SPAWAR

Systems Center
PACIFIC

Introduction to Open Computer Vision C++ Library

Distribution Statement A: Unlimited Distribution

Disclaimer

- Though SSC Pacific makes every effort to perform quality assurance on its training materials, the material in this presentation may inadvertently include technical inaccuracies or other errors. We would be grateful if users notify us of any errors or inaccuracies they may find.
- The presentation contains references to links and to third-party websites. These are provided for the convenience and interest of users and this implies neither responsibility for, nor approval of, information contained in these websites on the part of the U.S. Government. The USG makes no warranty, either express or implied, as to the accuracy, availability or content of information, text, graphics in the links/third party websites. The USG has not tested any software located at these sites and does not make any representation as to the quality, safety, reliability or suitability of such software, nor does this presentation serve to endorse the use of such sites.

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

Recursos necesarios

- ..\Day2\imagenes\apple.bmp
- ..\Day2\codigo\OpenCVEdgeDetect\OpenCVEdgeDetect.cpp
- ..\Day2\codigo\OpenCVThreshold\OpenCVEdgeThreshold.cpp

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

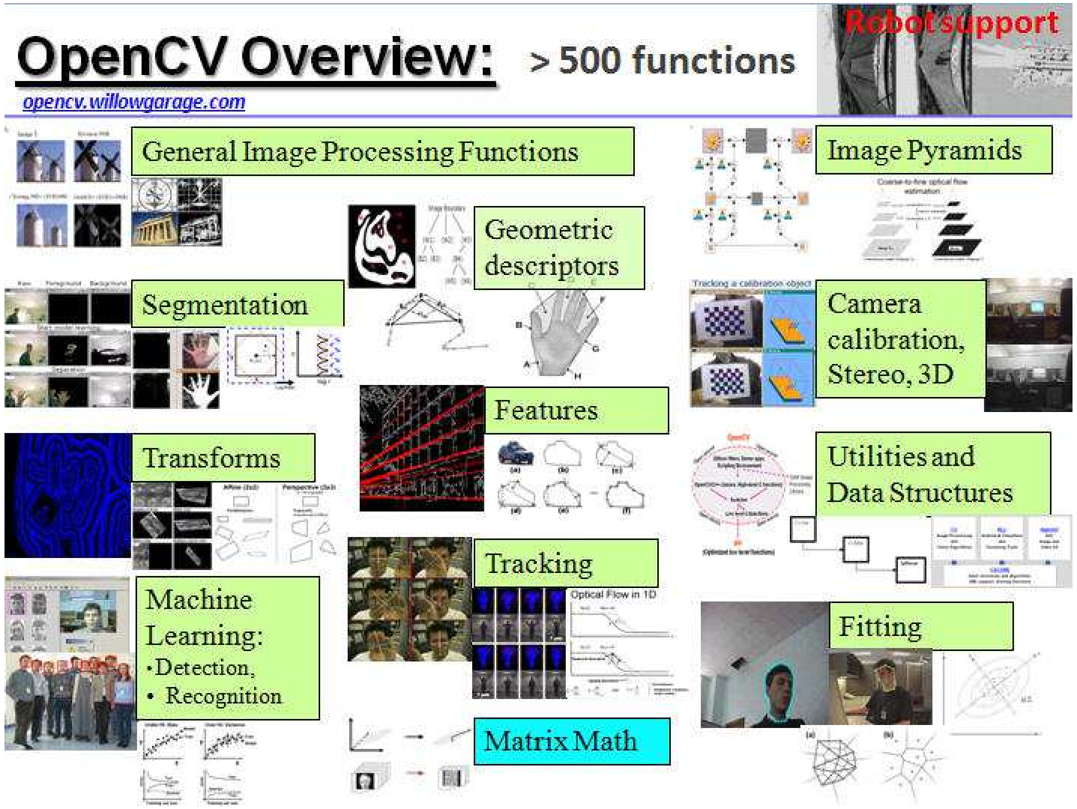
OpenCV Resources

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



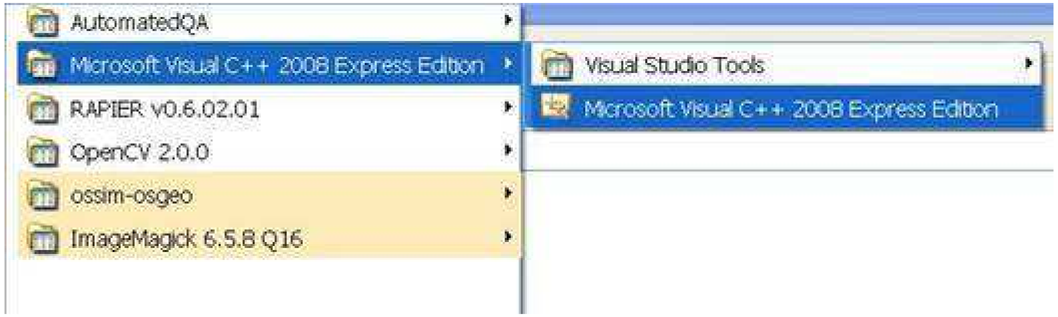
SPAWAR
Systems Center
PACIFIC

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

Distribution Statement A: Unlimited Distribution

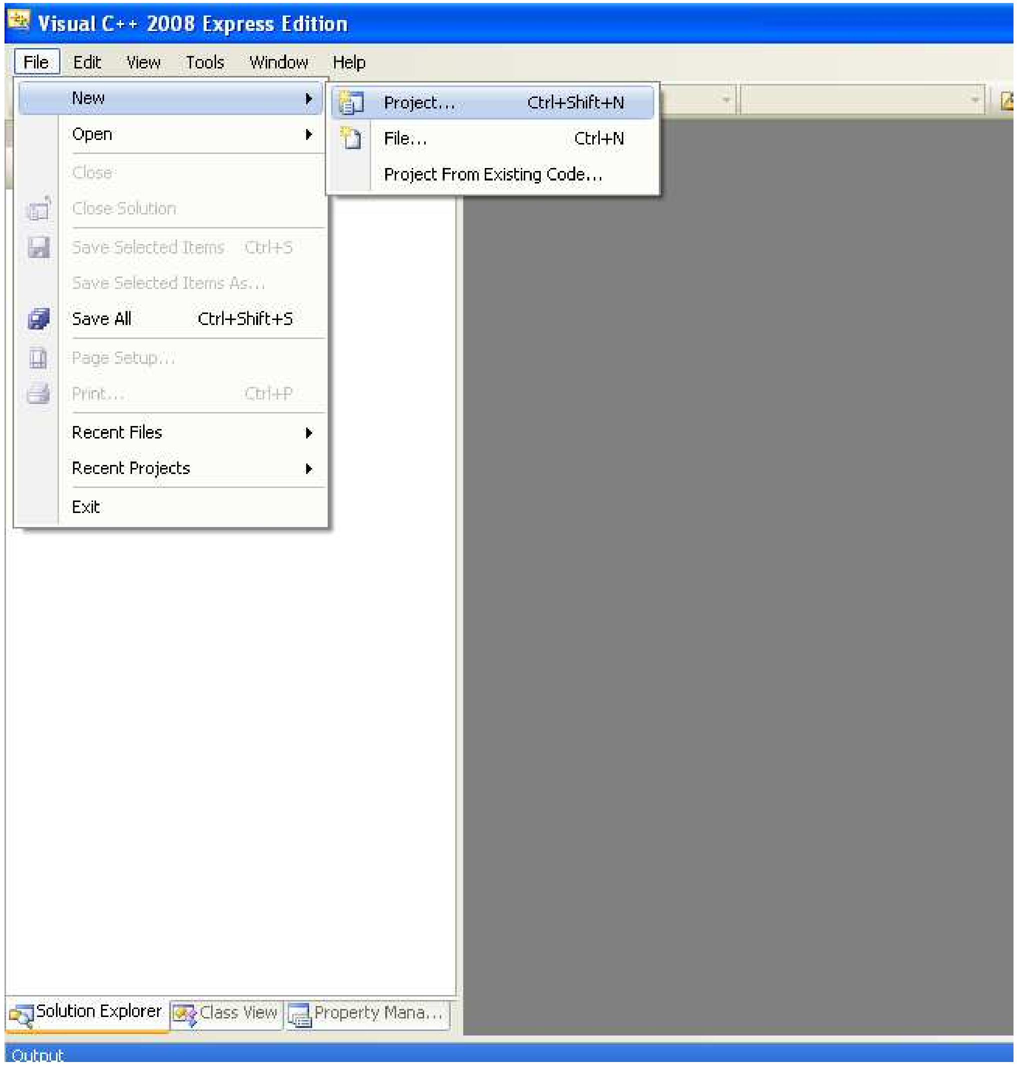
Creating A New MS VC++ Project

- Open Microsoft Visual C++ in the Start Menu



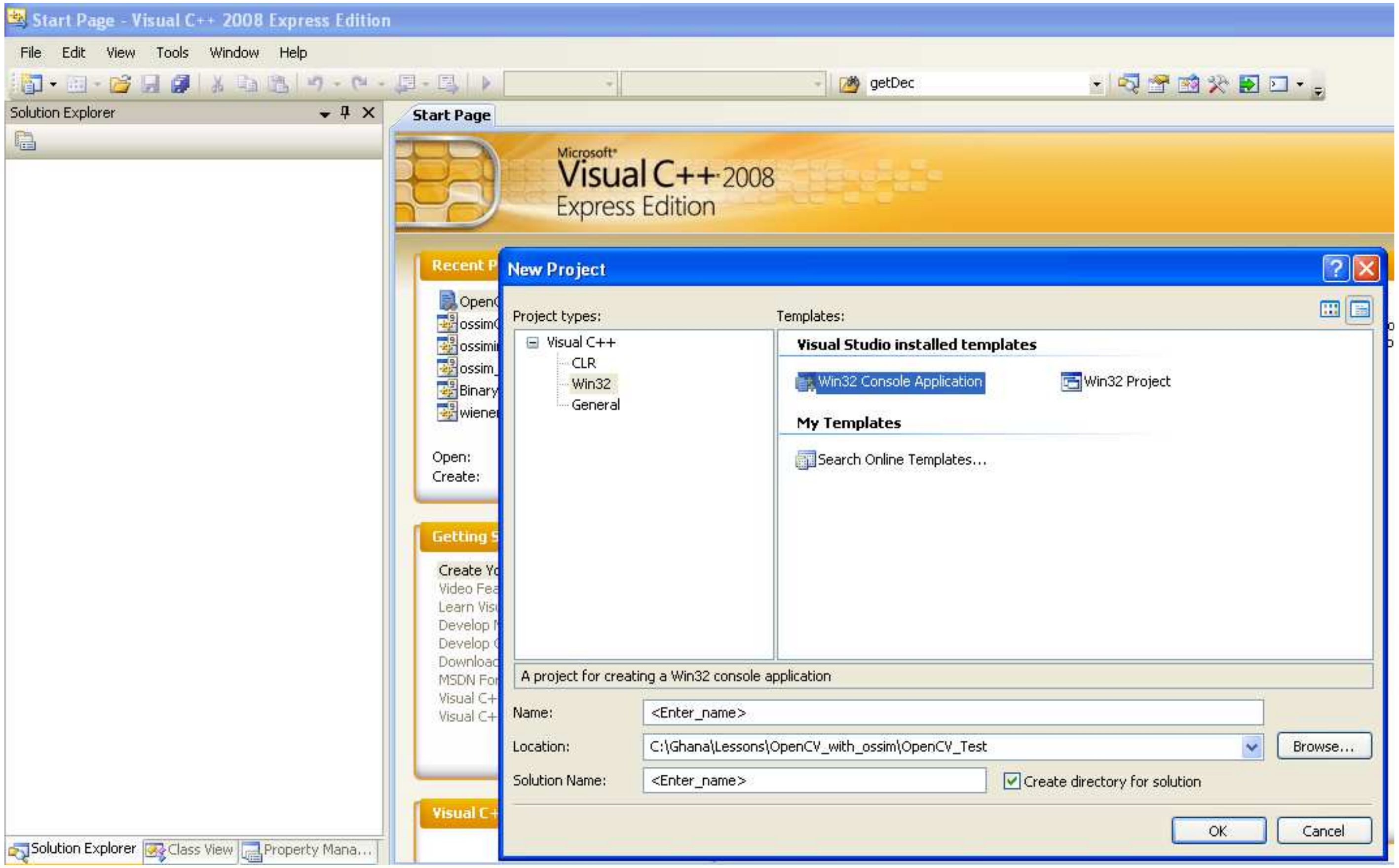
Writing an Example OpenCV Executable

- Select File->New->Project…



Writing Your Own Filter as an OSSIM Plugin

- Select Win32 on the left
- Select Win32 Console Application as the Template
- Choose a Name for your executable and a location, click OK



Start Page - Visual C++ 2008 Express Edition
File Edit View Tools Window Help
getDec
Solution Explorer
Start Page
Microsoft® Visual C++ 2008 Express Edition
Recent P
Getting S
Solution Explorer
Class View
Property Mana...

New Project

Project types:
- Visual C++
  - CLR
  - Win32
  - General

Templates:

Visual Studio installed templates
- Win32 Console Application
- Win32 Project

My Templates
- Search Online Templates...

A project for creating a Win32 console application

Name: <Enter_name>

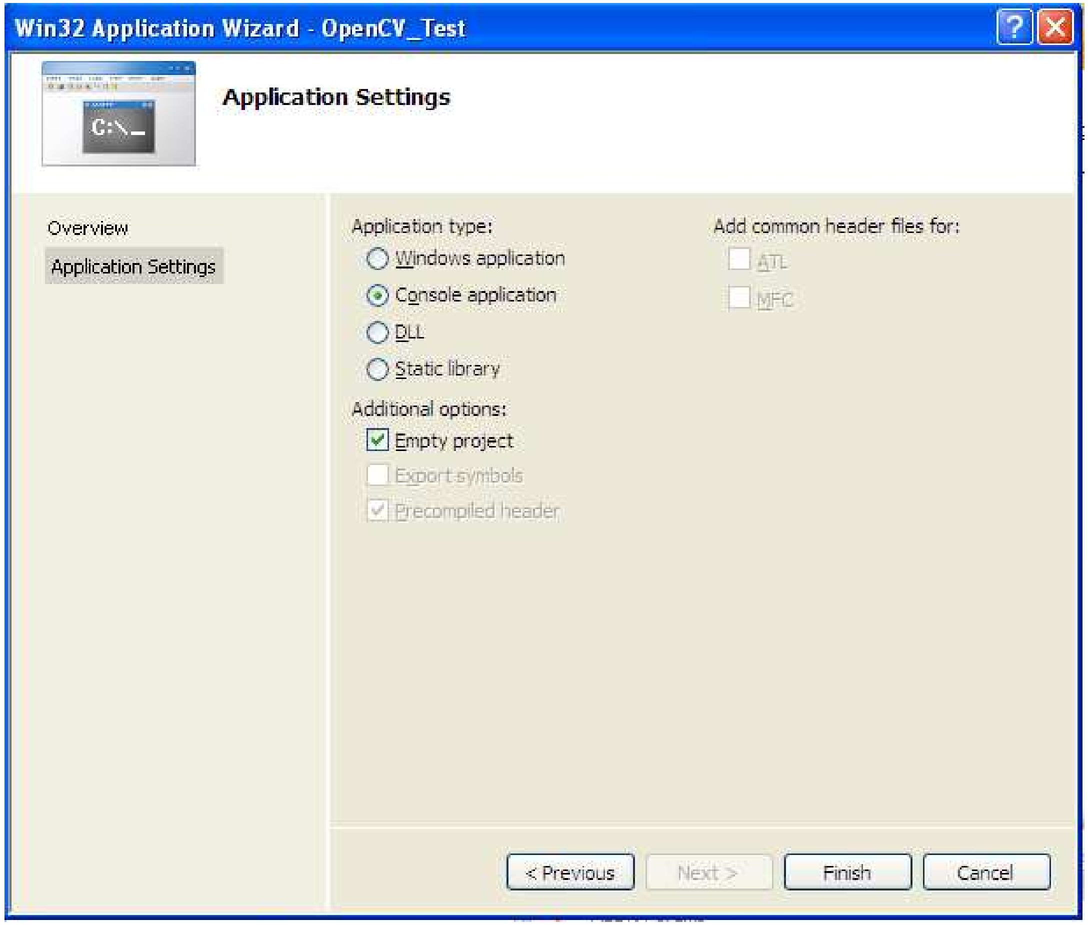
Location: C:\Ghana\Lessons\OpenCV_with_ossim\OpenCV_Test

Solution Name: <Enter_name>

☑ Create directory for solution

Writing an Example OpenCV Executable

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


Win32 Application Wizard - OpenCV_Test

Application Settings

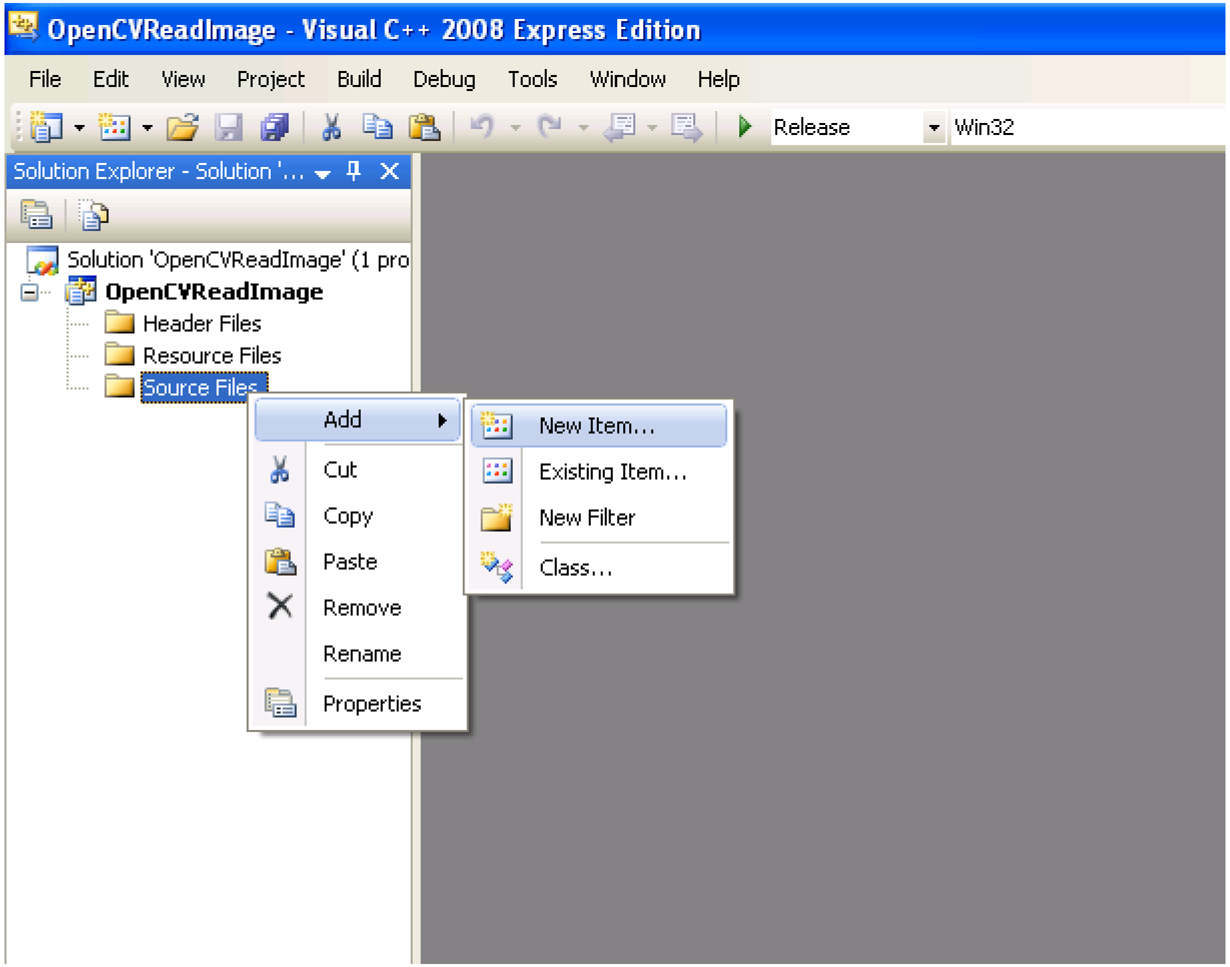
Application type:
○ Windows application
● Console application
○ DLL
○ Static library

Additional options:
☑ Empty project
□ Export symbols
☑ Precompiled header

Add common header files for:
□ ATL
□ MFC

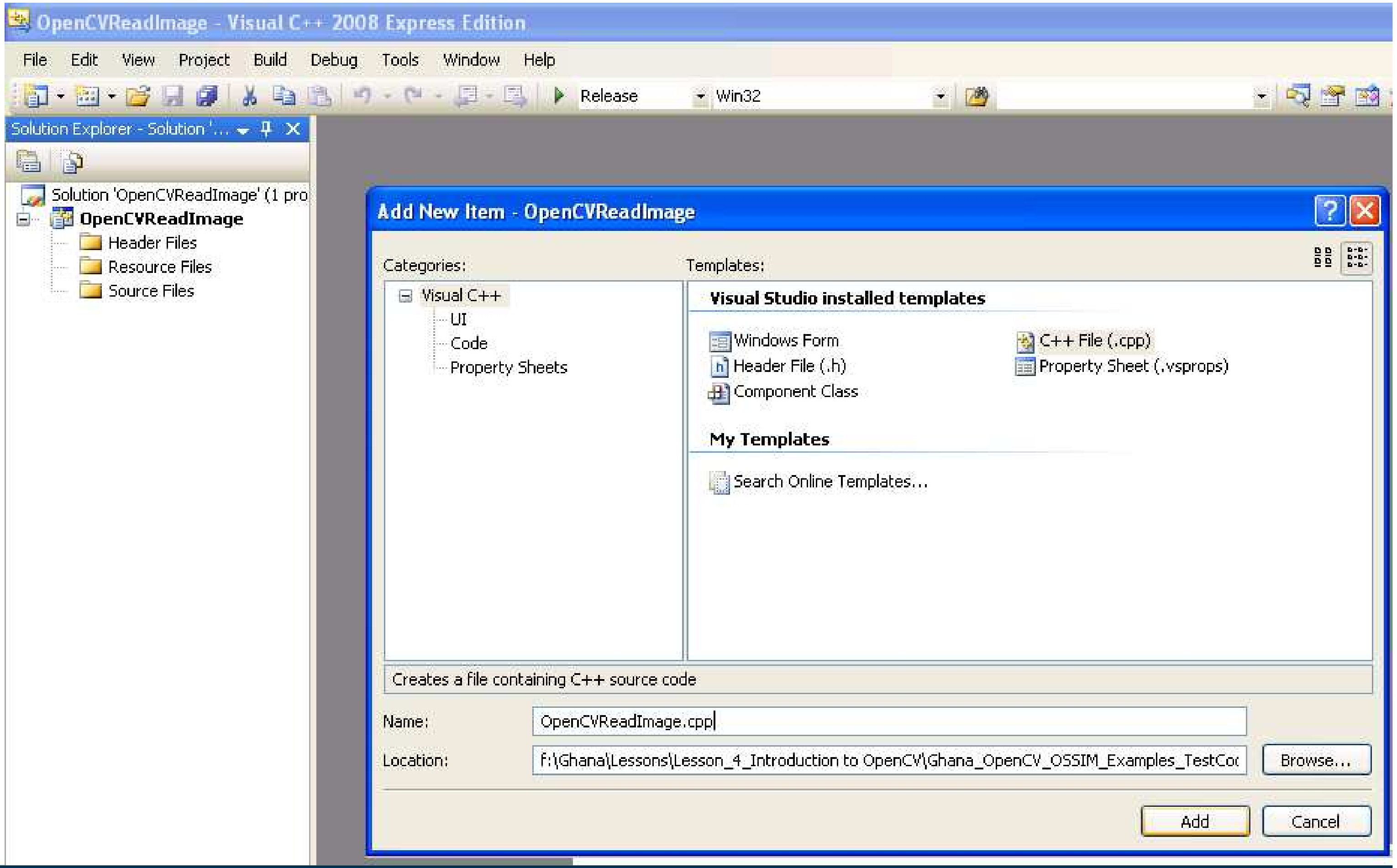
Writing an Example OpenCV Executable

- 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



Creates a file containing C++ source code

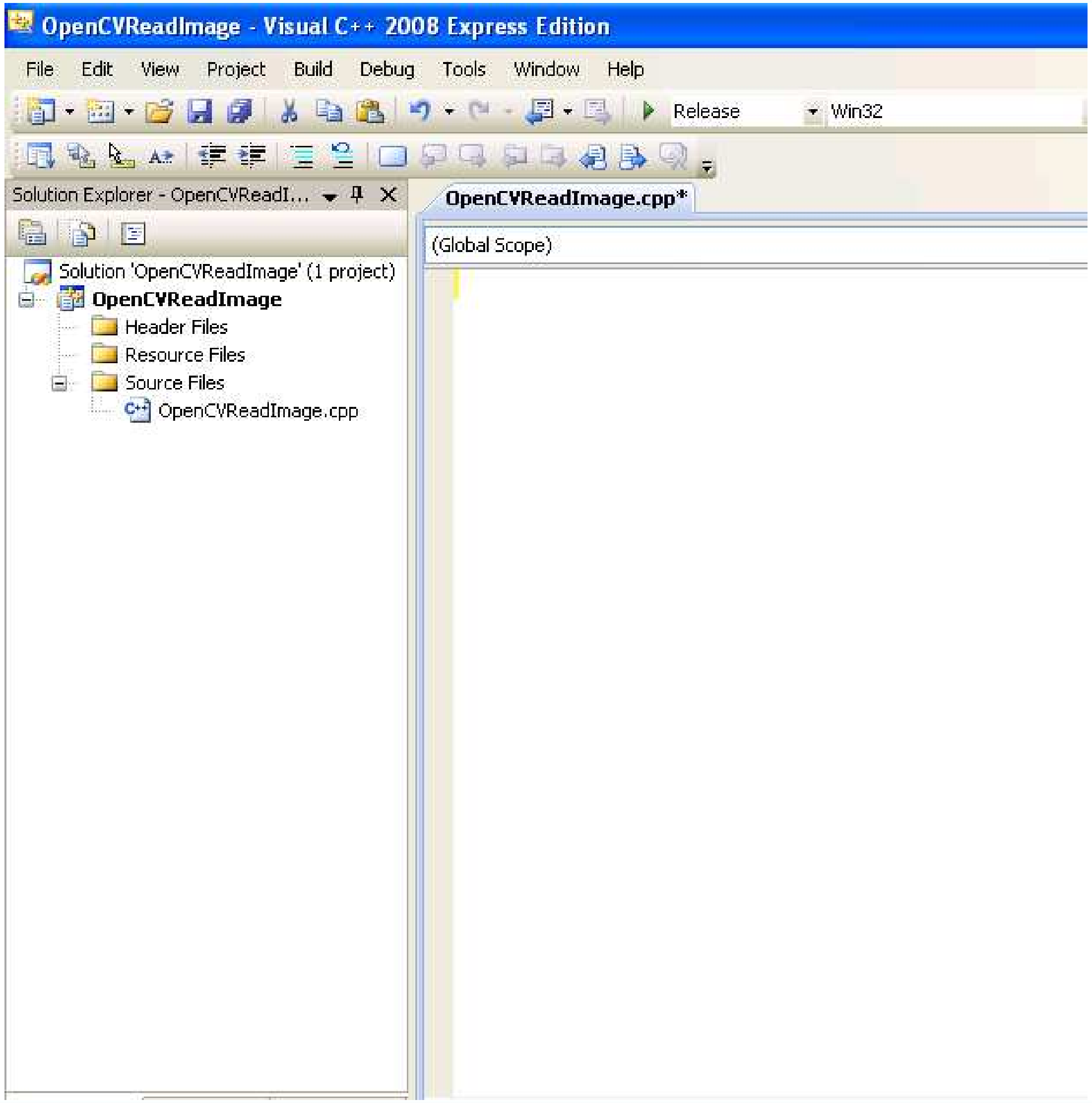
Name: OpenCVReadImage.cpp

Location: f:\Ghana\Lessons\Lesson_4_Introduction to OpenCV\Ghana_OpenCV_OSSIM_Examples_TestCo(

Writing an Example OpenCV Executable

- You should see an empty C++ file.

- We'll add code in a few slides.

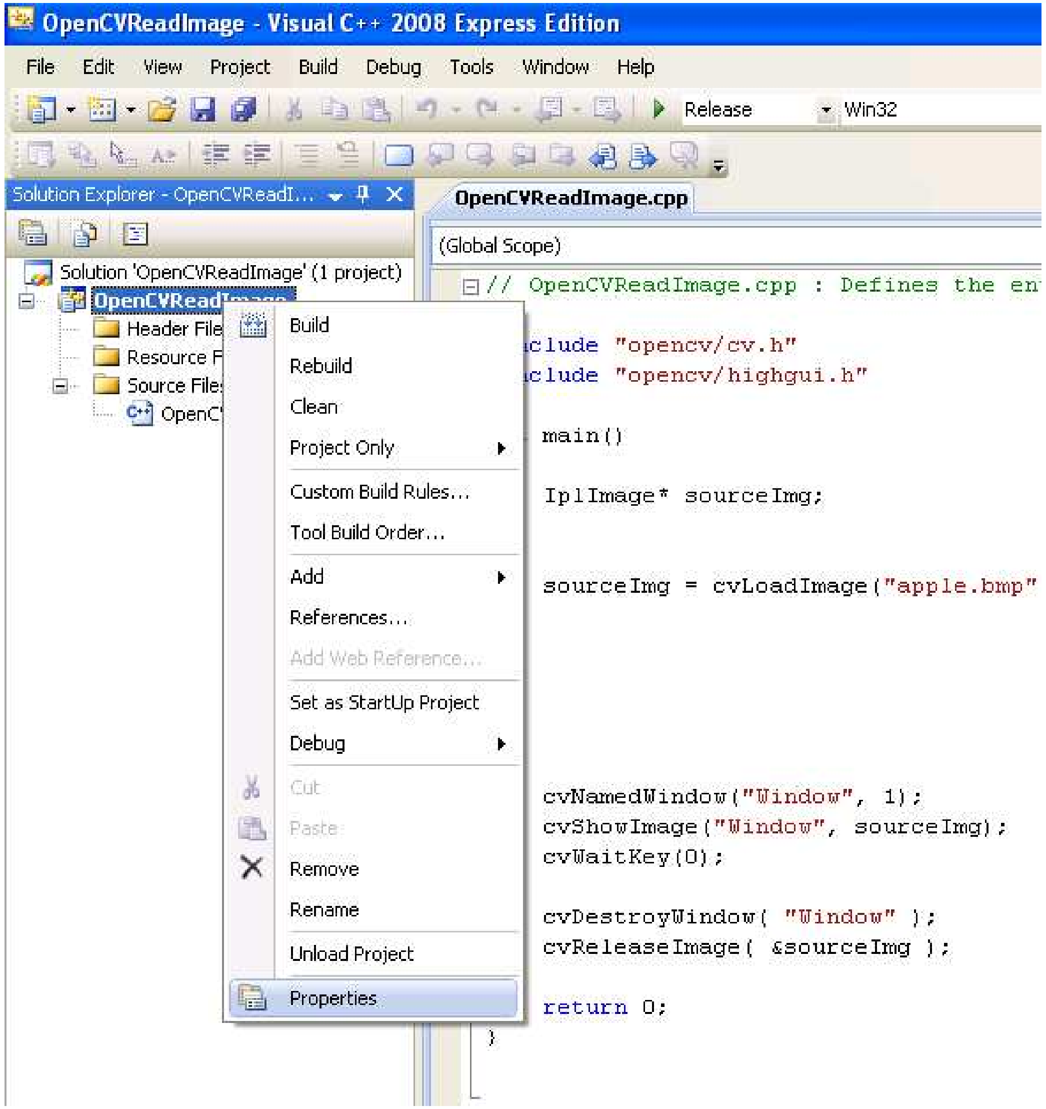

OpenCVReadImage - Visual C++ 2008 Express Edition
File Edit View Project Build Debug Tools Window Help
Release
Win32
Solution Explorer - OpenCVReadI...
Solution 'OpenCVReadImage' (1 project)
OpenCVReadImage
Header Files
Resource Files
Source Files
OpenCVReadImage.cpp
OpenCVReadImage.cpp*
(Global Scope)

OSSIM Training NOV11

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.

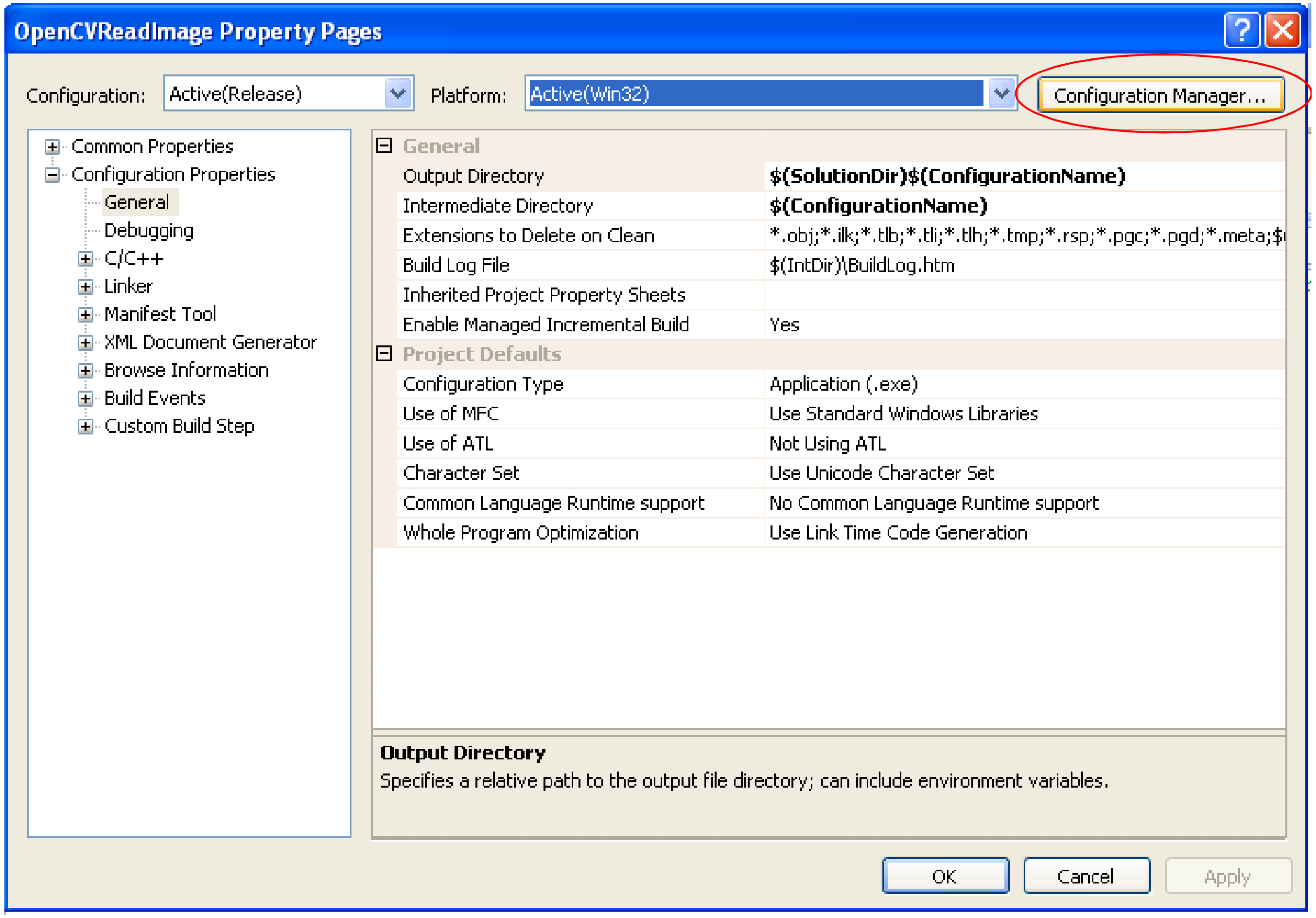
- Right click on the project name, then click on Properties



```
// OpenCVReadImage.cpp : Defines the en

clude "opencv/cv.h"
clude "opencv/highgui.h"

 main()

 IplImage* sourceImg;

 sourceImg = cvLoadImage("apple.bmp"

 cvNamedWindow("Window", 1);
 cvShowImage("Window", sourceImg);
 cvWaitKey(0);

 cvDestroyWindow( "Window" );
 cvReleaseImage( &sourceImg );

 return 0;
}
```

Writing an Example OpenCV Executable

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



Configuration: Active(Release)   Platform: Active(Win32)

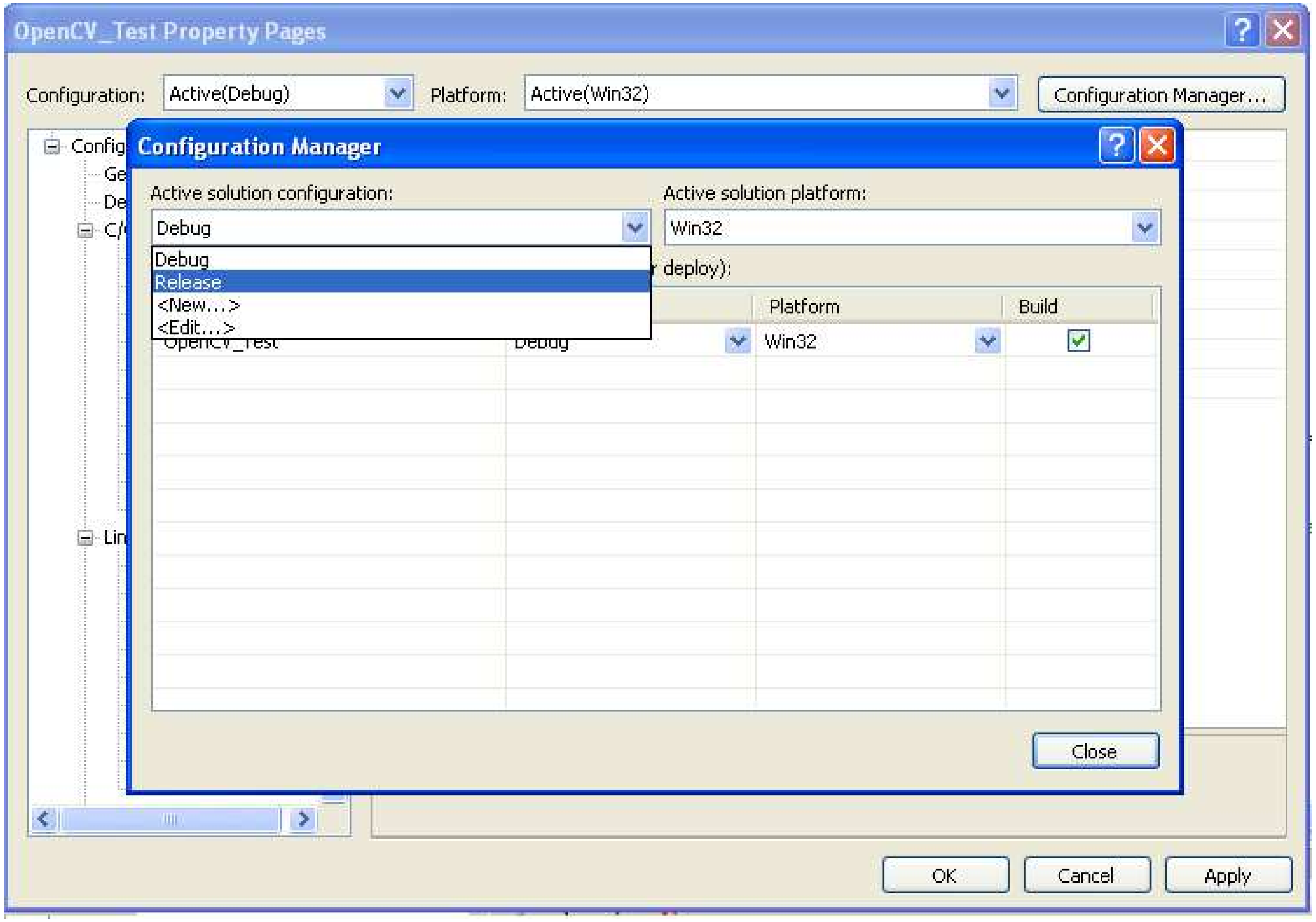
- Common Properties
- Configuration Properties
  - General
  - Debugging
  - C/C++
  - Linker
  - Manifest Tool
  - XML Document Generator
  - Browse Information
  - Build Events
  - Custom Build Step

General	
Output Directory	**$(SolutionDir)$(ConfigurationName)**
Intermediate Directory	**$(ConfigurationName)**
Extensions to Delete on Clean	*.obj;*.ilk;*.tlb;*.tli;*.tlh;*.tmp;*.rsp;*.pgc;*.pgd;*.meta;$(
Build Log File	$(IntDir)\BuildLog.htm
Inherited Project Property Sheets	
Enable Managed Incremental Build	Yes
Project Defaults	
Configuration Type	Application (.exe)
Use of MFC	Use Standard Windows Libraries
Use of ATL	Not Using ATL
Character Set	Use Unicode Character Set
Common Language Runtime support	No Common Language Runtime support
Whole Program Optimization	Use Link Time Code Generation

Output Directory
Specifies a relative path to the output file directory; can include environment variables.

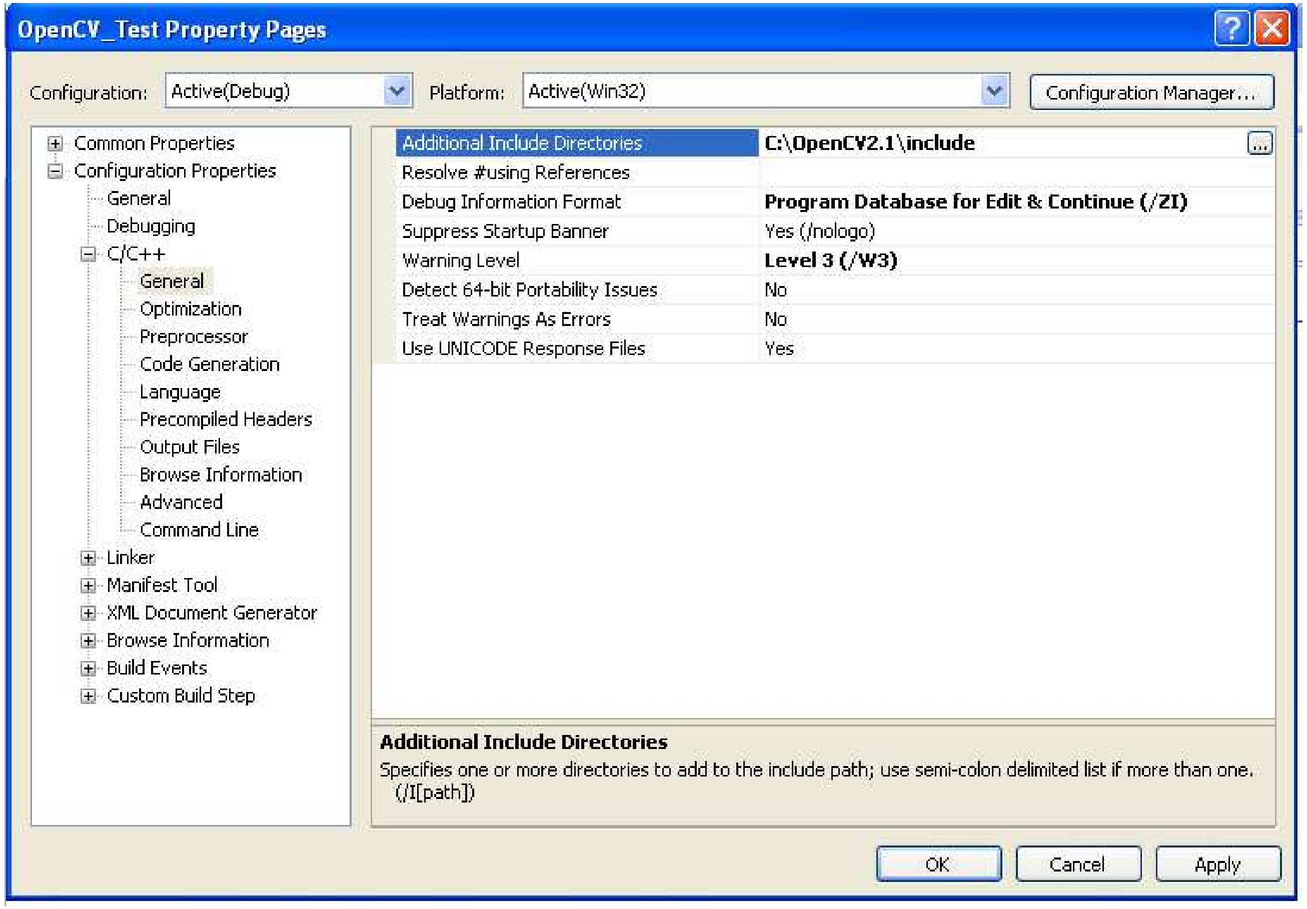
Writing an Example OpenCV Executable

- 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.



OpenCV_Test Property Pages

Configuration: Active(Debug) Platform: Active(Win32)

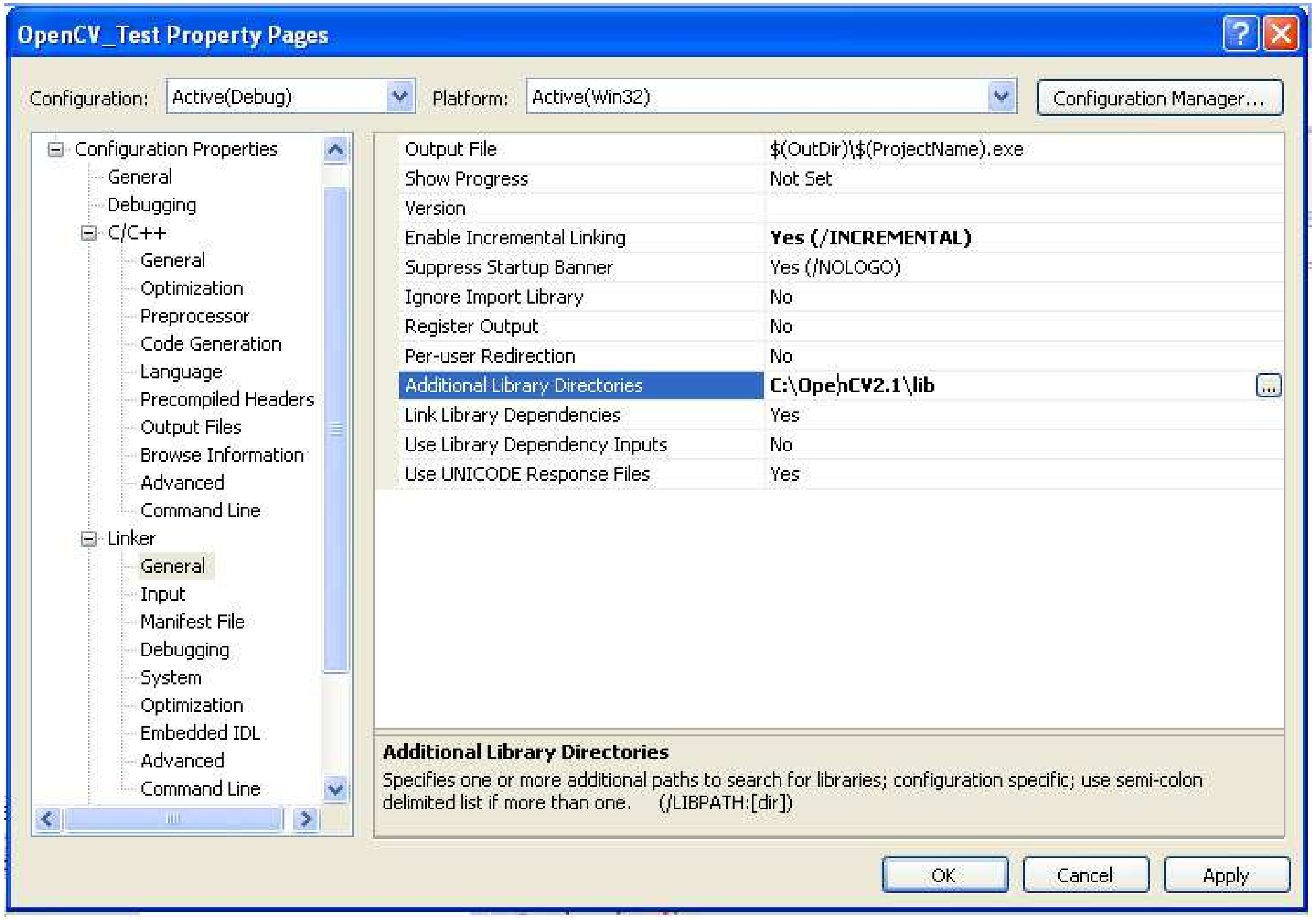
Additional Include Directories	C:\OpenCV2.1\include
Resolve #using References	
Debug Information Format	Program Database for Edit & Continue (/ZI)
Suppress Startup Banner	Yes (/nologo)
Warning Level	Level 3 (/W3)
Detect 64-bit Portability Issues	No
Treat Warnings As Errors	No
Use UNICODE Response Files	Yes

Additional Include Directories

Specifies one or more directories to add to the include path; use semi-colon delimited list if more than one. (/I[path])

Writing an Example OpenCV Executable

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



Output File	$(OutDir)\$(ProjectName).exe
Show Progress	Not Set
Version	
Enable Incremental Linking	**Yes (/INCREMENTAL)**
Suppress Startup Banner	Yes (/NOLOGO)
Ignore Import Library	No
Register Output	No
Per-user Redirection	No
Additional Library Directories	**C:\OpenCV2.1\lib**
Link Library Dependencies	Yes
Use Library Dependency Inputs	No
Use UNICODE Response Files	Yes

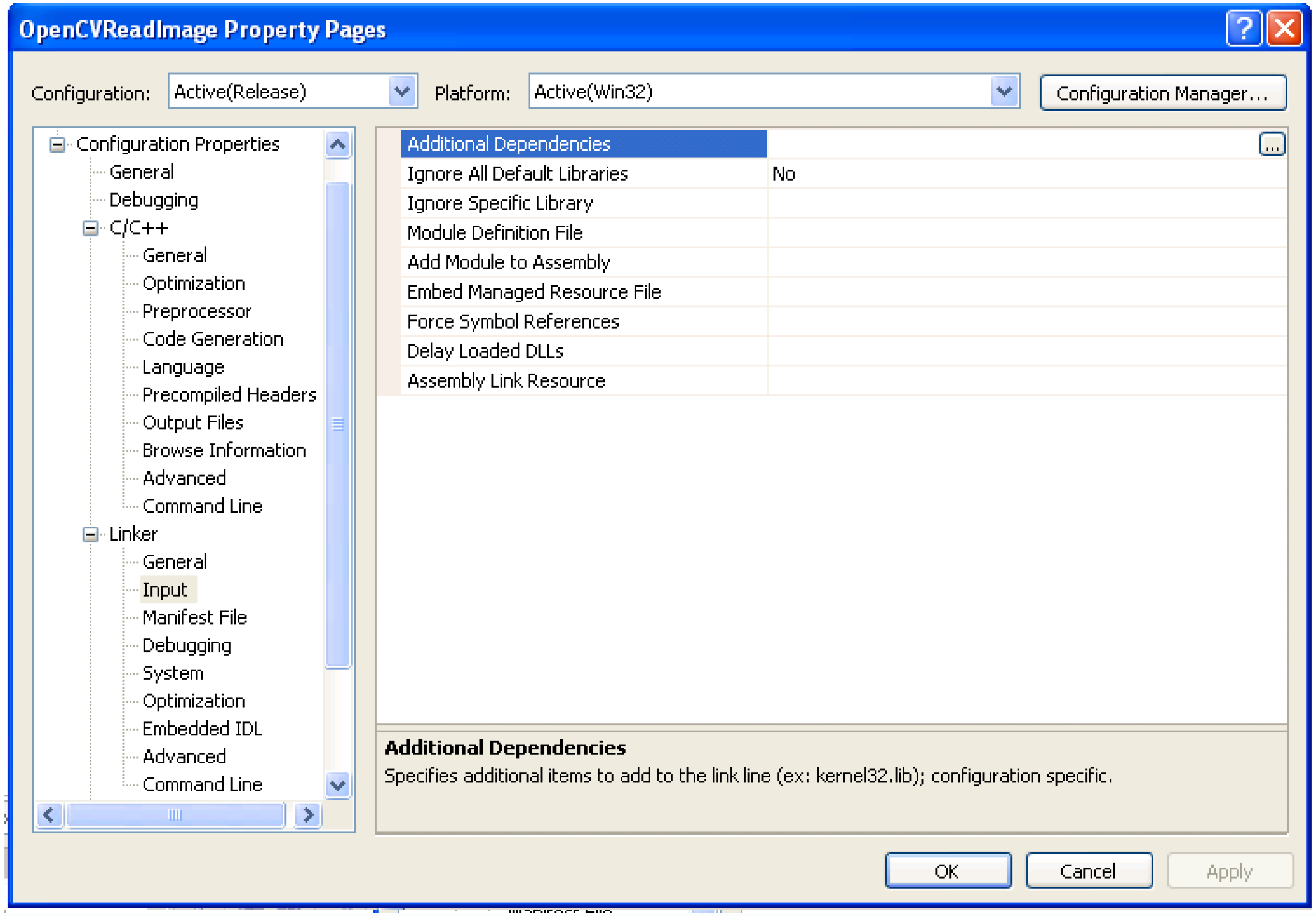
Additional Library Directories

Specifies one or more additional paths to search for libraries; configuration specific; use semi-colon delimited list if more than one.   (/LIBPATH:[dir])

SPAWAR Systems Center PACIFIC

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.



OpenCVReadImage Property Pages

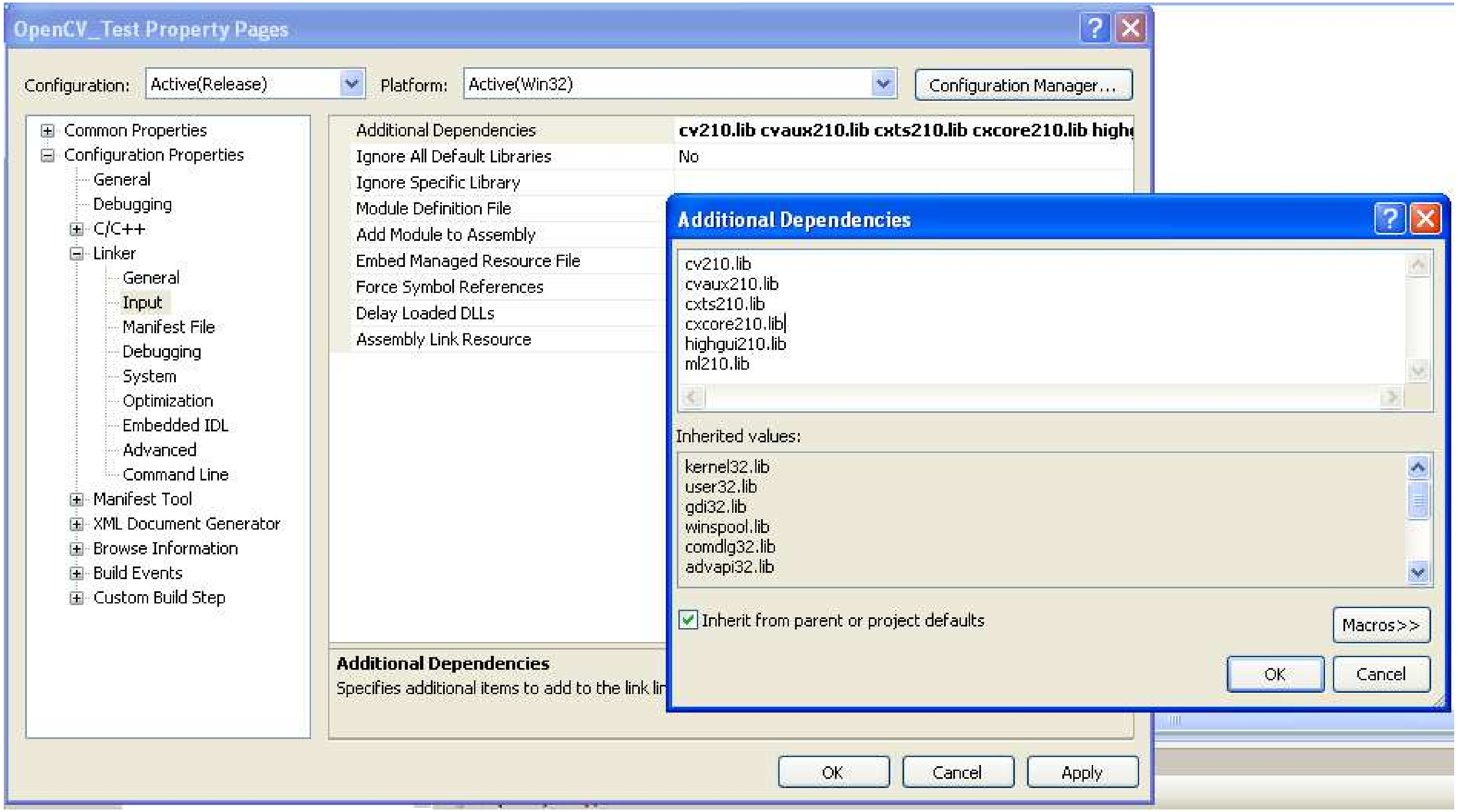
Configuration: Active(Release)
Platform: Active(Win32)

Property	Value
Additional Dependencies	
Ignore All Default Libraries	No
Ignore Specific Library	
Module Definition File	
Add Module to Assembly	
Embed Managed Resource File	
Force Symbol References	
Delay Loaded DLLs	
Assembly Link Resource	

Additional Dependencies
Specifies additional items to add to the link line (ex: kernel32.lib); configuration specific.

Writing an Example OpenCV Executable

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



OpenCV_Test Property Pages

Configuration: Active(Release) Platform: Active(Win32)

Additional Dependencies: cv210.lib cvaux210.lib cxts210.lib cxcore210.lib highg

Ignore All Default Libraries: No

Additional Dependencies

Specifies additional items to add to the link lin

Additional Dependencies

```
cv210.lib
cvaux210.lib
cxts210.lib
cxcore210.lib
highgui210.lib
ml210.lib
```

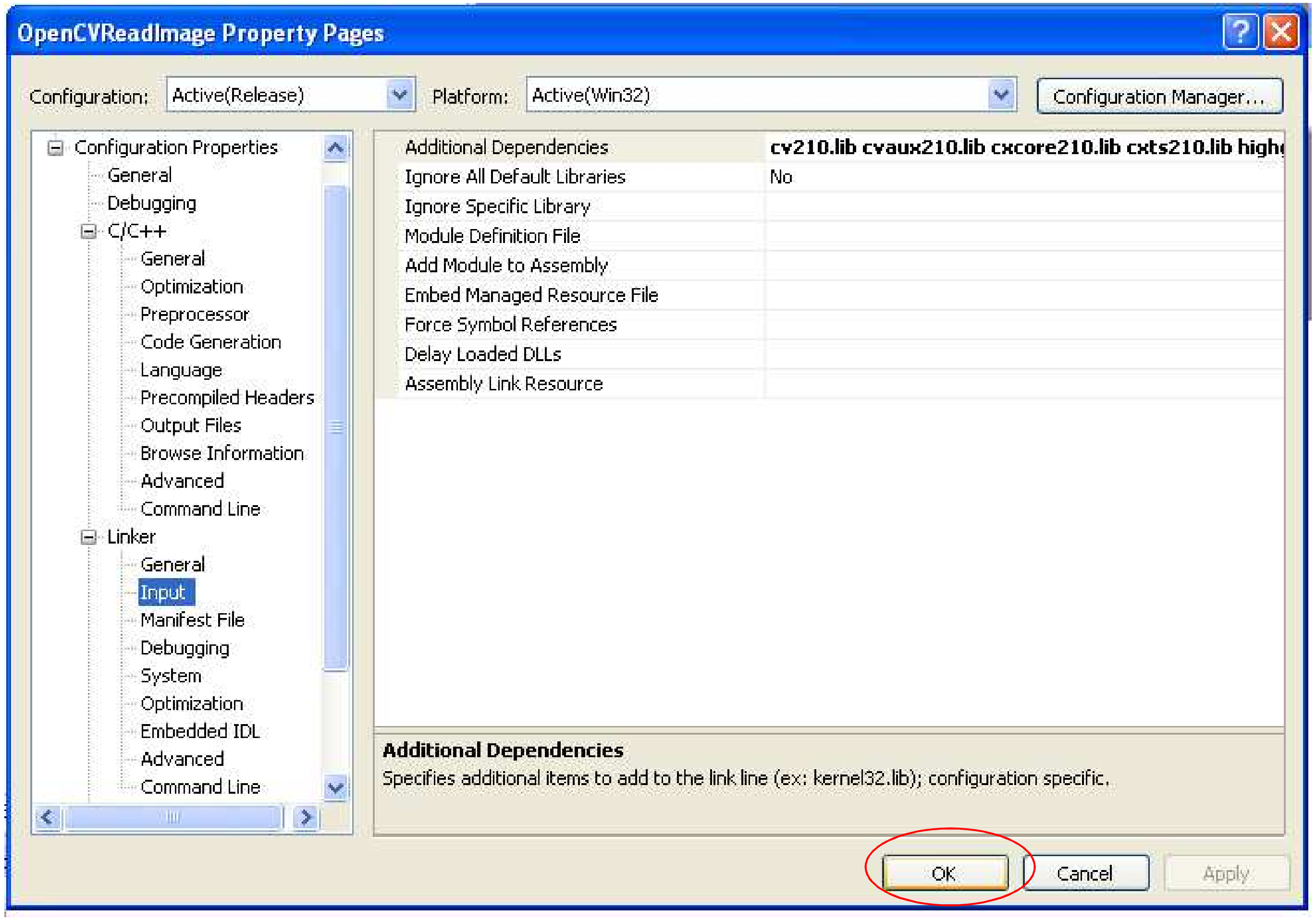
Inherited values:

```
kernel32.lib
user32.lib
gdi32.lib
winspool.lib
comdlg32.lib
advapi32.lib
```

☑ Inherit from parent or project defaults

Writing an Example OpenCV Executable

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



OpenCVReadImage Property Pages

Configuration: Active(Release) Platform: Active(Win32)

Property	Value
Additional Dependencies	cv210.lib cvaux210.lib cxcore210.lib cxts210.lib high
Ignore All Default Libraries	No
Ignore Specific Library	
Module Definition File	
Add Module to Assembly	
Embed Managed Resource File	
Force Symbol References	
Delay Loaded DLLs	
Assembly Link Resource	

Additional Dependencies

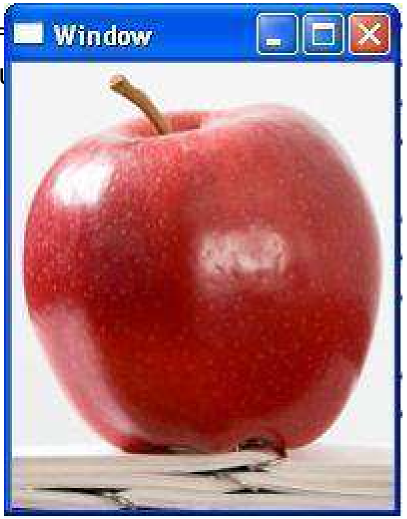
Specifies additional items to add to the link line (ex: kernel32.lib); configuration specific.

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

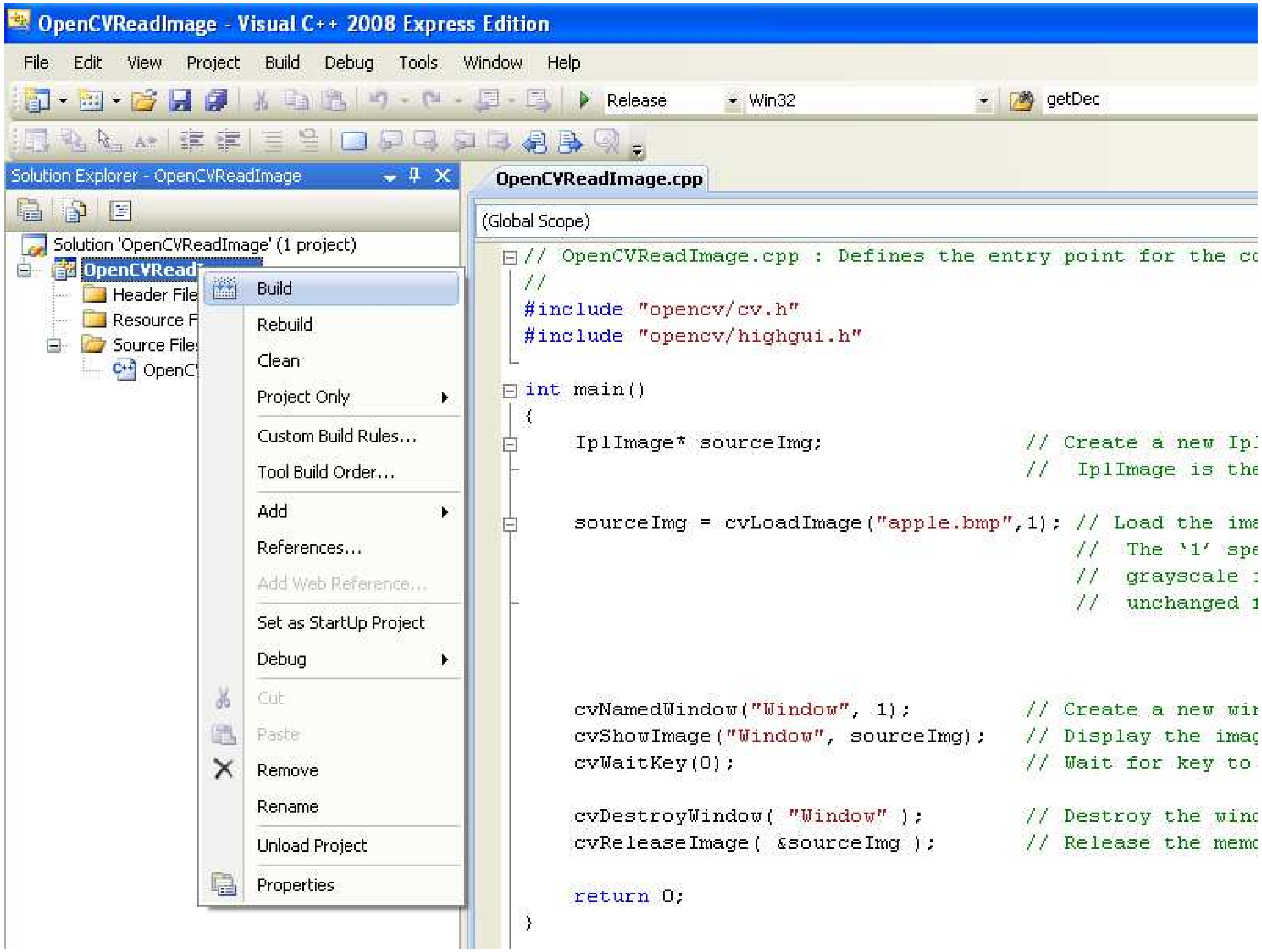
- ## Creating and Reading an Image

```
#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 inf
                                                                            //  u

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

    cvWaitKey(0);                                                    // Wait for key

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

    return 0;
}
```



Writing an Example OpenCV Executable

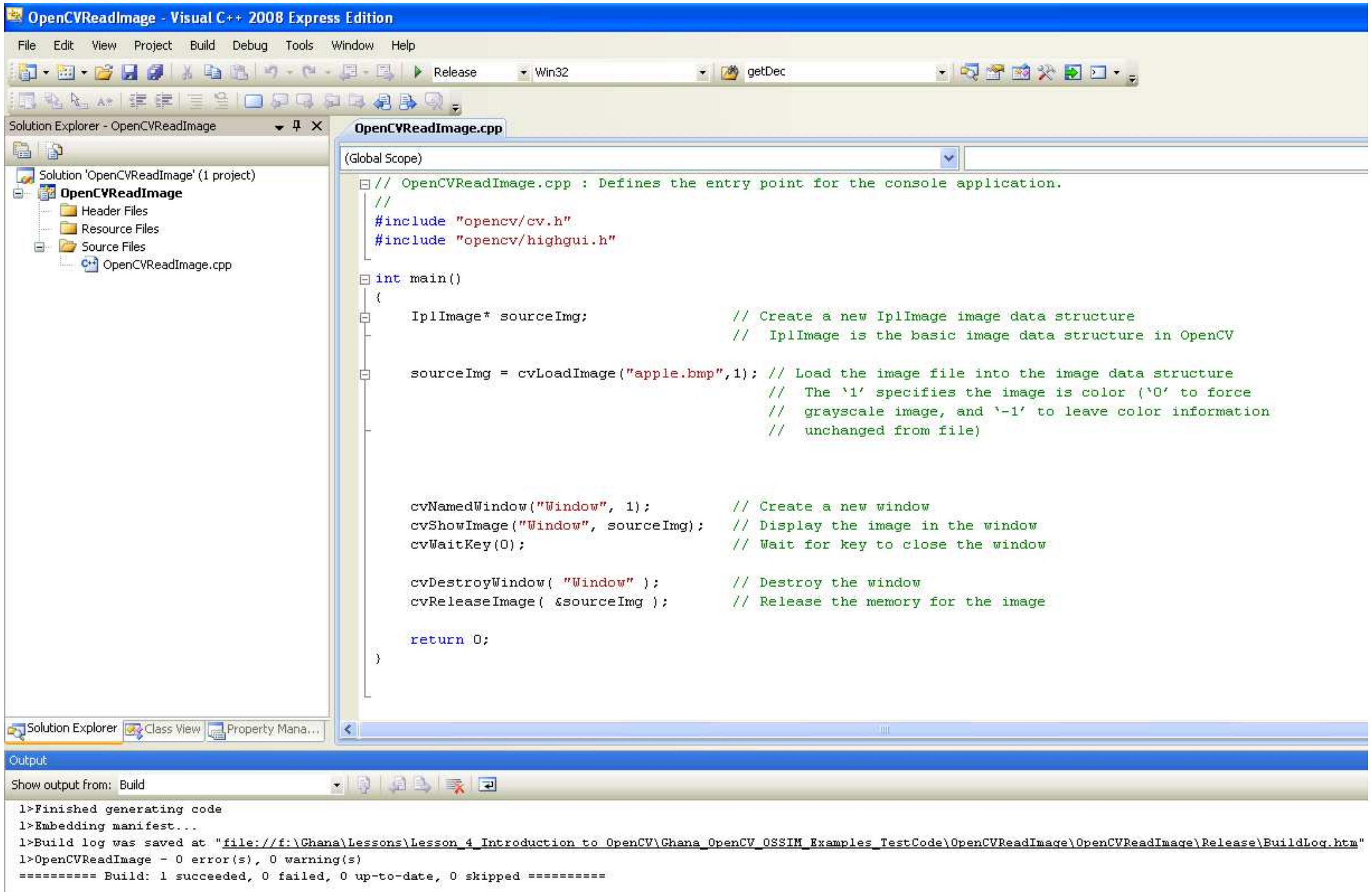
- Once you have completed your code, right click on the project, then click build.



```
// OpenCVReadImage.cpp : Defines the entry point for the co
//
#include "opencv/cv.h"
#include "opencv/highgui.h"

int main()
{
    IplImage* sourceImg;                     // Create a new Ipl
                                             //  IplImage is the

    sourceImg = cvLoadImage("apple.bmp",1); // Load the ima
                                            //  The '1' spe
                                            //  grayscale i
                                            //  unchanged i



    cvNamedWindow("Window", 1);              // Create a new wi
    cvShowImage("Window", sourceImg);        // Display the imag
    cvWaitKey(0);                            // Wait for key to

    cvDestroyWindow( "Window" );             // Destroy the wind
    cvReleaseImage( &sourceImg );            // Release the memo

    return 0;
}
```

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.

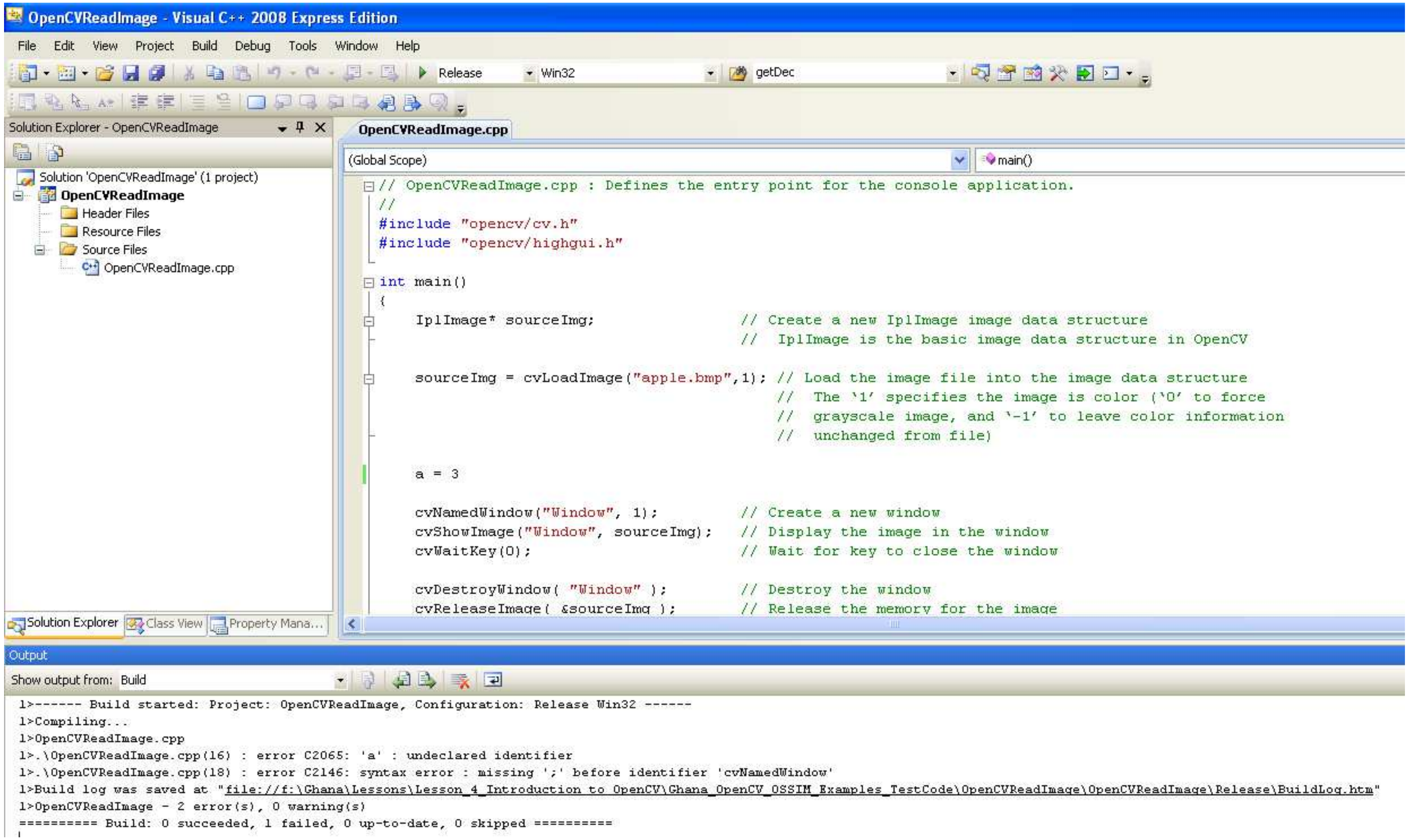


```
// OpenCVReadImage.cpp : Defines the entry point for the console application.
//
#include "opencv/cv.h"
#include "opencv/highgui.h"

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;
}
```

```
1>Finished generating code
1>Embedding manifest...
1>Build log was saved at "file://f:\Ghana\Lessons\Lesson_4_Introduction to OpenCV\Ghana_OpenCV_OSSIM_Examples_TestCode\OpenCVReadImage\OpenCVReadImage\Release\BuildLog.htm"
1>OpenCVReadImage - 0 error(s), 0 warning(s)
========== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped ==========
```

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.

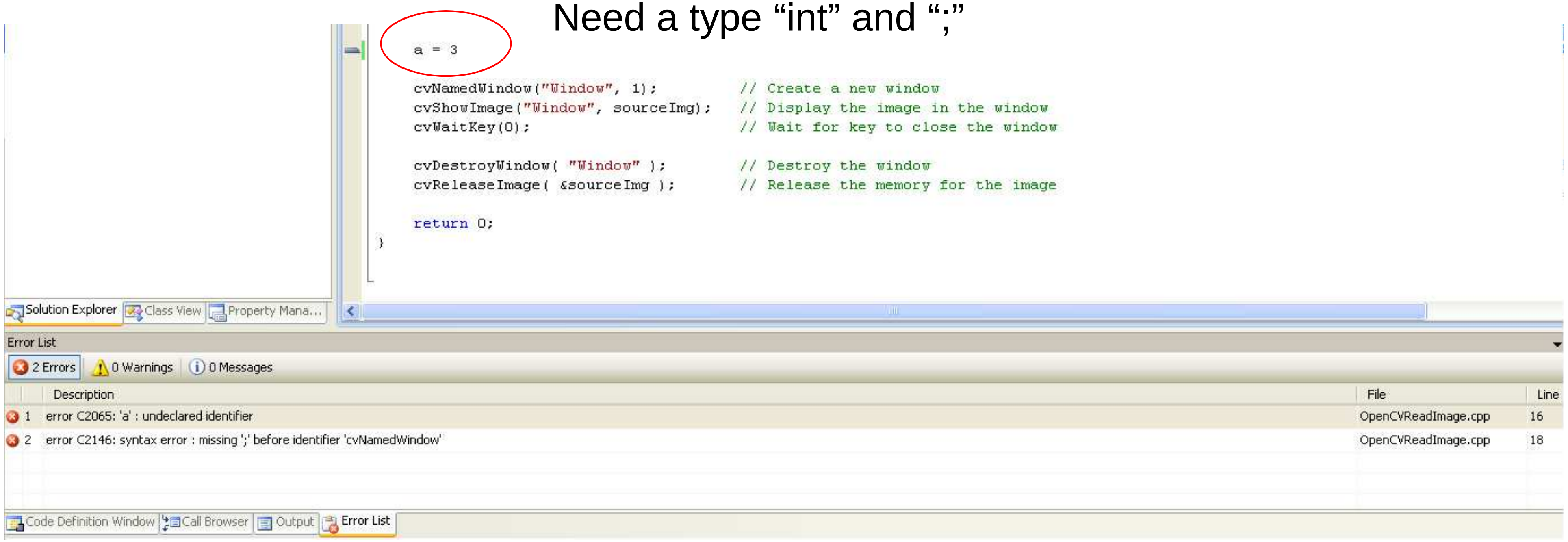


```
// OpenCVReadImage.cpp : Defines the entry point for the console application.
//
#include "opencv/cv.h"
#include "opencv/highgui.h"

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)

    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
```

```
1>------ Build started: Project: OpenCVReadImage, Configuration: Release Win32 ------
1>Compiling...
1>OpenCVReadImage.cpp
1>.\OpenCVReadImage.cpp(16) : error C2065: 'a' : undeclared identifier
1>.\OpenCVReadImage.cpp(18) : error C2146: syntax error : missing ';' before identifier 'cvNamedWindow'
1>Build log was saved at "file://f:\Ghana\Lessons\Lesson_4_Introduction to OpenCV\Ghana_OpenCV_OSSIM_Examples_TestCode\OpenCVReadImage\OpenCVReadImage\Release\BuildLog.htm"
1>OpenCVReadImage - 2 error(s), 0 warning(s)
========== Build: 0 succeeded, 1 failed, 0 up-to-date, 0 skipped ==========
```

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;
}
```

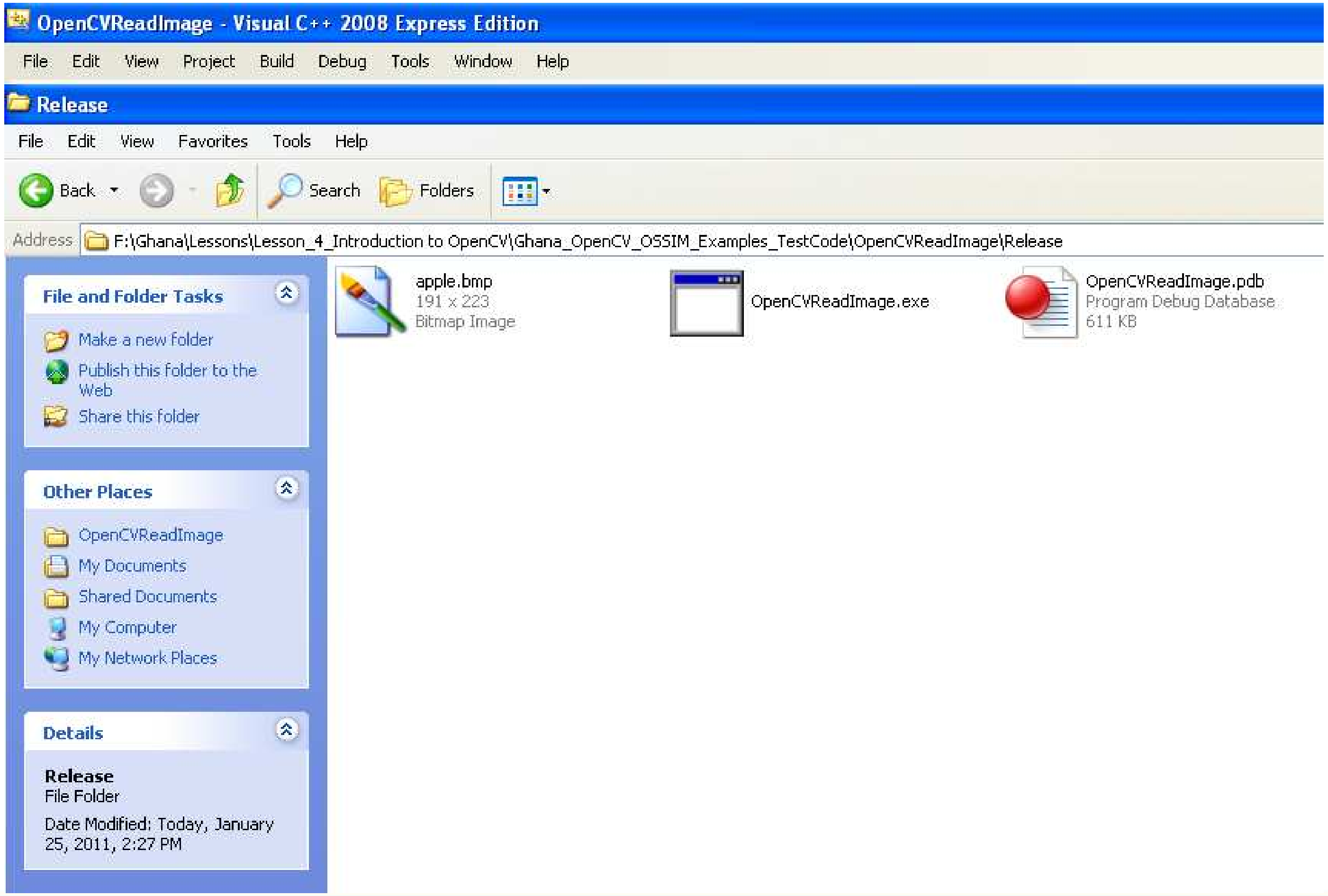
Error List

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

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.


OpenCVReadImage - Visual C++ 2008 Express Edition
File Edit View Project Build Debug Tools Window Help
Release
File Edit View Favorites Tools Help
Back
Search
Folders
Address

F:\Ghana\Lessons\Lesson_4_Introduction to OpenCV\Ghana_OpenCV_OSSIM_Examples_TestCode\OpenCVReadImage\Release

Release
File Folder
Date Modified: Today, January 25, 2011, 2:27 PM

apple.bmp
191 x 223
Bitmap Image

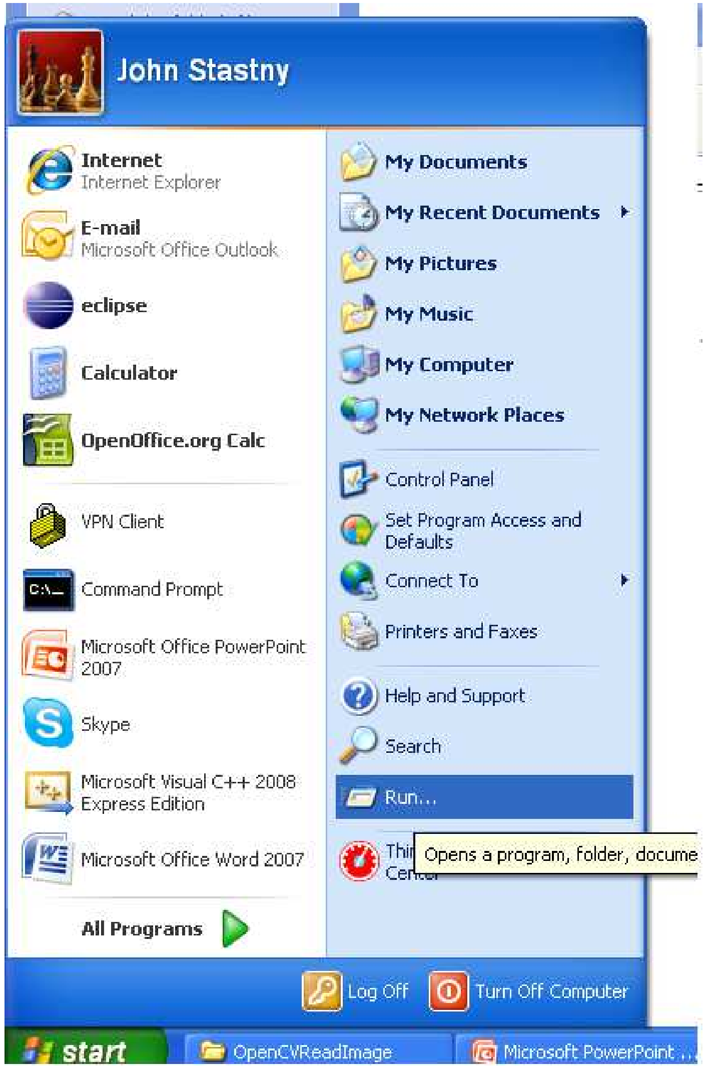
OpenCVReadImage.exe

OpenCVReadImage.pdb
Program Debug Database
611 KB

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.

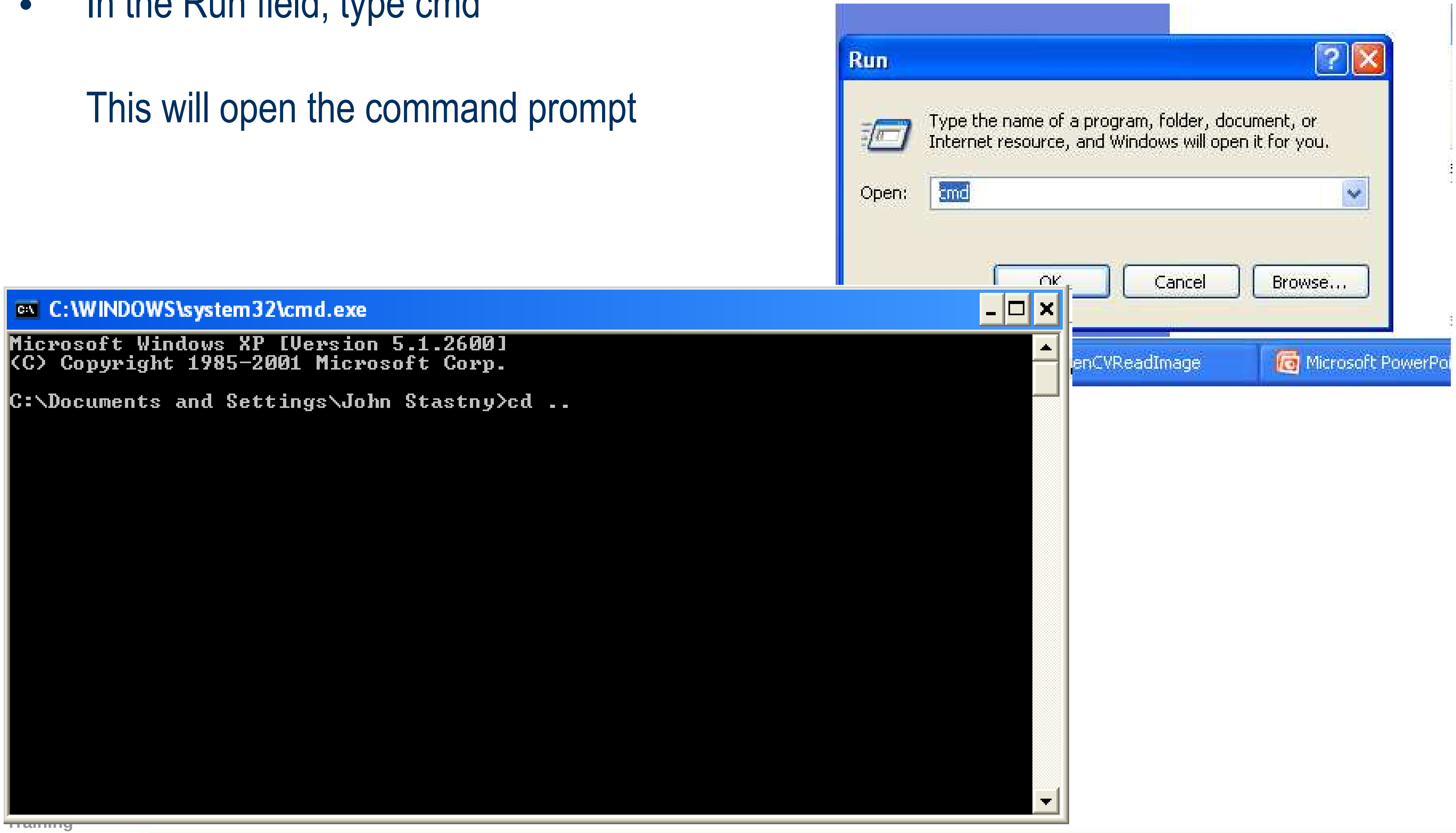
- Go to Program Files-Run



Writing an Example OpenCV Executable

- In the Run field, type cmd

  This will open the command prompt



Run

Type the name of a program, folder, document, or Internet resource, and Windows will open it for you.

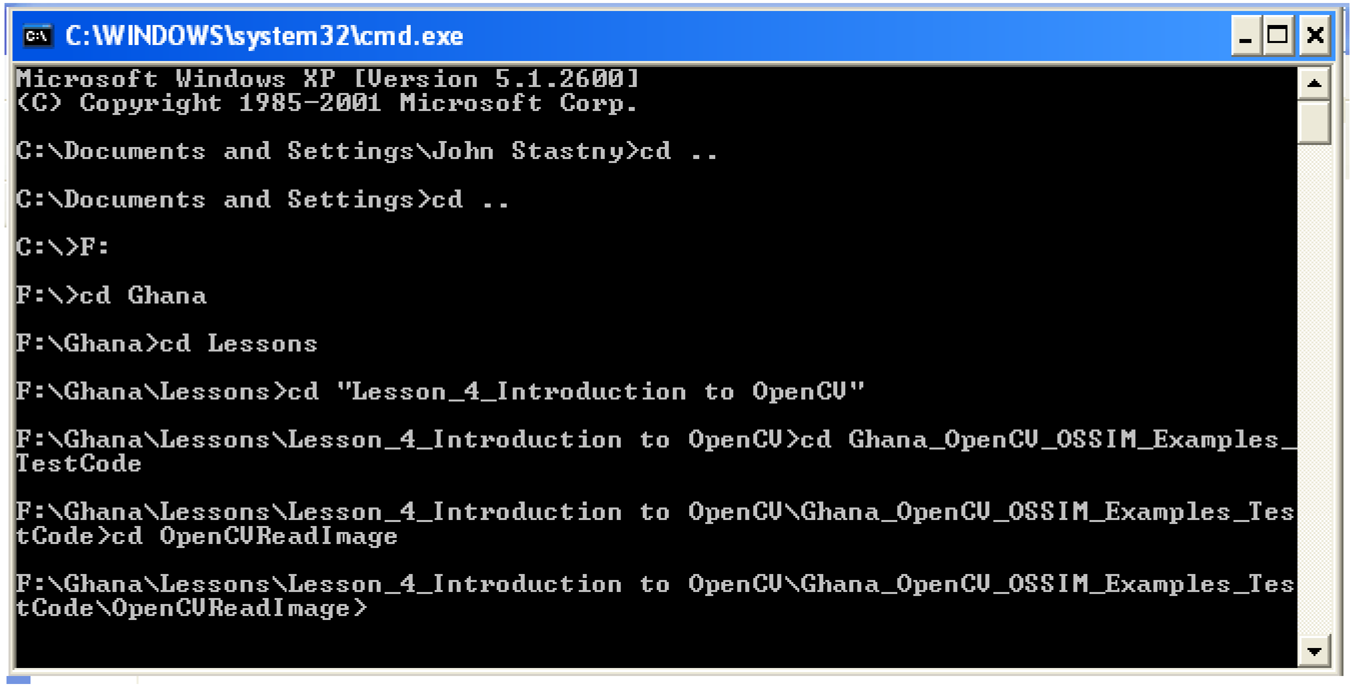
Open: cmd

C:\WINDOWS\system32\cmd.exe

```
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\John Stastny>cd ..
```

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.

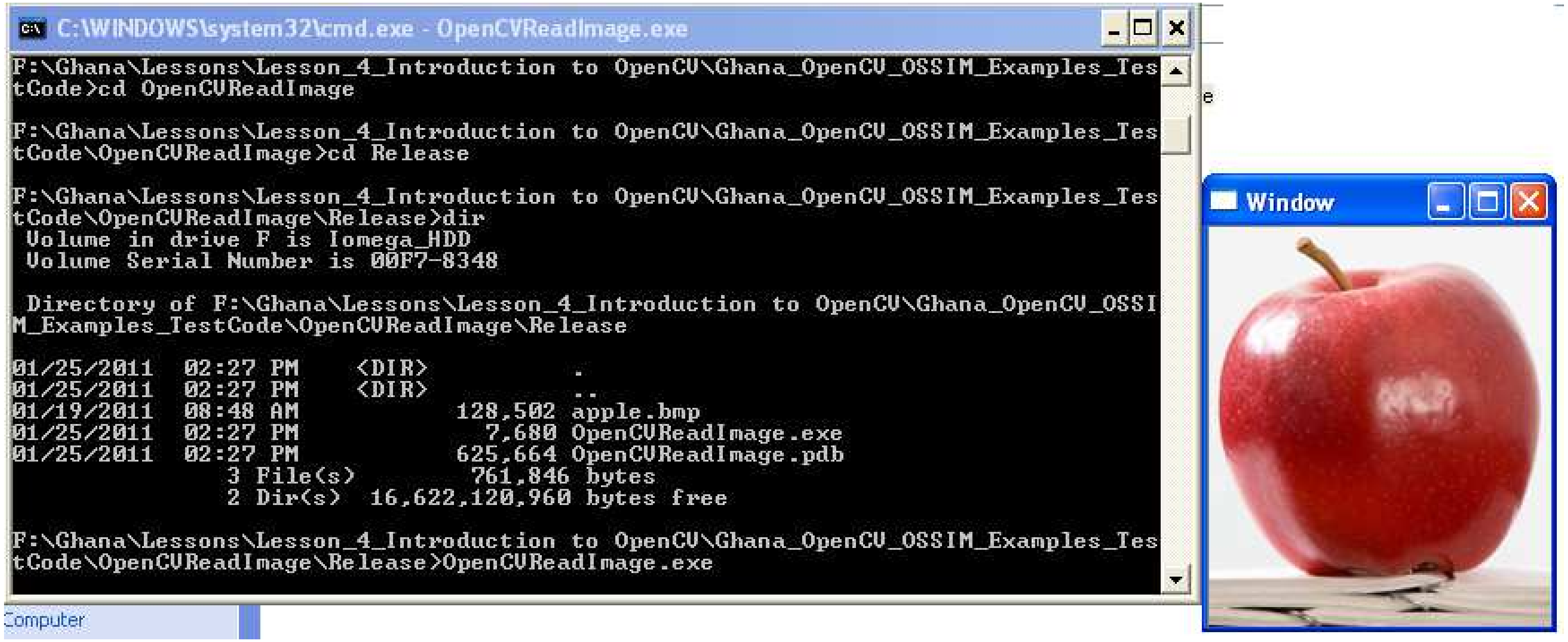


```
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\John Stastny>cd ..

C:\Documents and Settings>cd ..

C:\>F:

F:\>cd Ghana

F:\Ghana>cd Lessons

F:\Ghana\Lessons>cd "Lesson_4_Introduction to OpenCV"

F:\Ghana\Lessons\Lesson_4_Introduction to OpenCV>cd Ghana_OpenCV_OSSIM_Examples_
TestCode

F:\Ghana\Lessons\Lesson_4_Introduction to OpenCV\Ghana_OpenCV_OSSIM_Examples_Tes
tCode>cd OpenCVReadImage

F:\Ghana\Lessons\Lesson_4_Introduction to OpenCV\Ghana_OpenCV_OSSIM_Examples_Tes
tCode\OpenCVReadImage>
```

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.

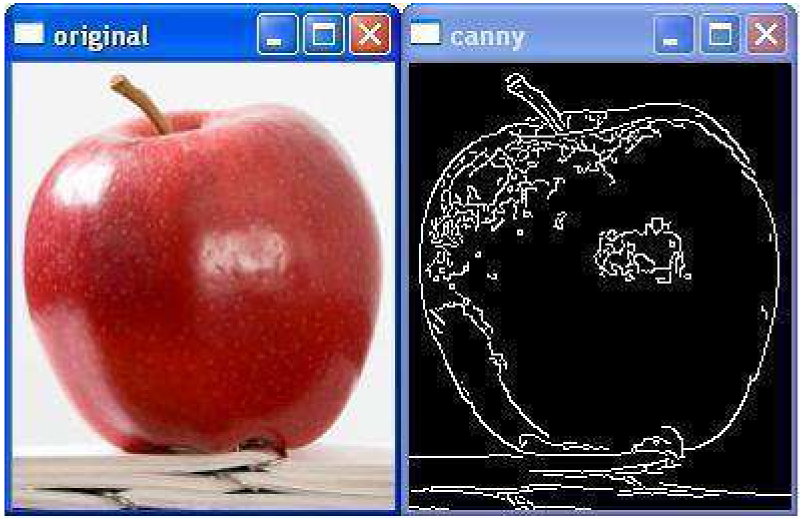


```
F:\Ghana\Lessons\Lesson_4_Introduction to OpenCV\Ghana_OpenCV_OSSIM_Examples_Tes
tCode>cd OpenCVReadImage

F:\Ghana\Lessons\Lesson_4_Introduction to OpenCV\Ghana_OpenCV_OSSIM_Examples_Tes
tCode\OpenCVReadImage>cd Release

F:\Ghana\Lessons\Lesson_4_Introduction to OpenCV\Ghana_OpenCV_OSSIM_Examples_Tes
tCode\OpenCVReadImage\Release>dir
 Volume in drive F is Iomega_HDD
 Volume Serial Number is 00F7-8348

 Directory of F:\Ghana\Lessons\Lesson_4_Introduction to OpenCV\Ghana_OpenCV_OSSI
M_Examples_TestCode\OpenCVReadImage\Release

01/25/2011  02:27 PM    <DIR>          .
01/25/2011  02:27 PM    <DIR>          ..
01/19/2011  08:48 AM           128,502 apple.bmp
01/25/2011  02:27 PM             7,680 OpenCVReadImage.exe
01/25/2011  02:27 PM           625,664 OpenCVReadImage.pdb
               3 File(s)        761,846 bytes
               2 Dir(s)  16,622,120,960 bytes free

F:\Ghana\Lessons\Lesson_4_Introduction to OpenCV\Ghana_OpenCV_OSSIM_Examples_Tes
tCode\OpenCVReadImage\Release>OpenCVReadImage.exe
```

Simple Operations- Edge Detection, Thresholding

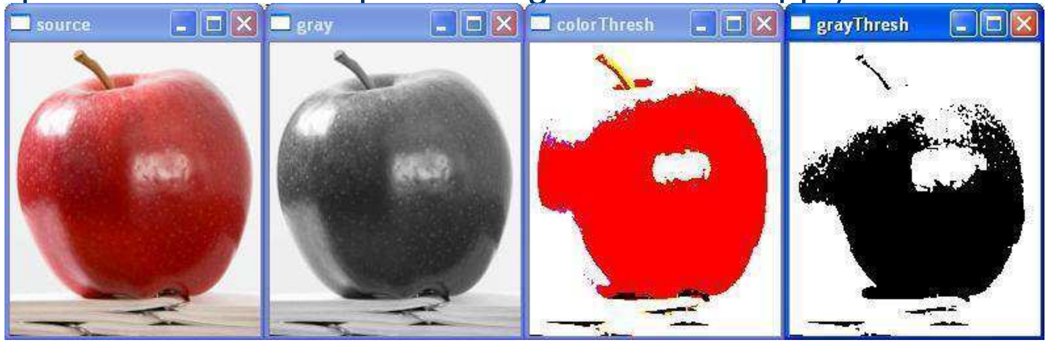
- Canny Edge Detection (..\03_OpenCV_and_blobs\examples\OpenCV_examples
  - \OpenCVEdgeDetect\OpenCVEdgeDetect.cpp)

```
...

// 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

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.
```

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;
}
```

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.
```