VisionLab 7.5

.NET Quick Start

www.openwire.org
www.mitov.com

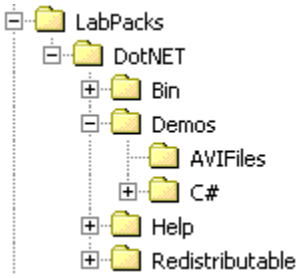
Index

Installation

VideoLab comes with an installation program. Just start the installation by double-clicking on the Setup.exe file and follow the installation instructions.

Where is VisionLab?

After the installation VideoLab is located under a single root directory. The default location is C:\Program Files\LabPacks or C:\Program Files (x86)\LabPacks on 64 bit systems. During the installation the user has the option to select alternative directory. Here is how the directory structure should look like after the installation:

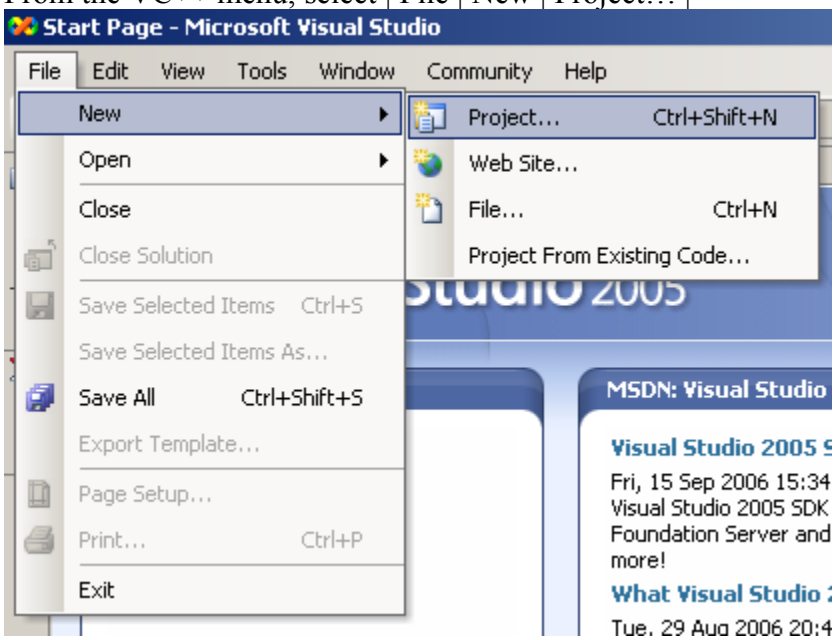


Under the “Demos” directory are located the demo files. The help files and the documentation are located under the “Help” directory. The component . NET 2.0/3.5/4.0 assemblies and the redistributable DLL files are located under the “Bin” directory.
It is a great idea to start by opening and compiling the demo files. The demo projects ware developed with Visual C# 2005.

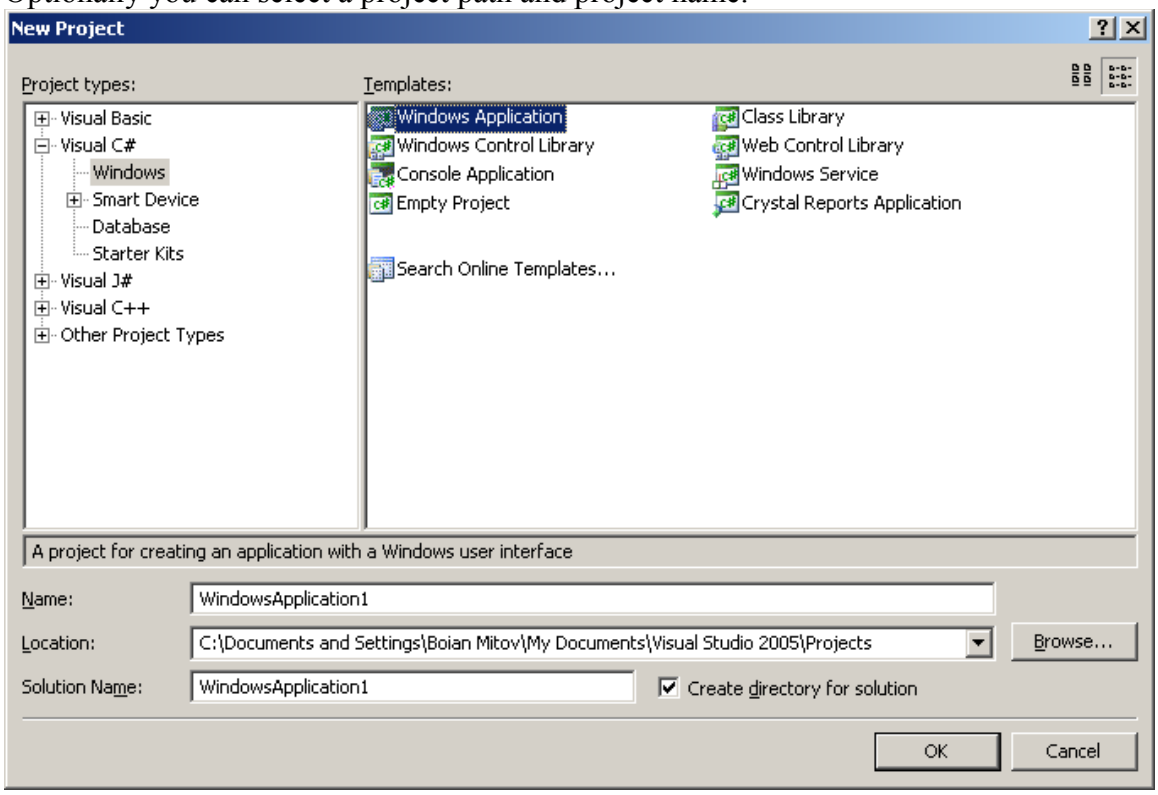
Creating a new VisionLab project in Visual C#

All of the examples in this manual start with creating a C# Windows .NET based project. The following chapters will assume that you have created the project and will teach you how to add specific VisionLab functionality.

Start by creating a new project.
From the VC++ menu, select | File | New | Project… |



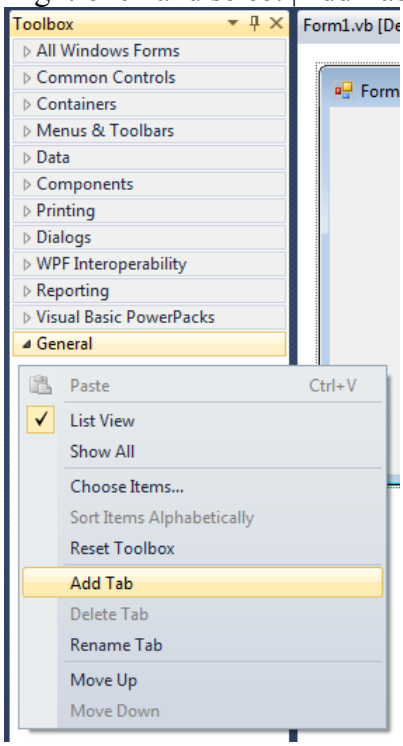
In the "New Project" dialog select | Visual C# | Windows Application |
Optionally you can select a project path and project name:



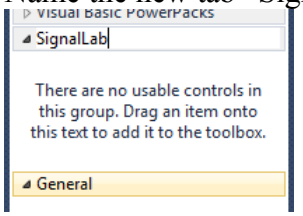
Click OK.

Installing the components on the Toolbox

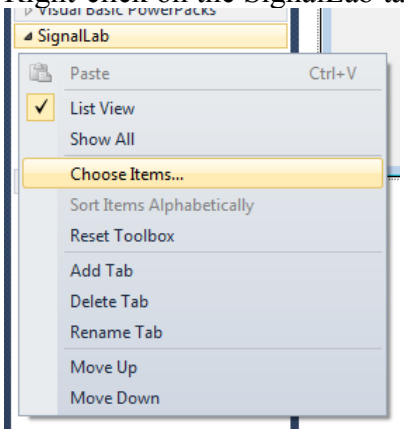
Before using the components in your project, you will have to install them on the component Toolbox.
The install in version 3.1 and up will automatically install the components on the toolbar, however if it fails, or if you have selected not to do so during the installation, here is a way to install the components manually:
We assume that you have already created a project, and the toolbox with the .NET components has appeared.
Open the component toolbox and expand the General section.
Right-click and select |Add Tab| from the menu:



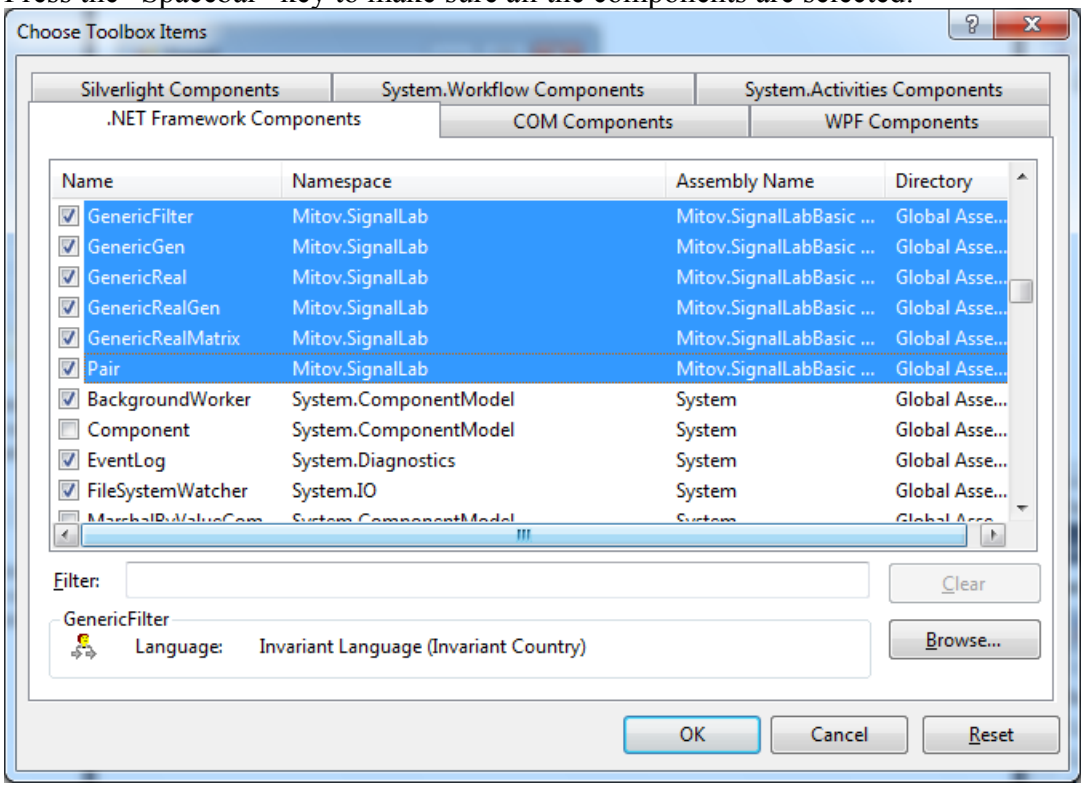
Name the new tab “SignalLab”:



Right-click on the SignalLab tab and select |Choose Items…| from the menu:

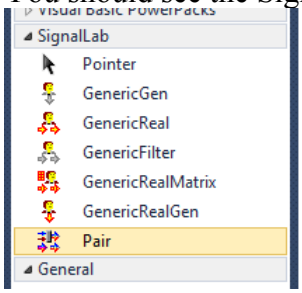


In the "Choose Toolbox Items" dialog select the components that belong to the Mitov.SignalLabBasic.DLL (You can order the components by "Assembly Name"). Press the "Spacebar" key to make sure all the components are selected:



Click OK.

You should see the SignalLabBasic components on your toolbox:



Close and restart the Visual Studio IDE, then reopen the project.

On the "SignalLab" tab install Mitov.SignalLabAdditional.dll.

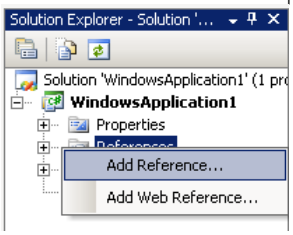
Repeat the same steps adding 4 more tabs named "MediaLab" "AudioLab", "VideoLab", and "VisionLab".

On the "MediaLab" tab install Mitov.MediaLabBasic.dll.
On the "VideoLab" tab install Mitov.VideoLabBasic.dll and Mitov.VideoLabAdditional.dll.
On the "VisionLab" tab install Mitov.VisionLab.dll.

Now you can start using the components in your .NET development.

Adding the necessary assembly references to your application

Visual studio will automatically add the assemblies being referenced when adding components to the project. If this mechanism fails, you can manually add the necessary assemblies as shown here:
In the "Solution Explorer" select the "References" node and right-click on it.
From the menu select |Add Reference…|



Navigate to the Select the Mitov.VideoLabBasic.dll from the LabPacks\Bin\2.0 subdirectory and add the necessary assemblies.
Here is the list of necessary assemblies:

- For Mitov.BasicLab.DLL – None.
- For Mitov.SignalLabBasic.DLL:
    a. Mitov.BasicLab.DLL
- For Mitov.AudioLabBasic.DLL:
    a. Mitov.BasicLab.DLL
- For Mitov.MediaLabBasic.DLL:

    a. Mitov.BasicLab.DLL
    b. Mitov.AudioLabBasic.DLL:
- For Mitov.VideoLabBasic.DLL:
    a. Mitov.AudioLabBasic.DLL
    b. Mitov.BasicLab.DLL
- For Mitov.VideoLabAdditional.DLL:
    a. Mitov.VideoLabBasic.DLL
    b. Mitov.AudioLabBasic.DLL
    c. Mitov.BasicLab.DLL
- For Mitov.VisionLab.DLL:
    a. Mitov.VideoLabAdditional.DLL
    b. Mitov.VideoLabBasic.DLL
    c. Mitov.AudioLabBasic.DLL
    d. Mitov.BasicLab.DLL

Creating a simple contour detection application

Create and setup a new project as described in the "Creating a new VideoLab project in Visual C#" chapter.

From the "VideoLab" tab on the Toolbox select and drop on the form the following two components:

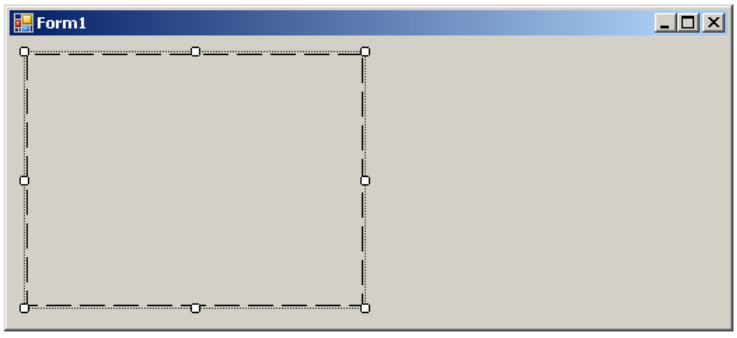
 - AVIPlayer

 - ImageDisplay

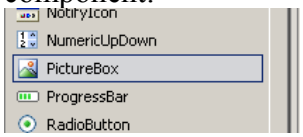
 - Canny

 - FindContours

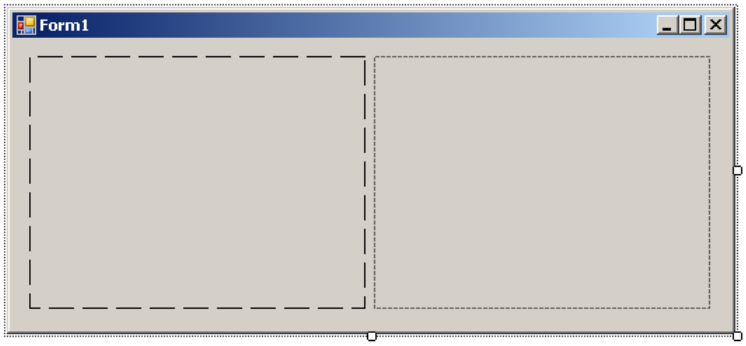
Arrange the ImageDisplay as shown here:



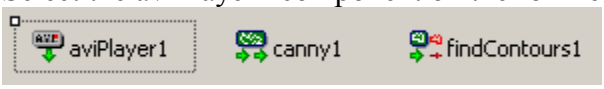
From the "Toolbox" switch to the "Common Controls" group and select the "PictureBox" component:



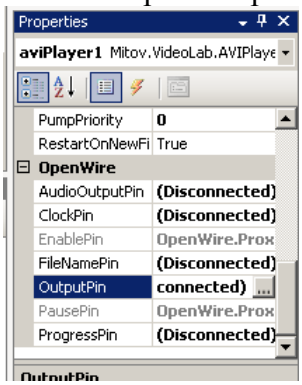
Place the component on the form and arrange it as shown here:



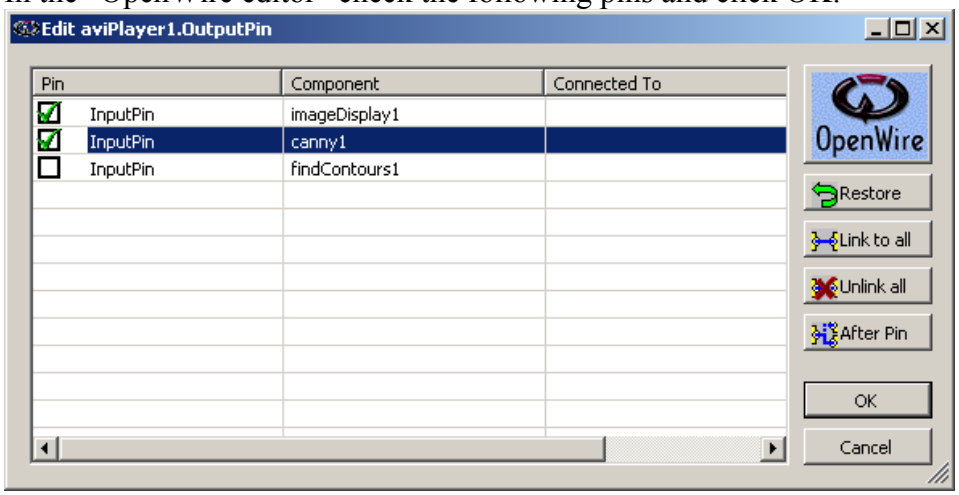
Select the aviPlayer1 component on the form editor:



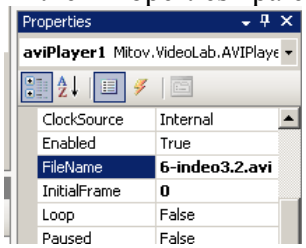
In the "Properties" palette go to the "OutputPin" property and click on the "…" button:



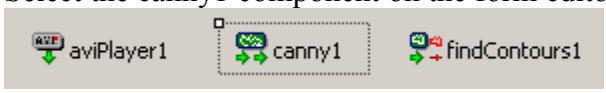
In the "OpenWire editor" check the following pins and click OK:



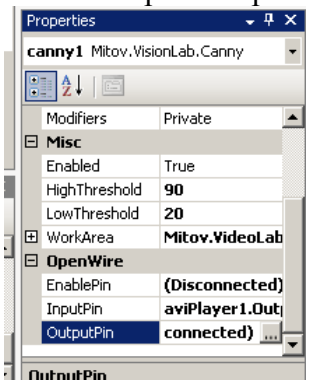
In the “Properties” palette go to the “FileName” property and set a video file to play:



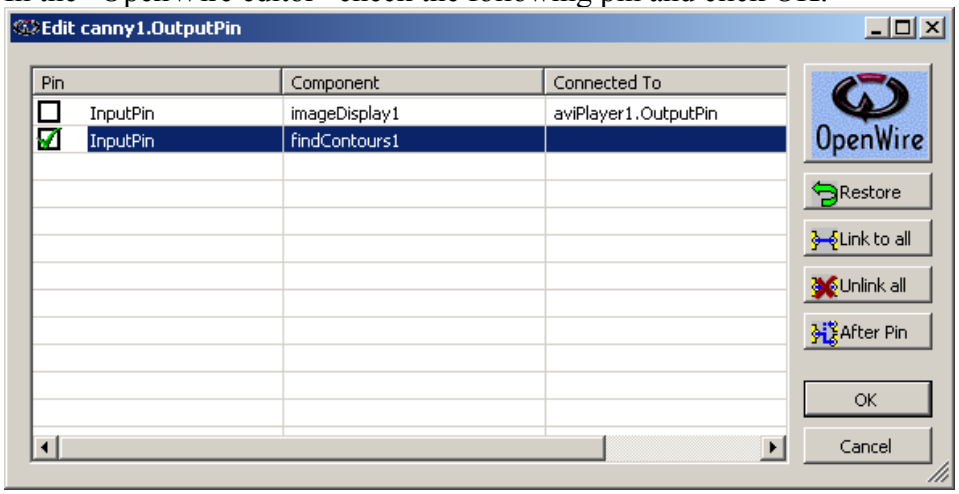
Select the canny1 component on the form editor:



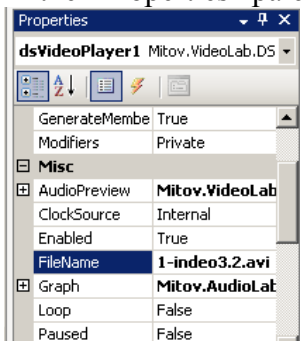
In the “Properties” palette go to the “OutputPin” property and click on the “…” button:



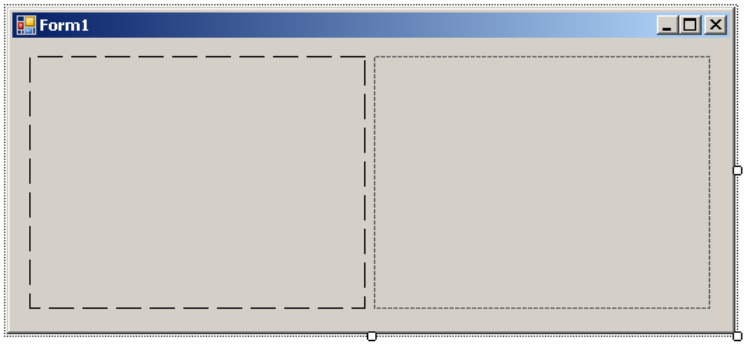
In the “OpenWire editor” check the following pin and click OK:



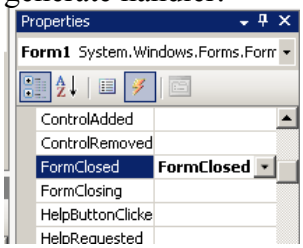
In the “Properties” palette go to the “FileName” property and set a video file to play:



Select the form:



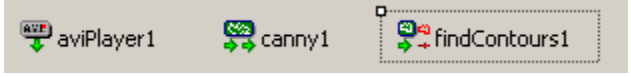
In the “Properties” palette switch to events and double click on the FormClosed event to generate handler:



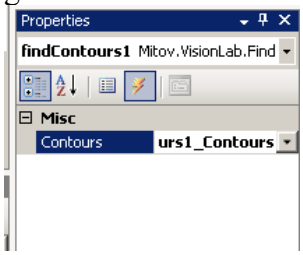
Add the highlighted line in the source file:

```
        private void Form1_FormClosed(object sender,
FormClosedEventArgs e)
        {
            aviPlayer1.Stop();
        }
```

Select the findContours1 component on the form editor:



In the “Properties” palette switch to events and double click on the Contours event to generate handler:

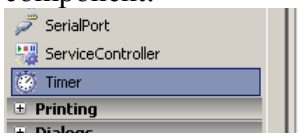


Add the highlighted line in the source file:

```
        bool m_ShowContours = true;

        private void findContours1_Contours(object Sender,
Mitov.VisionLab.ContoursEventArgs Args)

        {
            Mitov.VisionLab.Contours aContours = Args.Contours;


            int aContoursCount = aContours.Count;
            if (m_ShowContours)
            {
                System.Drawing.Bitmap ABitmap = new
System.Drawing.Bitmap(240, 180);
                Graphics g = Graphics.FromImage(ABitmap);


                System.Drawing.SolidBrush ABrush = new
SolidBrush(Color.White);
                g.FillRectangle(ABrush, 0, 0, 240, 180);
                System.Drawing.Pen APen;
                System.Drawing.Pen AGgeenPen = new
System.Drawing.Pen(Color.Green, 1);
                System.Drawing.Pen ABluePen = new
System.Drawing.Pen(Color.Blue, 1);
                System.Drawing.Pen ARedPen = new
System.Drawing.Pen(Color.Red, 1);
                for (int i = 0; i < aContoursCount; i++)
                {
                    Mitov.VisionLab.Contour aContour =
aContours.get_Items(i);
                    if (aContour.ContourType ==
Mitov.VisionLab.ContourType.Outer)
                        APen = AGgeenPen;


                    else
```

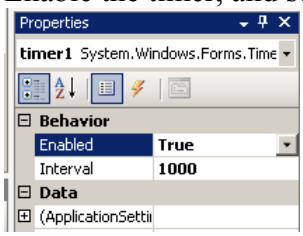
```
                        APen = ABluePen;

                    int ACount = aContour.Count;
                    for (int j = 1; j < ACount; j++)
                        g.DrawLine(APen, aContour.get_Items(j - 1),
aContour.get_Items(j));

                    g.DrawRectangle(ARedPen,
aContours.get_Items(i).BoundRect);
                }

                pictureBox1.Image = ABitmap;
                m_ShowContours = false;
            }
        }
```

From the “Toolbox” switch to the “Components” group and select the “Timer” component:



Select the timer1 component on the form editor:

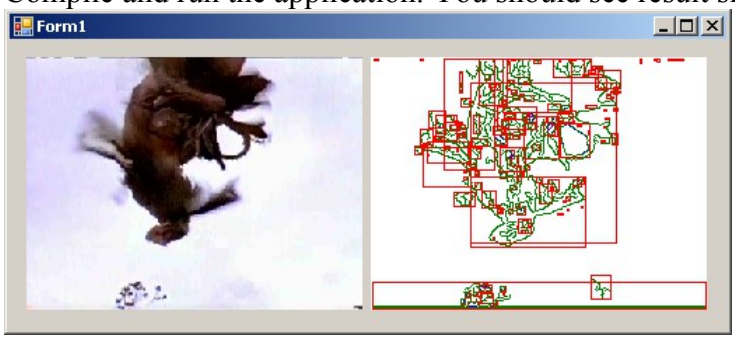
Enable the timer, and set the Interval to 1000:



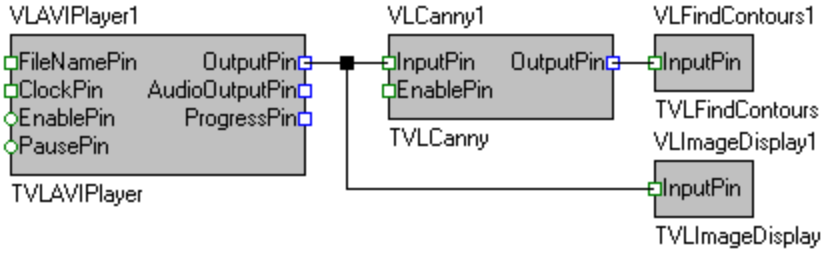
Double-click on the timer1 component, and add the highlighted lines in the source file:

```
        private void timer1_Tick(object sender, EventArgs e)
        {
            m_ShowContours = true;
        }
```

Compile and run the application. You should see result similar to this one:



Here are the OpenWire connections in this application:



Distributing your application

Once you have finished the development of your application you most likely will need to distribute it to other systems. Version 5.0.2 and higher of the library will move all the necessary DLL files in the Release directory of your project. You will only need to distribute the files in the directory. To use this feature, make sure that the “Copy Local” property is set for all the assemblies from the library. Please check with the Visual Studio help for your version of Video Studio on how to configure assemblies as private assemblies.

Deploying your 32 bit application with the IPP DLLs

The compiled applications can be deployed to the target system by simply copying the executable. The application will work, however the performance can be improved by also copying the Intel IPP DLLs provided with the library.
The DLLs are under the [install path]\LabPacks\IppDLL\Win32 directory([install path] is the location where the library was installed).
In 32 bit Windows to deploy IPP, copy the files to the [Windows]\System32 directory on the target system.
In 64 bit Windows to deploy IPP, copy the files to the [Windows]\SysWOW64 directory on the target system.
[Windows] is the Windows directory - usually C:\WINNT or C:\WINDOWS
This will improve the performance of your application on the target system.

Deploying your 64 bit application

The current version of the library requires when deploying 64 bit applications, the Intel IPP DLLs to be deployed as well.

The DLLs are under the [install path]\LabPacks\IppDLL\Win64 directory([install path] is the location where the library was installed).

To deploy IPP, copy the files to the [Windows]\System32 directory on the target system.
[Windows] is the Windows directory - usually C:\WINNT or C:\WINDOWS
This will improve the performance of your application on the target system.