

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, Fire Classification, Crowd Counting
사용 언어	C/C++
OS	Windows 11
설치환경	Visual Studio 2022 CUDA 12.1.0 cuDNN 8.9.6
Demo Dependency	OpenCV-4.9.0(자체 포함), TensorRT-8.6.1.6 (자체 포함), Opencvino-2023.2.0(자체포함)
GPU 최소 사양	RTX 30xx 필수

작성일: 2024 년 11 월

작성자: 박 천 수

이메일: cspk@skku.edu

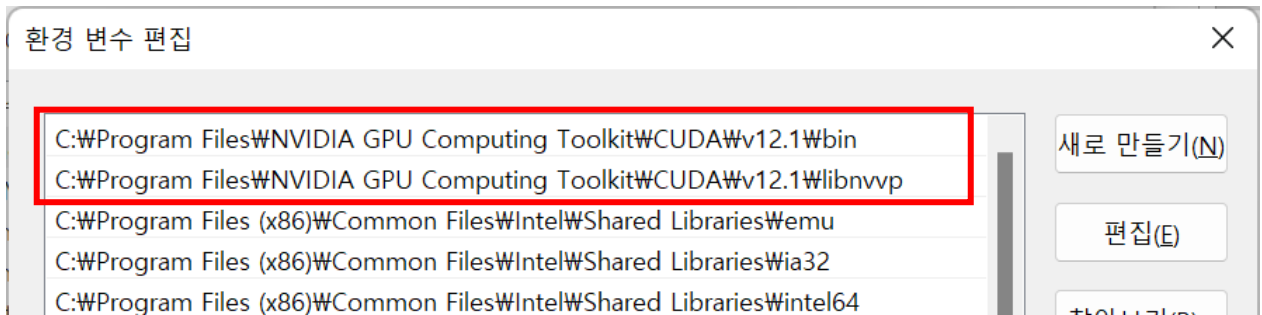
Part I. Demo 프로그램 설치 가이드

1. Visual Studio 2022 설치

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

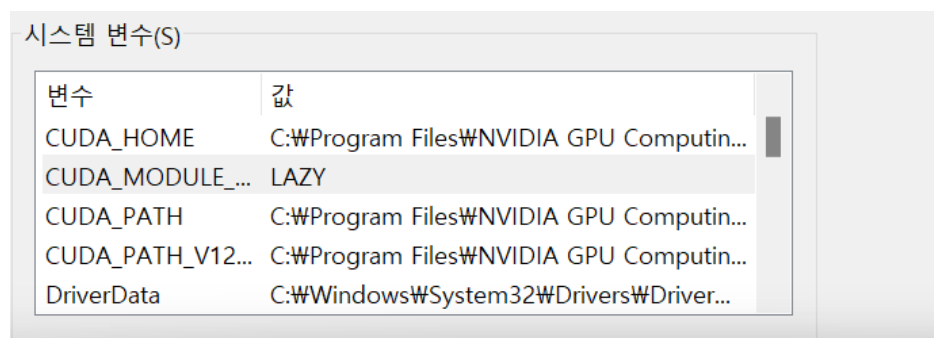
2. CUDA 설치

- A. CUDA Toolkit 12.1.0 설치
 - 링크: <https://developer.nvidia.com/cuda-toolkit-archive>
 - 설치 파일: cuda_12.1.0_531.14_windows.exe
 - 설치 시 기본 경로 사용 권장
- B. PATH 환경 변수에 아래 내용 추가
 - C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v12.1\bin
 - C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v12.1\libnvvp



C. Lazy loading 환경 변수 추가

- 변수 이름: CUDA_MODULE_LOADING, 변수 값: LAZY
- 추가 후 재부팅 필요
- 관련 내용: <https://docs.nvidia.com/cuda/cuda-c-programming-guide/#lazy-loading>



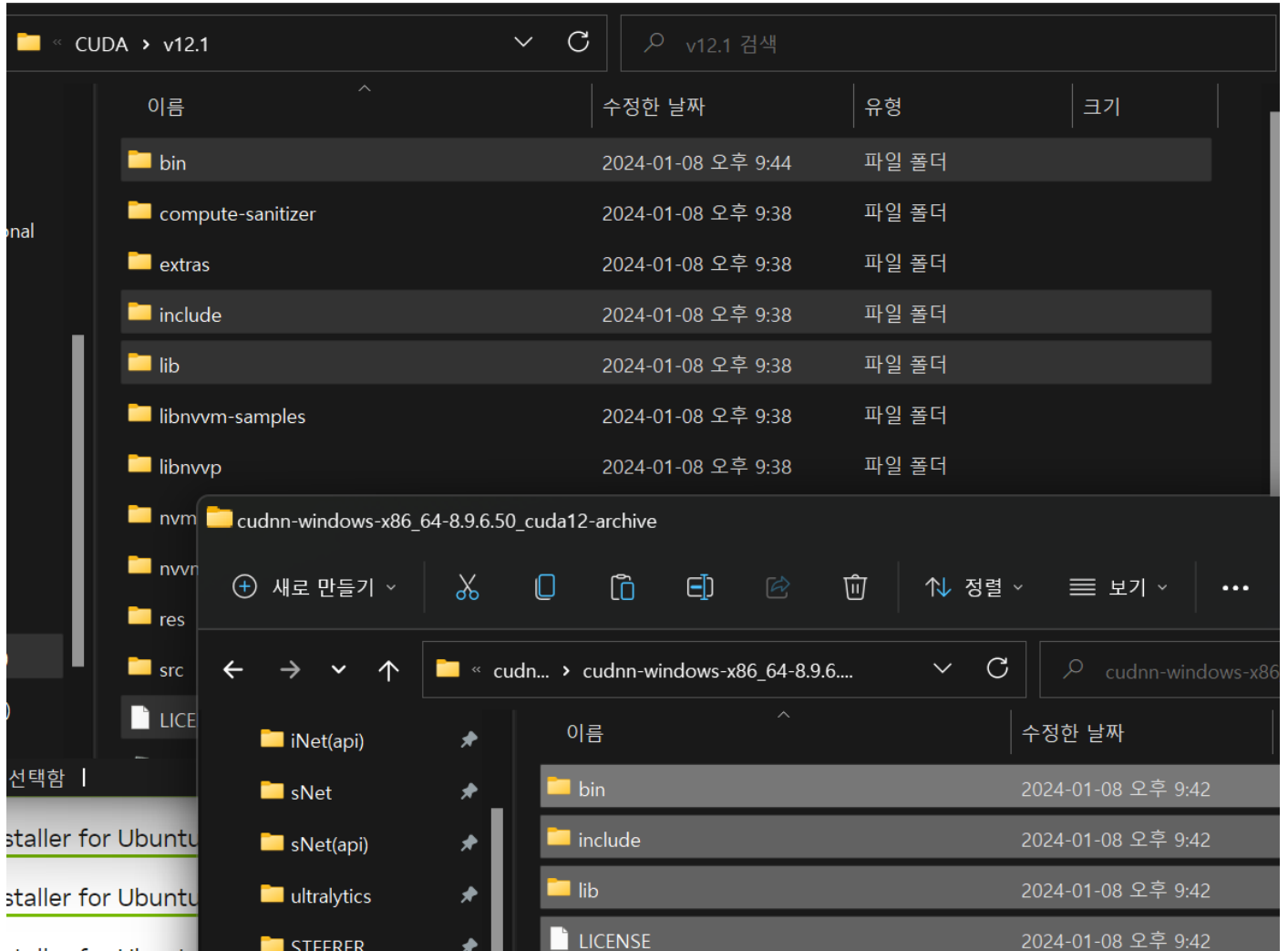
시스템 변수 편집

변수 이름(N):	CUDA_MODULE_LOADING
변수 값(V):	LAZY

3. cuDNN 설치

A. cuDNN 8.9.6 설치

- 설치링크: <https://developer.nvidia.com/rdp/cudnn-download>
- 설치 파일: cudnn-windows-x86_64-8.9.6.50_cuda12-archive.zip
 - 다운 받기위해 NVIDIA 회원 가입이 필요
- 압축을 풀고 bin, include, lib 폴더를 “C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v12.1” 위치에 덮어쓰는 방식으로 복사



4. Visual Studio 2022 설정

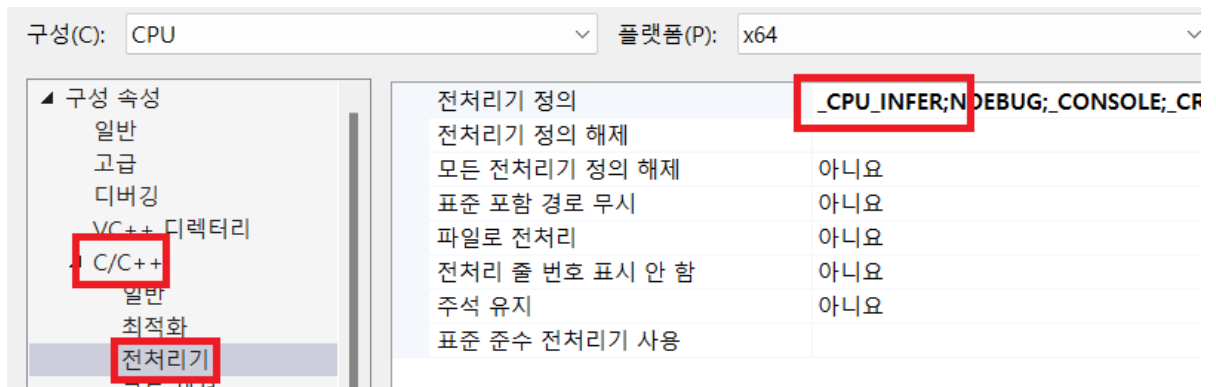
A. iNet-API-Demo Repository 클론 또는 복사

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

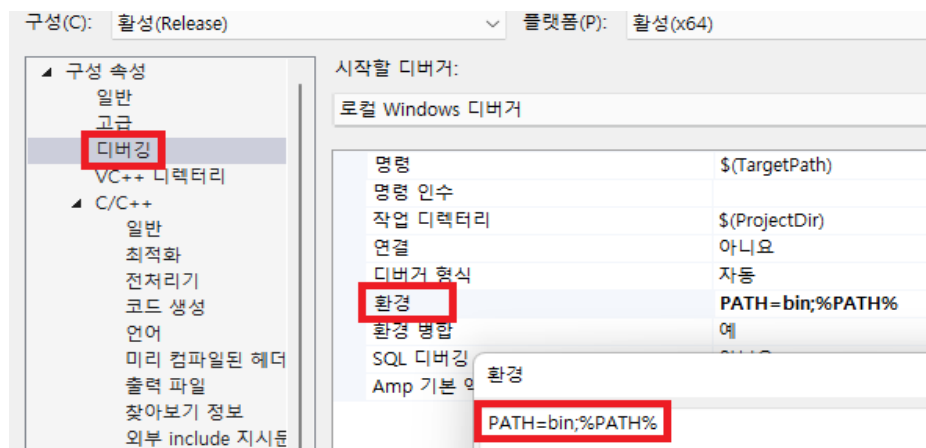
B. iNet-API-Demo.sln 솔루션을 열고 프로젝트 실행 환경 설정

- 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

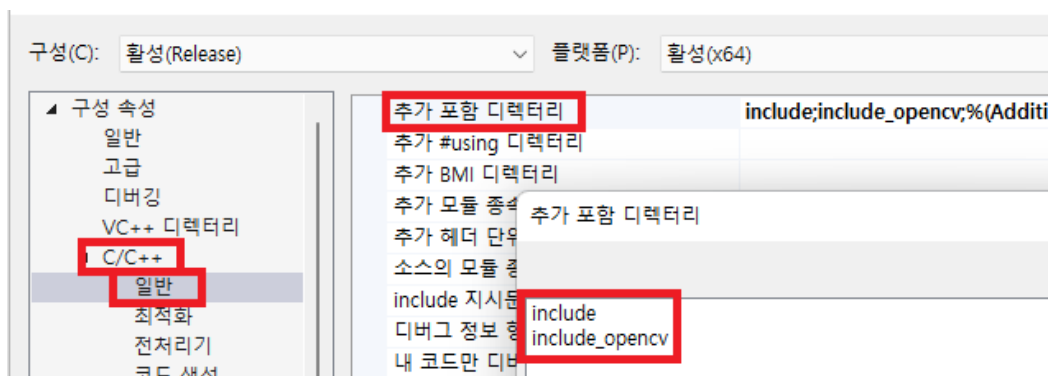
- 이동경로: C/C++ → 전처리기 → 전처리기 정의
- `_CPU_INFER` 전처리문 추가



- 다음 4.C 장부터는 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%`

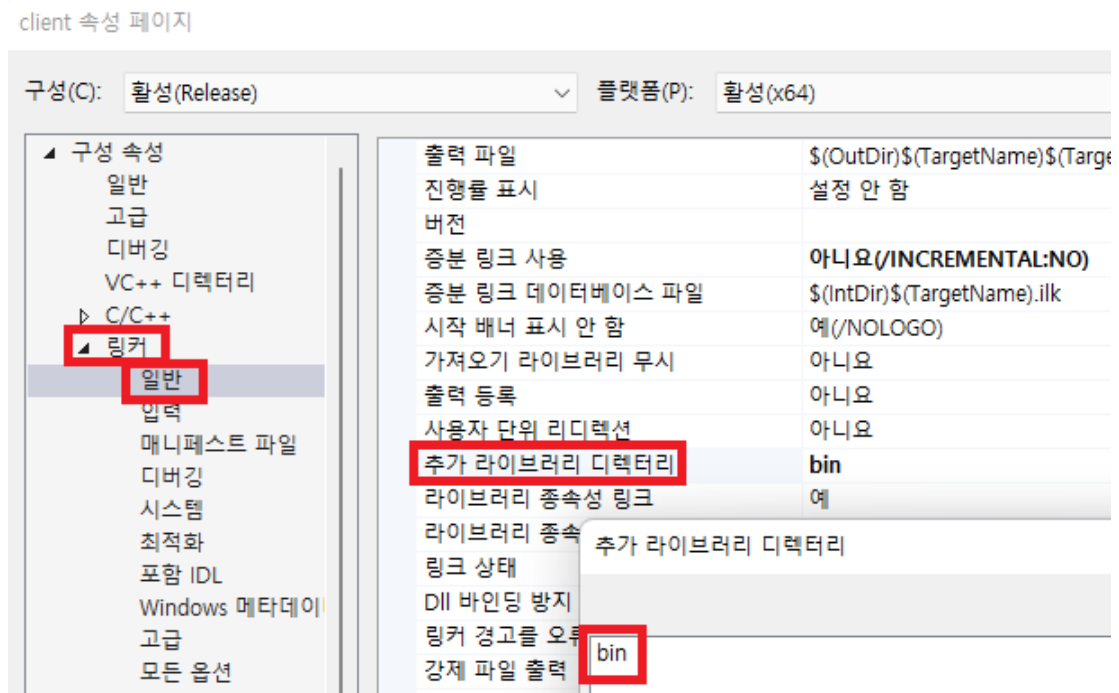


- D. 추가 포함 디렉터리 입력
 - 이동 경로: C/C++ → 일반 → 추가 포함 디렉터리
 - 추가 포함 디렉터리에 “include”와 “include_opencv” 입력



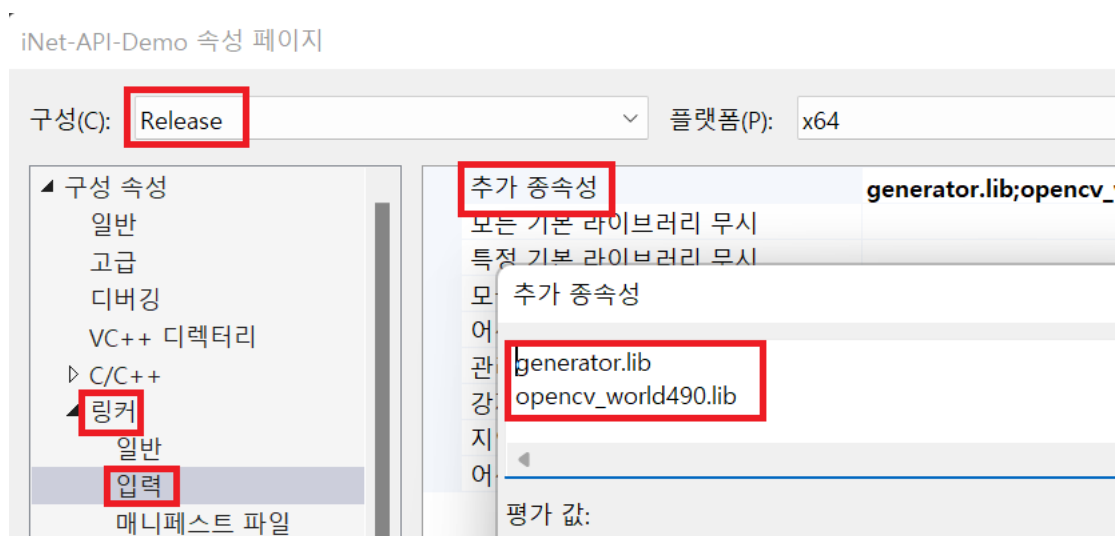
E. 추가 라이브러리 디렉터리 입력

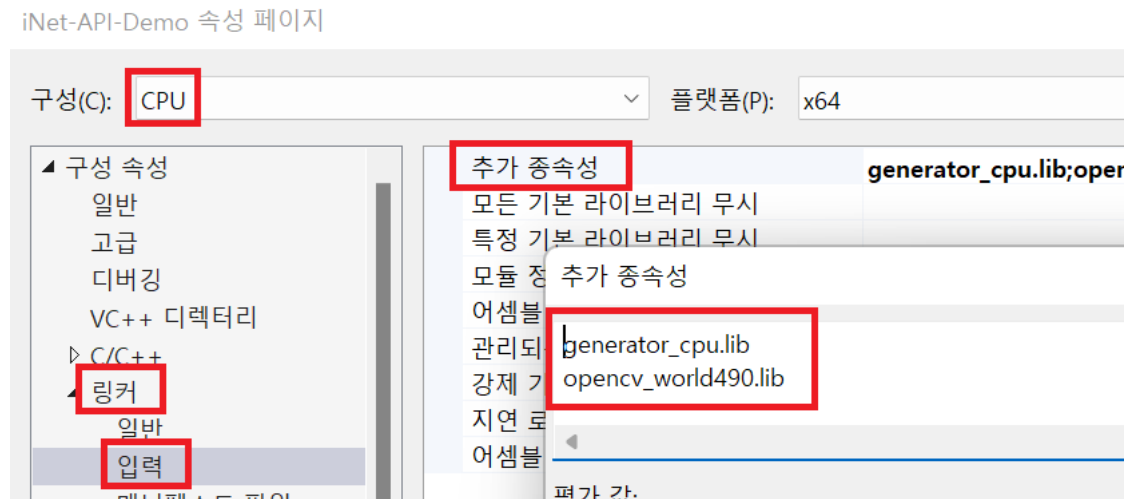
- 이동 경로: 링커 → 일반 → 추가 라이브러리 디렉터리
- 추가 라이브러리 디렉터리에 “bin” 입력



F. 추가 종속성 입력

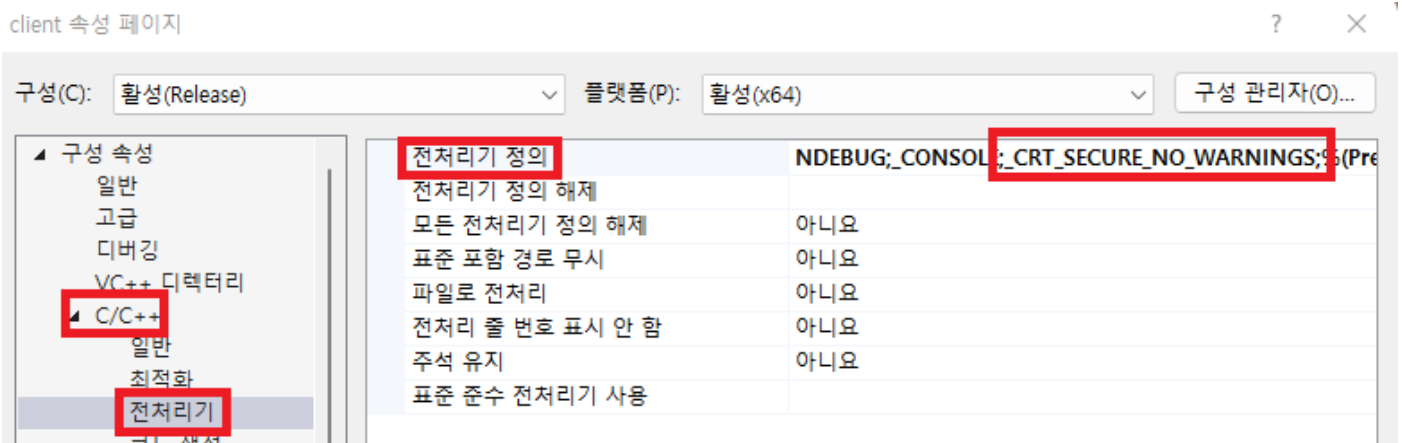
- 이동 경로: 링커 → 입력 → 추가 종속성
- 추가 종속성 편집(모드마다 다름)
 - Release 모드: “generator.lib”, “opencv_world490.lib”
 - CPU 모드: “generator_cpu.lib”, “opencv_world490.lib”





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



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

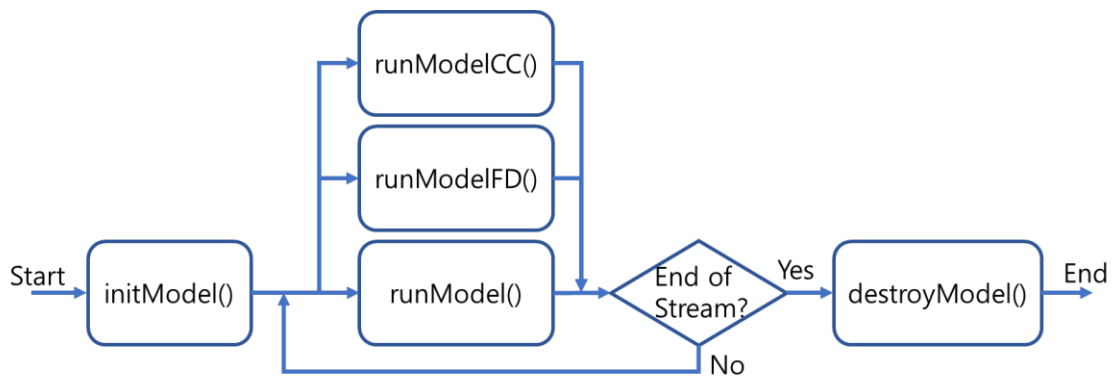
Part II. iNet Solution 사용 가이드

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"> • 솔루션 실행에 필요한 모델과 내부 저장소를 초기화
실행 1	runModel()	<ul style="list-style-type: none"> • frame 을 입력 받고 추론 동작 수행 • Object Detection, Tracking, Counting(Line & Zone), PAR 동작을 담당 • 검출된 사람 객체(dboxes)를 리턴
실행 2	runModelFD()	<ul style="list-style-type: none"> • frame 을 입력 받고 Fire Classification (FC) 추론 동작 수행(하위 호환을 위해 함수 이름은 FD 로 유지) • Fire 와 Smoke 분류 동작 성공 여부 리턴
실행 3	runModelCC	<ul style="list-style-type: none"> • frame 을 입력 받고 Crowd Counting(CC) 추론 동작 수행 • 검출된 density map 과 현재 프레임의 총 사람 수를 리턴
소멸	destroyModel()	<ul style="list-style-type: none"> • 생성된 모델과 저장소 공간 해제

- 초기화와 소멸 단계는 프로그램 시작시와 종료시에 각 1 번씩 호출되고, 실행 단계는 runModel(), runModelFD(), runModelCC()를 반복적으로 호출하며 실행



<그림> iNet 솔루션 동작 플로우

2. 솔루션을 이용한 프로그램 개발 작업 내용

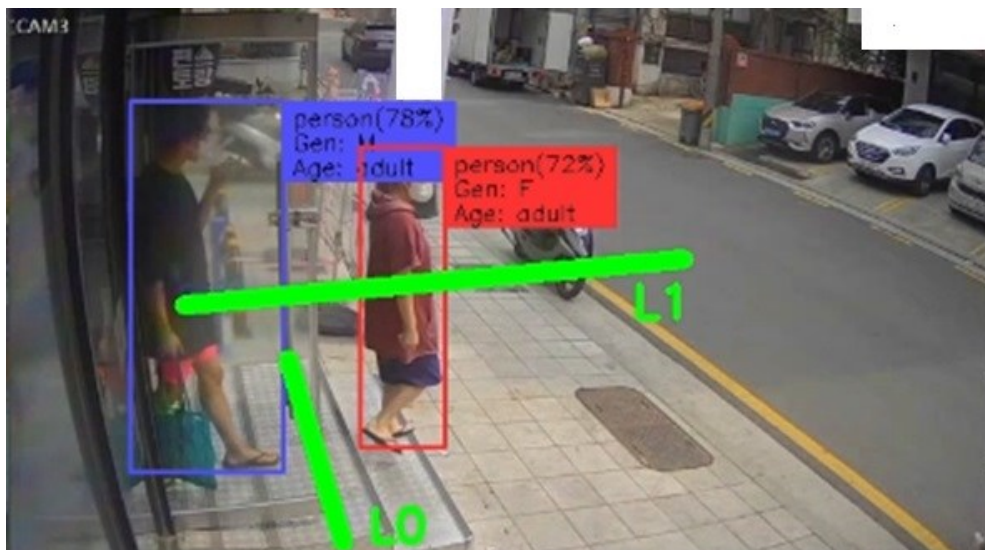
- 기본적으로 iNet 솔루션은 config.json 파일을 Parsing 해 Config 객체(cfg)를 생성하고 이를 이용해 전체 솔루션을 동작 시킴
 - config.json 의 models 필드를 통해 로딩되는 모델을 변경 가능
- config.json 파일을 설정 후 예제 코드와 같은 방식으로 초기화, 실행, 소멸 동작을 수행해야 함
- 현재 boost_mode 의 경우 최대 빠르기로 추론 동작을 수행하고, 그 외 경우 일정 시간동안 연속으로 이벤트가 발생하지 않을 시 저전력 방식(idle 모드)으로 동작

3. Counting 정보

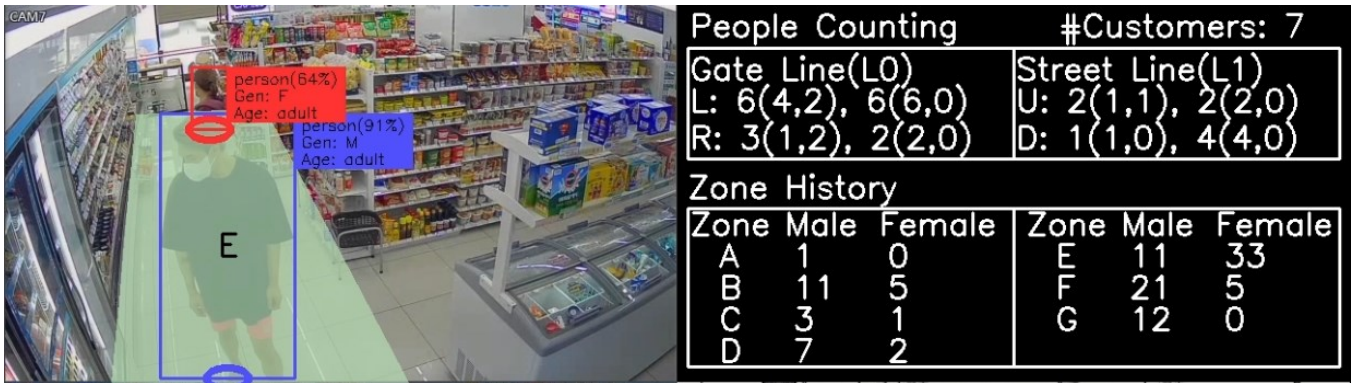
- 기본적으로 CntLine 을 지나가는 사람 수와 Zone 내부에 머무르는 사람 수를 카운팅. 사람 수를 셀 때는 성별(남, 여)과 연령층(어린이, 성인, 노인)을 고려해 총 6 가지 경우를 각각 카운팅
 - 각 카운팅 정보는 2 x 3 배열에 [남/여][어린이/성인/노인] 방식으로 저장되고 아래와 같은 형식으로 출력됨

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)



<그림> Line Counting & Zone Hitmap Example

4. API 함수 설명

bool initModel(**Config** &cfg)

Initialize model

- param cfg configuration struct

bool runModel(std::vector<**DetBox**> &dboxes, **int** &minObjSize, **ODRecord** &odRcd, **MinObj** &minObj, cv::**Mat** &frame, **int** vchID, **uint** frameCnt, **float** odScoreTh);

Run OD and PAR models for a single frame

- param dboxes return detected dboxes of the vchID video channel
- param minObjSize return the size of a detected small object
- param odRcd object detection record struct
- param minObj minimum size object deletion struct
- param frame input frame
- param vchID vchID of the input frame
- param frameCnt frameCnt of the input frame
- param odScoreTh threshold for filtering low confident object detections
- return runModel result(true: success, false: fail)

bool runModelFD(**FDRecord** &fdRcd, cv::**Mat** &frame, **int** vchID)

Run FD models for a single frame

- param frame input frame
- param fdRcd fire detection record struct
- param vchID vchID of the input frame
- return flag for the running result(true: success, false: fail)

bool runModelCC(cv::**Mat** &density, **CCRecord** &ccRcd, cv::**Mat** &frame, **int** vchID)

Run Crowd Counter for a single frame

- param density return the density of people
- param ccRcd crowd counting record struct
- param frame input frame
- param vchID vchID of the input frame
- param ccScoreTh threshold for filtering low confident detections
- return flag for the running result(true: success, false: fail)

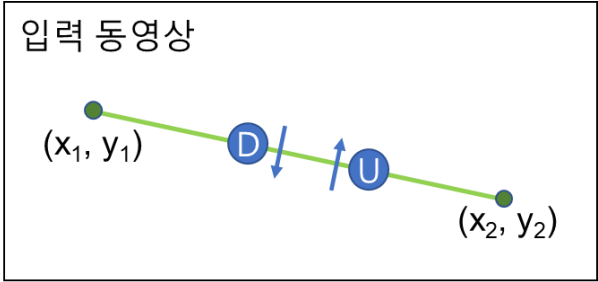
bool destroyModel()

Destroy model

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

5. config.json 파일 설정

구분	Name	Value
models	od_gpu	Object Detection 모델(GPU 버전)
	od_cpu	Object Detection 모델(CPU 버전)
	fd_gpu	Fire Classification 모델(GPU 버전)
	fd_cpu	Fire Classification 모델(CPU 버전)
	cc_gpu	Crowd Counting 모델(GPU 버전)
	cc_cpu	Crowd Counting 모델(CPU 버전)
	par_gpu	PAR 모델(GPU 버전)
	par_cpu	PAR 모델(CPU 버전)
global	apikey	솔루션 사용 키(aiprotect 고정)
	frame_limit	처리할 최대 프레임 수 - -1 로 설정하면 제한 없음
	recording	출력 프레임 녹화 On/Off
	debug_mode	출력 프레임에 debug 정보 포함 On/Off - 내부 사용(Test 경우 false 로 설정)
	boost_mode	엔진 성능 최대화 On/Off
	igpu_enable	설치 Device 의 CPU 가 내장 GPU 를 포함하는 경우 Enable
	input_files	로컬 비디오 파일 입력(경로 포함)
	output_files	로컬 비디오 파일 출력(경로 포함)
od	enable	물체 검출 On/Off
	score_th	물체 검출에 이용되는 기준 Score 값
	min_obj_params	Small object 제거 설정 파라미터를 아래 형식으로 순차적으로 입력 [vchID mode partition_0 partition_1 partition_2 partition_3] - vchID: 적용 속하는 Video Channel ID - mode: 0:none, 1: in dll, 2: in draw - partition_x: 해당 파티션의 크기 임계값

fd	enable	화재 검출 On/Off
	score_th_fire	fire 객체 검출에 이용되는 기준 Score 값(draw 에서 사용)
	score_th_smoke	smoke 객체 검출에 이용되는 기준 Score 값(draw 에서 사용)
Par	enable	PAR 검출 On/Off
line	param	<p>사용하는 Counting Line 파라미터를 아래 형식으로 순차적으로 입력</p> <p>[lineID vchID isMode x1 y1 x2 y2]</p> <ul style="list-style-type: none">- lineID: 고유 ID 값- vchID: 속하는 Video Channel ID- isMode: IS mode(0: people counting, 1: restricted area)- x1, y1: 직선을 구성하는 한점(Point 1)- x2, y2: 직선을 구성하는 다른 한점(Point 2) <div><p>입력 동영상</p><p>(카운팅 예)</p><p>UP: 남자 3(1/2/0), 여자 2(0/1/1)</p><p>Down: 남자 5(2/2/1), 여자 1(0/1/0)</p></div>
zone	param	<p>사용하는 Zone 파라미터를 아래 형식으로 순차적으로 입력</p> <p>[zoneID vchID isMode x1 y1 x2 y2 x3 y3 x4 y4]</p> <ul style="list-style-type: none">- zoneID: 고유 ID 값- vchID: 속하는 Video Channel ID- isMode: IS mode(0: people counting, 1: restricted area)- x1 y1 x2 y2 x3 y3 x4 y4: Zone 의 4 개 꼭지점 좌표(연속되는 방향으로 입력)
cc_zone	param	사용하는 ccZone 파라미터를 아래 형식으로 순차적으로 입력

		<div>[ccZoneID vchID x1 y1 x2 y2 x3 y3 x4 y4 ccLevelTh1, ccLevelTh2, ccLevelTh3]</div> <div><div>- LevelTh1: Leve1 임계 값(초과면 Level1)</div><div>- LevelTh2: Level2 임계 값(초과면 Level2)</div><div>- LevelTh3: Level3 임계 값(초과면 Level3)</div></div>
--	--	---

6. 사람 속성 검출

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

속성	설명
성별	남/여 성별 검출(정확도 96%) - DetBox-PedAtts-atts[0]: 0:Male, 1:Female
연령층	어린이/성인/노인 세 경우로 구분해 검출(정확도 85%) - DetBox-PedAtts-atts[1]: confidence to be child - DetBox-PedAtts-atts[2]: confidence to be adult - DetBox-PedAtts-atts[3]: confidence to be elder
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	동작 활성화도 측정(실신, 쓰러짐, 숙면 등 탐지) - DetBox-distVar: box center variation after temporal pooling

등장 시간	영상에 등장 후 머문 시간 검출(배회 여부 검출) - DetBox-inTime: time when this object is detected
이동 방향	동선 검출 - DetBox-(rxP, ryP): r`eference position in the previous frame

7. 화재 검출

- runModelFD() 함수에서 분류된 화재 정보를 FDRecord 객체에 저장
- FDRecord 객체에 현재 프레임의 fire 와 smoke 이벤트 발생 여부, 시간 축에서 fire 와 smoke 발생 확률을 각각 저장

```

struct FDRecord {
    int vchID;
    std::deque<float> fireProbs;    // fire probability
    std::deque<float> smokeProbs;  // smoke probability
    int fireEvent;                 // fire event
    int smokeEvent;                // smoke event
    int afterFireEvent;            // for internal usage in external server
};

```

8. 군중 집계

- runModelCC() 함수에서 입력된 영상의 군중 수를 집계
 - GPU 경우: 프레임의 총 인원과 최대 2 개의 ROI 설정 지원
 - CPU 경우: 1 개의 ROI 설정 지원
- 프레임 전체 영역 기준으로 최대 2 만명까지 집계 지원

```

struct CCRecord {
    int vchID;
    std::deque<int> ccNumFrames;    // people in the whole frame
    std::vector<CCZone> ccZones;   // ccZones
};

```

9. 평균 추론 동작 시간

- runModel() 함수 내부에서 복잡도 측정
 - Part 0(OD + Tracking + PAR): 18ms
 - Part 1(FD): 4ms
 - Part 2(CC_GPU): 45ms
 - Input Video Resolution: FHD, GPU: 3090, CPU: i9-10900X@3.70GHz, Batch Size: 1 frame

Average Delay: 67ms (OD: 18ms, FD: 4ms, CC: 45ms)