SPAWAR
Systems Center
PACIFIC

Introduction to Open Computer Vision C++ Library

Mr. John Stastny
SPAWAR Systems Center, Pacific
Phone: 619-553-4564
Email: john.stastny@navy.mil

Mr. Bryan Bagnall
SPAWAR Systems Center, Pacific
Phone: 619-553-4061
Email: bryan.bagnall@navy.mil

Mr. Lucas Keenan
SPAWAR Systems Center, Pacific
Phone: 619-553-3686
Email: lucas.keenan@navy.mil

SPAWAR Systems Center PACIFIC

Overview of Talk

- Introduction to OpenCV library
- Introduction to Microsoft Visual C++ and How to Create an OpenCV Executable
- OpenCV Examples: Opening and Displaying and Image, Thresholding, Edge Detection, and writing Output Image as jpeg.
- OpenCV Example: 2-D Wiener Filtering with input parameters
- Conclusions

SPAWAR Systems Center PACIFIC

What is OpenCV?

- Open Computer Vision library
- Collection of math, signal, and image processing functions
- Natively written in C/C++, but now works in Python
- Bindings for python, java, and other languages
- Written to be optimized for SSE instructions (fast)
- Now written in CUDA for GPU processing
- Uses Linpack linear algebra library, which is considered the fastest/best (Matlab uses this library)
- Capable of performing wide range of image/signal processing tasks

OpenCV Overview (sample of functions)

- Thresholding
- Edge Detection
- Hough Transforms/Line Detection/Circle Detection
- Fourier Transforms
- Histograms
- 2-D Image Filtering
- Shape matching
- Shape features (SIFT, SURF, etc)
- Linear algebra
- SVD, L2 minimization, QR Decomp, etc
- Image Matching (SIFT visual BOW's key point matching)
- Machine Learning (SVM, NN, Neural Networks, etc)
- Image arithmetic (add, subtract, multiply/devide images/constants)
- Line/curve fitting
- Random variables
- Contour processing
- Image writing (tiff, jpeg, etc)
- Support for multichannel images, regions of interest, and masks for most functions

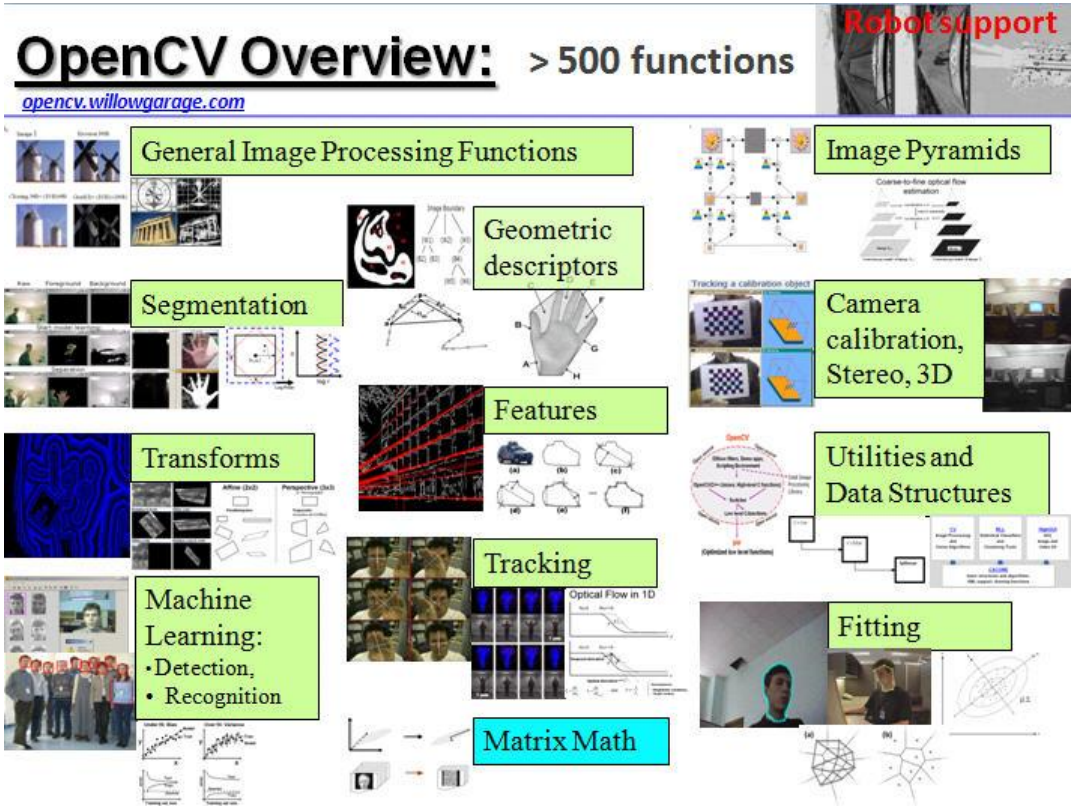
OpenCV Resources

Wiki (documentation) - http://opencv.willowgarage.com/documentation/cpp/index.html

Documentation for C++ API - http://opencv.willowgarage.com/documentation/cpp/index.html

Documentation for C API - http://opencv.willowgarage.com/documentation/c/index.html

Tutorials: http://www.cs.iit.edu/~agam/cs512/lect-notes/opencv-intro/opencv-intro.html

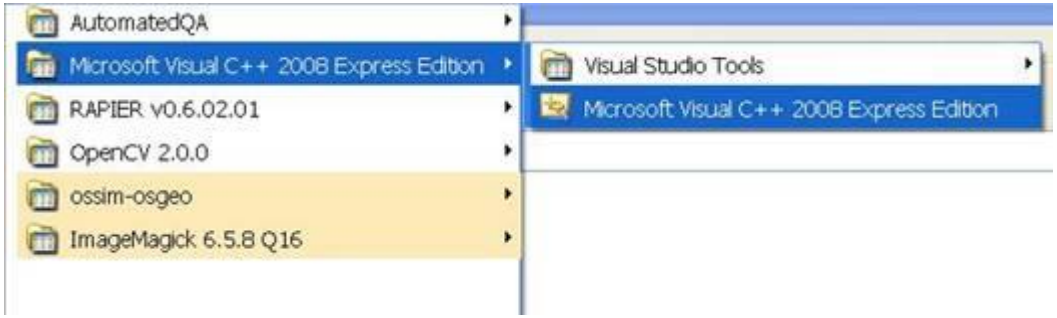


SPAWAR Systems Center PACIFIC

Introduction to Microsoft Visual C++ and How to Create an OpenCV Executable

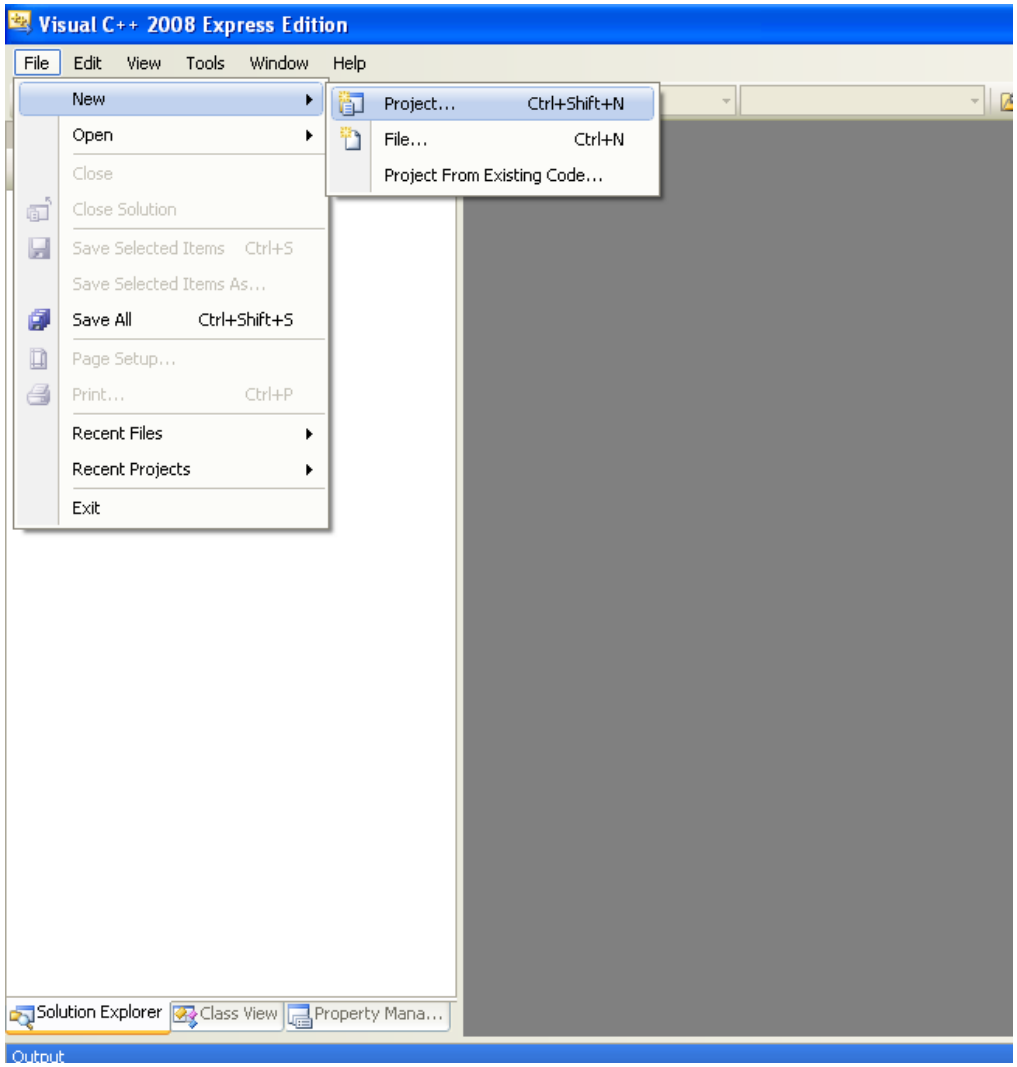
Creating A New MS VC++ Project

- Open Microsoft Visual C++ in the Start Menu



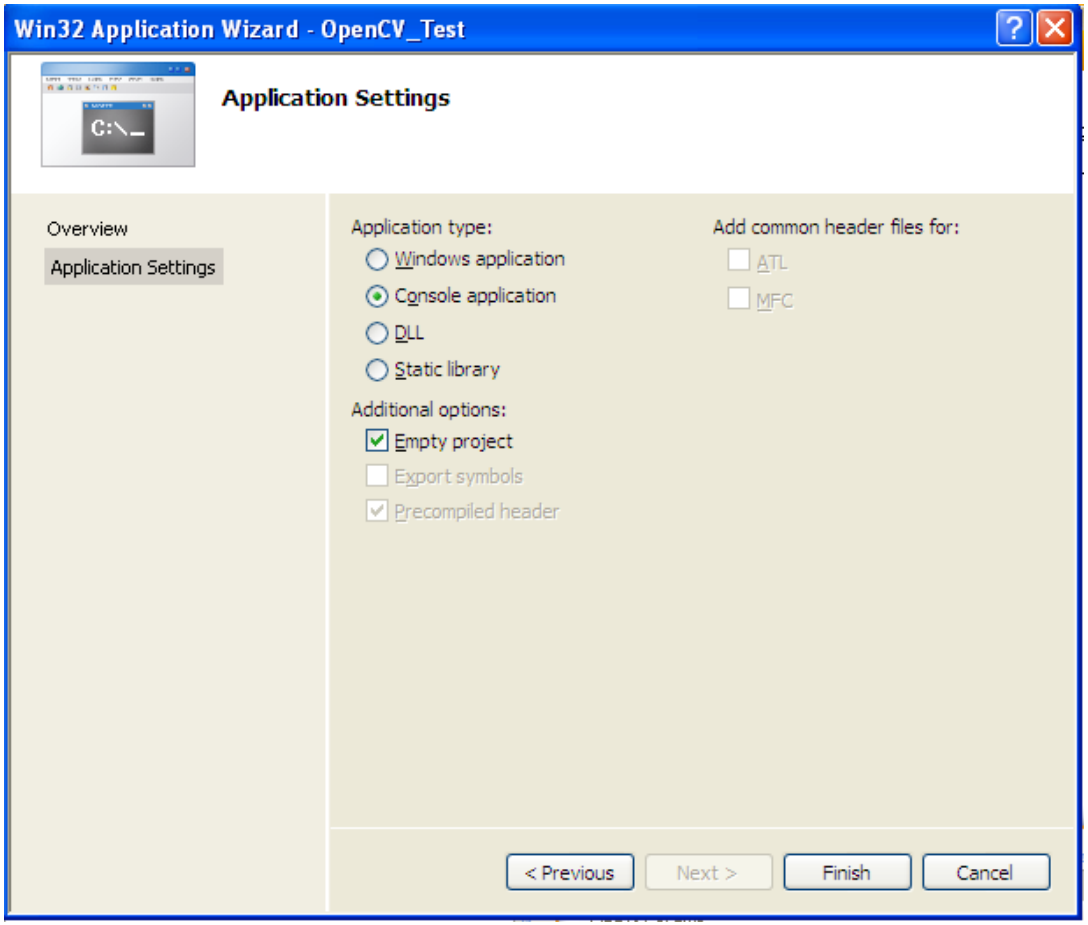
Writing an Example OpenCV Executable

- Select File->New->Project…



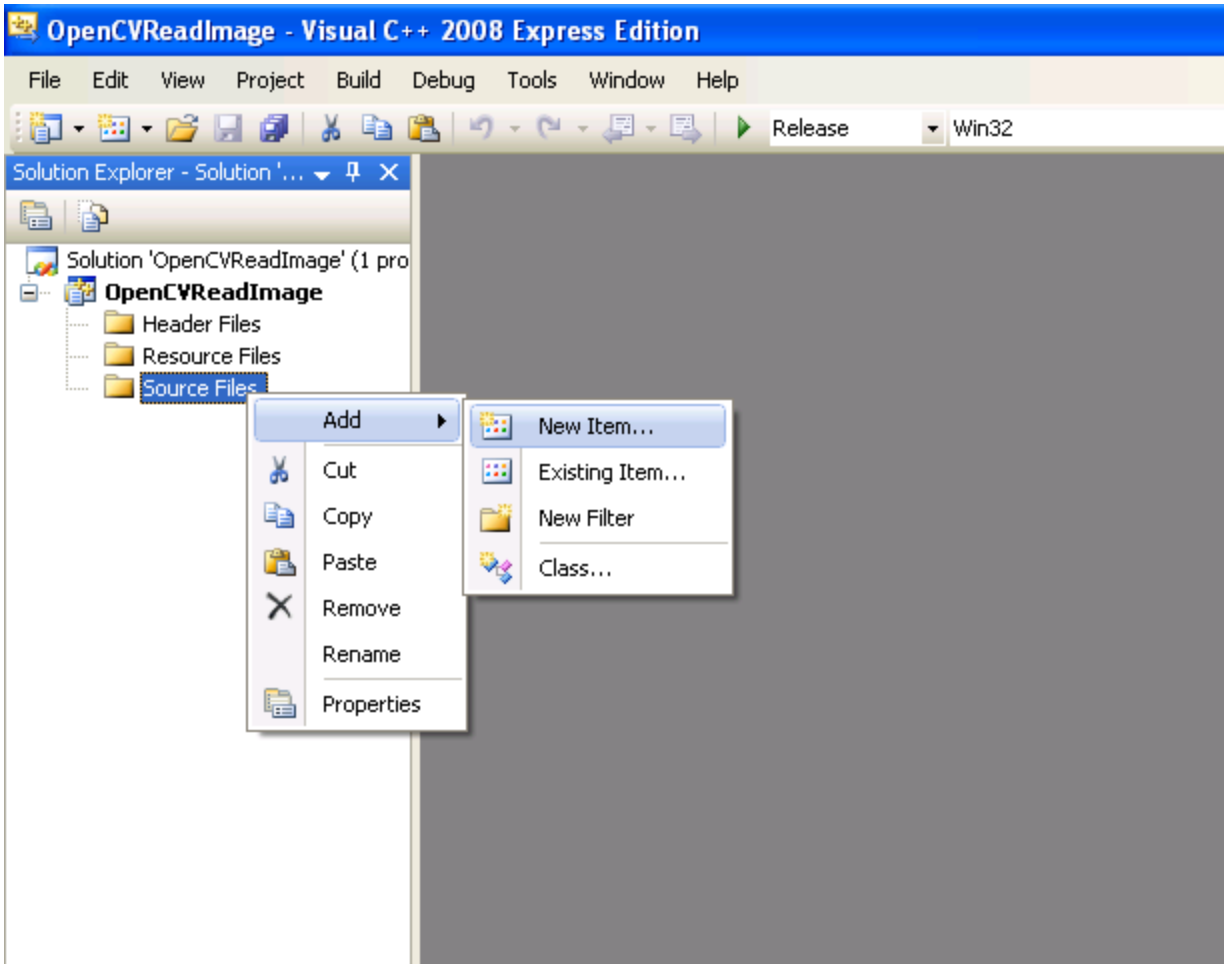
Writing an Example OpenCV Executable

- Select Application Settings on the left
- Click on the Console application button
- Click the Empty project button.
- Select Finish



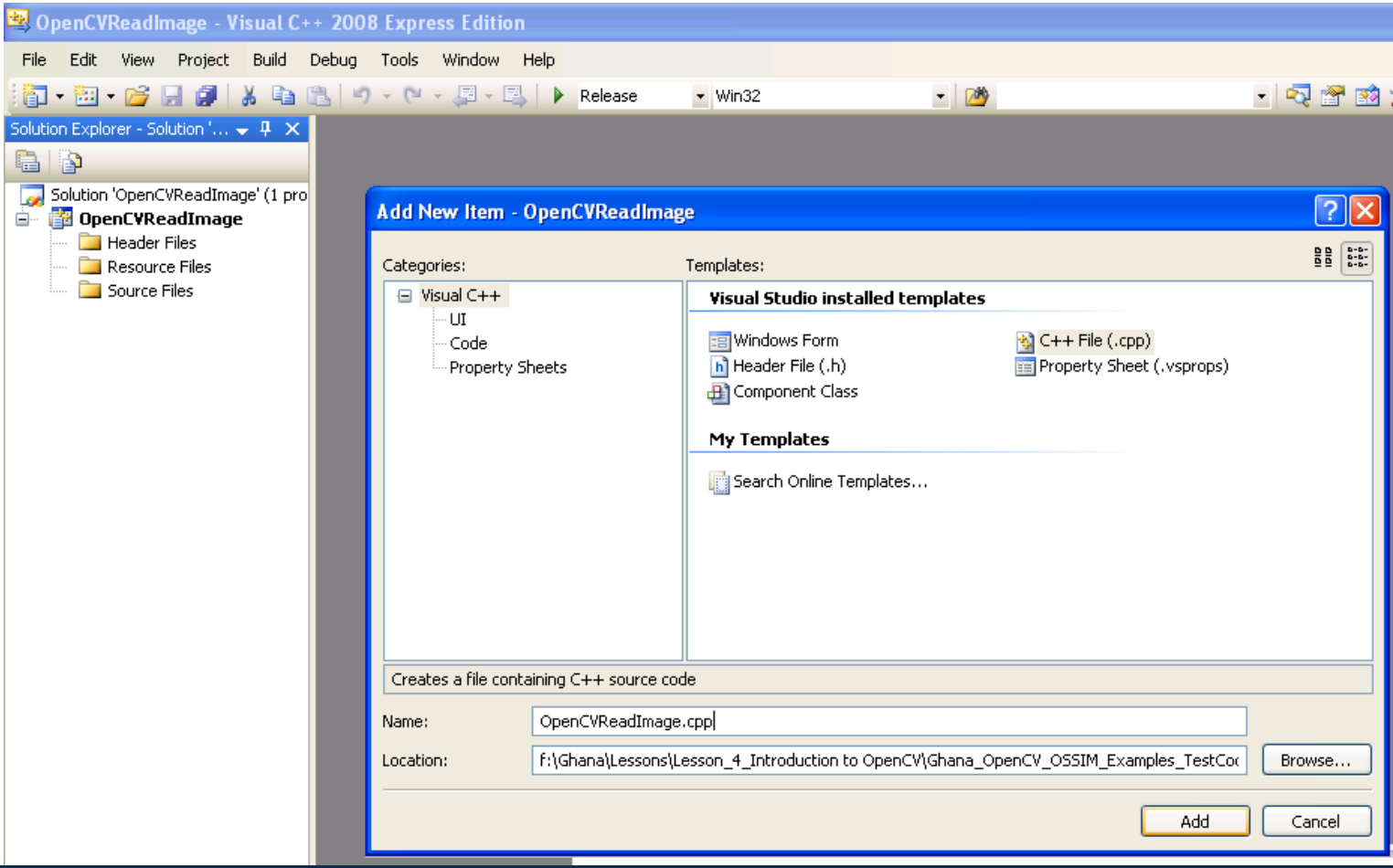
Writing an Example OpenCV Executable

- You should now have an empty project.
- Let's add a file.



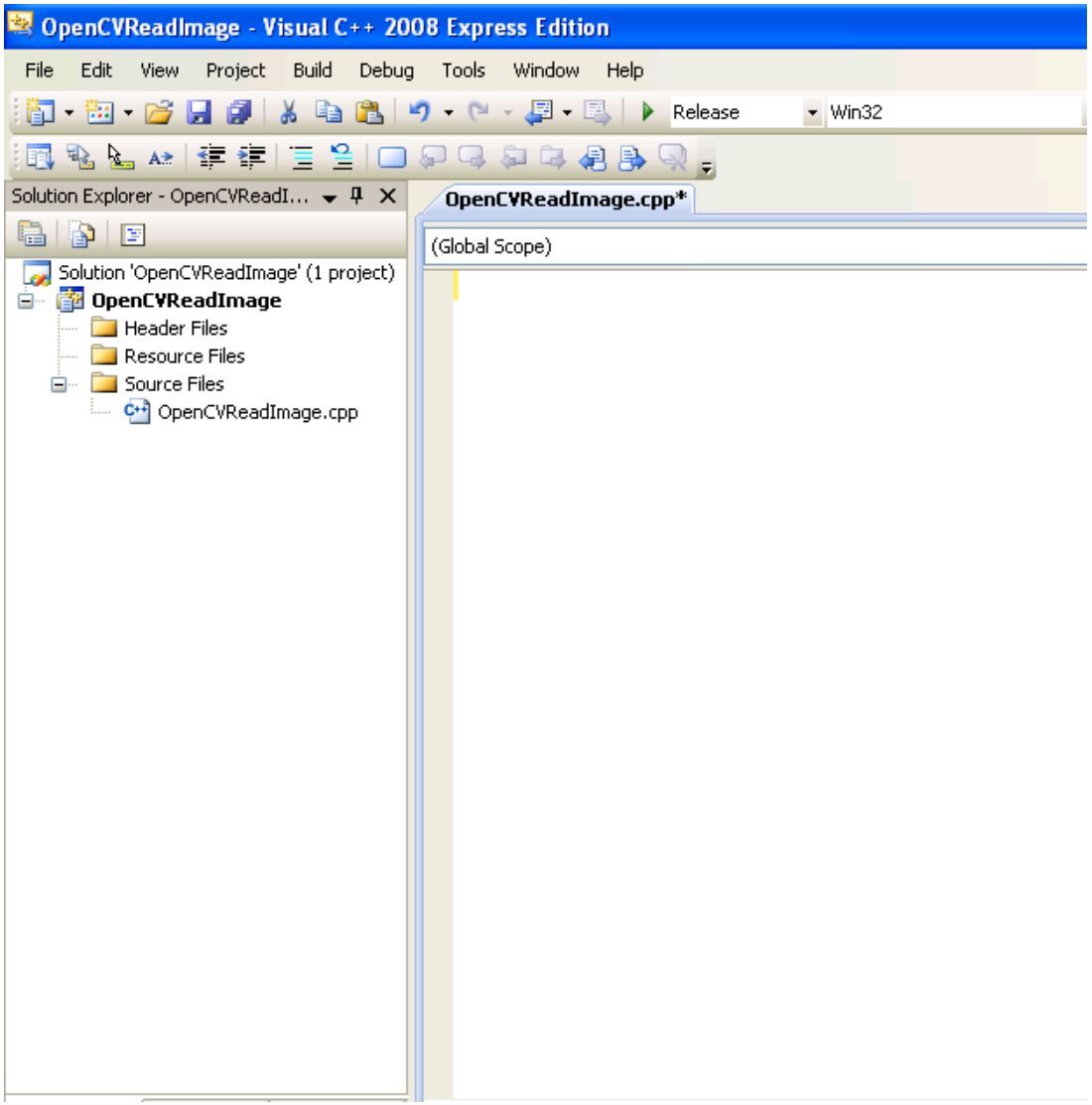
Writing an Example OpenCV Executable

- Choose C++ File (.cpp)
- The Name should be something like OpenCVReadImage.cpp



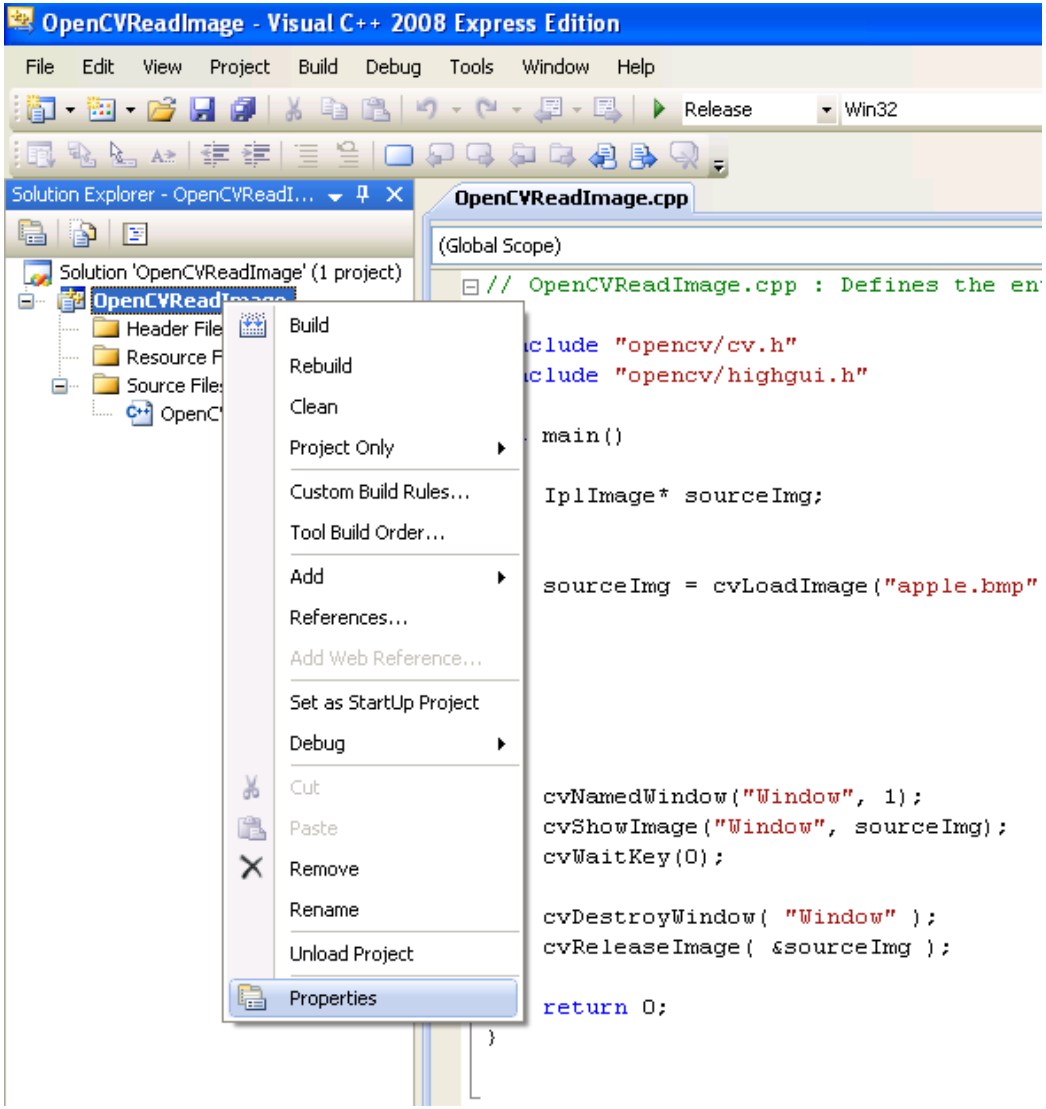
Writing an Example OpenCV Executable

- You should see an empty C++ file.
- We'll add code in a few slides.



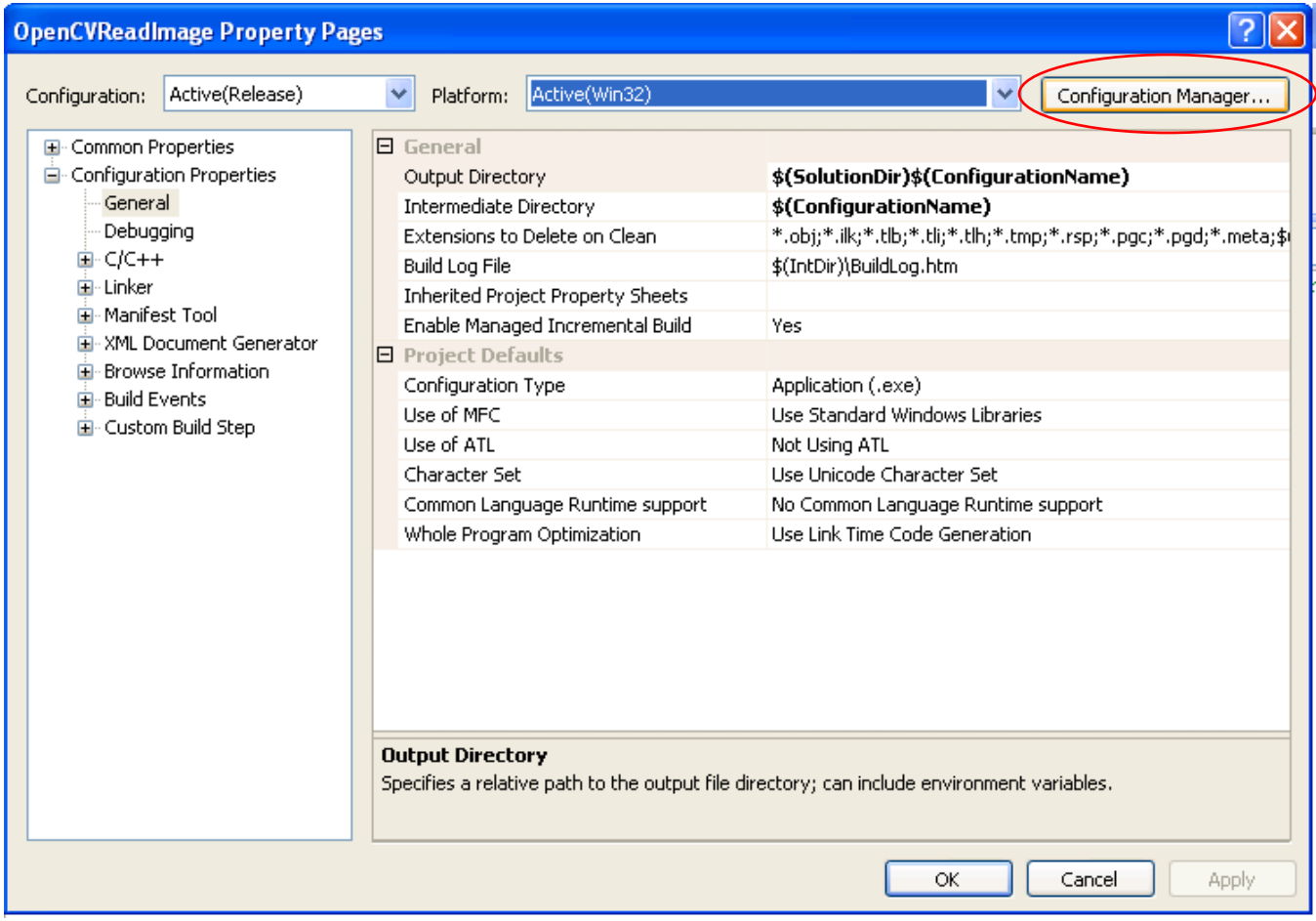
Writing an Example OpenCV Executable

- In C++, we must explicitly specify any libraries our functions will need to use. In this case, we will specify all of the OpenCV libraries.
- Right click on the project name, then click on Properties



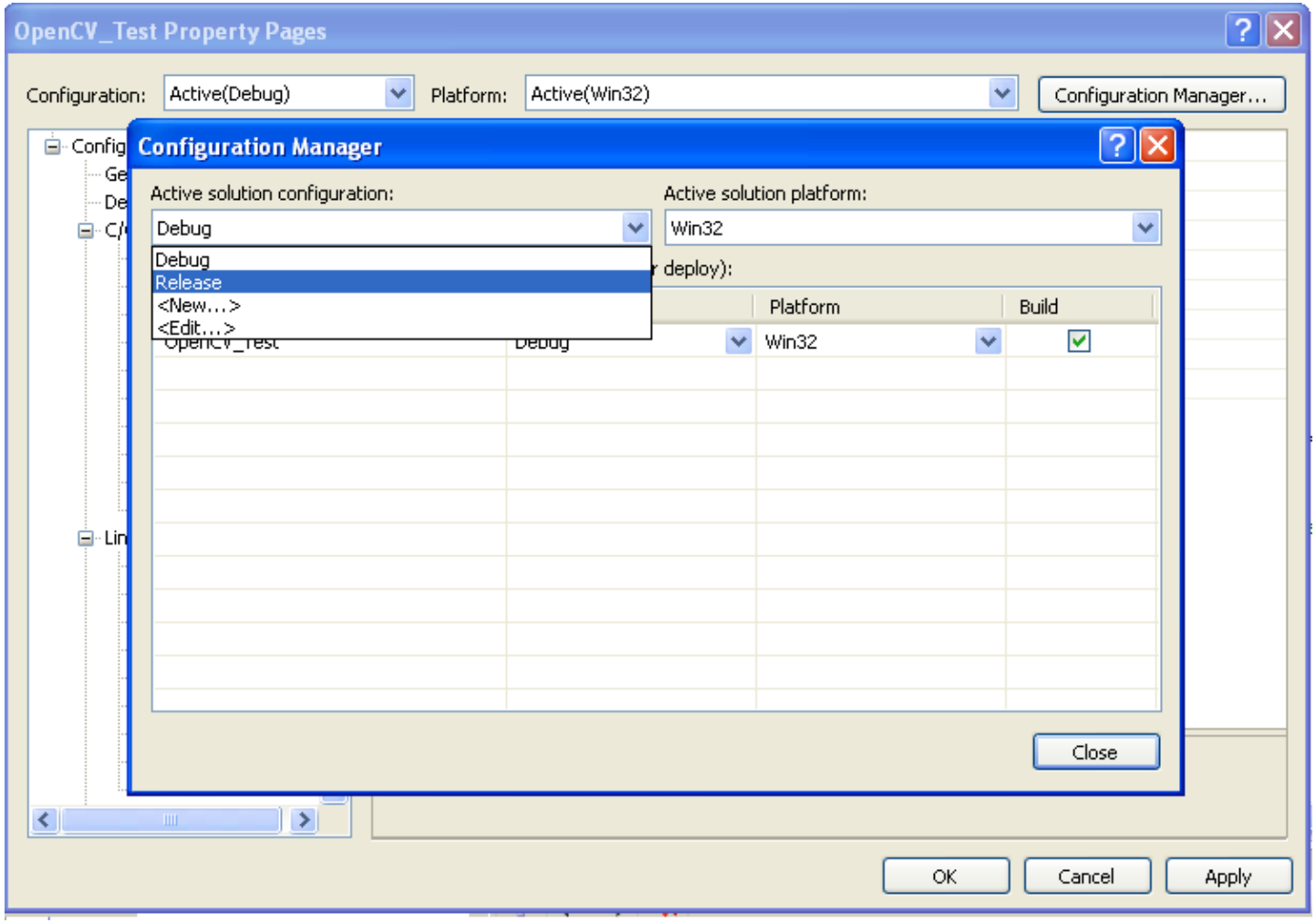
Writing an Example OpenCV Executable

- We will be producing with a Release (not Debug) version of our .exe
- Click on Configuration Manager… button



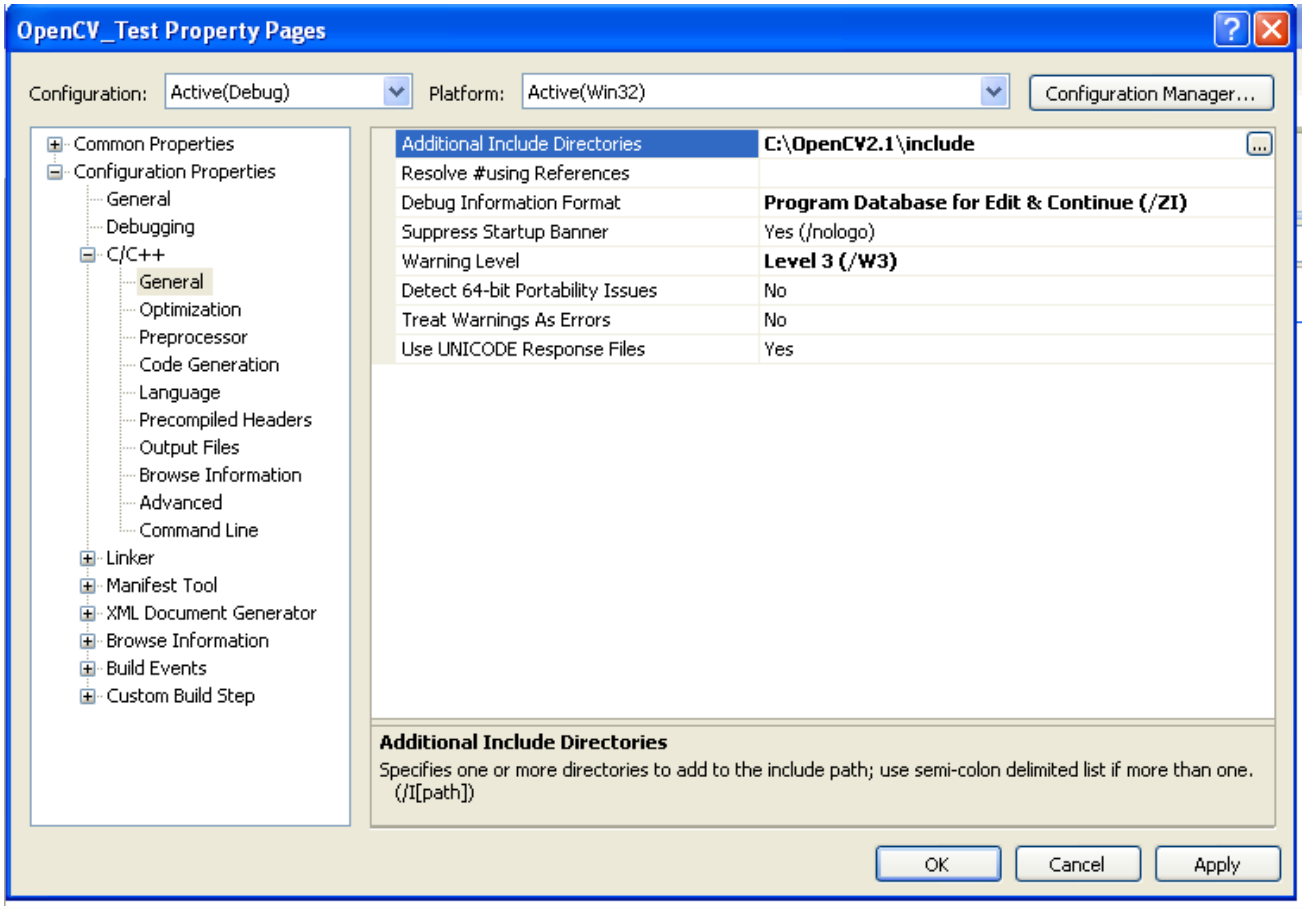
Writing an Example OpenCV Executable

- Change the Active solution configuration to Release and click close



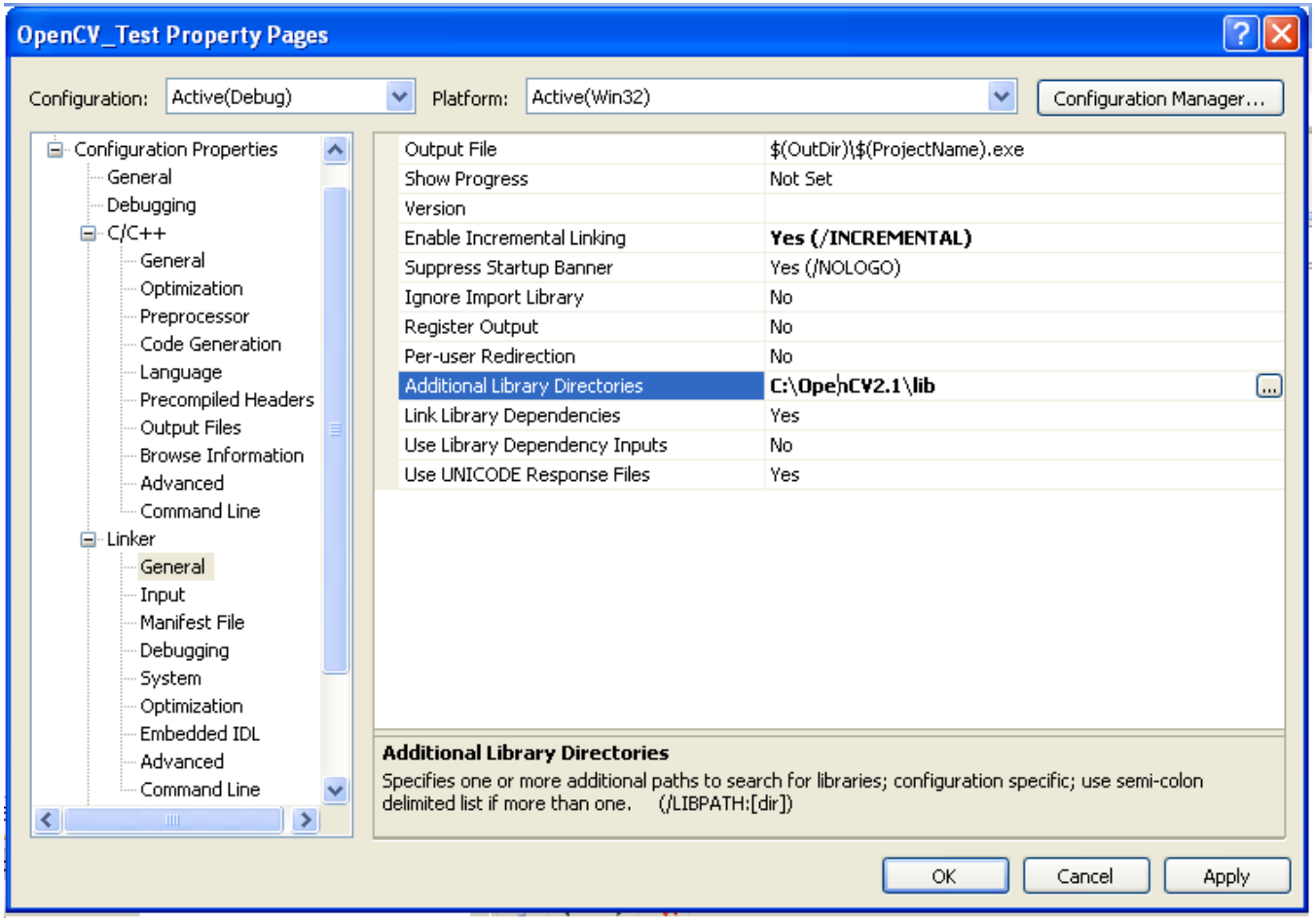
Writing an Example OpenCV Executable

- In the Properties Window, select General under C++, then click on Additional Includes Directories:
- Add C:\OpenCV2.1\include or wherever your OpenCV installation is location.



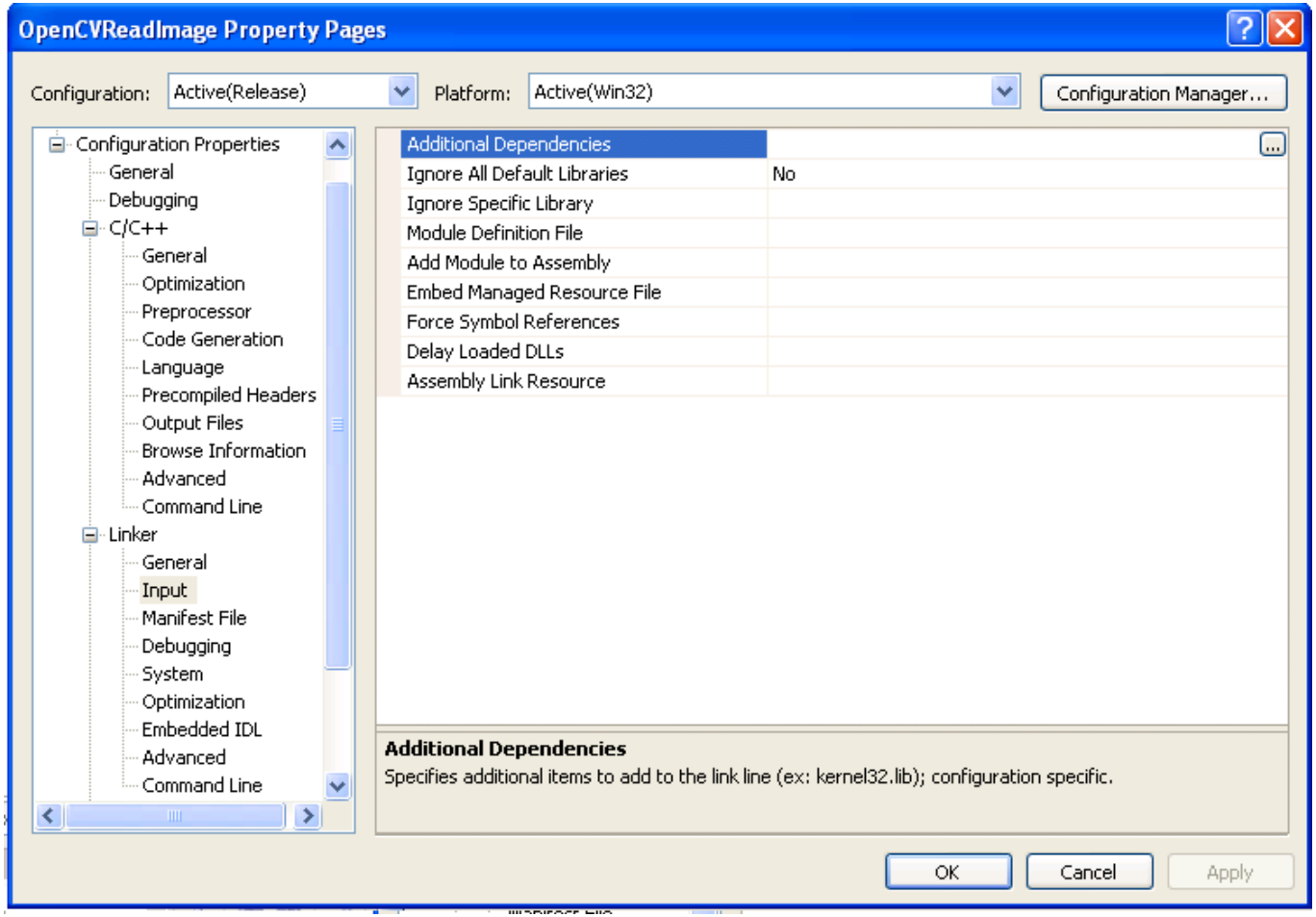
Writing an Example OpenCV Executable

- Expand the Linker Options and click on General. Add C:\OpenCV2.1\lib to the Additional Library Directories.



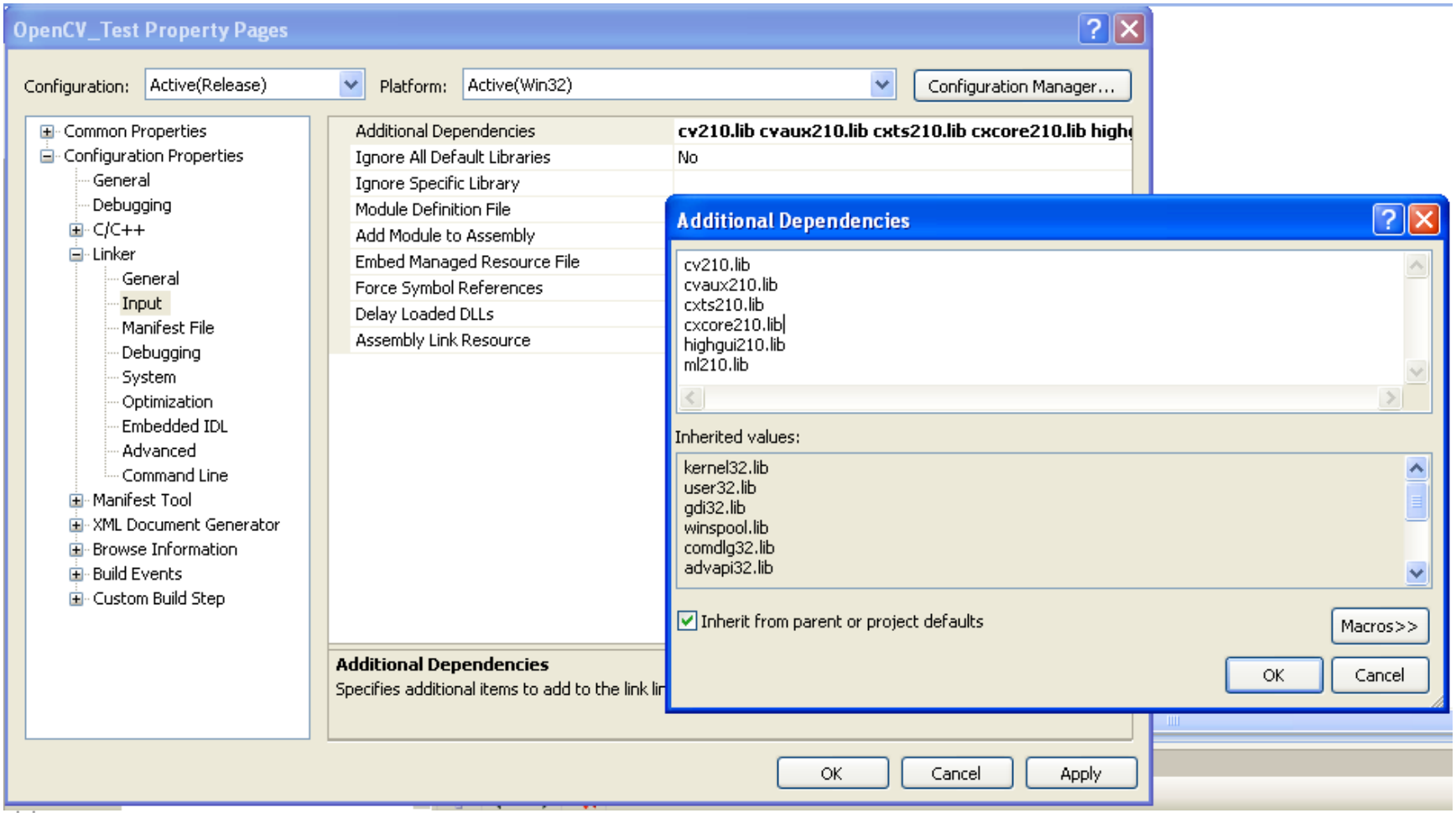
Writing an Example OpenCV Executable

- Under Linker, click on Input. Click on the white space to the right of Additional Dependencies. You'll see a … button. Click on this.



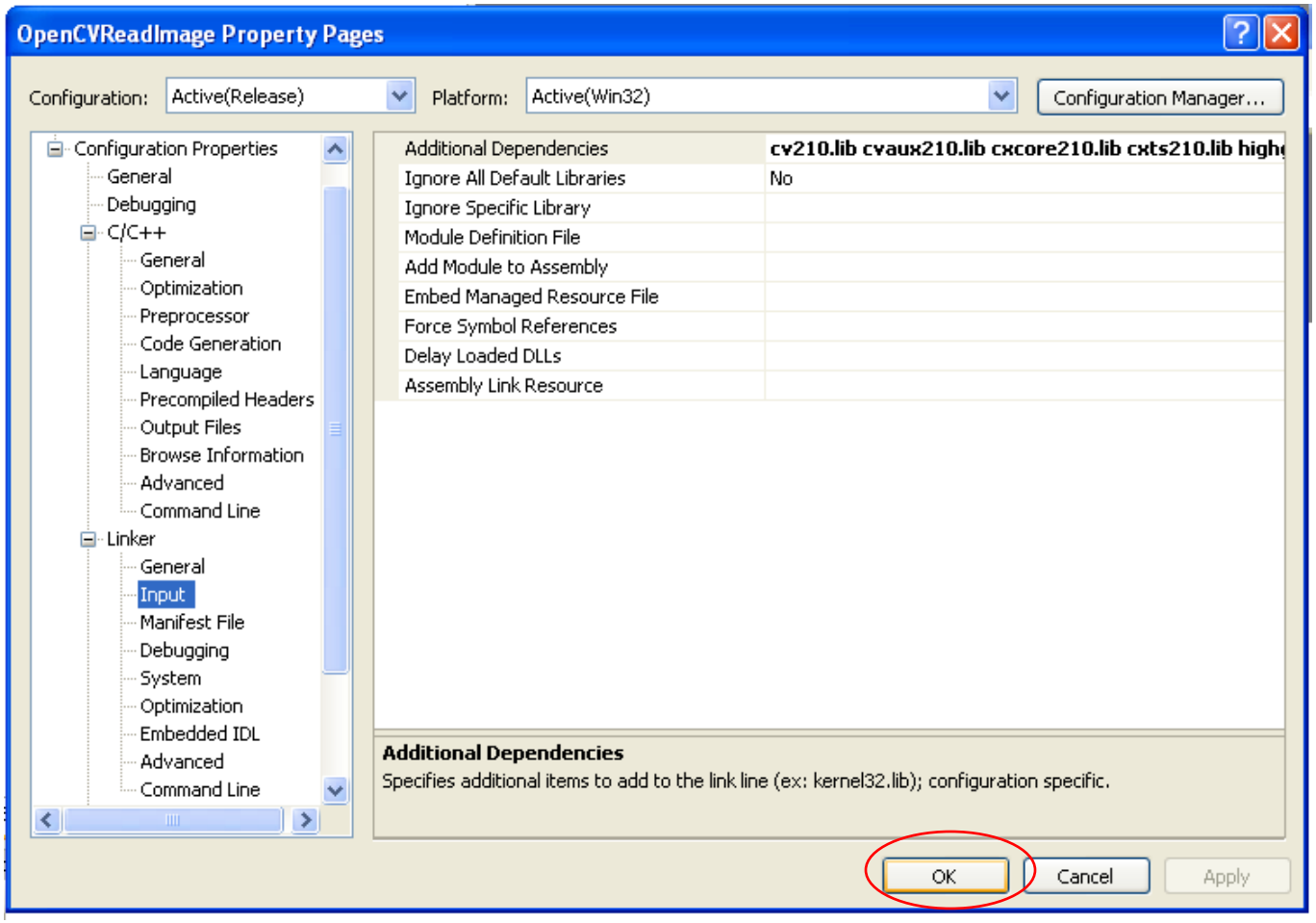
Writing an Example OpenCV Executable

- Add the OpenCV .lib file names below to the Additional Dependancies



Writing an Example OpenCV Executable

- No click OK, and we are ready to build our first OpenCV executable!



OpenCV Examples – Opening an Image, Edge Detection, Thresholding, and Writing the Result to a jpeg

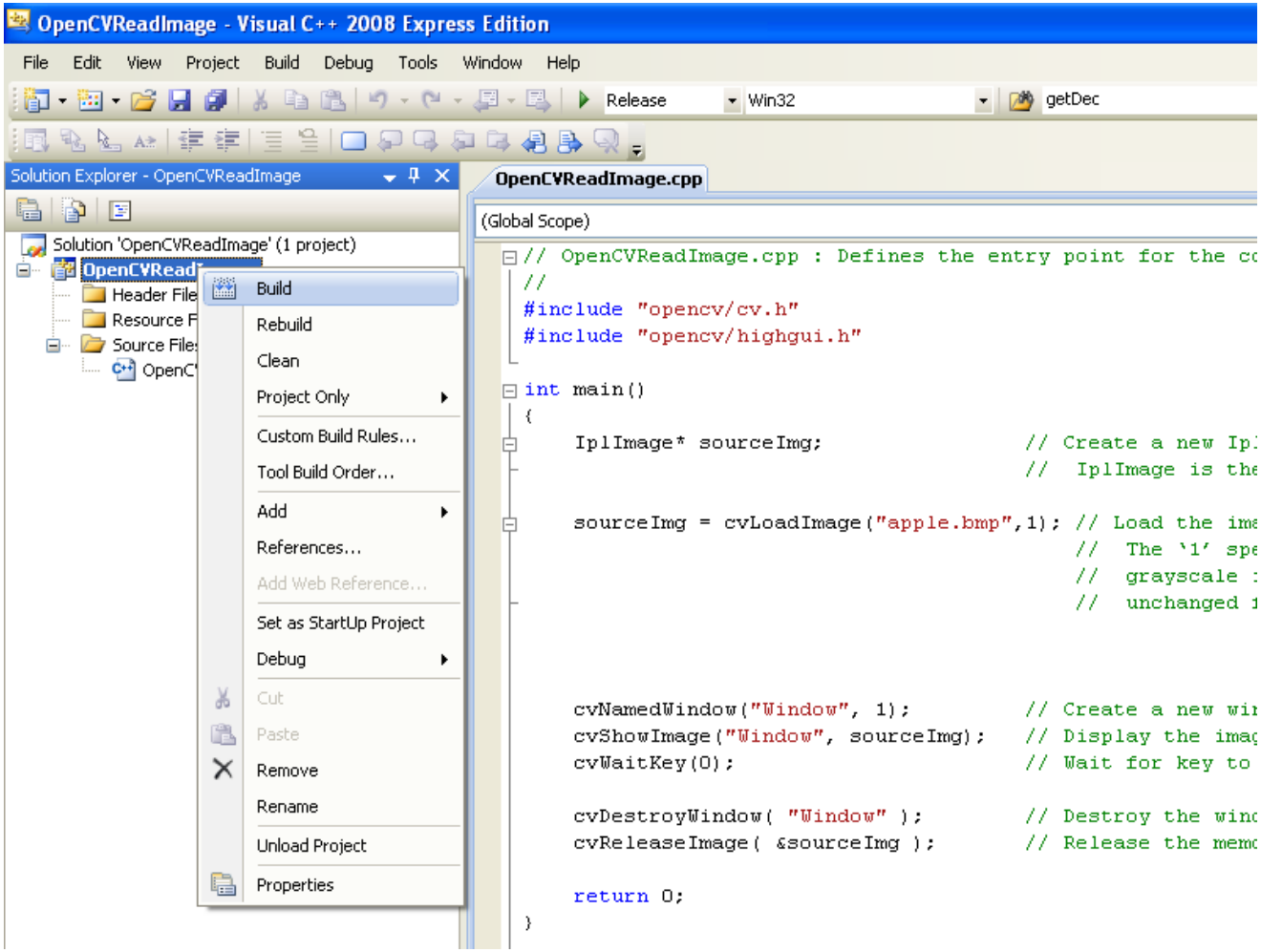
How to Create Image and Read it From a File

▼ Creating and Reading an Image

```
#include "opencv/cv.h"
#include "opencv/highgui.h"

int main()
{
    IplImage* sourceImg;                    // Create a new IplImage image data structure
                                            //  IplImage is the basic image data structure in OpenCV

    sourceImg = cvLoadImage("apple.bmp",1); // Load the image file into the image data structure
                                            //  The '1' specifies the image is color ('0' to force
                                            //  grayscale image, and '-1' to leave color information
                                            //  unchanged from file)

    cvNamedWindow( "Window", 1);            // Create a new window
    cvShowImage("Window", sourceImg);       // Display the image in the window

    cvWaitKey(0);                           // Wait for key to close the window

    cvDestroyWindow( "Window" );            // Destroy the window
    cvReleaseImage( &sourceImg );           // Release the memory for the image

    return 0;
}
```

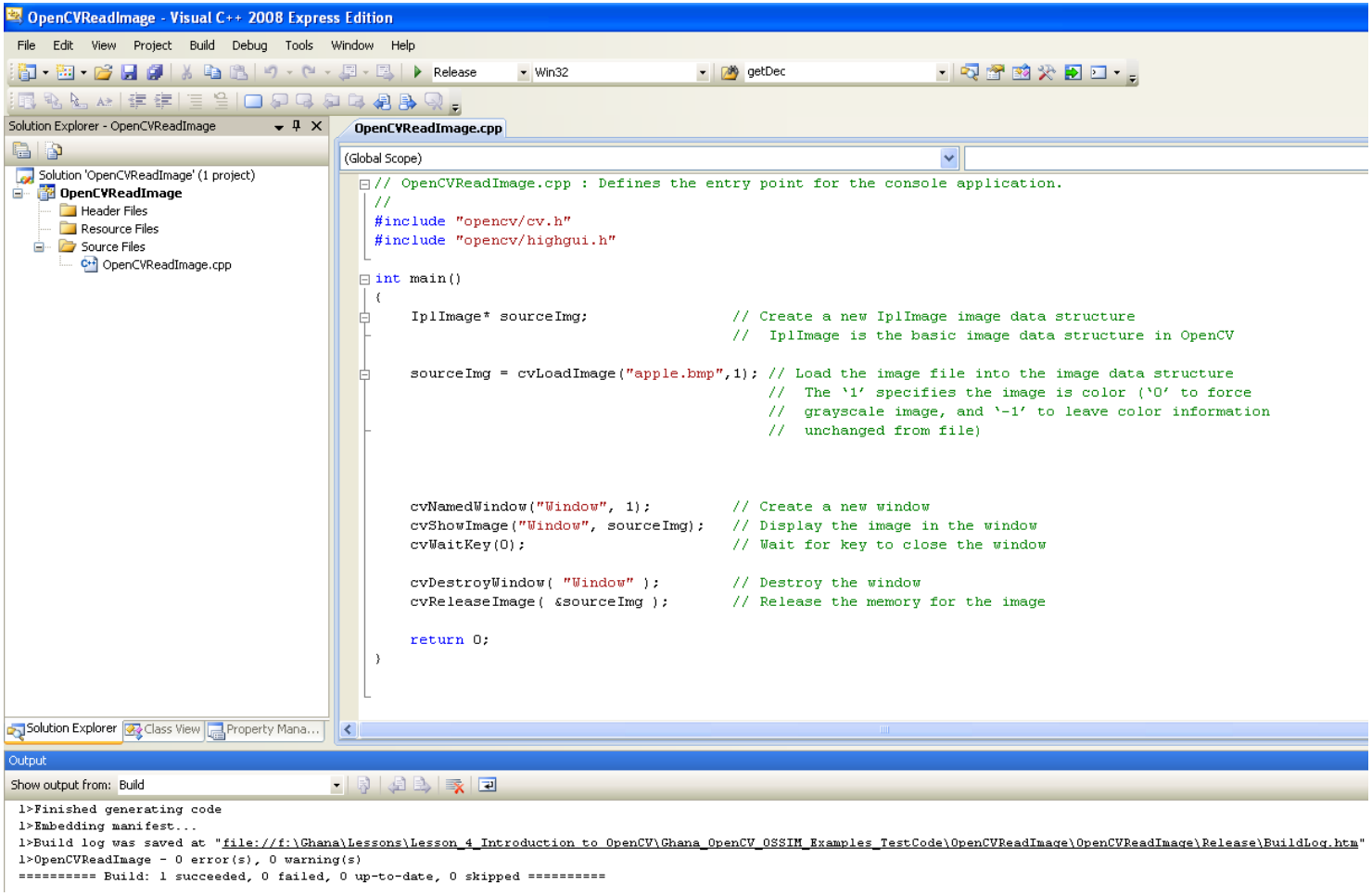
Writing an Example OpenCV Executable

- Once you have completed your code, right click on the project, then click build.



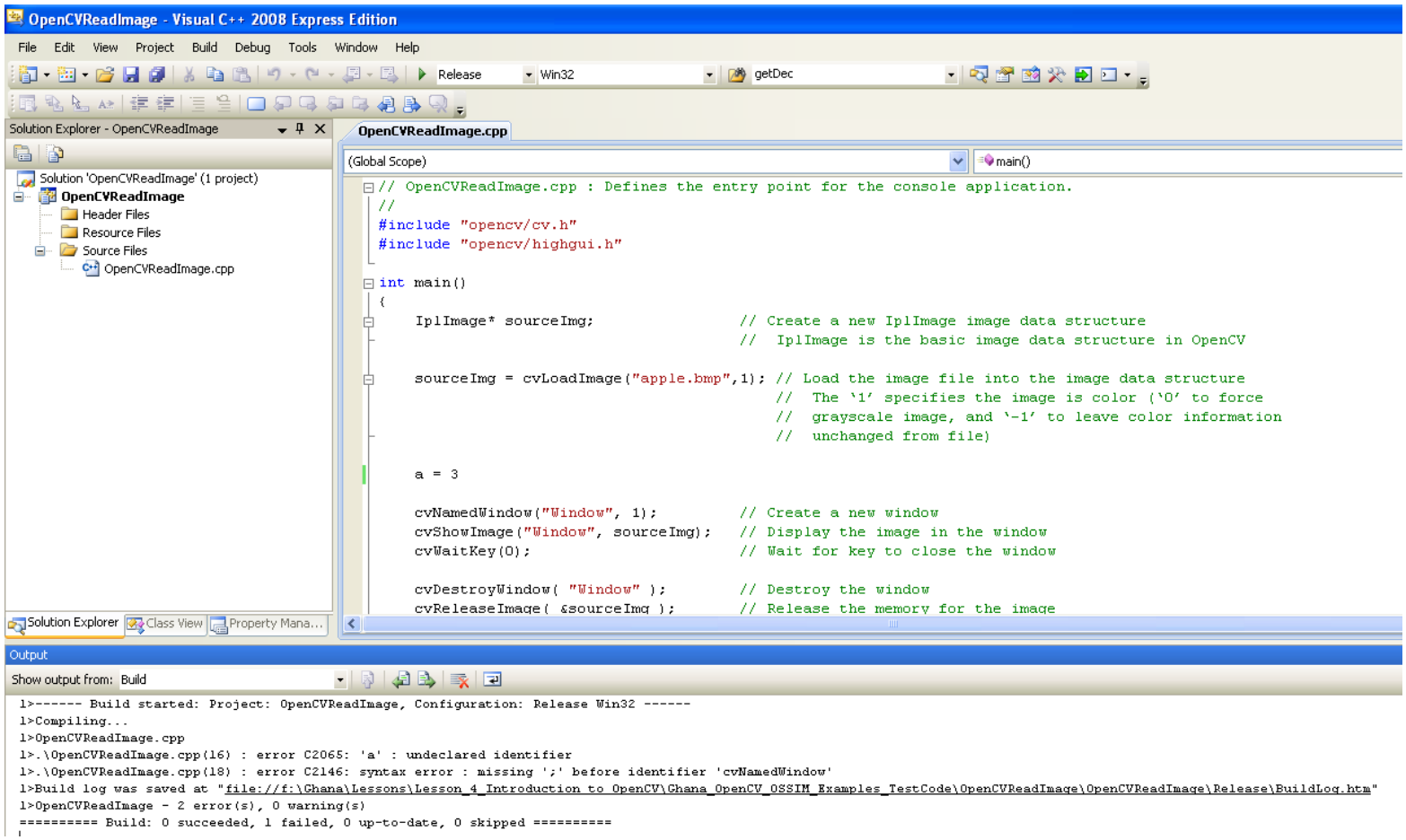
Writing an Example OpenCV Executable

- When the code is done being built, you'll see a message Build: 1 succeeded, 0 failed if there were no compiling errors.



Writing an Example OpenCV Executable

- If there are compiling errors, the Build will fail, and the errors will be listed with an explanation. If you don't understand an error, look it up online. Often, others have seen and solved the same error.



Writing an Example OpenCV Executable

- If you click the errors tab at the bottom, then click and error, it will take you to the line where the error is occuring and often you'll notice the error. Remember, it's not Matlab!!

Need a type "int" and ";"

```
    a = 3

    cvNamedWindow("Window", 1);          // Create a new window
    cvShowImage("Window", sourceImg);    // Display the image in the window
    cvWaitKey(0);                        // Wait for key to close the window

    cvDestroyWindow( "Window" );         // Destroy the window
    cvReleaseImage( &sourceImg );        // Release the memory for the image

    return 0;
}
```

Solution Explorer | Class View | Property Mana...

Error List

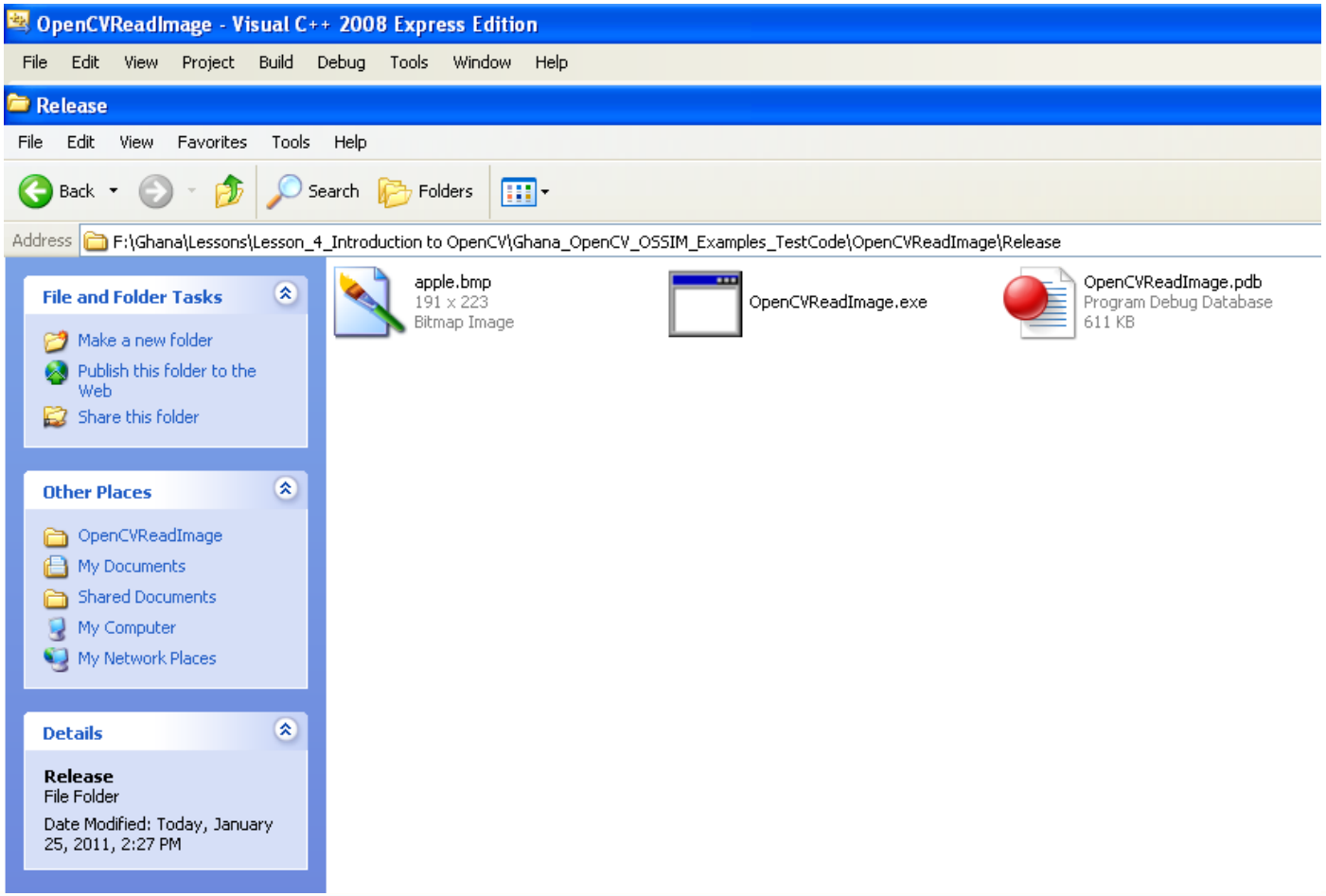
2 Errors | 0 Warnings | 0 Messages

	Description	File	Line
1	error C2065: 'a' : undeclared identifier	OpenCVReadImage.cpp	16
2	error C2146: syntax error : missing ';' before identifier 'cvNamedWindow'	OpenCVReadImage.cpp	18

Code Definition Window | Call Browser | Output | Error List

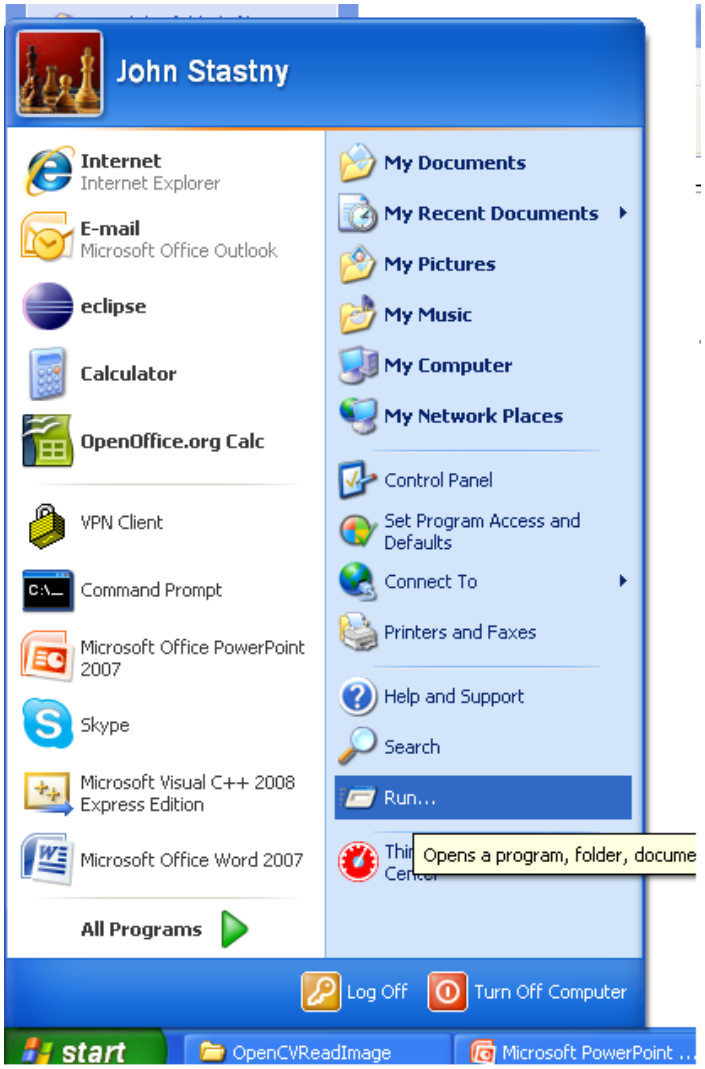
Writing an Example OpenCV Executable

- Once your code has been compiled successfully, you'll want to run it. First, let's make sure our output folder looks correct. You should grab the image apple.bmp from the Lesson 4 OpenCVReadImage Folder.



Writing an Example OpenCV Executable

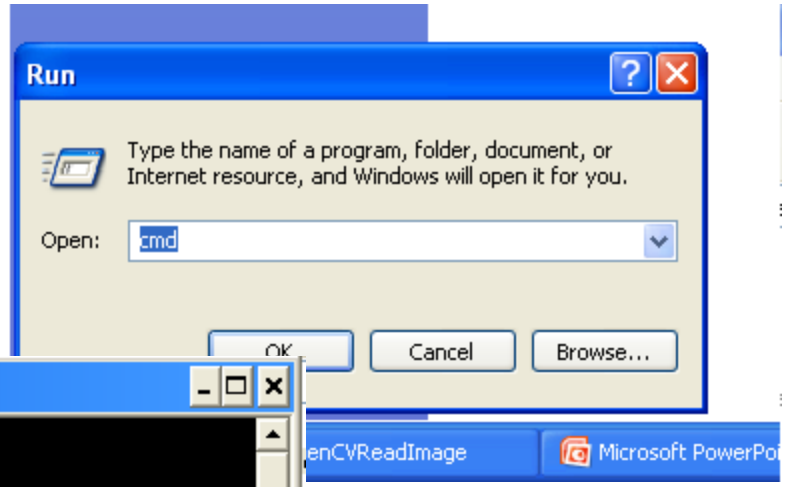
- To run your code, you can double click the executable, or run it from the command line. Let's go through the command line.
- Go to Program Files-Run

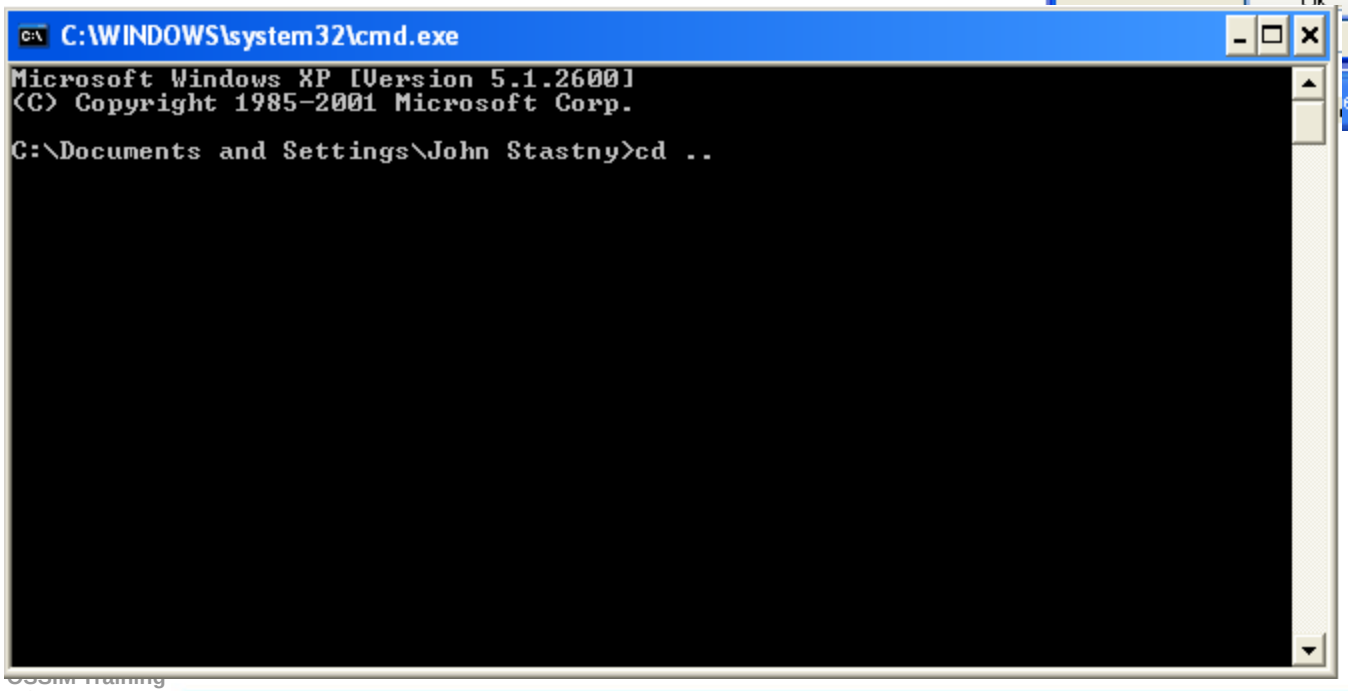


Writing an Example OpenCV Executable

- In the Run field, type cmd

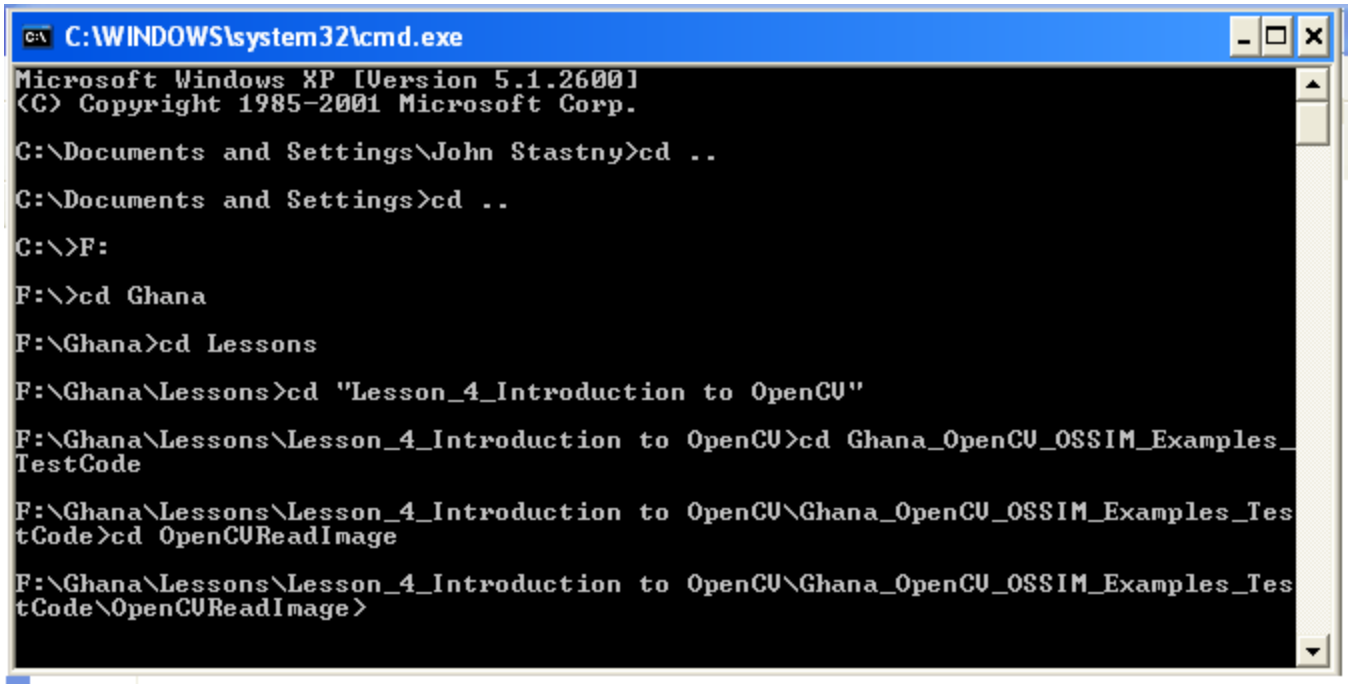
  This will open the command prompt





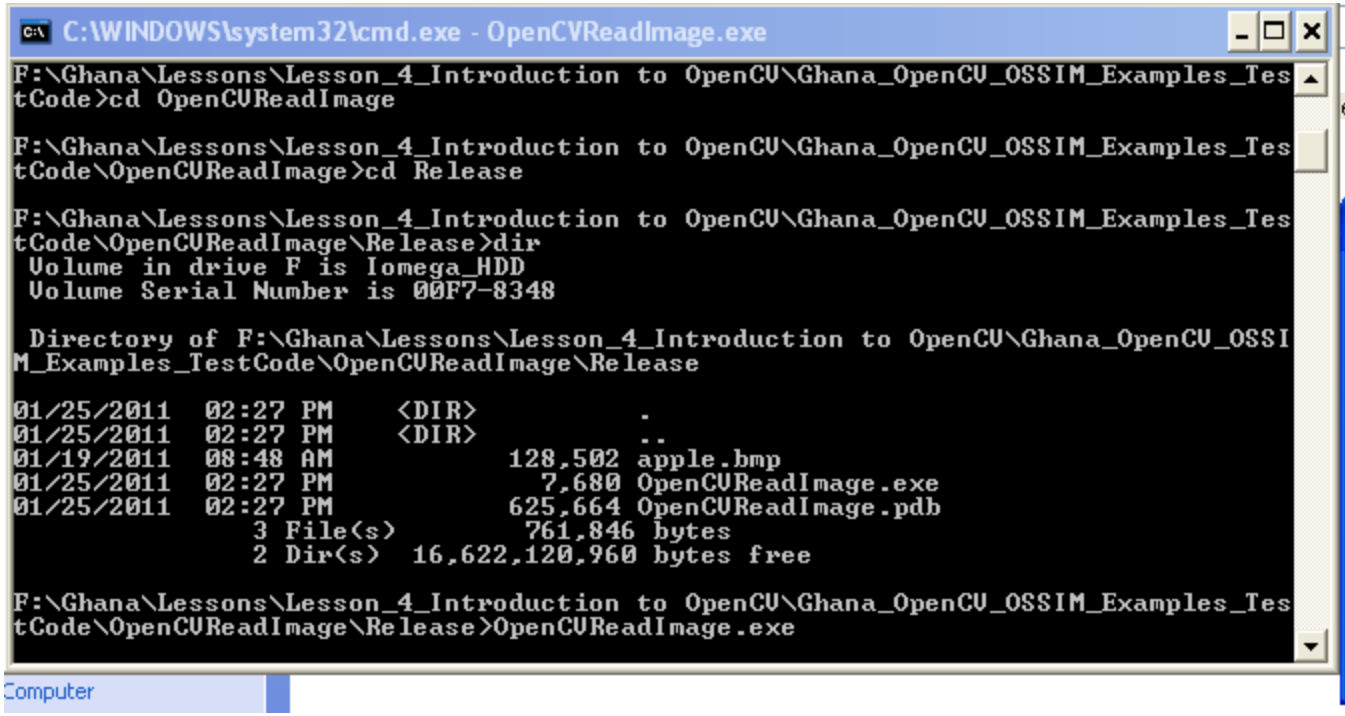
Writing an Example OpenCV Executable

- Using the cd commands, navigate to the folder where the executable is located. You can press tab to autocomplete directory names.



Writing an Example OpenCV Executable

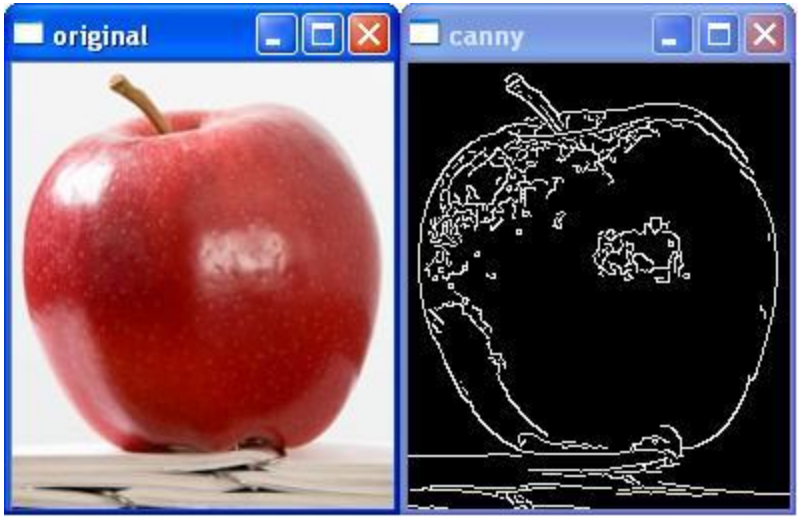
- Since all of the OpenCV .dll files are in our path already, we can run our executables.
- To do so, simply type the name of the executable.



Simple Operations- Edge Detection, Thresholding

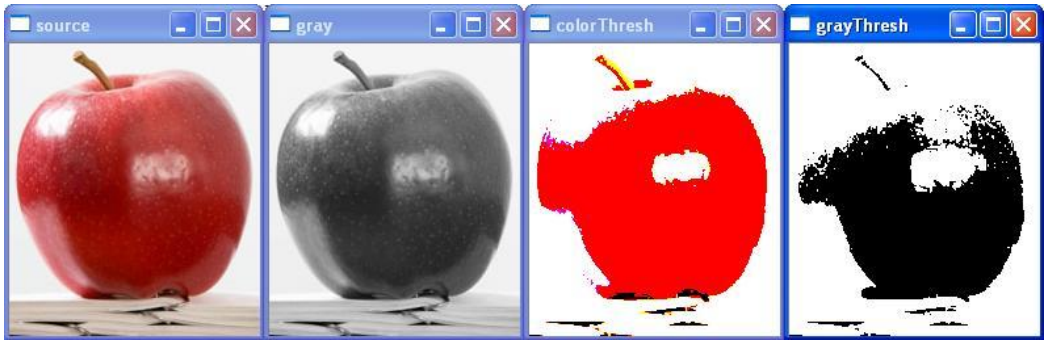
- Canny Edge Detection (..\03_OpenCV_and_blobs\examples\OpenCV_examples
  - \OpenCVEdgeDetect\OpenCVEdgeDetect.cpp)

```
...

// Load a color (3 channel RGB) image
sourceImg = cvLoadImage("apple.bmp",1);

// Create a single channel 1 byte image (grayscale image)
grayImg = cvCreateImage( cvSize(sourceImg->width, sourceImg->height), IPL_DEPTH_8U, 1 );

// Convert the original color image to grayscale image
cvCvtColor( sourceImg, grayImg, CV_BGR2GRAY );

// Create a grayscale image to store the Canny edge detection result
cannyImg = cvCreateImage( cvGetSize(sourceImg), IPL_DEPTH_8U, 1 );

// Canny Edge Detection
cvCanny(grayImg, cannyImg, 50, 150, 3);

cvNamedWindow( "original", 1 );
cvNamedWindow( "canny", 1 );
cvShowImage( "original", sourceImg );
cvShowImage( "canny", cannyImg );
cvWaitKey(0);

...
```



Simple Operations- Edge Detection, Thresholding

Thresholding (..\03_OpenCV_and_blobs\examples\OpenCV_examples\OpenCVThreshold\OpenCVEdgeThreshold.cpp)




```
...

IplImage* sourceImg;
IplImage* colorThresh;
IplImage* gray;
IplImage* grayThresh;

int threshold = 100, maxValue = 255;
int thresholdType = CV_THRESH_BINARY;

sourceImg = cvLoadImage("apple.bmp", 1);
colorThresh = cvCloneImage( sourceImg );
gray = cvCreateImage( cvSize(sourceImg->width, sourceImg->height), IPL_DEPTH_8U, 1 );
cvCvtColor( sourceImg, gray, CV_BGR2GRAY );
grayThresh = cvCloneImage( gray );

cvNamedWindow( "source", 1 );      cvShowImage( "source", sourceImg);
cvNamedWindow( "gray", 1 );        cvShowImage( "gray", gray );

cvThreshold( sourceImg, colorThresh, threshold, maxValue, thresholdType );
cvThreshold( gray, grayThresh, threshold, maxValue, thresholdType );

cvNamedWindow( "colorThresh", 1 );     cvShowImage( "colorThresh", colorThresh );
cvNamedWindow( "grayThresh", 1 );     cvShowImage( "grayThresh", grayThresh );
cvWaitKey(0);
...
```

How to display the results and write an output file

Displaying Images

```
cvNamedWindow( "source", 1 );         // Create a new image window
cvShowImage( "source", sourceImg);    // Display an image onto the new image window

cvWaitKey(0);                         // Wait for a keypress

cvDestroyWindow( "source" );          // Destroy the image window
```

Writing Output Image Files

```
// This method notifies user of image writing failures

char *outFileName = "outputimage.jpg";        // Output filename
if( !cvSaveImage( outFileName, image ) )      // "image" is an existing IplImage
      printf("Could not save: %s\n", outFileName );
else
      printf("Saved new image output file: %s\n", outFileName );
```

OpenCV Example: 2-D Wiener Filter with Input Arguments

SPAWAR Systems Center PACIFIC

2-D Wiener Filtering in OpenCV

- 2-D Wiener Filtering is a method for noise removal. The specific algorithm we discuss here is the same used by Matlab's wiener filtering function.

- Open the Microsoft .sln file ..\Day2\codigo\OpenCVWienerFilter

- We will go over each file in this solution, and discuss the new concepts, including input arguments, including header files for functions you write, and some standard C++ functions used.

2-D Wiener Filtering in OpenCV

```
/* test.c
*  Contains the main function for the wienerFilter executable.
*/

#include <math.h>              // Include math library
#include <string>              // Include stdlibc++ strings
#include <opencv/cv.h>         // Include the OpenCV header
#include <opencv/cxcore.h>     // Include OpenCV core
#include <opencv/cvaux.h>      // Include OpenCV Aux
#include <opencv/highgui.h>    // Include OpenCV HighGUI
#include "wienerFilter.h"      // Include the WinerFiltering functions
#include <iostream>            // For writing to streams (for example the cout
                               // stream, which we will discuss)
#include <fstream>             // Also for writing to streams


using namespace std;           // Usually will include this.
```

SPAWAR Systems Center PACIFIC

2-D Wiener Filtering in OpenCV

```
// Main function to call Wiener filtering
int main(int argc, char *argv[])
{

     // argc contains the number of arguments the user has passed
     // check to make sure argc makes sense

     if (argc < 2)
     {
          cout << "Usage: wienerFilter <input_image> " << endl;
          return -1;
     }

     // Declare the IplImage
     IplImage* input_img;

     // argv[1] is the first input argument, and we expect it to be the name
     // of an image we wish to process
     char* input_image_name = argv[1];
```

2-D Wiener Filtering in OpenCV

```
// Load the IplImage from the file specified by the user, check image

if((src = cvLoadImage(input_image_name, 0)) != 0)

 {
     // Declare a new IplImage, called input, but make it a 32 bit floating point image.
     IplImage* input_32Bit = cvCreateImage(cvSize(src->width,src->height), IPL_DEPTH_32F,1);

     // Convert the input image, which is 8-bit unsigned to a 32 bit floating point image
     cvConvert(input_img,input_32Bit);

     // Declare another IplImage to hold the ouput of the wiener filtering operation.
     // Make it the same size as the input_img

     IplImage* output =  cvCreateImage(cvSize(input_img->width,input_img->height),IPL_DEPTH_32F,1);

     // Perform 2-D Wiener filtering of image to remove noise.
     WienerFilter2D(input_32Bit, output);
```

2-D Wiener Filtering in OpenCV

```
    // Convert output from 32 bit floating point to 8 bit unsigned int
    IplImage* out = cvCreateImage(cvSize(input_img->width,input_img->height),IPL_DEPTH_8U,1);

    //convert output to be 8 bit unsinged int
    cvConvert(output,out);

    // Display input image and results
    showImage(src, out);

    cvReleaseImage(&input);
   }

   else  // Case where we can't open image

   {
    cout << "ERROR....unable to load image! " << endl;
   }

    return 0;
}
```

SPAWAR Systems Center PACIFIC

Conclusions

- OpenCV provides many image and signal processing functionalities.
- Can easily create/read/write images
- Perform operations on matrices (add, subtract, multiply, divide, etc)
- Higher level functions included (edge detection, thresholding, feature detection, etc)

Additional OpenCV Sample Code

- Can be found in ..\03_OpenCV_and_blobs\examples\OpenCV_examples\opencv_samples
  - Both source code and executable in same directory. Good way to see examples.
  - Examples showing how to perform detection, classification, and many other functions

Converting OSSIM ImageSource to an OpenCV IplImage

```
char* input = "SanDiego.ntf";
ossimInit::instance()->initialize();
ossimImageHandler *handler = ossimImageHandlerRegistry::instance()->open(ossimFilename(input));

if(handler) {
      ossimRefPtr<ossimImageData> imageSourceData;
      ossimIrect tileRect = handler->getBoundingRect(0);

      imageSourceData = handler->getTile(tileRect);

      IplImage *image = cvCreateImage(cvSize(tileRect.height(), tileRect.width()), IPL_DEPTH_8U, 1);
      CvScalar s;

      ossim_uint8 *inBuf = (ossim_uint8*)imageSourceData->getBuf(0);
      for (int i=0; i < tileRect.height(); i++) {
            for (int j=0; j < tileRect.width(); j++) {
                  s.val[0] = (int)(*inBuf);
                  cvSet2D(image,i,j,s);
                  ++inBuf;
            }
      }
      cvNamedWindow( "IplImage", CV_WINDOW_AUTOSIZE ); cvShowImage( "IplImage", image );
      cvWaitKey(0);

      cvDestroyWindow( "IplImage" ); cvReleaseImage(&image);
      delete handler;
}
else { cout << "Unable to open image = " << input << endl; }

ossimInit::instance()->finalize();          // call the finalize so the ossim can cleanup if needed.
```