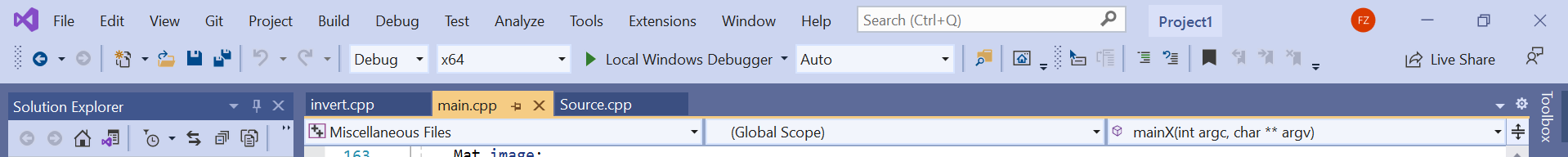
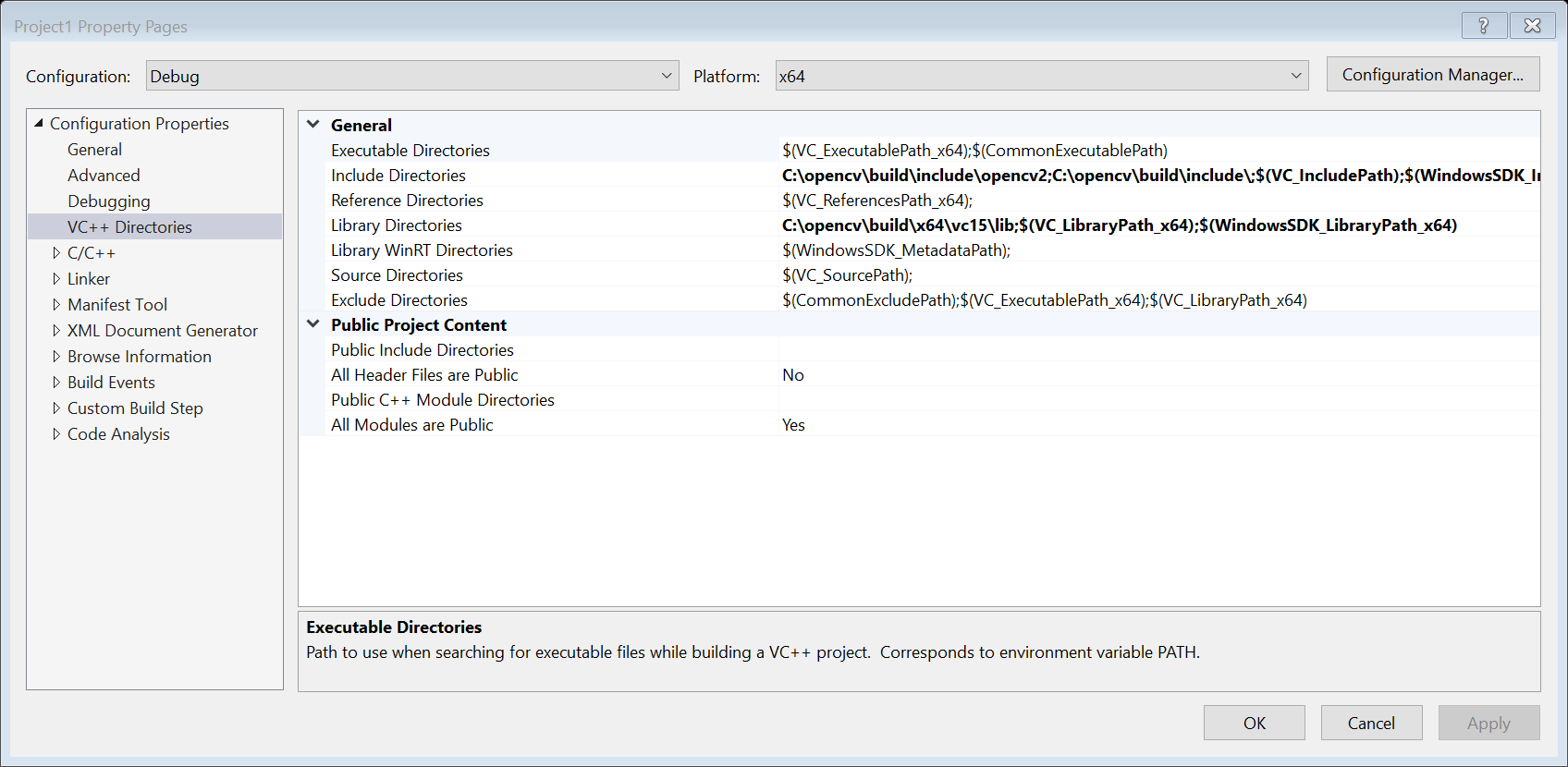
**OpenCV 4.5 setup on Windows 10/11 computers with Visual Studio 2017/2019/2022**

Fanglue Zhang

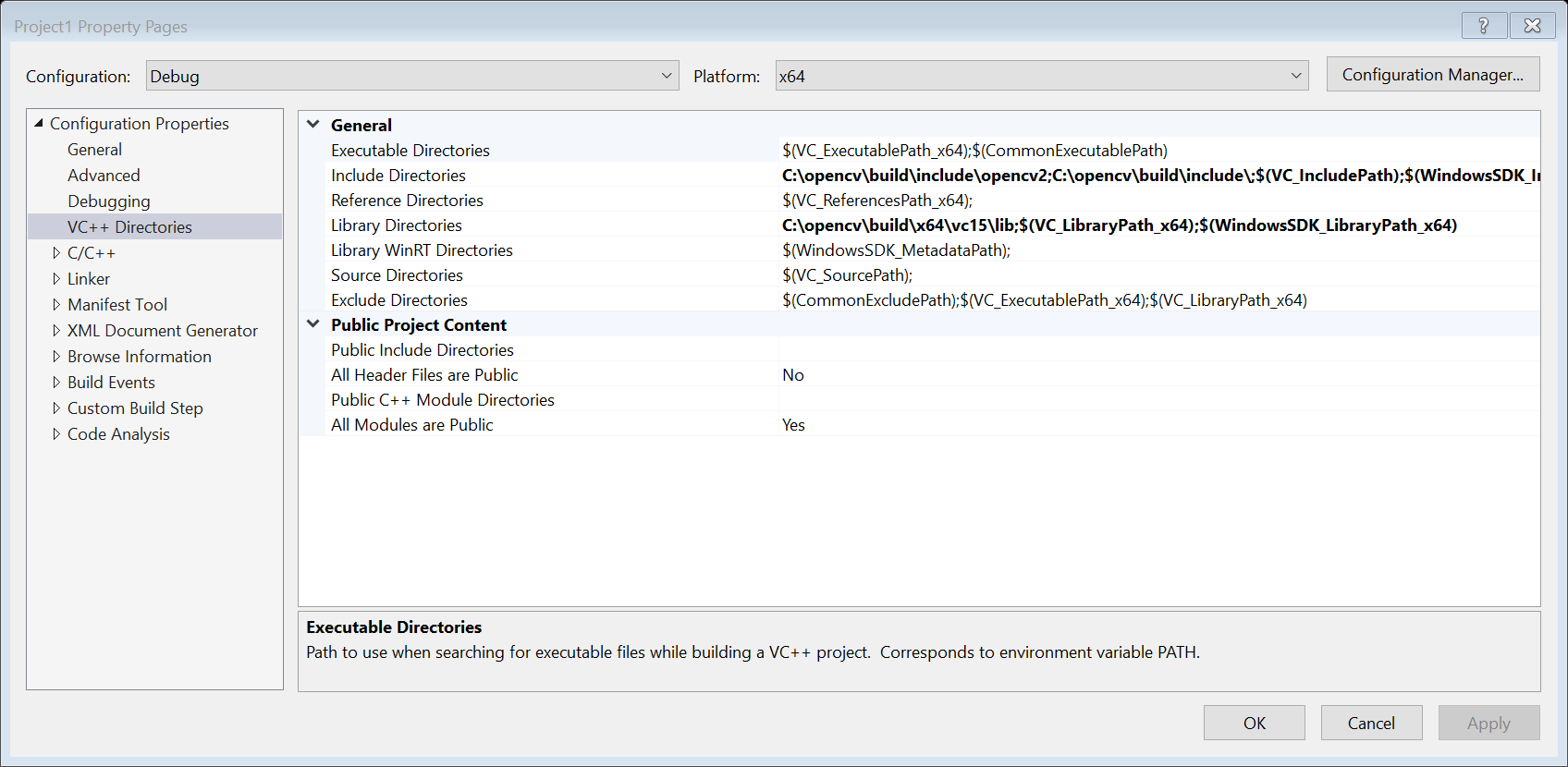
1. Create an empty C++ project. Make sure you are working for the “x64” system, and also make sure the configuration is for your working environment. In this document, I am using “Debug” mode for “x64” system.



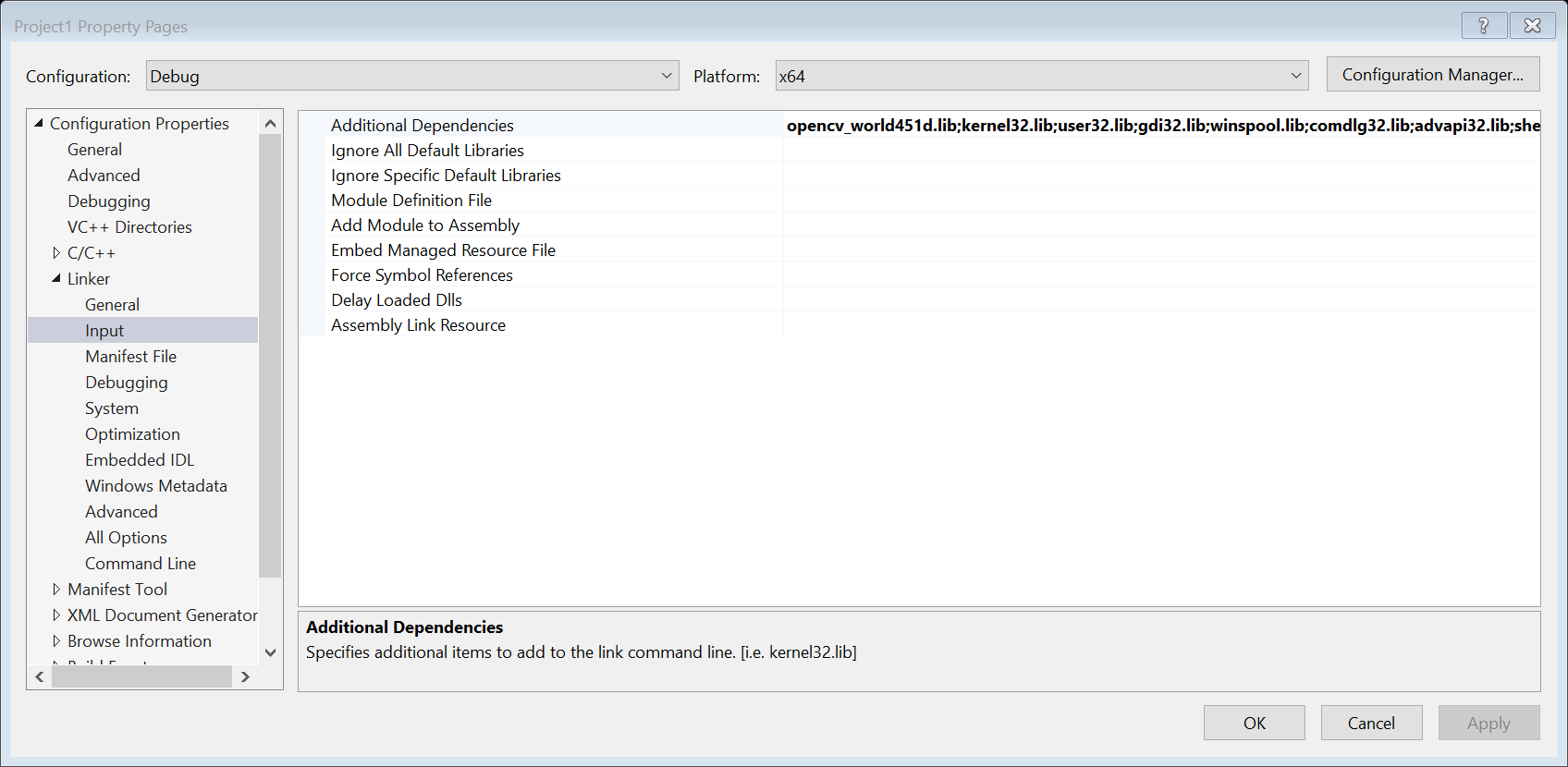
1. Add the additional include folders for OpenCV in your environment in the following page (right click your project 🡪 Properties🡪VC++ Directories). The paths to add are “<your local opencv folder>\build\include\opencv2” and “<your local opencv folder>\build\include”



1. Add the additional library folders for OpenCV in your environment in the following page (right click your project 🡪 Properties🡪VC++ Directories). The path to add is “<your local opencv folder>\build\x64\vc15\lib”



1. Tell the linker the library to link when generating executable files. Add “opencv\_world451d.lib” to “Additional Dependencies”. (If you are working for release mode, add “opencv\_world451.lib”)



1. Add the folder containing DLL files to make your program runnable.

Graphical user interface, text, application

Description automatically generated

Add the following files to your project and try whether it works:

===========================main.cpp========================

int main(int argc, char\* argv[]) {

const char\* imagename = "image.jpg";//image name

Mat img = cv::imread(imagename, 1);

Mat canvas = cv::Mat(img.rows, img.cols, CV\_8UC3); //unsigned char - byte 8 bits C3 -- 3 channels

Mat canvas2 = cv::Mat(10, 10, CV\_64FC1);

cv::resize(img, img, cv::Size(960, 540));

//If failed

if (img.empty()) {

fprintf(stderr, "Can not load image %s\n", imagename);

return -1;

}

//Display

img = cgraProcessing(img);

cv::imshow("image", img);

//To wait for your input

waitKey();

img = cv::Scalar(125, 125, 125) + img; //Blue, Green, Red

cv::Mat roi = img(cv::Rect(0, 0, img.cols / 2, img.rows / 2));

roi.setTo(0);

imshow("image", img);

waitKey();

return 0;

}

====================Header.h========================

#pragma once

#include<opencv2/opencv.hpp>

using namespace cv;

cv::Mat cgraProcessing(cv::Mat& r);

====================Header.cpp=======================

#include "Header.h"

cv::Mat cgraProcessing(cv::Mat& r) {

for (int i = 0; i < r.rows; ++i) { // rows

for (int j = 0; j < r.cols; ++j) { // cols

// r.at<Vec3b>(i, j) = Vec3b(255) - r.at<Vec3b>(i, j); //cv::Scalar(255, 255, 255)

// r.at<Vec3b>(i, j)[0] = 255 - r.at<Vec3b>(i, j)[0];

// r.at<Vec3b>(i, j)[1] = 255 - r.at<Vec3b>(i, j)[1];

// r.at<Vec3b>(i, j)[2] = 255 - r.at<Vec3b>(i, j)[2];

r.at<uchar>(i, 3 \* j + 0) = 255 - r.at<uchar>(i, 3 \* j);

r.at<uchar>(i, 3 \* j + 1) = 255 - r.at<uchar>(i, 3 \* j + 1);

r.at<uchar>(i, 3 \* j + 2) = 255 - r.at<uchar>(i, 3 \* j + 2); //CV\_8UC1, CV\_8UC3, CV\_32FC3,

}

}

return r;

}