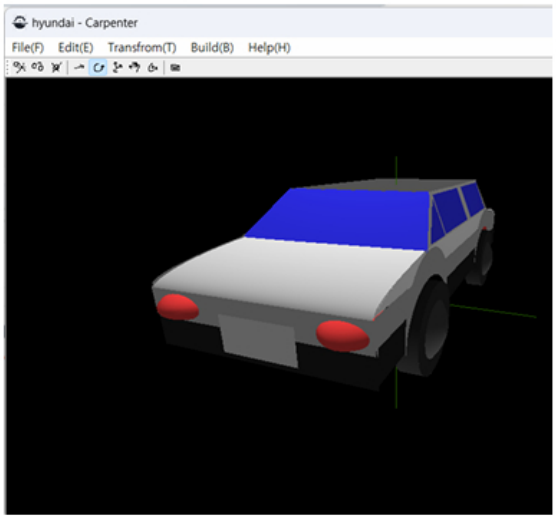
카펜터 (Carpenter) : 3D 모델 설계 소프트웨어

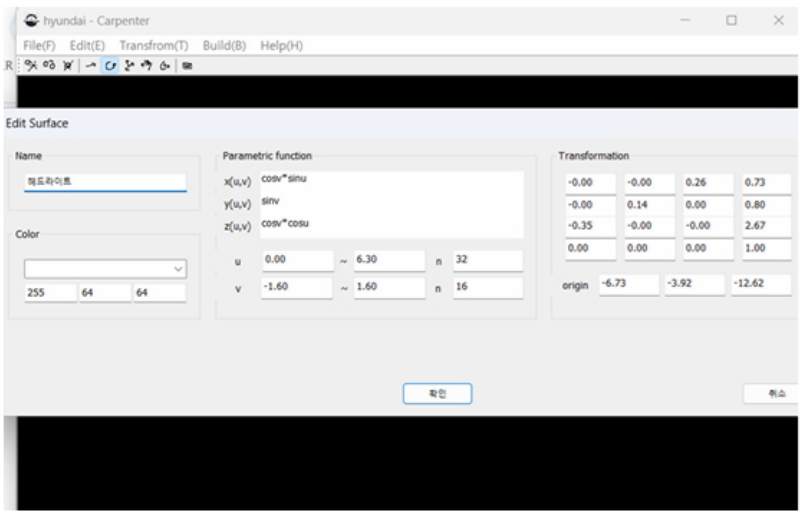
C++/MFC Desktop Application

Introduction

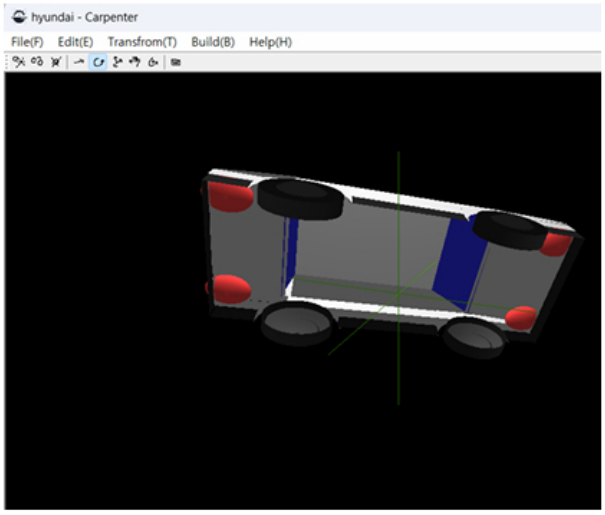
3D 모델 설계 (3D Model Design)



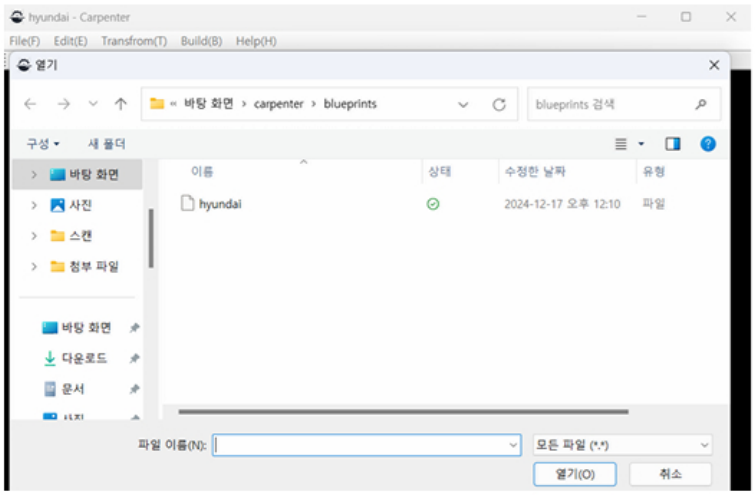
매개변수 곡면(Parametric Surfaces)



아핀 변환 (Affine Transformation)



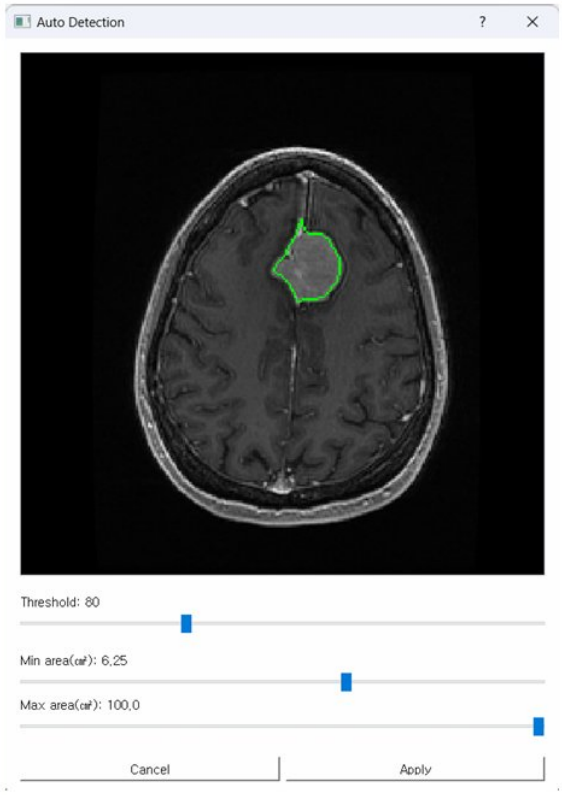
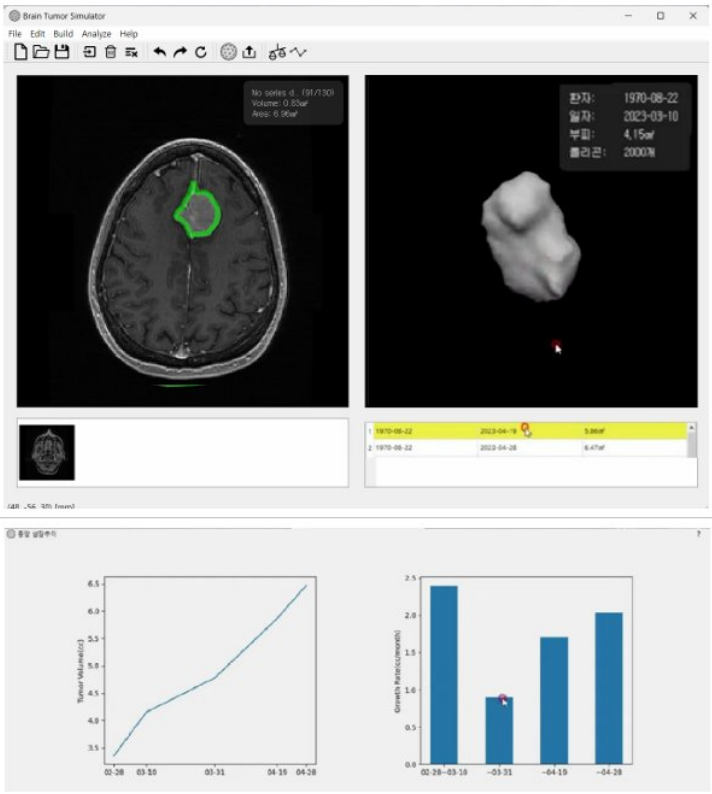
데이터 보존(Serialization)



응용 - 레이싱 게임 제작에 위 모델 사용 (Applied case)

뇌종양 분석기 (Brain Tumor Simulator)

Python/QT Application

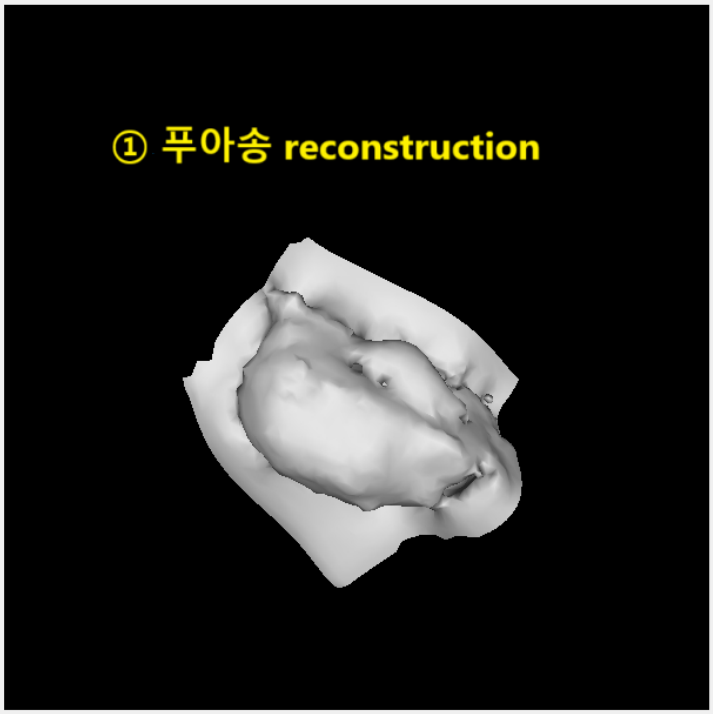
종양 컨투어 탐지 (Contour Detection)	종양 3D 재구성 (Poission Reconstruction)
	

Binarization, area threshold custumization

```python
def detectPixmapContour(pixmap_src: QPixmap, threshold: int, area_range: tuple[float]) -> Contou
    """
    주어진 pixmap에 contour detection을 적용한 결과(ContourDetectionResult)를 리턴합니다.

    Args:
        pixmap_src (QPixmap): contour detection을 적용할 pixmap
        threshold (int): binarization threshold 값. 0~255
        area_range (tuple[float]): 면적이 area_range[0], area_range[1] 사이에 있는 contour만 취한다

    Returns:
        ContourDetectionResult: detection 결과를 담은 데이터 구조체
    """
    # 원본 pixmap을 전처리하여 threshold를 기준으로 binary 이미지화 한다
    image = pixmapToCv2Image(pixmap_src)
    grayed = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    blurred = cv2.GaussianBlur(grayed, (11, 11), 0)
    _, binary_image = cv2.threshold(blurred, threshold, 255, cv2.THRESH_BINARY)

    # binary 이미지로부터 contours 를 계산한다
    contours, hierarchy = cv2.findContours(binary_image, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)

    # min_area보다 면적이 작은 noise contours를 걸러낸다
    contours_filtered = []
    for contour in contours:
        area = cv2.contourArea(contour)
        if (area >= area_range[0]) and (area <= area_range[1]):
            contours_filtered.append(contour)
            cv2.drawContours(image, [contour], -1, (0, 255, 0), 1)
```

표면 재구성 (SR) 과정

2024-12-19 05:37

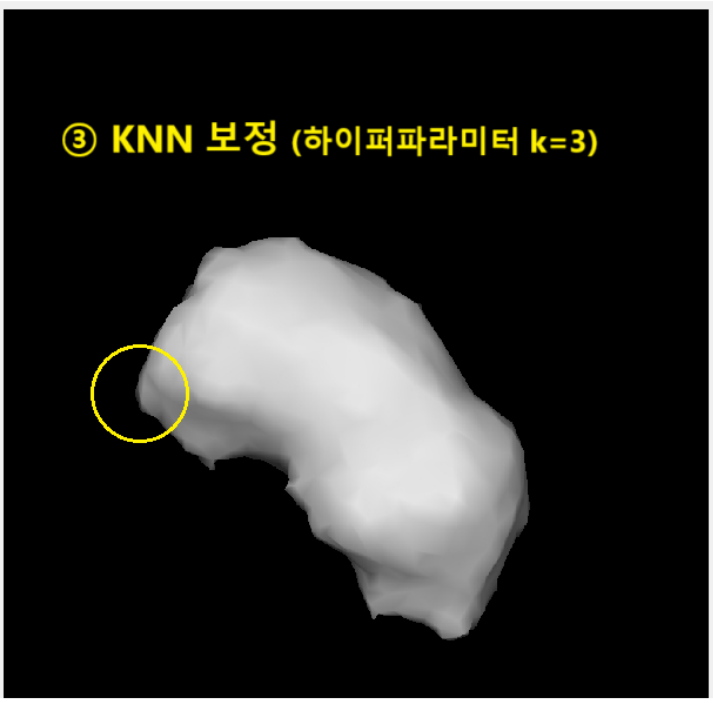


1	None	2024-12-19	0.00㎤
2	1970-08-22	2023-04-19	5.90㎤

```
points = np.array(points)
point_cloud = o3d.geometry.PointCloud()
point_cloud.points = o3d.utility.Vector3dVector(points)

point_cloud.estimate_normals()
mesh = o3d.geometry.TriangleMesh.create_from_point_cloud_poisso

mesh.compute_vertex_normals()
mesh.compute_triangle_normals()

return mesh
```

1	None	2024-12-19	0.00㎤
2	1970-08-22	2023-04-19	5.90㎤

```
mesh = o3d.geometry.TriangleMesh.create_from_point_cloud_pois
mesh = mesh.simplify_quadric_decimation(2000)

# 가장자리 제거
bbox = point_cloud.get_axis_aligned_bounding_box()
mesh = mesh.crop(bbox)

mesh.compute_vertex_normals()
mesh.compute_triangle_normals()

return mesh
```




```
# depth 보정 (KNN k=3)
point_cloud.orient_normals_consistent_tangent_plane(k=3)
point_cloud.normals = o3d.utility.Vector3dVector( - np.asarray

mesh = o3d.geometry.TriangleMesh.create_from_point_cloud_poiss
```

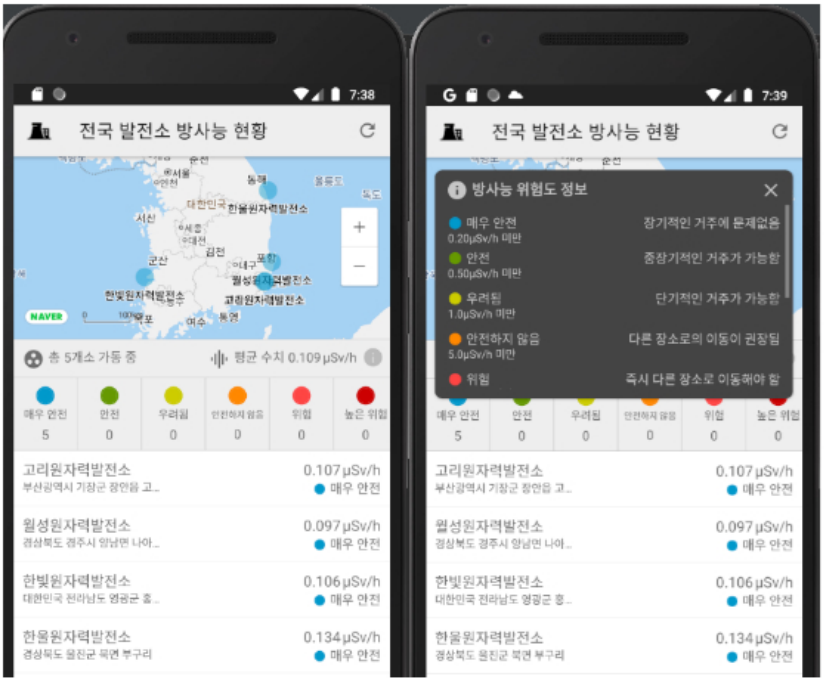

```
# depth 보정 (KNN k=15)
point_cloud.orient_normals_consistent_tangent_plane(k=15)
point_cloud.normals = o3d.utility.Vector3dVector( - np.asarray

mesh = o3d.geometry.TriangleMesh.create_from_point_cloud_poiss
```

원전 방사능 실시간 모니터링

Java/Android Application

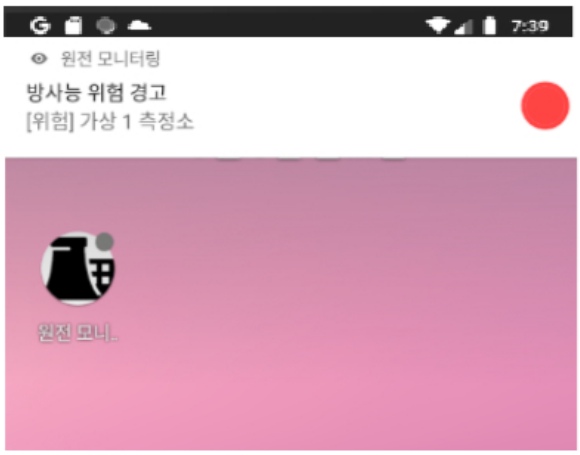
Introduction

전국 모니터링(Nationwide monitoring)



집중 모니터링(Focused monitoring)

가상 경보 시뮬레이션 (Virtual Warning Simulation)



- **원전 방사능 실시간 모니터링**은 Java / Android 로 작성된 공공 재난 안전 모바일 어플리케이션입니다.
  비상 경보와 같은 **미션 크리티컬**(MC) 기능의 시뮬레이션 기능을 제공합니다.
  - 전국 원자력발전소의 방사선량 및 안전도를 조회할 수 있다.
  - 기준치 이상의 방사선량이 관측되면 경고 알림을 발신한다.
  - 가상의 발전소를 통해 경고 알람 기능의 작동 여부를 점검할 수 있다.

Project Overview

Language	IDE
Java (1.8)	Android Studio (Ape)

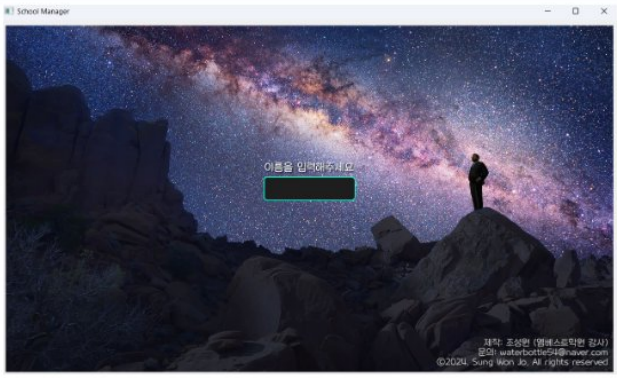
Author

- 조성원 (Sung Won Jo / David Jo)

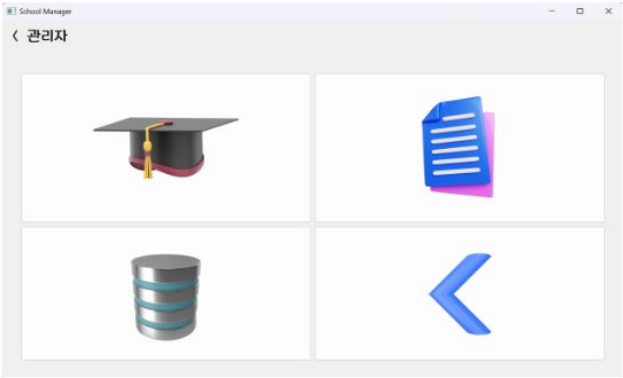
학습 관리자 (SE School Manager)

Introduction

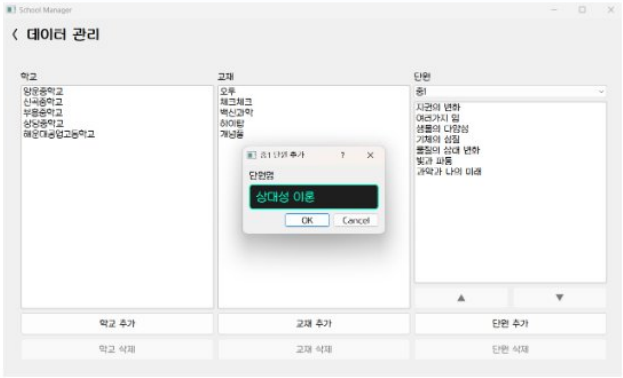
로그인(Login)



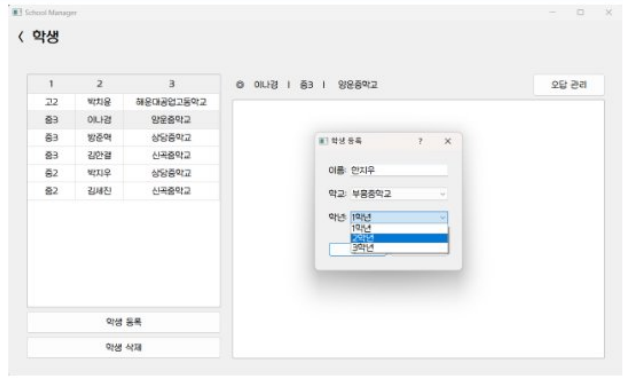
관리자 메뉴(Admin Menus)



데이터 관리(Data Management)

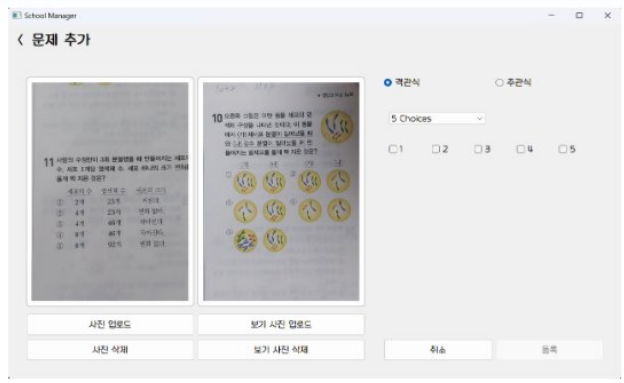


학생 등록(Student Registration)



문제 관리(Problem Management)

문제 등록(Problem Registration)



- **SE School Manager**는 **Qt5 / Python** 으로 작성된 **Desktop** 학습 관리 소프트웨어입니다.
  이 프로그램은 저의 파트타임 강사 경험을 기반으로 제작되었으며, 학원, 교습소에 적합한 용도입니다.
  SQLite DB 를 통해 제반 데이터 관리, 학생 관리, 문제 관리, 오답 관리 등의 기능을 제공합니다.

Architecture

- MVVM Pattern 을 사용하였다. 일관성 있는 데이터 제공을 위해 Repository Layer를 구현하였다.
- 스크린(-Fragment.py)마다 비즈니스 로직 처리를 위한 개별 뷰모델(-ViewModel.py)이 구현되어 있다.
- UI의 규모가 작지 않고 잦은 갱신이 요구되므로 Observer 패턴을 사용하였다.

Data Persistency

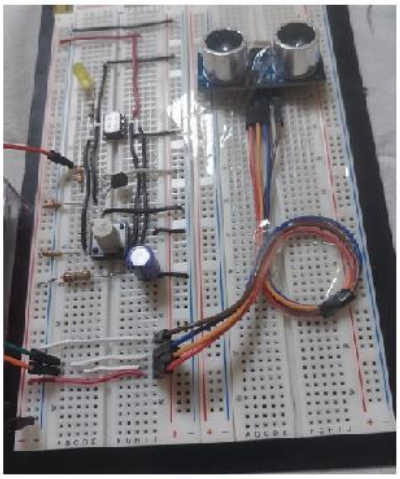
- SQLite Database : 필드가 2개 이상인 모델이 저장된다.
- Json File: 원시형, 배열 타입의 모델이 저장된다.

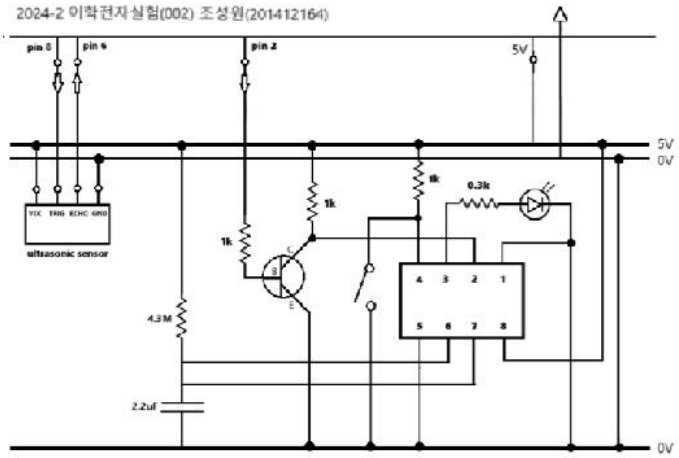
Author

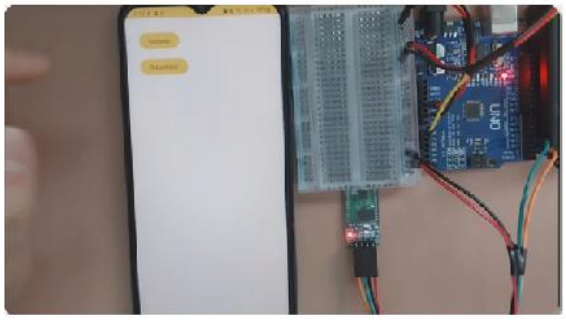
- 조성원 (Sung Won Jo)

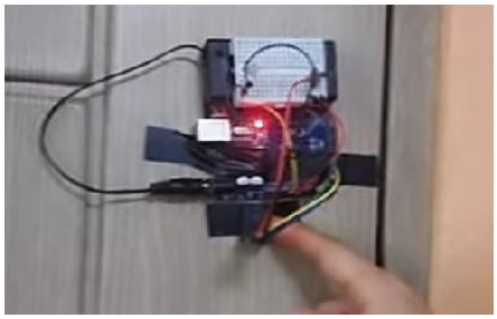
데스크탑 (Qt, MFC) / 모바일 개발(Android, Flutter)

마이크로프로세서 / 블루투스 Serial 통신









역학계 분석 / 시뮬레이션