실증적SW개발프로젝트 주간보고 (8주차)

작성일: 2025/04/28 팀명: Wiper

<table>
<tr><td rowspan="4">팀 활동
보고</td><td>활동일시</td><td>2025.04.21 ~2025.04.28</td></tr>
<tr><td>장 소</td><td>동아대학교 Makespace 창업실 5번방</td></tr>
<tr><td>참 석 자</td><td>박준현, 서지완, 손주석, 이준영, 최창욱</td></tr>
<tr><td>특이사항</td><td>없음</td></tr>
<tr><td>이번주
진행사항</td><td colspan="2">

1. 개발내용

자율 주행 알고리즘 구현

- SPI 통신 구현 및 객체 인식 데이터 송수신
- 회피 알고리즘 구현
- 라인트레이서 부착 및 구동 테스트 진행 완료
- Jetson orin nano 신호등 색 인식 제어 완료
- MCU보드 라인트레이싱, 초음파 센서 제어

비, 눈 환경의 디헤이징 테스트 완료

임시 트랙 제작

2. 팀원별 활동내용

손주석(팀장, 제어 알고리즘 구현, 발표 자료 제작)

내용

SPI 통신 구현 및 객체 인식 데이터 송수신

1. MCU 설정

i) ioc의 CS 핀을 NSS HARD INPUT으로 설정함.

ii) 수신할 데이터 자료형 설정 및 구현

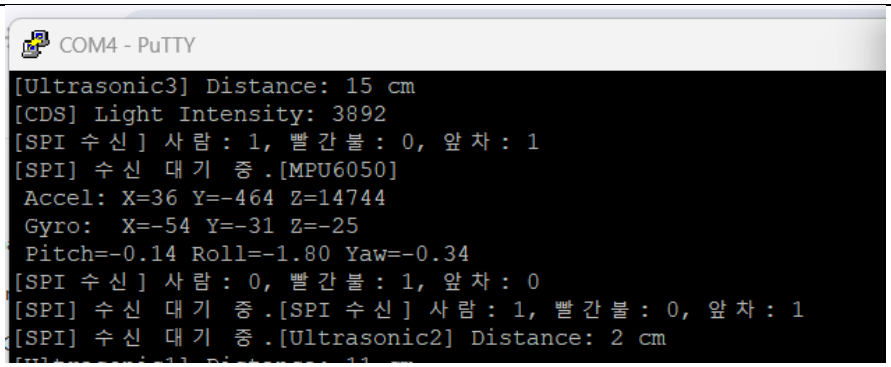
-> 객체별로 index를 할당한 bit연산을 수행해 자료형을 최소화함.

2. Jetson 설정

i) Jetson을 Master로 통신이 필요할 때마다 MCU를 NSS 방식으로 선택해 관리함.

ii) Dehazing과 Yolo를 통해 인식한 객체를 확인 한 후, 주행에 유의할 객체가 발생 또는 소멸한 경우 정보를 MCU에 전달해 제어 신호를 시행하도록 함.

</td></tr>
</table>



회피 알고리즘 구현

1. 객체 인식 데이터 관리
    i) 인식된 객체를 jetson으로부터 수신해 피해야 할 객체의 정보를 기록함.
2. 초음파 센서를 이용한 거리 감지
    ii) 초음파 센서를 이용해 해당 객체의 거리를 계산함.
3. 모터 드라이버 제어
    iii) 해당 객체와 충돌이 일어나지 않도록 현재는 멈추는 방식으로 작동하게 함.

발표 자료 작성

1. 팀원의 전체 업무 재정리 및 요약
2. 전체 발표 플로우 계획 및 팀원과 논의
3. 발표 자료 전체 작성 및 발표 연습을 통해 팀원과 피드백 진행
4. 발표 자료 사진 및 영상 촬영

주간 보고서 작성

박준현(팀원, 디헤이징, 주행 알고리즘)

내용

비, 눈 환경의 디헤이징 테스트

1. 비, 눈 환경에서의 디헤이징 성능 테스트
    i) 테스트를 하기 위한 비가오는 환경의 도로 이미지 수집
    ii) 각각의 이미지에 대한 디헤이징 실행
    iii) 디헤이징을 진행한 이미지를 yolo 객체 인식 테스트
2. 결과
    i) 비 디헤이징

전

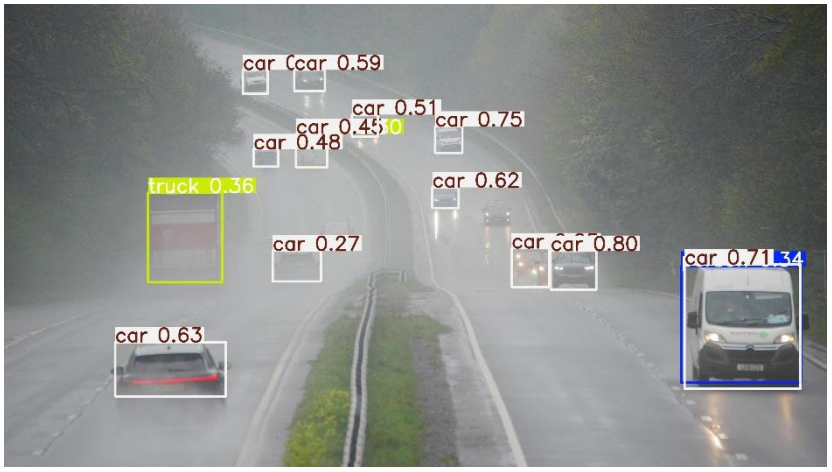
후

ii) 눈 디헤이징

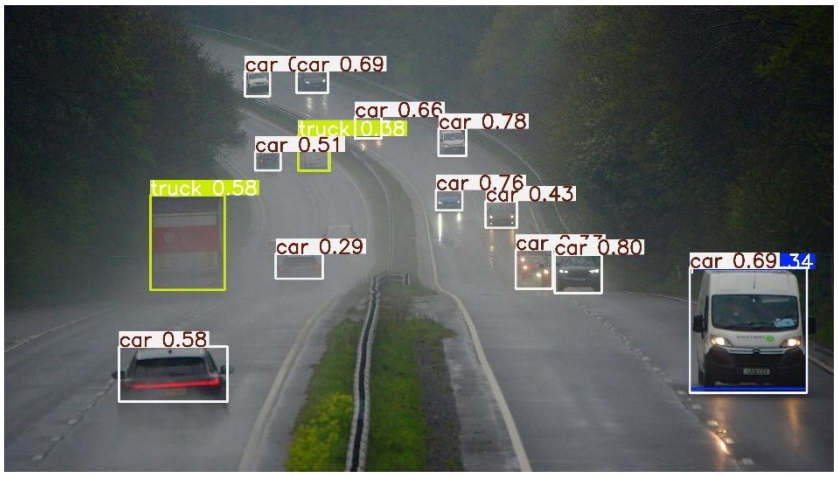
전

후

iii) 객체인식



디헤이징 x 비



디헤이징 o 비

디헤이징 x 눈

디헤이징 o 눈

iiii) 결론

비와 눈 환경에서도 문제 없이 디헤이징을 잘 적용된 것을 확인

특히 디헤이징을 적용하기 전에는 인식하지 못하던 객체들까지 잘 인식한것을 볼 수 있었음

안개와 같이 비, 눈 에서도 가능했던 이유:비, 눈, 안개 모두 공기 중에 작은 입자가 떠 있는 상태. 이 입자들이 빛을 산란시키며 카메라는 이 산란된 빛을 직접 받아들임 결국, 세가지 모두 희뿌옇고, 윤곽이 사라지며 가시거리가 짧아짐

디헤이징은 이러한 희뿌연 이미지를 개선하는데 특화되어 있어 3가지 case에서 모두 좋은 성능을 기록함

주행 알고리즘 개선

1. 주행 알고리즘 개선

i) 현재, 라인트레이서를 기반으로 주행 알고리즘이 작성되어있음, 그러나 현재 불안정하게 주행이 일어남.

ii) mcu 보드의 코드 작성을 하기 위한 환경 세팅을 완료하고, 주행 알고리즘 개선 방안 탐색중

서지완(팀원, 태스크 정의 파일 분리, 라인트레이서 부착 및 테스트, MCU 코드 리팩토링, 자율주행 알고리즘 구현 및 테스트 진행)

내용

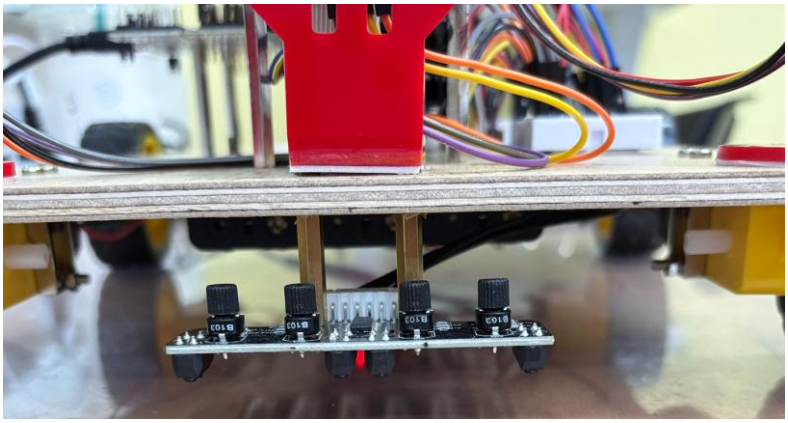
1. 태스크 정의 파일 분리

i) task_manage.c/h 파일을 생성하여 freertos.c에 있던 tasks 정의를task_manage.c/h에 분리하여 유지보수하기 편리하게 수정

ii) freertos.c에서는 tasks를 생성하여 실행만 하도록 구현

2. RC카에 라인트레이서 부착 및 테스트

i) RC카에 전면부 하단에 라인트레이서를 부착할 구멍을 뚫고, 지지대로 라인트레이서를 고정



ii) 라인트레이서를 부착 후, lineTracer.c/h 안에 라인트레이서 읽기 함수를 구현하여 테스트를 진행

```
[Line] L:0 C:1 R:0 -> Dir:2
[Line_Trace] 직진
```

3. MCU 코드 리팩토링

i) mpu6050.c

- 배열 기반 구조화
- ReadSensorData() 함수를 도입하여, 가속도/자이로 데이터 모두 처리

ii) bluetooth.c

- SendMotorCommand() 함수를 도입하여, 중복되는 UART 디버깅 출력 정리

4. 자율주행 알고리즘 구현

i) automotive.c

- LineTracerDriveDecision() 함수를 도입하여, 라인 감지 방향에 따라 Motor_Forward, Motor_Left, Motor_Right 호출하도록 구현
- 복구 로직 :

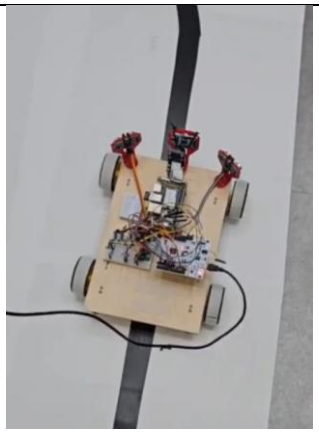
(1) 1초 이내: 좌/우 회전만 시도

(2) 1~3초: 후진 후 좌/우 회전

(3) 3초 초과: 정지

ii) task_manage.c

- StartLineTracerTask() 태스크 추가 : 라인트레이서 상태를 읽고 방향을 결정하도록 하였음



이준영(팀원, Jetson 신호등 색 인식, 라인 트레이싱과 초음파 센서 제어, 임시트랙 제작)

내용

Jetson 신호등 색 인식 제어

1. yolov5의 신호등 객체 안의 색 인식
    i) 신호등에 대해서 색이 있는 부분을 박스
    ii) 해당 박스의 색이 빨간색일 경우 spi로 명령을 보낼 수 있게 함.

라인 트레이싱과 초음파 센서 제어

1. 라인 트레이싱에서 빠른 속도 때문에 라인을 벗어나버리는 문제 발생
    i) 시작 속도에서 점차적으로 속도를 낮추어 원할한 제어가 되는 속도를 찾아보았음.
2. 초음파 센서가 정확한 인식을 못하는 문제 발생
    i) 조원들과 함께 문제점을 찾는 도중 최창욱 학생이 센서 타이머를 맞추고, 현재 환경에서 알맞은 값을 계산하도록 코드를 변경함으로 해결됨.
    ii) 추가적으로 30cm 안에 물체가 들어오면 차량의 속도를 0으로 하여 멈추고, 장애물이 사라질 경우 출발하도록 코드를 작성해보았음.

3. 임시로 테스트할 트랙 제작
    i) 바닥 재질은 마찰력이 어느 정도 일정한 박스를 이어 붙여 제작함.
    ii) 박스 위 라인 트레이싱을 위한 검정 테이프를 커브 각도에 맞게 설치

최창욱(팀원, GIT 연동, 트랙 제작, 초음파 센서 디버깅)

내용

GIT 연동

1. 진행 상황
    SSH 키 발급 및 연동 과정을 완료하여,
    현재 Git Bash를 통해 풀링(Pull)및 푸쉬(Push)모두 가능하도록 설정 완료.

임시 트랙 제작(설계)

1. 진행 상황

i) 처음에는 하얀 종이위에 검은색 테이프를 붙여 트랙을 제작했으나,
마찰력 부족문제 발생 → RC카 타이어에 찍찍이를 부착해 마찰력 보강 후 테스트 진행.
ii) 임시 트랙 재질 선별을 위해 우드락을 구매했으나, 마찰력이 충분하지 않아 부적합판명.
iii) 현재는 박스 여러 개를 이어 붙이고, 검은 테이프로 라인을 그려 임시 트랙을 구성하여 테스트 중.

2. 장애물 제작
i) 장애물 사진(사람, 자동차 뒷모습, 신호등)을 프린트.
ii) 재활용 불가능했던 우드락을 지지대로 활용하여 장애물 거치대 제작.
iii) 장애물 종류는 총 5개:
사람 (2종)
자동차 (2종)
신호등 (초록불, 빨간불)

초음파 센서 디버깅
1. 문제 인식
임시 트랙에서 정상 주행 도중, 초음파 센서 중앙(D2) 감지가 비정상적으로 작동하는 것을 발견
2. 디버깅 과정
i) 하드웨어 점검:
오실로스코프를 통해 TRIG, ECHO 신호를 분석 → 신호 자체는 정상.
선 연결 상태 점검 → D2 센서 선 연결 오류발견 및 수정.
ii) 소프트웨어 점검:
거리 계산 수식에 임시 보정값(×3.0)을 적용하여 먼 거리 감지에는 성공했으나,
30cm 이하에서 거리가 튀는 문제 발생.
클럭/타이머 세팅 점검 결과, 타이머 클럭이 16MHz인데 Prescaler를 잘못 설정(83)했던 것을 발견.
Prescaler를 수정하여 최종적으로 정상 거리 측정 성공.

라인트레이서 및 DC모터 테스트
1. 라인트레이서 테스트
초음파 디버깅 이전에 진행.
주행이 정상적으로 진행되지 않아 라인트레이서 문제로 판단, 팀원들과 토론 및 수정작업 진행.
수정 후 몇 차례 정상 주행 성공.
2. DC모터 테스트
주행 중 DC모터 타이어의 일부가 헛도는 현상 발생.
타이어 균형 문제를 해결하기 위해 다양한 시도:
DC모터에 종이를 끼워 균형 맞추기
타이어에 찍찍이 추가 감기
검은 테이프 감기
그러나 근본적인 해결은 되지 않아 추가적인 보완 필요.

개발계획 대비 진행현황

주차	계획 내용	달성 유무
1주차	자율주행과 관련된 정보 조사	달성
2주차	자율주행에 필요한 센서 조사	달성
3주차	자율주행 RC카에서 사용할 센서 선정	달성
4주차	RC카 주행을 위한 선행기술에 대한 추가 정보 조사	달성
5주차	구매한 센서를 MCU 및 Jetson에서 개별 테스트 진행	지연(지연 달성)
6주차	RC카 설계 진행	달성
7주차	트랙 제작 진행	지연(지연 달성)
8주차	자율주행을 위한 주행 알고리즘 개발 및 구현	**지연**
9주차	자율주행을 위한 주행 알고리즘 개발 및 구현	예정
10주차	기상 문제 없이 자율주행 테스트 진행 및 디버깅	〃
11주차	이미지 디헤이징과 센서 융합을 결합	〃
12주차	이미지 디헤이징과 센서 융합을 결합	〃
13주차	기상 악화 상황의 트랙에 대한 자율주행 테스트 진행	〃
14주차	디버깅 및 성능 개선	〃
15주차	최적화 및 자료 정리, 발표 진행	〃

다음주 계획

손주석

카메라의 촬영 영역과 초음파의 영역을 융합해 객체의 위치 추적의 정밀도 높이기
자율 주행 알고리즘 개선 및 라인 트레이싱 문제 해

박준현

주행 알고리즘을 개선

서지완

자율주행 알고리즘 디버깅 중 라인 인식 후 실시간으로 모터 제어를 할 수 있도록 수정할 예정

이준영

Yolov5를 통한 객체 인식을 함께 적용하여 차량을 더욱 원활하게 통제하는 방향을 추가하고, 현재 발생한 문제를 디버깅을 통해 해결하는 방향을 강구.

최창욱

라인트레이서 및 DC모터 추가 보완 및 테스트. 사전 계획했던 ESP32 기반 무선 통신 구조 (ESP32 WiFi Serial Bridge) 실제 구현 및 테스트 진행.

주요 결과물	RC카 완성