

3분반 1조 AD 프로젝트

# 창업연계공학설계입문

신무현 박채연 심혜린 정윤식 김미선



# 개요

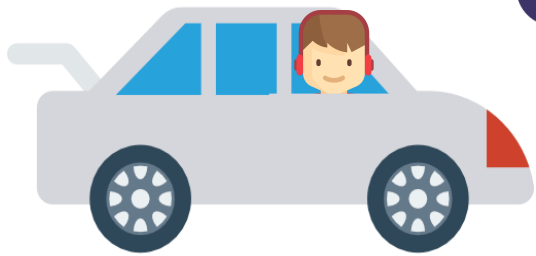
주제 : 학교 가는 길



## PROFILE

- 이름 : 국민이
- 학교 : 국민대학교

# 개요



# 개요

초음파 센서

usb

IMU 센서

Open CV

# 주행코스

- 9가지 미션 :
  1. 버스정류장에서 정차 - 글자인식
  2. 속도 줄이기 - QR 코드
  3. 원래 속도로 돌아오기 - QR 코드
  4. 언덕 통과하기 - IMU 센서
  5. 좌회전 - 차선 인식
  6. 속도 빠르게 올리기 - QR 코드
  7. 스쿨존에서 속도 줄이기 - 글자 인식
  8. 고라니가 지나갈 때까지 멈추기 - QR 코드 , 초음파 센서
  9. 목적지 도착 - QR 코드

오르막길, 내리막길



# 오르막길, 내리막길



# 설계과정

목표 : 오르막길, 내리막길 → 적절한 속도

속도 조절





# 설계과정

## - IMU 센서



□ Pitch → 차의 기울기 정도

□ Roll

□ Yaw

# 설계과정

- Pitch 값 계산

- 오르막길 : 속도  $\uparrow$   $\Rightarrow$  매끄러운 언덕 주행 0

- 내리막길 : 제어

# 구현과정

차량의 무게를 감당할 수 있는 언덕



# 구현과정

- IMU 데이터 발행 코드

```
import rospy
from diagnostic_msgs import DiagnosticArray
```

```
class ImuRead:
```

```
def __init__(self, topic):
    self.roll = 1
    self.pitch = -1
    self.yaw = -1
    rospy.Subscriber(topic, DiagnosticArray, self.read_data)
```

```
def read_data(self, data):
    status = data.status[0].values
    self.roll = status[0].value
    self.pitch = status[1].value
    self.yaw = status[2].value
```

```
def get_data(self):
    return float(self.roll), float(self.pitch), float(self.yaw)
```

Yaw 값 이용

But, 여전히 빠른 속도

- 오르막길 : 속도 ↑

=> 후진 이용 → 브레이크

- 내리막길 : 속도 ↓

# 표지판 인식



# 표지판 인식



# 설계과정

- 목표 : 교통 표지판 → QR 코드를 이용해 구현

## QR 코드

- 데이터 → 표지판의 이름을 텍스트 처리
- 핵심 설계 : QR 코드 인식

데이터 파악

XyCar 제어

# 구현과정

Pyzbar 모듈

└ OpenCV : QR코드 인식 → 데이터 전달 → 표지판 파악



# 구현과정

Pyzbar 모듈

└ OpenCV : QR코드 인식 → 데이터 전달 → 표지판 파악

# 구현과정

## - QR 코드 생성 코드

```
import qrcode

qr = qrcode.QRCode(
    version = 1,
    error_correction = qrcode.constants.ERROR_CORRECT_L,
    box_size = 10,
    border = 4,
)

qr.add_data('Information')
qr.make(fit = True)

img = qr.make_image()
img.save('filename.png')
```

# 구현과정

## 0 코드 작성

- 속도 제한 표지판 : 속도 변경
- STOP 표지판 : 완전히 멈춤
- 야생 동물 출현 주의 표지판 : 서행 + 물체 감지 → 멈춤

# 구현과정

## - 코드 구현

```
import pyzbar.pyzbar as pyzbar
import cv2
import matplotlib.pyplot as plt
import rospy
from sensor_msgs.msg import Image
from cv_bridge import CvBridge

import numpy as np

class QRCodeCheck:
    def __init__(self, topic):
        self.cam_img = np.zeros(shape=(480, 640, 3), dtype=np.uint8)
        self.bridge = CvBridge()
        rospy.Subscriber(topic, Image, self.conv_image)

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

# 구현과정

## - 코드 구현

```
def detect_QR(self):  
    img = self.cam_img  
    plt.imshow(img)  
  
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)  
  
    plt.imshow(gray, cmap='gray')  
  
    decoded = pyzbar.decode(gray)  
  
    for d in decoded:  
        print(d.data.decode('utf-8'))  
        print(d.type)  
  
        return str(d.data.decode('utf-8'))  
        cv2.rectangle(img, (d.rect[0], d.rect[1]), (d.rect[0] + d.rect[2], d.rect[1] + d.rect[3]), (0, 0, 255), 2)  
  
    plt.imshow(img)
```

---

# 글자 인식



# 글자 인식



# 설계과정

- 행동 구현 : OpenCV를 활용한 글자 인식
- 목표 : 'SCHOOL ZONE' 인식 → 속도 ↓  
          'BUS ZONE' 인식 → 잠시 멈춤



# 설계과정

목표 수행

： 카메라를 통해 영상 불러오기

## OCR 기술

이미지 속에 들어 있는 문자를 추출해 내는 기술

# 설계과정

- 글자 인식 방법

  - : 'Tesseract' 오픈 소스 활용

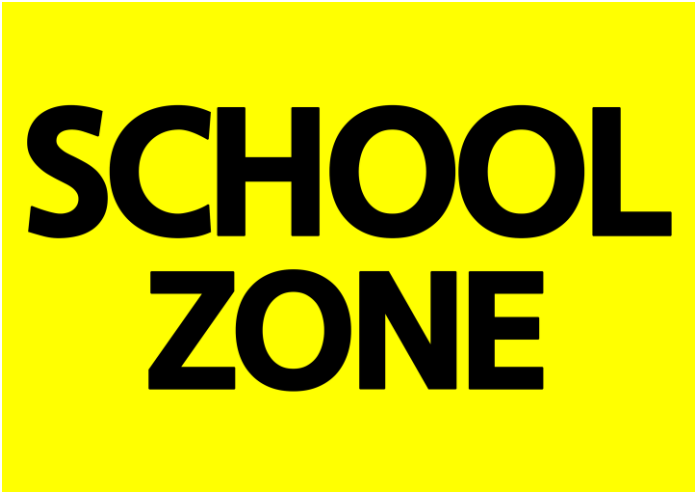
    - 'pytesseract' 사용

# 구현과정

- 인식률 : ~~영어~~ > 한글



**BUS  
ZONE**



**SCHOOL  
ZONE**

# 구현과정

## - 글자인식 테스트 코드

```
cap = cv2.VideoCapture(0)
```

```
ts = pytesseract.image_to_string(im)
```

```
while True:
```

```
    ret, frame = cap.read()
```

```
    hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)
```

```
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
```

```
    l = np.array([0,0,150])
```

```
    u = np.array([179,255,255])
```

```
    mask = cv2.inRange(hsv, l, u)
```

```
    if cv2.waitKey(1)>0:
```

```
        break
```

```
    cv2.imshow('test', mask)
```

```
    ts = pytesseract.image_to_string(gray)
```

```
    print(ts)
```

```
cap.release()
```

```
cv2.waitKey(0)
```

```
cv2.destroyAllWindows()
```

# 구현과정

0 Tesseract를 XyCar에서 사용하려면?

→ 자동차에 위 모듈 설치

```
sudo apt install tesseract-ocr tesseract-ocr-kor  
sudo pip3 install pytesseract
```

# 구현과정

## - XyCar에서의 코드 구현

```
import rospy
import cv2
import numpy as np
import pytesseract
from sensor_msgs.msg import Image
from cv_bridge import CvBridge

class numCheck:

    def __init__(self, topic):

        self.cam_img = np.zeros(shape=(480, 640, 3), dtype=np.uint8)
        self.bridge = CvBridge()
        rospy.Subscriber(topic, Image, self.conv_image)

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

```
def detect_numbers(self):

    gray = cv2.cvtColor(self.cam_img, cv2.COLOR_BGR2GRAY)

    if cv2.waitKey(1) & 0xFF == 27:
        quit()
    cv2.imshow("view", gray)

    ts = pytesseract.image_to_string(gray)
    print(ts)

    if 'SCHOOL' in ts:
        print('SCHOOL')
        return 'SCHOOL'
    elif 'BUS' in ts:
        print('BUS')
        return 'BUS'
```

# 영상 시청

깃허브 주소

[https://github.com/smh3223/CYG\\_AD\\_Project](https://github.com/smh3223/CYG_AD_Project)



# Thank you

