

# 초음파센서 기반 장애물 회피 주행

프로그래머스 자율주행 코스 1기 조정민

# 초음파 센서를 사용하여 장애물이 있으면 주행하는 기능 구현 :

## ultra\_gostop.py

```
#!/usr/bin/env python
#-*- coding:utf-8 -*-


import rospy
import time

from std_msgs.msg import Int32MultiArray
from xycar_motor.msg import xycar_motor

ultra_msg = None
motor_msg = xycar_motor()

def ultra_callback(data):
    global ultra_msg
    # 초음파센서 토픽이 들어오면 실행되는 콜백함수
    정의
    ultra_msg = data.data

def drive_go():
    global motor_msg
    motor_msg.speed = 20
    motor_msg.angle = 0
    pub.publish(motor_msg)
```




```
def drive_stop():
    global motor_msg
    motor_msg.speed = 0
    motor_msg.angle = 0
    pub.publish(motor_msg)

rospy.init_node("ultra_node")
rospy.Subscriber("xycar_ultrasonic", Int32MultiArray, ultra_callback)
pub = rospy.Publisher('xycar_motor', xycar_motor, queue_size=1)

while not rospy.is_shutdown():
    # 초음파센서 토픽이 오면 콜백함수가 호출되도록 세팅
    if ultra_msg == None:
        continue

    if (min(ultra_msg[1:4]) <= 40) :
        drive_stop()
    else:
        # 멈추지 않았을 때 계속 전진
        drive_go()
```



초음파 센서를 사용하여 장애물이 있으면 주행하는 기능 구현 :  
ultra\_gostop.launch

```
<launch>
  <node name="xycar_motor" pkg="xycar_motor" type="xycar_motor_b2.py" output="screen" >
    <param name="angle_offset" value="0" />
  </node>
  <!-- sensor_drive -->
  <node pkg="xycar_ultrasonic" type="xycar_ultrasonic.py" name="xycar_ultrasonic"/>
  <node name="ultra_driver" pkg="xycar_ultrasonic" type="ultra_gostop.py"
    output="screen"/>
</launch>
```

# 초음파 센서를 사용하여 장애물이 있으면 후진 후 주행하는 기능

구현: ultra\_drive.py

```
#!/usr/bin/env python
#-*- coding:utf-8 -*-
import rospy, time
from std_msgs.msg import Int32MultiArray
from xycar_motor.msg import xycar_motor
rospy.init_node("ultra_node")
motor_msg = xycar_motor()
ultra_msg = None
rate = rospy.Rate(10)
def ultra_callback(data):
    global ultra_msg
    # 초음파센서 토픽이 들어오면 실행되는 콜백함수 정의
    ultra_msg = data.data

def drive_go():
    global motor_msg
    motor_msg.speed = 30
    motor_msg.angle = 0
    pub.publish(motor_msg)
    rate.sleep()
```

# 초음파 센서를 사용하여 장애물이 있으면 후진 후 주행하는 기능

구현: ultra\_drive.py

```
def drive_opposite(angle):  
    global motor_msg  
    motor_msg.angle = 0  
    for sp in range(40):  
        motor_msg.speed = 10 - sp  
        pub.publish(motor_msg)  
        rate.sleep()  
    # 반대방향 주행  
    motor_msg.angle = 40 * (-angle) if (angle != 0) else 40  
    for sp in range(50):  
        motor_msg.speed = -5 + min(sp, 35)  
        pub.publish(motor_msg)  
        rate.sleep()  
  
rospy.Subscriber("xycar_ultrasonic", Int32MultiArray, ultra_callback)  
pub = rospy.Publisher('xycar_motor', xycar_motor, queue_size=1)  
  
time.sleep(3) #ready to connect lidar
```

```
while not rospy.is_shutdown():  
    # 초음파센서 토픽이 오면 콜백함수가 호출되도록 세팅  
    if ultra_msg == None:  
        continue  
    for i in range(1,4):  
        if(ultra_msg[i] <= 25):  
            drive_opposite(i-2)  
            break  
    else:  
        # 멈추지 않았을 때 계속 전진  
        drive_go()
```

초음파 센서를 사용하여 장애물이 있으면 후진 후 주행하는 기능  
구현: ultra\_drive.launch

```
<launch>
  <node name="xycar_motor" pkg="xycar_motor" type="xycar_motor_b2.py" output="screen" >
    <param name="angle_offset" value="0" />
  </node>
  <!-- sensor_drive -->
  <node pkg="xycar_ultrasonic" type="xycar_ultrasonic.py" name="xycar_ultrasonic"/>
  <node name="ultra_driver" pkg="xycar_ultrasonic" type="ultra_drive.py"
    output="screen"/>
</launch>
```



장애물 인식 영상

<https://youtu.be/ac1V57wUPA8>

장애물 인식후 회피 영상

<https://youtu.be/N7qTdl9soo4>