Image Processing in LabVIEW for FRC

Contents

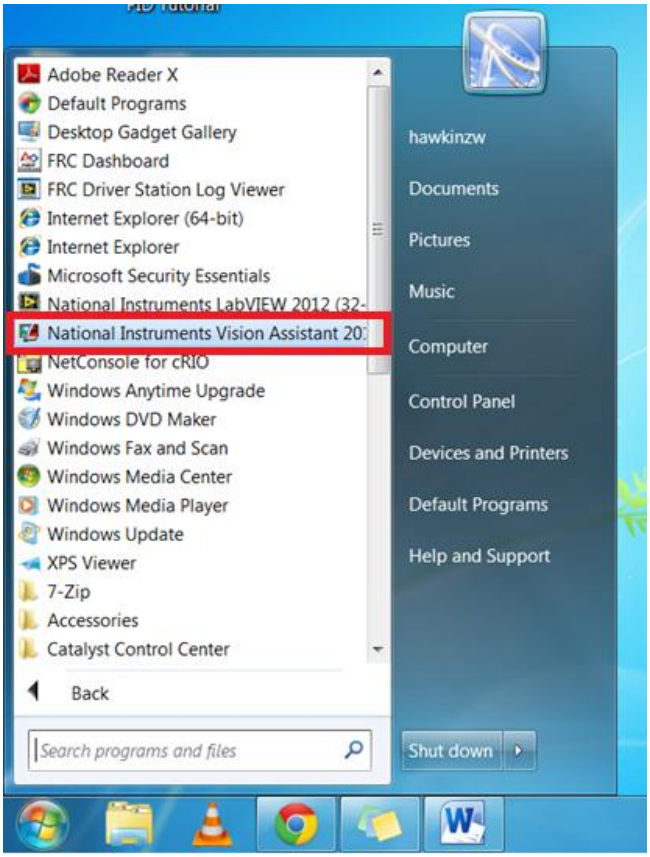
Introduction

This tutorial serves as an introduction to creating and implementing machine vision algorithms in LabVIEW. We'll introduce tools for developing algorithms and walk through how to integrate your vision code into a FRC framework.

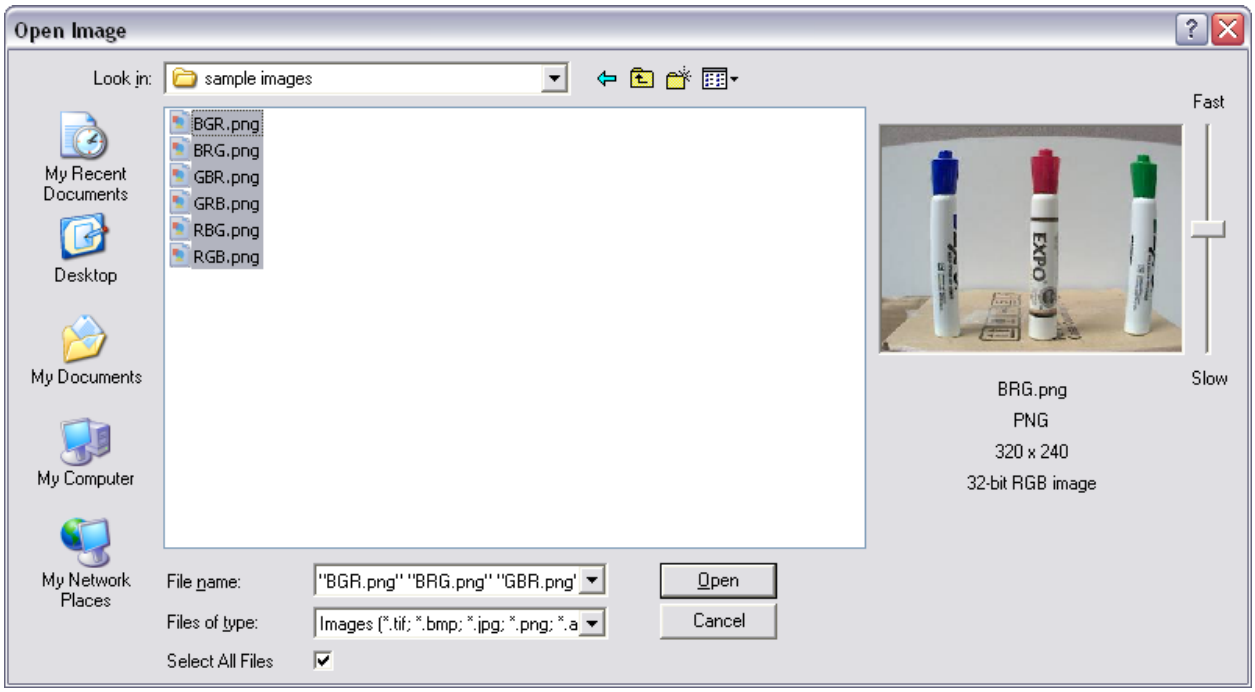
Using Vision Assistant

Designing a Script - In addition to LabVIEW, you have access to a powerful tool for prototyping and testing image processing applications called National Instruments Vision Assistant. Vision Assistant provides access to all the Vision functionality available in LabVIEW but in a prototyping environment. Let's explore how to use Vision Assistant by opening up a few sample images and creating a script:

1. Download and unzip the *sample images* folder attached at the bottom of this tutorial. These images were taken with the Axis Camera connected to a cRIO in the same configuration that will be used during competition.
2. Open the Vision Assistant by going to **Start»All Programs»National Instruments»Vision Assistant 2012»Vision Assistant 2012**. Select **Open Image** from the splash screen.

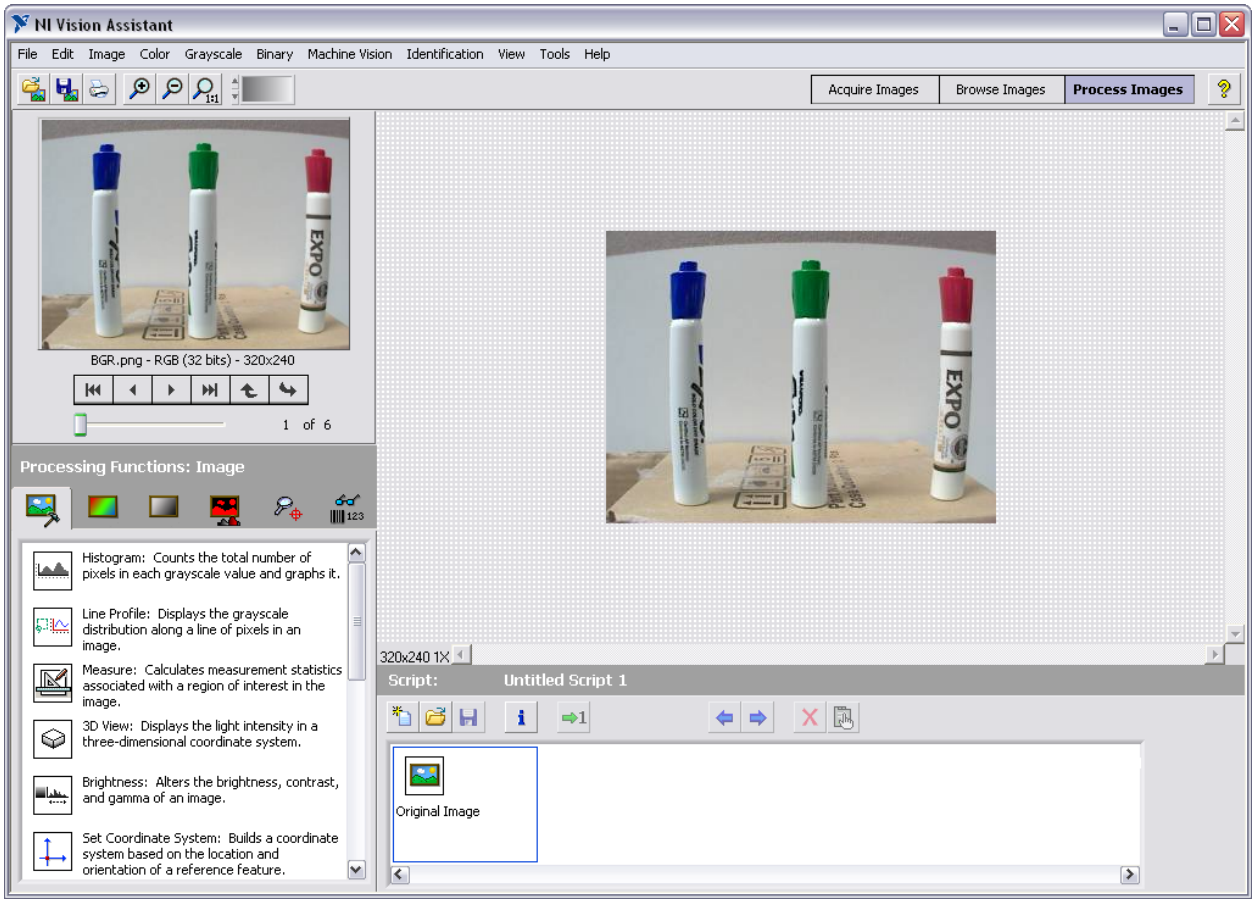


3. In the *Open Image* dialog box, you can hold the **Shift** key to select multiple images, or click on the **Select All Files** checkbox at the bottom of the dialog box. Once all the desired images have been selected, click the **Open** button.



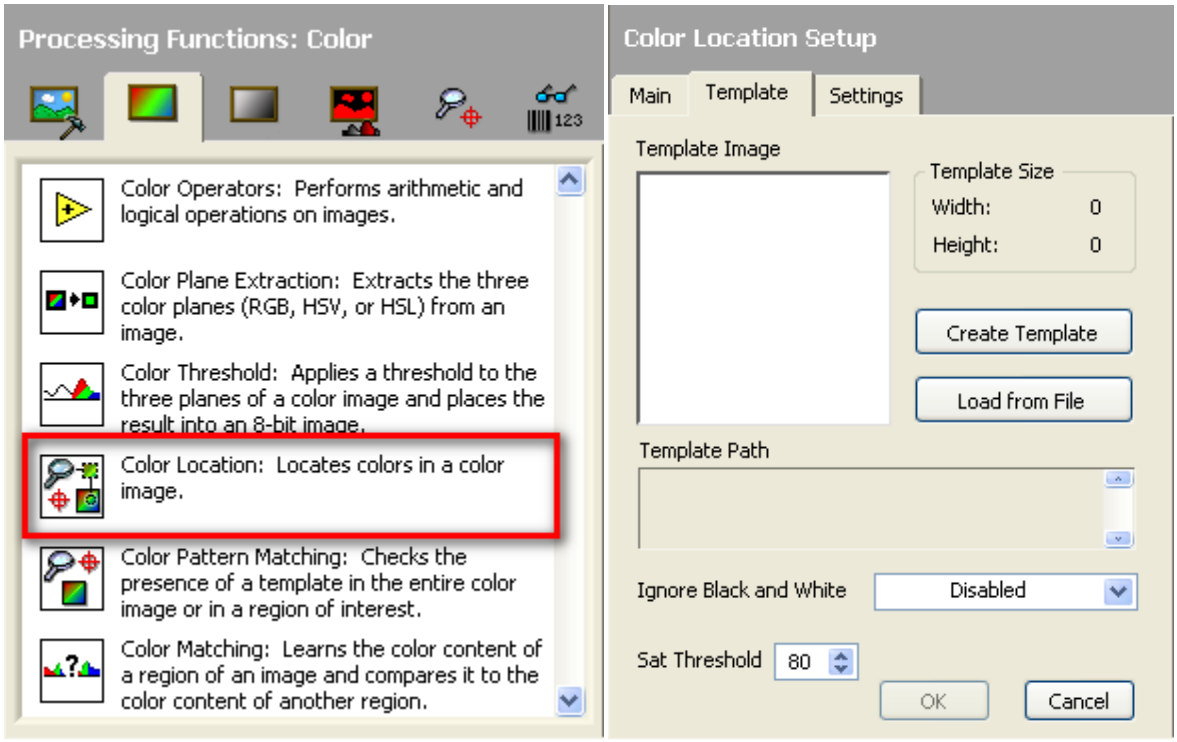
4. All the images you have selected are loaded into the image browser, and are accessible from the reference window in the upper left, or by switching to the browser view, by clicking on **Browse Images** in the upper right. Hover over a browser button to see what it does. You can also zoom in and out of the active image. The rest of the screenshots in this document have usually been zoomed in on once.

You will build up an image processing algorithm step by step. Steps in Vision Assistant provide the same functionality as the Vision VIs in LabVIEW. A series of steps is called a script.

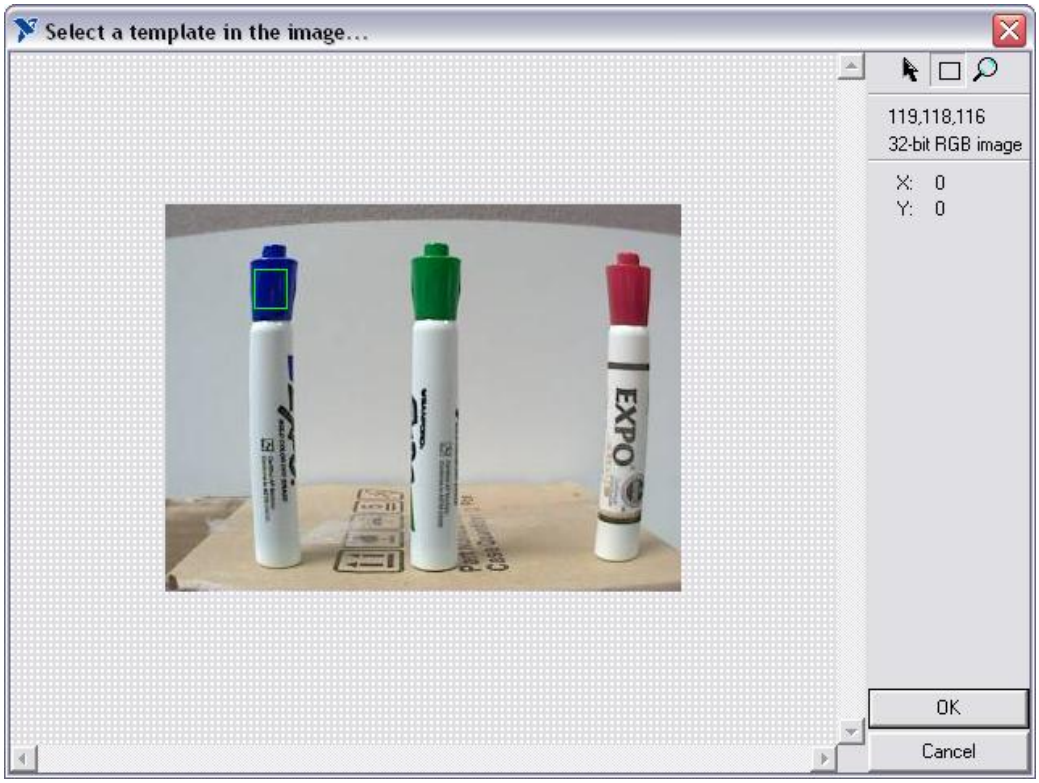


Let's say that our goal is to detect the red, green, and blue caps in our sample images. We'd also like to differentiate between them so that we could choose only to retrieve the blue markers. Or maybe we want to grab both the green and red and leave the blue. Furthermore, we'd like to know which order they are in, and how far apart they are. This information could be used to make a variety of decisions, such as moving motors to retrieve or manipulate the objects. Here's one way to do it:

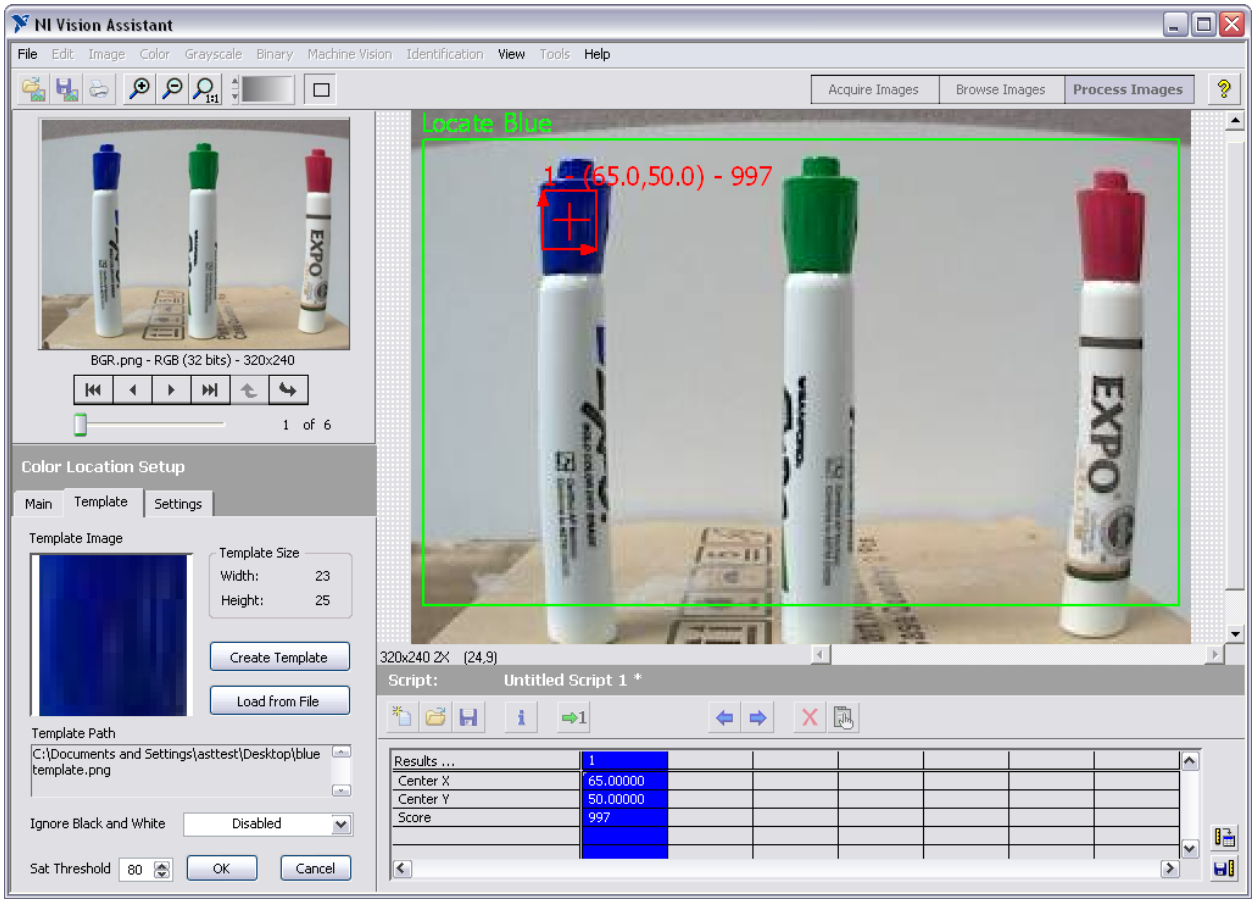
Vision Assistant conveniently contains a step called *Color Location*. We can use this step to create a template of the color that we want to find, and then to search for that template in our image. Go to the **Processing Functions: Color** tab and double-click on the **Color Location** step. This will bring up the step setup.



From the step setup, click the **Create Template** button. From the dialog box, click on the image and drag a rectangle within the blue area and click **OK**. Save the template as *blue template*.

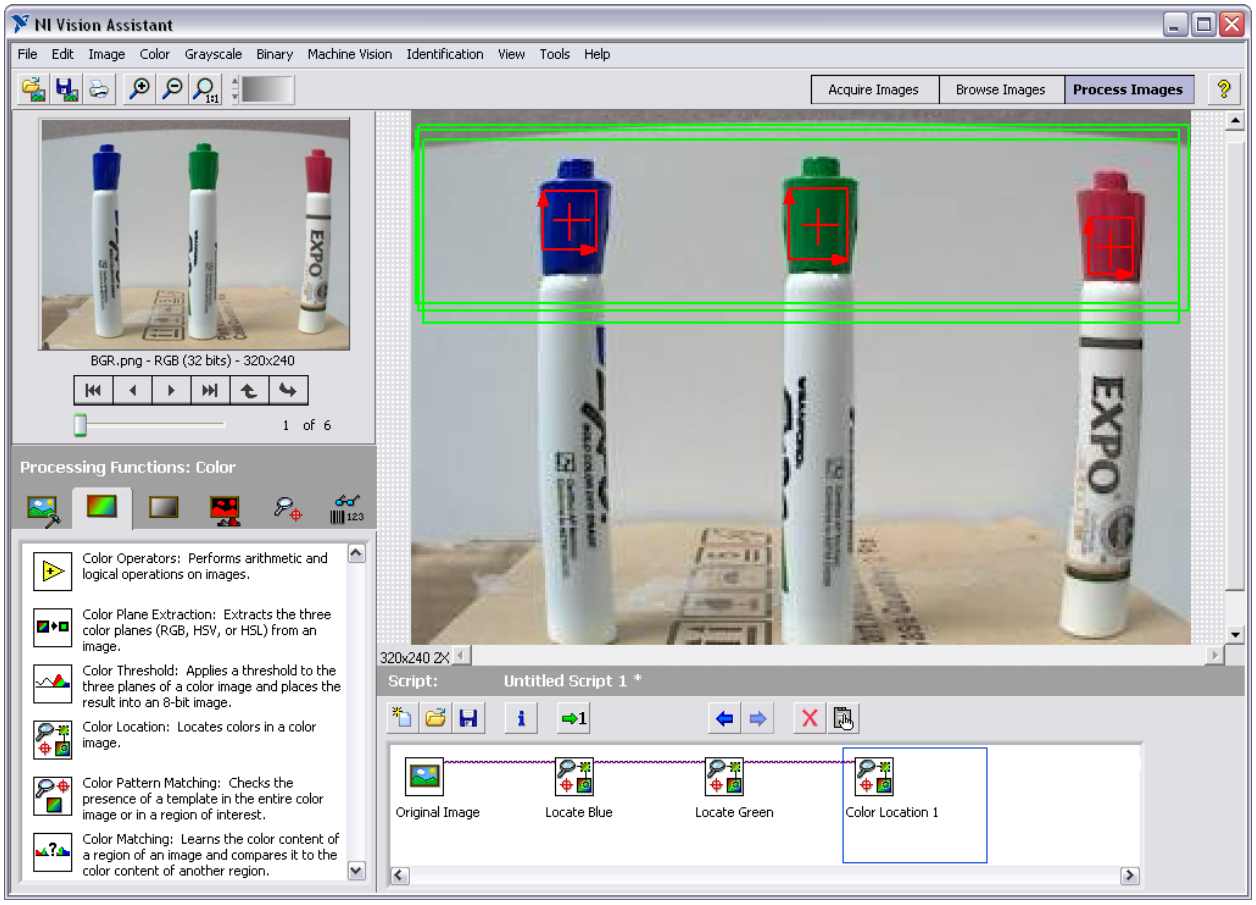


Notice how the results are displayed along with the search ROI (Region of Interest).

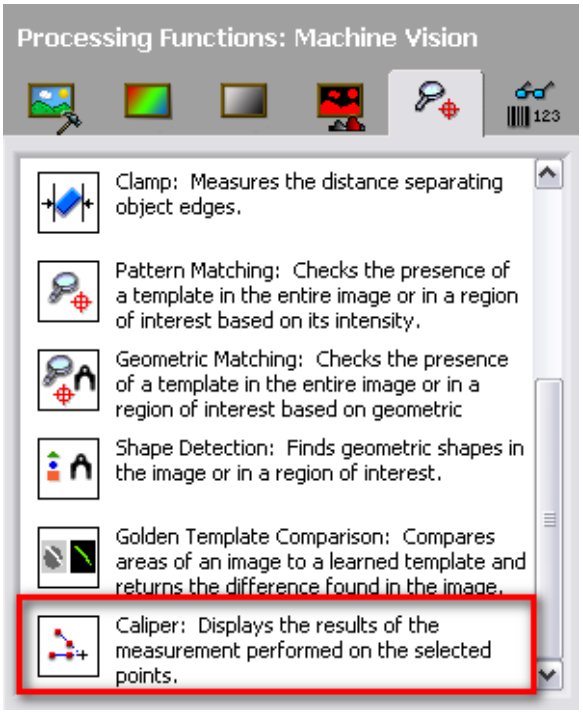


Name the step *Locate Blue* from the **Main** tab of the step setup, and then click **OK** to exit the step setup. The step you just configured will show up in the script view. You can go back and re-configure the step at any time by double-clicking on it. Try going back and making the ROI smaller. You should make your search ROIs as small as possible, while still encompassing all possible location of all your objects. The smaller the ROI, the faster the step will run because the algorithm is operating on a smaller area.

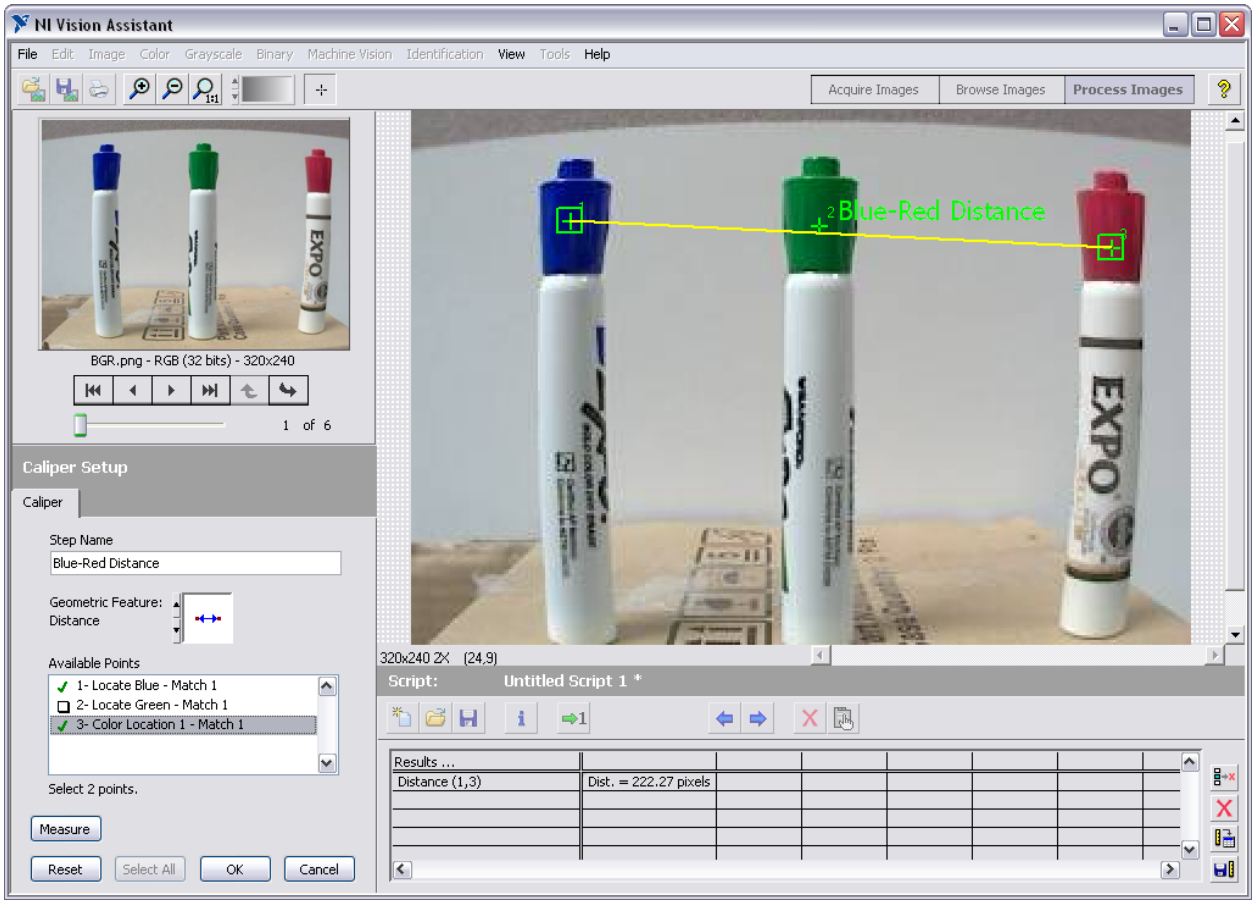
Repeat the above procedure, but for the green and red caps. You should have something similar to the following screenshot after adding a Locate Color step for red and green as well as blue.



Now let's measure the distance between the caps. Double-click on the **Caliper** step from the **Processing Functions: Machine Vision** tab (scroll down to find it).



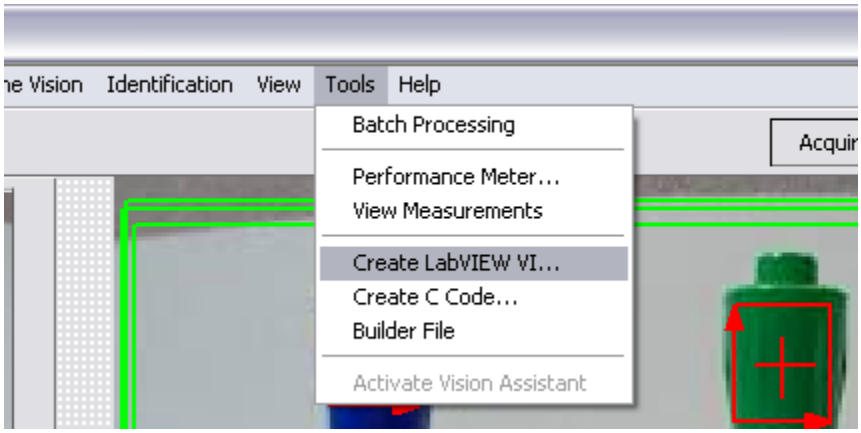
From the Caliper Setup tab, double-click on two points that you want to measure the distance between (select the **Locate Blue** and **Locate Red** matches), then click the **Measure** button. The results will show up, and you will see a yellow line connecting the points on your image. Click **OK** to exit the caliper step setup.



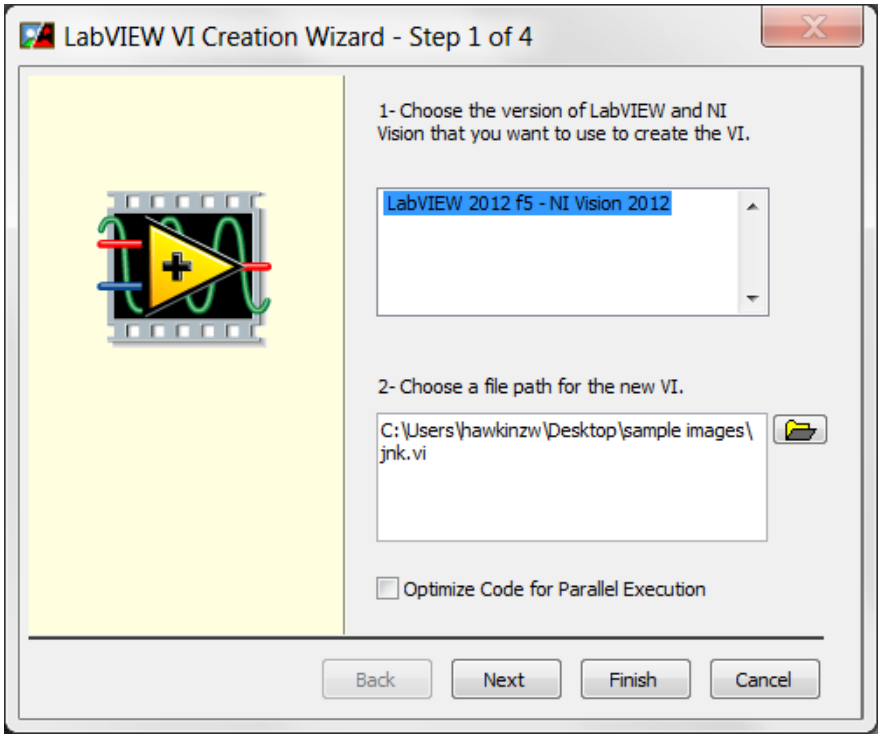
Repeat the above procedure to measure the blue-green and the green-red distances.

Now we have a working prototype that will distinguish, locate, and measure the relating distances between three colored caps. Cycle through the rest of the images in the browser to see if the prototype script correctly recognizes the caps even if their locations are swapped. You can do this by browsing to a new image, and then pressing the **Make Image Active** button, which is the right-most button from the browser toolbar.

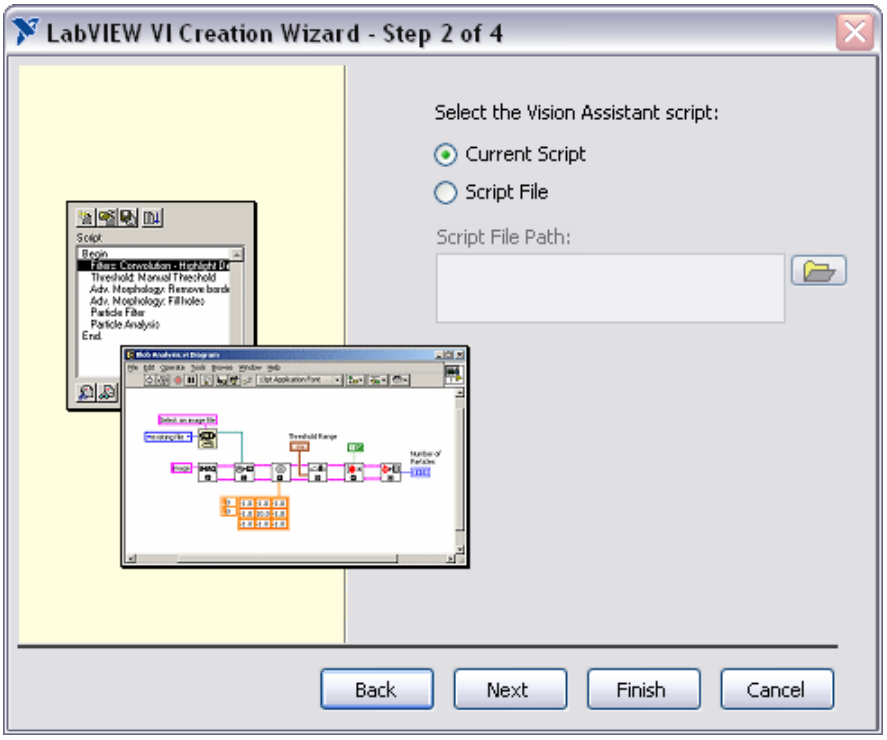
Creating a LabVIEW VI from Your Script- At this point you can create a LabVIEW VI from your script and start tailoring it to your needs. Go to the **Tools** menu, and select **Create LabVIEW VI.** Note that you can also create C Code.



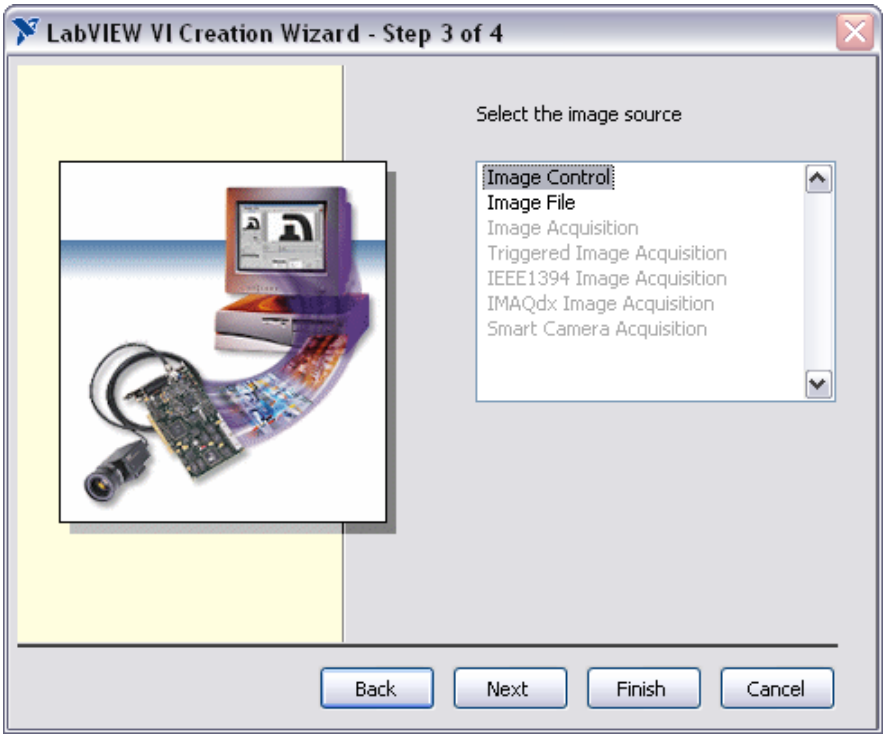
Browse to the file path where you would like to save the VI, then click **Next**.



Select **Current Script**, and click **Next**.

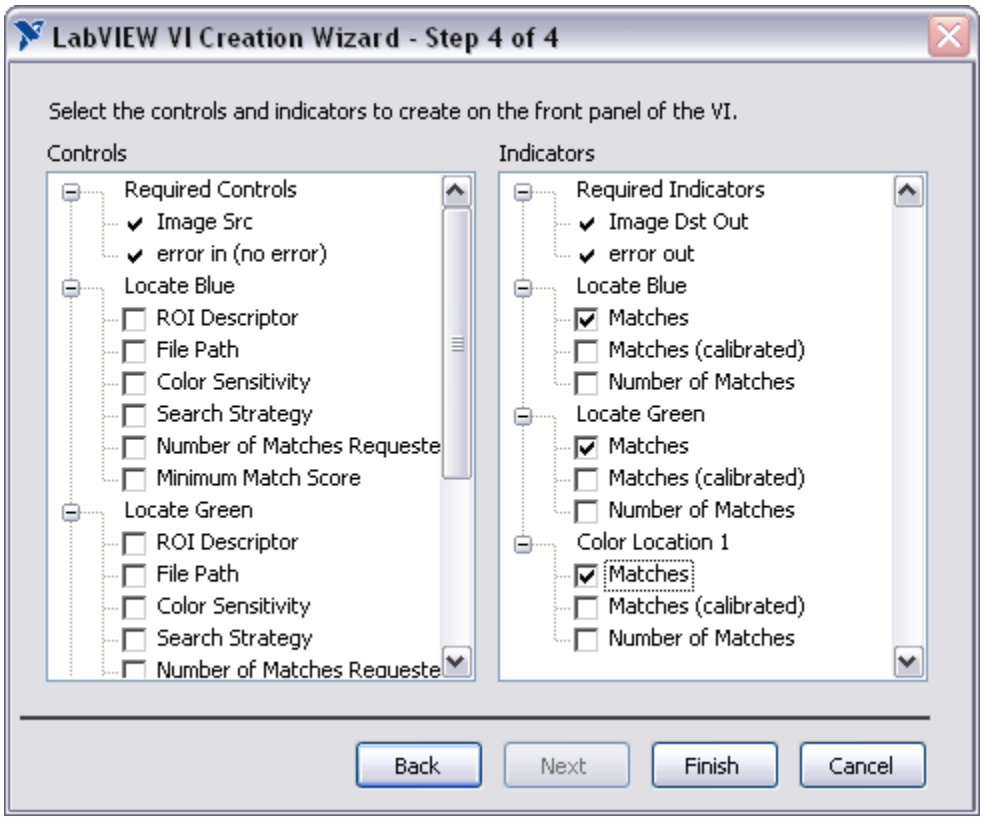


Select **Image Control**. We will be passing images to this VI from the Axis camera and not from file. Click **Next**.



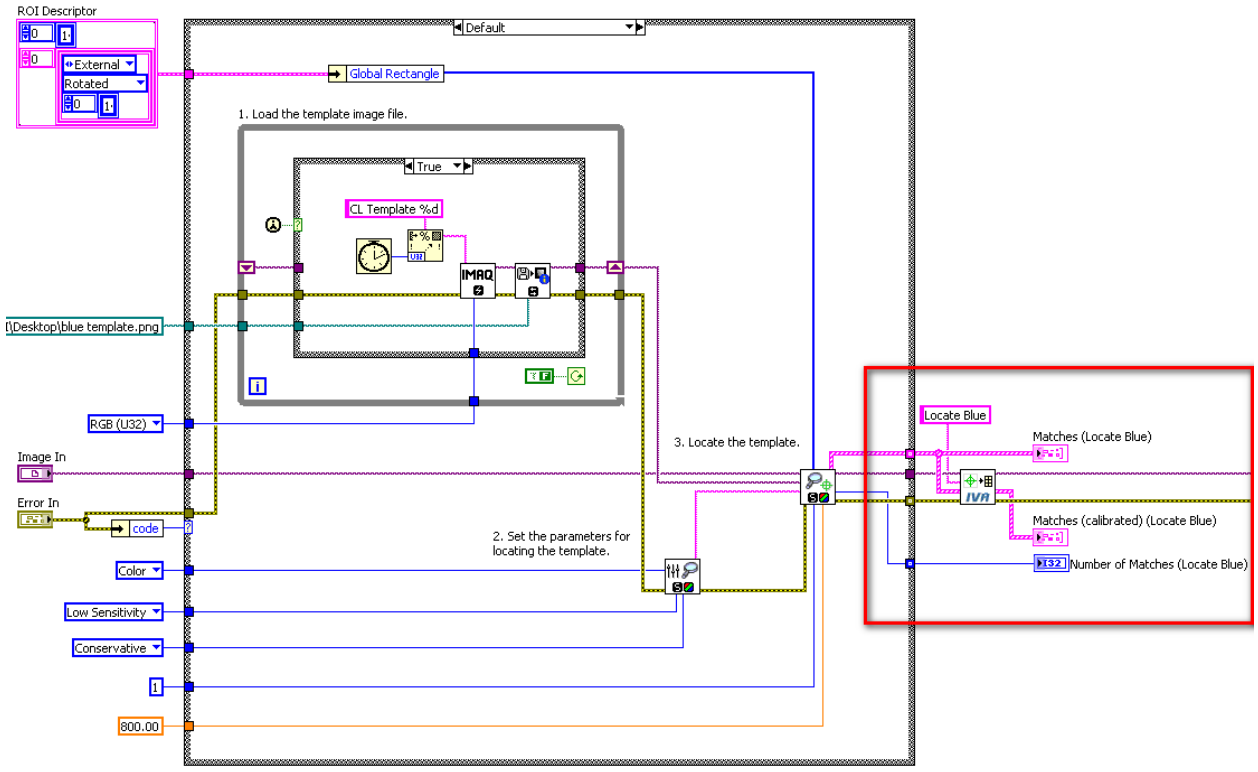
Select the controls and indicators you want Vision Assistant to create. For the boxes that you don't check, constants are created instead of controls, and no indicators are created. Usually

you would only select controls for values that you will need to change. Don't worry too much about over-selecting controls and indicators: once the VI is created, you can add or delete controls and indicators as needed. Once you are done selecting controls and indicators, click **Finish**.

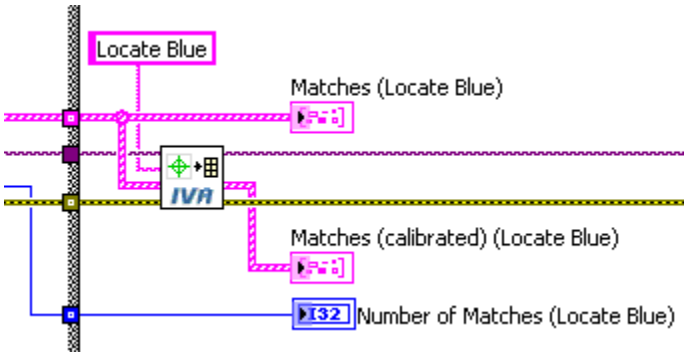


Vision Assistant will call LabVIEW and create a VI from your script. This may take a few moments. If you have to acknowledge that you have a certain number of days remaining for your trial version of LabVIEW FRC, and then the Getting Started screen pops up, you will have to repeat the process. Leave the Getting Started screen open, and go through the steps to create a VI from Vision Assistant again.

You can follow the data flow and see how it implements the same functionality as the script we created in Vision Assistant.



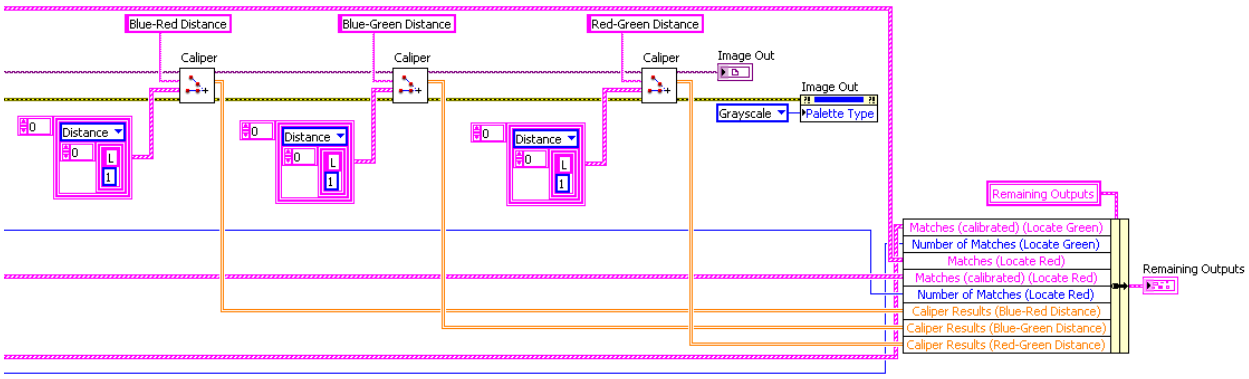
The section of code shown above corresponds to the first step of your script, the color location step. There will be three of these case structures, each one corresponding to one of your three color location steps. The region boxed in red on the right is reproduced below:



Any VIs that have the “IVA” label are subVIs that were generated from Vision Assistant. This particular subVI handles storing match results and calibration. If you don’t need to calibrate your image (convert pixel coordinates to real world coordinates), or you do that calibration elsewhere, then you can delete this VI and pass the image and error wires to the next case structure. It is always a good idea to go through the code generated from Vision Assistant to see what you want to keep, and what you can do without. The leaner you can make your code,

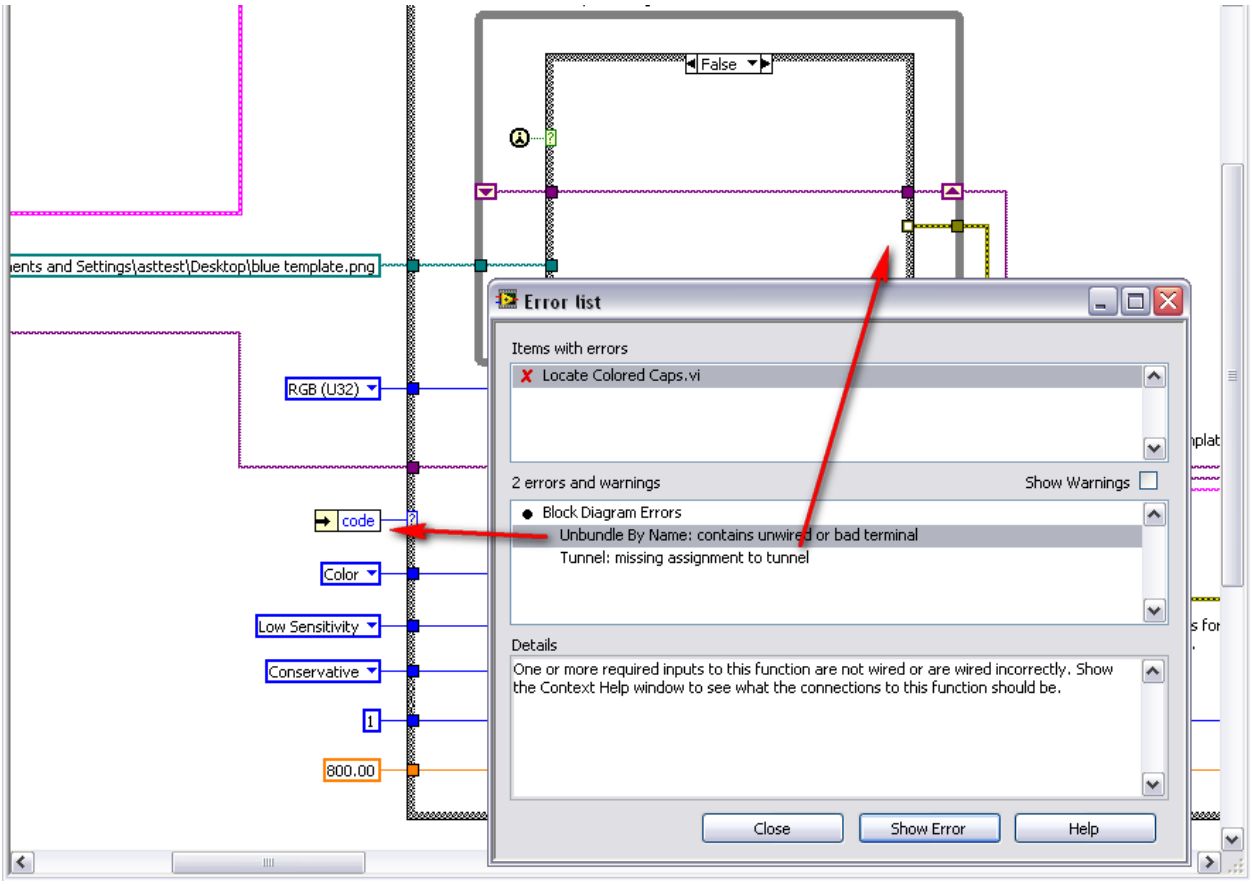
the faster it will execute, and in general, you will achieve a greater frame rate from your camera. The frame rate of the camera is determined by the time it takes the image acquisition and processing loop to execute. The more code you execute in this loop, the longer it is likely to take. You will open and edit the image acquisition loop of the basic FRC cRIO Robot Project a little later on in the tutorial.

The remaining code implements the caliper steps, and bundles all the remaining outputs into a cluster for ease of use when manipulating the controls and indicators on the front panel.

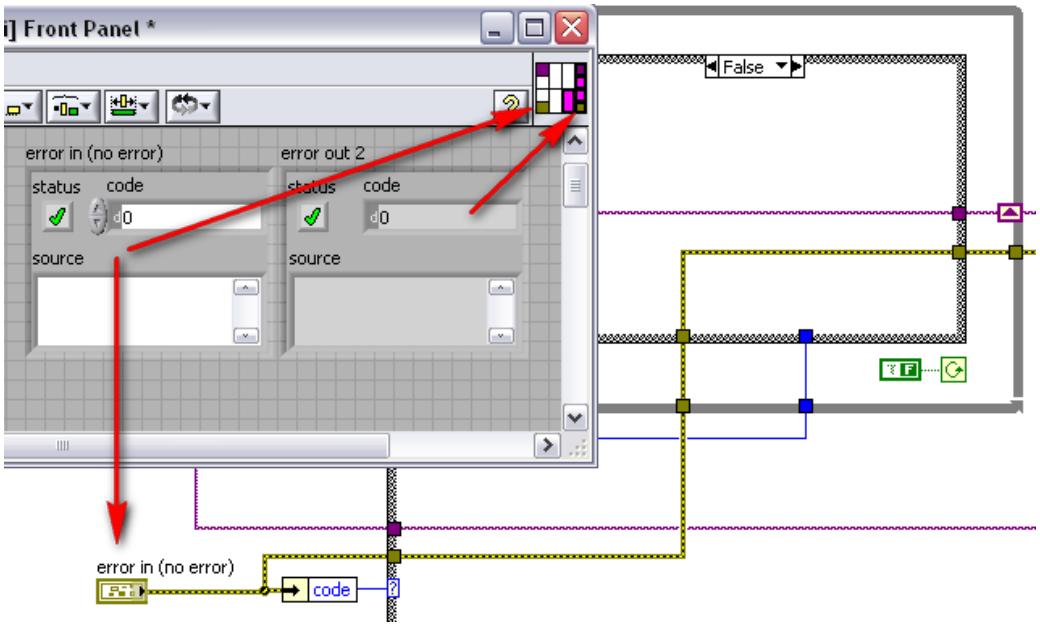


Notice that there is no overlay functionality. Vision Assistant does not include image overlays when generating LabVIEW code because overlays would add to the processing load, and are often not desired in the final version of the code. You can easily go back in and add overlays if you wish.

You will notice that the VI produced needs to be cleaned up a bit and may need some changes before it is executable. For this tutorial, there will be a few missing or unwired terminals or tunnels:



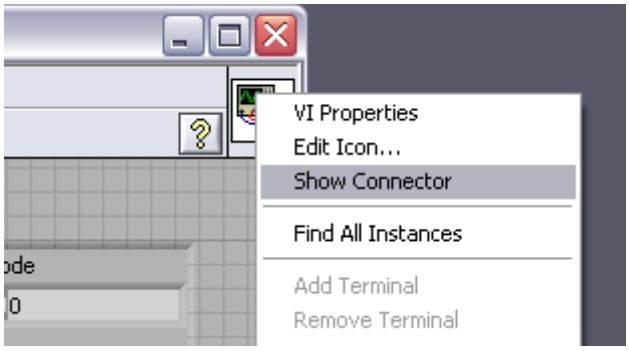
You can remedy these errors by right clicking on the tunnel or terminal that is missing an input and selecting **Create Constant.** Better yet, you could add error controls and then modify the connector pane of the subVI that you just created so that you can pass errors into and out of the subVI.



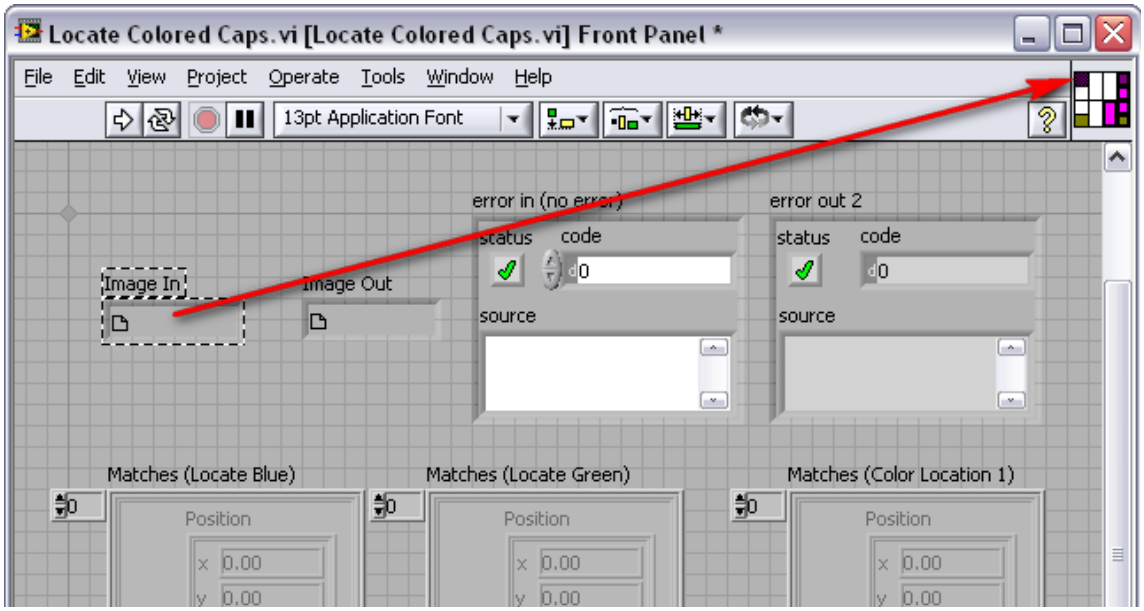
Once you have finished editing your VI so that it is no longer broken, you will still get errors when you run it because there is no valid image input to the *Image In* control. You will have to pass valid images from the axis camera to the *Image In* control before your VI will run correctly. To do this, you'll need to insert your VI into the cRIO robot code framework. Before we do this though, we need to make a couple more changes to get the VI ready to be used as a subVI:

1. Edit the connector pane to allow access to controls and indicators
2. Edit the icon so that we will be able to identify the VI on other block diagrams

Go to the front panel of the Locate Colored Caps VI, and right-click on the VI Icon, and select **Show Connector**.



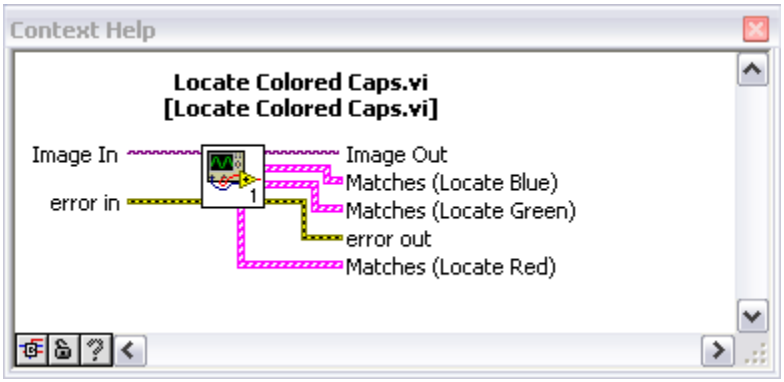
Each colored square (called a terminal) represents and input or output to a VI, just like the arguments of a function. You can click on a terminal, and the associated front panel object will gain focus. The top left terminal is associated with the *Image In* control.



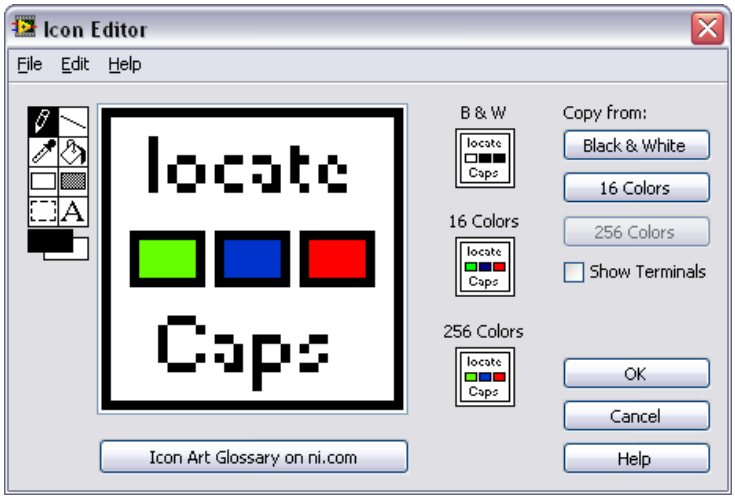
Make sure that the inputs and outputs that you want to access from your framework are connected. These will include:

- Image in
- Image out
- Error in
- Error out
- Results out

To associate a terminal with a front panel object, click on a blank (white) terminal and then click on the front panel object you want to associate with that terminal. The terminal will change color to represent the datatype of the object that you selected. Once you've finished associating the terminals, save the VI and then bring up the Context Help windows by going to **Help»Show Context Help** or by hitting **ctrl+h**. Now hover your mouse over the connector pane to see the terminals you just defined:



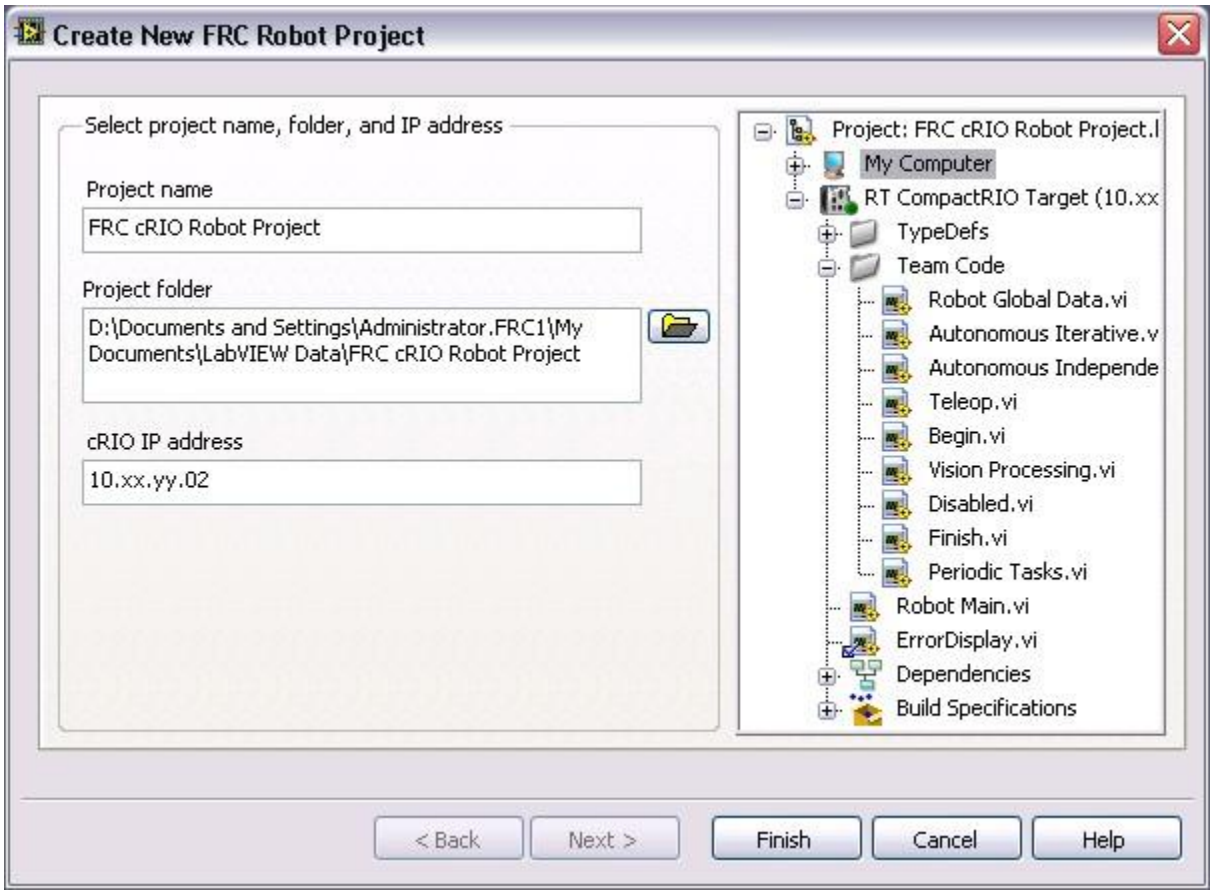
Now is also a good time to edit the VI icon so that you can recognize your VI in a larger application. Right-click on the icon and select **Edit Icon…**



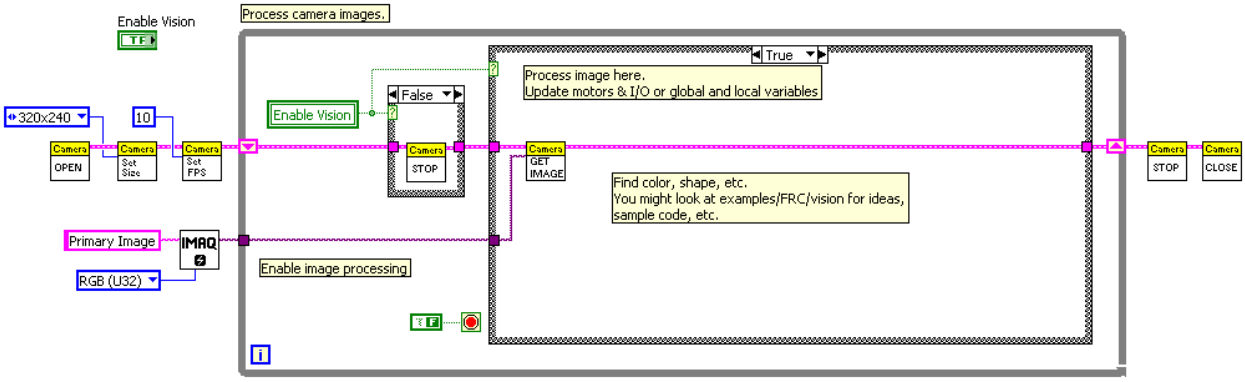
Once you finish editing the icon and arranging the front panel, save the VI.

Integration into a Framework

Now that you have a working VI that has been correctly configured for use as a subVI, it's time to integrate it into the FRC framework. Open a default FRC cRIO Robot Project, and make sure to configure it with the correct cRIO IP address based on your team number.



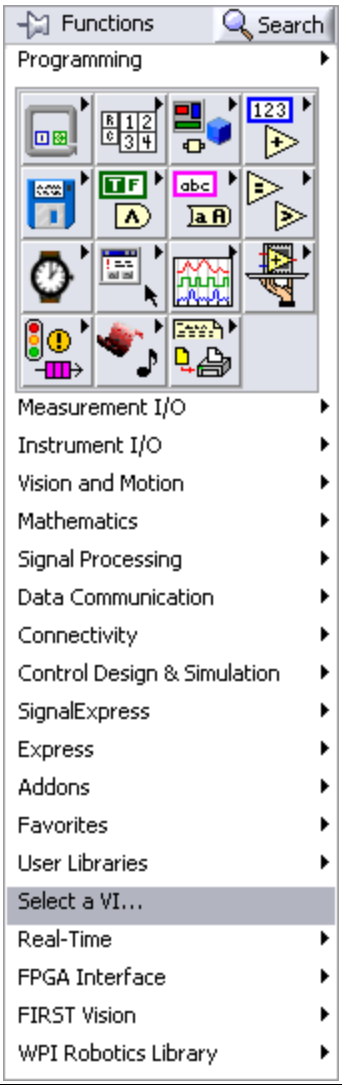
Open up the Robot Main VI, and find the image processing loop.

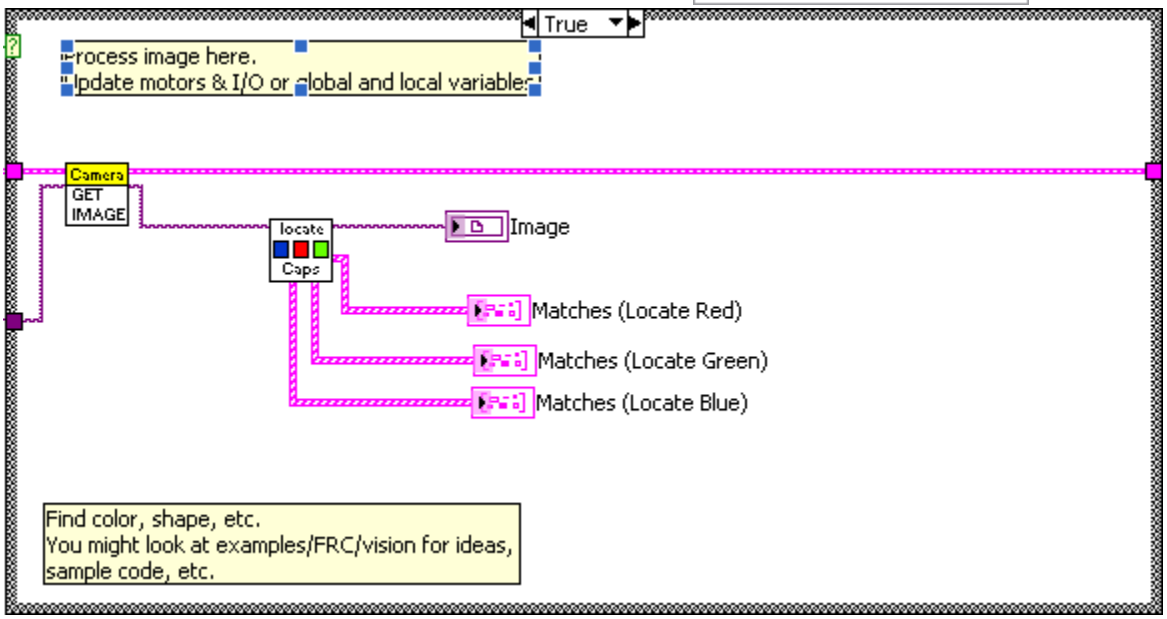


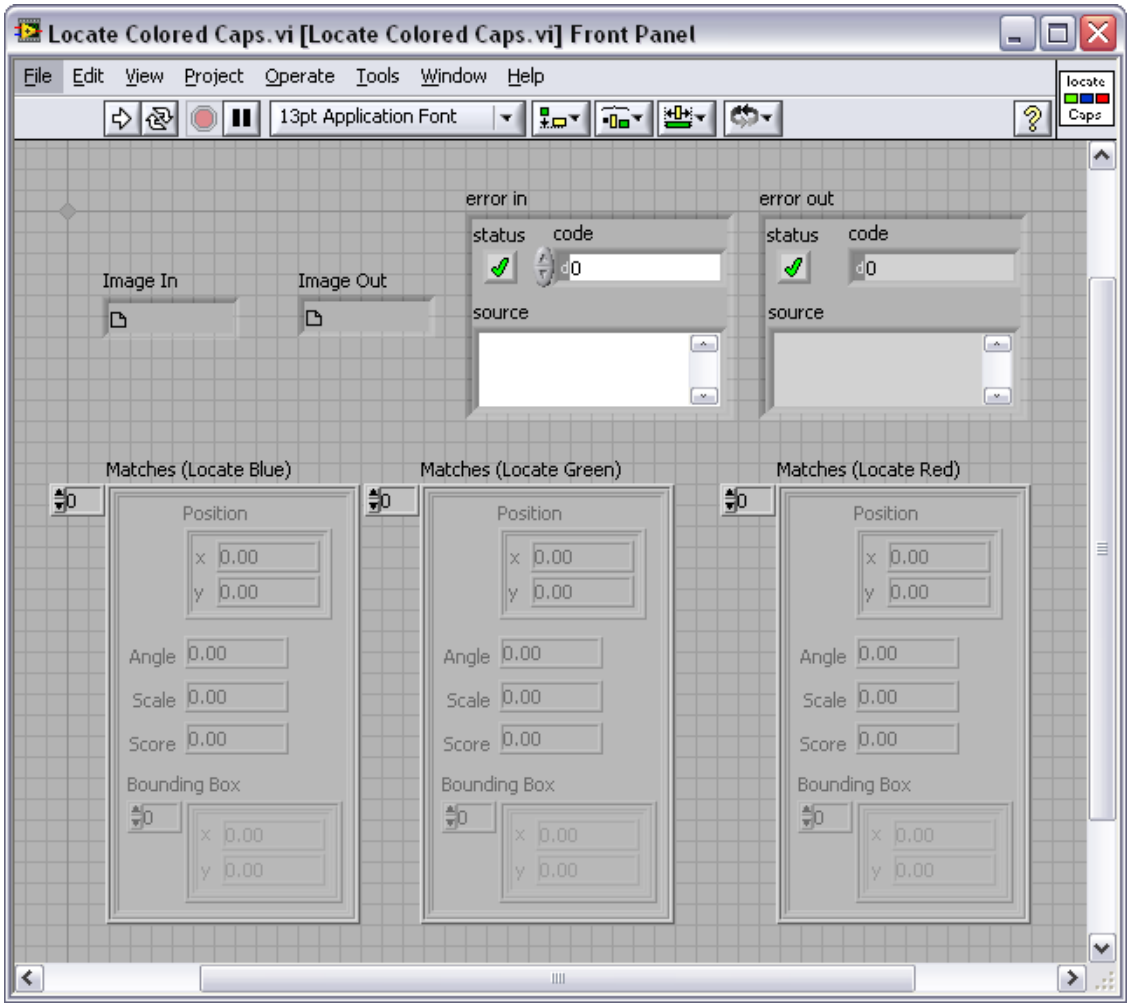
We will insert the Locate Colored Caps VI into the image processing loop, and pass it data from the Axis camera. For the purpose of this tutorial, we'll also add an image display. When testing your robot code, you'll want to use the Dashboard to view images instead of putting an image display in the code that will run on the cRIO.

Start by loading the Locate Colored Caps VI. Do this by using the **Select a VI…** option from the Functions palette. Wire the image output of the Get Image VI to the image input of the Locate Colored Caps VI. Place an Image Display on the front panel of the Basic Robot Main VI, and wire it to the image out of the Locate Colored Caps VI. Create indicators for each of the Match result clusters.

When you are finished, your code should look something like the following image:







One last thing that you will have to do to get this example working correctly on the cRIO is to make the template images that you saved back in Vision Assistant accessible to the code running on your cRIO. To do this you have to FTP the template images over to the cRIO and give your code the correct file paths to the new file locations. Follow the instructions in this document to learn how to FTP files to your cRIO:

FTP File Transfer to cRIO (Using Image Processing Requiring Template Images)

Now you should be all set to go ahead and test this out with Dry-Erase markers of your own! Does it work if you take images against a background other than white? Does it work with other red, blue, or green objects? How about if the lighting characteristics are different? How robust is the algorithm? You can adjust parameters by converting some of the constants in the Locate Colored Caps VI to controls, and then changing those control values, such as threshold value or templates, to work with your new objects.

Conclusion

Congratulations! You now know how to use the Vision Assistant to prototype machine vision algorithms, how to convert a Vision Assistant script to a VI, and how to insert that code as a subVI into a framework. You can use this same process to develop your own algorithms.

The method shown in this tutorial for finding colored objects is by no means the only, or even necessarily the best, way to search for and find any kind of colored object. There are many ways to search for, quantify, and track objects, and the task is not trivial. Perhaps performing a color threshold on one of the color planes and then employing some binary morphology (blob) analysis would be more robust for tracking applications, while the example given here is better suited for sorting applications in a more controlled environment. Feel free to explore the vision tutorials included with LabVIEW, as well as the example code provided on the FIRST website, and draw your own conclusions as to which methods will work best for you.

Sometimes it might even be more efficient to create a few different scripts, convert those scripts to VIs, and then merge the created code into one or multiple subVIs that you then insert into your framework. Remember: Vision Assistant is linear, while LabVIEW can process in parallel. Keep this in mind when leveraging the power of vision to build a better robot!

Additional Image Processing Resources

NI Vision Assistant Tutorial
NI Vision Concepts Manual
A Practical Guide to Machine Vision Lighting – Part I