㈜AIPro 기술문서 (Confidential)

AIPro sNet Solution Demo Guide

(API Version)

Part I. Demo 프로그램 설치 가이드

Part II. sNet Solution 사용 가이드

System Requirement & Dependency

항목	내용
포함 모델	Super Resolution
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)
GPU 최소 사양	RTX 3060 이상 필수

작성일: 2024 년 6 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

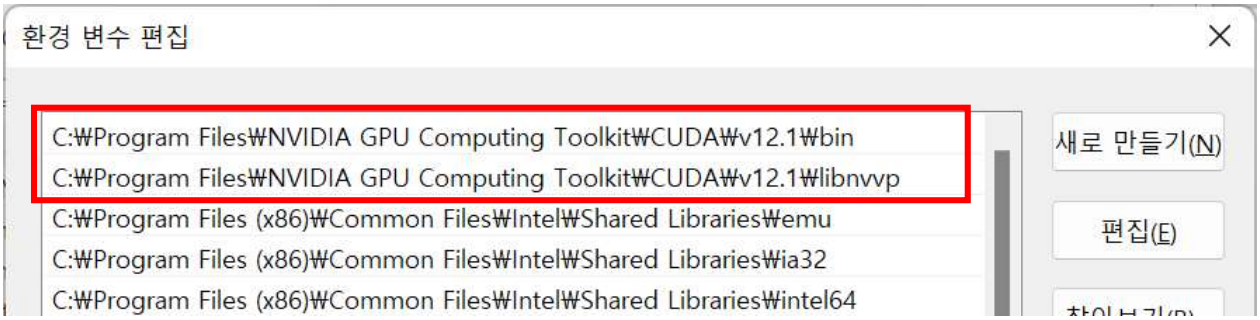
2. CUDA 설치

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

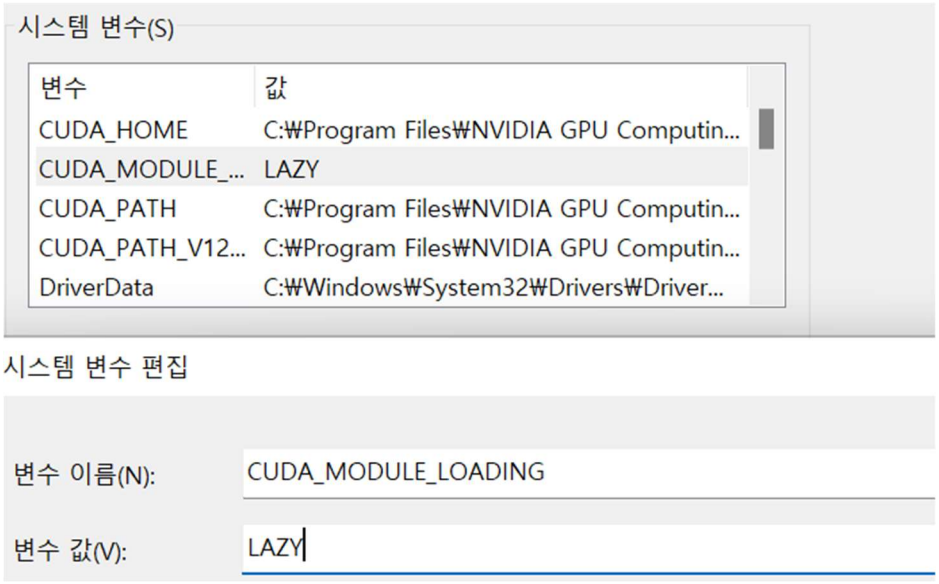
B. PATH 환경 변수에 아래 내용 추가

- 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

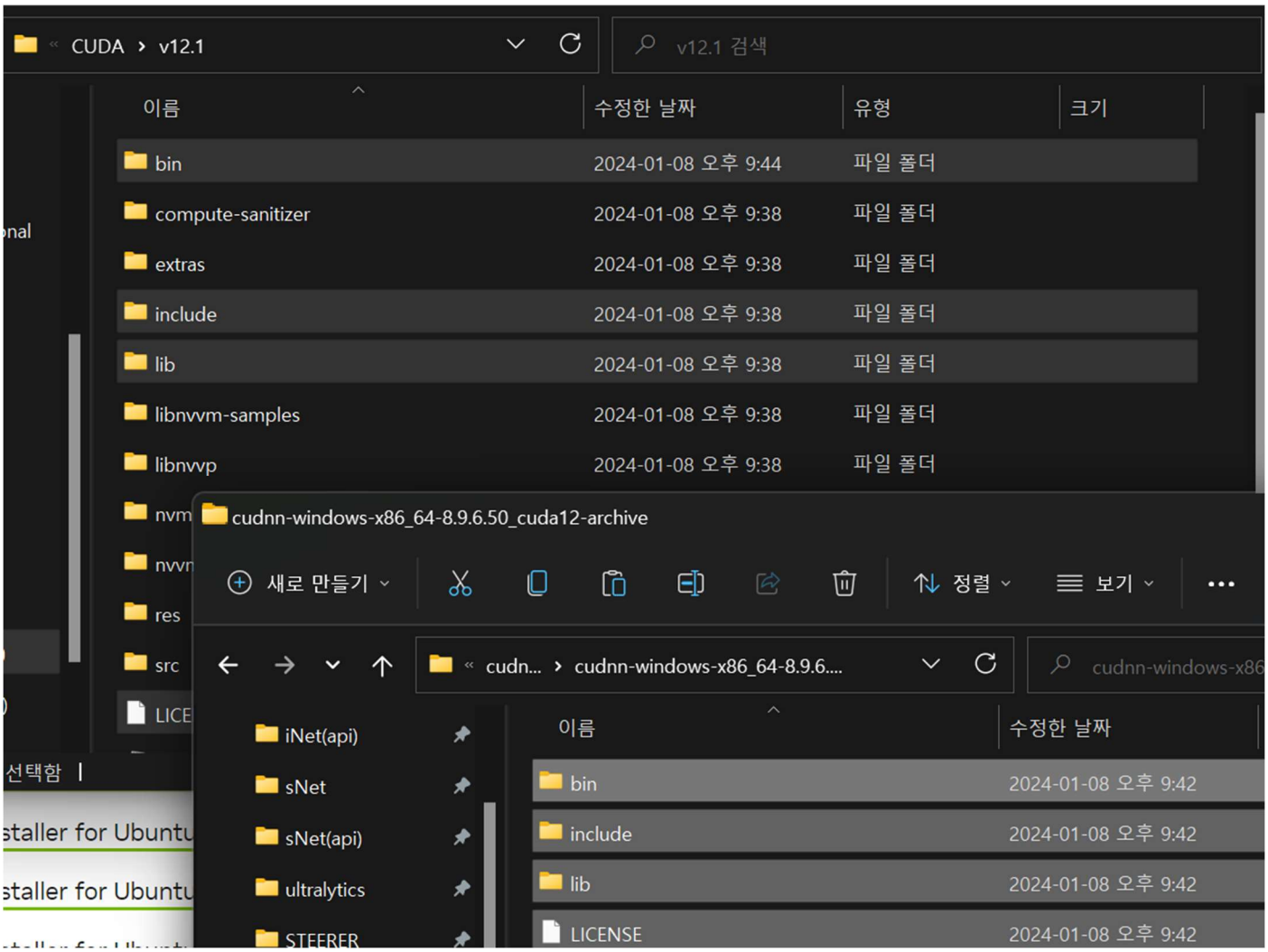
- 변수 이름: CUDA_MODULE_LOADING, 변수 값: LAZY
- 추가 후 재부팅 필요
- 관련 내용: https://docs.nvidia.com/cuda/cuda-c-programming-guide/#lazy-loading



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. sNet-API-Demo Repository 클론 또는 다운로드

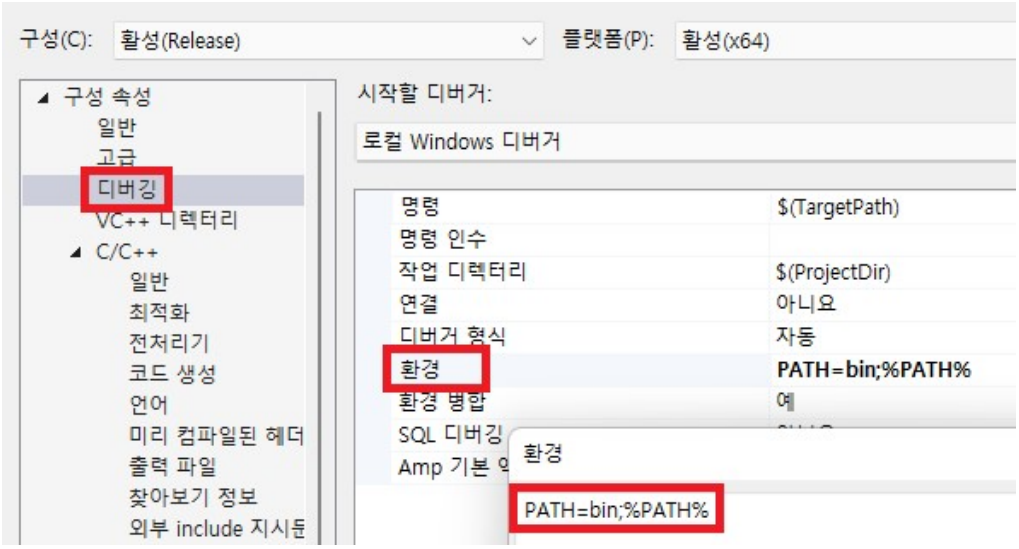
- Address: https://github.com/AIProCo/sNet-API-Demo

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

- 프로젝트 속성 페이지에서 구성을 “Release”, 플랫폼을 “x64”로 선택
- 실행할 때(디버깅 포함)도 "Release”, “x64”로 선택해야 함(Debug 모드 미지원)

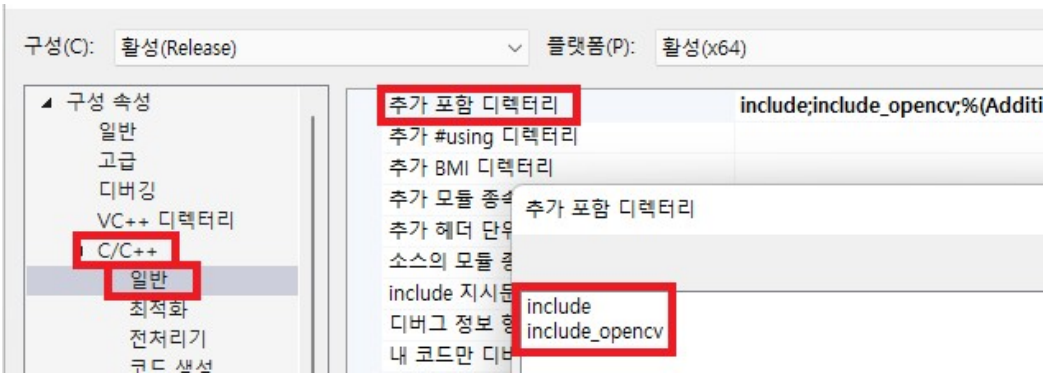
C. 솔루션 실행을 위한 로컬 PATH 변수 수정(시스템 PATH 변수에 영향 없음)

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



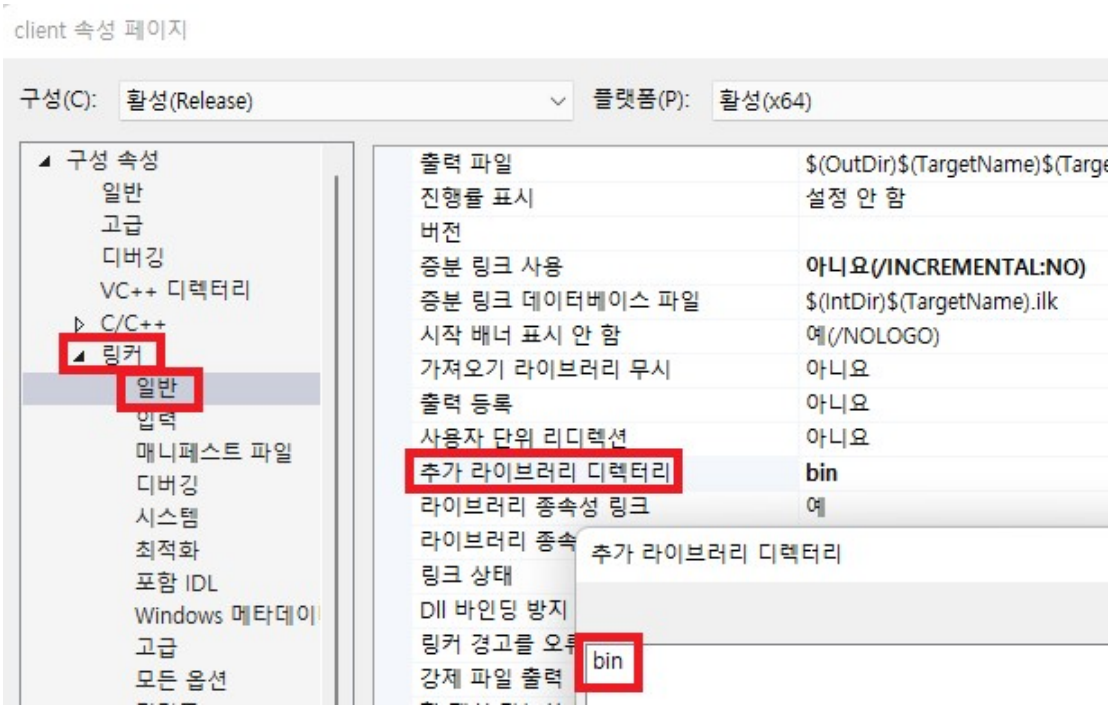
D. 추가 포함 디렉터리 입력

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



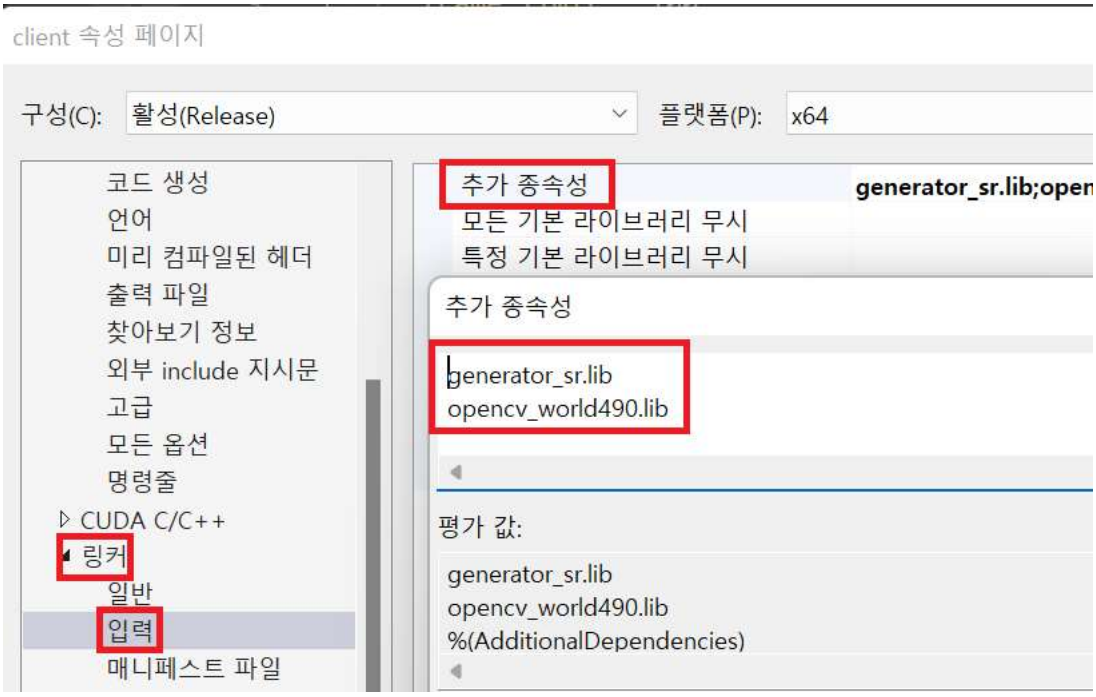
E. 추가 라이브러리 디렉터리 입력

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



F. 추가 종속성 입력

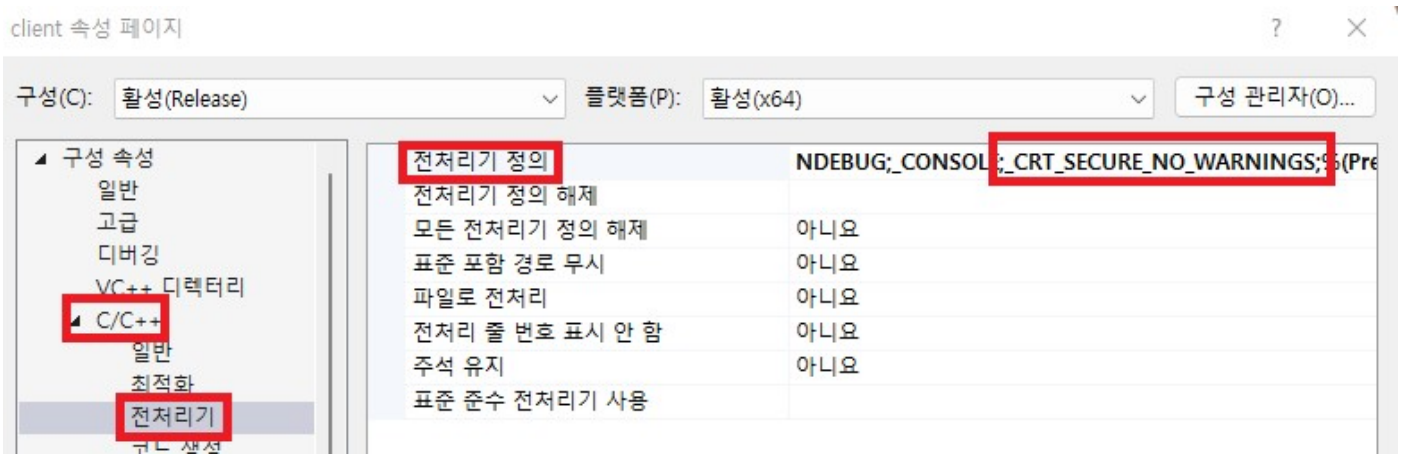
- 이동 경로: 링커 → 입력 → 추가 종속성
- 추가 종속성에 “generator_sr.lib”와 “opencv_world490.lib” 입력



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

A. "bin, inputs, videos(sNet).zip" 파일 다운로드 및 압축 해제. bin, inputs, videos 디렉토리를 솔루션 디렉토리(*.sln 파일과 같은 위치)로 복사

- Github repository 의 readme 문서에 포함된 링크 참고

6. Release 모드와 x64 플랫폼을 설정 후 실행

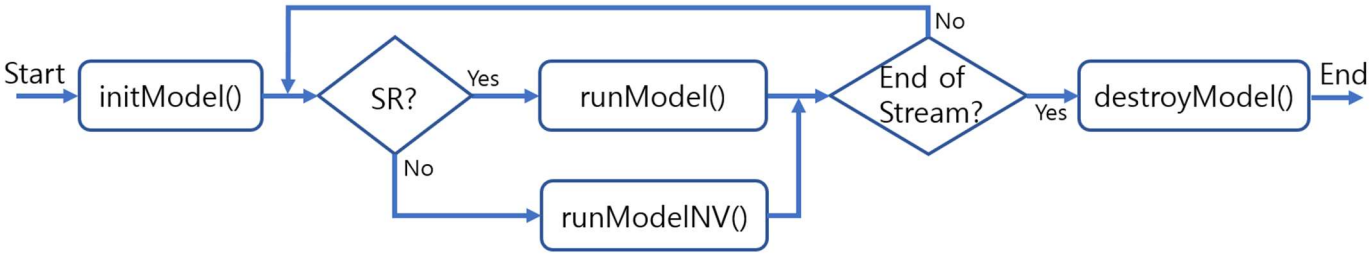
Part II. sNet Solution 사용 가이드

1. Solution 동작

■ AIPro sNet Solution 실행은 크게 생성, 실행, 소멸 3 단계로 이루어짐. 각 단계에 해당하는 함수와 세부 내용은 아래와 같음

단계	함수	내용
생성	initModel()	• SR 모델을 생성하는데 필요한 공통 파라미터 저장
실행 1	runModel()	• frame 을 입력 받고 SR 추론 동작 수행 • 고해상도 변환 영상(resFrame)을 출력
실행 2	runModelNV()	• frame 을 입력 받고 NV 추론 동작 수행 • 저조도 화질 개선 영상(resFrame)을 출력
소멸	destroyModel()	• 생성된 모델과 저장소 공간 해제

■ 초기화와 소멸 단계는 프로그램 시작시와 종료시에 각 1 번씩 호출되고, 실행 단계는 runModel() 또는 runModelNV()를 반복적으로 호출하며 실행



<그림> sNet 솔루션 동작 플로우

2. 솔루션을 이용한 프로그램 개발 작업 내용

■ 기본적으로 sNet 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 config.json 파일과 응용 프로그램 동작시 추출되는 데이터를 이용해 응용 프로그램에 맞는 cfg 객체를 생성 후 예제 코드와 같은 방식으로 실행 및 소멸 동작을 수행해야 함
- Test 할 기능에 맞게 변환 포맷 및 파라미터 입력 후 솔루션 실행
  - config.json 의 "scale_factor_x10" 파라미터를 이용해 대상 SR Model 또는 NV Model 자동 생성

3. API 함수 설명

```
bool initModel(std::string _srModelFileX2, std::string _srModelFileX1_5)
```

Initialize model

- param _srModelFileX2 x2 SR model file path
- param _srModelFileX1_5 x1.5 SR model file path
- return initialization result(true: success, false: fail)

```
bool runModel(Mat &frame, Mat &srFrame, int scaleFactorX10)
```

Run the SR model for an input frame

- param frame input frame
- param srFrame output SR frame
- param scaleFactorX10 scale factor(x10) from frame to srFrame
- return flag for the running result(true: success, false: fail)

```
bool runModelNV(Mat &frame, Mat &nvFrame)
```

Run the SR model for an input frame

- param frame input frame
- param nvFrame output NV frame
- 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)

4. config.json 파일 설정

구분	Name	Value
global	frame_limit	처리할 최대 프레임 수(-1: 파일 끝까지)
	input_files	비디오 입력(경로 포함)
	output_files	SR 비디오 출력(경로 포함)
	filter_enable	filter 기반 출력 Enable(SR 모드인 경우에만 동작) - Bilinear filter 방식을 이용해 입력 frame 을 upsample 하고 결과 영상을 저장
	scale_factor_x10	영상 변환 scaleFactor - 10: NV 모드 - 15: SR x1.5 모드 - 20: SR x2.0 모드

- Demo 프로그램의 동작 및 출력 영상 해상도는 입력 해상도와 scale_factor_x10 을 이용해 자동으로 계산됨
  - scale_factor_x10: 10→ NV 모드, 출력 해상도는 입력 해상도와 동일
  - scale_factor_x10: 15, 입력 해상도: 1280x720(HD), → SR 모드, 출력 해상도:1920x1080(FHD)
  - scale_factor_x10: 20, 입력 해상도: 1920x1080(FHD) → SR 모드, 출력 해상도:3840x2160(UHD)
- SR 의 경우 최대 1920x1080(FHD) 입력 해상도 지원
  - 가로, 세로 모두 FHD 크기 이하여야 함

5. 평균 추론 동작 시간

- runModel() 함수 평균 시간 복잡도 측정
  - GPU: RTX-3090, CPU: i9-10900X@3.70GHz
  - SD2FHD: 28ms/frame, HD2FHD: 45ms/frame
- runModelNV() 함수 평균 시간 복잡도 측정
  - GPU: RTX-3090, CPU: i9-10900X@3.70GHz
  - HD: 140ms/frame, FHD: 261ms/frame