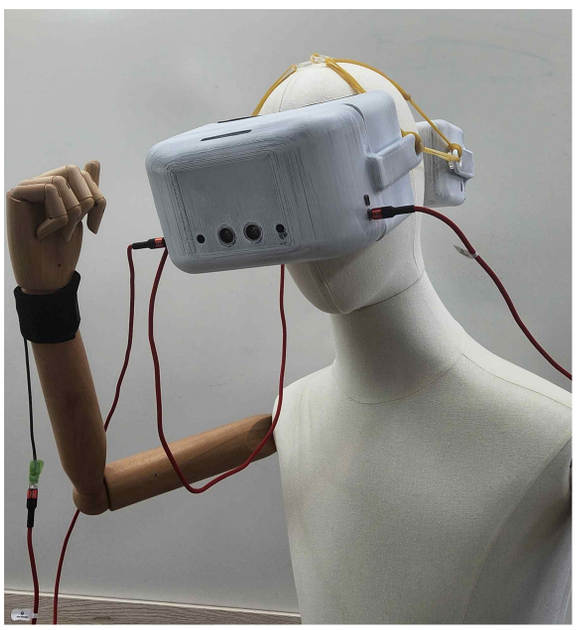
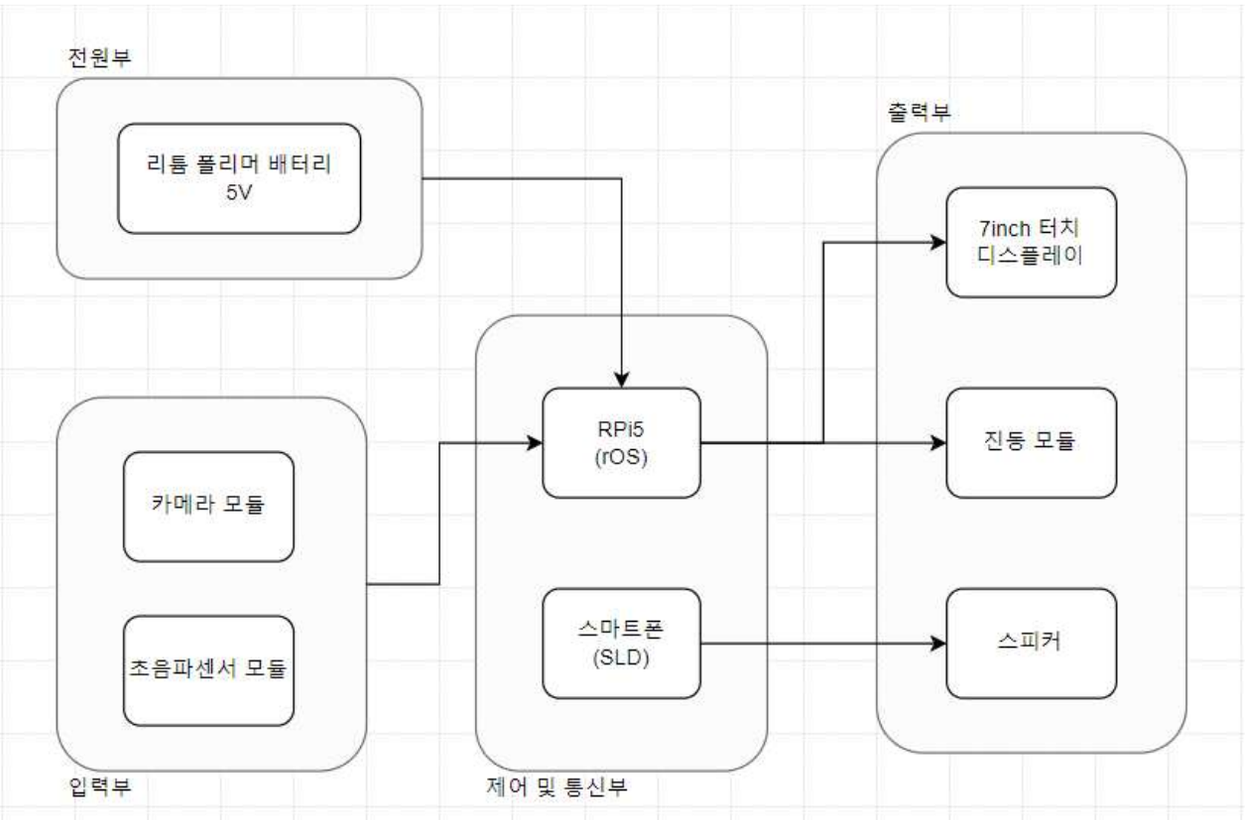
BL (Blind Lumos)



인천전자마이스터고등학교

사람들은 대부분 시각 장애라 하면, 빛을 완전히 인지하지 못하는 전맹을 생각하곤 한다. 하지만 실제로는 저시력이나 단안 실명, 시야각 결손도 시각 장애에 포함이 된다. 따라서, 시각 장애인의 대부분은 전맹이 아니라 빛을 인지하고, 사물의 윤곽은 알아볼 수 있는 시각 장애인의 비중이 훨씬 높다. 하지만 이들은 정확하지 않은 시각 정보에 크게 의존하는 경우가 많기 때문에 오히려 전맹을 가진 시각 장애인보다 더욱 위험한 상황에 처할 수 있다. 우리들은 이러한 문제점을 파악한 후, 시각 장애인들의 안전을 위한 장치(BL)를 고안해보게 되었다. 장치의 기본적인 기능으로는 내부의 각종 모듈을 통해 디스플레이로 사물의 정보를 송출할 수 있도록 하는 것이다. 그리고 사물을 사람의 눈에 잘 띄는 색상(예: 고대비)을 가진 네모 영역으로 이미지를 대체하여 사용자가 세상과 안전하게 상호작용 할 수 있도록 도움을 주게 한다. 또한, 정면에 존재하는 사물이 사용자와 얼마나 가까이 있는지를 알리기 위해 손목 밴드의 진동 모듈을 통하여 사용자에게 직접적으로 거리에 대한 정보를 제공한다. 만약, 사용자가 장치를 잃어버렸을 경우에 대비하여 전용 APP(SLD)과 장치를 블루투스를 통해 서로 연동함으로써 사용자가 장치와 멀리 떨어져 있어도 쉽게 장치를 찾을 수 있도록 하는 기능이 존재한다.



1. 동작 설명
   가. 전원이 공급되고 있는 장치를 착용하여 장치의 동작을 확인한다.
   나. 사용자의 주위에 위험한 사물이 있거나, 사물이 가까이 있는지 살펴본다.
   다. 사용자의 주위에 위험한 사물이 있을 경우, 빨간색으로 강조된 네모 영역 이미지가 사물을 표시하여 보여준다.
   라. 사용자의 주위에 사물이 가까이 있을 경우, 손목 밴드에 탑재되어 있는 진동 모듈이 거리에 따라 진동의 세기가 다르게 동작하여 사용자에게 위험 수준을 알린다.

2. 부가 요소
   가. 장치 내부의 디스플레이를 통해 사물의 네모 영역 이미지를 보고 세상과 상호작용 할 수 있다,
   나. 위험한 상황에 노출되지 않도록 정면에 있는 사물이 사용자와 얼마나 가까이 있는지를 진동 모듈을 통해 알 수 있다.
   다. 장치의 디스플레이에 블루라이트 차단 필터 효과를 적용할 수 있다.
   라. 전용 App과 장치를 연동할 경우에 장치가 사용자와 떨어져 있어도 어디에 있는지 전용 App을 통해 알 수 있다.
   마. 전용 App의 화면을 슬라이딩하여 손쉽게 장치의 색약 보정을 할 수 있다.

3. 전원부

가. 리튬 폴리머 배터리 5V

: 장치 내부의 Raspberry pi 5에 5V의 전원을 공급한다.

4. 프로세서 제어부

가. Raspberry pi 5 (8GB)

: 안드로이드 앱과 블루투스 통신을 하고, Raspberry pi 5와 연결된 카메라 모듈 및 초음파센서로부터 받아온 정보를 7inch 다스플레이에 실시간 송출한다. 또한, 사물을 인식하여 그에 맞는 네모 영역 이미지를 생성해 보여준다.

나. Android 및 iOS 어플리케이션 (SLD)

: 어플리케이션의 화면 속 버튼을 눌렀을 때, Raspberry pi 5와 연결된 스피커를 통해서 소리를 출력한다.

5. 입력부

가. 카메라 모듈

: Raspberry pi 5와 연결하여 사용자가 바라보는 시야 정보를 카메라 센서를 통해 디지털신호로 반환한다.

나. 초음파센서

: Raspberry pi 5와 연결하여 정면에 존재하는 사물과의 거리를 측정한다.

6. 출력부

가. 7inch 터치 디스플레이

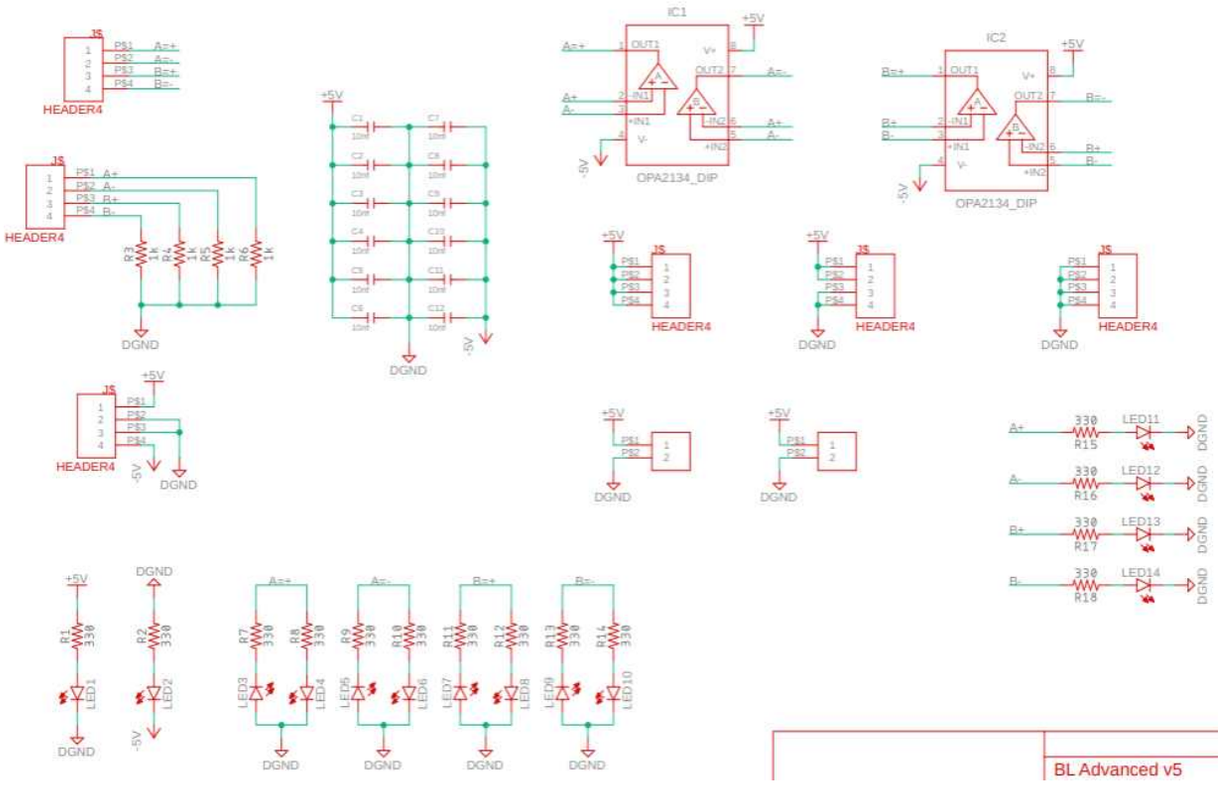
: Raspberry pi 5와 연결된 카메라가 받아온 디지털신호와 사물의 네모 영역 이미지를 출력하여 보여준다.

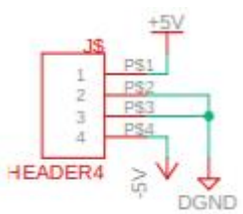
나. 진동 모듈 (Taptic Engine)

: Raspberry pi 5와 연결된 초음파센서에서 받아온 데이터 값이 일정한 수치에 가까워지게 되면 진동 모듈에서 진동이 시작된다, 사물과의 거리가 가까울수록 진동 모듈이 크게 진동하고, 사물과의 거리가 멀수록 진동 모듈이 작게 진동한다.

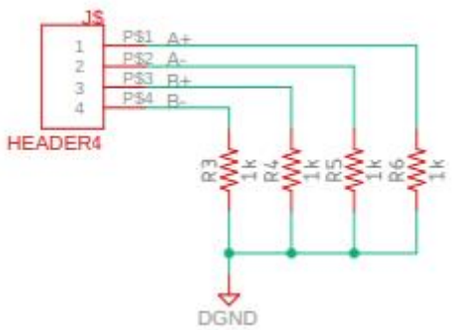
다. 스피커

: Android 및 iOS 어플리케이션(SLD)을 통해 신호를 받으면 스피커에 소리를 재생시킨다.

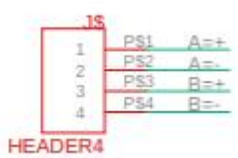




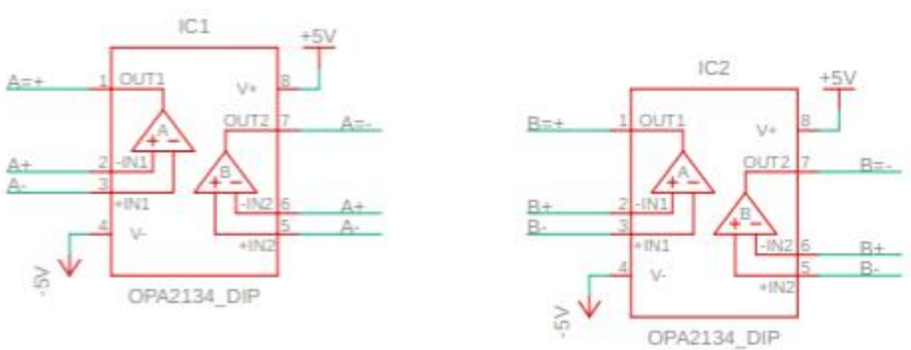
리튬 폴리머 배터리 5V로부터 전원을 공급받는 전원부이다. 이를 통해 각종 프로세서에 필요한 전원을 공급하여 프로세서를 동작시킬 수 있다.



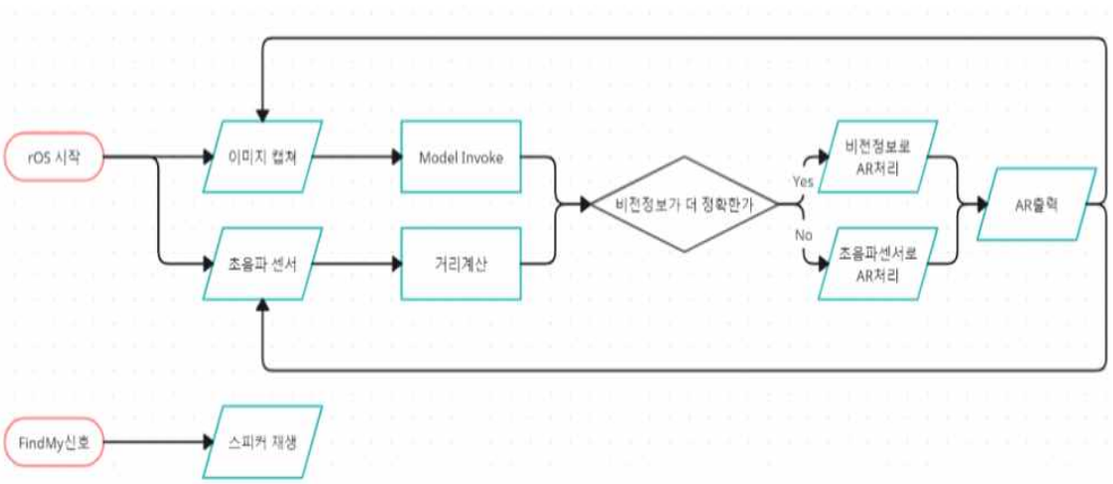
입력을 받는 입력부이다. 진동 모듈이 동작하기 위해 Raspberry pi 5의 PWM 값을 입력받아 OP-Amp로 양과 음의 파형을 동작시키기 위한 회로이다.



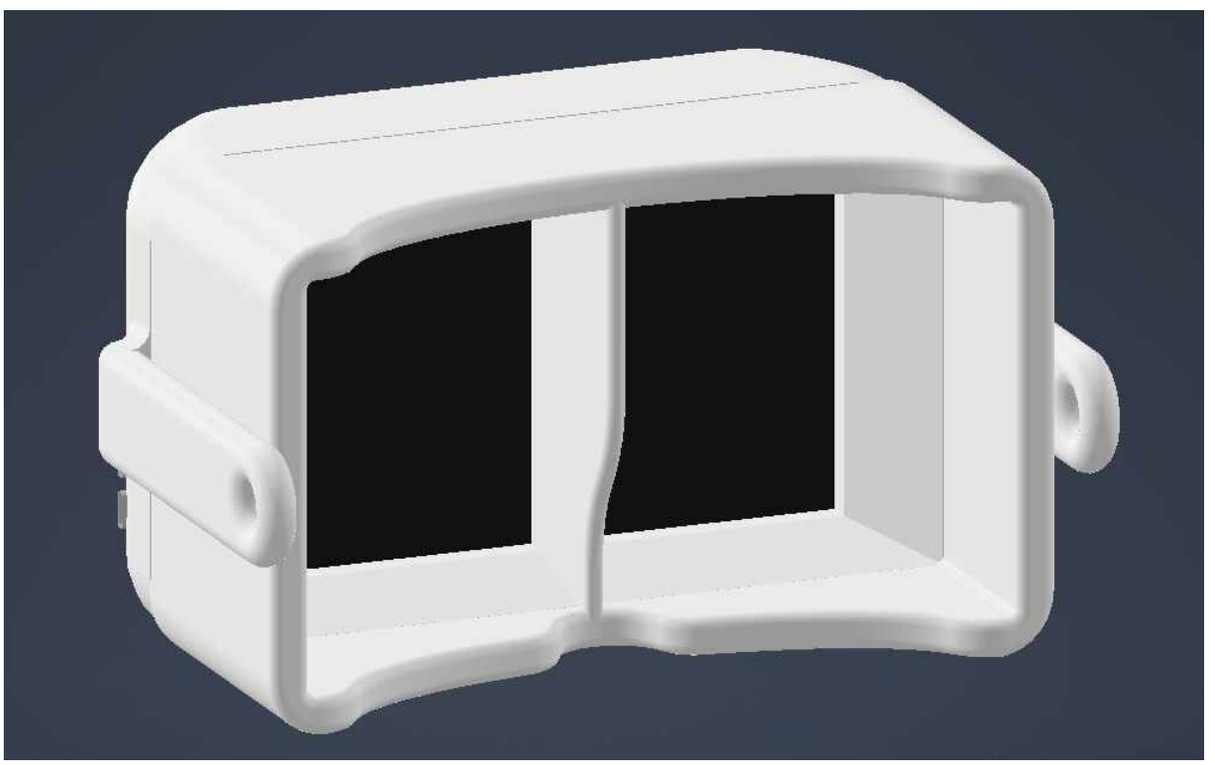
출력이 되는 출력부이다. OP-Amp로 Raspberry pi 5의 PWM 파형이 입력되면 듀티비에 따라 진동 모듈의 진동 모터 속도가 변화하여 진동의 세기를 조절한다.

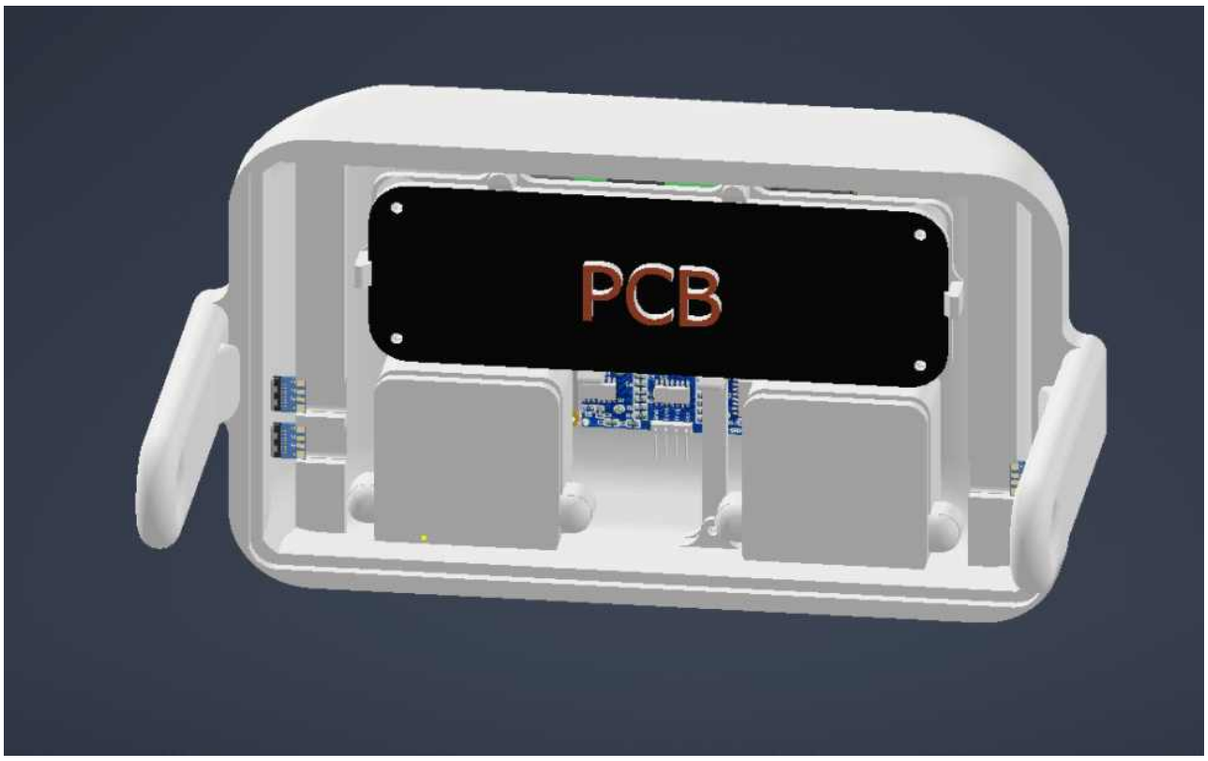


Raspberry pi 5로부터 입력을 받아 진동 모듈의 출력을 제어하는 OP-Amp 회로이다.

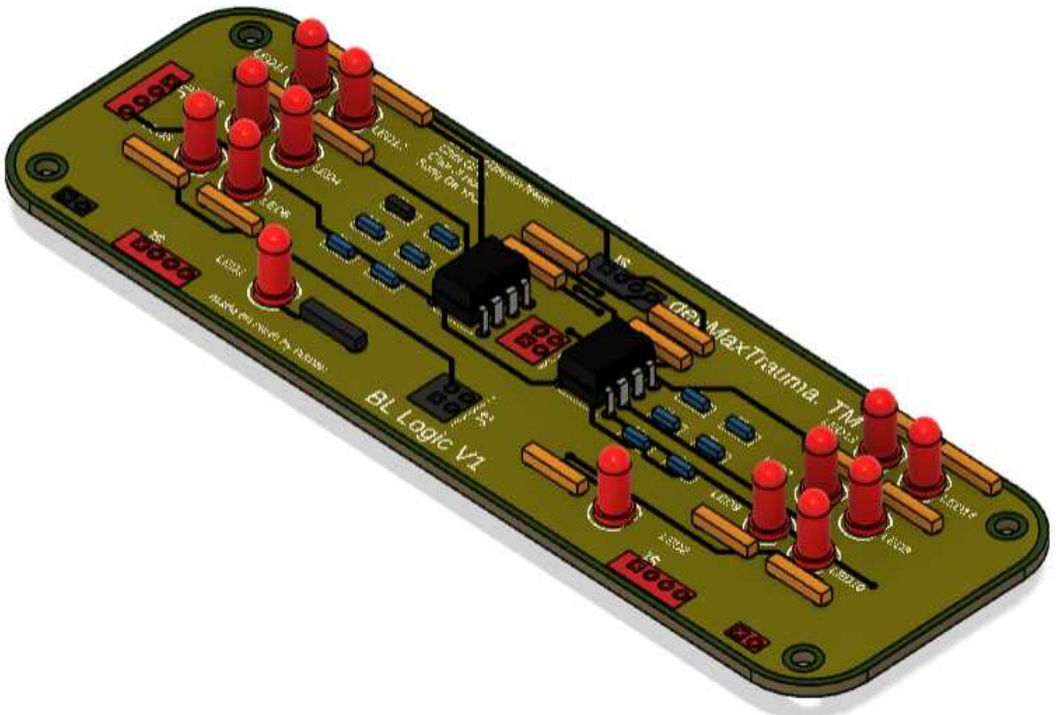


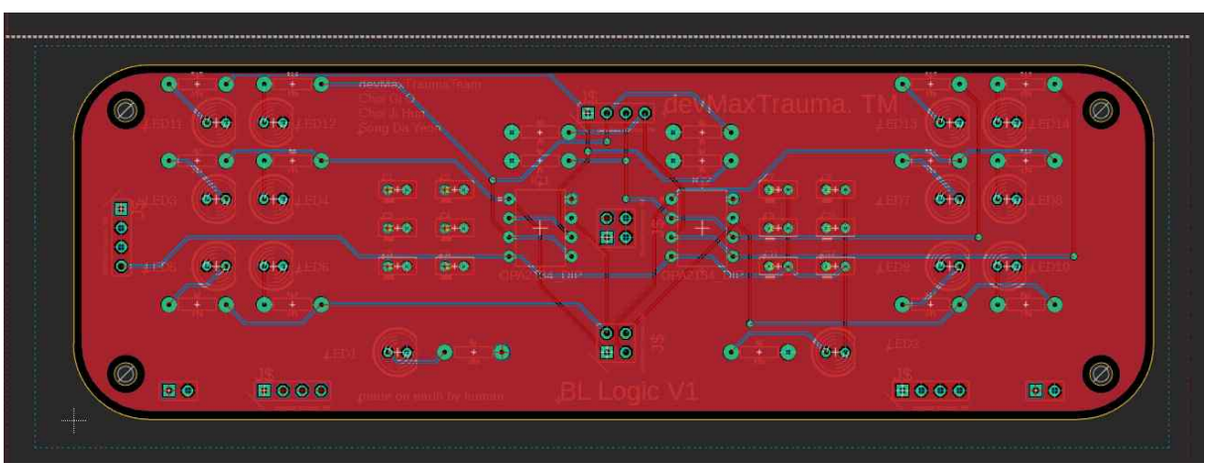
장치에는 Raspberry pi 5가 내장되어 있으며, 카메라 모듈과 초음파센서로부터 데이터를 받아 프로그램의 동작을 수행한다. 또한, 받아온 데이터를 바탕으로 진동 모듈의 동작을 수행한다. Raspberry pi 5로 실행되는 프로그램은 카메라 모듈로 입력받은 데이터를 통해 사물과 사용자 간의 거리를 디스플레이에 표시하고, AI를 이용해 사물을 인식하여 네모 영역의 이미지를 디스플레이에 표시한다. 스마트폰의 장치 전용 App에서는 Raspberry pi 5와 블루투스 연동이 되어있을 때, 화면의 버튼을 누르면 장치 내부의 스피커에서 소리가 재생된다.



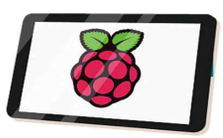


장치의 프레임은 Autodesk 인벤터 프로페셔널 2025 프로그램을 이용해 3D 모델링을 3D 프린터로 출력해 만들었다. 이때, 부품과 부품 사이를 연결할 수 있는 다양한 방법들 중에서 측면의 홈에 딱 들어맞는 고정이 될 수 있도록 3D 모델을 디자인했다.





PCB를 제작한 의도는 진동 모듈에 5V 전원과 -5V 전원을 인가해주기 위해서였다. 하지만 부품의 전원 공급을 위한 것뿐만 아니라 신호의 잡음을 줄이기 위해서 Top면과 Bottom면에 동박을 씌우고, OP-Amp의 주변에 Decoupling Capacitor를 배치해 노이즈 성분을 접지로 내보낼 수 있도록 PCB 기판을 설계하게 되었다.

	부품명	사진	규격	단위	수량
보드	라즈베리파이5 (Raspberry pi 5)		라즈베리파이5 (Raspberry pi 5), 8GB	개	1
	라즈베리파이5 쿨러		액티브 쿨러	개	1
	초음파센서		JS-34738	개	1
	라즈베리파이5 카메라		25mm x 23mm x 9mm	개	1
	라즈베리파이 터치스크린		7inch	개	1
	진동 모듈 (Taptic Engine)		아이폰 7+용 Taptic Engine	개	2
	스피커		V5.2 800mAh 80x38mm	개	1
	리튬 폴리머 배터리		10000mAh	개	5
기구물	장치 프레임		필라멘트, 16.3cm x 23.5cm x 11.9cm	개	1

이 작품은 우리가 처음 계획했었던 사물 인식 기능과 사용자와 사물간의 거리 측정 기능, 그리고 사용자에게 위험도를 알리는 기능을 모두 수행하는 만족스러운 결과를 만들어 냈다.

하지만, 우리는 추가적으로 개발하려고 했었던 가속도센서를 이용한 추락 인지 기능을 구현하지 못했다. 또한, 해당 작품이 우리가 예상했었던 무게를 초과하였기 때문에 착용하는 사람에 따라 장치가 무겁다고 느낄 수 있을 것 같다.

앞으로 우리는 지금까지 개발된 기능들을 조금 더 체계적으로 정리한 다음, 구현해내지 못했던 추락 인지 기능을 개발하여 장치에 추가할 예정이다. 그리고 기능이 추가된 만큼 늘어난 장치의 무게를 줄이기 위한 방법을 찾기 위해 꾸준히 장치를 시험해볼 예정이다.

우리들은 이 장치를 통해서 세상에 존재하는 많은 시각 장애인들이 안전하게 세상과 상호작용 할 수 있었으면 한다.

참고문헌

[1] https://www.apple.com/kr/newsroom/2023/06/introducing-apple-vision-pro/

[2] https://www.meta.com/kr/quest/quest-3/

[3] http://www.kbufac.or.kr/Board/News/Detail?ContentSeq=2908&Page=19

1. 프로젝트 수업 계획서

월	주	일	단 원	주요업무	비 고	
					(작성되어야 할 내용, 제출평가)	확인 (지도교사)
3	2	3/4 ~ 3/8	프로젝트 구성안	제품명/디자인 및 동작 계획		
	3	3/11 ~ 3/15	프로젝트 구성안	기능 구상 및 역할 분담		
	4	3/18 ~ 3/22	부품 발주	프로젝트 필요 부품 구상 및 발주 신청		
	5	3/25 ~ 3/29	소프트웨어	플로우차트 작성		
4	1	4/1 ~ 4/5	소프트웨어	기본 소프트웨어 구현		
	2	4/8 ~ 4/12	관련 학습	앱 기능 구현 방법 학습		
	3	4/15 ~ 4/19	부품 발주	필요 부품 발주 추가 신청		
	4	4/22 ~ 4/26	소프트웨어	기본 소프트웨어 구현		
	5	4/29 ~ 4/30	하드웨어	하드웨어 3D 모델링 제작		
5	1	5/1 ~ 5/3	소프트웨어	기본 소프트웨어 구현		
	2	5/7 ~ 5/10	소프트웨어	심화 소프트웨어 구현		
	3	5/13 ~ 5/17	소프트웨어	심화 소프트웨어 구현		
	4	5/20 ~ 5/24	소프트웨어	앱 기능 구현		
	5	5/27 ~ 5/31	소프트웨어	앱 기능 구현		
6	2	6/3 ~ 6/7	부품 발주	필요 부품 발주 신청		
	3	6/10 ~ 6/14	하드웨어	하드웨어 블록도 작성		
	4	6/17 ~ 6/21	소프트웨어	심화 소프트웨어 구현		
	5	6/24 ~ 6/28	하드웨어	기본 하드웨어 구성		

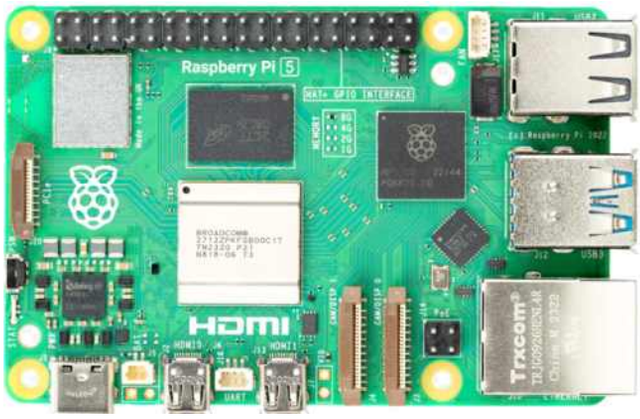
월	주	일	단 원	주요업무	비 고	
					(작성되어야 할 내용, 제출평가)	확인 (지도교사)
7	1	7/1 ~ 7/2	하드웨어	기본 하드웨어 구성		
	2	7/8 ~ 7/12	하드웨어	심화 하드웨어 구성		
	3	7/15 ~ 7/19	소프트웨어	심화 소프트웨어 구현		
	4	7/22 ~ 7/26	소프트웨어	심화 소프트웨어 구현		
	5	7/29 ~ 7/31	소프트웨어	심화 소프트웨어 구현		
8	1	8/1 ~ 8/2	하드웨어	심화 하드웨어 구성		
	2	8/5 ~ 8/9	하드웨어	심화 하드웨어 구성		
	3	8/12 ~ 8/16	소프트웨어	소프트웨어 구동 확인 및 수정		
	4	8/19 ~ 8/23	하드웨어	기구물과 하드웨어 결합		
	5	8/26 ~ 8/30	하드웨어	하드웨어 테스트 및 수정		
9	1	9/2 ~ 9/6	성능 구현	소프트웨어 시연 및 수정		
	2	9/9 ~ 9/13	성능 구현	소프트웨어 시연 및 수정		
	3	9/16 ~ 9/20	성능 구현	소프트웨어 시연 및 수정		
	4	9/23 ~ 9/27	성능 구현	소프트웨어 시연 및 수정		

2. 과제 수행 진도표

내용 \ 일정		3월 1~3주	3월 4~5주	4월 1~2주	4월 3~5주	5월 1~2주	5월 3~5주	6월 1~2주	6월 3~4주	7월 1~2주
하드웨어	초안 작성									
	회로 설계									
	부품 발주									
	블럭도									
	테스트보드									
	보고서									
소프트웨어	초안 작성									
	플로우차트									
	소프트웨어 구현									
	학습									
	보고서									
시스템	초안 작성									
	기구물 디자인									
	조립 및 시현									
	보고서									

내용 \ 일정		7월 3~5주	8월 1~3주	8월 4~5주	9월 1~2주	9월 3~5주	10월 1~2주	10월 3~5주	11월 1~2주	11월 3~5주
하드웨어	회로설계									
	시뮬레이션									
	구조물제작									
	보고서									
소프트웨어	소프트웨어 구현									
	학습									
	보고서									
시스템	조립 및 시현									
	보고서									

가. 라즈베리파이5 (Raspberry pi 5)



라즈베리 파이는 영국 잉글랜드의 라즈베리 파이 재단이 학교와 개발도상국에서 기초 컴퓨터 과학의 교육을 증진시키기 위해 개발한 신용카드 크기의 싱글 보드 컴퓨터이다. 초기의 라즈베리 파이는 엘레멘트14/프리미어 파넬, RS 콤포넌트와의 허가된 제조 협정을 통해 제작되었다.

Raspberry pi 5는 이전 세대에 비해 브로드컴 BCM2712 쿼드코어 Arm Cortex-A76 프로세서를 탑재하여 2-3배 이상 빨라지고 가능한 최고의 성능을 위해 라즈베리파이 재단에서 자체적으로 설계된 반도체가 포함된 패키지인 RP1 I/O 컨트롤러를 내장하여 지금까지의 라즈베리파이와는 차원이 다른 경험을 제공한다. USB 3는 더 빠른 전송 속도를 위해 더 많은 총 대역폭을 제공하고, 카메라와 DSI 디스플레이 커넥터는 서로 바꿔 사용하거나 각각 하나씩 사용하는 등 두 개를 동일하게 사용할 수 있다. 우리가 사용한 Raspberry pi 5의 규격은 2.4GHz ARM 프로세서, 8GB의 RAM이다. 작품에서는 AR 플랫폼을 제공하고 전용 APP과 소켓 통신을 하여 각종 부품들을 제어한다.

나. 진동 모듈 (Taptic Engine)

Apple에서 개발한 진동 모터를 이용한 엔진이다. 일반적인 진동 부품보다 내구도가 강하며, 더 섬세하게 작동한다고 Apple은 소개하고 있다. 기존 진동 모터와 같은 느낌을 내는 것도 당연히 가능하며, 일반적인 진동 모터와는 다르게 버튼을 누르는 듯한 매우 짧고 경쾌한 진동을 울리게 하는 것도 가능하다는 특징이 있다. Apple은 Taptic Engine을 이용하여 3D Touch 피드백이나 Apple Watch의 모든 진동 피드백을 구현했다. 이는 구조상 Taptic Engine에는 최대 크기의 진동까지 도달하는데 거의 딜레이가 존재하지 않기 때문에 가능한 것이지만, Taptic Engine은 단방향으로만 직선운동을 하는 리니어 모터란 특성 때문에 바닥에 두었을 때에는 진동이 울리는 것을 알아채기 힘들다는 것이 단점으로 꼽힌다. 작품에서는 초음파센서가 받아온 값에 따라 사람에게 위험도를 알리는 용도로 사용된다.

다. 카메라 모듈

카메라모듈은 렌즈를 통해 들어온 이미지를 센서를 통해 디지털신호로 변환시키는 부품이다. 카메라 모듈은 크게 이미지 센서와 렌즈 모듈, IR-Filter, Package 등으로 구성된다. 이미지 센서는 CCD(Charge Coupled Device)센서와 CMOS(Complementary Metal Oxide Semi-conductor)로 분류 된다. CCD센서는 일반적으로 CMOS보다 질 좋은 화질을 구현할 수 있다는 장점이 있으며, 단점은 가격이 비싸고 전력의 소모량이 크다는 것이 있다. CMOS는 양산성이 우수해서 가격이 저렴하지만, 단점으로는 상대적으로 화질이 떨어지고 노이즈가 발생하며 사이즈가 크다는 것이 있다. 렌즈 모듈(Lens Module)은 구면은 유리로 만들고, 비 구면은 플라스틱 사출로 만든다. 작품에서 카메라 모듈은 사용자가 바라보는 시각 정보를 받아오기 때문에 다른 부품들 보다도 매우 중요한 역할을 한다.

[Android App(SLD) 코드]

// 메인코드 //

```
package com.example.mdp_project;

import static android.speech.tts.TextToSpeech.ERROR;
import android.Manifest;
import android.annotation.SuppressLint;
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.bluetooth.BluetoothSocket;
import android.content.Context;
import android.content.pm.PackageManager;
import android.os.Bundle;
import android.os.Vibrator;
import android.speech.tts.TextToSpeech;
import android.util.Log;
import androidx.appcompat.app.AppCompatActivity;
import androidx.core.app.ActivityCompat;
import androidx.fragment.app.Fragment;
import androidx.viewpager2.widget.ViewPager2;
import com.tbuonomo.viewpagerdotsindicator.DotsIndicator;
import java.io.IOException;
import java.io.OutputStream;
import java.lang.reflect.Method;
import java.util.ArrayList;
import java.util.Locale;
import java.util.UUID;

public class MainActivity extends AppCompatActivity {
    private static final int SPEECH_REQUEST_CODE = 0;
    PageAdapter pageAdapter;
    ViewPager2 viewPager2;
    ArrayList<Fragment> fragList = new ArrayList<Fragment>();
    private TextToSpeech tts;

    private static final String DEVICE_ADDRESS = "2C:CF:67:28:A6:D6";
    // 라즈베리 파이 블루투스 주소
    private static final UUID MY_UUID = UUID.fromString("00001101-0000-1000-8000-00805F9B
34FB");

    private static BluetoothAdapter bluetoothAdapter;
    private static BluetoothSocket bluetoothSocket;
    private static OutputStream outputStream;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        bluetoothAdapter = BluetoothAdapter.getDefaultAdapter();
        BluetoothDevice device = bluetoothAdapter.getRemoteDevice(DEVICE_ADDRESS);
        tts = new TextToSpeech(this, new TextToSpeech.OnInitListener() {
            @Override
            public void onInit(int status) {
                if (status != ERROR) {
                    // 언어를 선택한다.
                    tts.setLanguage(Locale.KOREAN);
```

```
                    tts.setPitch(1.0f);           // 음성 톤은 기본 설정
                    tts.setSpeechRate(2.0f);     //음성 속도 기본 설정
                    tts.speak("기본 화면 입니다.", TextToSpeech.QUEUE_ADD, null);
                }
            }
        });
        new Thread(new Runnable() {
            public void run() {
                try {
                    bluetoothSocket = createBluetoothSocket(device);
                    if (ActivityCompat.checkSelfPermission(MainActivity.this, Manifest.permission.
  BLUETOOTH_CONNECT) != PackageManager.PERMISSION_GRANTED) {
                        String[] permissions =
  new String[]{Manifest.permission.BLUETOOTH_CONNECT};
                        // TODO: Consider calling
                        //    ActivityCompat#requestPermissions
                        // here to request the missing permissions, and then overriding
                        //     public void onRequestPermissionsResult(int requestCode, String[]
permissions,
                        //                                            int[] grantResults)
                        // to handle the case where the user grants the permission. See the
documentation
                        // for ActivityCompat#requestPermissions for more details.
                        ActivityCompat.requestPermissions(MainActivity.this, permissions, 01);
                    }
                    bluetoothSocket.connect();
                    tts.speak("연결되었습니다", TextToSpeech.QUEUE_ADD, null);
                    outputStream = bluetoothSocket.getOutputStream();
                    Log.d("MainActivity", "Bluetooth connected");
                } catch (IOException e) {
                    Log.e("MainActivity", "Error connecting to Bluetooth", e);
                }
            }
        }).start();

  ~ 중략 ~

    @SuppressLint("MissingPermission")
    public static BluetoothSocket createBluetoothSocket(BluetoothDevice device) throws
IOException {
        try {
            final Method m = device.getClass().getMethod("createRfcommSocket", new
Class[]{int.class});
            return (BluetoothSocket) m.invoke(device, 1);
        } catch (Exception e) {
            Log.e("MainActivity", "Could not create Insecure RFComm Connection", e);
        }
        return device.createRfcommSocketToServiceRecord(MY_UUID);
    }

    public static void sendSignal(String signal) {
        try {
            outputStream.write(signal.getBytes());
            Log.d("MainActivity", "Signal sent: " + signal);
        } catch (IOException e) {
            Log.e("MainActivity", "Error sending signal", e);
        }
    }
}

~ 이하 생략 ~
```

[Rapsberry pi 5(rOS) 코드]

```
// 메인코드 //
import numpy as np
import RKernel

rk = RKernel.RKernel()
fps_history = []

# resize window 360 360
rk.cv.namedWindow("ROS", rk.cv.WINDOW_NORMAL)
if rk.key_engine.get_key("ROSARDisplayEnabled").get("value"):
    ar_width = rk.key_engine.get_key("ARDisplayWidth").get("value")
    ar_height = rk.key_engine.get_key("ARDisplayHeight").get("value")
    rk.cv.resizeWindow("ROS", ar_width, ar_height)
else:
    rk.cv.resizeWindow("ROS", 320, 320)

boot_start_time = rk.time.time()
while rk.time.time() - boot_start_time < 5:
    if rk.key_engine.get_key("ROSARDisplayEnabled").get("value"):
        rk.cv.imshow("ROS", rk.make_ar_frame(rk.splash_screen))
    else:
        rk.cv.imshow("ROS", rk.splash_screen)
    rk.cv.waitKey(1)

if rk.key_engine.get_key("CameraDevice").get("value") == "macbook pro":
    camera = rk.cv.VideoCapture(0)
elif rk.key_engine.get_key("CameraDevice").get("value") == "iphone":
    camera = rk.cv.VideoCapture(1)
elif rk.key_engine.get_key("CameraDevice").get("value") == "raspberry pi":
    camera = rk.picamera2.Picamera2()
    camera.configure(camera.create_preview_configuration(main={"size": (320, 320)}))
    camera.start()

else:
    print("warning! Invalid camera device. Using default device.")
    camera = rk.cv.VideoCapture(0)

if camera is None:
    print("camera open failed")
    rk.shutdown()

last_update_time = rk.time.time()
while True:
    if rk.key_engine.get_key("CameraDevice").get("value") == "raspberry pi":
```

```
            frame = camera.capture_array()
            frame = rk.cv.cvtColor(frame, rk.cv.COLOR_BGR2RGB)
            if frame is None:
                print("raspberry pi camera read failed")
                break
        else:
            capture_success, frame = camera.read()
            if not capture_success:
                print("camera read failed")
                break

        if frame.shape[0] != 320 or frame.shape[1] != 320:
            # make new_frame but don't flect it
            target_width = 320
            target_height = 320
            aspect_ratio = float(target_height) / frame.shape[0]
            dsize = (int(frame.shape[1] * aspect_ratio), target_height)
            new_frame = rk.cv.resize(frame, dsize)

            # cut the new_frame width to 320 center
            new_frame = new_frame[:,
                        new_frame.shape[1] // 2 - target_width // 2: new_frame.shape[1] // 2 +
target_width // 2]
        else:
            new_frame = frame

        if rk.key_engine.get_key("ROSModelActive").get("value"):
            rk.process_frame(new_frame)

            boxes_idx, classes_idx, scores_idx = 0, 1, 2
            boxes = rk.model.get_tensor(rk.model_output_details[boxes_idx]['index'])[0]
            classes = rk.model.get_tensor(rk.model_output_details[classes_idx]['index'])[0]
            scores = rk.model.get_tensor(rk.model_output_details[scores_idx]['index'])[0]

            for i in range(len(scores)):
                class_name = rk.label_engine.get_label(int(classes[i] + 1)).get("value")
                class_color = rk.color_engine.get_color(class_name)
                if scores[i] > 0.5:
                    box = boxes[i] * [320, 320, 320, 320]
                    text_x, text_y = 0, 0  # for text position
                    if rk.key_engine.get_key("ROSMindDisplayWay").get("value") == "filled box":
                        inverted_color = (255 - class_color[0], 255 - class_color[1], 255 -
class_color[2])
                        # outer line
                        rk.cv.rectangle(new_frame, (int(box[1]), int(box[0]), int(box[3]), int(box[2])),
inverted_color, 2)
```

```
                    # inner box
                    rk.cv.rectangle(new_frame, (int(box[1]), int(box[0])), (int(box[3]), int(box[2])),
class_color, -1)
                    # put text center of the box
                    text_size = rk.cv.getTextSize(class_name, rk.cv.FONT_HERSHEY_SIMPLEX,
0.5, 2)[0]
                    text_x = int((box[1] + box[3]) / 2 - text_size[0] / 2)
                    text_y = int((box[0] + box[2]) / 2 + text_size[1] / 2)
                    rk.cv.putText(new_frame, class_name, (text_x, text_y),
rk.cv.FONT_HERSHEY_SIMPLEX, 0.5,
                                   inverted_color, 2)

                elif rk.key_engine.get_key("ROSMindDisplayWay").get("value") == "outlined box":
                    rk.cv.rectangle(new_frame, (int(box[1]), int(box[0])), (int(box[3]), int(box[2])),
class_color, 2)
                    text_x = int(box[1])
                    text_y = int(box[0])
                    rk.cv.putText(new_frame, class_name, (text_x, text_y),
rk.cv.FONT_HERSHEY_SIMPLEX, 0.5,
                                   class_color, 2)

                if rk.key_engine.get_key("DistanceDisplayEnabled").get("value"):
                    object_average_width = rk.label_engine.get_label(int(classes[i] +
1)).get("average_width")
                    box_width_pixel = box[3] - box[1]
                    distance = rk.calculate_distance(object_average_width, box_width_pixel)
                    if rk.key_engine.get_key("DistanceDisplayWay").get("value") ==
"invertedColorInBox":
                        inverted_color = (255 - class_color[0], 255 - class_color[1], 255 -
class_color[2])
                        # draw text in center of the box
                        rk.cv.putText(new_frame, distance, (text_x, text_y + 20),
rk.cv.FONT_HERSHEY_SIMPLEX, 0.5,
                                      inverted_color, 2)

        if rk.key_engine.get_key("ROSDisplayFPSEnable").get("value"):
            update_time = rk.time.time()
            fps = 1 / (update_time - last_update_time)
            fps_history.append(fps)
            if len(fps_history) > 10:
                fps_history.pop(0)
            last_update_time = update_time
            avg_fps = np.mean(fps_history)
            rk.cv.putText(new_frame, f"FPS: {avg_fps:.2f}", (10, 20), rk.cv.FONT_HERSHEY_SIMPLEX,
0.5, (0, 255, 0), 2)
```

```
    if rk.key_engine.get_key("ROSARDisplayEnabled").get("value"):
        new_frame = rk.make_ar_frame(new_frame)

    rk.cv.imshow("ROS", new_frame)

    if rk.cv.waitKey(1) & 0xFF == 27:
        break

rk.shutdown()

// RKernel 코드 //

from boot.RTensor import model

print("RKernel is booting up...")

print("defining pre def methods...")


def make_error(error_code: str, error_message: str):
    pass
    print("E" + error_code + ": " + error_message)
    exit(error_code)


print("methods pre def defined.")

print("Loading Third Party imports...")
try:
    pass
    import cv2 as cv
except ImportError:
    pass
    make_error("1001", "cv2 not found.")
except Exception as e:
    pass
    make_error("1001-1", str(e))
try:
    pass
    import time
except ImportError:
    pass
    make_error("1002", "time not found.")
except Exception as e:
    pass
    make_error("1002-1", str(e))
```

```
try:
    pass
    import picamera2
except ImportError:
    pass
    print("Picamera2 not found.")
except Exception as e:
    pass
    make_error("1003", str(e))
try:
    pass
    import threading
except ImportError:
    pass
    make_error("1005", "threading not found.")
except Exception as e:
    pass
    make_error("1005-1", str(e))
try:
    pass
    import numpy as np
except ImportError:
    pass
    make_error("1006", "numpy not found.")
except Exception as e:
    pass
    make_error("1006-1", str(e))
try:
    pass
    import sys
except ImportError:
    pass
    make_error("1007", "sys not found.")
except Exception as e:
    pass
    make_error("1007-1", str(e))
try:
    pass
    import os
except ImportError:
    pass
    make_error("1008", "os not found.")
except Exception as e:
    pass
    make_error("1008-1", str(e))
print("Third Party Imports loaded.")
```

```
print("Loading RKernel imports...")
try:
    pass
    import boot.RKey as key_engine
except ImportError:
    pass
    make_error("1101", "RKey not found.")
except Exception as e:
    pass
    make_error("1101-1", str(e))
try:
    pass
    import boot.RSound as sound_engine
except ImportError:
    pass
    make_error("1102", "RSound not found.")
except Exception as e:
    pass
    make_error("1102-1", str(e))
try:
    pass
    import boot.RFPS as fps_engine
except ImportError:
    pass
    make_error("1103", "RFPS not found.")
except Exception as e:
    pass
    make_error("1103-1", str(e))
try:
    pass
    import boot.RTensor as tensor_engine
except ImportError:
    pass
    make_error("1104", "RTensor not found.")
except Exception as e:
    pass
    make_error("1104-1", str(e))
try:
    pass
    import boot.RLabel as label_engine
except ImportError:
    pass
    make_error("1105", "RLabel not found.")
except Exception as e:
    pass
```

```
    make_error("1105-1", str(e))
try:
    pass
    import boot.RColor as color_engine
except ImportError:
    pass
    make_error("1106", "RColor not found.")
except Exception as e:
    pass
    make_error("1106-1", str(e))
try:
    pass
    import boot.RBluetooth as bluetooth_engine
except ImportError as e:
    pass
    print("RBluetooth not found.")
except Exception as e:
    pass
    make_error("1107", str(e))
try:
    pass
    import boot.RNotification as notification_engine
except ImportError:
    pass
    make_error("1108", "RNotification not found.")
except Exception as e:
    pass
    make_error("1108-1", str(e))
try:
    pass
    import boot.RTTS as tts_engine
except ImportError:
    pass
    make_error("1109", "RTTS not found.")
except Exception as e:
    pass
    make_error("1109-1", str(e))
# try:
#     import boot.RGPIO as gpio_engine
# except ImportError:
#     print("RGPIO not found.")
# try:
#     if "boot.RGPIO" in sys.modules: import boot.RUSS as ultrasonic_engine
# except ImportError:
#     make_error("1111", "RUSS not found.")
# try:
```

```
#       if "boot.RGPIO" in sys.modules: import boot.RTaptic as taptic_engine
# except ImportError:
#       make_error("1112", "RTaptic not found.")

~ 중략 ~

# set_tensor_input()
if key_engine.get_key("ROSBootChimeEnabled").get("value"):
    pass
    sound_engine.play("boot/res/StartUp.mp3")

started_time = time.time()
splash_display_time = key_engine.get_key("ROSSplashScreenTime").get("value")
while time.time() - started_time < splash_display_time:
    pass
    boot_logo(time.time() - started_time, splash_display_time)
    tick_screen()
    # debug
    # if hard_warning_icon is None:
    #       kernel_panicked = True
    kernel_panicked = kernel_panic_check()
    if cv.waitKey(1) & 0xFF == key_engine.get_key("ROSOffKey").get("value"):
        pass
        shutdown()

tts_engine.sound_engine = sound_engine
notification_engine.sound_engine = sound_engine

~ 이하 생략 ~
```

보다 더 자세한 코드 내용은 https://github.com/devMaxTrauma을 통해 확인해주시길 바랍니다.

<최지오>

이번 프로젝트를 통해 다양한 기술을 실제로 적용해보면서 많은 것을 배울 수 있었습니다. 이번 프로젝트에서도 Inventor를 활용해 3D 모델링을 진행했습니다. 이전에도 3D 모델링을 해본 경험이 있었지만, 이번에는 더 복잡하고 정교한 모델을 설계하면서 기술적인 깊이를 더할 수 있었습니다. 모델링 과정에서의 디테일한 수정과 최적화를 통해 설계의 완성도를 높이는 법을 익혔습니다. Eagle을 사용해서는 PCB 설계를 했는데, 실제 회로를 설계하고 시뮬레이션해 볼 수 있어 전자공학의 기본 원리를 더 깊이 이해할 수 있었습니다. 설계한 회로가 실제로 작동하는 것을 보며 설계의 중요성을 다시 한번 실감하게 되었어요. Python과 OpenCV를 활용해 이미지 처리와 컴퓨터 비전을 구현했습니다. 이미지를 분석하고 처리하는 과정에서 OpenCV의 다양한 기능을 활용할 수 있었고, 특히 실시간 데이터 처리의 가능성을 엿볼 수 있었던 부분이 인상적이었습니다. 또한, TensorFlow Lite를 사용해 모델을 경량화하고 모바일 환경에서도 딥러닝 모델을 구현하는 작업도 진행했어요. 모델 최적화 과정에서 성능과 정확도 사이의 균형을 맞추는 것이 쉽지 않았지만, 이를 통해 모바일 환경에서도 AI를 적용할 수 있는 가능성을 확인할 수 있었습니다. 이번 프로젝트는 다양한 기술을 실무에 적용해보는 좋은 기회였으며, 특히 각 기술의 강점을 이해하고 어떻게 활용할지에 대한 감각을 기를 수 있었습니다. 도전과 성장이 공존했던 프로젝트였고, 이러한 경험들이 앞으로의 학습과 실무에 큰 도움이 될 것이라 믿습니다.

<송다연>

MDP 발표회를 준비하던 중간에 취업교육을 듣게 되면서 한동안 학교를 나오지 못했던 적이 있었습니다. 하지만 취업교육이 끝나 다시 학교로 돌아왔을 때, 저희 팀원들이 자신들이 맡았던 소프트웨어 개발의 많은 부분을 완성해주어서 저 또한 의욕을 잃지 않고 MDP에 열심히 참여할 수 있었던 것 같습니다. 이 과정에서 팀에 생긴 저의 빈 자리를 메꾸기 위해 저는 팀원들과 종종 소통을 하고, 팀원들이 저의 도움을 필요로 하면 내키지 않고 바로 도움을 주었던 기억이 있습니다. 이 덕분에 저희 팀원들 또한 팀원의 공백이 있었음에도 불구하고 끝까지 맡은 바를 완벽히 해내어준 것 같습니다. 그리고 팀원들과 큰 마찰없이 기분 좋게 MDP 발표회를 마무리할 수 있게 되어 굉장히 기분이 좋습니다. 다음에도 공동 프로젝트에 참여할 기회가 생긴다면 소통을 최우선으로 팀원들과 함께 즐거운 마음으로 프로젝트에 참여할 것 같습니다. 감사합니다.

<최지훈>

MDP 발표회에서 앱 개발을 담당하면서 많은 것을 배웠습니다. 처음부터 끝까지 앱 개발의 과정을 직접 경험하면서 기술적인 성장뿐만 아니라 팀과의 소통의 중요성을 다시금 느낄 수 있었습니다. 전기전자 전공의 특성상 다양한 하드웨어와 소프트웨어가 결합된 프로젝트였는데, 그 속에서 앱이 어떻게 전체 시스템과 유기적으로 작동하는지 이해하는 데 많은 시간이 걸렸습니다. 하지만 하나씩 이해 해가며 문제점을 해결하고 무사히 발표회를 하게 되어 뿌듯합니다.