

# AIPro iNet Solution

## Demo Guide

### (API Version)

Part I. Demo 프로그램 설치 가이드

Part II. iNet Solution 사용 가이드

#### System Requirement & Dependency

항목	내용
포함 모델	Object Detection, Object Tracking, Person Attribute Recognition, Pose Estimation, Action Recognition
사용 언어	C/C++
OS	Windows 11
설치환경	Visual Studio 2022 CUDA 11.6.2 cuDNN 8.4.0
Demo Dependency	OpenCV-4.5.5(자체 포함), TensorRT-8.4.2.4 (자체 포함), Opencvino(자체포함)
GPU 최소 사양	RTX 30xx 이상 필수

작성일: 2023 년 5 월

작성자: 박 천 수

이메일: [cspk@skku.edu](mailto:cspk@skku.edu)

## Part I. Demo 프로그램 설치 가이드

### 1. Visual Studio 2022 설치

- A. Community 버전(무료) 설치 가능
- B. (중요)CUDA 와 cuDNN 설치전에 Visual Studio 를 먼저 설치해야 함

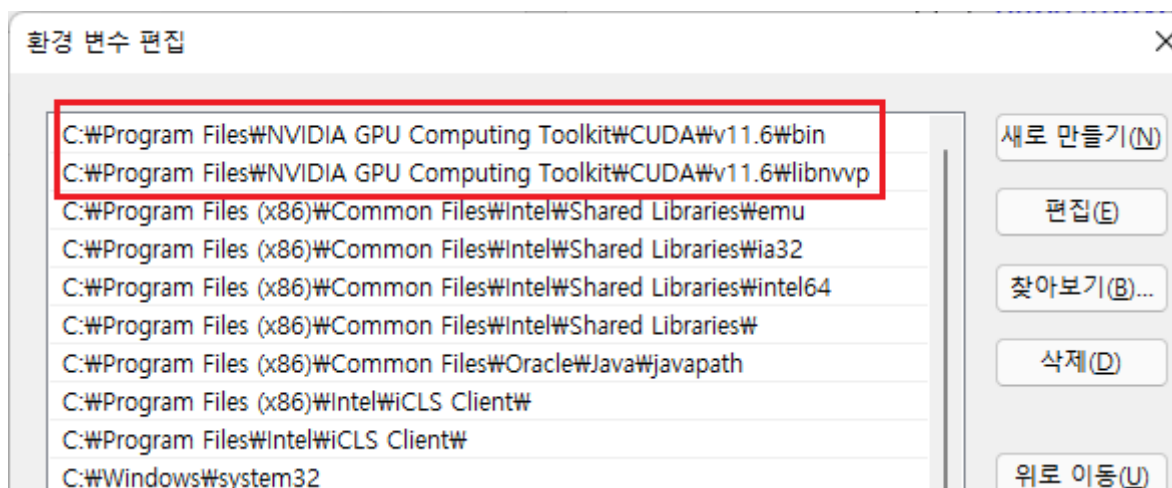
### 2. CUDA 설치

#### A. CUDA Toolkit 11.6.2 설치

- 링크: <https://developer.nvidia.com/cuda-toolkit-archive>
- 설치 파일: cuda\_11.6.2\_511.65\_windows.exe
- 설치 시 기본 경로 사용 권장

#### B. PATH 환경 변수에 아래 내용 추가

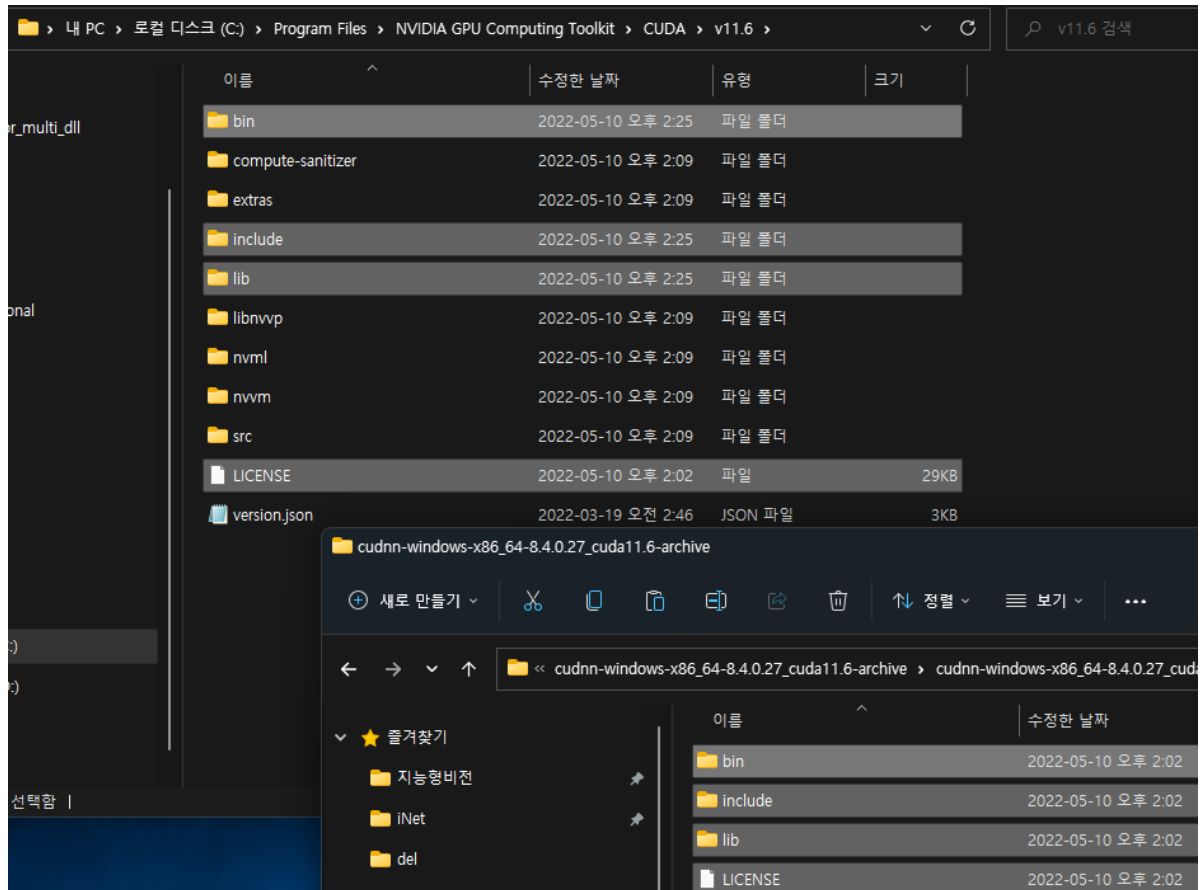
- C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v11.6\bin
- C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v11.6\libnvvp



### 3. cuDNN 설치

#### A. cuDNN 8.4.0 설치

- 설치링크: <https://developer.nvidia.com/rdp/cudnn-download>
- 설치 파일: cudnn-windows-x86\_64-8.4.0.27\_cuda11.6.exe
  - 다운 받기위해 NVIDIA 회원 가입이 필요
- 압축을 풀고 bin, include, lib 폴더를 “C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v11.6” 위치에 덮어쓰는 방식으로 복사



## B. zlibwapi.dll 설치

- cudnn 8.4 는 zlib 의 zlibwapi.dll 을 내부적으로 이용
  - zlibwapi.dll 이 설치가 안된 경우 “Could not locate zlibwapi.dll. Please make sure it is in your library path!” 에러 메시지가 출력됨
- 아래 페이지에서 ZLIB\_DLL 링크를 통해 zlib123dllx64.zip 을 다운
  - <https://docs.nvidia.com/deeplearning/cudnn/install-guide/index.html#install-zlib-windows>

### 3.1.3. Installing zlib

zlib is a data compression software library that is needed by cuDNN.

#### Procedure

1. Download and extract the zlib package from **ZLIB DLL**. Users with a 32-bit machine should downl

**Note:** If using Chrome, the file may not automatically download. If this happens, right-click the l

2. Add the directory path of **zlibwapi.dll** to the environment variable PATH.

- zlib123dllx64.zip 파일 압축을 풀고 dll\_x64 위치의 zlibwapi.dll 파일을 “C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v11.6\bin” 위치에 복사

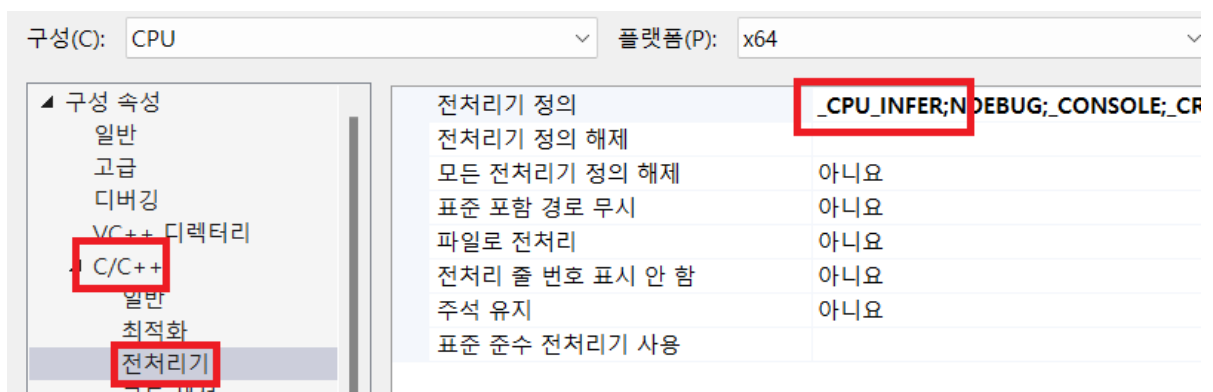
## 4. Visual Studio 2022 설정

### A. iNet-API-Demo Repository 클론 또는 복사

- Address: <https://github.com/AIProCo/iNet-API-Demo>

### B. iNet-API-Demo.sln 솔루션을 열고 프로젝트 실행 환경 설정

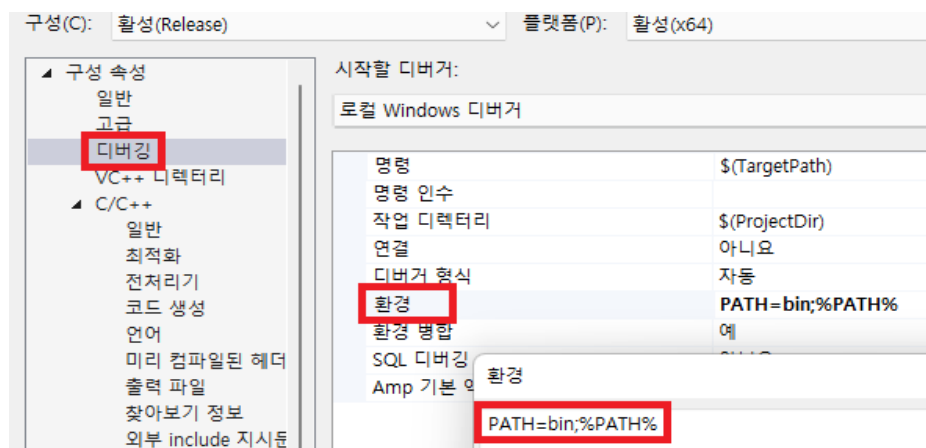
- **Nvidia GPU 를 사용해 모델을 추론하는 경우 솔루션 구성 “Release”를 사용하고, CPU 또는 Intel 내장 GPU 를 사용하는 경우 “CPU”를 사용**
- 솔루션 구성 CPU 전처리기 수정
  - 이동경로: C/C++ → 전처리기 → 전처리기 정의
  - `_CPU_INFER` 전처리문 추가



- 4.C 부터는 Release 와 CPU 구성 모두에 공통으로 적용

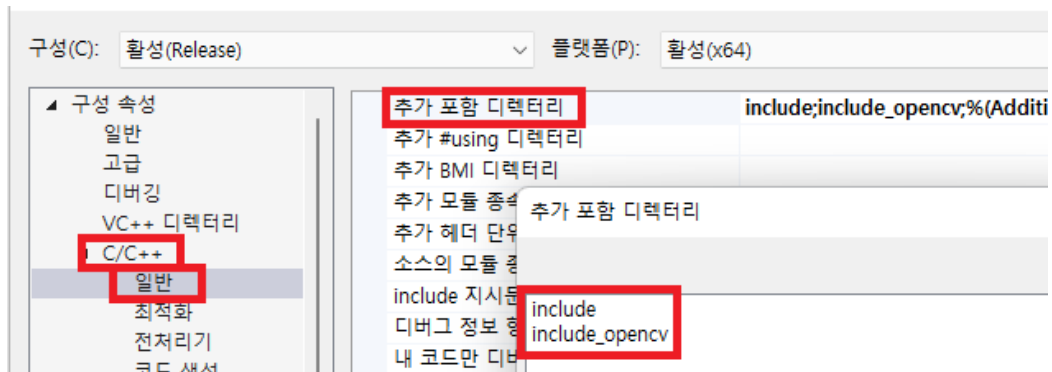
### C. 솔루션 실행을 위한 로컬 PATH 변수 수정(시스템 PATH 변수에 영향 없음)

- 이동 경로: 디버깅 → 환경
- PATH 변수에 “bin” 폴더와 %PATH% 입력(해당 위치 dll 를 읽을 수 있게 됨)
  - 입력 예시: `PATH=bin;%PATH%`



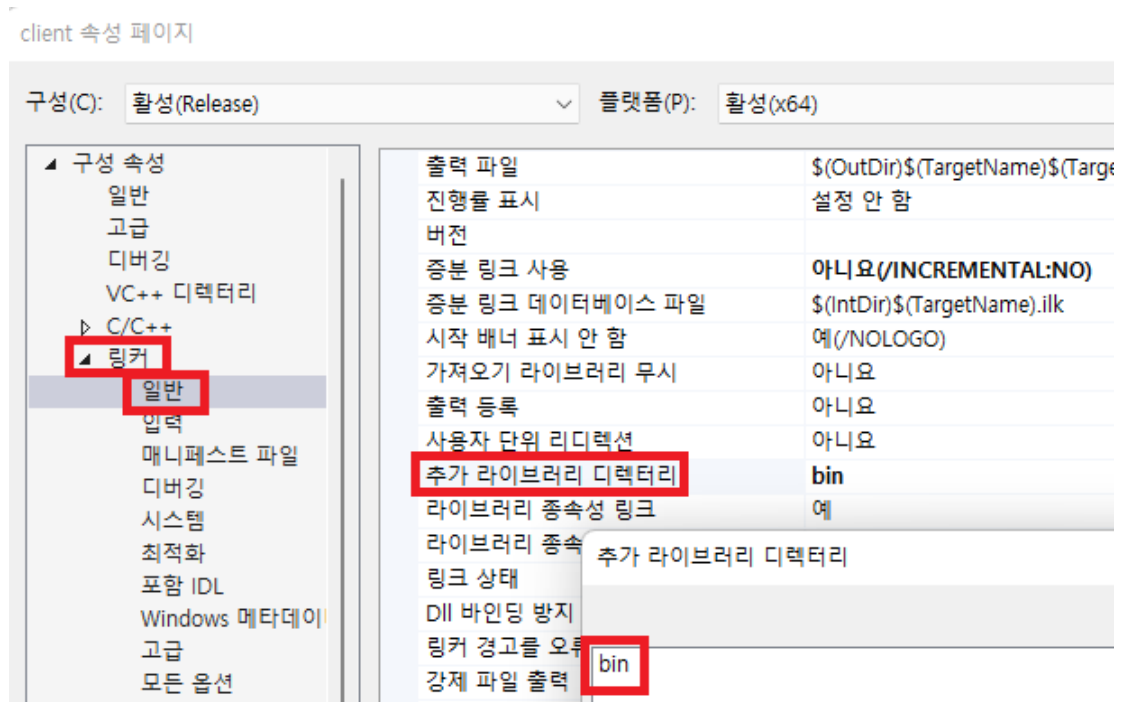
## D. 추가 포함 디렉터리 입력

- 이동 경로: C/C++ → 일반 → 추가 포함 디렉터리
- 추가 포함 디렉터리에 “include”와 “include\_opencv” 입력



## E. 추가 라이브러리 디렉터리 입력

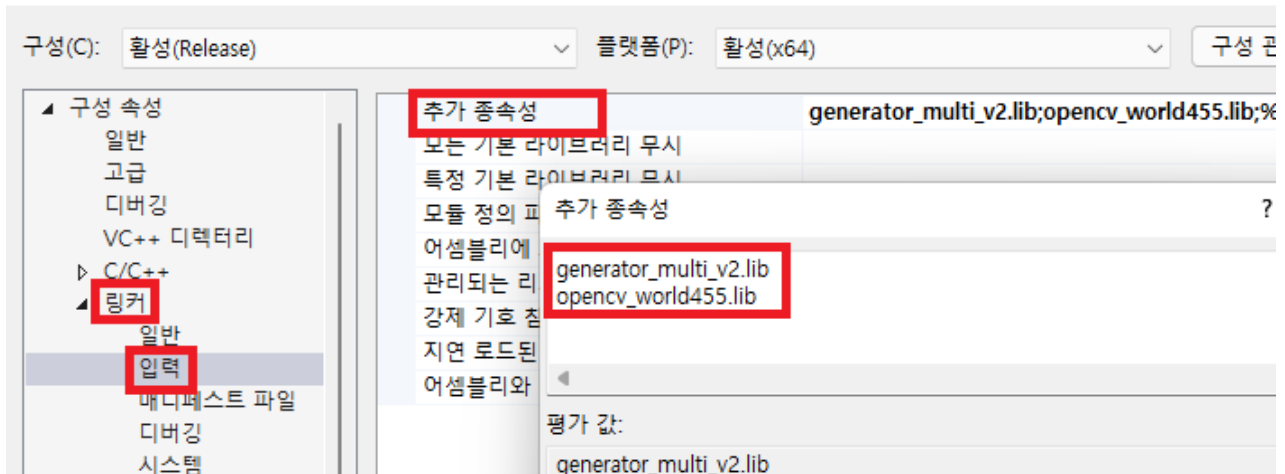
- 이동 경로: 링커 → 일반 → 추가 라이브러리 디렉터리
- 추가 라이브러리 디렉터리에 “bin” 입력



## F. 추가 종속성 입력

- 이동 경로: 링커 → 입력 → 추가 종속성
- 추가 종속성에 “generator\_multi\_b.lib”와 “opencv\_world455.lib” 입력

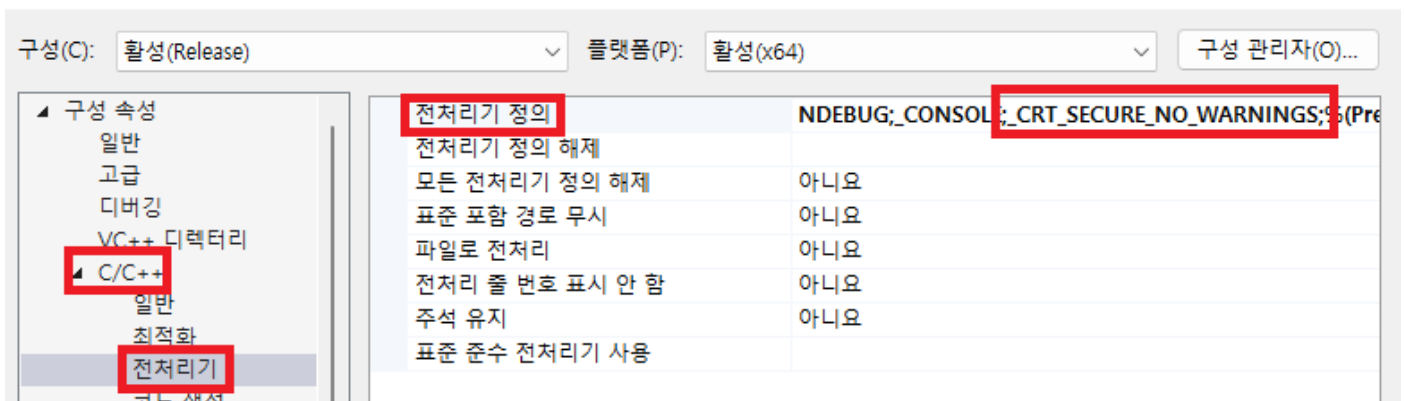
client 속성 페이지



### G. 보안 에러 Disable

- MS 보안 함수 강제 사용 Disable
  - (참고!) 에러 메시지: error C4996: 'localtime': This function or variable may be unsafe.
- 이동 경로: C/C++ → 전처리기 → 전처리기 정의
- 전처리기 정의에 “\_CRT\_SECURE\_NO\_WARNINGS” 입력

client 속성 페이지



## 5. 필요 파일 설치

- “bin, inputs, videos.zip” 파일 다운로드 및 압축 해제. bin, inputs, videos 디렉토리를 솔루션 디렉토리(\*.sln 파일과 같은 위치)로 복사
  - 링크는 github repository 의 readme 문서 참고

## 6. Release 모드 또는 CPU 모드를 설정 후 실행

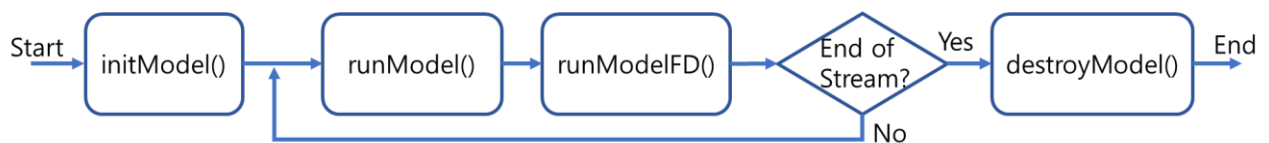
## Part II. iNet Solution 사용 가이드

### 1. Solution 동작

- AIPro iNet Solution 실행은 크게 초기화, 실행, 소멸 3 단계로 이루어짐. 각 단계에 해당하는 함수와 세부 내용은 아래와 같음

단계	함수	내용
초기화	initModel()	<ul style="list-style-type: none"> <li>• 솔루션 실행에 필요한 모델과 내부 저장소를 초기화</li> </ul>
실행 1	runModel()	<ul style="list-style-type: none"> <li>• frame 배치를 입력 받고 추론 동작 수행</li> <li>• Object Detection, Tracking, Counting(Line &amp; Zone), PAR, Pose Estimation, Action Recognition 동작을 담당</li> <li>• 검출된 물체 박스, Pose(Skeleton), 인식 동작을 리턴</li> </ul>
실행 2	runModelFD()	<ul style="list-style-type: none"> <li>• frame 배치를 입력 받고 Fire Detection(FD) 추론 동작 수행</li> <li>• 검출된 Fire와 Smoke 객체를 리턴</li> </ul>
소멸	destroyModel()	<ul style="list-style-type: none"> <li>• 생성된 모델과 저장소 공간 해제</li> </ul>

- 초기화와 소멸 단계는 프로그램 시작시와 종료시에 각 1 번씩 호출되고, 실행 단계는 frame 배치를 구성 후 runModel()과 runModelFD()를 반복적으로 호출하며 실행



<그림> iNet 솔루션 동작 플로우

### 2. 솔루션을 이용한 프로그램 개발 작업 내용

- 기본적으로 iNet 솔루션은 config.json 파일을 Parsing 해 Config 객체(cfg)를 생성하고 이를 이용해 전체 솔루션을 동작 시킴. 솔루션을 이용한 프로그램을

개발하기 위해서는 cfg 객체를 초기화하는 parseConfigAPI() 함수 내용을 각 응용 프로그램에 맞게 수정해야 함

- parseConfigAPI() 함수 내부의 상수 값은 변경하지 않는 것을 권장

- 미리 설정된 json 파일과 응용 프로그램 동작시 추출되는 데이터를 이용해 응용 프로그램에 맞는 cfg 객체를 생성 후 예제 코드와 같은 방식으로 초기화, 실행, 소멸 동작을 수행해야 함

### 3. Counting 정보

- 기본적으로 CntLine 을 지나가는 사람 수와 Zone 내부에 머무르는 사람 수를 카운팅. 사람 수를 셀 때는 성별(남, 여)과 연령층(어린이, 성인, 노인)을 고려해 총 6 가지 경우를 각각 카운팅

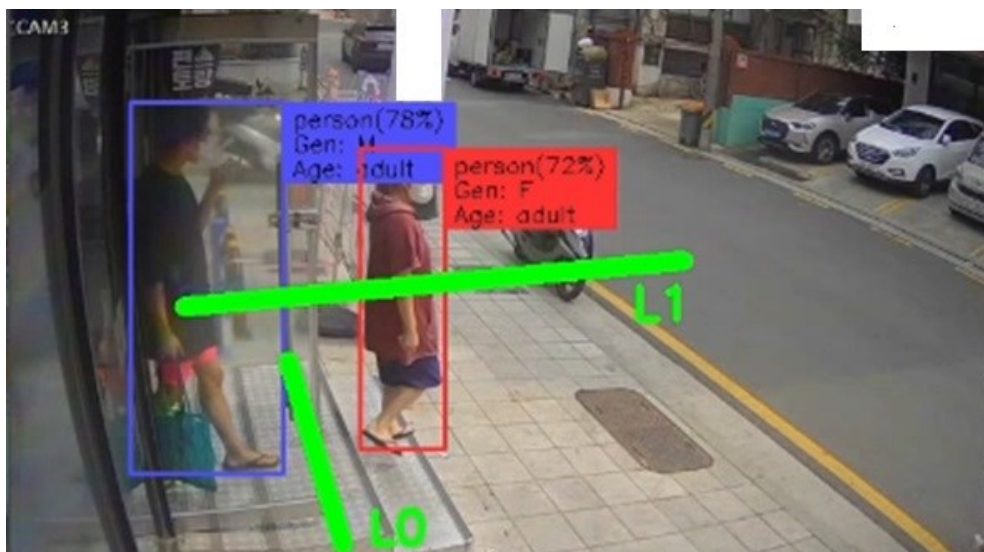
- 각 카운팅 정보는 2 x 3 배열에 [남/여][어린이/성인/노인] 방식으로 저장되고 아래와 같은 형식으로 출력됨

M\_Total(M\_Child, M\_Adult, M\_Elder) F\_Total(F\_Child, F\_Adult, F\_Elder)

- 각 CntLine 은 사람의 이동 방향을 고려해 Up/Down 또는 Left/Right 방향으로 지나가는 사람 수를 카운팅하고, 각 Zone 은 현재 내부에 위치한 사람 수와 현재까지 존재한 사람의 수를 히트맵(1 초에 1 번 카운팅) 방식으로 저장

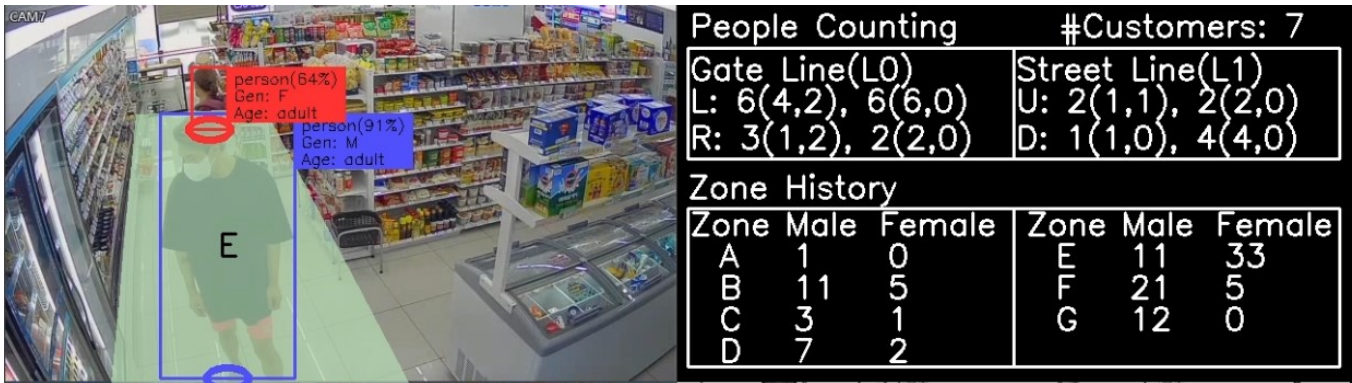
- CntLine 은 카운팅을 위한 TotalUL[2][3], TotalDR[2][3] 배열을 포함

- Zone 은 히트맵을 위한 curPeople[2][3], hitMap[2][3] 배열을 포함



<그림> Line Setting Example (L0 and L1)





&lt;그림&gt; Line Counting &amp; Zone Hitmap Example

#### 4. API 함수 설명

**bool** initModel(Config &cfg, ODRecord &odRcd, FDRecord &fdRcd)

Initialize model

- param cfg configuration struct
- param odRcd object detection record struct
- param fdRcd fire detection record struct
- return initialization result(true: success, false: fail)

**bool** runModel(vector<vector<DetBox>> &dboxesMul, vector<Mat> &frames, vector<int> &vchIDs, vector<uint> &frameCnts, float odScoreTh, float actScoreTh)

Run OD, PAR, Pose, and Action models for a frame batch

- param dboxesMul return detected dboxes of all video channels(vchIDs)
- param frames batch of frames
- param vchIDs vchIDs of batched frames
- param frameCnts frameCnts of batched frames
- param odScoreTh threshold for filtering low confident object detections
- param actScoreTh threshold for filtering low confident action recognitions
- return runModel result(true: success, false: fail)

**bool** runModelFD(vector<vector<FireBox>> &fboxesMul, vector<Mat> &frames, vector<int> &vchIDs, vector<uint> &frameCnts, float fdScoreTh)

Run FD models for a frame batch

- param fboxesMul return detected fboxes of all video channels(vchIDs)
- param frames batch of frames
- param vchIDs vchIDs of batched frames
- param frameCnts frameCnts of batched frames
- param fdScoreTh threshold for filtering low confident detections

- return flag for the running result(true: success, false: fail)

**bool** destroyModel()

Destroy model

- param None
- return flag for destruction result(true: success, false: fail)

**bool** resetCntLineAndZone(**ODRecord** &odRcd)

Reset CntLine and Zone configuration

- param odRcd record struct
- return flag for reset(true: success, false: fail)

**bool** resetCntLineAndZoneRecord()

Reset CntLine and Zone record

- param None
- return flag for reset(true: success, false: fail)

**bool** resetFD (**FDRecord** &fdRcd)

Reset FD record

- param fdRcd record struct
- return flag for reset(true: success, false: fail)

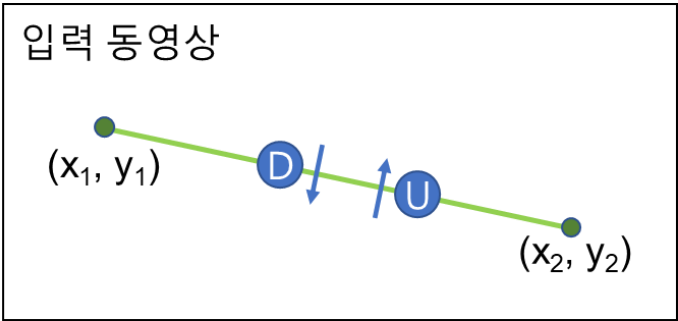
**bool** resetFDRecord()

Reset fd record

- param None
  - return flag for reset(true: success, false: fail)
-

## 5. config.json 파일 설정

구분	Name	Value
global	apikey	솔루션 사용 키(aiprotest 고정)
	frame_limit	처리할 최대 프레임 수
	input_files	비디오 입력(경로 포함)
	output_files	비디오 출력(경로 포함)
od	enable	물체 검출 On/Off
	score_th	물체 검출에 이용되는 기준 Score 값
fd	enable	화재 검출 On/Off
	score_th	화재 검출에 이용되는 기준 Score 값
par	enable	PAR 검출 On/Off
pose	enable	Pose 검출 On/Off
	score_th	Skeleton 을 그릴 때 이용되는 기준 Score 값
act	enable	Action 검출 On/Off
	score_th	Act 검출 신뢰도 기준 Score 값
line	param	<p>사용하는 Counting Line 정보를 아래 형식으로 순차적으로 입력</p> <p>[line_id vchID x1 y1 x2 y2]</p> <ul style="list-style-type: none"> <li>- line_id: 고유 ID 값</li> <li>- vchID: 속하는 Video Channel ID</li> <li>- x1, y1: 직선을 구성하는 한점(Point 1)</li> <li>- x2, y2: 직선을 구성하는 다른 한점(Point 2)</li> </ul>

		<p>입력 동영상</p>  <p>(카운팅 예) UP: 남자 3(1/2/0), 여자 2(0/1/1) Down: 남자 5(2/2/1), 여자 1(0/1/0)</p>
zone	param	<p>사용하는 Zone 정보를 아래 형식으로 순차적으로 입력</p> <p>[zone_id vchID isRestricted x1 y1 x2 y2 x3 y3 x4 y4]</p> <ul style="list-style-type: none"> <li>- zone_id: 고유 ID 값</li> <li>- vchID: 속하는 Video Channel ID</li> <li>- isRestricted: 접근 제한 구역(내부적으로 미사용) <ul style="list-style-type: none"> <li>- x1 y1 x2 y2 x3 y3 x4 y4: Zone 의 4 개 꼭지점 좌표(연속되는 방향으로 입력)</li> </ul> </li> </ul>

## 6. 사람 속성 검출

- runModel() 함수에서 DetBox 객체의 PedAtts 와 다른 멤버 변수에 속성 입력

속성	설명
성별	<p>남/여 성별 검출(정확도 96%)</p> <ul style="list-style-type: none"> <li>- DetBox-PedAtts-atts[0]: 0:Male, 1:Female</li> </ul>
연령층	<p>어린이/성인/노인 세 경우로 구분해 검출(정확도 85%)</p> <ul style="list-style-type: none"> <li>- DetBox-PedAtts-atts[1]: confidence to be child</li> <li>- DetBox-PedAtts-atts[2]: confidence to be adult</li> <li>- DetBox-PedAtts-atts[3]: confidence to be elder</li> </ul>

전체 속성 (옷차림, 색상, 소지품 등)	<pre> #define NUM_ATTRIBUTES 30  /// number of attributes #define ATT_GENDER 0      /// gender should be the #define ATT_AGE_CHILD 1 #define ATT_AGE_ADULT 2 #define ATT_AGE_ELDER 3 #define ATT_HAIR_LEN_SHORT 4 #define ATT_HAIR_LEN_LONG 5 #define ATT_UBODY_LEN_SHORT 6 #define ATT_UBODY_COL_BLACK 7 #define ATT_UBODY_COL_BLUE 8 #define ATT_UBODY_COL_BROWN 9 #define ATT_UBODY_COL_GREEN 10 #define ATT_UBODY_COL_GRAY 11 #define ATT_UBODY_COL_PINK 12 #define ATT_UBODY_COL_PURPLE 13 #define ATT_UBODY_COL_RED 14 #define ATT_UBODY_COL_WHITE 15 #define ATT_UBODY_COL_YELLOW 16 #define ATT_UBODY_COL_OTHER 17 #define ATT_LBODY_LEN_SHORT 18 #define ATT_LBODY_COL_BLACK 19 #define ATT_LBODY_COL_BLUE 20 #define ATT_LBODY_COL_BROWN 21 #define ATT_LBODY_COL_GRAY 22 #define ATT_LBODY_COL_WHITE 23 #define ATT_LBODY_COL_OTHER 24 #define ATT_LBODY_TYPE_TROUSER_SHORT 25 #define ATT_LBODY_TYPE_SKIRT_DRESS 26 #define ATT_BACKPACK 27 #define ATT_BAG 28 #define ATT_HAT 29 </pre>
움직임	<p>동작 활성화도 측정(실신, 쓰러짐, 숙면 등 탐지)</p> <ul style="list-style-type: none"> <li>- DetBox-distVar: box center variation after temporal pooling</li> </ul>
등장 시간	<p>영상에 등장 후 머문 시간 검출(배회 여부 검출)</p> <ul style="list-style-type: none"> <li>- DetBox-inTime: time when this object is detected</li> </ul>
이동 방향	<p>동선 검출</p> <ul style="list-style-type: none"> <li>- DetBox-(rxP, ryP): reference position in the previous frame</li> </ul>

## 7. 동작 인식

- runModel() 함수에서 DetBox 객체의 Action 정보 입력
  - DetBox-actID: action ID in actIDMapping

- DetBox-actConf: confidence of the current act

## ■ Action ID Mapping

17 가지 개별 동작 (ID: Label)			
0: Hand on mouth	1: Pick up	2: Throw	3: Sit down
4: Stand up	5: Clapping	6: Reading/writing	7: Hand wave
8: Kick	9: Cross hands	10: Staggering	11: Fall down
12: Punch/Slap	13: Push	14: Walk	15: Squat down
16: Run			

## 8. 화재 검출

- runModelFD() 함수에서 검출된 FireBox 객체의 정보를 fboxesMul 객체에 입력
  - Fire 와 Smoke 객체 검출(class ID 가 2 개)
- fdRcd 객체에 fire 발생 확률과 smoke 발생 확률을 각각 입력

## 9. 평균 추론 동작 시간

- runModel() 함수 내부에서 복잡도 측정
  - Part 0: OD + Tracking + PAR
  - Part 1: Pose Estimation + Action Recognition
  - Input Video Resolution: FHD, Model Input Resolution: 960x544, GPU: 2080Ti, CPU: i9-10900X@3.70GHz, Batch Size: 1 frame

OD+Track+PAR: 18ms/frame, POSE+ACT: 18ms/frame

```
C# Microsoft Visual Studio Debug Console
Frame 98>      OD+Track+PAR: 20ms      POSE+ACT: 20ms
Frame 98>      OD+Track+PAR: 16ms      POSE+ACT: 16ms
Frame 99>      OD+Track+PAR: 18ms      POSE+ACT: 18ms
Frame 99>      OD+Track+PAR: 16ms      POSE+ACT: 16ms
Frame 99>      OD+Track+PAR: 17ms      POSE+ACT: 17ms
Frame 99>      OD+Track+PAR: 14ms      POSE+ACT: 14ms
Frame 100>     OD+Track+PAR: 20ms      POSE+ACT: 20ms
Frame 100>     OD+Track+PAR: 17ms      POSE+ACT: 17ms
Frame 100>     OD+Track+PAR: 17ms      POSE+ACT: 17ms
Frame 100>     OD+Track+PAR: 16ms      POSE+ACT: 16ms

Average Inference Time> OD+Track+PAR: 18ms      POSE+ACT: 18ms

Output file(s):
    videos/in0_o.mp4
    videos/in1_o.mp4
    videos/in2_o.mp4
    videos/in3_o.mp4

Terminate program!
```