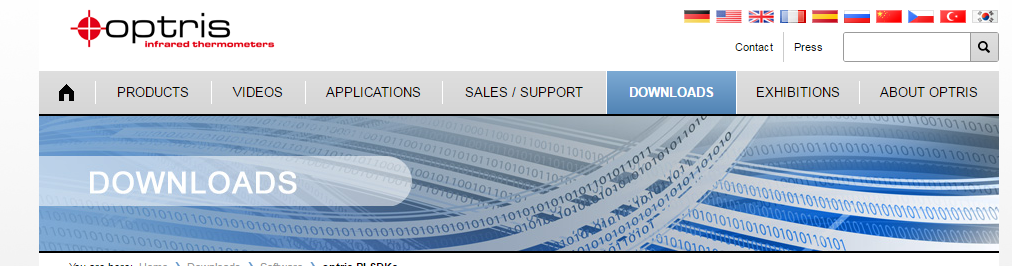
PI Connect – IR Camera Appication and API in c#

This document and the example project shows how to use the PI connect IR camera from <http://www.optris.com>



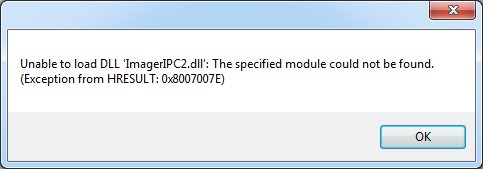
The PI connect C# wrapper is based on the ImageIP2.dll for 32 bit

Or the ImagerIPC2x64.dll for the 64-bit version.

Optris supply to us a C# wrapper called IPC2.cs

You can find it in the Drivers folder of this repository.

If you open the application and it cannot find the dll , it will preset the message from the application core messaging:

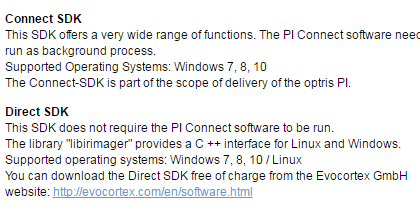


You can put the dll in the release or debug directory of the current application or in the Path of your computer.

But we know that the PATH tends to grow and windows don’t like that in 2017

There are two options of SDk

One that requires that he PI Camera connect will run ( IPC ) and one without:

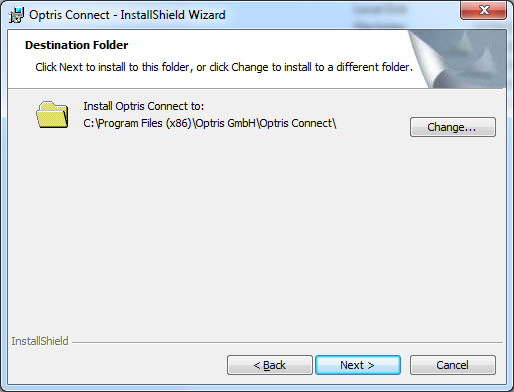


Download it from here:

<http://www.optris.com/downloads-software>

The Connect-SDK is part of the scope of delivery of the optris PI.





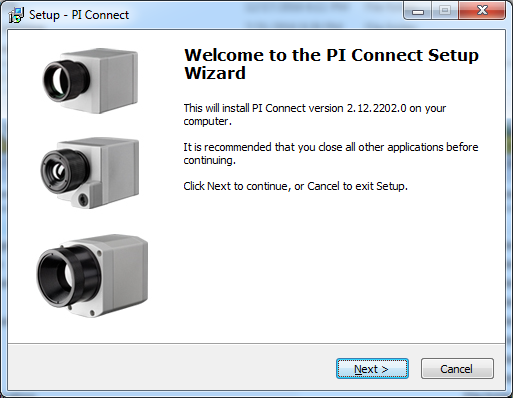
Download the application.

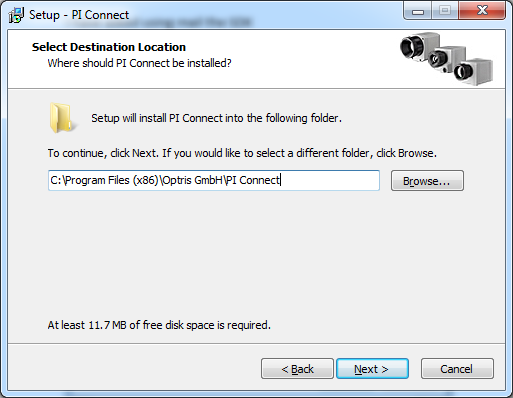
Our application will work with IPC to communicate with it.  
The PI Connect needs to be open.

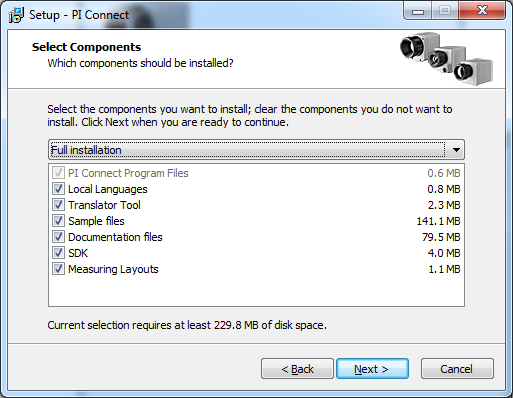
I have asked using mail the SDK

And got: pi-connect-rel-2-12-2202-0.zip

This is the installer:



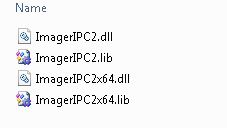




Note that there is **SDK**!!

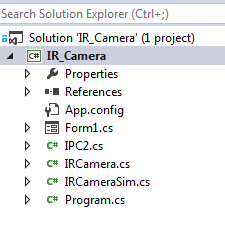
After installation is complete, we can find the DLL for the imager.

C:\Program Files (x86)\Optris GmbH\PI Connect\SDK\Lib\v120



The DLL we need.

The visual studio 2013 solution consist of:



Window form,

IPC2 from optris

IRCamera - a C# class that I wrap the IPC2 which the application will use.

The window form does not talk directly with the IPC2.cs but through my class.

How to use the IR Camera class:

|  |
| --- |
| IRCamera IR;  try  {  IR\_Camera.IRCamera.Callback p = new IRCamera.Callback(IRApiCallback);  IR = new IRCamera(p);  IR\_Camera.IRCamera.\_NewFrame n = new IRCamera.\_NewFrame(NewFrame);  IR\_Camera.IRCamera.\_OnFrameInit o = new IRCamera.\_OnFrameInit(OnFrameInit);  IR.SetNewFrameCallbacks(n, o);  IR.OpenIRimeger(false);  }  catch (Exception err)  {  MessageBox.Show(err.Message);  } |

You can create a thread to show the temperatures:

|  |
| --- |
| void ShowIR()  {  while (m\_running)  {  bool errors;  float[] TempArea = IR.TempMesure(0, out errors, errorsVect);  label6.Text = String.Format("Target1-Temp: {0:##0.0}°C", TempArea[0]);  label7.Text = String.Format("Target2-Temp: {0:##0.0}°C", TempArea[1]);  label8.Text = String.Format("Target3-Temp: {0:##0.0}°C", TempArea[2]);  label9.Text = String.Format("Target4-Temp: {0:##0.0}°C", TempArea[3]);  label10.Text = String.Format("Target5-Temp: {0:##0.0}°C", TempArea[4]);  Thread.Sleep(200);  }  } |

In case you want to view the IR camera video bitmaps in your app, follow the code

IR\_Camera.IRCamera.\_NewFrame n = new IRCamera.\_NewFrame(NewFrame);

IR\_Camera.IRCamera.\_OnFrameInit o = new IRCamera.\_OnFrameInit(OnFrameInit);

If those Functions NewFRame and OnFrameInit