


DU-도전학기 결과보고서

과제명	시각장애인을 위한 실내 안내로봇		
참여자	성명	소속	학번
	이00	전기공학전공	
	한00	전기공학전공	
	홍00	전기공학전공	
지도교수 의견	김00	전기공학전공	
	짧은 기간 내에 전공지식을 살려 발전된 공학적 지식을 습득하였고 이를 기반으로 작품을 완성하였음. 이번 경험을 토대로 앞으로의 개인 능력의 향상을 기대해 볼 수 있겠음.		
	(소속) 전기공학전공	(성명) 도용태	(서명)  는 날인)

1. 도전 과제 내용

과제 내용: 시각장애인을 위한 실내 안내 로봇 제작

- 경로에 따라 사용자에게 위치와 방향을 안내
- 장애물을 감지하여 충돌을 방지
- 시각장애인이 사용하기 편하도록 하드웨어 설계
- 카메라를 이용해 주변 사물 파악

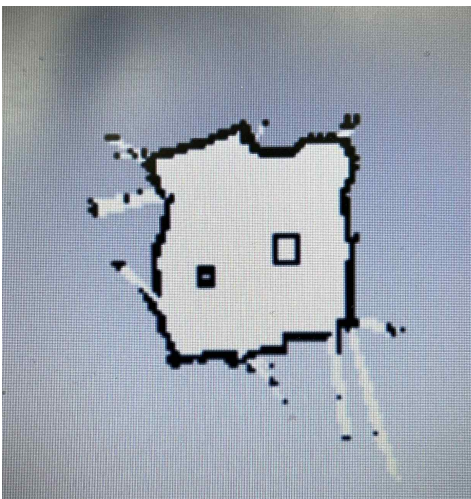
2. 도전 과제 수행 결과 및 성과

■ 시각장애인을 위한 실내 안내 로봇



시각장애인을 위한 실내 안내 로봇

■ 실내 지도 저장 및 주행 기능 개발 (이00)



```
while(self.get_dist(goal_pose) >= tolerance):

    if(cnt4print >= 10):
        cnt4print = 0
        self.print_pose()

    cnt4print = cnt4print + 1

    t.linear.x = self.get_lin_x(goal_pose)
    t.linear.y = t.linear.z = 0

    t.angular.x = t.angular.y = 0
    t.angular.z = self.get_ang_z(goal_pose)

    self.pub.publish(t)
    self.rate.sleep()

t.linear.x = t.angular.z = 0
self.pub.publish(t)
print("Arrived at target positon")
```

```
def __init__(self):
    rospy.init_node('move_to_goal', anonymous = True)
    self.sub = rospy.Subscriber('/odom', Odometry, self.get_odom )
    self.pub = rospy.Publisher( '/cmd_vel', Twist, queue_size = 10)
    self.rate = rospy.Rate(10)

    self.pos_x_2d = 0.0
    self.pos_y_2d = 0.0
    self.theta_2d = 0.0
    self.prev_theta_2d = 0.0
    self.theta_2d_sum = 0.0
```

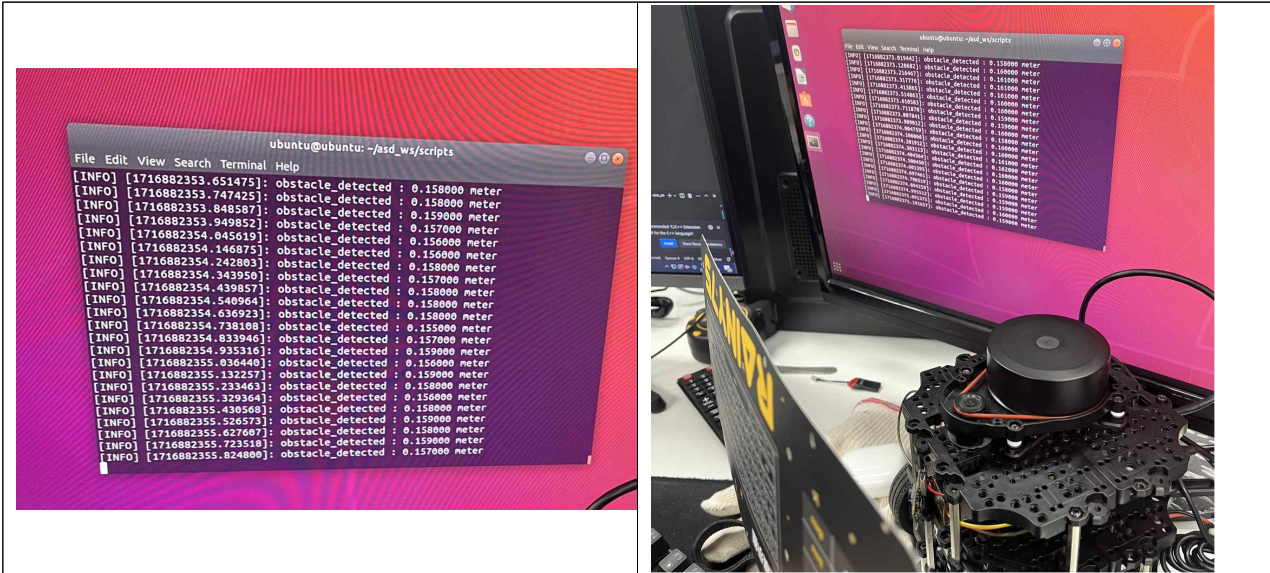
```
def move2goal(self):
    goal_pose = Pose()

    goal_pose.x = input("Input x goal: " )
    goal_pose.y = input("Input y goal: " )
    tolerance = input("Input tolerance: ")
```

- 실내 지도 저장 및 활용 환경 구축
- 저장된 지도 내에서 좌표를 통한 로봇의 초기 좌표 설정
- 목표 위치의 좌표 입력을 통한 좌표까지의 주행 기능 개발

- 로봇의 좌표와 목표좌표, 각도가 일치하면 주행을 종료하는 기능 개발

■ 장애물 감지 기능 개발 (김00)



```
def callback(scan):
    lidar_distance = 1.0
    danger_detected = False
    obstacle_distance = 0.5
    noise_count = 0
    danger_threshold = 5

    for distance in scan.ranges:
        if distance < obstacle_distance:
            noise_count += 1
        else :
            noise_count = 0

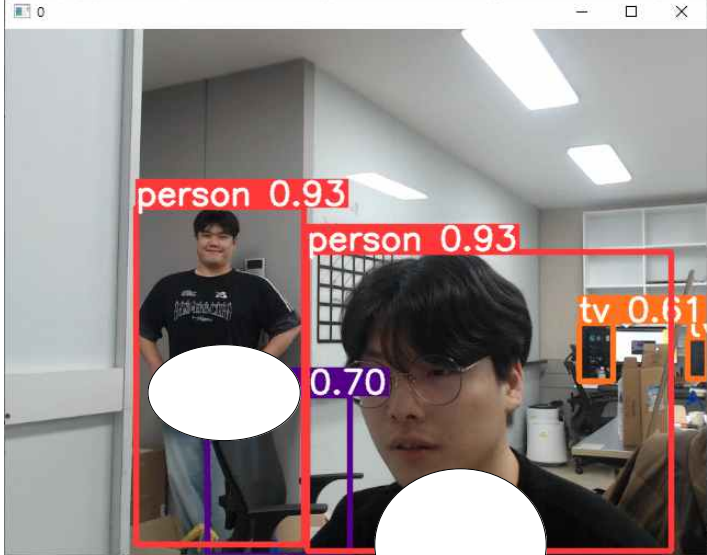
    if noise_count >= danger_threshold:
        rospy.loginfo("obstacle_detected : %f meter", distance)
        danger_pub.publish(True)
    else:
        rospy.loginfo("safe")
        danger_pub.publish(False)

def main():
    global danger_pub
    rospy.init_node('obstacle_detector_node', anonymous=True)
    rospy.Subscriber('/scan', LaserScan, callback)
    danger_pub = rospy.Publisher('/danger_signal', Bool, queue_size=10)
    rospy.spin()
```

LiDAR를 활용한 장애물 감지 알고리즘

- LiDAR센서 주변의 특정 각도, 거리 내에 존재하는 장애물 감지 알고리즘 생성
- 장애물 감지 시 생성된 danger_signal 노드로 Bool형태로 신호 전송(안전 : 0, 위험 : 1)
- 노이즈 제거를 위해 센서가 5번 이상 감지 시 신호를 보낼 수 있도록 noise_count 생성
- 센서가 감지하는 장애물의 거리를 실시간으로 출력
- LiDAR센서와 1미터 거리에 떨어진 장애물부터 감지할 수 있도록 설정 (50cm이내에 감지시 위험신호 발생)

■ 카메라를 통한 객체 인식 기능 개발 (한OO)



```
while 1:
    r, f = cap.read()
    detect = 0
    gray = cv2.cvtColor(f, cv2.COLOR_BGR2GRAY)

    # 노이즈 제거 (가우시안 블러 사용)
    gray = cv2.GaussianBlur(gray, (5, 5), 0)

    obj2_detect = obj2.detectMultiScale(gray, 1.2, 3)
    obj_detect = obj.detectMultiScale(gray, 1.2, 3)

    if len(obj_detect):
        detect = 1
        for (x, y, w, h) in obj_detect:
            cv2.rectangle(f, (x, y), (x+w, y+h), (0, 0, 255), 2)
    elif len(obj2_detect):
        detect = 1
        for (x, y, w, h) in obj2_detect:
            cv2.rectangle(f, (x, y), (x+w, y+h), (255, 0, 0), 2)

    pub.publish(detect)

    cv2.imshow('Video', f)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

cap.release()
cv2.destroyAllWindows()
```

객체 인식 알고리즘

- OpenCV를 사용해 객체 인식 알고리즘 생성
- 객체인식 시 object_detect 노드로 int 형태의 신호 전송
- 가우시안 블러를 사용해 노이즈 제거
- 데이터 학습을 통해 다양한 사물 인식 가능
- 높은 인식률을 위해 그레이 스케일 등 다양한 이미지 전처리
- 시각화를 위한 코드 삽입

■ 음성인식, 소리 출력 알고리즘 개발, 개발환경 구축, 구동부 개발 (홍00)

```
pub = rospy.Publisher('voice_recognition', String, queue_size=1)
rospy.init_node('voice_recognition', anonymous=True)
rate = rospy.Rate(10)

r = sr.Recognizer()

mic = sr.Microphone(device_index = 1)
with mic as source:
    r.adjust_for_ambient_noise(source)
    audio = r.listen(source, timeout = 5, phrase_time_limit = 5)

try:
    result = r.recognize_google(audio, language = "ko-KR")
    print(result)
except speech_recognition.UnknownValueError:
    print("음성 인식 실패")
except speech_recognition.RequestError:
    print("HTTP Request Error 발생")
except speech_recognition.WaitTimeoutError:
    print("WaitTimeout Error 발생")

pub.publish(result)
```

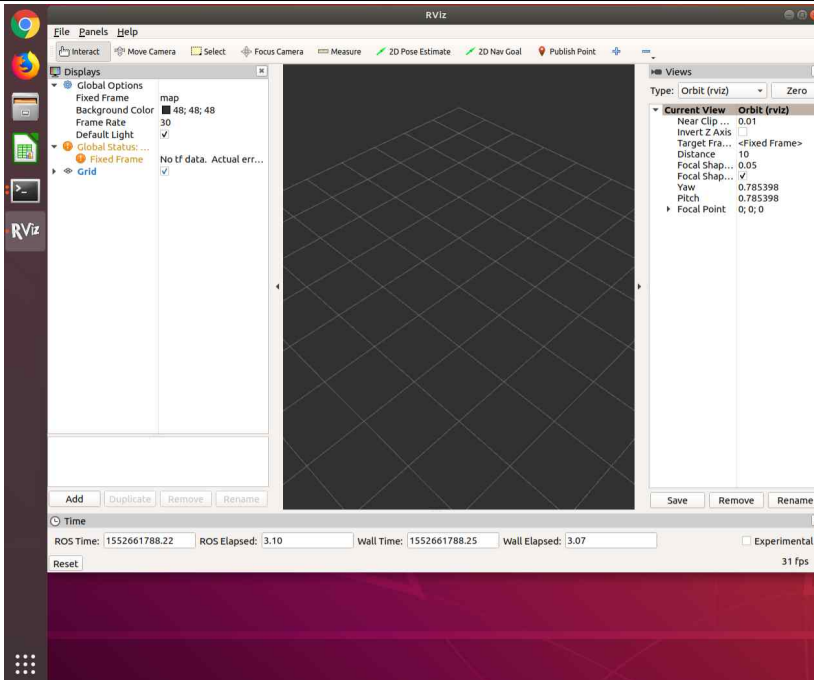
음성인식 알고리즘

```
import sys
reload(sys)
sys.setdefaultencoding('utf-8')

from playsound import playsound
from gtts import gTTS

if __name__ == '__main__':
    tts = gTTS(
        text='시설 안내중입니다. 피해서 지나가주시면 감사드리겠습니다.',
        lang='ko', slow=False
    )
    tts.save('human_detect.mp3')
```

음성 지정 및 저장 알고리즘



개발 환경 구축

- raspberry pi, ros melodic 등 초기 개발 환경 구축
- 아두이노를 이용해 OpenCR 및 모터 구동 부분 구현
- 회로 조립 및 하드웨어 결합
- Python을 활용해 TTS 기능을 구현
- 지정된 문구를 mp3파일로 변환하여 저장 및 실행 가능한 알고리즘 생성
- 음성인식 API와 Python을 활용하여 음성인식 알고리즘 생성
- 음성인식을 하여 생성된 문장을 voice_recognition 노드를 통해 전송

3. 자기 평가

이름	자기평가
김00	ROS를 활용하여 LiDAR센서의 데이터를 받아오고 값을 정제하여 새로운 노드를 통해 데이터를 보내는 방식을 습득했는데, 생각보다 어렵지만 굉장히 뜻깊은 경험이었고 이를 통해 한단계 더 발전할 수 있는 계기가 된 것 같다.
이00	실내 안내 로봇을 개발하면서 목표 위치까지의 네비게이션 주행을 중점으로 두고 개발을 주로 했는데 중간에 어려운 부분이 있더라도 포기하지 않고 팀원들과 함께 개발해 나가니 좋은 결과가 나왔고 나에게도 좋은 경험이었다.
한00	이번 팀 프로젝트 공동 개발을 진행하면서 기존에 내가 활용 가능했던 기술들을 응용해 더욱 발전된 작품을 만들 수 있어서 자기개발 측면에서 좋은 경험이었다.
홍00	음성인식, TTS출력에 중점을 두고 개발을 했는데 이 과정에서 개발환경이 리눅스인 상태에서 진행하니 굉장히 오류가 많이 발생해 애를 먹은 기억이 컸다. 그래도 리눅스 환경 내에서 개발 경험을 토대로 나에게도 좋은 경험이 될 수 있었고 나의 진로를 결정할 때에도 활용할 수 있도록 앞으로도 노력해야겠다.

4. 최종 결과물

■ 시각장애인을 위한 실내 안내 로봇



- 실내 지도 저장 및 주행 기능 개발 (이00)
- 장애물 감지 기능 개발 (김00)
- 카메라를 통한 객체 인식 기능 개발 (한00)
- 음성인식, 소리 출력 알고리즘 개발, 개발환경 구축, 구동부 개발 (홍00)