

# Emgu CV

## with C# and Visual Studio (1)



CJ Chung  
Computer Science

# What is Emgu CV?

- A cross platform .Net wrapper to the OpenCV image processing library.
- Allowing OpenCV functions to be called from .NET compatible languages such as C#, VB, VC++, etc.
- The wrapper can run on Windows, Linux, Mac OS X, iPhone, iPad and Android devices.

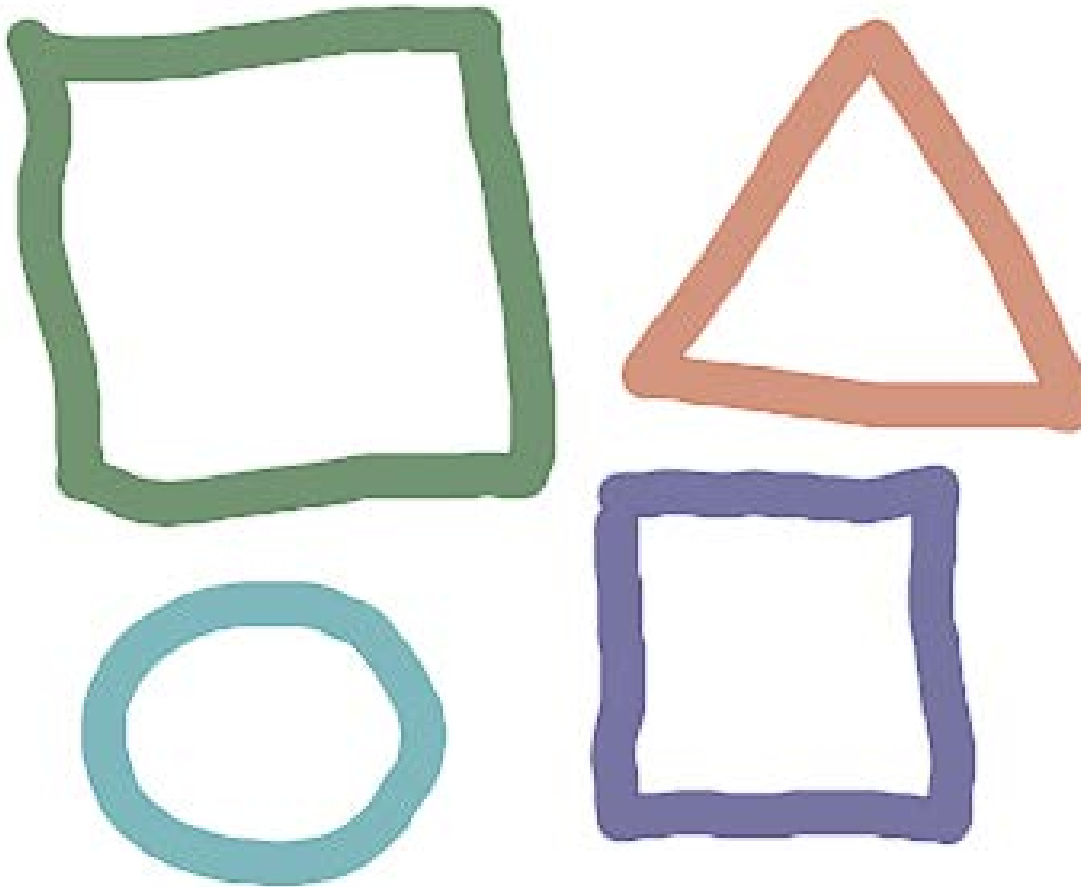
# What is Open CV?



- Open Source Computer Vision Library
- A library of functions mainly aimed at real-time computer vision / image processing
- Developed by Intel, and now supported by Willow Garage and Itseez
- Free for use under the open source BSD license
- The library is cross-platform
- It will use proprietary optimized routines to accelerate itself for Intel processors

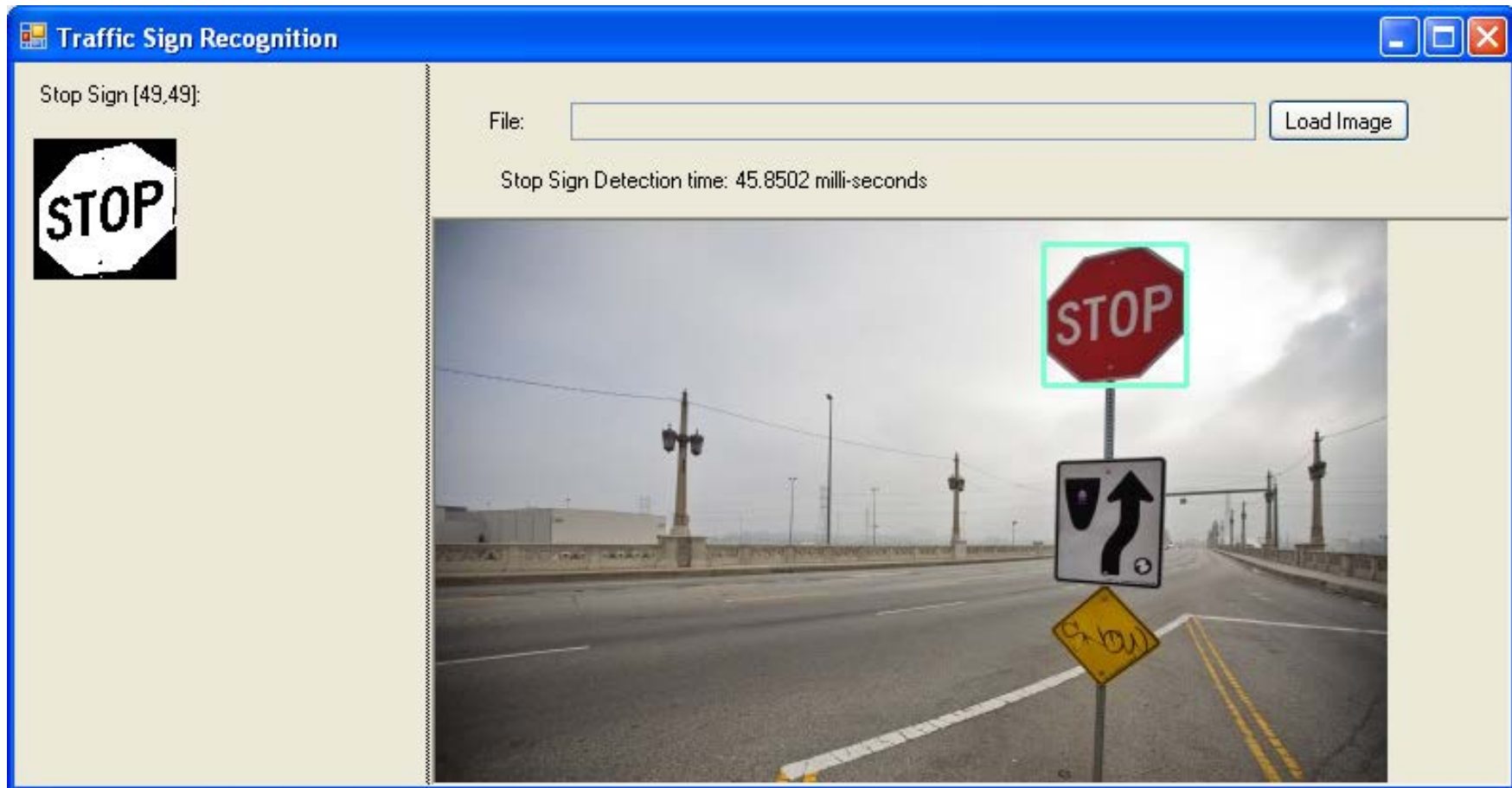
# What can we do? – Sample project

Shape detection



# What can we do? – Sample project

## Stop sign detection



# What can we do? – Sample projects

## Face detection

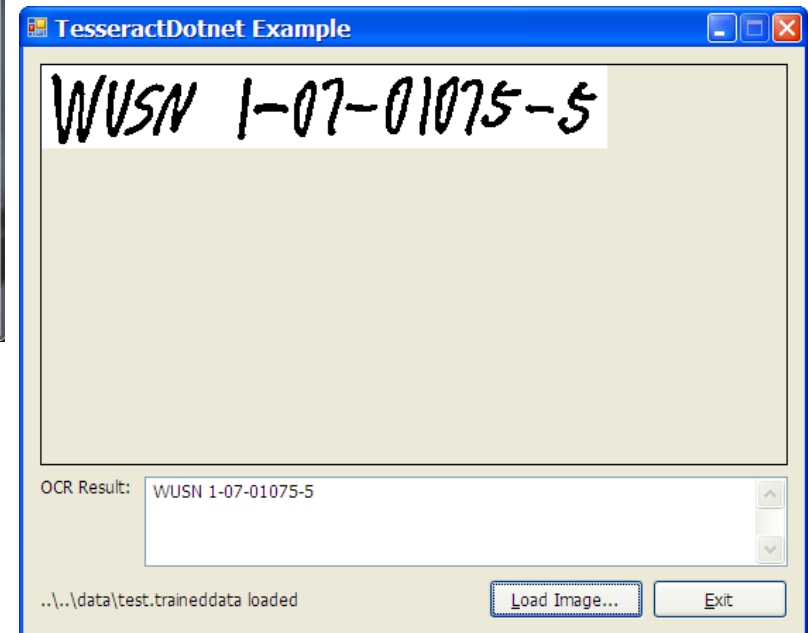
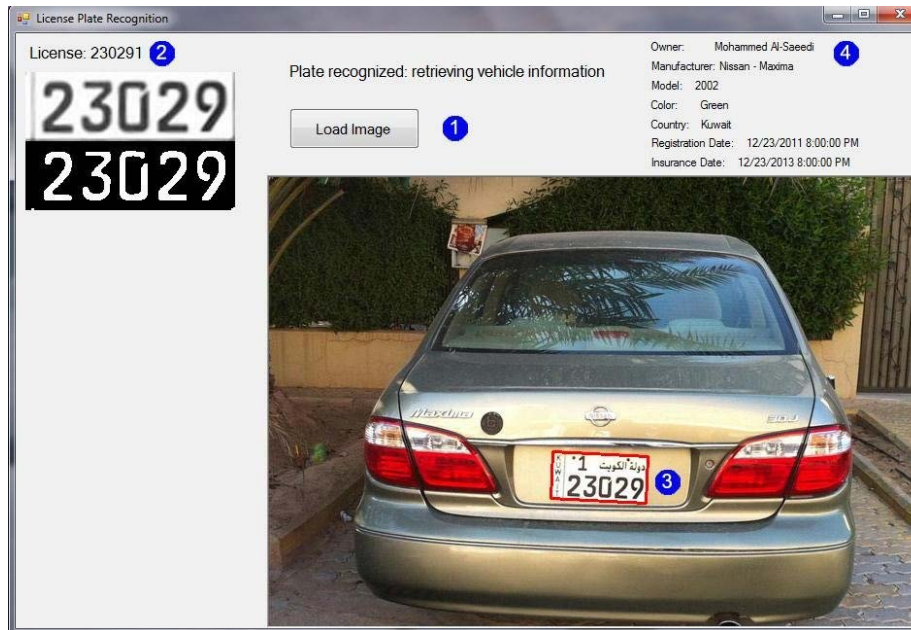


## Pedestrian detection



# What can we do? – Sample projects:


## License Plate Recognition (LPR) Optical Character Recognition (OCR)



# HOW TO SETUP EMGU CV








# Downloading Emgu

- Download Emgu Version **2.4.9-beta** from the following link  [libemgucv-windows-universal-cuda-2.9.0.1922-beta](http://sourceforge.net/projects/emgucv-windows-universal-cuda-2.9.0.1922-beta)
- <http://sourceforge.net/projects/emgucv/files/emgucv>
- Choose the correct version for the your OS

Looking for the latest version? [Download libemgucv-windows-universal-cuda-2.9.0.1922-beta.exe \(214.0 MB\)](#)

[Home](#) / [emgucv](#) / 2.4.9-beta

Name ↕	Modified ↕	Size ↕	Downloads / Week ↕	
↑ Parent folder				
<a href="#">libemgucv-ios-2.4.9.1933-r3-simulat...</a>	2014-04-15	22.3 MB	5	 ⓘ
<a href="#">libemgucv-windows-universal-cuda-...</a>	2013-09-26	511.3 MB	55	 ⓘ
<a href="#">libemgucv-windows-universal-cuda-...</a>	2013-09-26	214.0 MB	1,083	 ⓘ
<a href="#">demo.emgu.cv.Android-2.4.9.1910-b...</a>	2013-08-31	50.6 MB	4	 ⓘ
<a href="#">libemgucv-android-x86-simulator-onl...</a>	2013-08-31	24.1 MB	3	 ⓘ
Totals: 5 Items		822.2 MB	1,150	

Same!

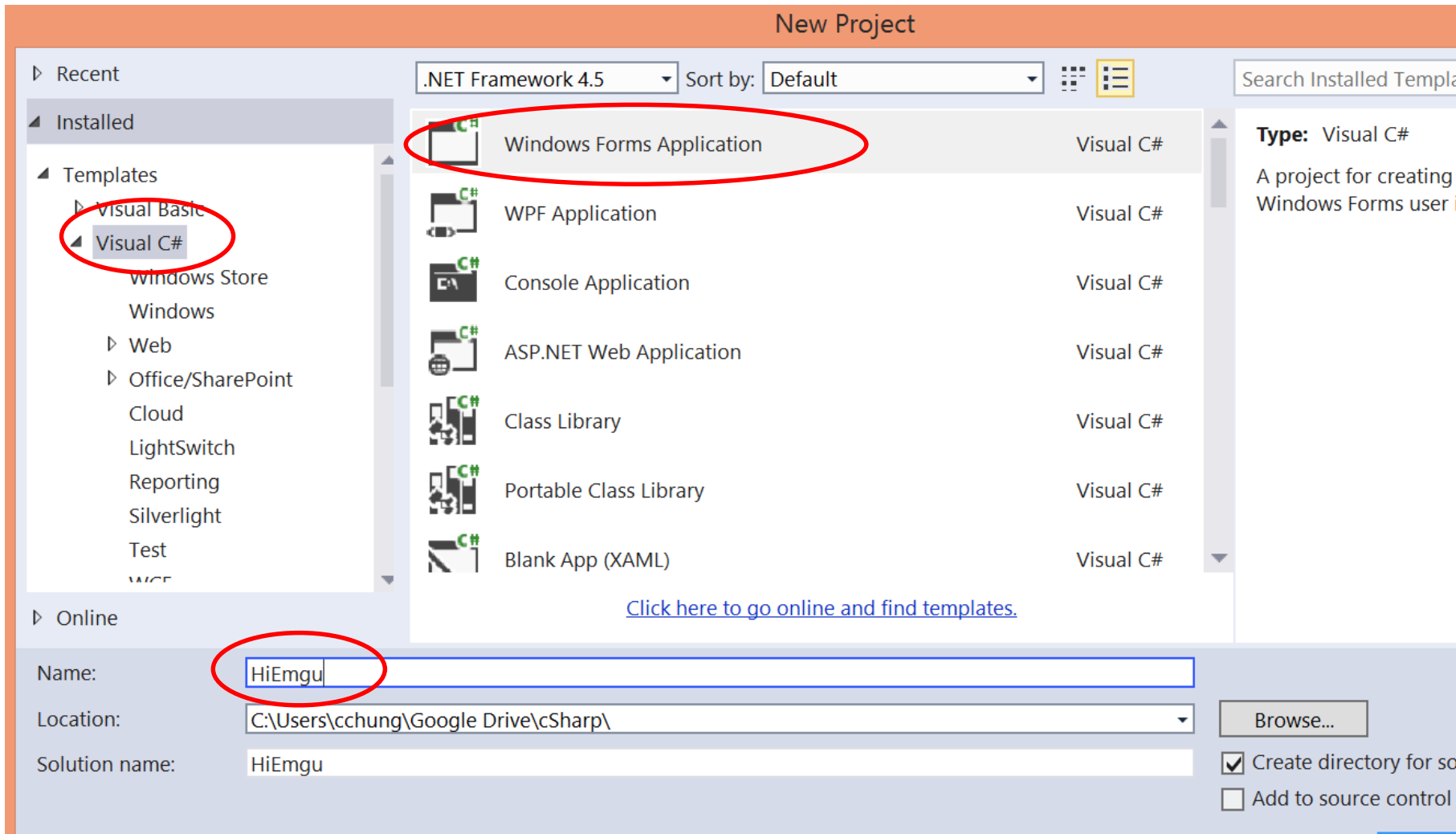
# Installing Emgu

- Once downloaded, run the `libemgucv-windows-universal-cuda-2.9.0.1922-beta.exe`
- If you choose default options, it will be installed under `C:\Emgu\`
- **Note:** This instruction is for LTU laptop (Lenovo Yoga) with 64 bit OS

# Emgu Setup in VS (Step 1)

First, select “Visual C#”

Create a new Windows Forms Application, “HiEmgu”

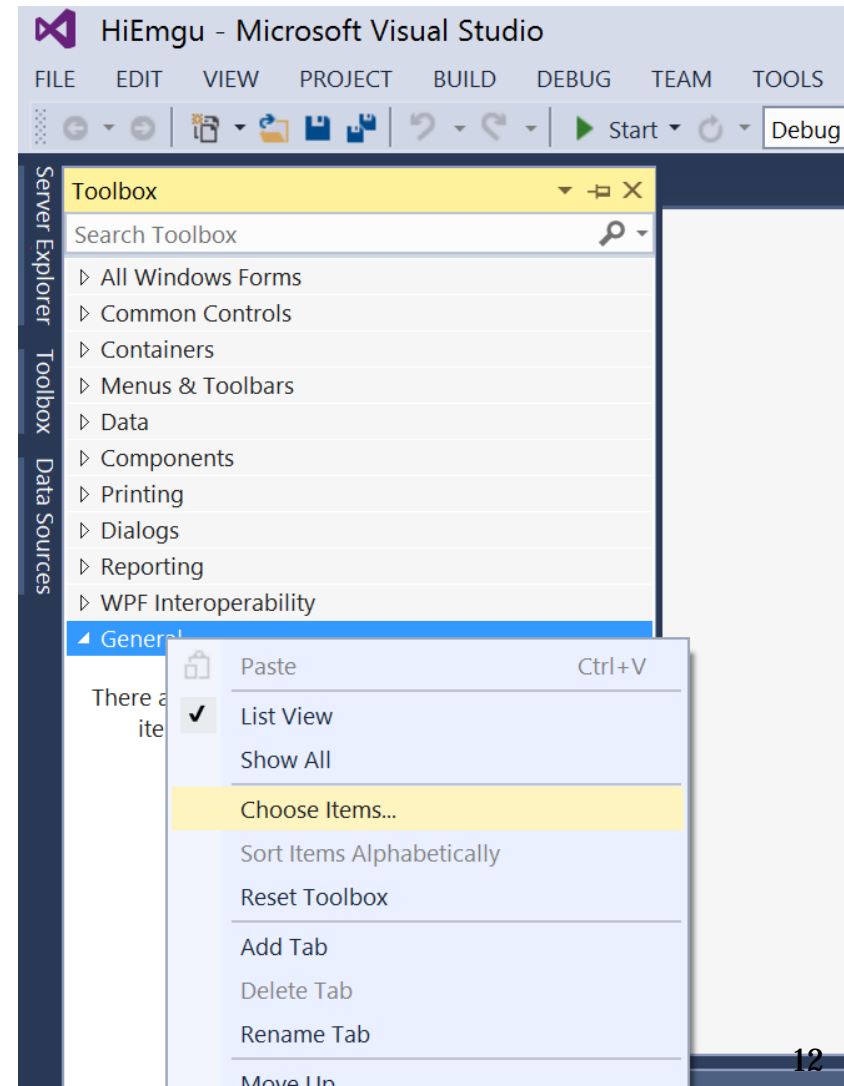


# Emgu Setup (**Step 2**) - Adding ImageBox Control in Visual Studio (i), **Once**

[http://www.emgu.com/wiki/index.php/Add\\_ImageBox\\_Control](http://www.emgu.com/wiki/index.php/Add_ImageBox_Control)

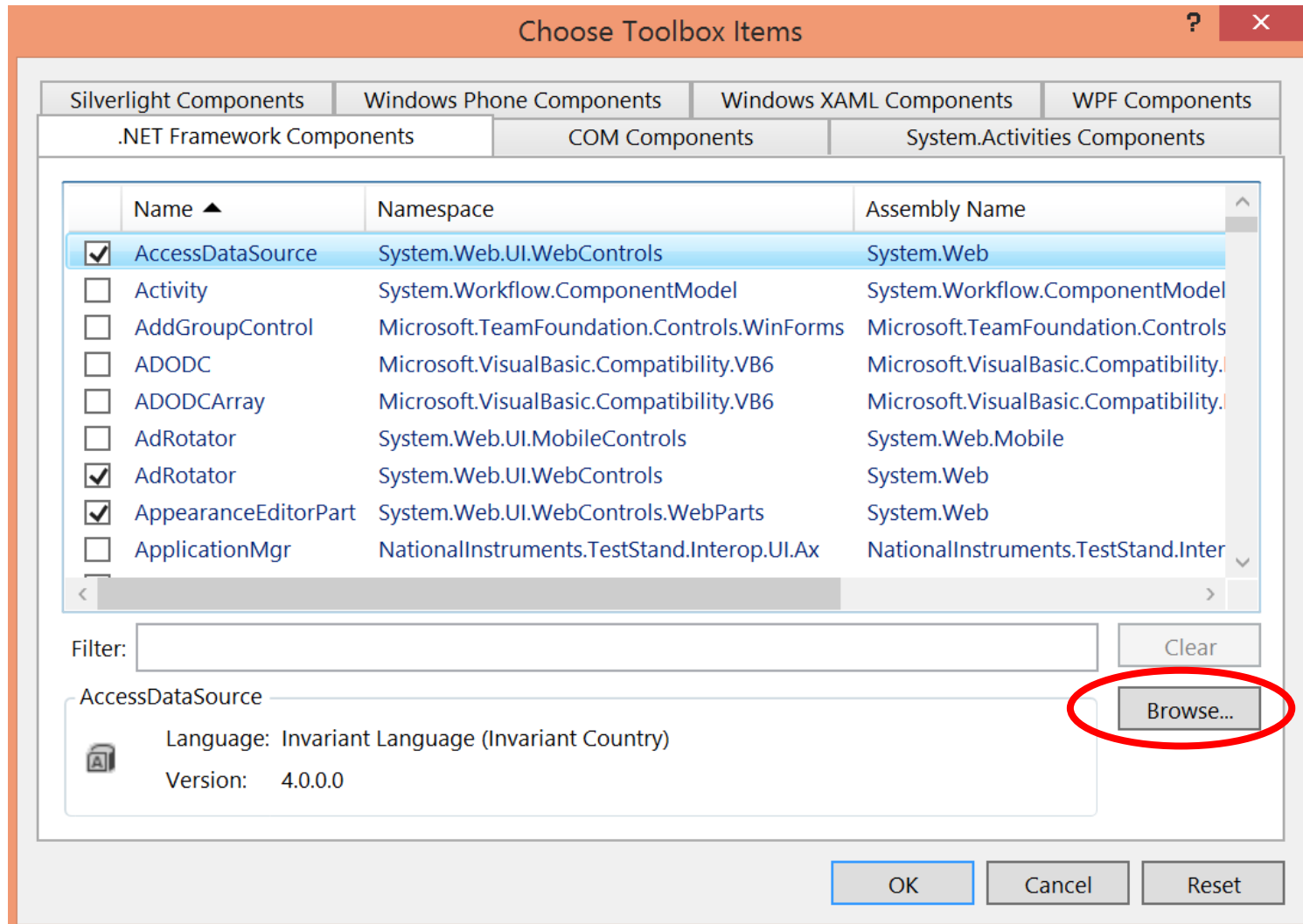
In Designer view / **Toolbox**, right click in the empty space of 'General' column. This should pop up a selection menu, where 'Choose Items' selection is available.

*In case you need to re-importing ImageBox control, right click on “General” and click on “Reset Toolbox”*



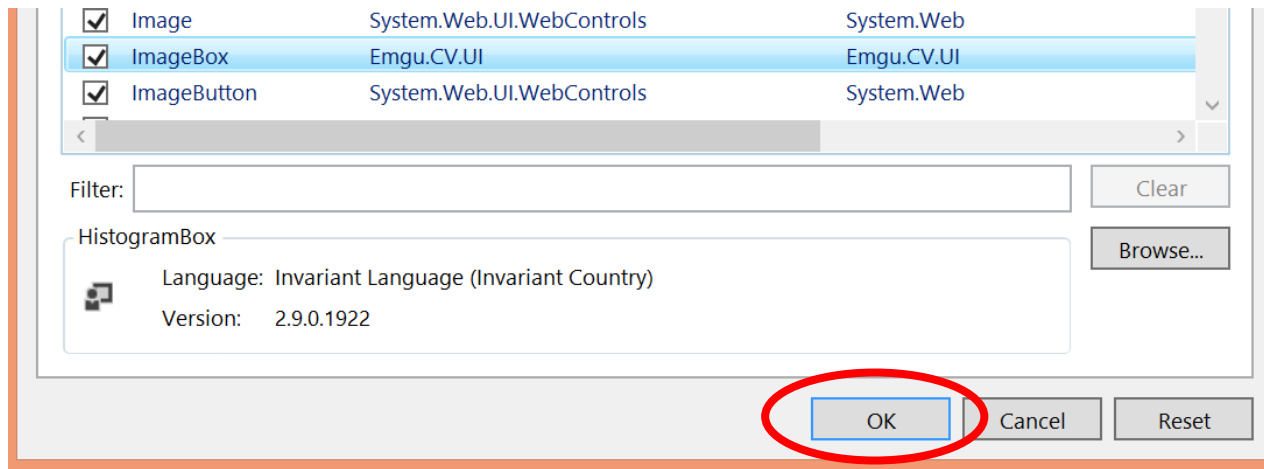
# Emgu Setup (Step 2) - Adding ImageBox Control in Visual Studio (ii) ), **Once**

Click on the 'Browse..' button.



# Emgu Setup (Step 2) - Adding ImageBox Control in Visual Studio (iii) ), **Once**

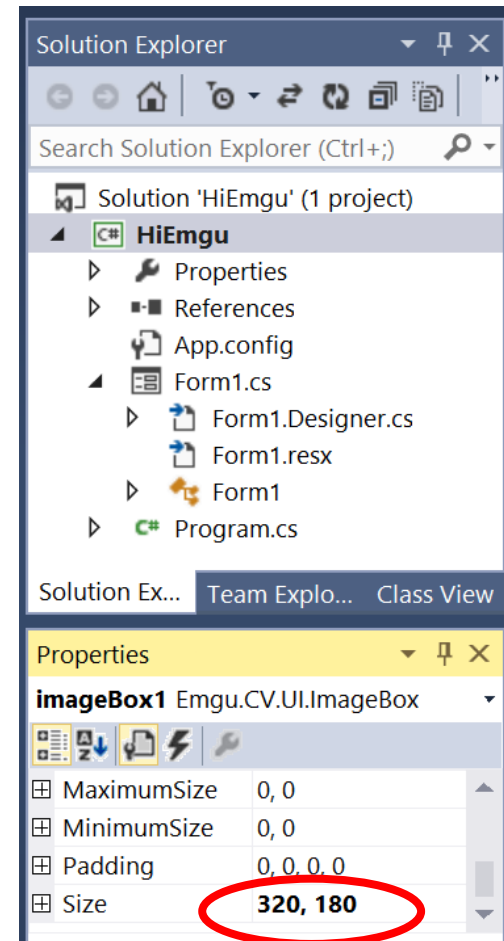
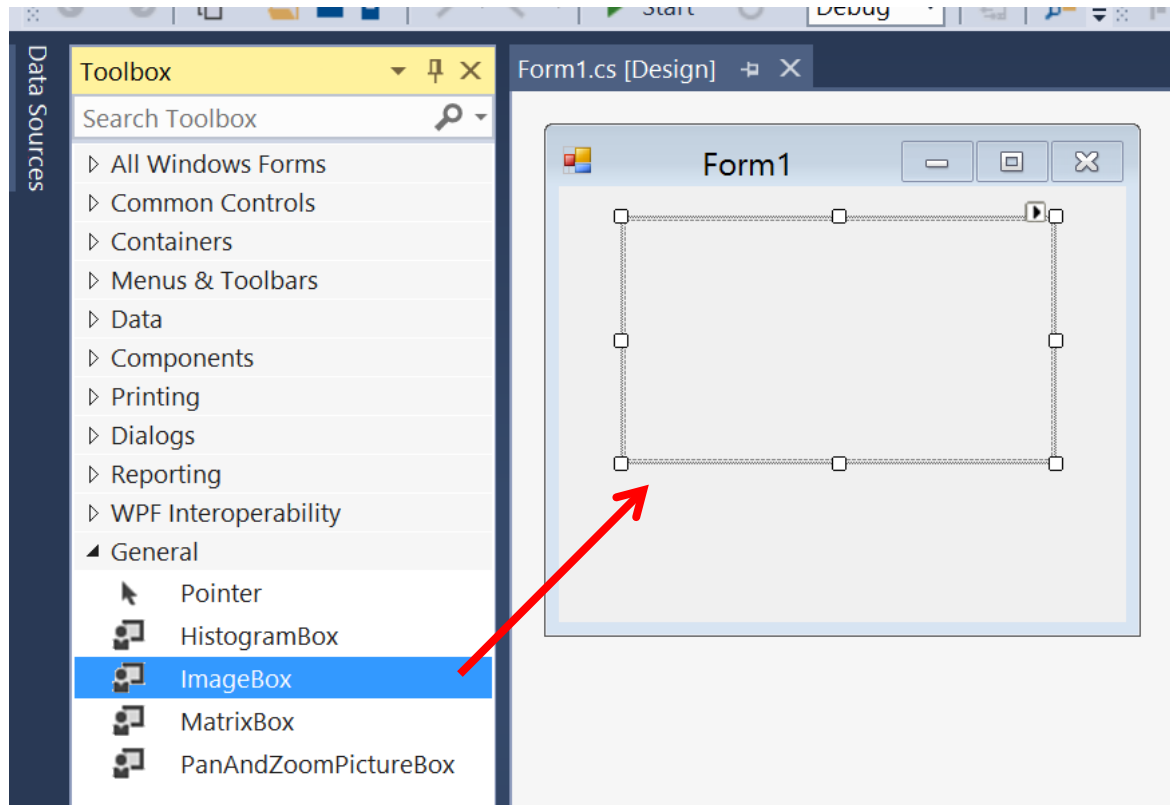
- Select **Emgu.CV.UI.dll** file from: **C:\Emgu\emgucv-windows-universal-cuda 2.9.0.1922\bin**
- Click the 'Open' button
- Now you should notice the **ImageBox** control has been added to the 'Choose Toolbox Items' dialog. Click 'Ok'.



*If you get an error message here, re-launch Visual Studio*

# Emgu Setup in VS (Step 3)

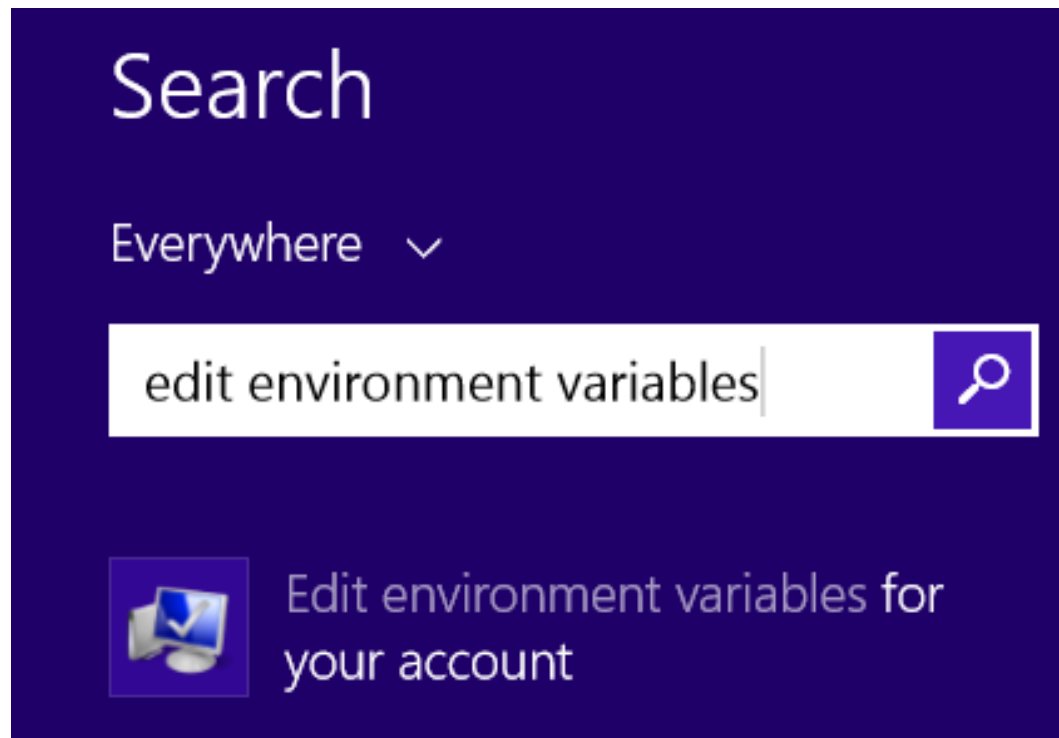
Drag and drop an **ImageBox** component on to the current Windows Form



Check the default aspect ratio of your camera. Yoga uses **16:9**, (320, 180)

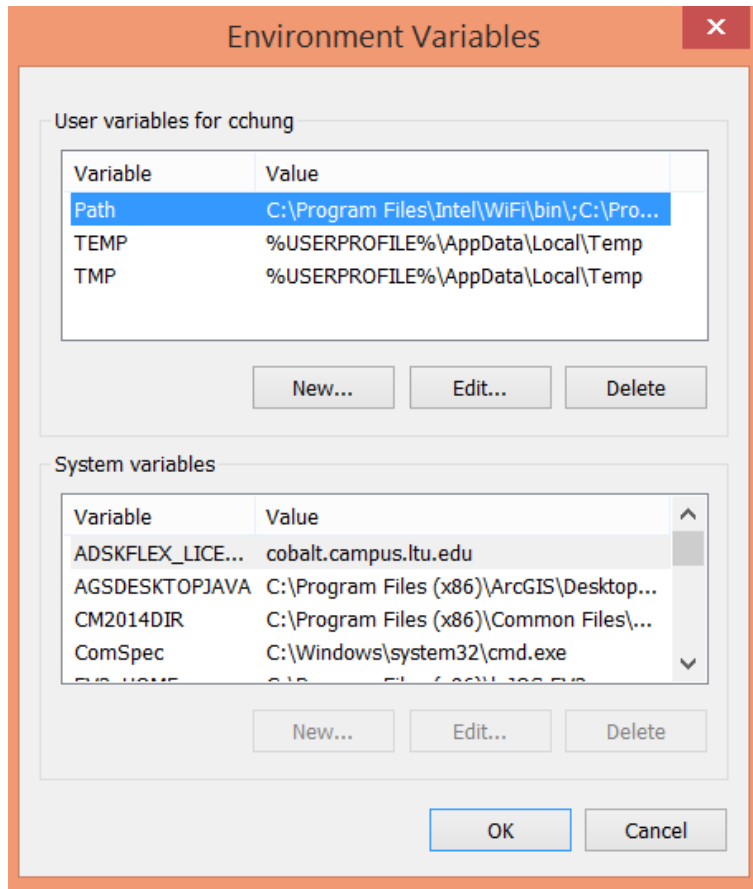
# Setup path for the Project \*.DLL References (**Step 4, i**), **Once**

- Search “Edit environment variables” and click on





# Setup path for the Project \*.DLL References (Step 4, ii), **Once**



Click on Edit... button

Carefully add a semicolon (;) at the end of the Path variable.

Then append the path to the Emgu CV ..\bin\x64 folder after the semicolon.

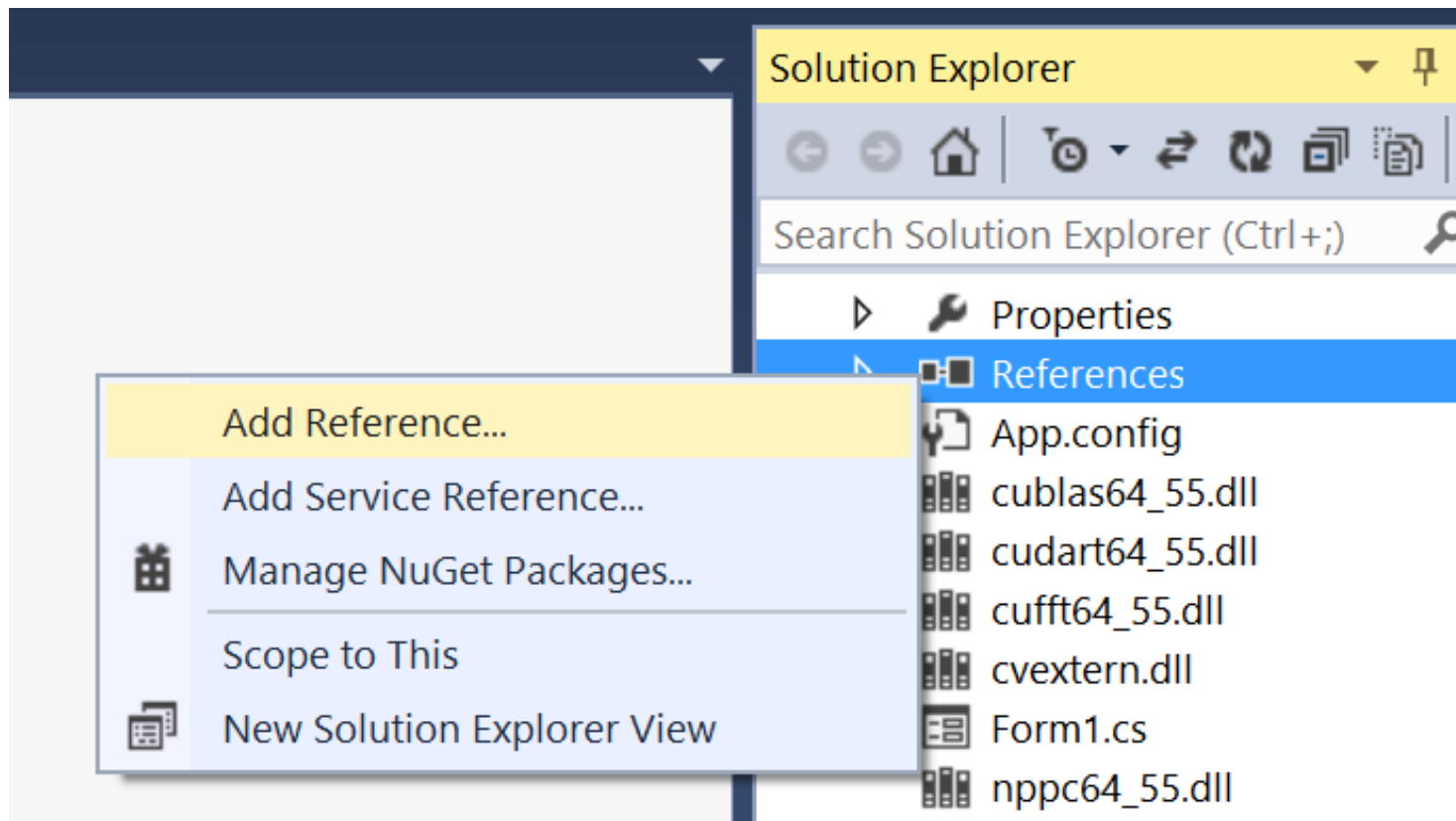
**Warning: Keep the current!!!!**

For example:

;C:\Emgu\emgucv-windows-universal-cuda 2.9.0.1922\bin\x64

# Emgu Setup (Step 5) project references (i)

- Reference the three .dll files in the project by right clicking the 'References' folder and selecting 'Add Reference'

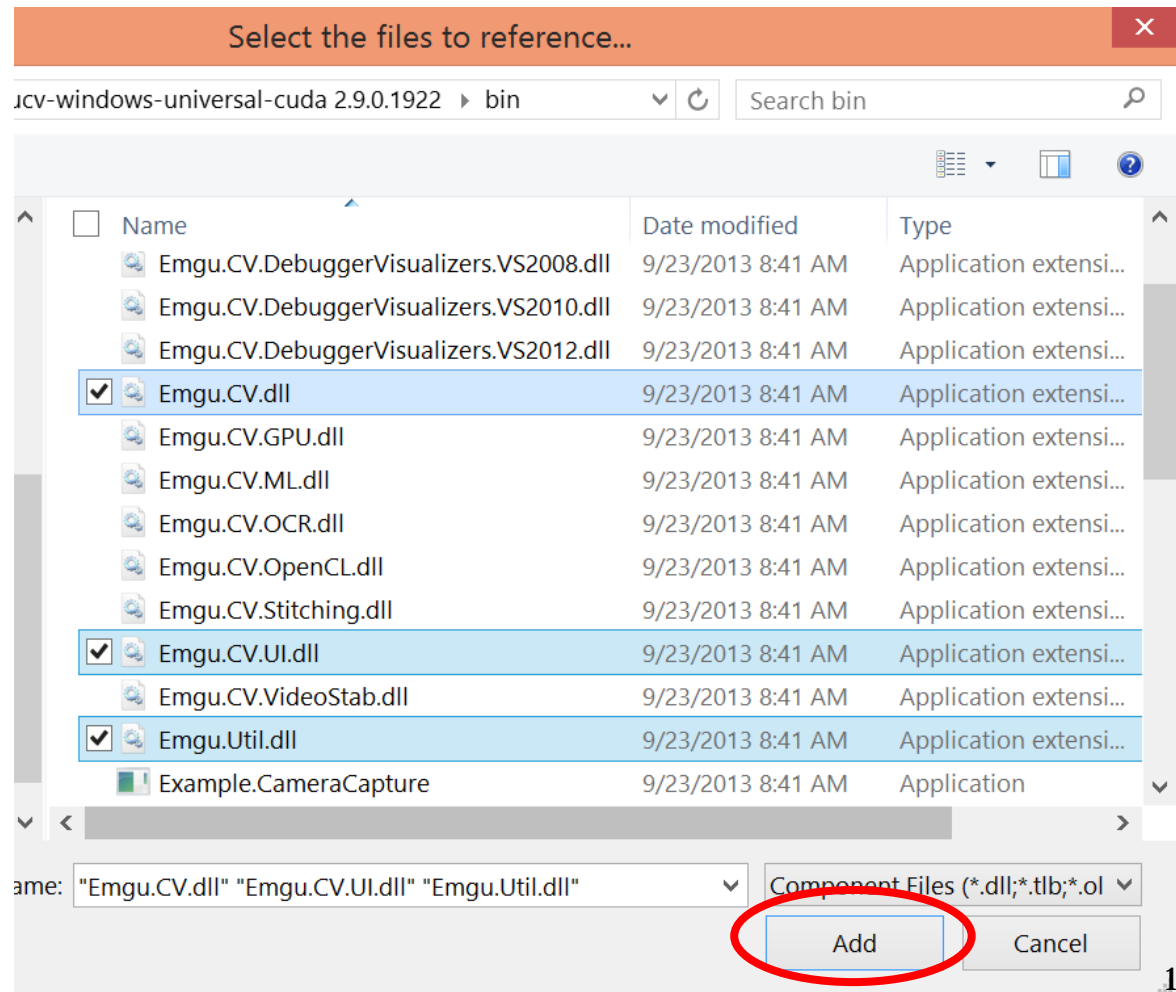


# Emgu Setup (Step: 5) project references (ii)

Using the 'Browse' tab, navigate to C:\Emgu\emgucv-windows-universal-cuda 2.9.0.1922\bin and select the following files:

- Emgu.CV
- Emgu.CV.UI
- Emgu.Util

Then click on  
Add -> OK  
button.



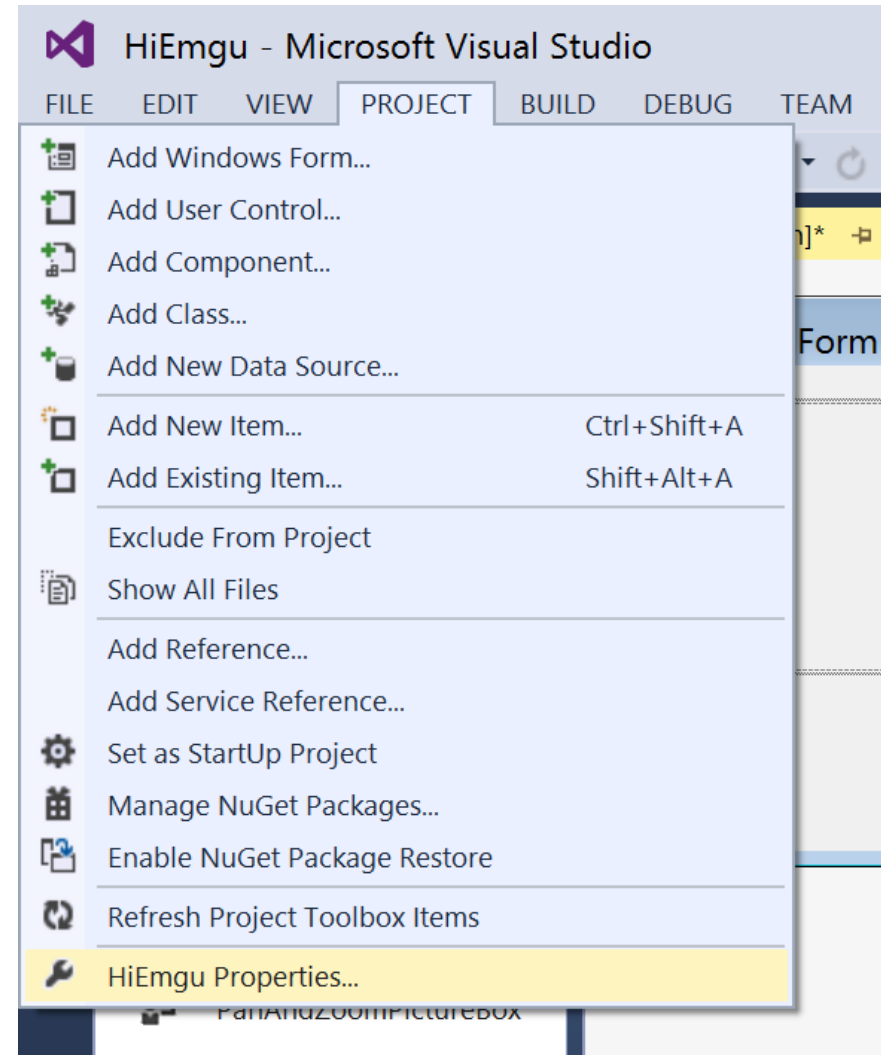
# Emgu Setup in VS (Step: 6)

## platform target (i)

The final step is to change the 'Platform target' to **x64**, if you are on a 64 bit OS

From the 'Project' menu select

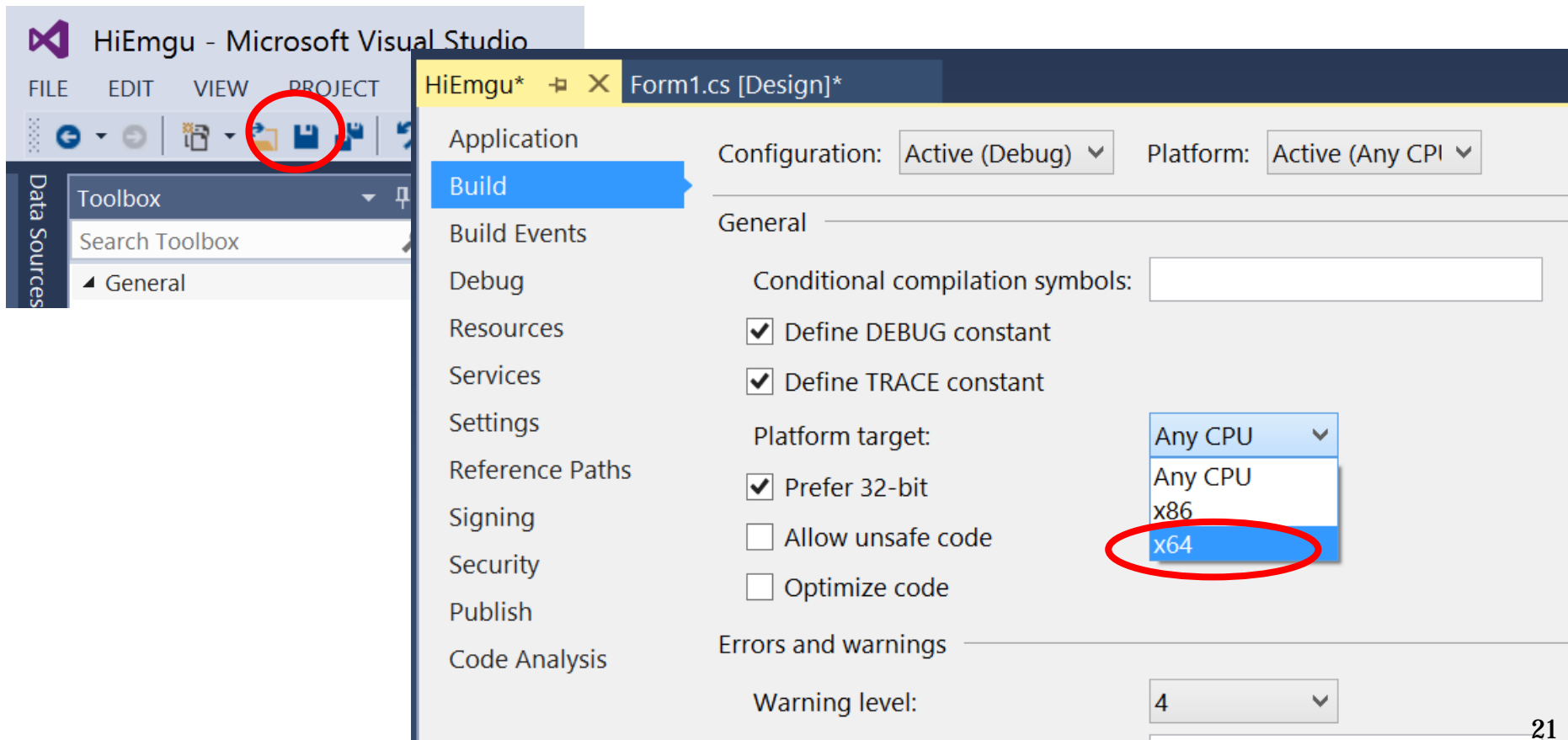
*ProjectName*  
Properties...



# Emgu Setup in VS (Step: 6)

## platform target (ii)

Select **x64** for the 'Platform target' in the Build tab, then click on Save.



**(STEP 7)**

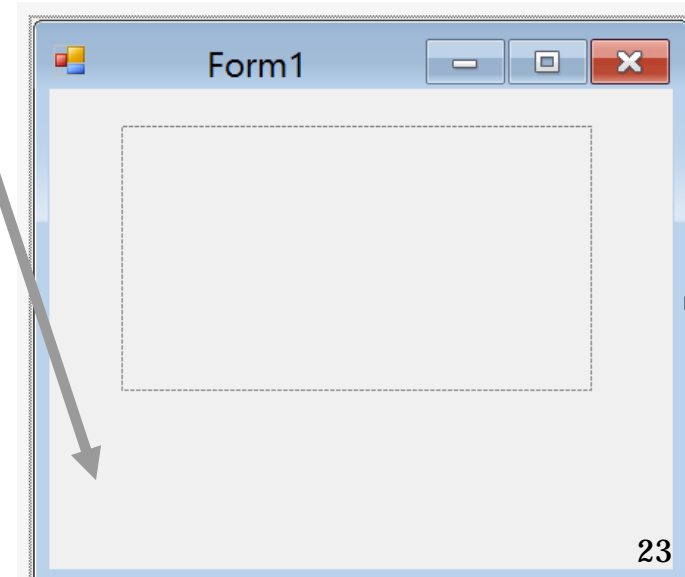
***It is highly recommended to  
re-launch Visual Studio now  
then***

**COMPLETE THE  
FIRST EMGU CV PROGRAM**

# Hello EmguCV with C# (1/6)

- Once the Emgu references have been setup in VS, you can begin writing your C# code
- First double click outside ImageBox area to get Form1.cs
- Then include the referenced Emgu files in your Forms.cs file:

```
using Emgu.CV;  
using Emgu.Util;  
using Emgu.CV.Structure;
```



# Hello EmguCV with C# (2/6)

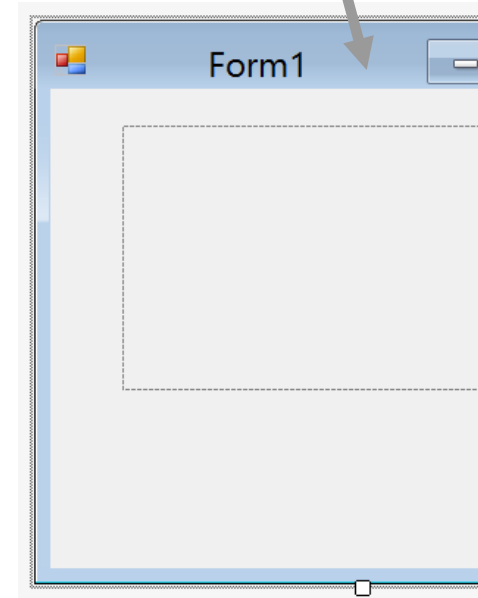
- Next, you will need to get an instance of the Capture class that allow you to capture images from your input device, inside the Form1 class:
  - `Capture _capture = null;`
- Add the Capture instance as a Class variable since we will use it in several of the Class methods.



# Hello EmguCV with C# (3/6)

- Click on “Form Bar” to create Form1\_Load method
- In the form’s Load method you will have to:
  1. Create a new instance of the Capture class
  2. Create an event handler to receive image frames from the input device
  3. Start the Capture instance

```
private void Form1_Load(object sender, EventArgs e)
{
    _capture = new Capture();
    _capture.ImageGrabbed += Display_Captured;
                          // Event Handler
    _capture.Start();
}
```



# Hello EmguCV with C# (4/6)

- In the Capture event handler you need to:
  1. Retrieve the captured image frame
  2. Resize it to fit your imageBox element
  3. Display the image in the imageBox

```
void Display_Captured(object sender, EventArgs e)
{
    Image<Bgr, Byte> frame =
        _capture.RetrieveBgrFrame().Resize(imageBox1.Width,
        imageBox1.Height, Emgu.CV.CvEnum.INTER.CV_INTER_LINEAR);
    imageBox1.Image = frame;
}
```

# Hello EmguCV with C# (5/6)

- As a last step, you need to override the `onClosing` method to stop the `Capture` instance. ***This will prevent an exception from being thrown when the program is closed.***

```
protected override void OnClosing(CancelEventArgs e)
{
    _capture.Stop();
    base.OnClosing(e);
}
```

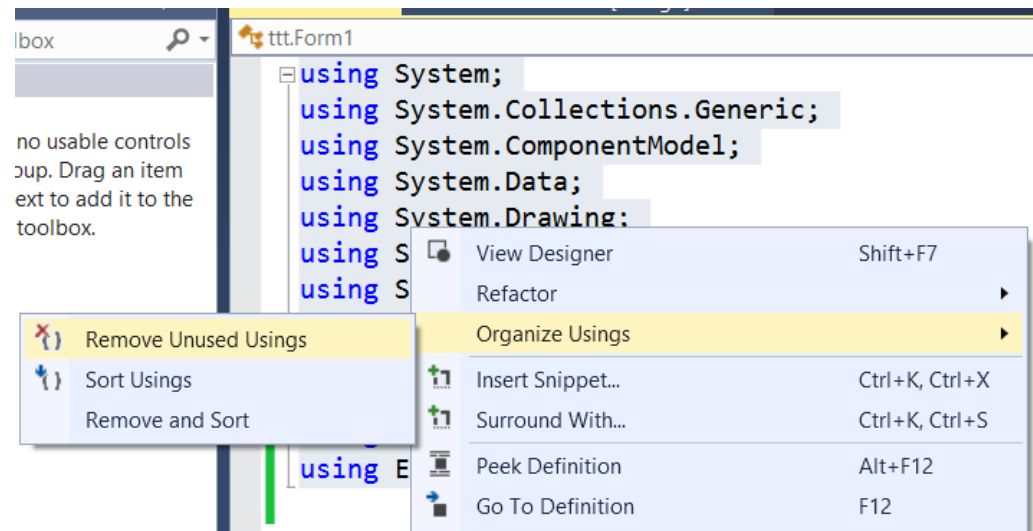
*This step may be omitted in the 2.4.9beta version*

# Hello EmguCV with C# (6/6)

## Remove Unused Usings

```
using System;  
using System.Collections.Generic;  
using System.ComponentModel;  
using System.Data;  
using System.Drawing;  
using System.Linq;  
using System.Text;  
using System.Threading.Tasks;  
using System.Windows.Forms;  
using Emgu.CV;  
using Emgu.Util;  
using Emgu.CV.Structure;
```

namespace ...



```
using System;
using System.Drawing;
using System.Windows.Forms;
using Emgu.CV;
using Emgu.CV.Structure;
```

```
namespace HiEmgu
```

```
{
    public partial class Form1 : Form
    {
        Capture _capture = null;
        public Form1()
        { InitializeComponent();
        }
    }
}
```

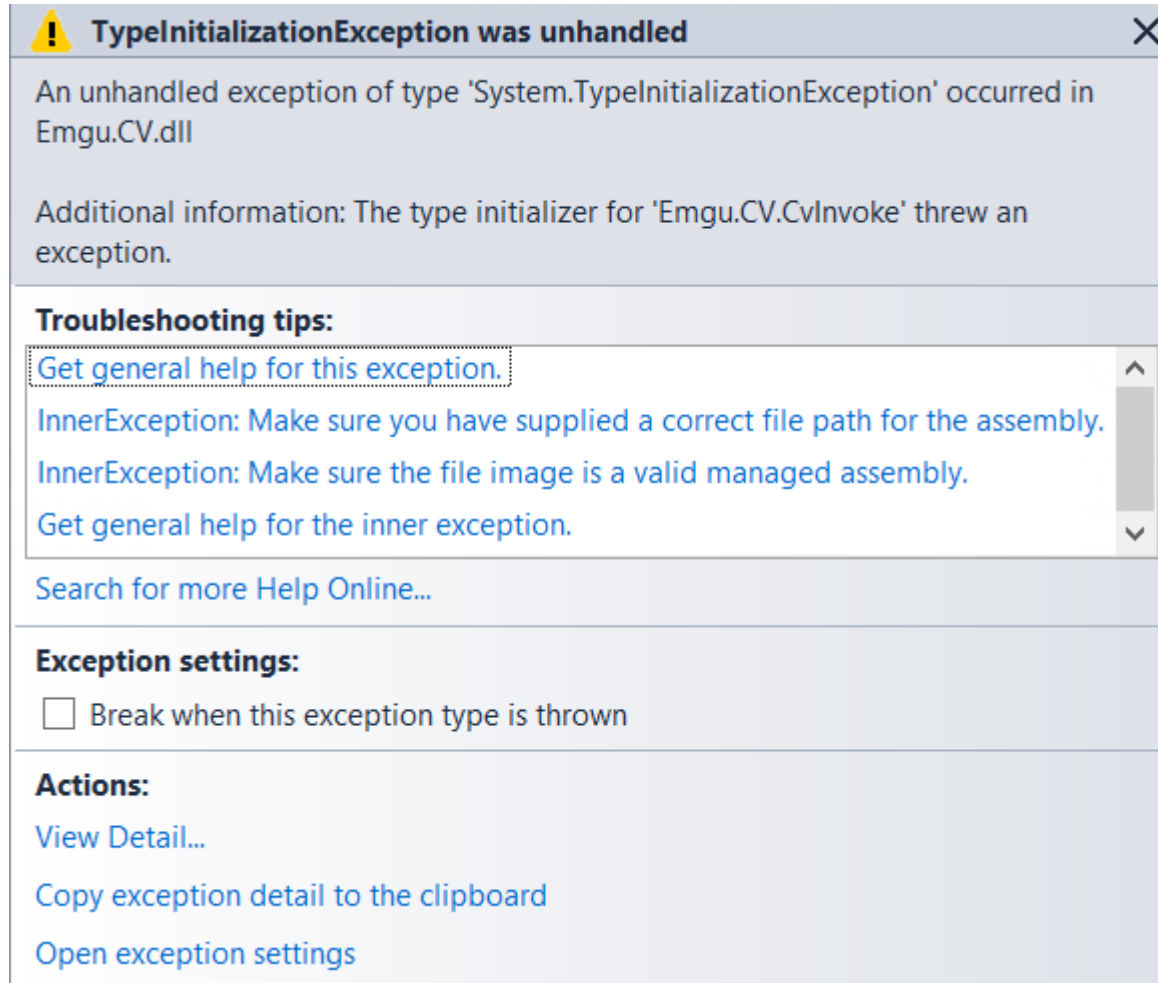
```
private void Form1_Load(object sender, EventArgs e)
{
    _capture = new Capture();
    _capture.ImageGrabbed += Display_Captured; // Grab Event Handler
    _capture.Start();
}
```

```
void Display_Captured(object sender, EventArgs e)
{
    Image<Bgr, Byte> frame = _capture.RetrieveBgrFrame().Resize
        (imageBox1.Width,
         imageBox1.Height, Emgu.CV.CvEnum.INTER.CV_INTER_LINEAR);
    imageBox1.Image = frame;
}
}
```



# **TROUBLE SHOOTING**

# Most common error: **Type Initialization Exception**



x64 selected?

DLLs are  
accessible?

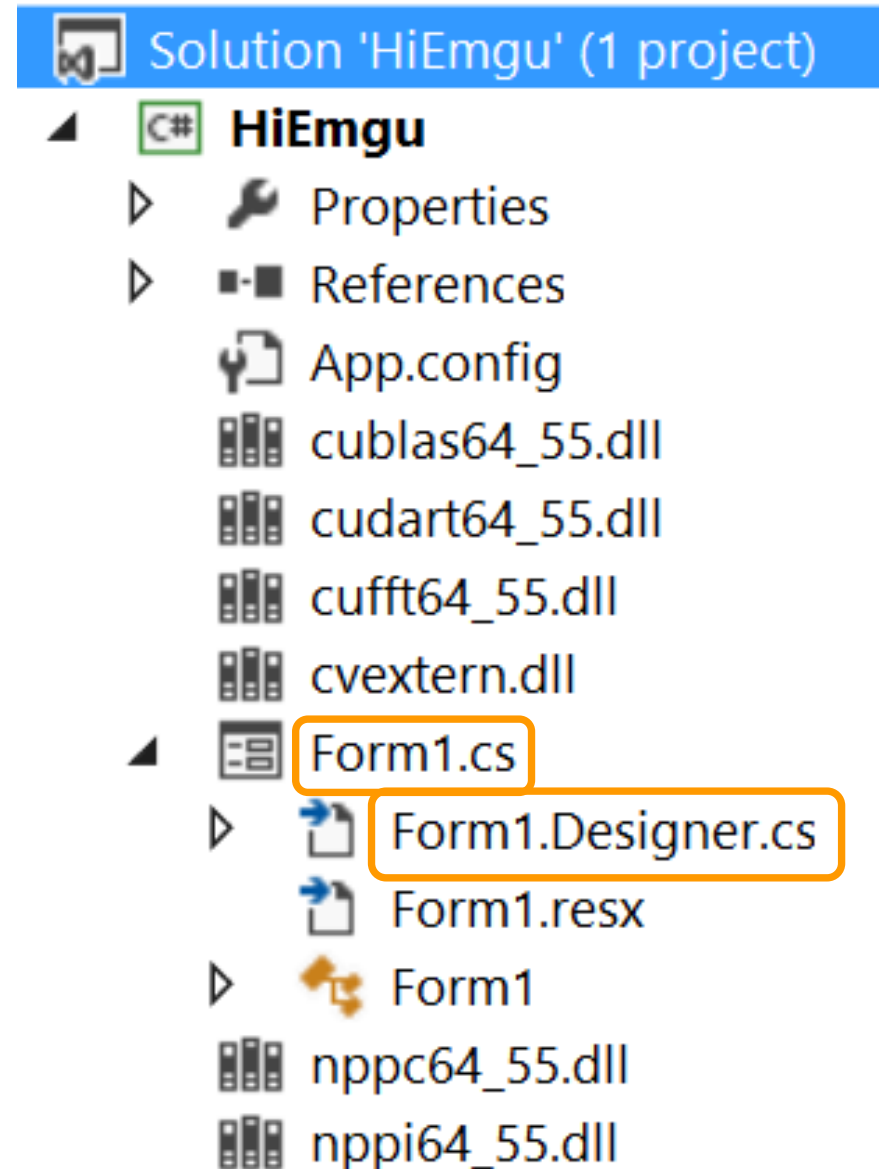
# Trouble Shooting: '**Access Violation**' runtime error

- It may occur for the first run after setup. Re-launch Visual Studio and/or reboot your computer
- Check if the camera is working: built-in webcam ID is 0



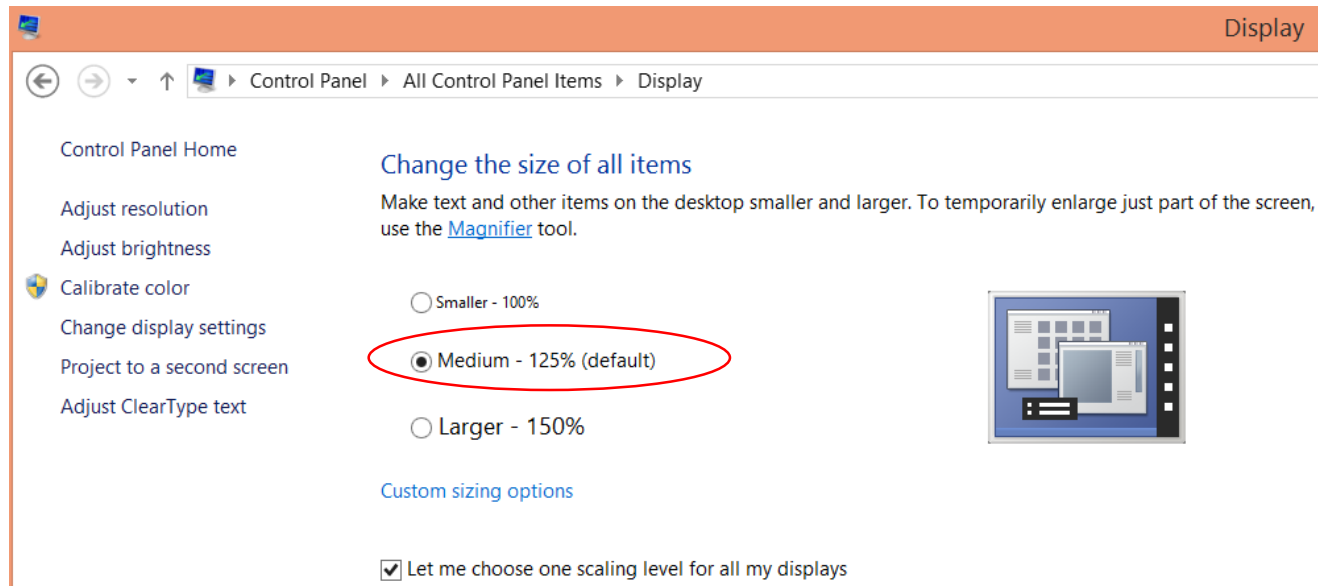
# Troubleshooting

- Did you rename anything?
- Did you delete anything?
- Did you double click the imageBox?
- Check Form1.Designer.cs matches names and the event handlers in Form1.cs
- Are dlls set to “copy always”?  
In case you use alternative way (Step 4)
- ...



# Trouble Shooting: **imageBox** **Width and Height values**

- **Set a breakpoint to check the values**
- If different from the values set on Properties, then change Font size setting in Control Panel - Display to “default”.



# Trouble Shooting: Other tips

- Build -> Clean Solution
- Close Visual Studio and re-launch
- Restart your laptop
- ImageBox is not shown when running → FormBar was not clicked to create **Form1\_Load** method, possibly
- Remember: you cannot just copy and paste whole programs! – recreate project, redesign GUI, re-double click for each control

# Let's recreate the same project using the setup

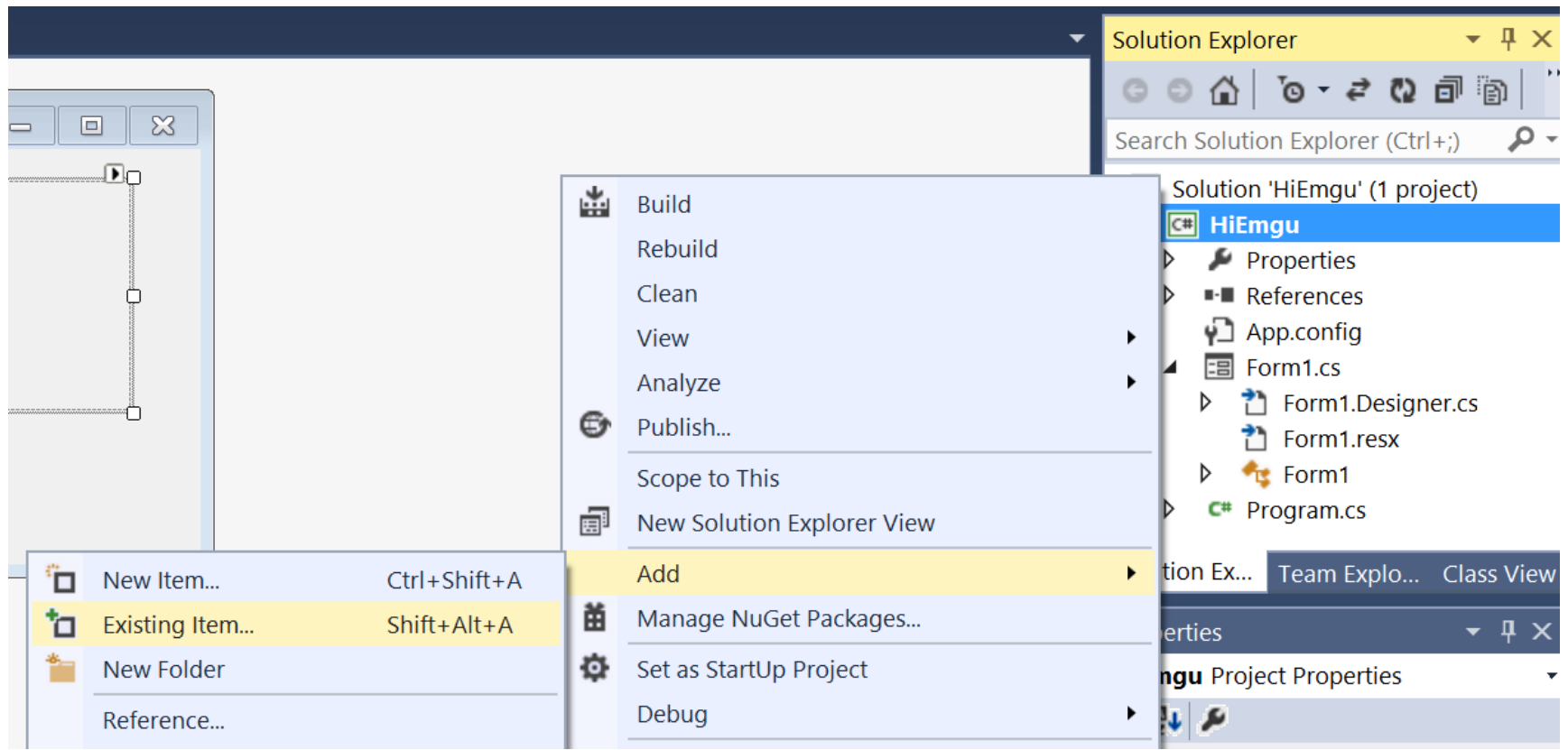
- Step 1: Name Solution/Project
- Step 2 (only once): Adding ImageBox Control in VS
- Step 3: Drag/drop an ImageBox on the Form
- Step 4 (only once): Setup Windows *path* for \*.DLL
- Step 5: Add the three \*.DLL References
- Step 6: Change the 'Platform target' to x64
- Step 7: Complete the code and change the form bar title from Form1 to something else

**ALTERNATIVE METHOD  
FOR STEP (4) TO SETUP**

**JUST FOR FUTURE  
REFERENCE  
PLEASE DO NOT USE**

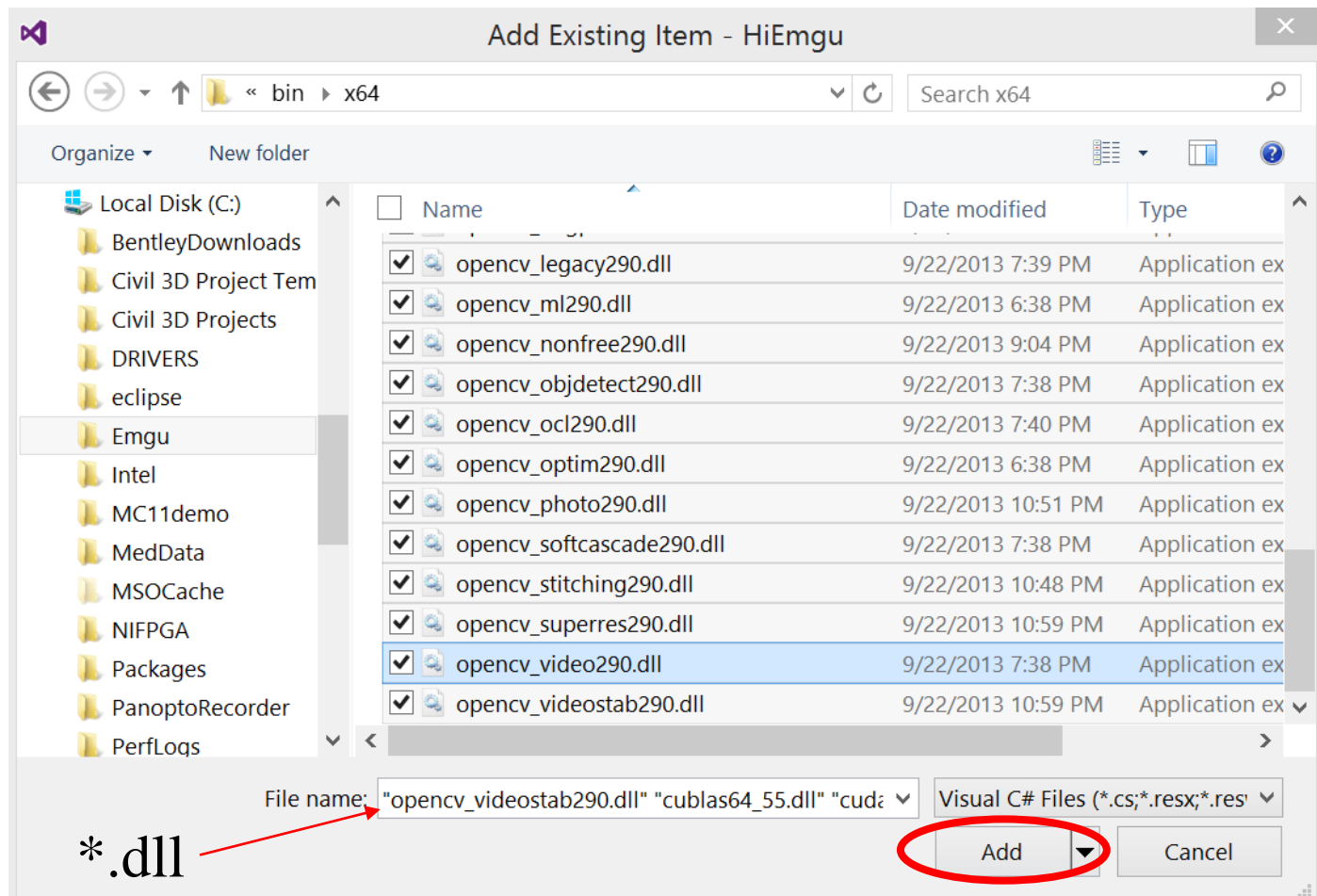
# Alternative way for (Step 4) - \*.DLL References (i)

- To use the Emgu functions contained in multiple .dll files, they need to be referenced in the project
- To reference the .dll files, right click your **Project\_Name** in the Solution Explorer -> Add -> Existing Item...



# Alternative way for (Step 4) - \*.DLL References (ii)

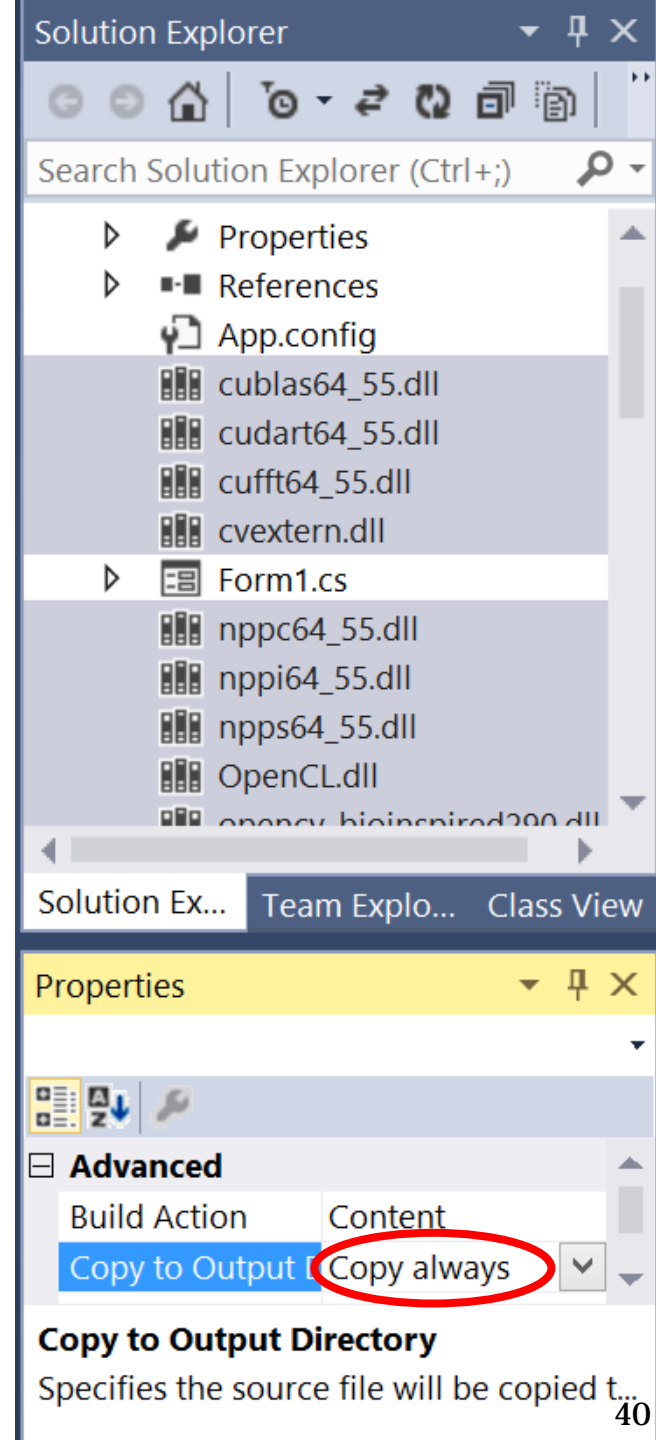
- Navigate to C:\Emgu\emgucv-windows-universal-cuda 2.9.0.1922\bin\x64
- *filter* by Executable Files (Enter \*.dll as File Name)
- **Add** all .dll files contained in the folder



# Alternative way for (Step 4)

## - \*.DLL References (iii)

- Select all .dll files added in the previous step (unselect Form1.cs) and change the 'Copy to Output Directory' property to **'Copy always'**
- *This will copy these files to the build directory so they can be referenced when running your Windows Forms application*
- *This will make the projects bigger but easier to get working*





# Pros/Cons of the alternative way “copying” \*.DLL References

## ■ Advantages

- Easy release. Just copy the whole project folder to deliver the code
- Emgu CV version independent

## ■ Disadvantages

- Complex to setup
- Project folder size becomes large (over 1.1 GB)