

Cognex Designer Advanced – Accessing VPro Tools



COGNEX

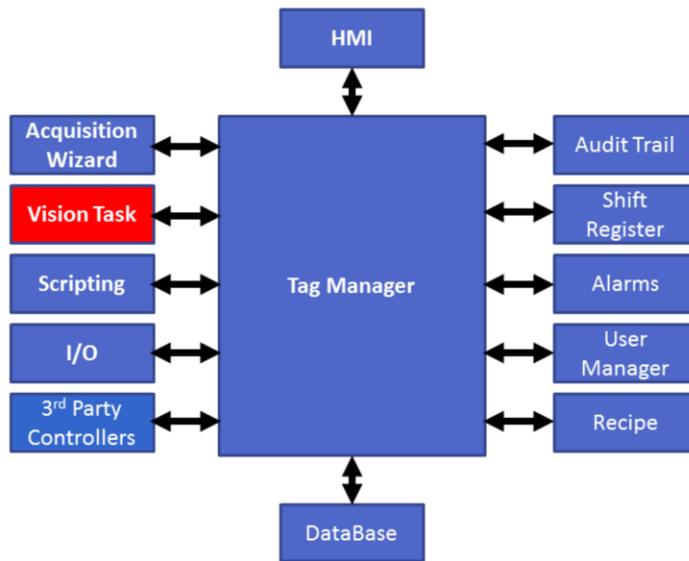
Objective

Cognex Designer Advanced –Accessing VisionPro Tools

- Explore different access methods to the tools
 - Understand what is needed to gain access to VisionPro tools
 - Learn how to bring a VisionPro function forth to the HMI
- ❖ Lab: Give user ability to retrain PatMax through the HMI

COGNEX

Accessing VisionPro Tools



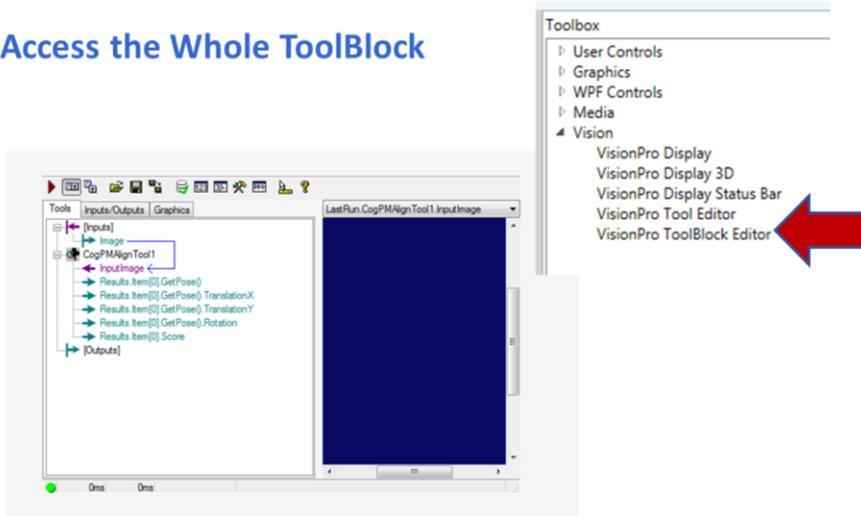
3 | © 2014 Cognex Corporation

COGNEX

In this section we will focus on gaining additional access to the VisionPro application.

Different Ways To Access Tools

Access the Whole ToolBlock



4 | © 2014 Cognex Corporation

COGNEX

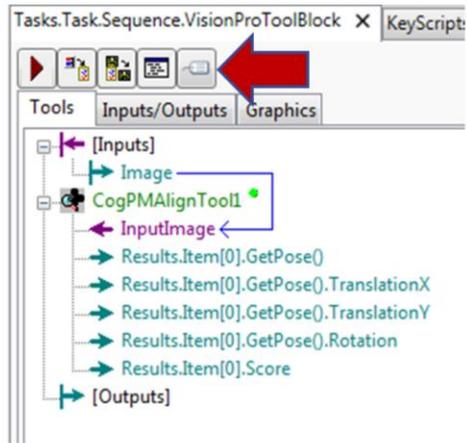
We can add a ToolBlock editor to our HMI that can allow our user to be able to get access to the VisionPro ToolBlock. The entire ToolBlock will go into the editor when the application is in Test Mode or deployed. This could even include adding new tools to the VisionPro application.

Note: Care must be taken to make sure that the VisionPro application is not running when accessing this ToolBlock object.

Measures should be taken to make sure the ToolBlock is not running when the user is accessing it. This may include putting it on a separate page that the means to go to it is user controlled as well as checking to see if the ToolBlock isRunning.

Creating Tags

Getting Access

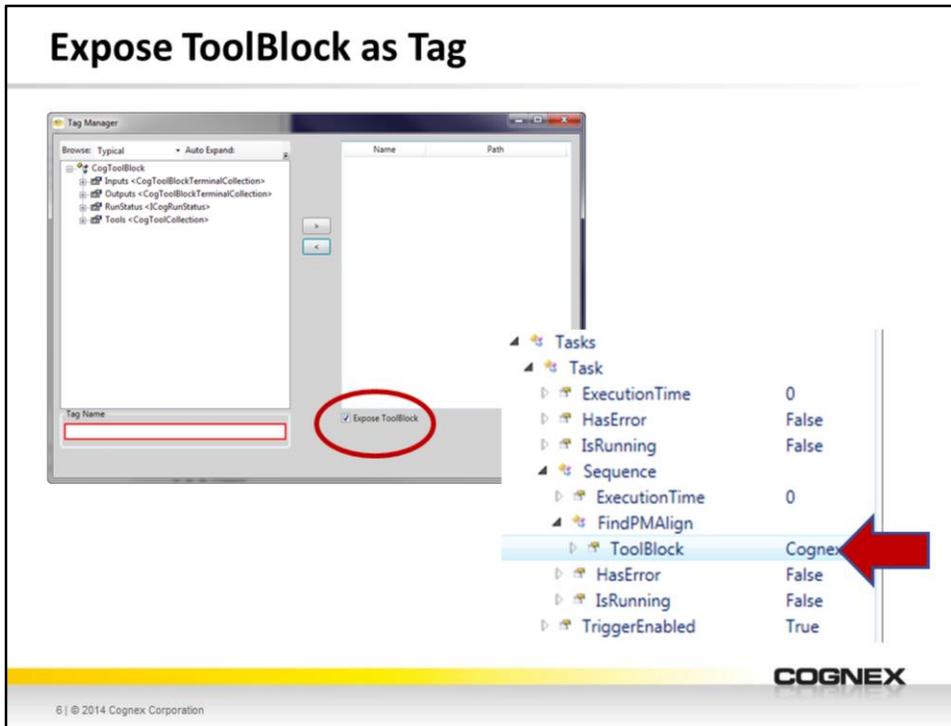


COGNEX

5 | © 2014 Cognex Corporation

You can get access to the tools within VisionPro by adding tags at the ToolBlock level in the Cognex Designer.

Expose ToolBlock as Tag

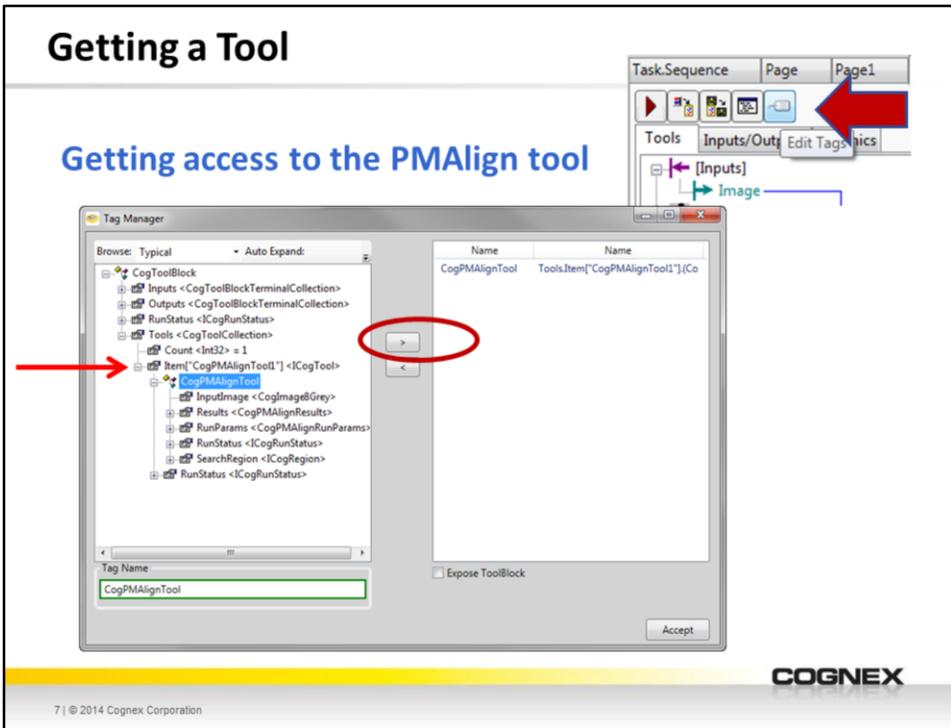


Click the “Expose ToolBlock” checkbox at the bottom of the right column.

Now the ToolBlock will show-up as an independent tag in the Tag Browser. This is what the Subject Property of the ToolBlockEditor will reference when bringing the ToolBlock to the HMI.

Getting a Tool

Getting access to the PMAIalign tool



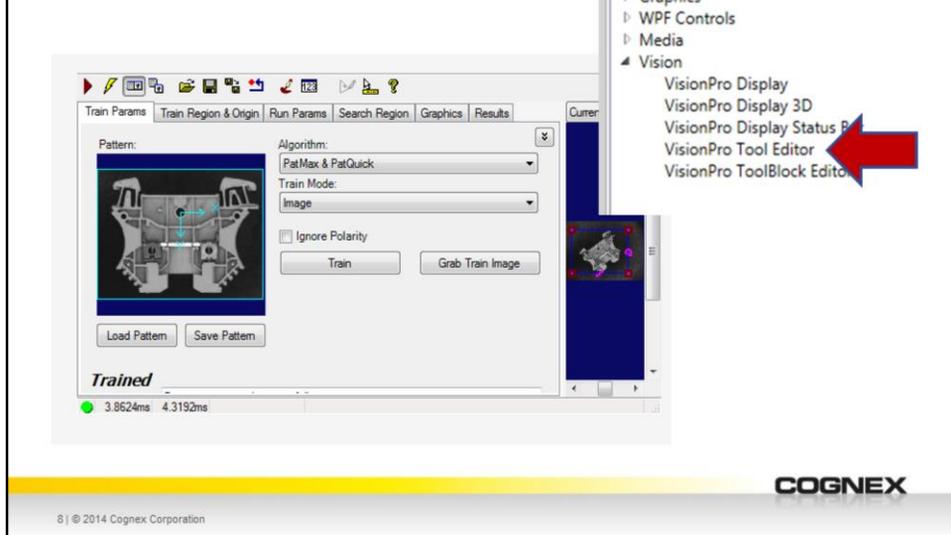
In our designer application, we want to get access to the Search region of the PMAIalign tool. In order to do so, we need to bring out the actual tool so we can gain access to the graphics area.

Go to the EditTags button and find the tool to be brought out to the HMI. Press the right pointing arrow to add the tool to the list of tags.

Note: Make sure you grab the Tool and not the Item (which holds more than just the tool).

Access to a Tool

Ability to tweak a Tool



The tool is now a Tag in the application so it can be assigned to the Subject Property in the ToolEditor.

This tool differs from the ToolBlockEditor in the sense that it brings access to a single tool out

Custom Access

Add Display and Button

- Display will reference the Input Image from the ToolBlock



COGNEX

9 | © 2014 Cognex Corporation

The display subject should reference the InputImage generated from the ToolBlock. It may look like:

```
$Tasks.Task.Sequence.FindPMAAlign.LastRun.CogPMAAlignTool1.InputImage
```

Accessing the Search Region

Script for Button's Mouse Click



```
Task.Sequence Page Button.OnClick X
public static void OnClick(System.Object sender, System.EventArgs e)
1 var display = $System.GetInternalObject("Pages.Page.ImageDisplay") as Cognex.VisionPro.Display.CogDisplay;
2 var rectangle = $Tasks.Task.Sequence.FindPMAAlign.CogPMAAlignTool.SearchRegion as CogRectangleAffine;
3
4 display.InteractiveGraphics.Add(rectangle, "searchRegion", false);
5
```

COGNEX

10 | © 2014 Cognex Corporation

This is the script that is needed to get access to the Search region

```
var display =
$System.GetInternalObject("Pages.Page.ImageDisplay") as Cognex.VisionPro.
Display.CogDisplay;
var rectangle =
$Tasks.Task.Sequence.FindPMAAlign.CogPMAAlignTool.SearchRegion as CogRect
angleAffine;

display.InteractiveGraphics.Add(rectangle, "searchRegion", false);
```

The first line is the magic. It allows the Cognex Designer to get access to the VisionPro - in this case the display. It can also be used to get access to devices and the communication card.

Reference DLLs

Make sure the Correct DLLs are referenced as well as the Using Directive

The screenshot displays two windows from Visual Studio. On the left is the 'Properties' window, showing the 'References' section under the 'Common' category. It lists two references: 'Cognex.VisionPro' and 'Cognex.VisionPro.Display'. On the right is the 'References' window, which is divided into two sections. The top section, titled 'Name', lists 'Cognex.VisionPro.dll' and 'Cognex.VisionPro.Core.dll', with the latter highlighted in blue. The bottom section, titled 'Local References', lists 'Cognex.VisionPro.Display.Controls.dll', 'Cognex.VisionPro.PMAlign.dll', 'System.Drawing.dll', and 'System.Windows.Forms.dll'.

COGNEX

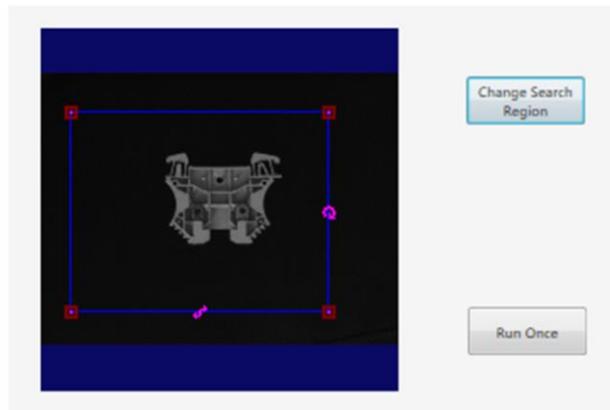
11 | © 2014 Cognex Corporation

The DLLs for the VisionPro Display Controls as well as the PMAlign need to be referenced. We also need to reference the System Drawing and Windows Forms DLLs.

We also have to include Cognex.VisionPro as a Using Directive so that the application knows how to get access to the CogRectangleAffine.

Change Search Region

Access to the Search Region



12 | © 2014 Cognex Corporation

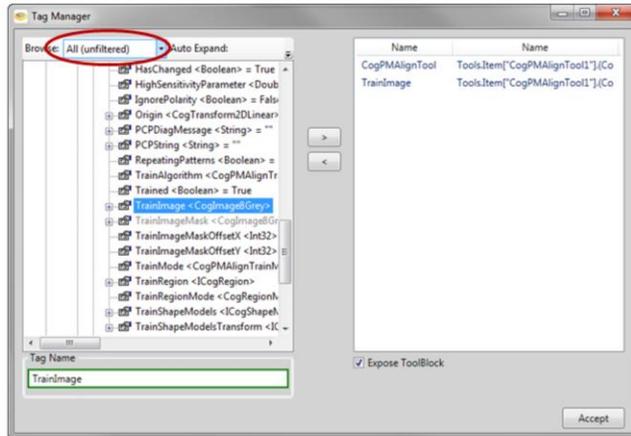
COGNEX

Now when we click on the Change Search Region button, we have access to the search region within the VisionPro PMAAlign tool.

Another button was added to run the application so that we can make sure it works as expected.

Access to Retraining PatMax

Add the TrainImage



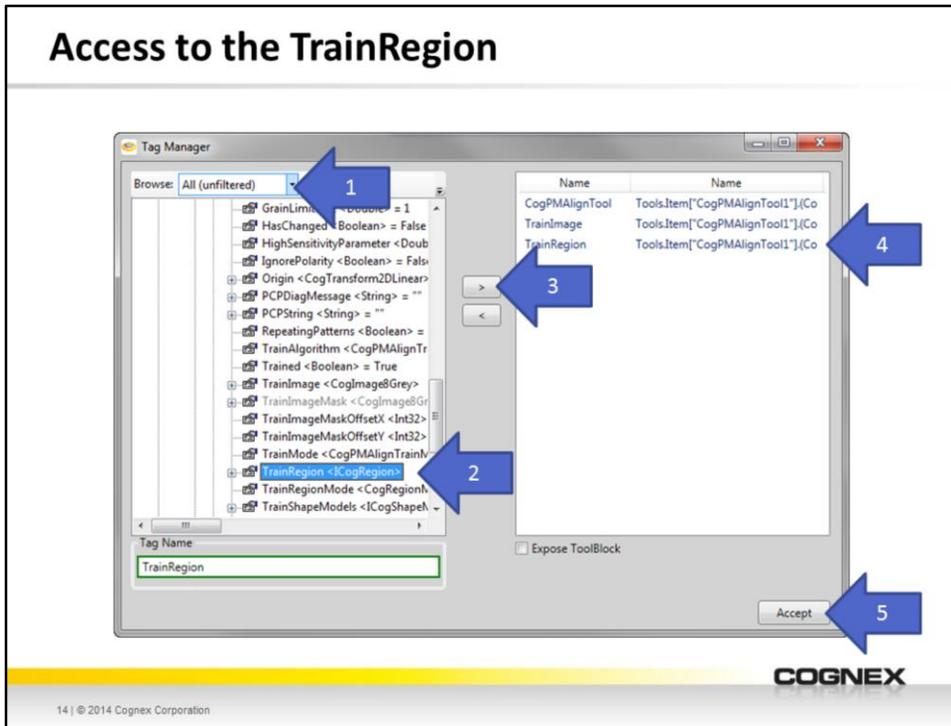
13 | © 2014 Cognex Corporation

COGNEX

Now we have to add the TrainImage found under the PMAAlignTool1 -> Pattern. You will need to Browse with All (unfiltered) to be able to get access to this object.

We need the TrainImage to be able to set the new training region and TrainRegion.

Access to the TrainRegion

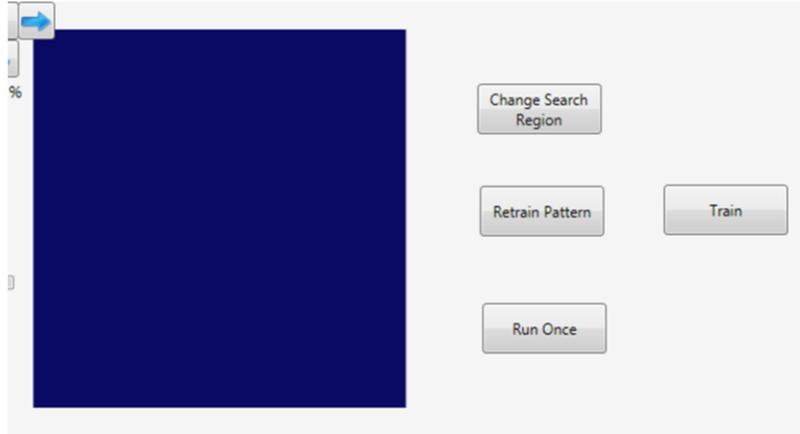


In the same area, we need to get to the Train Region so we can interact with it.

- 1) Select "All (unfiltered)" from the browse pull-down.
- 2) Navigate to and select
CogToolBlock → Tools → Item["CogPMAAlignTool1"] → CogPMAAlignTool → Pattern → TrainRegion
- 3) Make it accessible by adding it to the right column using the right arrow button
- 4) Verify it has been added to the right column
- 5) Click Accept to confirm changes

Adjust HMI

Add two more buttons



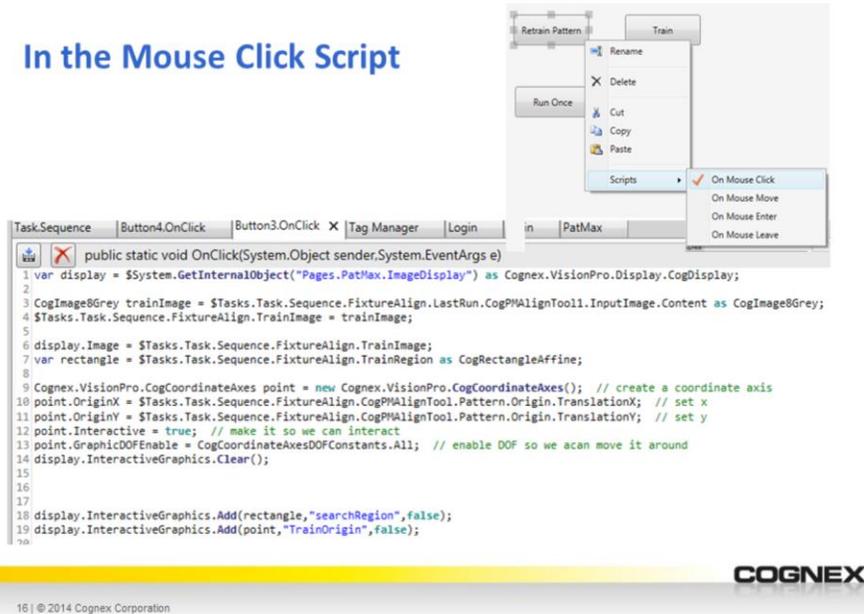
COGNEX

15 | © 2014 Cognex Corporation

Two more buttons need to be added. One will allow us to gain access to the training region like we did for the search region. The other will retrain the PatMax pattern

Script for the Change Train Region

In the Mouse Click Script



```
public static void OnClick(System.Object sender, System.EventArgs e)
1 var display = $System.GetInternalObject("Pages.PatMax.ImageDisplay") as Cognex.VisionPro.Display.CogDisplay;
2
3 CogImage8Grey trainImage = $Tasks.Task.Sequence.FixtureAlign.LastRun.CogPMAAlignTool1.InputImage.Content as CogImage8Grey;
4 $Tasks.Task.Sequence.FixtureAlign.TrainImage = trainImage;
5
6 display.Image = $Tasks.Task.Sequence.FixtureAlign.TrainImage;
7 var rectangle = $Tasks.Task.Sequence.FixtureAlign.TrainRegion as CogRectangleAffine;
8
9 Cognex.VisionPro.CogCoordinateAxes point = new Cognex.VisionPro.CogCoordinateAxes(); // create a coordinate axis
10 point.OriginX = $Tasks.Task.Sequence.FixtureAlign.CogPMAAlignTool.Pattern.Origin.TranslationX; // set x
11 point.OriginY = $Tasks.Task.Sequence.FixtureAlign.CogPMAAlignTool.Pattern.Origin.TranslationY; // set y
12 point.Interactive = true; // make it so we can interact
13 point.GraphicDOFEnable = CogCoordinateAxesDOFConstants.All; // enable DOF so we acan move it around
14 display.InteractiveGraphics.Clear();
15
16
17
18 display.InteractiveGraphics.Add(rectangle, "searchRegion", false);
19 display.InteractiveGraphics.Add(point, "TrainOrigin", false);
20
```

COGNEX

16 | © 2014 Cognex Corporation

The Train region is somewhat similar to the Search Region though the twist is getting the current image in – remember the “Grab Train Image” button in the PatMax tool? That’s what we need to do. Here is the script:

```
var display =
$System.GetInternalObject("Pages.PatMax.ImageDisplay") as Cognex.VisionPro.Display.CogDisplay;
```

```
CogImage8Grey trainImage =
$Tasks.Task.Sequence.FixtureAlign.LastRun.CogPMAAlignTool1.InputImage.Content as CogImage8Grey;
$Tasks.Task.Sequence.FixtureAlign.TrainImage = trainImage;
```

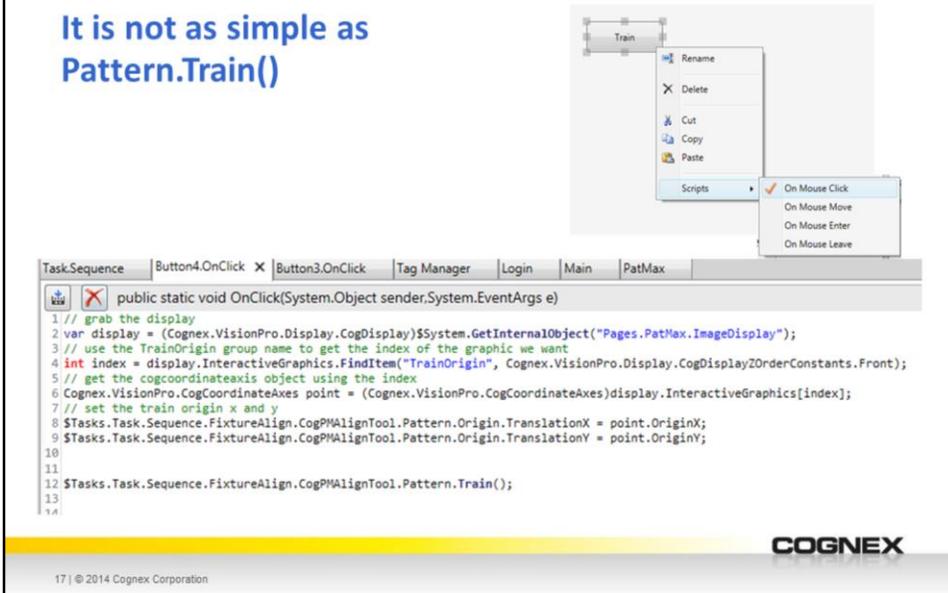
```
display.Image = $Tasks.Task.Sequence.FixtureAlign.TrainImage;
var rectangle = $Tasks.Task.Sequence.FixtureAlign.TrainRegion as CogRectangleAffine;
```

```
Cognex.VisionPro.CogCoordinateAxes point
= new Cognex.VisionPro.CogCoordinateAxes(); // create a coordinate axis
point.OriginX =
$Tasks.Task.Sequence.FixtureAlign.CogPMAAlignTool.Pattern.Origin.TranslationX; // set x
point.OriginY =
$Tasks.Task.Sequence.FixtureAlign.CogPMAAlignTool.Pattern.Origin.TranslationY; // set y
point.Interactive = true; // make it so we can interact
point.GraphicDOFEnable = CogCoordinateAxesDOFConstants.All; // enable DOF so we acan
move it around
display.InteractiveGraphics.Clear();
```

```
display.InteractiveGraphics.Add(rectangle,"searchRegion",false);  
display.InteractiveGraphics.Add(point,"TrainOrigin",false);
```

Train Button

It is not as simple as
`Pattern.Train()`



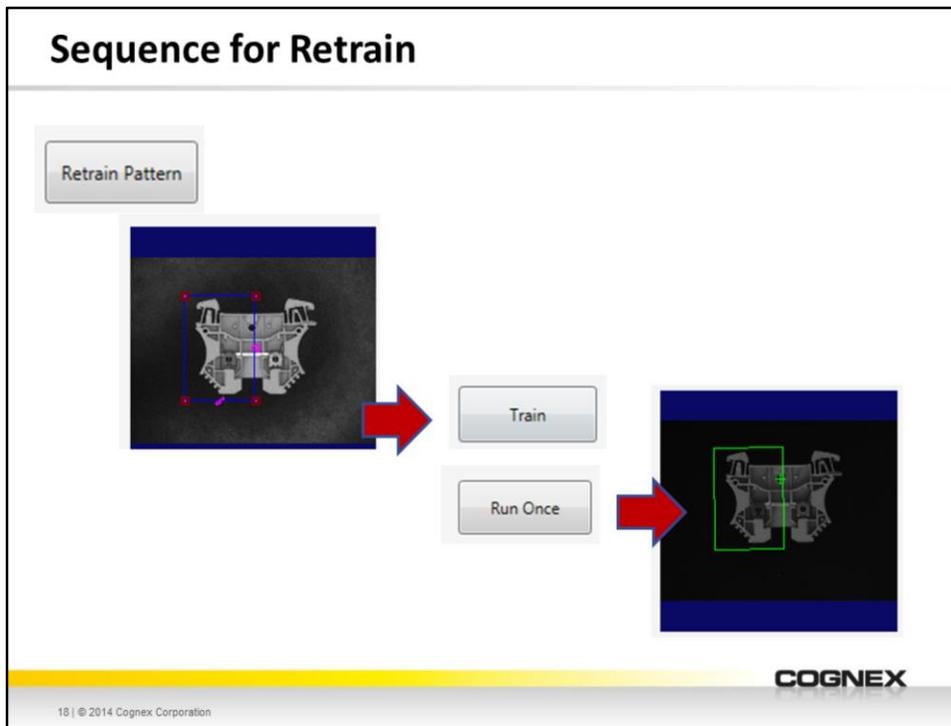
The screenshot shows a software interface with a context menu open over a 'Train' button. The menu options are: Rename, Delete, Cut, Copy, Paste, and Scripts. The 'Scripts' submenu is expanded, showing: On Mouse Click (checked), On Mouse Move, On Mouse Enter, and On Mouse Leave. Below the menu is a code editor window with the following C# code:

```
public static void OnClick(System.Object sender, System.EventArgs e)
1 // grab the display
2 var display = (Cognex.VisionPro.Display.CogDisplay)$System.GetInternalObject("Pages.PatMax.ImageDisplay");
3 // use the TrainOrigin group name to get the index of the graphic we want
4 int index = display.InteractiveGraphics.FindItem("TrainOrigin", Cognex.VisionPro.Display.CogDisplayZOrderConstants.Front);
5 // get the cogcoordinateaxis object using the index
6 Cognex.VisionPro.CogCoordinateAxes point = (Cognex.VisionPro.CogCoordinateAxes)display.InteractiveGraphics[index];
7 // set the train origin x and y
8 $Tasks.Task.Sequence.FixtureAlign.CogPMAAlignTool.Pattern.Origin.TranslationX = point.OriginX;
9 $Tasks.Task.Sequence.FixtureAlign.CogPMAAlignTool.Pattern.Origin.TranslationY = point.OriginY;
10
11
12 $Tasks.Task.Sequence.FixtureAlign.CogPMAAlignTool.Pattern.Train();
13
14
```

We need to get the data from the Axis after the dragging is stopped and then retrain with that information:

```
// grab the display
var display
= (Cognex.VisionPro.Display.CogDisplay)$System.GetInternalObject("Pages.PatMax.ImageDisplay");
// use the TrainOrigin group name to get the index of the graphic we want
int index =
display.InteractiveGraphics.FindItem("TrainOrigin", Cognex.VisionPro.Display.CogDisplayZOrderConstants.Front);
// get the cogcoordinateaxis object using the index
Cognex.VisionPro.CogCoordinateAxes point
= (Cognex.VisionPro.CogCoordinateAxes)display.InteractiveGraphics[index];
// set the train origin x and y
$Tasks.Task.Sequence.FixtureAlign.CogPMAAlignTool.Pattern.Origin.TranslationX = point.OriginX;
$Tasks.Task.Sequence.FixtureAlign.CogPMAAlignTool.Pattern.Origin.TranslationY = point.OriginY;

$Tasks.Task.Sequence.FixtureAlign.CogPMAAlignTool.Pattern.Train();
```



When we go into Test mode, we can retrain the pattern by going through the following steps:

- Press the Retrain Pattern button to get access to the training region
- Set the training region where we want it
- Press the Train button to retrain PatMax with the new training region
- Press the Run Once button to see if it works

If the pattern successfully retrained, we should see a new output indicative of the new region in our display.

Summary

Accessing VisionPro Tools

- Understood different access methods to the tools
- Learned what is needed to gain access to VisionPro tools
- Explored how to bring a VisionPro function forth to the HMI

COGNEX