**스마트모빌리티설계**

**[과제3] IMU 센서 프로그래밍**

1. **코드**

#!/usr/bin/env python

import rospy

import time

from sensor\_msgs.msg import Imu

from tf.transformations import euler\_from\_quaternion

from std\_msgs.msg import String

Imu\_msg = None

def imu\_callback(data):

global Imu\_msg

Imu\_msg = [data.orientation.x, data.orientation.y, data.orientation.z, data.orientation.w]

def determine\_direction(current\_yaw, prev\_yaw):

diff = current\_yaw - prev\_yaw

if diff > 3.14:

diff -= 6.28

elif diff < -3.14:

diff += 6.28

if abs(diff) < 0.02:

return "Straight"

elif diff > 0:

return "Left"

else:

return "Right"

def send\_direction():

rospy.init\_node("imu\_direction\_monitor")

rospy.Subscriber("/imu", Imu, imu\_callback)

pub = rospy.Publisher("/direction", String, queue\_size=10)

prev\_yaw = None

rate = rospy.Rate(1) # 1Hz

while not rospy.is\_shutdown():

if Imu\_msg == None:

continue

(roll, pitch, yaw) = euler\_from\_quaternion(Imu\_msg)

if prev\_yaw is None:

prev\_yaw = yaw

continue

direction = determine\_direction(yaw, prev\_yaw)

prev\_yaw = yaw

rospy.loginfo("Yaw: %.4f, Direction: %s" % (yaw, direction))

pub.publish(String(data=direction))

rate.sleep()

if \_\_name\_\_ == "\_\_main\_\_":

try:

send\_direction()

except rospy.ROSInterruptException:

pass

1. **코드 설명**
   1. **imu\_callback() 함수**
      1. 기존의 roll\_pitch\_yaw.py와 동일하며, imu 정보를 Quaternion 형태로 받아 전역변수에 저장합니다.
   2. **determine\_direction() 함수**
      1. 현재 yaw 값과 이전 yaw 값을 비교하여 자동차가 직진, 좌회전, 우회전 중인지 판단하는 함수입니다. 좌회전의 경우 yaw 값이 증가, 우회전의 경우 yaw 값이 감소하기 때문에, 현재 yaw 값과 이전 yaw 값의 차이가 양수인 경우 좌회전, 음수인 경우 우회전으로 판단하도록 구현하였습니다. 또한 yaw 값이 미세하게 변한 경우는 직진하고 있다고 판단하도록 threshold 값을 0.02로 지정하였습니다.



* 1. **send\_direction() 함수**
     1. 전체 프로그램의 주요 함수로, euler\_from\_quaternion() 함수를 통해 Quaternion 형식의 IMU 정보를 (roll, pitch, yaw)로 변환하고, determine\_direction() 함수를 반복적으로 호출하여 자동차의 진행 방향을 1초마다 /direction 토픽으로 publish하면서, 현재 yaw 값과 direction을 출력하는 함수입니다.
  2. **기존의 roll\_pitch\_yaw.py와 차이점**
     1. String 메세지 사용

/direction 토픽으로 문자열을 publish하기 위해 std\_msgs의 String을 사용합니다.

* + 1. prev\_yaw 변수 추가

determine\_direction() 함수에서 현재 yaw 값과 이전 yaw값과 비교를 하기 위해 이전 yaw값을 저장하는 변수를 추가하였습니다.

1. **실행 결과 및 분석**

아래 사진은 roll\_pitch\_yaw\_hw.launch를 실행한 후, /direction 토픽을 echo한 결과입니다. 좌우 결과를 비교해보면 yaw 값이 -0.2609이 된 순간부터 /direction 토픽을 echo하였음을 알 수 있습니다.

또한 이전 yaw 값인 -0.4026에서 -0.2609로 증가하여 차량이 좌회전을 한 상태에서 Left를 출력한 것으로 보아 프로그램이 잘 판단하고 있으며, /direction 토픽으로 잘 publish하고 있음을 알 수 있습니다. 추가적으로 rqt\_graph를 확인하여도 /direction 토픽으로 잘 publish하고 있음을 확인할 수 있습니다.

이후 결과들을 확인하여도 yaw 값이 증가/감소함에 따라 좌회전/우회전을 잘 판단함을 확인할 수 있고, determine\_direction()에서 구현한대로 yaw 값이 0.02 미만으로 미세하게 변한 경우 직진으로 잘 판단함을 확인할 수 있습니다.



텍스트, 소프트웨어, 멀티미디어 소프트웨어, 스크린샷이(가) 표시된 사진

AI가 생성한 콘텐츠는 부정확할 수 있습니다.