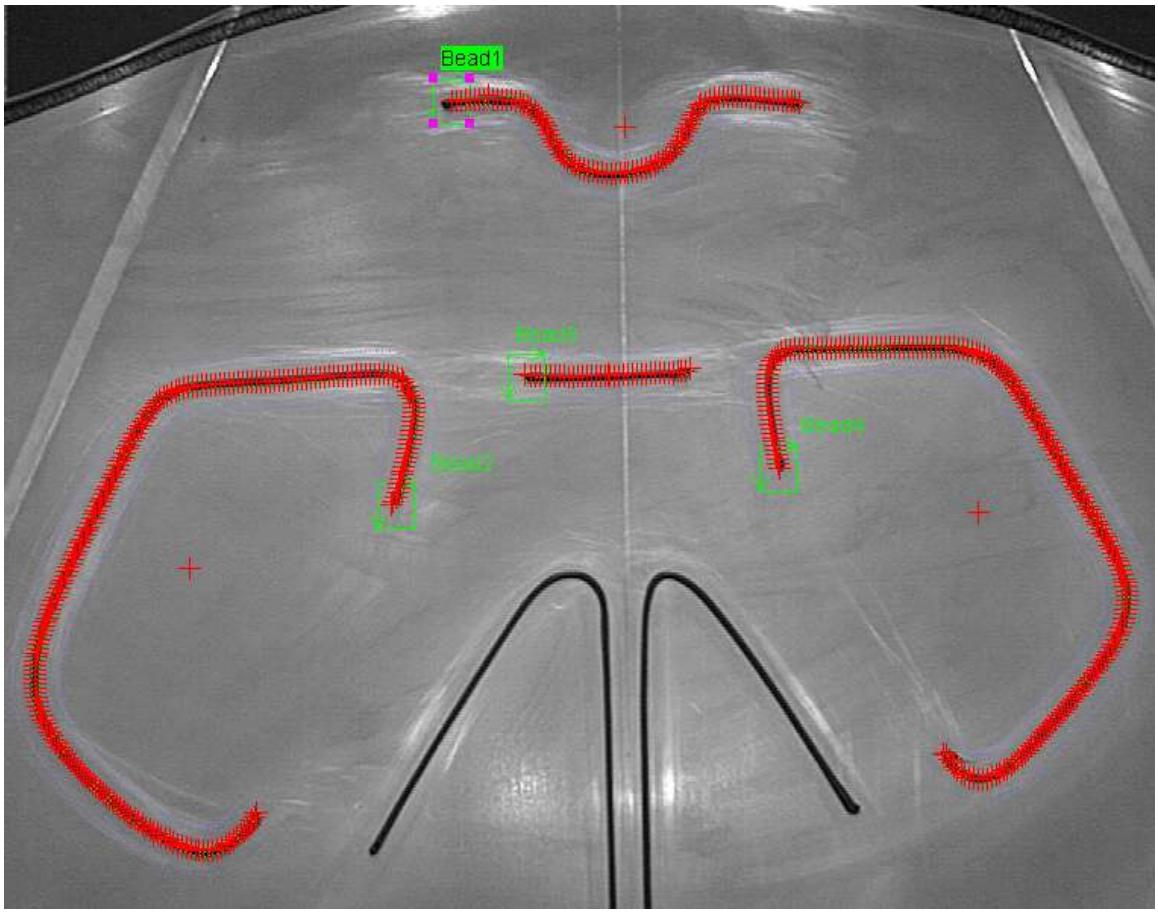
Sherlock 7 Technical Resource

Teledyne DALSA Incorporated
Industrial Products

Ben Dawson
Document Revision: 10 August 2012

Copyright © 2012, Teledyne DALSA, Inc.
ALL RIGHTS RESERVED

BEAD TOOL ALGORITHM



Inspecting glue beads on truck hoods (bonnets) using Bead Tool

Bead Tool [algorithm]

The Bead Tool algorithm inspects a bead (thin line) of material. A typical application is inspecting beads of glue that attach liner material to a car or truck chassis. Here is a bead of adhesive (dark material) that has been dispensed onto a metal part:

The bead may vary in width, might be larger at the start or end, and there might be some smearing or feathering of the bead material.

A bead has a specific spatial pattern (x,y positions) that we call the "path". The Bead Tool learns a bead path and the thicknesses at each point along the path. A bead is inspected by comparing its path and thicknesses with the learned bead's path and thicknesses.

Physical Setup

Arrange the camera, lighting, and optics so that the bead is clearly seen against the background and that distracting texture, shadows, and reflections are minimized. The bead "color" can be dark ("black") on a lighter background or light ("white") on a darker background.

The bead should be positioned in the camera's field of view such that all parts of the bead are well away from the image edges and that the bead thickness is always 4 or more pixels.

When possible, position parts so the bead path always has the same starting position and orientation. When this is not possible, you could use a search algorithm to align the position and orientation of the inspected bead with the learned bead path.

Most glue fluoresces under ultraviolet (UV) light, so you may be able to get a good bead image by illuminating the part with UV light in an enclosure that blocks ambient light. Some glue manufactures add a fluorescent material to their glue to make them more visible under UV light. Some countries ban the use of these fluorescence enhancers.

Bead Tool Setup and Annotation

1. Create a rectangle ROI (only Rectangle ROIs are allowed) and place it around the start of the bead. The width of the rectangle should be a little larger than the largest diameter of the bead. The largest diameter of a bead is often at the start or end of the bead. Make the rectangle width about the same size as the height, so you have close to a square ROI. The side of the ROI from which the bead exits defines the area searched for the bead, so smaller sizes make learning and inspection faster and can prevent the bead tool from "jumping" to other image features, but at the risk of losing the bead when it turns. You can see the search area using ***search limits*** annotation.

2. Rotate the ROI rectangle (square) such that one side is approximately perpendicular to the bead path at the start of the bead, say within 5 degrees. For example:

3. Estimate the thickness of the bead where it is thinnest. You can use the mouse or a caliper tool to estimate the bead thickness at the thinnest point. This thinnest point must be 4 or more pixels, as detecting thinner bead widths is unreliable. In the above image, a caliper at LineA was used to estimate the bead thickness. This value will be used for the ***bead thickness*** parameter.

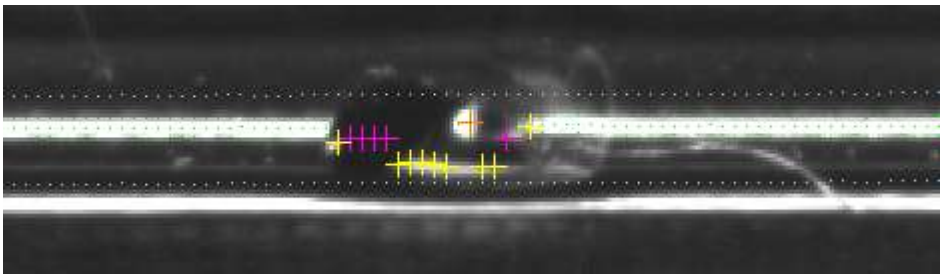
4. Add the Bead Tool algorithm to this ROI and make sure ***operating mode*** is "setup parameters" (the default when the algorithm is added). In this mode, you can set parameters without the Bead Tool running each time you change a parameter, making setup faster and easier

5. You can set Annotate parameters to provide diagnostic annotations:
   - ***search limits*** – Marks the limits of the area searched for the bead path. The default is to mark the limits with light blue dots (see images following).
   - ***path*** – Marks the bead path edges and center points. The default is to mark path points with green dots.
   - ***too thick points*** – Mark points on the bead path where the bead is thicker than the ***max thickness limit*** (during learning) or the ***max thickness tolerance*** (during inspection). The default is to mark these points with red crosses.
   - ***too thin points*** – Mark points on the bead path where the bead is thinner than the ***min thickness limit*** (during learning) or the ***min thickness tolerance*** (during inspection). The default is to mark these points with yellow crosses.

- ***missing points*** – Mark points on the bead path where there is a gap in the bead. The default is to mark these points with magenta crosses.
- ***ignored points*** – Mark points that have been removed from the inspection measures because they are part of a short "run" of bead defects. See ***ignore defect runs*** <= parameter.

We recommend that you turn off the default annotations (red crosses) for learned and inspected point Outputs (Readings). Do this by double-clicking on the "learned path pts." Output and in the "edit output reading" window, setting the value in "Display in image window:" to "[None]" and then clicking "OK". Do the same for the "inspect path pts." Output. This greatly reduces the visual clutter. The annotation provide in Bead Tool is a lot easier to see. The cover image on this document shows bead paths with the default annotations enabled.

Here is an example of a bead inspection with the Output points turned off (the default red crosses). The search limits, set by the bead exit side of the starting rectangle (not shown), are shown by light blue dots. The found center and edges of the inspected bead path are shown by light green dots. Missing points are shown in magenta, points that are too thin in yellow and points that are too thick in orange. Other Output points, such as the "inspect path centroid" have been turned off.



Once Bead Tool has been set up and is working in your inspection, turn off the annotations (select "no draw" for the color) to reduce the inspection cycle time.

Bead Path Detection

The "Detect" group of parameters set up the detection of the bead path within the image. They should be set before the algorithm learns the bead path ("learn bead path"), but can be changed at run time (in "inspect bead" mode). This allows you to adjust the bead detection at run time, perhaps for changes in bead contrast due to lighting changes, etc.

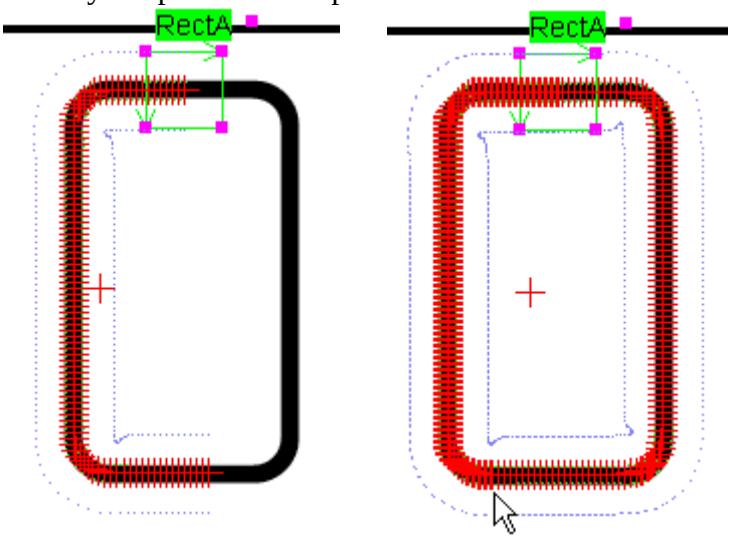
- ***bead color*** can be set to "black" or "white". "black" detects a dark bead on a lighter background, and "white" detects a lighter bead on a dark background.
- ***bead thickness*** is the approximate thickness of the bead, as measured in Step 3 of the previous section. This parameter's range is 1 to 128 pixels, with a default of 16 pixels. Adjust ***bead thickness*** for best results. Larger ***bead thickness*** values use more image values to detect bead path edges and so can find fainter and fuzzier beads, but larger values might cause the bead to "jump" to neighboring image structure that have stronger contrast.
- ***minimum contrast*** sets the minimum contrast a bead needs to have with respect to its background in order to be detected. The bead path will stop being learned

or inspected if the bead contrast falls below this value. During inspection, points on the bead below this contrast are marked as "missing". This parameter ranges from 1 to 511 with a default of 50. Adjust for best results: Higher contrast values reduce incorrect detection of lower contrast image structure near the bead path, while lower contrast values allow the Bead Tool to follow the bead path into shadows.

Bead Path Learning

1. The "Learn" group of parameters are used only for learning and changing them during inspection is not recommended as they can disrupt the inspection.

   - ***starting direction*** is the direction that the bead path leaves the starting ROI rectangle. The choices are: "right", "left", "up" or "down".
   - ***number of steps*** is the number of steps the Bead Tool will take along the bead path while learning the bead path. This parameter's range is 1 to 32,000 with a default of 100. The Bead Tool steps by 3 pixels along the bead path so the range is 3 to 96,000 pixels. You should set ***number of steps*** to approximately 1/3 the length (in pixels) of the bead path you want to inspect. During learning the Bead Tool will learn the bead path until either the contrast of the bead falls below ***minimum contrast*** or it uses up the ***number of steps***. The following example on the left shows the case where ***number of steps*** is too small, so that the bead path is not learned all the way around the bead. On the right, the ***number of steps*** is too large, so that some points on the bead are learned twice (the path stops at the arrow). If the bead is not a loop, then you can set ***number of steps*** to a large number and only the points on the path will be learned.



   - ***min thickness limit*** and ***max thickness limit*** set the width limits for learning points along the bead path. Bead path locations that are outside these limits are learned but marked as "failed". Adjust these limits so that all the points you want to inspect are within these limits. Check the "learned too thin count" and "learned too thick count" outputs (readings) for correct values. These limits can be used to suppress "smears" and "strings" on the bead. These parameters range from 0.1 to 1024.0 pixels, with a default of 5.5 and 16.4 pixels, respectively. NOTE: These limits are not ***search limits*** (shown by default as blue dots). ***search limits*** define

the area in which the Bead Tool searches for bead edge points and these limits are set by the dimension of the side of the ROI that the Bead Tool exits from.

2. After you have set the parameters (approximately), switch ***operating mode*** to "learn bead path" and this will run the algorithm and learn the bead path. You will see the learned bead path as red crosses (if you have the standard output annotation enabled) or as dots if using the ***path*** option. Adjust the "Detect" and "Learn" parameters until you are satisfied with the bead path that is learned. Each time you make a parameter change, the Bead Tool re-learns the path. If you need to make many adjustments, it can be faster to set ***operating mode*** back to "setup parameters" so that the Bead Tool doesn't take time to relearn the bead path for each parameter change. Then set ***operating mode*** to "learn bead path" to learn the path.

3. When you save the program (Investigation), the bead path and widths are saved as a file of type .bed. This is a text file so you can look at it, but don't edit it as changes can cause the Bead Tool to fail.

4. Set the ***operating mode*** back to "set up parameters" to set the inspection parameters.

Inspecting the Bead Path

1. Once you are satisfied with the result of learning the bead path, set the ***min thickness tolerance*** and ***max thickness tolerance*** parameters in the "Inspect" parameters section. These parameters range from 0.0 to 1024.0 pixels with a default of 1.5 pixels. Unlike the thickness <u>limits</u> set in the "Learn" parameters, these <u>tolerances</u> set the amount that the inspected bead is allowed to vary in width from the widths learned during the learning of the bead path. For example if the ***min thickness tolerance*** is 1.5 pixels, and the learned width at a point along the bead path is 22 pixels, then the bead thickness can be no less than 22 – 1.5 = 20.5 pixels in width at that point to be accepted as good. Similarly, a ***max thickness tolerance*** of 2.0 pixels would allow bead thicknesses of up to 22 + 2 = 24 pixels before rejecting the bead point as defective. You can adjust these parameters in any ***operating mode***, but they only are applied when the ***operating mode*** is "inspect bead".

2. The ***ignore defect runs <=*** parameter sets the length of "runs" of defective bead points to ignore when taking measures such as average thickness. A "run" is a sequence of bead points (not pixels) that are defective: missing, outside of the image, too thin, or too thick. The default value of 0 keeps runs of any lengths. For example, setting this parameter to 2 will cause runs of 1 or 2 defective bead points to be marked as "ignored" and so not included in the summary statistics.

3. Now switch ***operating mode*** to "inspect bead". Sherlock automatically inspects the bead and reports points that match the learned path and thickness as accepted. Points outside the learned path or outside the thickness tolerances are marked as bad. See the next section for details on output (readings).

Outputs from Bead Tool

Bead Tool's Outputs (readings) can be grouped into:

- *Learned Outputs* – Output values from learning the bead path
- *Inspect Outputs* – Output values from inspecting the bead path
- *Defect Detail Outputs* – Additional details on the inspected bead path defects

Learned Outputs

- "learned num. pts." [Double] – Number of points learned along the bead path
- "learned avg. thickness" [Double] – Average thickness of learned bead
- "learned std. dev. thickness" [Double] – Standard deviation of bead path thicknesses
- "learned min thickness" [Double] – Minimum thickness of bead in learned path
- "learned min thickness pt." [Point] – Location of minimum thickness point on path
- "learned max thickness" [Double] – Maximum thickness of bead in learned path
- "learned max thickness pt." [Point] – Location of maximum thickness point on path
- "learned path pts." [Point array] – Learned path points
- "learned thicknesses" [Double array] – Thicknesses (in pixels) of learned path points
- "learned path centroid" [Point] – Centroid (center of gravity) of learned path points
- "learned path start" [Point] – Starting point of learned bead path
- "learned path end" [Point] – End point of learned bead path
- "learned too thin count" [Double] – Number of points less than min. thickness limit
- "learned too thick count" [Double] – Number of points larger than max. thickness limit
- "learned num pts. failed" [Double] – Number of point widths to thick or thin

Inspect Outputs

The bead path inspection classifies each bead path point into 1 of 6 categories:

- Ignored – This bead point was marked by the ignore defect runs <= parameter
- Valid – This bead point exists and was within specified tolerances
- Outside – This bead point was outside the Image Window (remember that this algorithm starts in the ROI but uses the entire Image Window).
- Missing – This bead point was missing
- Too Thin – This bead point's thickness was below the specified thickness tolerance
- Too Thick – This bead point's thickness was above the specified thickness tolerance

The following Inspect Outputs use only bead points classified as Valid, Too Thin or Too Thick:

- "inspect avg. thickness" [Double] – Average thickness of inspected bead
- "inspect std dev thickness" [Double] – Standard deviation of bead path thicknesses
- "inspect min thickness" [Double] – Minimum thickness of bead in inspected path
- "inspect min thickness pt." [Point] – Location of minimum thickness point on path
- "inspect max thickness" [Double] – Maximum thickness of bead in inspected path
- "inspect max thickness pt." [Point] – Location of maximum thickness point on path
- "inspect path centroid" [Point] – Centroid (center of gravity) of inspected path points

- "inspect num pts. below tol." [Double] – Number of point widths below min. tolerance. Ignored, missing, and off image points are not included in this number.
- "inspect num pts. above tol." [Double] – Number of point widths above max. tolerance. Ignored, missing and off image points are not included in this number.

The following outputs use all the points in the inspected bead path:

- "inspect num pts." [Double] – Number of points inspected
- "inspect path pts." [Point array] – Inspected path points
- "inspect thicknesses" [Double array] – Thicknesses (in pixels) of inspected path points
- "inspect path start" [Point] – Starting point of inspected bead path (uses all points)
- "inspect path end" [Point] – End point of inspected bead path (uses all points)

This Output counts all failures, but does not include Ignored points:

- "inspect num pts. failed" [Double] – Number of point whose width was below tolerance (too thin), above tolerance (too thick), missing, or off the image. Ignored points are not included in this number.

Defect Details Outputs:

These outputs provide detailed information about defects in the bead.

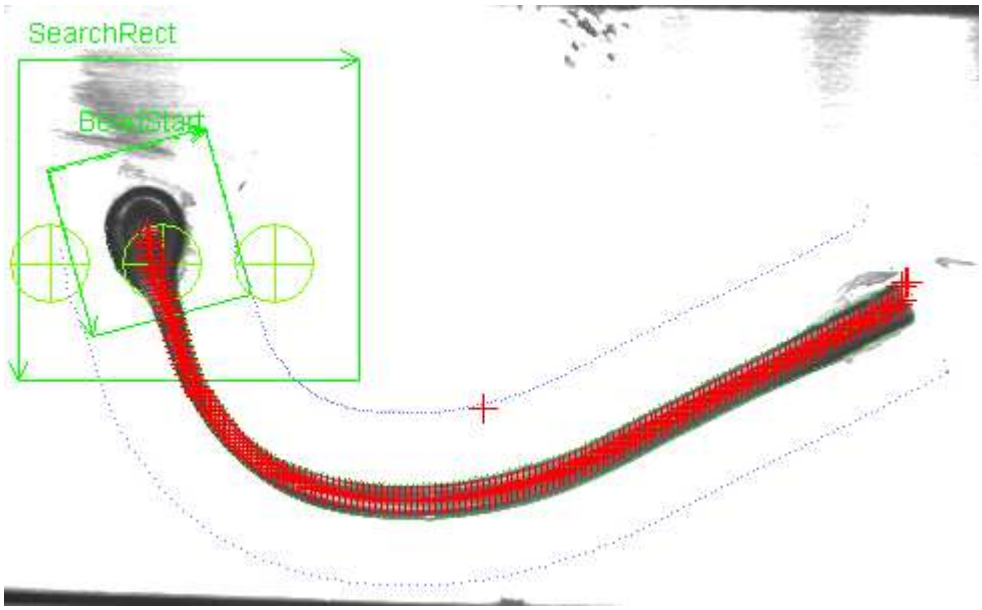
- "inspect pts. code" [Double array] – An array having a code number for each point in the inspected bead path. The code number is one of the following:
  - 0 = This point is ignored (was part of a short run of defects)
  - 1 = This point is valid, that is, within the specified thickness tolerances
  - 2 = This point is outside of the Image Window and should not be used
  - 3 = This point is missing, typically due to a gap in the bead
  - 4 = This bead point's thickness is below tolerance (too thin)
  - 5 = This bead point's thickness is above tolerance (too thick)
- "defect run count" [Double] – Number of defect runs in this bead inspection, after ignored points are excluded. A "defect run" is a sequence of bead points that are defective – either missing, off the image, too thick or too thin. A run can be as short as one invalid bead point.
- "defect index" [Double array] – An array of numbers that are indexes into the full array of inspected bead points and indicate the start of a run of defective bead points. The length of this array is given by "defect run count". Each run's length is given by its corresponding entry in the "defect run length" array.
- "defect run length" [Double array] – The length of each defect run. A run can be as short as one defective (invalid) bead point. The length of this array is given by "defect run count". Using this array and the "defect index" array, you can quickly find and step through defective bead points within a much larger array of bead points.

Using Alignment with the Bead Tool

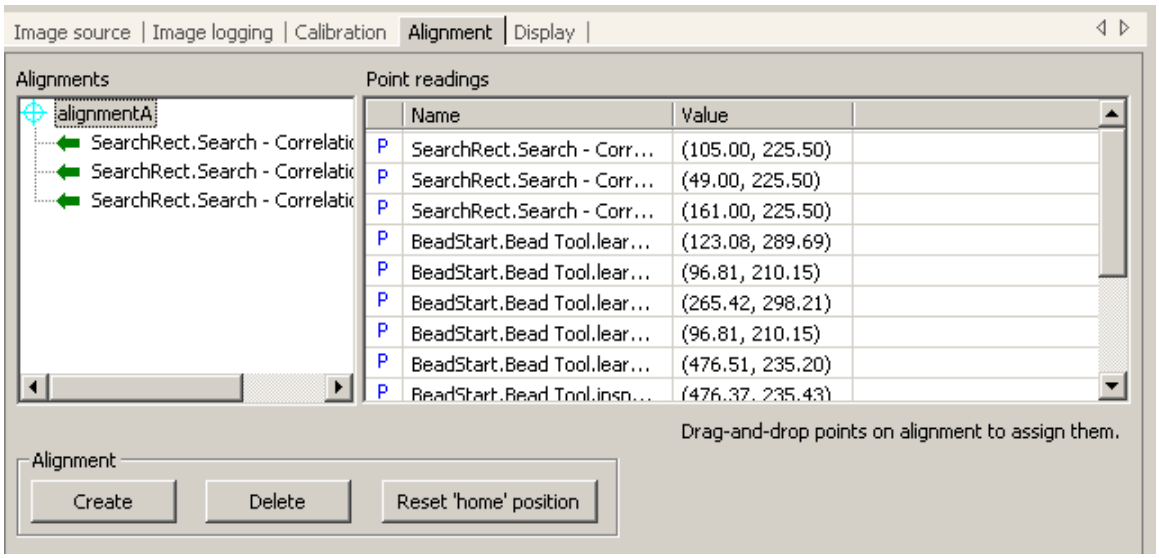
You should try to position or "stage" the bead path such that its position and angle only vary a small amount. If this is not possible, then you can use Sherlock's alignment capabilities to adjust the position and angle (but not size) of the bead path's starting ROI.

Alignment is done in the usual way. We will step through an example to show you how this works.

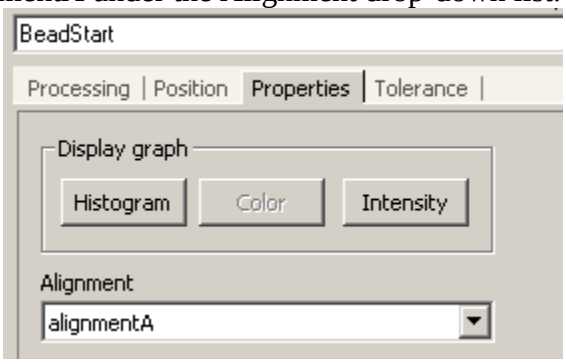
1. Set up the Bead Tool as described above. Here we call the Rectangle ROI around the start of the bead path "BeadStart".
2. Set up another Rectangle ROI around the start of the bead or another feature on the bead and move this ROI to before the BeadStart ROI in the Program Window. We call this the SearchRect. Select one of Sherlock's "Search" algorithms and assign it to SearchRect. Set up the search algorithm to find at least three points so you can align both position (x,y) and angle. Once the search algorithm has learned the pattern in SearchRect, enlarge SearchRect so that it will encompass the position and angle changes in the BeadStart ROI. In the image below, we used the "Search – Correlation" algorithm to learn the image pattern at the start of the bead path and then enlarged SearchRect to allow for movement of the start of the bead path.



3. Now you have to associate the alignment values found by the search with the BeadStart ROI, so that the location of the bead start found in SearchRect can be used to modify the bead path values that start in BeadStart.
4. In the Image Window's Options (double click the Image Window outside of any ROI or use the toolbar), under the Alignment tab, create a new alignment and add the points from the search ROI. You will add three (or more) points to allow for changes in position and angle. First create a new alignment (called alignmentA in this example). Then drag the three or more reference points from your search ROI to this alignment. The Alignment screen will look like:



5. Close the Alignment screen. Open the BeadStart ROI (double click on BeadStart in the Image Window or the Program window) and go to the Properties tab. In this screen, select alignmentA under the Alignment drop-down list.



6. The alignment computed in SearchRect is now automatically applied to BeadStart. For example, here is the result when the bead path is rotated 10 degrees and moved. Often one or two points (shown as magenta crosses) at the end of the bead path will "fall off" the bead path when it is rotated or translated. This is due to sampling error and can be supressed by shortening the bead path by a few steps.