Sherlock 7 Technical Resource

Teledyne DALSA Corp., Industrial Products (IPD)
www.teledynedalsa.com/ipd 978.670.2002 (U.S.A.)

Document Revision: August 9, 2012

Search Edge

Search utilities

A common task in machine vision applications is to find a feature in acquired images. This may be done for several reasons:

- To verify that a part (resistor, bolt, label, pill) is present on every part
- To determine how well an instance of the feature in a newly-acquired image matches a "golden template" instance of the feature
- To determine the location of a feature relative to other features

Sherlock 7 includes several search tools.

	Strengths	Limitations
Correlation Creates a map of the grayscale values within the ROI	Accurate and fast Very good at coping with changes in illumination and focus Can find multiple matches	Allowing for image scaling and rotation can increase execution time significantly
Geometric Creates a model of edges in the feature	Finds matches at any rotation Can find multiple matches	Does not allow for image scaling
Line Based Creates a model of straight edges in the feature	Optimized to look for straight edges Finds matches at any rotation Allows for scaling Generally runs faster than **Geometric**	Can find only one match
Edge Creates a model of edges in the feature **Note: Search Edge** requires an additional license. It is not part of the standard Sherlock release.	Very accurate Allows for image scaling and rotation Can be trained on more than one feature	

Search Edge is an edge-based search tool. It creates a **model** of the edges in the feature it is trained on, and then looks for matches for that model in new images.

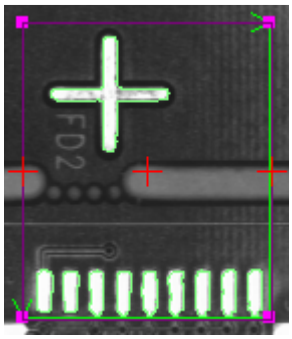
Search Edge is a two-pass algorithm. During the first pass, it determines the approximate locations of candidate matches. During the second pass, it determines the precise locations of final matches from among the candidates.

Search Edge has several parameters you set that determine which edges are included in the model at train time, and which edges in new images are compared to the edges in the model at search time.

To use **Search Edge**

1. Train a good example of the feature to create the **model**
2. Define the search area
3. Acquire a new image, search for matches for the model and return the matches' locations, match scores, and other information

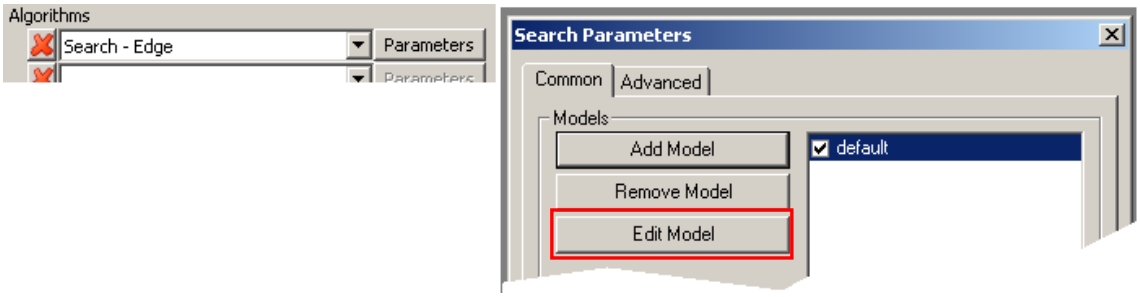
Creating the model (training)



Surround the feature with a rectangular ROI and select the **Search Edge** algorithm from the ROI's algorithm list. As soon as the algorithm is selected, the edges are detected using default parameters, and the model is created.

After the model has been created, the algorithm is put into run (search) mode.

Search Edge has many parameters that determine what constitutes an edge in the trained model. You can change the parameters from the **Model Editor**. To display the Model Editor, click the **Edit Model** button on the algorithm's **Search Parameters** dialog.



Resolution

Resolution determines how much of the information in an edge (a contour) is used when creating the model. The lower the resolution, the less information used.

Automatic – If enabled, the contour resolutions are calculated by analyzing all the models. If not enabled, the following **Coarse** and **Fine** parameters are used.

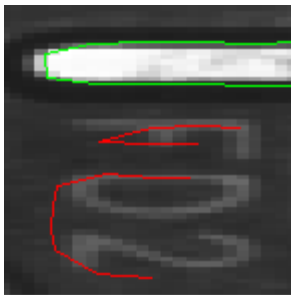
Coarse – Specifies how much the contours in the model are scaled down for the first pass of the search, when approximate locations of matches are determined.

Fine – Specifies how much the contours in the model are scaled down for the second pass of the search, when precise locations of matches are determined.

For **Coarse** and **Fine**, a value of 1 results in no scaling; a value of *n* reduces the contour resolution by a factor of *n*. Valid values for **Coarse** and **Fine** are 1 to 16. **Coarse** cannot be less than **Fine**. For example, if **Fine** is 3, **Coarse** must be in the range of 3 to 16. Setting **Coarse** and **Fine** to the same value results in the same degree of model contour resolution being used for both passes of the search.

Here is a model with **Coarse** set to 2.

Here is the same model with **Coarse** set to 4. Notice that the model's edges less accurately follow the true edges in the image.



Edge Strength Mode – Specifies the method for calculating the edge strength threshold for detecting contours in the model image. This parameter adjusts the sensitivity of the edge strength threshold in order to control the amount of edge points detected in the model image

Low – Calculates a low-sensitivity adaptive threshold from the input image so that a small number of contour points are detected.

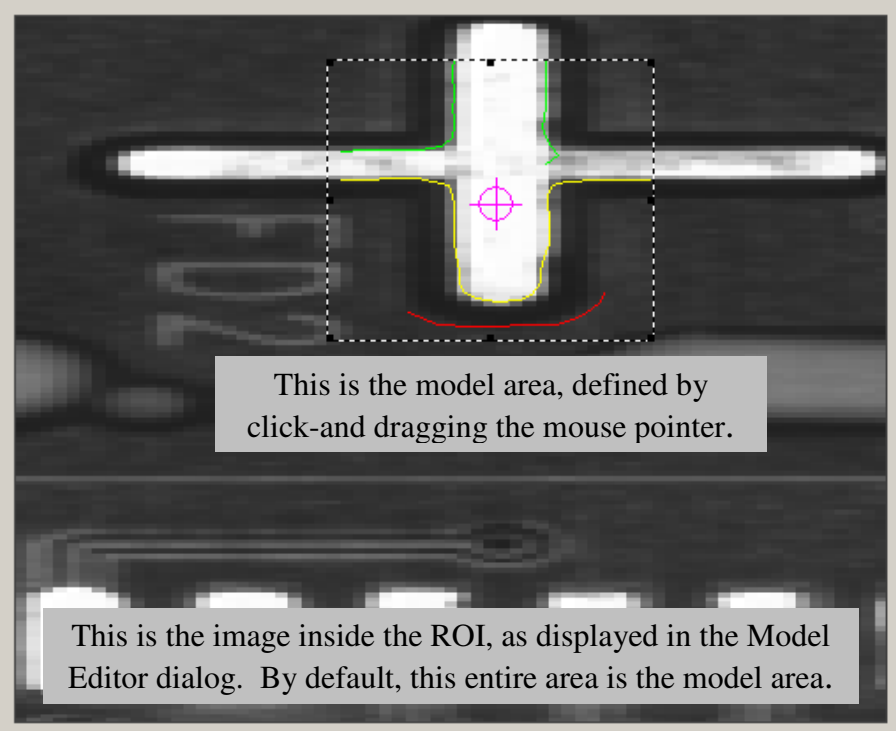
Normal – Calculates a normal-sensitivity adaptive threshold from the input image so that a reasonable number of contour points are detected.

High – Calculates a high-sensitivity adaptive threshold from the input image so that a large number of contour points are detected.

Manual – Uses a manual threshold to determine the number of contour points detected.

Edge Strength Threshold – The threshold used if **Edge Strength Mode** is set to **Manual.**

You can click-and-drag the mouse pointer inside the model display window to define a sub-rectangle of the ROI to be the model area. The model area is shown as a dashed line.



You must click the **Apply** button after changing any of the preceding parameters for them to take effect.

Contour Selection Mode – Controls the amount of contours that will be automatically selected from the detected contours at both the coarse and fine resolutions. After the contour detection phase (controlled by **Edge Strength Mode**), a contour selection phase is applied in order to select a subset of contours to be used by the search algorithm.

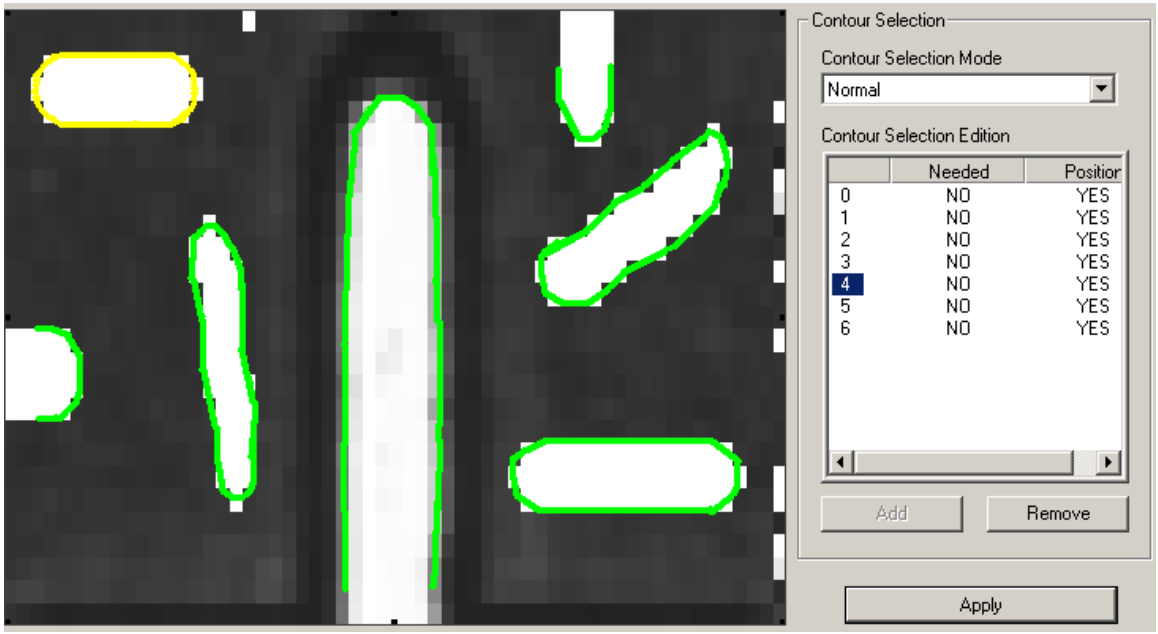
None – No contours are selected.
Low – A smaller amount of contours than the normal are selected.
Normal – The optimal amount of contours are selected.
High – A larger amount of contours than the normal are selected.
All – All contours are selected.

Contour Selection Editor – You can explicitly remove and add contours from the model, and set a contour's presence and/or position to be determining factors in a match's failure or success.

To remove a contour from the model

1. Select the contour in the image or in the list of contours. The selected contour is colored yellow.
2. Click the **Remove** button. After the contour is removed from the model, it is colored red.

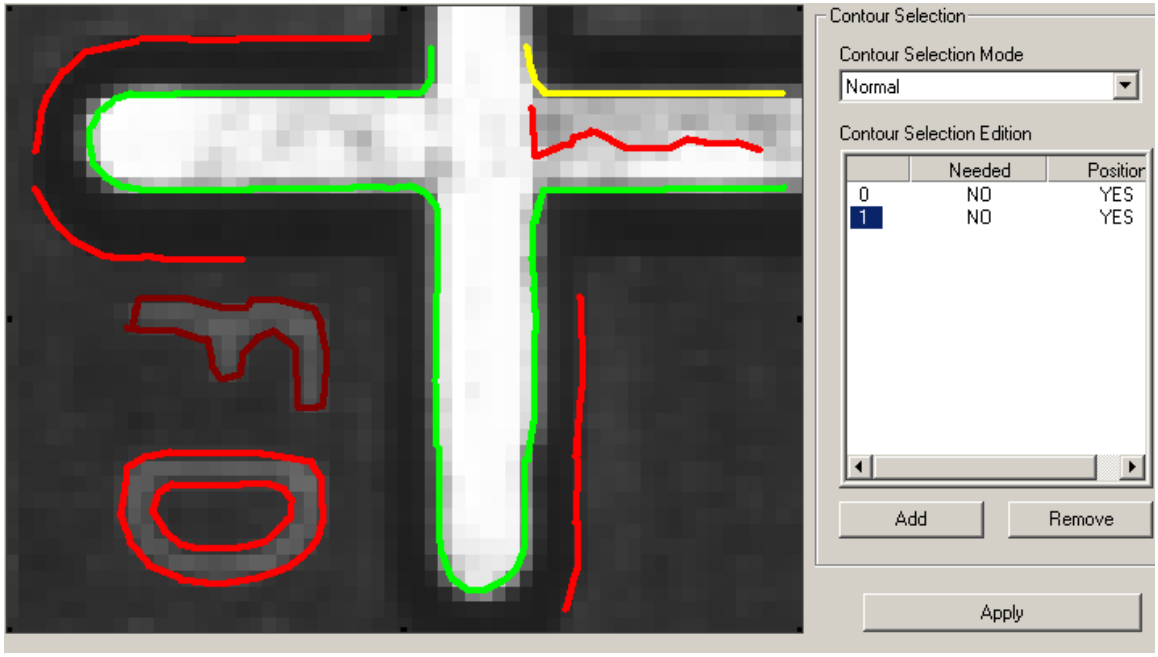
In this image, contour 4 has been selected. When the **Remove** button is clicked, the contour will be removed from the model.



To add a contour to the model

1. Double-left-click on a contour to select the entire contour; a selected contour is colored maroon. If you want to add only a segment of a contour to the model, single left-click on the segment. The selected segment is colored maroon; the rest of the contour is colored blue/cyan. This is a visual indication that you haven't selected the entire contour. (Note that the definition of a "segment" is somewhat vague; it is not always possible to select exactly the segment you want.)
2. Click the **Add** button. The contour is colored green to indicate that it part of the model

In this image, the contour defining the sideways 'F' has been selected. When the **Add** button is clicked, the contour will be added to the model.



If a contour in the model must be found during the search for a match to succeed, change its **Needed** state to **YES** by clicking on the contour in the image area or the contour number in the Contour Selection Editor, and double-clicking on **NO**. You can set the **Needed** state for a contour to **YES** for either or both the **Coarse** mode and **Fine** mode. Setting the **Needed** state to **YES** for the **Coarse** mode removes a potential match from further examination during the second pass of the algorithm if the contour is not found during the first pass. This can save some processing time if the search area contains many potential matches, only some of which include crucial contours.

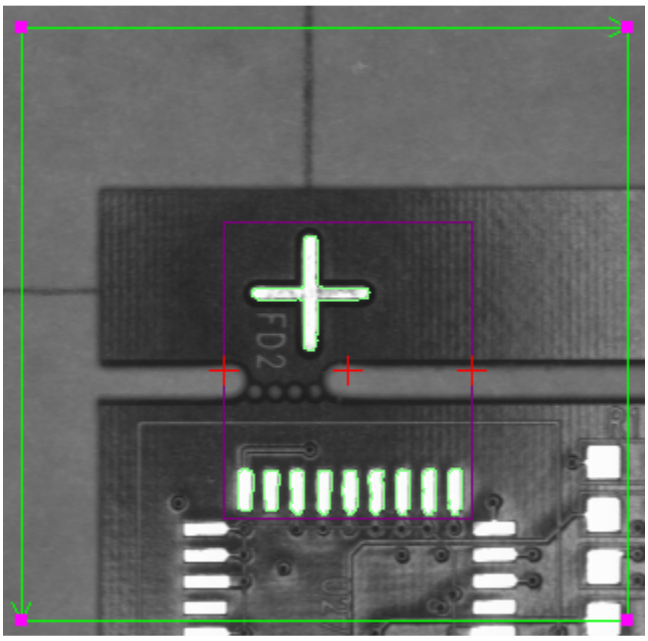
If a contour must be found in the same position relative to the upper-left corner of the model's defining rectangle as at the time of model creation, set its **Position** state to **YES** (the default). If the contour's position within the model's rectangle is irrelevant, set its **Position** state to **NO**.

Clicking the **Apply** button resets the contour list back to its original state. Any contours you removed are replaced, and any contours you added are removed. The **Needed** states for all contours are set to **NO**, and the **Position** states are set to **YES**.

Searching for the model

Define the search area

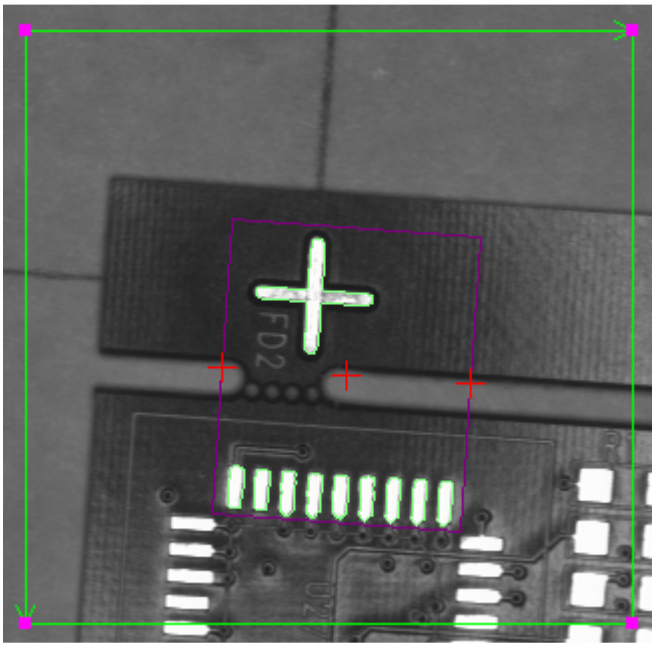
Expand the same ROI to define the search area. A smaller search area results in less processing time, so restrict the search area to that part of the image where you know matches to the model will occur.



Acquire a new image and search

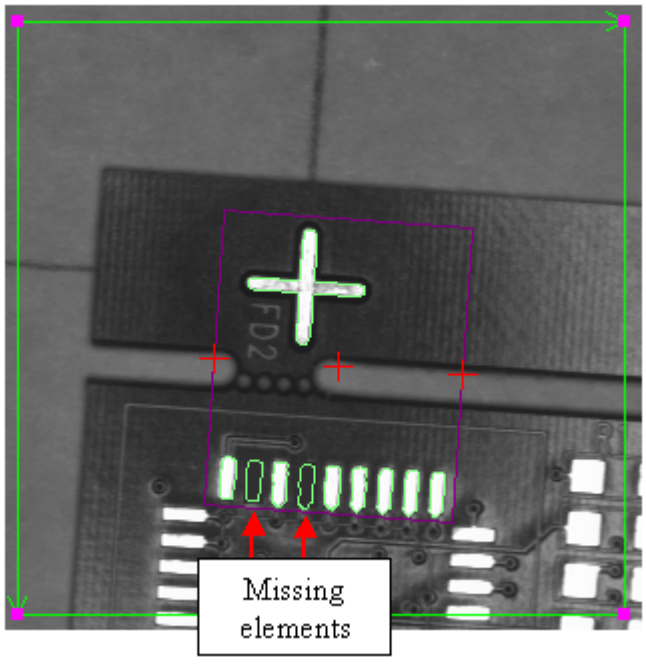
In run mode, **Search Edge** finds the best matches for model, and returns the location and score of the found matches. **Search Edge** searches the full 360 degrees to find the best matches to the trained edges.

RectA.Search - Edge.matches count	1.00
RectA.Search - Edge.location	(336.08, 280.02)
RectA.Search - Edge.angle	4.07
RectA.Search - Edge.score	100.00



Here a less-than-perfect match was found; note the lower score.

RectA.Search - Edge.matches count	1.00
RectA.Search - Edge.location	(336.08, 280.02)
RectA.Search - Edge.angle	4.07
RectA.Search - Edge.score	90.66



Model management

Each instance of the **Search Edge** algorithm can be trained on multiple models. These models can be variations on the same pattern, or completely different patterns.

Add Model – Add a new model to the search. You will be prompted for a name for the new model.

Remove Model – Remove the selected (highlighted) model from the search. This cannot be undone.

If an instance of **Search Edge** has been trained on multiple models, models can be selected and unselected, so that they are searched for (selected) or not searched for (unselected) at run time. However, models can be selected and unselected only through the algorithm's **Parameters** dialog, not at run time; that is, you cannot select or unselect a model by setting a variable or calling an instruction as the Sherlock investigation executes

Search parameters

Common parameters

Maximum number of matches – By default, the algorithm finds one (the best) match for the models it has been trained on. If you set **Maximum number of matches** to *n*, where *n* is 2 or greater, the algorithm returns information on the *n* best matches. The algorithm returns information on fewer than *n* matches if not enough matches that meet the **Minimum score** are found. If you have trained multiple models, **Maximum number of matches** determines the total number of matches for all the models, not the number of matches for each model.

Minimum score – Only matches with scores equal to or greater than this value are returned.

Enable Timeout / Timeout (ms) – If **Enable Timeout** is set to **True**, the algorithm will search for up to **Timeout** milliseconds for matches.

Min. rotation – The greatest counterclockwise angle of rotation at which the algorithm will look for a match. Expressed as a negative angle from 0 to -180.

Max. rotation – The greatest clockwise angle of rotation at which the algorithm will look for a match. Expressed as a positive angle from 0 to 180.

Min. scale – The smallest scaling factor at which the algorithm will match the feature. Expressed as a floating-point number, where 1.0 means no downward scaling, 0.5 means 50%.

Max. scale – The largest scaling factor at which the algorithm will find the feature. Expressed as a floating-point number, where 1.0 means no upward scaling, 2.5 means 250%.

Advanced parameters

You can set the **Advanced** parameters to determine what constitutes an edge at search time. Some of these parameters are the same as parameters you set to determine whether an edge is included in the model. Because these parameters control the operation of the algorithm at search time, there is no visual display of their effects at setup time.

Resolution

Automatic – If enabled, the contour resolutions are calculated by analyzing all the models. If not enabled, the following **Coarse** and **Fine** parameters are used.

Coarse – Specifies how much the contours in the search area are scaled down during the first pass of the search, when approximate locations of matches are determined.

Fine – Specifies how much the contours in the search area are scaled down during the second pass of the search, when precise locations of matches are determined.

For **Coarse** and **Fine**, a value of 1 results in no scaling; a value of *n* reduces the contour resolution by a factor of *n.* Valid values for **Coarse** and **Fine** are 1 to 16. **Coarse** cannot be less than **Fine**. For example, if **Fine** is 3, **Coarse** must be in the range of 3 to 16. Setting **Coarse** and **Fine** to the same value results in the same degree of contour resolution being used for both passes of the search

 Entering high values for **Coarse** and **Fine** may result in the loss of edge information to the point where **Search Edge** cannot reliably find a match for the model.

Edge strength – Specifies the minimum edge strength for a contour to be detected during the search.

Low – Calculates a low-sensitivity adaptive threshold from the input image so that a small amount of contour points are detected.

Normal – Calculates a normal-sensitivity adaptive threshold from the input image so that a reasonable amount of contour points are detected.

High – Calculates a high-sensitivity adaptive threshold from the input image so that a large amount of contour points are detected.

Manual – Uses a manual threshold specified by **Fixed value** to determine the amount of contour points detected.

Models mode – Calculates a threshold from the input image using the model's edge strength mode. The model's mode is specified by SetModelEdgeStrengthMode**.**

Models value – Uses a threshold corresponding to the model's edge strength value. This threshold is calculated from the model image at the training stage.

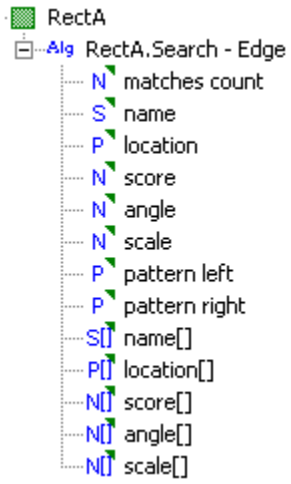
Models value and mode – Uses the lesser of the thresholds from **Models mode** and **Models value**.

Search Effort – A low setting (0) provides a faster search that may miss matches that are partly occluded. A high setting (10) is useful for finding partly occluded matches in cluttered or noisy images, or for models made up of small contours at the fine level.

Accuracy Effort – A low setting (0) provides rough positioning of the matches and faster execution. A high setting (10) provides high accuracy positioning of the matches but slower execution.

Show pattern – If **True**, the edges of found patterns are displayed at run time; if **False**, they are not. This is a matter of viewer preference only; it does not affect algorithm execution. If **Show pattern** is set to **True** and the algorithm finds no matches, no edges are displayed.

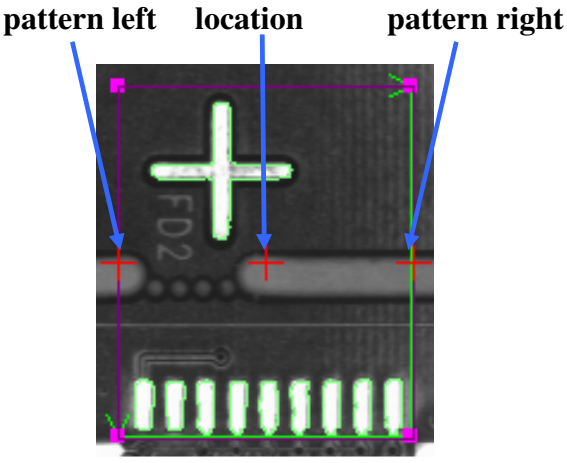
Returned values (readings)



matches count – The number of matches found.
name – The name of the best match.
location – The location of the best match.
score – The score of the best match. A perfect match returns a score of 100.
angle – The angle of the best match. The angle is returned in degrees or radians, depending on the setting in Sherlock's **Options → Application** dialog.
scale – The scale of the best match, expressed as a floating-point number, where 1.0 means the match is the same scale as the trained model, and 0.5 means the match is 50% of the size of the trained model.
pattern left and **pattern right** – Two points created by the algorithm, centered about **location** and separated by the width of the ROI when the model is trained. **Pattern left** and **pattern right** can be used as landmarks in an alignment object that needs to calculate rotation.

If **Maximum number of matches** is greater than 1 and more than one match is found:
name[] – The names of the matched models, ordered by score
location[ ] – The locations of the matches, ordered by score.
score [ ] – The scores of the matches, from highest to lowest.
angles[ ] – The angles of the matches, ordered by score.
scale[ ] – The scales of the matches, ordered by score.

The first element of each array contains the same information as its corresponding single-value reading. (For example, **name[0] = name**.)



RectA.Search - Edge.pattern left	(277.00, 279.00)
RectA.Search - Edge.location	(339.00, 279.00)
RectA.Search - Edge.pattern right	(401.00, 279.00)