rospy.init_node( “motor_pub” )
publisher = rospy.Publisher( “xycar_motor_msg” ,Int32MultiArray, queue_size=1)

# rospy가 켜져있는동안 직진
while not rospy.is_shutdown():
        drive(90, 120)
```

ROS Launch 파일 작성과 실행방법은 다음과 같다.

```
<launch>
        <node pkg=” xycar_b2”  type=” xycar_b2_motr.py”  name=” xycar_B2_motor” />
        <node pkg=” motor_pub”  type=” motor_pub.py”  name=” motor_pub” />
<launch>
```

>>> roslaunch 패키지이름 런치파일이름.launch

2.2.6. 모터 급정거 제어

2.2.5. 모터 제어를 통해 모터를 제어할 수 있었다. 하지만 Xycar의 모터제어 방식이 모터 노드에 모터의 속도를 90(정지)으로 설정하는 메시지를 발행하는 것이기 때문에, Xycar가 급정거를 하지 않고 제동 거리가 길게 나타나는 현상을 보였다. 따라서, 급정거를 하기 위해 정지와 동시에 후진을 조금씩 하는 코드를 삽입하였고 다음과 같다.

```
 for i in range(2):
self.driver.drive(90,90)
time.sleep(0.1)
self.driver.drive(90,60)
time.sleep(0.1)
self.driver.drive(90,90)
```

두 번의 반복문을 돌면서 정지와 후진을 짧은 시간 내에 반복하면서 급정거시에 앞으로 더 가는 현상을 제거할 수 있었다.

2.2.7. 초음파 센서를 이용한 주행 제어

Xycar에서 초음파 센서값을 가져오는 과정은 아두이노(하드웨어)가 초음파 센서값을 가져와 USB 시리얼 통신을 이용하여 여덟개의 초음파 센서 측정값을 /ultra_sonic_pub 노드에 보낸다. 이 노드는 /ultrasonic 토픽에 데이터를 발행한다. 새로운 코드를 작성해서 /ultrasonic 토픽을 구독하면 초음파 센서값을 이용할 수 있다.

구독 노드 코드는 다음과 같다.

```
Data = None

# 구독노드 설정
def listener():
        rospy.init_node( ‘ultrasonic_sub’ )
        rospy.Subscriber( ‘ultrasonic’ , Int32MultiArray, callback)

# 콜백함수를 설정해서 데이터가 발행될 때마다 가져오게 함.
def callback(data):
        global Data
        Data = data.data

# 데이터 사용하기
def callback_data():
        global Data
        print(str(Data))

# 구독노드 설정함수 호출
listener()
```

ROS Launch 파일 작성과 실행방법은 다음과 같다.

```
<launch>
        <node pkg=” ultrasonic”  type=” ultra_sonic_publish.py”  name=” ultra_sonic_pub” />
        <node pkg=” ultrasonic_sub”
type=” type=” ultrasonic_sub.py”  name=” ultrasonic_sub”  output=” screen” />
</launch>
```

```
>>> roslaunch 패키지 이름 런치파일.launch
```

2.2.8. opencv를 활용한 카메라 이미지 처리

USB 카메라를 이용한 비전 처리는 LineDetector 클래스의 구현을 통해 이루어졌다. 성공적인 구현을 위해 주어진 여러 문제를 풀어나가야 했다.

먼저, USB 카메라로부터 이미지를 수신받기 위해 자이트론 측에서 구현한 노드가 발행하는 메시지

토픽인 '/usb_cam/image_raw' 을 conv_image 메서드를 콜백 함수로 하여 구독하였다.

```
        self.bridge = CvBridge()
        rospy.Subscriber(topic, Image, self.conv_image)
```
<CvBridge 객체의 생생과 '/usb_cam/image_raw' 토픽의 구독>

그리고 메시지가 발행 되면 호출되는 conv_image 메서드에서 이미지 정보를 OpenCV와 NumPy가 처리할 수 있는 자료형으로 변환하기 위해 cv_bridge 모듈의 CvBridge 클래스의 객체를 생성하여 imgmsg_to_cv2 메서드를 이용해 rgb8 형식의 바이너리 데이터를 bgr8 형식의 numpy.ndarray로 변환하고 멤버변수인 cam_img에 저장하였다.

```
    def conv_image(self, data):
        self.cam_img = self.bridge.imgmsg_to_cv2(data, 'bgr8')
```
<conv_image 메서드의 구현>

그리고 본격적으로 비전 처리를 위한 과정 중 첫번째로 Bird's-eye view로의 변환을 구현하였다. 차량에 부착된 USB 카메라의 시선이 보는 차선은 원근이 적용되어 있기에 실제 차선과 비교해 심하게 왜곡되어 있고 이는 직관적인 값 검출에 어려움을 준다. 그러므로, 마치 위에서 보는 것처럼 변환을 하는 과정이 필요하다. 이 과정에는 source, destination 정점 배열을 이용해 source의 정점들을 destination으로 왜곡시키는 것이 포함된다.

```
        self.before = np.array([[0, 315], [639, 305], [170, 260], [460, 250]], dtype='float32')
        self.after = np.array([[0, 100], [100, 100], [0, -100], [100, -100]], dtype='float32')
```
<Bird's-eye view로의 변환을 위한 source, destination 정점 배열>

이를 달성하기 위해, cv2 모듈의 getPerspectiveTransform 함수로 source에서 destination으로 변환하는 행렬을 계산한 뒤 warpPerspective 함수를 통해 2차원인 픽셀의 위치가 변환된 3차원 상의 평면에 투영되어 실제 이미지의 왜곡 과정을 진행한다.

```
        tdSize = (100, 210)
        m = cv2.getPerspectiveTransform(self.before, self.after)
        topdown = cv2.warpPerspective(frame, m, tdSize)
...
        edges = cv2.warpPerspective(edges, m, tdSize)
```
<실제 Bird's-eye view로의 변환>

두번째로, 허프 변환을 통한 직선 검출을 위해 이미지 전처리 과정을 거쳤다. 카메라를 통해 받아온 이미지는 각종 노이즈가 있을 뿐만 아니라 허프 변환 알고리즘에 적합한 형태가 아니기 때문이다. 허프 변환에서 직선을 검출하는 원리는 극좌표계에서의 직선 표현에 사용되는 theta, radius을 좌표평면에 도입한 파라미터 공간을 discrete하게 만들어 두고 이미지의 각 픽셀에 대해 그 픽셀과의 교점이 있는 모든 직선의 파라미터를 구하여 파라미터 공간에서의 각 점들의 값을 늘려주고, 임계치를 두어 그 이상인 점들을 존재하는 직선으로 고려하는 것이다. 그렇기에, 우선 bgr 값을 갖는 색상을 가진 3채널 이미지를 cv2.cvtColor 함수를 통해 흑백의 1채널 그레이스케일 이미지로 변환하였다. 파라미터 공간에서의 한 점은 스칼라 값을 갖기에 bgr 형태의 3차원 벡터 값은 값 누적 결정에 모호하게 작용하기 때문이다. 그

후, 존재할 수 있는 노이즈를 opencv에서 구현되어 있는 가우시안 필터인 cv2.GaussianBlur 함수를 통해 제거하였다. 마지막으로, 정보의 강건성과 최적화를 위해 정보 희소화 과정을 거쳤다. 일반 이미지는 모든 픽셀이 밀집된 형태이므로 각각 픽셀이 값을 갖기에 외곽선만 표현된 이미지에 비해 연산 대상이 많아 성능에 부정적인 영향을 미칠 뿐만 아니라 직선은 외곽선의 포함될 가능성이 높기 때문에 외곽선만을 이용하는 것이 전체 픽셀을 이용하는 것보다 노이즈가 적어 더욱 강건하기 때문이다. 외곽선 검출을 위해 cv2.Canny 함수를 이용하여 그레이스케일 이미지에서 각 픽셀마다 이웃 픽셀과의 밝기 차이를 기준으로 이진화하였다.

```
gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
gray = cv2.GaussianBlur(gray, (3, 3), 0)
edges = cv2.Canny(gray, 50, 150, apertureSize=3)
```

<올바른 형태로의 변환 및 외곽선 검출을 통한 정보 희소화>

마지막으로, 실제로 허프 변환을 거쳐 직선들의 파라미터를 얻어냈다. OpenCV에서 제공하는 cv2.HoughLines 함수를 사용하면 입력된 이미지와 임계치 등의 인자에 따라 조건에 맞는 직선들을 반환한다. 각각의 직선마다 고유의 theta, radius를 갖기 때문에 가장 빈도가 높은 theta 값을 차선의 것이라고 간주하고 처리하였다. 또, theta가 y축을 기준으로 0 ~2PI 값을 갖기에 좀 더 직관적인 -PI/2 ~ PI/2 형태로 변환하였다.

```
lines = cv2.HoughLines(edges, 1, np.pi / 180, 80)
...
                    index = cnts.index(max(cnts))
                    if cnts[index] == 0:
                        break
                    theta = vals[index] / cnts[index]
                    if theta > math.pi / 2:
                        theta -= math.pi
                    thetas.append(theta)
```

<허프변환 및 형태 변환, 최빈값 선택 과정>

주행 알고리즘이 완성되고 나서는 여러 문제가 발견되었고 그 문제를 해결하기 위해 다양한 시도를 해보았다. 그 중, 인자를 수정하여 가장 효과적이고 큰 변화를 준 것이 두 가지가 있다. 첫번째는, 시야를 좁히는 것이었다. 초기 코드에서는 시야가 과하게 먼 나머지 너무 빨리 조향을 하여 트랙을 벗어나는 경우가 빈번히 있었다. 이 문제는 Bird's-eye view 변환에 사용되는 source, destination 정점 배열의 수정을 통해 시야를 눈에 띄게 줄여 해결하였다. 두번째 문제는 가우시안 블러의 필터 크기가 너무 컸다는 것이었다. 노이즈를 제거하기 위해 삽입한 코드였는데, 너무 강도가 심했는지 차선과 바닥의 경계가 모호해져, Canny 외곽선 검출에서 선으로 인식되지 않았기 때문에 이 또한 차선을 이탈하는 문제를 빚어냈다. 이 문제도 간단하게 가우시안 블러 함수의 인자에서 필터의 크기를 7x7에서 3x3으로 낮추어 해결하였다.

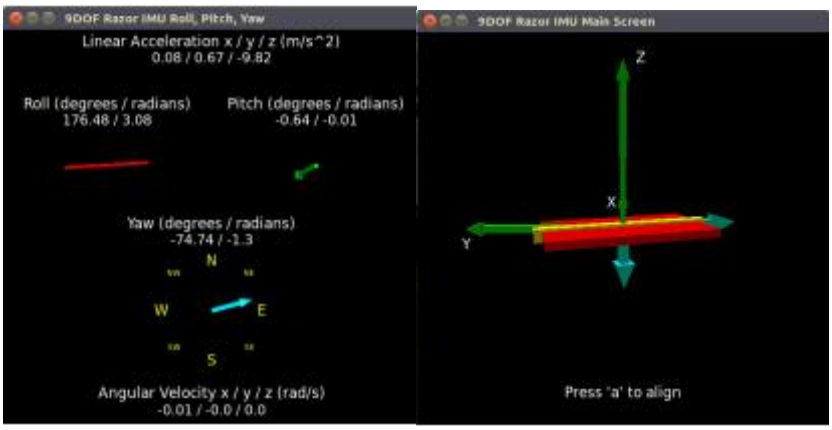
2.2.9. IMU 센서 제어

Xycar에 내장되어 있는 IMU 센서에 대해 학습하고 토론하는 시간을 가졌다. IMU(Inertial Measurement Unit)은 관성 측정 장치라고 불리며 세가지 정보를 얻어 ROS 노드로 발행한다. 세 정보는 각각 선가속도, 각속도, 절대방향이다. 선가속도 측정은 가속도계(accelerometer)를 이용하고, 각속도는 자이로스코프(gyroscope)를 통해 얻어내며, 절대방향은 자력계(magnetometer)로 알아낸다. 직교하는 세

방향의 축 각각에 대해 이 센서들을 하나씩 조합하여 구성된다.

각 축의 회전각은 Pitch, Roll, Yaw라고 불리며 Pitch는 이동 방향에 대하여 수직을 이루는 수평면 위의 축, Roll은 평행한 수평면의 축, Yaw는 이동 방향에 수직을 이루는 수직면 위의 축의 회전각을 담당한다. 이 구조는 인간을 포함한 대부분 포유동물의 내이에 존재하는 전정기관과 유사하다. 세 반고리관이 서로 직각으로 위치되어있으며 머리 움직임을 뇌에 전달하여 몸의 균형을 담당한다.

IMU 센서에 대한 학습을 위해 razor_imu_9dof 패키지의 razor-pub-and-display.launch를 통해 시각화된 IMU 센서의 정보를 관찰하였다.



<시각화된 IMU 센서의 정보들>

시각화를 통해 IMU 센서의 작동방식과 정보의 형태를 직관적으로 파악할 수 있었으며 IMU 센서의 값을 이용한 코드 구성에 도움을 될 것이라고 판단했다.

조원끼리의 토론을 통해 IMU 센서를 활용한 트랙 주행 보정 등의 아이디어가 도출되었지만 실제로 이용되지 않았다.

2.2.10. 카메라와 모터를 활용한 자율 주행1

자율주행 과제에서 모터의 조향과 가속은 AutoDrive 클래스에서 LineDetector가 인식한 차선 데이터로부터 이루어졌으며 여러 과정을 통해 그 데이터를 가공하여 모터 노드에 메시지를 발행하여 달성하였다.

가장 처음으로 시도한 방식은 얻은 차선의 theta 값에 따라 일정한 비율로 조향, 감속을 하는 것이었다. 조향을 담당하는 메서드인 steer와 가속을 담당하는 메서드인 accelerate에서 각각 theta에 대한 비율인 지역변수 K를 선언하고 값을 임의로 넣어 구성하였다. 간단한 방식이었기에 코드 작성 후 바로 시험주행을 할 수 있었고, 결과는 예상보다 뛰어났다. 간헐적으로 트랙을 벗어나거나 차선을 밟는 일이 있었지만 간단한 수정으로 개선할 수 있어 보였다.

그 후에는 theta의 값을 단계 별로 나눠 K 값을 정하는 방식으로 바꾸었다. theta의 절댓값이 작을 때는 너무 조금 조향하고 값이 너무 클 때는 과하게 조향하였기 때문이다. 그런 문제를 최소화하고자 theta가 작을 때는 K를 크게 설정하고 값이 클 때는 K를 작게 설정하는 식으로 수정하였다. 또, 가속의 정도에 따른 안정성은 theta의 값과 큰 관계가 없다는 것을 관찰하고 항상 같은 속도로 주행하도록 하였다.

그렇게 수정한 뒤에는 좀 더 안정적으로 주행하게 되었지만 트랙의 중심으로 주행하지 않고 한 방향을 쏠려 주행하는 문제가 남아있었기에 여러 해결방안을 생각해보았고 시험하였다. 첫번째로 시도한 방법은 LineDetector에서 왼쪽, 오른쪽 차선을 각각 구해 두 차선의 중앙에서 주행하게 조향하는 것이였다. 이 방법은 기존의 theta를 이용한 조향과 잘 맞지 않았고 각 방식이 서로 반대의 방향으로 조향하게

명령하여 값이 상쇄되는 문제가 있었다. 이 문제를 해결하기 위해 직진 상황에서만 중앙으로 주행하게 바꿔보았으나 잘 되지 않았다. 결국 이 방법을 사용하지 않기로 결정하였다. 대신, 직진 상태에서 그렇듯 매우 작은 theta 값이 입력되는 경우에는 K의 값을 매우 크게 주어 쏠림을 보정하고, 기본적으로 조향 각도와 theta 값 자체에 오프셋을 주어 해결하였다.

이렇게 Training 모드에서의 주행이 완성되었으나 Race 모드는 Training 모드의 두 배의 속도를 낼 수 있었기에 더 빠른 주행을 위해서는 Race 모드에서 동작하는 코드로 기존 코드를 바꿔주어야 했다. Race 모드는 Training 모드에 비해 매우 불안정했으며 전반적인 K의 값을 더 크게 해줘야 했고, 조향 시의 안정성이 더욱 떨어졌기 때문에 조향 시작을 기준으로 3초간 속도를 조금 낮추는 식으로 안정성을 확보했다. 그리고 LineDetector에서의 이미지 가공 과정의 수정 또한 안정성에 큰 도움이 되었다.

이러한 값 수정과 로직 추가로 Race 모드에서도 비교적 안정적인 주행을 할 수 있게 되었으며 더이상의 큰 문제는 발견되지 않았다.

2.2.11. 카메라, 초음파 센서, 모터를 활용한 자율주행

2.2.11.을 통해 카메라로 차선을 인지하고 Xycar의 모터를 활용해 자율 주행을 할 수 있었다. 창업연계공학설계입문 수업의 두번째 주행과제를 수행하기 위해 추가적으로 자율 주행 스튜디오 트랙 1 lap을 돌고 난 뒤, 초음파 센서를 활용해 정지하는 기능을 추가하게 되었다.

첫번째로 초음파 센서의 센서값 그대로를 받아들이고 특정 거리 이하로 인지되면 멈추는 코드를 작성하였다. 하지만, 주행 중에 초음파 센서를 사용하게 되면서 이전에 쏜 초음파 때문에 생긴 간섭과 자율주행 스튜디오의 구조상 초음파 값이 특정 거리 이하로 들어오게 되었다. 따라서 첫번째 방법은 실패하였다.

두번째로, 초음파 센서의 센서값을 큐에 3개 혹은 5개씩 쌓아두고 평균값 필터를 사용하여 초음파 센서의 값을 다루게 되었다. 하지만 첫번째와 마찬가지로 자율 주행 스튜디오 구조상 코너 부분에서 벽을 장애물로 인지하고 멈추는 문제가 발생하였다. 따라서 두번째 방법또한 실패하였다.

세번째로, 초음파 센서의 앞부분 센서값들의 삼각비를 이용해보았다. 정면을 바라보는 센서값은 그대로 활용하고, 옆에서 대각선 방향을 바라보고 있는 초음파 센서값에 (정면을 바라보는 센서 방향과 대각선 방향을 바라보고 있는 센서의 방향의 사이각을 theta라 하자) cos (theta)를 곱하면 정면을 바라보는 센서값에 수렴하게 된다. 따라서 이렇게 나온 3개의 값을 활용해 필터를 적용하였다. 하지만 이 방법 또한 초음파 센서가 3개중 2개 이상 노이즈 값이 들어오면 오류를 만들었고 실패하였다.

네번째로, 세번째 방법을 적용한 상태로, 평소에는 초음파 센서값을 인지하지 않고 있다가 Xycar가 우회전을 크게 한 시점부터 초음파 센서값을 인지하는 것이다. 이는 자율 주행 스튜디오의 구조를 활용한 것이고, 마지막 피니쉬 라인까지 도달하기 이전에 코너 구간이 존재하기 때문에 활용할 수 있는 방법이었고 초음파 센서의 값 자체를 받는 구간이 좁아지면서 오류가 덜 발생하였고 성공하였다.

2.2.12 조도에 따른 Xycar의 해상도 조절

소프트웨어를 이용한 공학적 설계는 많은 경우 현실 세계의 제약에 맞닥뜨리는 경우가 많다. 소프트웨어 자체에서의 알고리즘 수정이나 구현도 중요하지만, 때로는 물리적, 기구적인 해결책이 소프트웨어의 해결책보다 우선할 수도 있다. 조금이나마 이런 제약을 줄이기 위한 방법 중 하나로 카메라의 조정과 설정을 변경해보았다. 카메라를 설정하는 기구적 설정, 광학적 설정, 디지털 이미지 설정 중, 우리는 광학적 설정으로 주변 밝기에 따라 서로 다른 노출값 적용을 필요로 하는 노출도 조정을 중점으로 차선 추종 알고리즘을 정교화 하는 방법을 적용해 보았다. 유레카 프로젝트 자율주행 경진대회를 위해 대회

전날에 연습 주행을 하면서 차가 햇빛이 많이 들어오는 부분에서는 차선을 오인해 선을 인식하지 못하고 S자 구간에서 이탈하는 일이 발생했기 때문에 노출도 조정을 통해 Xycar 카메라 영상으로 주행을 할 때, 이전에 발생했던 차선을 인식하고 이에 따라 차가 주행하도록 하는 과정에서 차선 인식 중 필요하지 않은 값들이 들어와 차선 인식에 혼란을 주어 주행이 올바르게 되지 않았던 부분이 조정을 통해 필요하지 않은 값들이 들어오지 않게 되어 주행을 성공적으로 마칠 수 있었다. 아래의 코드는 노출도 조정을 마친 뒤의 코드이다.

```
<launch>
    <node name="usb_cam" pkg="usb_cam" type="usb_cam_node" output="screen">
        <param name="video_device" value="/dev/v4l/by-id/usb-HD_Camera_Manufacturer_USB_2.0_Camera-video-index0" />
        <param name="autoexposure" value="false"/>
        <param name="exposure" value="100"/>
        <param name="image_width" value="640" />
        <param name="image_height" value="480" />
        <param name="pixel_format" value="yuyv" />
        <param name="camera_frame_id" value="usb_cam" />
        <param name="io_method" value="mmap"/>
    </node>
    <node name="ultrasonic" pkg="ultrasonic" type="ultra_sonic_publish.py" />
    <node name="motor_control" pkg="xycar_b2" type="xycar_b2_motor.py" />
    <node name="autodrive" pkg="auto_drive" type="autodrive.py" output="screen" />
</launch>
```

노출도의 조정 방법으로는, Xycar는 내재되어 있는 autodrive.launch 파일에서 usb_cam 노드안의 param들중, autoexposure과 exposure의 값을 변경하여 노출을 조정할 수 있다. 따라서, autoexposure은 사용하지 않도록 위의 코드에서 false로 하여 사용하지 않도록 설정해주고 exposure의 값은 기본으로 설정되어 있는 150보다 작게 설정하면 노출값이 작아져 조도의 영향을 덜 받기 때문에 값을 100으로 줄여 빛을 차선을 오인하는 일이 없도록 하였다. 이렇게 주변 조도 환경에 따른 알맞은 노출을 설정함으로써 윤곽선 추출 및 이진화를 통한 차선 인식의 정밀도를 높일 수 있었다. 이를 적용하고 난 후 Xycar를 주행시켰을 때 그 전의 차선 이탈을 보완 할 수 있었고 우리는 Xycar을 통해 완벽한 자율주행을 하기 위해 카메라의 조정과 설정이 필요한 것을 느꼈다.

2.2.13. Xycar의 주행 모드

Xycar의 주행 모드는 Sport mode, Race mode, Training 모드로 3개가 있다. Xycar의 주행 모드는 XL-5 ESC(Electronic Speed Control)에 의해 제어되며 EZ-SET 버튼의 누르는 시간에 따라 변경된다. 처음 EZ-SET 버튼을 누른 상태로 기다리면 전원이 인가되고 캘리브레이션을 지나 순서대로 빨간 불이 1번, 2번, 3번 깜빡인다.

Sport mode는 빨간 불이 1번 깜빡이는 상태를 나타내고 Xycar가 Sport mode 또는 Race mode로 설정되면 Training 모드의 2배 속도로 설정되고 주행된다.

3. 3장 연구의 결과

3.1. 주행 테스트 결과

창업연계공학설계입문의 주행 과제1은 Xycar의 앞뒤로 장애물이 위치하고 있을 때, 장애물과의 일정 거리 내에 정차하면서 앞뒤로 주행하는 것이었다. 주행 과제 1에서 첫번째 Lap에서 바로 성공하였다.

창업연계공학설계입문의 주행과제2는 국민대학교 7호관의 자율주행스튜디오에 있는 트랙을 한바퀴 완주하고, 완주한 뒤에 출발점으로부터 일정 거리에 위치한 장애물 앞에서 정차하는 것이다. 주행 과제2에서도 첫번째 Lap에서 바로 성공하였다.

유레카 프로젝트 자율주행 경진대회에서 2위를 수상하였으며, Lab Time은 30초대를 기록하여 기록상 1위를 달성하였으나 곡선구간에서 이탈 판정을 받아 5초 패널티를 받게 되었다.

4. 결 론

4.1. 제언

유레카 프로젝트 자율주행 경진대회에 출전하였던 경험을 바탕으로, 해당 대회에 대해 몇 가지 개선사항을 제안한다.

먼저, 유레카 프로젝트 자율주행 경진대회의 개최 시기가 부적절하였다. 올해 유레카 프로젝트 자율주행 경진대회는 13주차에 개최되었다. 그러나, 자율주행 경진대회보다 단순한 알고리즘을 요구하는 창업연계공학설계입문 교과목의 주행과제는 14주차로 일정이 맞춰져 있다. 즉, 차선 추종 주행 알고리즘에 대하여 모든 지식을 습득하지 못한 채, 대회용 프로그래밍 코드를 작성해야 했었다. 이에 대해 개선사항으로, 16주차에 대회 일정을 잡는 것이 여러모로 합당하다고 본다. 이에 대한 이득적인 측면으로는 먼저, 본 교과목에서 추구하고자 했던 공학적 설계방식에 부합하는 교육과정 커리큘럼을 가질 수 있다. 창업연계공학설계입문 교과목의 2차 주행과제는 차선 추종 주행을 하되, 완주만을 목적으로 삼았던 주행과제 코드를 자율주행 경진대회에서 더 좋은 기록을 내기 위해 가다듬고, 효율적으로 개선해 나가는 것이 본 교과목에서 추구한 공학적 접근에 가까운 순번이다. 그 다음으로, 학생들이 더 나은 결과를 얻어갈 수 있다. 유레카 프로젝트 자율주행 경진대회에서 완주에 실패한 조는 생각 이상으로 많았다. 그러나, 이에 비해 창업연계공학설계입문 교과목의 주행과제에서 주행에 실패한 조는 경진대회보다 적었다. 이는 당연한 수순으로, 대회 준비기간이 극단적으로 짧았기에 이러한 현상이 발생한 것이다. 창업연계공학설계입문 교과목 수강생들이 차선 추종 주행에 대해 본격적으로 다루기 시작하는 지점은 11주차로 본 대회의 일정과 2주라는 짧은 시간밖에 차이 나지 않는다. 따라서 학생들이 차선 추종 주행에 대해 완벽하게 배운 상태가 아닌 미흡한 상태에서 대회에 출전하였기에, 상대적으로 자율주행 경진대회에서 주행에 실패한 조가 많을 수 밖에 없었다. 그러나, 이를 수업보충주간인 16주차로 일정을 바꾸게 된다면, 보다 제대로 된 성과를 이루는 조들이 많아질 수 있다.

또한, 대회 진행 방식이 공식적으로 이루어지지 않았다는 점 역시 개선이 필요하다. 학생들에게 공식적으로 전달된 사항은 대회일정뿐이었으며, 자세한 대회 규정 및 공지는 모두 구두로 이루어져 대회의 추가적인 질문사항은 물론, 대회의 기본적인 조건 모두 확인이 힘들었다. 따라서, 기본적인 대회 규정같은 경우, 가상대학으로 수강생 전원에게 공지하는 것이 더 바람직할 것이다. 그 다음으로, 시상식 이전까지 출전한 학생들이 본인들의 대회 성적을 확인할 수 없었다는 점 역시 문제사항이다. 물론, 시상식에

서 순위권 팀들을 발표하는 것이 더 박진감 있고 기대감을 증폭시킬 수 있을지 모르지만 반대로, 대회 성적에 감점이 있는 팀의 경우 시상식 이전까지 그러한 사항을 알 수 없다. 이는 감점 요소에 대한 이의제기가 불가능하거나, 항소할 수 없다는 문제점으로 이어지게 된다. 따라서, 대회에 출전한 학생들에게 대회 성적을 순위를 제외하고 1차적으로 개별통지가 이루어진 후 시상식을 치르는 것이 올바른 대회 진행 방식이라 할 수 있다.

4.2. 결론

공학적 접근 방식을 통하여, 자율주행 구동체인 Xycar를 자율주행하는 것이 우리의 유레카 프로젝트의 최대 목표였다. 한학기 동안, 공학설계와 ROS 기초부터, 모형자동차의 제어를 위한 소프트웨어, 자율주행 알고리즘까지, 자율주행 소프트웨어의 기본적인 내용을 연구하였다. Xycar가 충돌 및 차선 이탈 없이 자율주행하기 위해서는, 초음파 센서를 제어하여, 장애물과의 거리를 측정해

장애물과 충돌을 방지하고, 카메라로 받아 들인 영상을 HSV 컬러 코드로 변환한 후, 이진화 작업을 거친 뒤에 차선을 검출하여 차선에 따라 주행할 수 있다. 허나, 실제 교통상황에서 자율주행 자동차는 장애물 탐지와 차선 추종 주행이 가능한 것만으로, 완벽한 자율주행 알고리즘이라 할 수 없다. 자율주행 자동차는 주행 중 신호등이 있다면 신호등의 교통신호를 인식해야 하며, 교통 표지판이 있다면, 해당 표지판을 인식해 표지판의 내용을 준수해야 할 것이다. 즉, 자율주행 자동차가 실제 도로에서 주행하기 위해서는 보다 더 고도화된 알고리즘과 다양한 실제 교통상황에 대처할 수 있는 알고리즘을 필요로 할 것이다.

참고문헌

- 창업연계공학설계입문 강의 PPT
https://xycar.cs.kookmin.ac.kr/

- Xycar 공식 사이트
http://xytron.co.kr/

부록

- 소스코드
연구에 사용된 코드는 모두 github의 repository에 업로드되어있으며 링크는
https://github.com/shinkeonkim/EurekaProject-Xycar-5 이다.

II. 주차별 활동 보고서 (6 주차)

<table>
<tr><td>팀명</td><td colspan="5">5조</td></tr>
<tr><td>과제참여
인원수</td><td colspan="2">5</td><td>회의참여
인원수</td><td colspan="2">5</td></tr>
<tr><td>연구 주제</td><td colspan="5">ROS의 python을 활용한 모터, 초음파 제어</td></tr>
<tr><td>활동일시</td><td colspan="2">6주차</td><td>장 소</td><td colspan="2">자율 주행 스튜디오</td></tr>
<tr><td>활동 내용</td><td colspan="5">01. 초음파 센서 제어
02. 모터 제어
03. PWM</td></tr>
<tr><td rowspan="7">참여자 명단</td><td>연번</td><td>소속(전공)</td><td>이름</td><td>서명</td><td>비고</td></tr>
<tr><td>1</td><td>소프트웨어융합학과</td><td>김신건</td><td>김신건</td><td></td></tr>
<tr><td>2</td><td>소프트웨어융합학과</td><td>강경수</td><td>강경수</td><td></td></tr>
<tr><td>3</td><td>소프트웨어융합학과</td><td>고현성</td><td>고현성</td><td></td></tr>
<tr><td>4</td><td>소프트웨어융합학과</td><td>곽다윗</td><td>곽다윗</td><td></td></tr>
<tr><td>5</td><td>소프트웨어융합학과</td><td>김예리</td><td>김예리</td><td></td></tr>
<tr><td>6</td><td></td><td></td><td></td><td></td></tr>
<tr><td>활동내용 및
회의
논의내용
기재</td><td colspan="5">XyCar의 초음파 센서와 모터를 제어하기 위하여, 토픽을 구독/발행하는 노드 학습

해당 주차는 1차 주행 과제가 주어진 기간이었다. 주행 과제는 다음과 같다. XyCar의 초음파 센서를 이용하여, 직진과 후진을 3번씩 반복하면서, 장애물을 인식하고 일정 거리에 정차해야 한다. 또한, 3번의 직진 주행마다, 차량의 속도를 천천히, 보통, 빠르게로, 각기 다르게 조절해야 한다.

XL-5 ESC: 주행과제 외에도 차량의 주행 모드를 설정하는 방법에 대해 학습하였다. XL-5 ESC는 모든 동작이 EZ-SET 버튼을 통해 이루어진다. 전원을 켠 후, EZ-SET 버튼을 누르는 시간에 따라, Xycar의 주행 모드가 변경된다. 주행 모드에는 스포츠 모드, 레이싱 모드, 트레이닝 모드로 이</td></tr>
</table>

	루어져 있다. 카메라: 2차 주행 평가와 유레카 프로젝트 자율 주행 경진 대회에 중요하게 다뤄질 카메라 이미지를 발행하는 토픽을 구독하고, 이에 대한 데이터를 화면에 나타내는 간단한 실습을 진행하였다. 과제: 해당 주차까지 학습한 내용을 바탕으로, 초음파 센서를 통해 장애물과 거리를 인식하고, 장애물로부터 정해진 지점에 정차해야 한다. - 논제: 1) Xycar의 초음파 센서의 경우, 오작동으로 인해, 흔히 '튀는 값'이라고 불리는 잘못된 값을 받아 올 가능성이 있다. 이에 대한 마련책으로, 필터를 적용하여 튀는 값을 최대한 걸러내어, 정상적인 값들을 인식하게끔 필터를 주행 과제 프로그래밍 코드에 적용하자는 의견이 나왔다. 2) XL-5 ESC의 주행 모드의 차이점에 대해, 논의가 오고 갔다. 추후, 이는 유레카 프로젝트 자율 주행 경진대회에서 빠른 기록을 내기 위해, 주행 모드 중 레이싱 모드가 이용되었고 트레이닝 모드의 코드를 수정해야 했다.

II. 주차별 활동 보고서 (7 주차)

<table>
<tr><td>팀명</td><td colspan="5">5조</td></tr>
<tr><td>과제참여
인원수</td><td colspan="2">5</td><td>회의참여
인원수</td><td colspan="2">5</td></tr>
<tr><td>연구 주제</td><td colspan="5">초음파 센서를 이용해 장애물 인지</td></tr>
<tr><td>활동일시</td><td colspan="2">7주차</td><td>장 소</td><td colspan="2">자율 주행 스튜디오</td></tr>
<tr><td>활동 내용</td><td colspan="5">01. 초음파 센서의 제어와 필터
02. ESC의 모드 변경</td></tr>
<tr><td>참여자 명단</td><td colspan="5">
<table>
<tr><th>연번</th><th>소속(전공)</th><th>이름</th><th>서명</th><th>비고</th></tr>
<tr><td>1</td><td>소프트웨어융합학과</td><td>김신건</td><td>김신건</td><td></td></tr>
<tr><td>2</td><td>소프트웨어융합학과</td><td>강경수</td><td>강경수</td><td></td></tr>
<tr><td>3</td><td>소프트웨어융합학과</td><td>고현성</td><td>고현성</td><td></td></tr>
<tr><td>4</td><td>소프트웨어융합학과</td><td>곽다윗</td><td>곽다윗</td><td></td></tr>
<tr><td>5</td><td>소프트웨어융합학과</td><td>김예리</td><td>김예리</td><td></td></tr>
<tr><td>6</td><td></td><td></td><td></td><td></td></tr>
</table>
</td></tr>
<tr><td>활동내용 및
회의
논의내용
기재</td><td colspan="5">주행모드와 1차 주행 과제 수행 (장애물 탐지 전후진)
해당 주차는 1차 주행 과제가 주어진 기간이었다. 주행 과제는 다음과 같다. XyCar의 초음파 센서를 이용하여, 직진과 후진을 3번씩 반복하면서, 장애물을 인식하고 일정 거리에 정차해야 한다. 또한, 3번의 직진 주행마다, 차량의 속도를 천천히, 보통, 빠르게로, 각기 다르게 조절해야 한다.

- XL-5 ESC: 주행과제 외에도 차량의 주행 모드를 설정하는 방법에 대해 학습하였다. XL-5 ESC는 모든 동작이 EZ-SET 버튼을 통해 이루어진다. 전원을 켠 후, EZ-SET 버튼을 누르는 시간에 따라, Xycar의 주행 모드가 변경된다. 주행 모드에는 스포츠 모드, 레이싱 모드, 트레이닝 모드로 이루어져 있다.</td></tr>
</table>

	- 카메라: 2차 주행 평가와 유레카 프로젝트 자율 주행 경진 대회에 중요하게 다뤄질 카메라 이미지를 발행하는 토픽을 구독하고, 이에 대한 데이터를 화면에 나타내는 간단한 실습을 진행하였다. - 과제: 해당 주차까지 학습한 내용을 바탕으로, 초음파 센서를 통해 장애물과 거리를 인식하고, 장애물로부터 정해진 지점에 정차해야 한다. - 논제: 1) Xycar의 초음파 센서의 경우, 오작동으로 인해, 흔히 '튀는 값'이라고 불리는 잘못된 값을 받아 올 가능성이 있다. 이에 대한 마련책으로, 필터를 적용하여 튀는 값을 최대한 걸러내어, 정상적인 값들을 인식하게끔 필터를 주행 과제 프로그래밍 코드에 적용하자는 의견이 나왔다. 2) XL-5 ESC의 주행 모드의 차이점에 대해, 논의가 오고 갔다. 추후, 이는 유레카 프로젝트 자율 주행 경진대회에서 빠른 기록을 내기 위해, 주행 모드 중 레이싱 모드가 이용되었고 트레이닝 모드의 코드를 수정해야 했다.

II. 주차별 활동 보고서 (8 주차)

<table>
<tr><td>팀명</td><td colspan="5">5조</td></tr>
<tr><td>과제참여
인원수</td><td colspan="2">5</td><td>회의참여
인원수</td><td colspan="2">5</td></tr>
<tr><td>연구 주제</td><td colspan="5">ros 개념</td></tr>
<tr><td>활동일시</td><td colspan="2">8주차</td><td>장 소</td><td colspan="2">자율 주행 스튜디오</td></tr>
<tr><td>활동 내용</td><td colspan="5">01. ros 명령어
02. catkin directory</td></tr>
<tr><td rowspan="7">참여자 명단</td><td>연번</td><td>소속(전공)</td><td>이름</td><td>서명</td><td>비고</td></tr>
<tr><td>1</td><td>소프트웨어융합학과</td><td>김신건</td><td>김신건</td><td></td></tr>
<tr><td>2</td><td>소프트웨어융합학과</td><td>강경수</td><td>강경수</td><td></td></tr>
<tr><td>3</td><td>소프트웨어융합학과</td><td>고현성</td><td>고현성</td><td></td></tr>
<tr><td>4</td><td>소프트웨어융합학과</td><td>곽다윗</td><td>곽다윗</td><td></td></tr>
<tr><td>5</td><td>소프트웨어융합학과</td><td>김예리</td><td>김예리</td><td></td></tr>
<tr><td>6</td><td></td><td></td><td></td><td></td></tr>
<tr><td>활동내용 및
회의
논의내용
기재</td><td colspan="5">ROS 패키지와 같은 자율주행 소프트웨어 개념화
2학기의 절반을 보내면서, 지금까지 학습한 자율주행 소프트웨어 개념을 단단히 잡기 위해, 초음파 센서, 모터, 카메라를 이용한 토픽 구독/발행 노드를 다시 재작성해보며, 학기 초 다소 낯설고 어려웠던 개념으로 다가왔던 ROS 패키지 명령어와 catkin workspace에 대해 다시 개념을 정립하는 시간을 가졌다.
- ROS: 한 조원의 경험 중, 라인 트레이싱 로봇, 아두이노 등의 명령어를 다뤘던 기억을 토대로 ROS 패키지를 학습하였다.

- 공학설계: 공학설계를 과학적 접근 방식과 비교하며 학습한 결과, 공학설계에 대한 이해 효과가 매우 높았다.</td></tr>
</table>

II. 주차별 활동 보고서 (9 주차)

<table>
<tr><td>팀명</td><td colspan="5">5조</td></tr>
<tr><td>과제참여
인원수</td><td colspan="2">5</td><td>회의참여
인원수</td><td colspan="2">5</td></tr>
<tr><td>연구 주제</td><td colspan="5">초음파 센서를 이용해 장애물 인지, 주행 제어</td></tr>
<tr><td>활동일시</td><td colspan="2">9주차</td><td>장 소</td><td colspan="2">자율 주행 스튜디오</td></tr>
<tr><td>활동 내용</td><td colspan="5">01. Xycar 주행 제어
02. 초음파 센서값 필터 적용
03. Xycar 급정거 제어</td></tr>
<tr><td rowspan="7">참여자 명단</td><td>연번</td><td>소속(전공)</td><td>이름</td><td>서명</td><td>비고</td></tr>
<tr><td>1</td><td>소프트웨어융합학과</td><td>김신건</td><td>김신건</td><td></td></tr>
<tr><td>2</td><td>소프트웨어융합학과</td><td>강경수</td><td>강경수</td><td></td></tr>
<tr><td>3</td><td>소프트웨어융합학과</td><td>고현성</td><td>고현성</td><td></td></tr>
<tr><td>4</td><td>소프트웨어융합학과</td><td>곽다윗</td><td>곽다윗</td><td></td></tr>
<tr><td>5</td><td>소프트웨어융합학과</td><td>김예리</td><td>김예리</td><td></td></tr>
<tr><td>6</td><td></td><td></td><td></td><td></td></tr>
<tr><td>활동내용 및
회의
논의내용
기재</td><td colspan="5">초음파 센서에 필터 적용 & 1차 주행 과제 평가 - 장애물 탐지 전후진
해당 주차는 1차 주행 과제를 성공적으로 해내고자, 조원들의 많은 시간과 노력이 소모되었다. Xycar의 전후진 중 장애물 탐지 과정 속에 초음파 센서의 ‘튀는 값’을 미연에 방지하고자, 필터를 도입하기로 결정하였다.
- 필터: 도입할 필터는 2가지 종류가 지목되었다. 첫 번째는 평균값 필터로, 초음파 센서로 인식한 장애물 거리를 담아, 평균을 산출하는 방식으로, 만약 평균과 차이가 심한 값이 들어온다면, 해당 값은 배제하는 필터이다. 이와 달리 중간값 필터는 말 그대로, 중간값을 산출하는 방식으로, 중간값에 비해 비정상적인 값이 들어온다면, 해당 값을 배제한다.

- 정확도: 주행 과제 중 전진 시, 가장 빠른 속도로 전진할 경우, 정차하</td></tr>
</table>

	는 위치를 조정하기 힘들다는 문제가 있었다. 장애물로부터 정확한 거리에서 정차하고자, 두가지 방안을 마련하였다. 첫 번째 방법으로는 장애물은 인식하고 차량을 정차시킬 때, 차량이 관성으로 인해 미끄러지는 문제를 막고자, 주행방향으로부터 역방향으로 주행을 걸어주었다. 또 다른 방법은 초음파 센서의 값을 순환시키는 주기를 빠르게 조정하는 방안이다. 즉, 좀 더 많은 장애물과의 거리 값을 빠르게 받아들여, 정확한 위치에 정차하는 방안이다. - 논제 1) Xycar의 메인 보드 배터리의 잔량과 모터 배터리의 잔량이 Xycar 차량에 미치는 영향 - (해당 주차 보고서의 전문가와의 면담 내용)

II. 주차별 활동 보고서 (10 주차)

<table>
<tr><td>팀명</td><td colspan="5">5조</td></tr>
<tr><td>과제참여
인원수</td><td colspan="2">5</td><td>회의참여
인원수</td><td colspan="2">5</td></tr>
<tr><td>연구 주제</td><td colspan="5">공학자로서 지켜야 할 윤리 규범</td></tr>
<tr><td>활동일시</td><td colspan="2">10주차</td><td>장 소</td><td colspan="2">자율 주행 스튜디오</td></tr>
<tr><td>활동 내용</td><td colspan="5">01. 공학자로서 지켜야 할 윤리 규범</td></tr>
<tr><td rowspan="7">참여자 명단</td><td>연번</td><td>소속(전공)</td><td>이름</td><td>서명</td><td>비고</td></tr>
<tr><td>1</td><td>소프트웨어융합학과</td><td>김신건</td><td>김신건</td><td></td></tr>
<tr><td>2</td><td>소프트웨어융합학과</td><td>강경수</td><td>강경수</td><td></td></tr>
<tr><td>3</td><td>소프트웨어융합학과</td><td>고현성</td><td>고현성</td><td></td></tr>
<tr><td>4</td><td>소프트웨어융합학과</td><td>곽다윗</td><td>곽다윗</td><td></td></tr>
<tr><td>5</td><td>소프트웨어융합학과</td><td>김예리</td><td>김예리</td><td></td></tr>
<tr><td>6</td><td></td><td></td><td></td><td></td></tr>
<tr><td>활동내용 및 회의 논의내용 기재</td><td colspan="5">공학적 접근 방식과 공학 윤리와 규범, 공학자로서의 직업 정신
해당 주차는 자율 주행 구동체를 제어하기 위한 활동보다, 공학자로서 갖춰야 할 도덕적 윤리에 대하여 논의하였다.

- 공학 윤리: 공학 윤리는 일반 윤리 개념에서 확장된 개념으로, 공학자로서 마땅히 져야 할 도덕적 책임을 일컫는다. 공학자는 본인의 역할에 충실히 임해야 하며, 직업을 통해 윤리적 기준을 지키려고 노력함으로써, 공공의 선을 달성하여야 한다. 이러한 도덕적 가치는 우수한 공학 프로젝트를 위한 기준의 기본 조건이다. 허나, 대부분의 공학 활동은 기업을 통해 이루어지기에, 공학자는 기업의 제약에 의해 발생하는 윤리적 딜레마를 겪게 된다.

- 윤리 규범: 윤리 규범은 공학자들이 대중의 건강과 안전 그리고 복지</td></tr>
</table>

	를 위해 봉사해야 한다는 서약의 역할을 수행하며, 공학자가 좀 더 효율적으로 대중에게 봉사할 수 있게 하고, 스스로를 통제할 수 있게 하는 신뢰의 기능을 수행함과 동시에, 직업적 의무를 다 한 공학자에게 발생할 수 있는 비난으로부터 보호하는 후원의 기능 역시 수행한다. 허나 규범이 완벽하거나, 궁극적일 수는 없기에, 윤리 규범의 남용과 제약은 오히려 역효과를 가져올 수 있으며, 서로 다른 항목들이 충돌하였을 때 모호함이 발생하며, 규범간에 우선순위에 대한 지침이 없다. - 논제 1) 서로 다른 윤리 규범이 충돌하는 상황이 발생하였을 때, 공학자는 어떤 도덕적 신념과 규범에 따라야 하는가

II. 주차별 활동 보고서 (11 주차)

<table>
<tr><td>팀명</td><td colspan="5">5조</td></tr>
<tr><td>과제참여
인원수</td><td colspan="2">5</td><td>회의참여
인원수</td><td colspan="2">5</td></tr>
<tr><td>연구 주제</td><td colspan="5">HSV 이미지 처리, OpenCV 차선 검출</td></tr>
<tr><td>활동일시</td><td colspan="2">11주차</td><td>장 소</td><td colspan="2">자율 주행 스튜디오</td></tr>
<tr><td>활동 내용</td><td colspan="5">01.HSV를 이용한 이미지 처리
02.OpenCV를 이용하여 차선검출</td></tr>
<tr><td rowspan="8">참여자 명단</td><td>연번</td><td>소속(전공)</td><td>이름</td><td>서명</td><td>비고</td></tr>
<tr><td>1</td><td>소프트웨어융합학과</td><td>김신건</td><td>김신건</td><td></td></tr>
<tr><td>2</td><td>소프트웨어융합학과</td><td>강경수</td><td>강경수</td><td></td></tr>
<tr><td>3</td><td>소프트웨어융합학과</td><td>고현성</td><td>고현성</td><td></td></tr>
<tr><td>4</td><td>소프트웨어융합학과</td><td>곽다윗</td><td>곽다윗</td><td></td></tr>
<tr><td>5</td><td>소프트웨어융합학과</td><td>김예리</td><td>김예리</td><td></td></tr>
<tr><td>6</td><td></td><td></td><td></td><td></td></tr>
<tr><td></td></tr>
<tr><td>활동내용 및
회의
논의내용
기재</td><td colspan="5">Xycar의 카메라로 받아들인 이미지를 OpenCV를 이용하여 자율주행 스튜디오의 트랙의 하얀색 차선 검출

이번 주차를 기준으로 차선 추종 주행 알고리즘에 대한 연구 및 회의가 시작되었다. 먼저, 카메라로 취득한 영상에서 자율주행 스튜디오의 하얀색 선을 추출하기 위해서는 BGR 코드로 되어 있는 영상을 HSV 컬러 코드로 변환하는 작업을 진행해야 한다. 이후 HSV 컬러 코드 중 명도를 이용해, 오직 하얀색과 검정색으로만 색깔을 나타내는 이진화를 적용한 후 영상에서 Xycar가 인식해야 할, 하얀색 선의 위치를 찾아내는 알고리즘을 구현하여야 한다. 이때, 차선 인식에 방해되는 노이즈를 제거하기 위해, Gaussian Blur 기법을 이용할 수 있다. 이러한 이론을 토대로, 차선 추종 주행 알고리즘을 작성하였으나, 자율주행 스튜디오의 환경상, 자율</td></tr>
</table>

	주행 시 고려해야 할 점이 몇 가지 있었다. 1) 자율주행 스튜디오의 환경적 문제점 2) 자율주행 스튜디오 트랙의 넓은 폭 3) 안랙의 넓은 폭에 비해, 차선의 상대적으로 적은 폭 4) 강한 반사광과 특정 구간의 어두움 5) 트랙 정중앙의 하얀색 선 - 해결 방안 1) 차선을 잃어버렸을 경우의 대책을 마련한다 2) HSV 컬러코드를 이진화할 때, 기준 명도 범위를 조정한다

II. 주차별 활동 보고서 (12 주차)

<table>
<tr><td>팀명</td><td colspan="5">5조</td></tr>
<tr><td>과제참여
인원수</td><td colspan="2">5</td><td>회의참여
인원수</td><td colspan="2">5</td></tr>
<tr><td>연구 주제</td><td colspan="5">차선 추종 알고리즘 설계</td></tr>
<tr><td>활동일시</td><td colspan="2">주차</td><td>장 소</td><td colspan="2">자율 주행 스튜디오</td></tr>
<tr><td>활동 내용</td><td colspan="5">01. IMU의 개념
02.차선추종 알고리즘 설계방법</td></tr>
<tr><td rowspan="7">참여자 명단</td><td>연번</td><td>소속(전공)</td><td>이름</td><td>서명</td><td>비고</td></tr>
<tr><td>1</td><td>소프트웨어융합학과</td><td>김신건</td><td>김신건</td><td></td></tr>
<tr><td>2</td><td>소프트웨어융합학과</td><td>강경수</td><td>강경수</td><td></td></tr>
<tr><td>3</td><td>소프트웨어융합학과</td><td>고현성</td><td>고현성</td><td></td></tr>
<tr><td>4</td><td>소프트웨어융합학과</td><td>곽다윗</td><td>곽다윗</td><td></td></tr>
<tr><td>5</td><td>소프트웨어융합학과</td><td>김예리</td><td>김예리</td><td></td></tr>
<tr><td>6</td><td></td><td></td><td></td><td></td></tr>
<tr><td>활동내용 및
회의
논의내용
기재</td><td colspan="5">IMU 센서와 차선 추종 주행 알고리즘 설계 및 구현

지금까지는 다루지 않았던 IMU 센서에 대한 개념과 응용을 얻을 수 있었다. 또한, 차선 추종 주행 알고리즘을 창업연계공학설계입문 교과목에서 학습한 알고리즘과 다르게 개편하기로 결정하였다.

IMU: 관성 측정 장치인 IMU는 가속도계와 회전 속도계, 자력계의 조합을 사용하여, 특정 물체에 가해지는 힘이나 회전 각속도를 측정하는 장치이다. 각각, 선가속도를 측정하기 위해서는 가속도계를, 각속도를 측정하기 위해서는 자이로스코프를, 절대방향을 알아내기 위해서는 자력계를 이용하는데, 이때 직교하는 세 방향의 축 주위의 회전각을 나타내는 Roll, Pitch, Yaw를 이용한다.</td></tr>
</table>

	- 차선 추종 주행: 우리가 창업연계공학설계입문에서 학습하였던 차선 추종 주행 알고리즘은 왼쪽과 오른쪽의 차선 위치 좌표값을 알아내어, 이를 평균내어 도출한 위치의 방향으로 자율 주행 구동체의 회전 방향을 결정하는 것이다. 허나, 이는 11주차에서 논의하였던 바와 같이, 자율주행 스튜디오 환경상, 문제점이 발견되었고 새로운 알고리즘을 도입하자는 아이디어가 나왔다. - 새로운 알고리즘: 허프 변환을 이용해 차선 추종 주행 알고리즘을 새롭게 구성하였다. 카메라 입력을 왜곡하여, 사다리꼴 모양의 차선을 위에서 아래로 본 모양의 직사각형으로 만들고, Canny로 외곽선을 검출한 후, 허프 변환을 통해, 검출한 직선을 차선이라 가정하고 영상의 모든 하얀색 선의 각도를 판단한 뒤, 해당 각도들의 평균인 theta에 따라, 자율주행 구동체의 회전 방향을 결정하도록 작성하였다.

II. 주차별 활동 보고서 (13 주차)

<table>
<tr><td>팀명</td><td colspan="5">5조</td></tr>
<tr><td>과제참여
인원수</td><td colspan="2">5</td><td>회의참여
인원수</td><td colspan="2">5</td></tr>
<tr><td>연구 주제</td><td colspan="5">OpenCV를 이용해 Xycar 자율 주행 구현, 창연공 Ad 프로젝트 구현</td></tr>
<tr><td>활동일시</td><td colspan="2">13주차</td><td>장 소</td><td colspan="2">자율 주행 스튜디오</td></tr>
<tr><td>활동 내용</td><td colspan="5">01.차선 추종 알고리즘을 통한 Xycar 자율 주행
02.창연공 Ad 프로젝트</td></tr>
<tr><td>참여자 명단</td><td colspan="5"><table>
<tr><th>연번</th><th>소속(전공)</th><th>이름</th><th>서명</th><th>비고</th></tr>
<tr><td>1</td><td>소프트웨어융합학과</td><td>김신건</td><td>김신건</td><td></td></tr>
<tr><td>2</td><td>소프트웨어융합학과</td><td>강경수</td><td>강경수</td><td></td></tr>
<tr><td>3</td><td>소프트웨어융합학과</td><td>고현성</td><td>고현성</td><td></td></tr>
<tr><td>4</td><td>소프트웨어융합학과</td><td>곽다윗</td><td>곽다윗</td><td></td></tr>
<tr><td>5</td><td>소프트웨어융합학과</td><td>김예리</td><td>김예리</td><td></td></tr>
<tr><td>6</td><td></td><td></td><td></td><td></td></tr>
</table></td></tr>
<tr><td>활동내용 및
회의
논의내용
기재</td><td colspan="5">유레카 프로젝트 자율주행 경진대회를 차선 추종 주행 알고리즘 개선 및 고도화

12주차까지 설계/구현한 차선 추종 주행 알고리즘을 Xycar에 적용한 후, 유레카 프로젝트 자율주행 경진대회에 참가하였다. 자율주행 경진대회는 11월 29일 금요일에는 예비 주행으로, 주행이 잘 되는지 점검을 하며, 본 대회는 11월 30일 토요일에 조마다 정해진 시간대에 참석하여, 자율주행 자동차 Xycar가 차선 이탈 및 충돌 없이 자율주행 스튜디오의 트랙 한바퀴 완주에 걸리는 시간을 측정한다. 대회에 출전하기에 앞서, 우리 조는 안정적인 완성된 자율주행 알고리즘과 위험성은 다소 붙지만 더 빠른 속도로 달릴 수 있는 알고리즘 중, 자율주행 경진대회에서 1등을 노리고자 마음을 먹었기에, 더 좋은 기록을 낼 수 있는 Xycar의 주행 모드중 레이싱 모드에 맞추어 개발을 다시 시작하였다.</td></tr>
</table>

	- 레이싱 모드: 레이싱 모드는 일반적인 트레이닝 모드와 달리, 후진이 불가능한 모드이다. 대신, 실제 경주대회에서 쓰는 모드인 만큼, Xycar의 모터의 기능을 최대한으로 끌어내어 매우 빠른 속도로 달리는 것이 가능하기에, 해당 모드로 완주가 가능하다면, 대회에서도 좋은 성적을 낼 수 있을 것이라는 생각이었다. 기존 코드에서 조향각과 속도가 theta, K와 같은 변수를 이용한 최적화된 수식으로 되어있었기에, 이전보다 수월하게 레이싱 모드의 자율주행을 구현할 수 있었다. 허나, 레이싱 모드 특유의 빠른 속도와 자율주행 스튜디오의 반사광때문에, 충돌하거나 회전하지 못하는 경우가 상당수였기에, 이를 해결하는 것이 관건이었다. - 영상 녹화: 만약, Xycar의 카메라를 이용해, Xycar가 직접 차선을 인식하는 화면을 본다면, 주행할 당시, 어떤 구간에서 트랙의 선을 놓치게 되는지, 어디가 어려운 코스인지 파악할 수 있다고 생각했기에, Xycar의 카메라에 비춰지는 영상을 VNC로 연결된 노트북의 화면에 띄움과 동시에, Xycar를 자율주행하였다. 허나, 카메라 인식이 매우 느려지는 등, 성능에 큰 차질이 생기기에, 이는 포기할 수 밖에 없었다. 화면에 실시간으로 띄우는 것이 아닌, 영상을 녹화하는 것은 Xycar에 무리를 주지 않을 것이라 여겨졌기에, 이시윤 교수님과의 면담을 통해, 자율주행 중 카메라 영상을 녹화하는 방법에 대해 알아내었고, 레이싱 모드로의 자율주행에 이를 이용하였다.

Ⅲ. 프로젝트 수행 후기

연번	팀장(원)	역할 수행 내용	수행 후기
1	김신건	*유레카 프로젝트, 창업연계공학설계입문 수업에서 우리 조의 조장을 맡게 되었다. 조장으로서 팀원들에게 역할을 분배했고 프로젝트를 수행했다. 창업연계공학설계입문 2차 주행에서 장애물을 감지하고 정지해야 하는 주행을 하기 위한 코드를 제작하였다.*	*고등학교에서 로봇 동아리 RAPID를 통해 축구 로봇을 구현한 경험이 새록새록 떠올랐다. 축구 로봇을 구현하기 위해 소프트웨어를 개량하고, 아두이노, 회로, 센서를 다루었던 경험과 이번 Xycar 실습을 한 경험이 겹쳐 보여서 이전의 추억이 떠오름과 동시에 과거의 경험들을 활용할 수 있었다. 그리고 대학교에 와서 새로 배우는 임베디드 개념을 익히고 바로 적용해볼 수 있었으며, ROS에 대한 여러 개념을 배울 수 있었다. 그리고 팀원들과 함께 Xycar를 다루고 역할을 분배한 뒤, 프로젝트를 진행하면서 다양한 경험을 할 수 있었다.하지만 이번 Xycar 실습에서는 너무나 아쉬운 점이 많았다. 아직도 Xycar가 껐다 키거나 시간이 조금만 흘러도 다른 움직임을 보이는 이유를 파악을 못했다. 그리고 한 학기동안 진행되기에 벅찬 내용이라 생각될 정도로 바쁜 일정으로 ROS, Xycar에 대한 내용을 배우게 되면서 ROS, OpenCV, Xycar에서 더 알고 싶지만 넘긴 개념들도 있고 이해되지 않은 내용도 있어서 무척이나 아쉽다. 이후 이 수업을 듣게 되는 학생들은 조금 더 여유로운 일정과 내용을 가지고 학습을 하게 되면 너무나 좋은 수업이 될 것 같다고 생각하게 되었다.*
2	강경수	*유레카 프로젝트*	*내 인생의 임베디드 시스템과 자율주행*

		자율주행 경진대회를 위한 회의 중, 서기 역할을 수행하면서, 주행 시, 변경사항 및 충돌/이탈 원인을 기록하였으며, 코드 개선 사항에 대하여 아이디어를 내었다.	구동체는 햄스터 로봇뿐이던 19학번 신입생이었다. 정시전형으로 입학한 나에게는 소프트웨어적 지식이 전무했다. 따라서 이러한 팀 프로젝트가 주어졌을 때, 나는 '걸림돌은 되지 말자'는 생각이 들었다. 해당 프로젝트를 진행하면서, 자율주행 기초와 ROS 패키지 및 구독-발행 방식, 공학적 접근 방식의 공학설계 등 다양한 소프트웨어적 지식을 쌓을 수 있는 기회였다고 생각한다. 초음파 센서, IMU 센서, 카메라로 입력받은 데이터를 가공/처리하여 모터에게 명령을 자율주행 자동차를 주행시키는 과정의 원리를 이해하고 직접 프로그래밍을 하면서, 실제 자율주행 자동차 및 임베디드 시스템의 원리를 이해하는데 큰 도움이 되었다. 또한, Xycar가 원하는 대로 움직여주지 않거나, 실제 경주 대회에서도 이상 현상이 발생하였을 때, 팀원들이 서로 나서서 적극적으로 행동하고, 예기치 못한 상황에서도 당황하지 않고 협업하였기에, 팀원들이 버팀목 역할을 해주어, 포기하지 않을 수 있었던 것 같다. 이를 계기로 다시 한번 팀원들과 협동성이 목표 달성에 많은 영향을 끼친다는 것을 느낄 수 있었다.
3	고현성	자율주행 과제에서 주행 시 조향, 가속 판단을 위한 차선 검출 알고리즘을 시점 변환과 직선 검출을 이용해 제작하였다. 또, 우리 조의 AD 프로젝트의 주제 중 하나인 신호등 불빛에 따른 주행에서 신호등 인식 로직을 원 허프 변환과 HSV 색상 공간, 이진화 등을	주어진 문제를 공학적인 방식으로 해결하기 위해 새로운 방식을 도입하면서 결과의 질이 향상되는 것이 인상적이었다. 이 프로젝트를 진행하며 깨달은 점은 기존에 구현된 로직의 파라미터를 최적화하는 것보다 새로운 로직을 더 강건하게 구현하는 것이 더 강력하다는 것이다. 그 점은 앞으로도 하드웨어를 활용한 공학 프로젝트 뿐만 아니라 순수 소프트웨어 구현 및 테스트에서도 도움을 줄 것이라 기대한다. 로직을 구현하며 Bird's-eye view와 직선 허프 변환, 원 허프 변환 등의 저명한 알고리즘을 OpenCV, NumPy와 같은 고수준 라이브러리에서 직접 구현 및 사용해본 경험

		통해 구현하였다.	또한 차후의 소프트웨어 개발에 도움을 줄 것이라 생각한다. 또, 팀 프로젝트를 진행하며 팀원과의 의사소통 및 역할분담 등에서 배운 점이 있다고 생각한다. 서로 협업하며 주어진 과제를 수행했던 경험이 앞으로의 다른 팀 프로젝트에서도 도움이 될 것이라 생각한다.
4	곽다윗	초음파 센서로 장애물 인식하는 과제에서 주행 결과와 이슈를 기록하는 일과 간단한 코드 작성, 자율 주행과제에서는 차 옮기는 역할, 코드리뷰와 의견 제시, AD 과제에서는 앱인벤터를 가지고 블루투스 통신을 이용해 신호등앱을 만들었다.	ROS, OpenCV, Python3등을 사용하여 Xycar을 제어하는 활동을 통해 임베디드 개발을 할 때 하드웨어가 어떤 식으로 데이터를 주고 받는지, 이미지와 동영상의 프레임을 어떻게 개발에 활용할 수 있는지, 초음파 센서, 모터, 카메라의 작동원리등을 배울 수 있어서 좋았다. 또한 여러개의 과제를 통해 코드를 한번만 사용하는 것이 아니라 다음 과제에도 활용하면서 코드의 재활용성의 장점과 코드의 재활용과 유지보수를 쉽게할 수 있게 수식을 사용하여 값들이 연결되게 짜고, 모듈의 종속성은 최대한 없애는게 중요하다는 것을 깨닫게 되었다. 팀프로젝트를 진행하면서 팀원들이 서로 각자 맡은 역할을 말하지 않아도 하고 있고 팀원들이 활동을 적극적으로 참여해서 조별과제의 좋은 기억으로 남을 거 같다.
5	김예리	코드 구현에 대한 의견 제시, 주행 과제 시 코드 수정 이슈에 기록, AD 과제 알고리즘에 필요한 요소 제시	ROS 부터 Xycar가 카메라의 영상입력에 관련한 다양한 도구를 제공하는 OpenCV를 모두 처음 접하게 되어 새로운 지식에 대한 설렘도 있었지만, 처음에는 아무런 기초 지식이 없어 이해하기에 버거운 점도 많았다. 하지만 과제나 이번 주행대회를 준비하면서 조원들과 함께 주행 알고리즘을 구현하였고, 이는 창업연계공학설계 입문 수업에서 배운 내용 이외에 더 많은 것을 배울 수 있는 시간이 되었다. 또한, 마지막 ad project를 할 때 큰 도움이 되어 처음 배울 때보다 더 수월하게 할 수 있었다. 그리고 혼자 해봤을 때 이해가 되지 않는 부분은 조원들에게

			물어봐가면서 알고리즘쪽이나 OpenCV에 대한 이해도를 더 높일 수 있었고, 자율 주행 대회를 위한 알고리즘 구현을 통해 그 동안 주행 과제로 해 왔던 알고리즘에 대해 다시 한 번 되짚어 볼 수 있는 시간이 되었다. 본 강의가 조별 활동이라 개개인의 역할 분담이 중요했는데, 조원들간의 역할 분담이 잘 되어 있었고, 의견 조율도 원만하게 해결하였고, 또 모두가 열심히 해서 더 좋은 시너지를 낼 수 있었던 것 같다.