



악보를 볼 줄 몰라도 연주가 가능하다.

머신러닝 기능을 활용한 피아노 전반 지시 어플리케이션



Intel Project

연주 도우미

TEAM : 낙원상가
양승용 오태영 박철우 최은호

Contents

- 01 프로젝트 개요**
- 02 기술적 접근**
- 03 학습 과정 및 성과**
- 04 문제점 및 해결사항**
- 05 비즈니스 가치 및 기대효과**
- 06 향후 발전 방향**

01

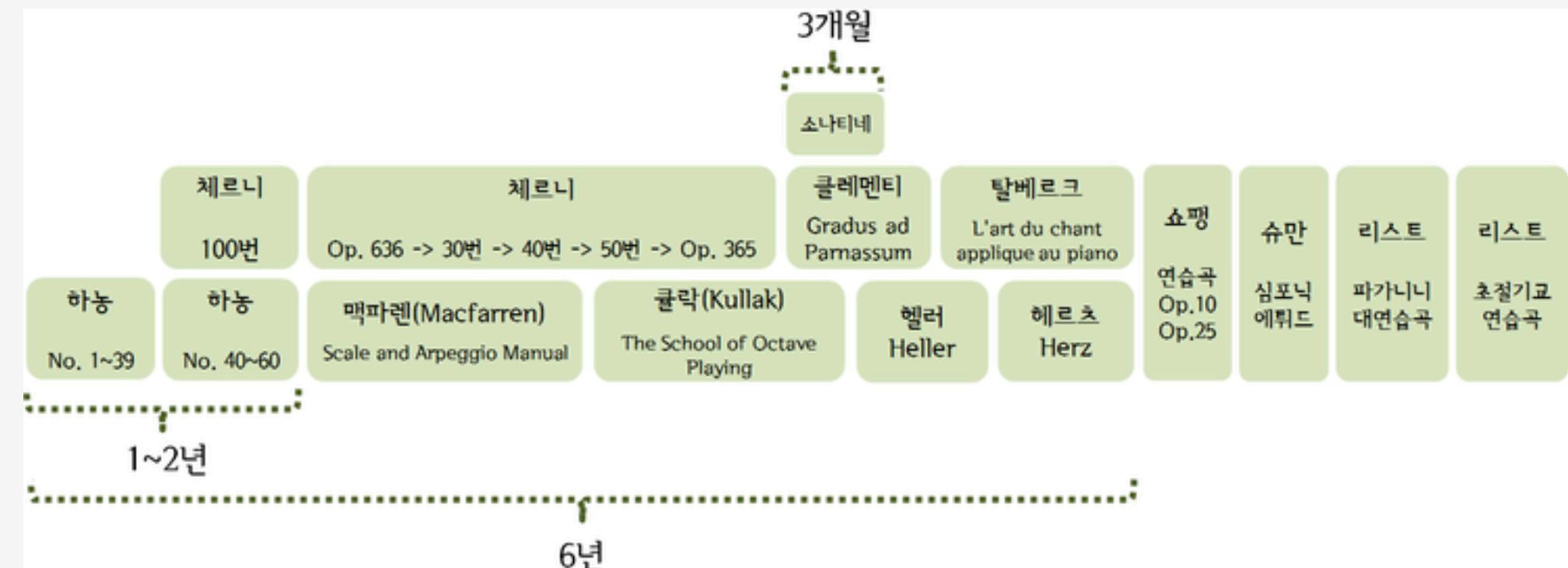
프로젝트 개요

1. 주제 선정 배경

피아노 및 악기를 독학하면서 가장 큰 어려움 중 하나는 악보 읽기입니다. 귀로 음악을 듣고 즉각적으로 해당 음을 정확하게 누를 수 있는 사람들도 있지만, 그런 능력을 가진 사람은 극히 일부에 불과합니다. 대부분의 사람들은 악보를 읽는데 많은 시간이 필요하고, 특히 악보를 아직 배우지 않은 어린이들에게는 더 큰 장애물이 될 수 있습니다.

어린이들은 악보를 읽지 않고도 연주를 시작할 수 있는 환경이 필요합니다. 이를 해결하기 위해, 악보 없이 연주 가능한 프로그램을 제공함으로써 어린이들이 악기를 배우는데 있어 더 쉽게 접근하고, 그들의 첫 시작을 돋는 시스템을 만들면 큰 도움이 될 것입니다. 이러한 아이디어에서 출발하여, 어린이들이 즐겁고 직관적으로 악기를 배울 수 있는 프로그램을 개발하고자 이 주제를 선정하게 되었습니다.

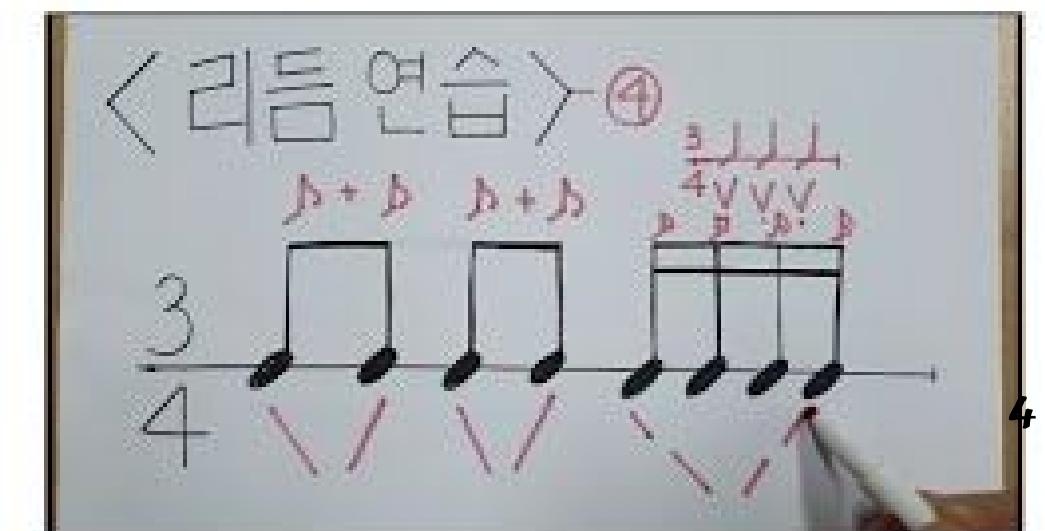
- 피아노 배움 과정



- 피아노 기초 연습

기초 리듬 연습 (6)

하농 4번

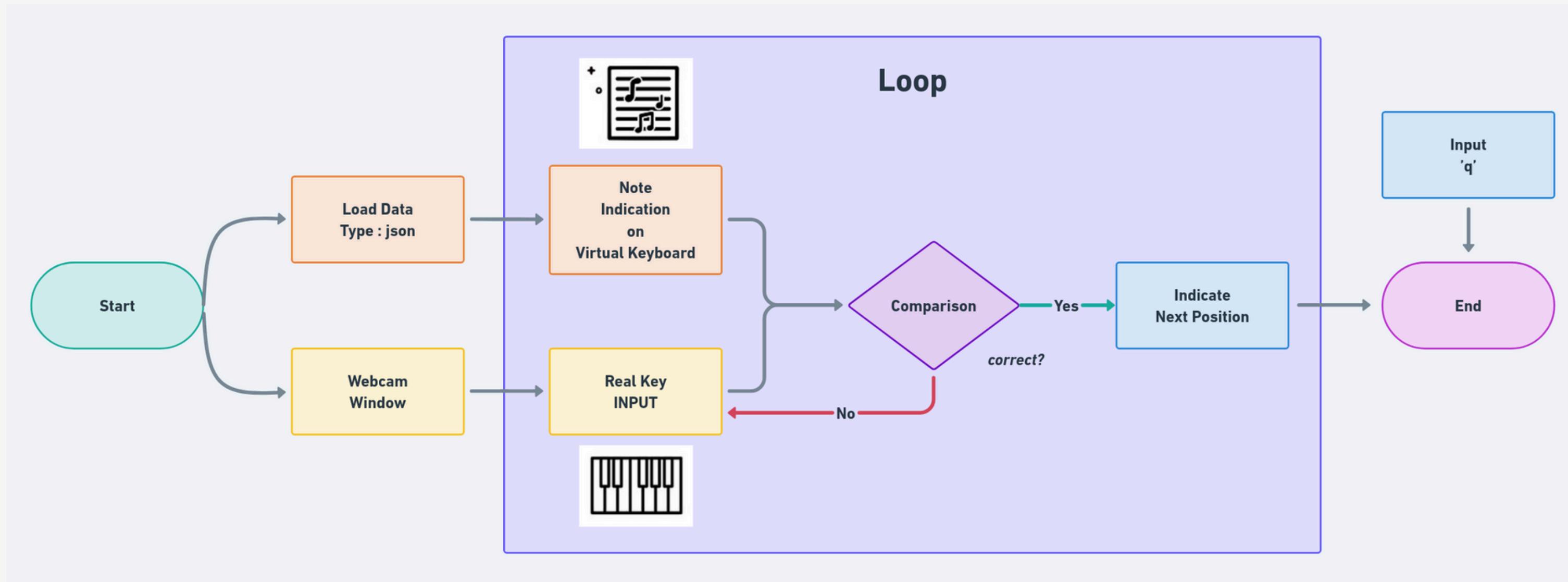


01

프로젝트 개요

프로젝트 목적

- 악보 없이 자유롭게 연주할 수 있는 피아노 학습 환경 제공
- AI 기반 실시간 피아노 학습 시스템
- 핵심 기능: 자동 건반 인식, 다음 음 예측, 실시간 피드백 제공



1. 환경 설정

- Python 설치 = Python 3.7 이상 설치
- 가상 환경 설정 (선택 사항): 가상 환경을 사용하면 프로젝트마다 독립적인 패키지 환경을 유지
- 필요한 디렉토리 및 파일 준비: YOLO 모델(best_1_1680.pt) 파일을 준비 및 코드 내 경로 지정

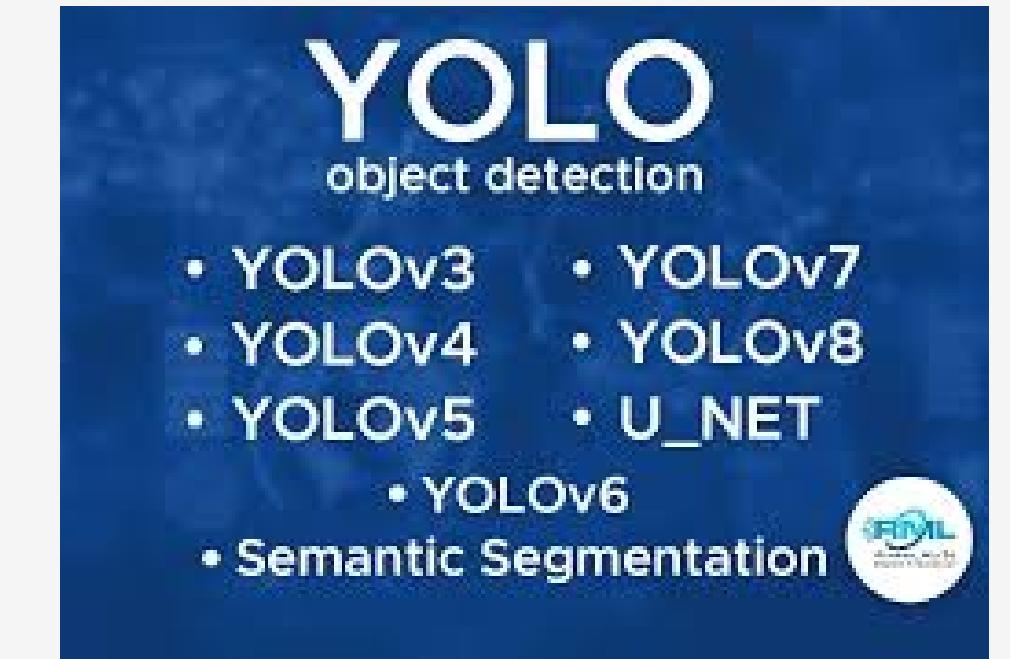


2. 주요 패키지 설치

- 패키지 설치
 - ultralytics: YOLO 모델을 사용 패키지
 - opencv-python: OpenCV를 사용하여 웹캡 처리
 - pyqt5: PyQt5 GUI 프레임워크
 - numpy: YOLO 모델 결과 처리
 - json: 기본 Python 모듈, JSON 파일을 처리 시 사용

- 기본 환경 버전

`numpy>=1.21.0`
`opencv-python>=4.5.5`
`ultralytics>=8.0.0`
`PyQt5>=5.15.4`



02

기술적 접근

- YOLO 모델을 활용한 실시간 음표 감지
 - OpenCV와 YOLO 모델을 이용해 웹캠에서 음표를 실시간 감지
 - 감지된 음표 UI에 업데이트 및
- 가상 피아노 키보드
 - PyQt5를 사용하여 가상 피아노 키보드 생성
- JSON 파일을 통한 연주
 - JSON 파일을 통해 사용자가 연주할 음표를 로드 및 해당 음표를 가상 피아노 키보드에서 차례대로 하이라이트
- UI 상호작용
 - PyQt5의 QPushButton, QLabel 위젯을 사용하여 버튼 클릭, 텍스트 표시, 실시간 영상 표시 등의 사용자 상호작용 처리

02

기술적 접근

1. 웹캠을 통한 실시간 음표 감지:

YOLO 모델을 사용하여 실시간으로 웹캠에서 촬영된 이미지를 분석하고, 특정 음표(예: C, D, E 등)를 감지하고 웹캠에서 받은 프레임을 OpenCV와 PyTorch 기반 YOLO 모델을 사용하여 처리합니다. YOLO 모델의 결과에서 각 객체(음표)의 클래스 ID를 확인하고, 가장 높은 신뢰도를 가진 음표를 `detected_note` 변수에 저장합니다.

2. 가상 피아노 키보드

`KeyboardView` 클래스로 가상 피아노 키보드를 표시하는 Qt 위젯으로, 키보드의 각 음표(C, D, E 등)를 흰 건반과 검은 건반으로 나누어 그려 각 건반은 `QPainter`를 사용하여 그리며, 마우스 클릭 이벤트가 발생하면 해당 건반을 누른 것으로 처리하여 콘솔에 출력합니다.

사용자가 감지된 음표를 따라 연주할 때, 해당 음표의 건반을 하이라이트(색상 변경)하여 사용자가 어떤 음표를 따라가야 하는지 명확히 보여줍니다.

3. 음표에 맞춰 연주

JSON 파일에는 연주할 음표들(예: {"note": "C"})이 순차적으로 정의하여 사용자가 JSON 파일을 로드하면, 해당 파일에 포함된 음표들이 순차적으로 가상 키 보드에서 하이라이트되어 실제로 플레이되는 음표를 확인할 수 있습니다.

음표는 `play_notes_from_json` 메서드를 통해 처리되며, 그에 맞춰 키보드에서 연주하는 음표가 자동으로 하이라이트되어 각 음표는 `QTimer`를 사용하여 정해진 시간 간격으로 하이라이트를 전환하며 연주됩니다.

4. 음표 감지 및 맞춤 체크:

감지된 음표는 "self.detected_note"에 저장되며, 현재 연주해야 할 음표와 비교하여 정확히 맞는지 확인합니다.

사용자가 실제로 연주하는 음표가 맞으면 `self.is_note_correct = True`가 설정되고, 그 후 음표 인덱스 (`note_index`)가 증가하여 다음 음표로 넘어갑니다. `check_detected_note` 메서드는 이 로직을 수행하며, 맞는 음표가 감지되면, 가상 키보드에서 하이라이트된 키를 리셋하고, 새로운 음표로 변경합니다.

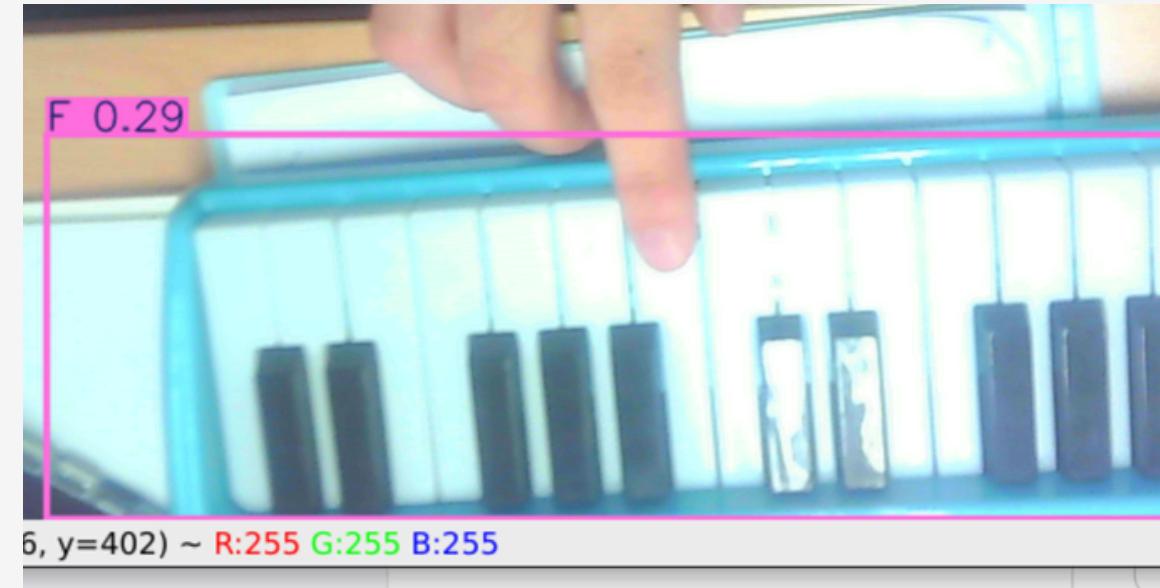
5. UI 및 버튼 상호작용

PyQt5 UI 사용하여 웹캠 화면, 가상 키보드, JSON 파일 로드 버튼, 음표 플레이 버튼, 음표 감지 상태 표시 레이블 등이 있습니다. 사용자가 JSON 파일을 로드하거나 음표를 연주하는 버튼을 클릭하면 그에 따라 동작이 수행됩니다. 각 버튼은 슬롯 메서드와 연결되어 있으며, 이벤트 기반으로 동작합니다. 예를 들어, "Play Notes" 버튼을 클릭하면 `play_notes_from_json` 메서드가 실행되어 연주를 시작합니다.

1단계 ~ 4단계 학습 개선 과정 (라벨링 및 데이터 증강)

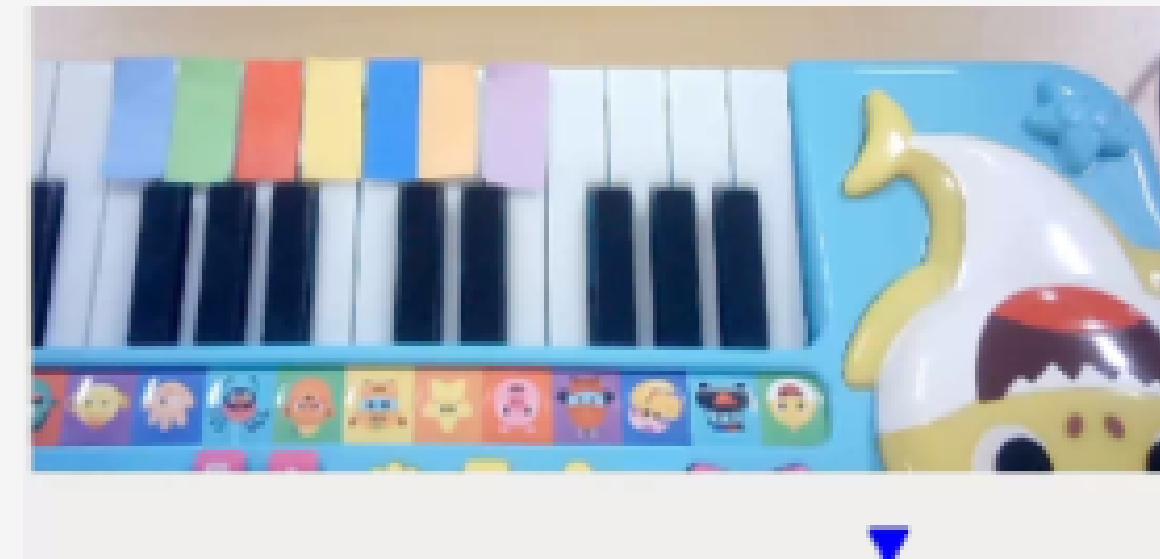
1단계 - 피아노 전체 라벨링 및 YOLO 모델 학습

- 문제 사항: 전체 피아노 건반을 라벨링하여 학습을 진행했으나,
높은 복잡도와 유사한 형태로 인해 모델이 건반을 정확히 인식하지 못하는 결과 발생.
- 해결책: 색종이 부착을 통한 인식률 개선 및 라벨링 보강.



2단계 - 건반에 색종이를 부착하여 인식률 개선

- 문제 사항: 색종이를 각 건반에 부착하여 모델을 학습시킨 결과, 30%의 인식률을 달성했으나 여전히 불안정 한 인식률을 보임.
- 해결책: 색종이 부착을 통한 인식률 개선 및 라벨링 보강.



2단계 - 건반에 색종이를 부착하여 인식률 개선

- 문제 사항: 색종이를 각 건반에 부착하여 모델을 학습시킨 결과, 30%의 인식률을 달성했으나 여전히 불안정 한 인식률을 보임.
- 해결책: 건반을 하나씩 분리하여 라벨링하고, 개별적으로 이미지 해석 범위를 좁히는 접근 방식으로 개선.

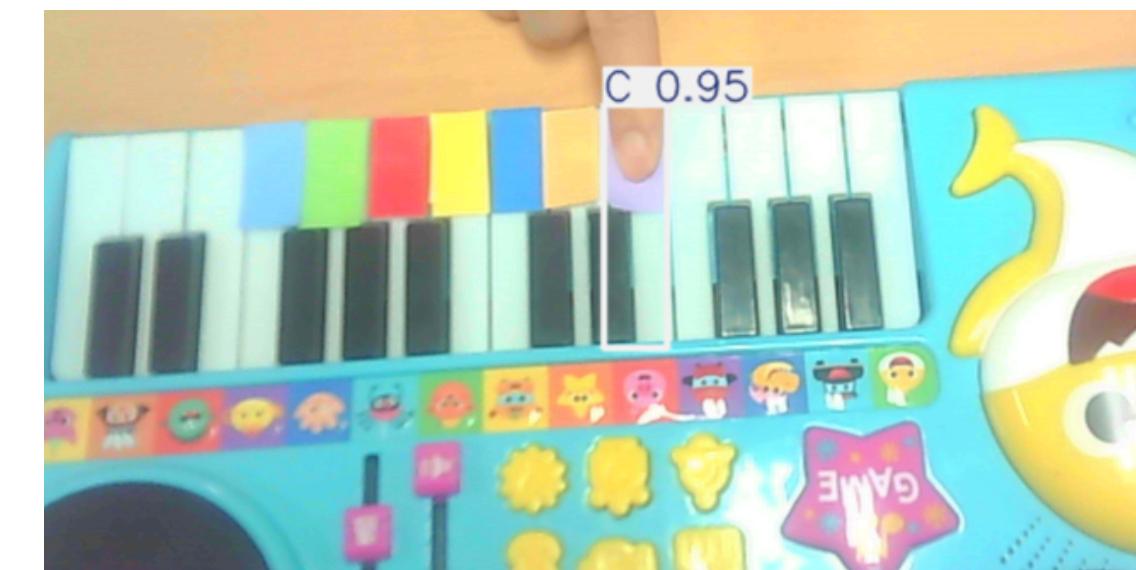
3단계 - 건반 2분할 학습

- 문제 사항: 건반을 두 부분으로 분할하여 학습시킨 결과, 60% 인식률을 달성했으나 여전히 일부 건반 인식 어려움.
- 해결책: 건반을 7개 기본 음으로 나누어 학습하고, 데이터 증강(Augmentation)을 적용하여 인식 정확도 향상.



4단계 - 건반 7분할 학습 및 학습 데이터 증강(Data Augmentation)

- 도레미파솔라시로 7개의 건반을 각기 분할하여 학습시킨 결과, 100% 인식률을 달성하며 모델이 각 건반을 정확하게 인식함.
- 모델의 일반화 성능 향상을 통해 다양한 조명 조건과 환경에서 일관된 성능 유지



- **연속된 동일한 건반에 대한 인식 오류**

- 건반 입력 후 다음 건반 입력 사이에 딜레이를 추가
- 해당 시간 동안 데이터 처리를 하지 않으면서 문제 해결

- **가상 건반과 실제 건반 간 색상 불일치로 인해 사용자에게 시각적 혼동 초래**

- 가상 건반의 색상을 실제 건반에 적용한 색상과 통일
- 기존에 파란색으로 건반을 하이라이트 하면서 순서를 표시
- 건반 위에 화살표로 표시하면서 혼동을 줄이고 좀 더 직관적으로 인지 가능하게 함

- **조명 또는 색 온도 변화에 따른 인식 성능 저하**

- augmentation 항목을 추가하여 학습한다면 개선될 것으로 예상됨

비즈니스 가치 및 기대효과

- 사용자 경험: 악보 없이 자유로운 연주, 실시간 피드백 제공
- 교육적 가치: 피아노 학습의 접근성을 높이고, 악보를 몰라도 연주 가능
- 기술적 우위: 기존 학습 시스템과 비교하여 차별화된 AI 기술 및 실시간 기능

1. 양손으로 건반 할 수 있게 hand estimation
2. 악기의 확장
3. 가상 키보드
4. 조립 가이드
5. 조종 어시스턴트
6. 가상 가이드북

목표 일정 및 진행도

							계획	진행	
3조 프로젝트 계획 및 진행사항									
목표	상세내용	담당자	1일차(수)	2일차(목)	3일차(금)	4일차(월)	5일차(화)	6일차(수)	7일차(목)
프로젝트 구조 설계	기능 정의 디렉토리 구조 설정 클래스 다이어그램 작성	최은호, 박철우 오태영, 양승용							
기본 클래스 구현	DataHandler 클래스 구현 검토 기본 예외처리 구현 패키지구조 생성	오태영, 최은호							
Virtual Keyboard UI 기본 구조 (core)	메인 윈도우 레이아웃 설계 키보드 디스플레이 영역 구현 기본 컨트롤 버튼 구현	양승용							
키 인식 ML알고리즘 구현 (core)	OpenCV, Openvino 손 동작 및 키 인식 알고리즘 구현 인식 정확도 개선 보정 알고리즘 구현	오태영, 박철우							
core 기능 통합 및 동기화	웹캠 데이터와 UI 동기화 악보 데이터 실시간 로딩	오태영, 최은호							
기능 통합테스트 및 보완점 수정	컴포넌트 통합 테스트 성능 테스트 버그 수정	박철우, 양승용							
발표자료		양승용, 오태영							

							계획	진행	
진행도 : 2/7 (12월 16일)		3주 프로젝트 계획 및 진행사항							
목표	상세내용	담당자	1일차(수)	2일차(목)	3일차(금)	4일차(월)	5일차(화)	6일차(수)	7일차(목)
프로젝트 구조 설계	기능 정의 디렉토리 구조 설정 클래스 디어그램 작성	최은호, 박철우 오태영, 양승용	■						
기본 클래스 구현	DataHandler 클래스 구현 검토 기본 예외처리 구현 패키지구조 생성	오태영, 최은호	■						
Virtual Keyboard UI 기본 구조 (core)	메인 윈도우 레이아웃 설계 키보드 디스플레이 영역 구현 기본 컨트롤 버튼 구현	양승용		■	■				
키 인식 ML알고리즘 구현 (core)	OpenCV, Openvino 손 동작 및 키 인식 알고리즘 구현 인식 정확도 개선 보정 알고리즘 구현	오태영, 박철우				■	■		
core 기능 통합 및 동기화	웹캠 데이터와 UI 동기화 악보 데이터 실시간 로딩	오태영, 최은호					■	■	
기능 통합테스트 및 보완점 수정	컴포넌트 통합 테스트 성능 테스트 버그 수정	박철우, 양승용						■	■
발표자료		양승용, 오태영					■	■	■

