

# User's Manual

**ACTI✓3LOBFINDER**

**You Name It, We Find It**

Microsoft, Windows, Windows NT, Windows 2000, Windows XP, Visual Basic, Microsoft .NET, Visual C++, Visual C#, and ActiveX are either trademarks or registered trademarks of Microsoft Corporation.

All other nationally and internationally recognized trademarks and tradenames are hereby recognized.

Copyright © 2000-2008 by MVTec Software GmbH, München, Germany



Edition 1	November 2000	(ActivVisionTools 1.0)
Edition 2	Februar 2001	(ActivVisionTools 1.2)
Edition 3	April 2001	(ActivVisionTools 1.3)
Edition 4	September 2001	(ActivVisionTools 2.0)
Edition 5	November 2002	(ActivVisionTools 2.1)
Edition 6	January 2005	(ActivVisionTools 3.0)
Edition 7	February 2006	(ActivVisionTools 3.1)
Edition 8	May 2008	(ActivVisionTools 3.2)

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without prior written permission of the publisher.

More information about ActivVisionTools can be found at:

<http://www.activ-vision-tools.com>

# How to Read This Manual

---

This manual explains how to use `ActivBlobFinder` to extract the blobs (objects) you look for from images. It describes the functionality of `ActivBlobFinder` and its cooperation with other `ActivVisionTools` with Visual Basic examples. Before reading this manual, we recommend to read the manual [Getting Started with ActivVisionTools](#), which introduces the basic concepts of `ActivVisionTools` and the [User's Manual for ActivView](#) to learn how to load and display images.

To follow the examples actively, first install and configure `ActivVisionTools` as described in the manual [Getting Started with ActivVisionTools](#). For each example in this manual, there is a corresponding Visual Basic project; these projects can be found in the subdirectory `examples\manuals\activblobfinder` of the `ActivVisionTools` base directory you selected during the installation (default: `C:\Program Files\MVTec\ActivVisionTools`). Of course, you can also create your own Visual Basic projects from scratch.

We recommend to **create a private copy of the example projects** because by experimenting with the projects, you also change their *state*, which is then automatically stored in the so-called description files (extension `.dsc`) by `ActivVisionTools`. Of course, you can restore the state of a project by retrieving the corresponding description file from the CD.





# Contents

---

<b>1</b>	<b>About ActivBlobFinder</b>	<b>1</b>
1.1	Introducing ActivBlobFinder . . . . .	2
1.2	The Sub-Tools of ActivBlobFinder . . . . .	6
<b>2</b>	<b>Using ActivBlobFinder</b>	<b>9</b>
2.1	Specifying Regions of Interest . . . . .	10
2.2	Inspecting the Image . . . . .	12
2.3	Extracting Blobs . . . . .	14
2.4	Processing Extracted Blobs . . . . .	20
<b>3</b>	<b>Combining ActivBlobFinder with other ActivVisionTools</b>	<b>25</b>
3.1	Analyzing Blobs . . . . .	26
3.2	Evaluating Results . . . . .	28
3.3	Output of Results . . . . .	30
<b>4</b>	<b>Tips &amp; Tricks</b>	<b>33</b>
4.1	Adapting the Display of Results . . . . .	34
4.2	Configuring the Two Execution Modes . . . . .	36



# Chapter 1

## About ActivBlobFinder

---

This chapter will introduce you to the features and the basic concepts of ActivBlobFinder. It gives an overview about ActivBlobFinder’s *master tool* and its *support tools*, which are described in more detail in [chapter 2](#) on page 9.

<b>1.1</b>	<b>Introducing ActivBlobFinder</b> . . . . .	<b>2</b>
<b>1.2</b>	<b>The Sub-Tools of ActivBlobFinder</b> . . . . .	<b>6</b>

## 1.1 Introducing ActivBlobFinder

With the help of ActivBlobFinder, you can extract so-called *blobs* from images. Generally speaking, a blob is a region in the image whose pixels differ in some way from the surrounding part of the image. A blob can have any shape and may even contain holes. Extracted blobs can be called the “foreground” of an image, the rest the “background”.

ActivBlobFinder offers multiple methods for extracting blobs, ranging from very fast and simple ones to methods that extract blobs even from a difficult background. Moreover, it allows to process extracted blobs further, e.g., to fill holes.

### The Different Extraction Methods

How to extract blobs of course depends on the way the blobs differ from the background. Nevertheless, all the methods described in the following have something in common: They are based upon specifying a so-called *threshold*; pixels on one side of the threshold are classified as belonging to the background, on the other side they belong to a blob.

**Global Thresholding.** The simplest method is to classify pixels according to their gray value: In the example depicted in [figure 1.1 a](#), blobs correspond to the dark parts (low gray values) on a light background (high gray values). In such a case, you can easily choose a threshold so that all pixels belonging to blobs lie below this threshold. This method can be called *global thresholding* as one threshold is applied globally, i.e., for all pixels. Global thresholding can also be applied using two thresholds, extracting either the gray values inside or outside the thresholds as the objects.

**Automatic Thresholding.** If the gray values corresponding to the blobs or the background vary from image to image, it may be impossible to find one threshold suitable for all images. For such cases, ActivBlobFinder provides a method that automatically determines a suitable threshold for each image. The underlying assumption is that two gray values dominate in the image: the gray value of the background and that of the blobs. The threshold is then set in the middle of these two values.

The automatic thresholding method as described above may fail when extracting thin characters as in [figure 1.1 b](#), because there will not be a second dominating gray value. Therefore, there exists a second variant of this method which is optimized for extracting characters. This method only determines the dominating gray value corresponding to the background; relative to this gray value you can then specify the actual threshold.

**Local Thresholding.** Global thresholding is a very fast method; however, it will fail when pixels cannot be classified by their gray value alone, like for example in [figure 1.1 c](#): Here, the gray value of characters to be extracted (indicated by the vertical line in the gray value



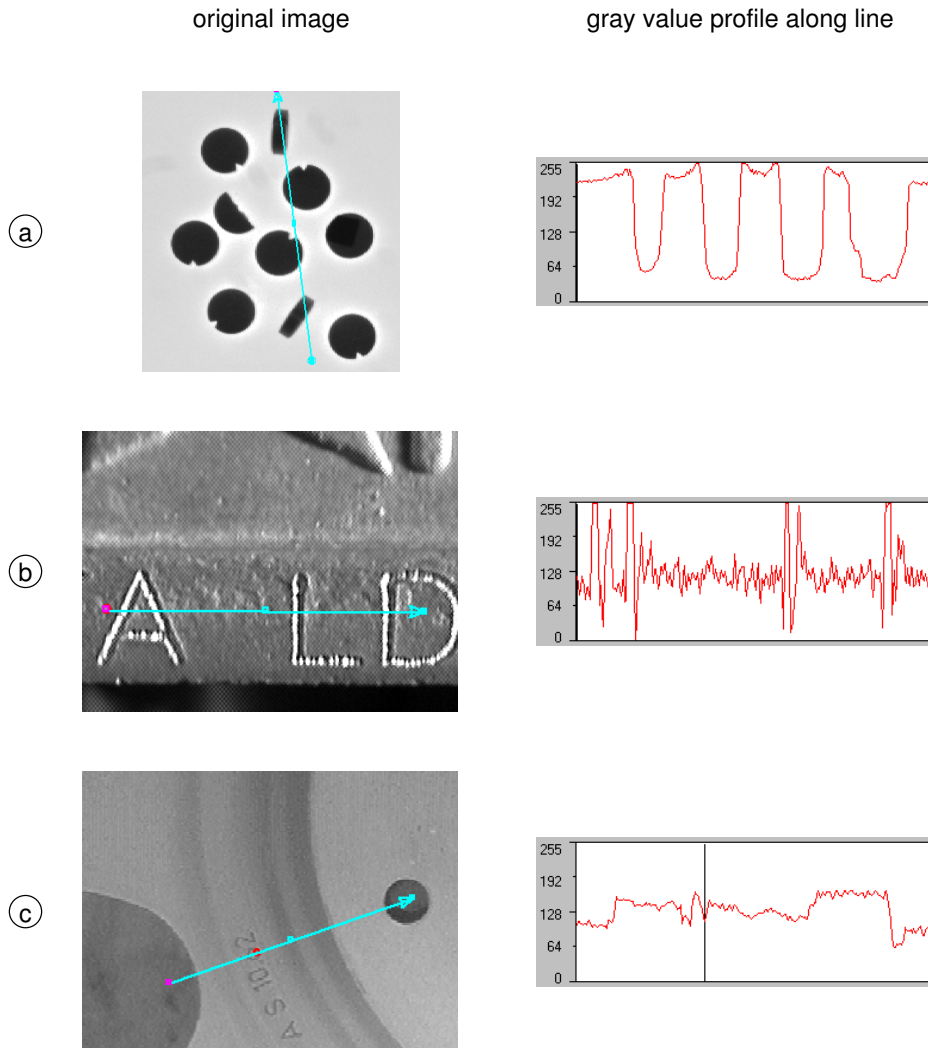


Figure 1.1: When to use which thresholding method: (a) global threshold, (b) automatic threshold for characters, (c) local threshold.

profile) lies in between values corresponding to different parts of the background. Thus, it is impossible to find a suitable global threshold in this case. However, the characters differ *locally* from the background. To extract such blobs, ActivBlobFinder therefore provides a method that

uses a local threshold corresponding to the difference in gray values between a blob and its local background, together with information about the expected size of the blobs.

### Further Processing

In some cases, the extracted blobs may not correspond exactly to the parts of the image you wanted to extract. As for example shown in [figure 1.2 a](#), the extracted blobs may have holes or a “ragged” contour. Therefore, ActivBlobFinder allows you to process extracted blobs further, i.e., modify their shape. Such operations are also called *morphological operations*.

[Figure 1.2b](#) shows the result of *filling gaps*: The two blobs corresponding to the lowest pin and to the arrow now form one blob. As a side effect, the contours were smoothed and the holes filled.

Another operation allows you to *fill holes* up to a given size without any other effect on the shape of the blobs (compare [figure 1.2 c](#) to [figure 1.2 a](#)).

You may also *remove bulges* from the contour of a blob (see [figure 1.2 d](#)) or, the “reverse” operation, *fill indentations* (see [figure 1.2 e](#)). A note on the latter: This operation is very similar to filling gaps. The difference is that the filling of indentations is limited to the blobs themselves; this means that gaps between blobs are not closed (compare [figure 1.2 e](#) to [figure 1.2 b](#)).

### The Role of ROIs

When using ActivBlobFinder, regions of interest have to be placed with care. On the one hand, they should of course be as small as possible to speed up image processing. However, when applying automatic thresholding an ROI must contain enough background so that ActivBlobFinder can identify the corresponding gray value and find a suitable threshold as described above.

### Results

The results of ActivBlobFinder are “just” the extracted blobs, i.e., the corresponding regions in the image. These blobs can then be analyzed using other ActivVisionTools. For example, you can use ActivFeatureCalc to calculate various shape features for each blob. The results of ActivFeatureCalc can then be further evaluated using ActivDecision, before you output them via ActivFile, ActivSerial, or ActivDigitalIO; furthermore, you can access results via the programming interface (see the [User’s Manual for ActivFeatureCalc](#) for more information).

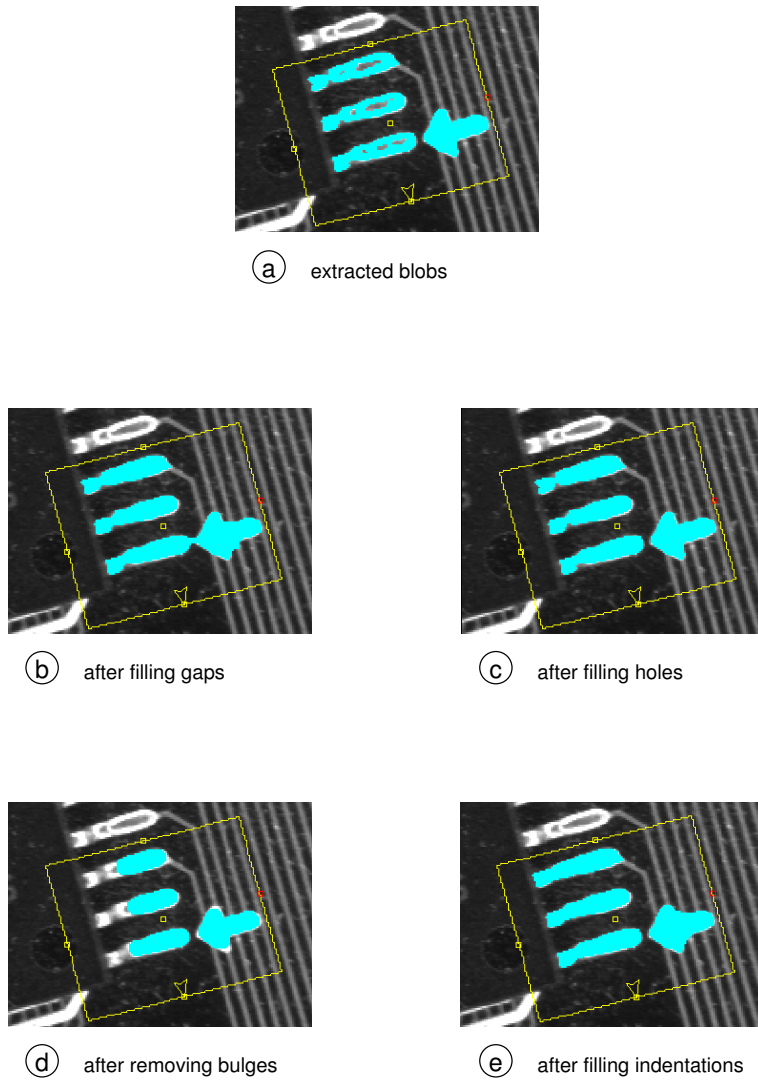


Figure 1.2: Further processing of blobs: (a) default extraction, (b) filling gaps, (c) filling holes, (d) removing bulges, (e) filling indentations.

## 1.2 The Sub-Tools of ActivBlobFinder

Beside its *master tool*, ActivBlobFinder provides one *support tool*. In [figure 1.3](#) they are depicted together with other ActivVisionTools that you will use in a typical ActivBlobFinder application.



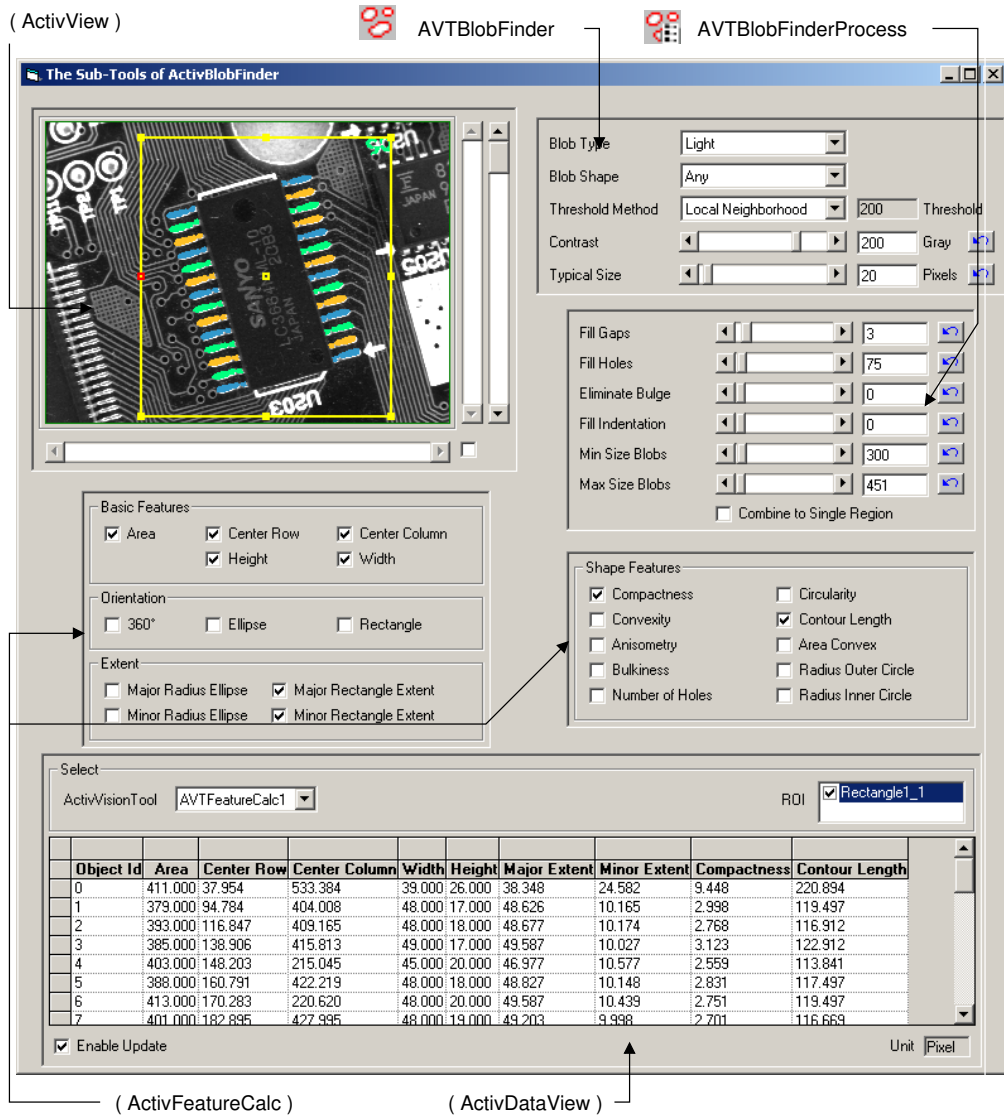
AVTBlobFinder is the *master tool* of ActivBlobFinder. It provides multiple extraction methods optimized for different blob shapes and background.

How to use AVTBlobFinder is described in more detail in [section 2.3](#) on page 14.



AVTBlobFinderProcess is a *support tool* of ActivBlobFinder. It allows to further process extracted blobs, e.g., fill holes.

How to use AVTBlobFinderProcess is described in [section 2.4](#) on page 20.



ActivBlobFinder

Figure 1.3: The sub-tools of ActivBlobFinder together with suitable other tools.



## Chapter 2

# Using ActivBlobFinder

---

This chapter will explain how to use ActivBlobFinder to extract different types of blobs and to further process the blobs. Furthermore, it shows how to use the *inspection tools* ActivZoom and ActivLineProfile to determine suitable parameters for the extraction and the processing of blobs.




<b>2.1</b>	<b>Specifying Regions of Interest</b>	<b>10</b>
<b>2.2</b>	<b>Inspecting the Image</b>	<b>12</b>
<b>2.3</b>	<b>Extracting Blobs</b>	<b>14</b>
2.3.1	Extracting Blobs Using Fixed Thresholding	14
2.3.2	Extracting Blobs Using Automatic Thresholding	16
2.3.3	Extracting Blobs Using a Local Threshold	18
<b>2.4</b>	<b>Processing Extracted Blobs</b>	<b>20</b>

## 2.1 Specifying Regions of Interest Using


ActivBlobFinder lets you choose between different shapes of ROIs, e.g., arbitrarily oriented rectangles or ellipses. You create an ROI using AVTViewROI, which is a *support tool* of ActivView.

### Visual Basic Example

#### Preparation for the following example:

- Open the project rois/blob\_rois.vbp. Alternatively, create a new project and place the following tools on the form (in parentheses the icon you have to double-click with the left mouse button): AVTView () , AVTViewROI () , and AVTBlobFinder ().
- Execution the application (Run > Start or via the corresponding button). Open AVTViewFG by clicking into AVTView with the right mouse button and selecting Image Acquisition in the popup menu. Select the image misc\pills\_01 in the combo box Input File.

The following steps are visualized in [figure 2.1](#).

- ① First, you have to tell AVTViewROI to create an ROI for ActivBlobFinder by selecting the corresponding entry in the combo box ActivVisionTool. In this box, all ActivVisionTools are listed that have been placed upon the form. By default, the tools are referenced by the name of the corresponding ActiveX control plus a counter to distinguish between multiple instances of a control on the form. In our example, there is only one item in the combo box, AVTBlobFinder1.
- ② Next, select the desired shape of the ROI, for example a rectangle.
- ③ To draw the ROI, move the mouse in the image while keeping the left mouse button pressed. Please experiment at this point with the different shapes. The creation of a polygonal ROI () is more complex: By the first mouse movement, the first side of the polygon is created. You can add a new corner point by clicking on the line with the left mouse button; when you drag it by keeping the mouse button pressed, new polygon sides are created, where you can again add new corner points. To delete a corner point, drag it onto a neighboring corner point.
- ④ You can now move the ROI by dragging its pick point in the middle. By dragging the outer pick points you modify its shape. Again, please experiment to get familiar with the ROIs. Note that for a polygonal ROI the “middle point” appears at the center of gravity of the ROI and therefore changes whenever the polygon is modified; besides, it may even lie outside the ROI for concave polygons.



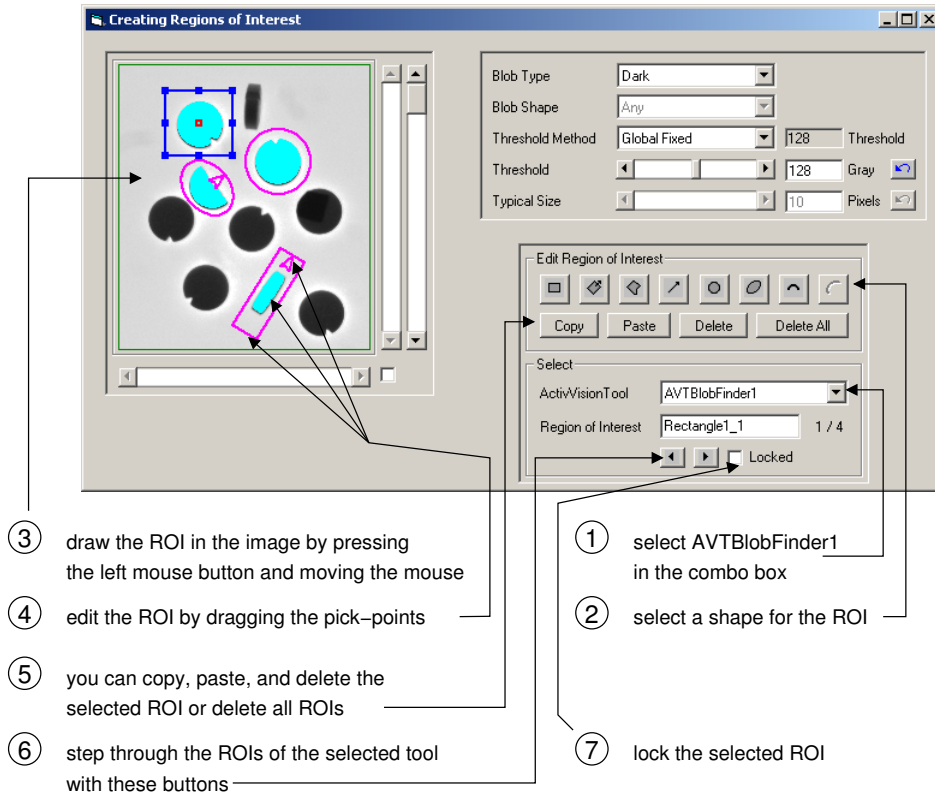




Figure 2.1: Creating a region of interest.

- ⑤ To create a second ROI, repeat step ②. You can also copy, paste, and delete the selected ROI or delete all ROIs.
- ⑥ ROIs can be selected by clicking next to it in the image. Alternatively, use the two arrow buttons   to step through all the ROIs of the selected tool.
- ⑦ By checking  Locked you can lock the selected ROI and thus prevent it from accidental editing.



Note that AVTBlobFinder also allows to use arbitrary regions as ROIs. These ROIs, however, can only be set via the programming interface of ActivVisionTools. An example can be found in the Advanced User's Guide for ActivVisionTools, [section 3.4.1.2](#) on page 72.

## 2.2 Inspecting the Image Using and

To get optimal extraction results, you should “describe” the blobs you want to extract as closely as possible. In this section, we show how to obtain this information by inspecting the image using the *inspection tools* `ActivZoom` and `ActivLineProfile`. In the following sections, these techniques will then be used to help you extract the blobs you look for.

### Visual Basic Example

#### Preparation for the following example:

- If you worked on the previous example, you may continue using this project. At design time, add `AVTZoom` and `AVTLineProfile` by double-clicking  and .
- Otherwise, open the project `inspecting\blob_inspecting.vbp`.
- Execute the application (`Run > Start` or via the corresponding button). Open `AVTViewFG` by clicking into `AVTView` with the right mouse button and selecting `Image Acquisition` in the popup menu. Select the image `misc\pills_01` in the combo box `Input File`.

The following steps are visualized in [figure 2.2](#).

- ① In the combo box `Scaling` of `AVTZoom`, you can select the zooming scale.
- ② When the mouse pointer is inside the displayed image, the surrounding part of the image is zoomed and displayed in `AVTZoom`. Furthermore, the gray value at the current position (marked with a red box in the zoomed image) is displayed in the text box `Gray Value`. The position itself is shown in the text boxes `Row` and `Column`.
- ③ To inspect a gray value profile, you create the line (or arc) of inspection using `AVTView-ROI`. First, select `AVTLineProfile1` in the combo box `ActivVisionTool`.
- ④ Then, select a shape and draw the ROI in the image as described in the previous section. The gray value profile along that line is displayed automatically. Note that in a rectangular ROI the pixel values are averaged perpendicular to the main axis of the rectangle. Experiment with different ROIs and watch the resulting gray value profiles.
- ⑤ For an easier inspection, `ActivLineProfile` provides a “cursor” which can be moved along the line. The position of this cursor is marked by a vertical line in `AVTLineProfile` and on the corresponding ROI in the image. The position itself is displayed in the text box `(Row, Col)`, the gray value in the text box `Gray Value`.
- ⑥ You can move the cursor by dragging the line with the left mouse button.

③ select AVTLineProfile1 in the combo box

④ select a shape for the ROI and draw it in the image

① select the zooming scale

② when the mouse is inside the display the zoomed image part is displayed together with gray value and position

⑤ the vertical line in the profile marks the position of the "cursor", whose position and gray value is displayed

⑥ move the "cursor" by dragging the line

The screenshot shows the following interface elements:

- Top Panel:** Blob Type (Dark), Blob Shape (Any), Threshold Method (Global Fixed), Threshold (128), Typical Size (10).
- Edit Region of Interest:** Copy, Paste, Delete, Delete All buttons.
- Select:** ActivisionTool (AVTLineProfile1), Region of Interest (Line\_1).
- Line Profile:** A graph showing intensity values (0 to 255) across a range of pixels (0 to 192). A red vertical line is positioned at approximately x=110.
- Gray Values:** (Row,Col) [122,118], Gray Value 233, Average 137.187, Min Gray 16, Deviation 100.359, Max Gray 255.

Figure 2.2: Inspecting the image.

## 2.3 Extracting Blobs Using

The following sections explain the different methods ActivBlobFinder offers to extract blobs.

### 2.3.1 Extracting Blobs Using Fixed Thresholding

As described in [chapter 1](#) on page 1, the simplest (and fastest) way to extract blobs is to classify pixels as belonging to a blob or to the background by comparing their gray value to a fixed global threshold. This method is well-suited, e.g., to the “black-and-white” images that stem from setups with a so-called *rear illumination*. In such a setup, objects are placed on a transparent surface between light source and camera; in the resulting image, the objects correspond to dark blobs on a light background (see e.g. [figure 2.3](#)). ActivBlobFinder also allows to use two thresholds.

#### Visual Basic Example

##### Preparation for the following example:

- If you worked on the previous example, you may continue using this project. At design time, remove AVTZoom and AVTViewROI.  
Otherwise, open the project `extract_fixed/blob_extract_fixed.vbp`.
- Execute the application and load the image `misc\pills_01`. If necessary, create some ROIs; AVTViewROI can be opened via the right mouse button menu of AVT-View.

The following steps are visualized in [figure 2.3](#).

- ① Select 'Global Fixed' in the combo box Threshold Method.
- ② In the combo box Blob Type, select whether the blobs to be extracted are darker or lighter than the background; in our example, select 'Dark'.
- ③ As the default value for the threshold is suited to this image, the blobs corresponding to the pills in the image are already extracted, without any action on your part. To take a closer look at what's happening, create a line-shaped ROI for AVTLineProfile as described in the previous chapter and place it over some of the pills. As you can see now, the blobs have a gray value of about 30-40, while the background corresponds to gray values well over 200.
- ④ Now experiment with different thresholds by using the slider Threshold or by specifying a value in the text field beside it. In [figure 2.3](#), the threshold is too low, therefore the pill boundaries are not extracted.

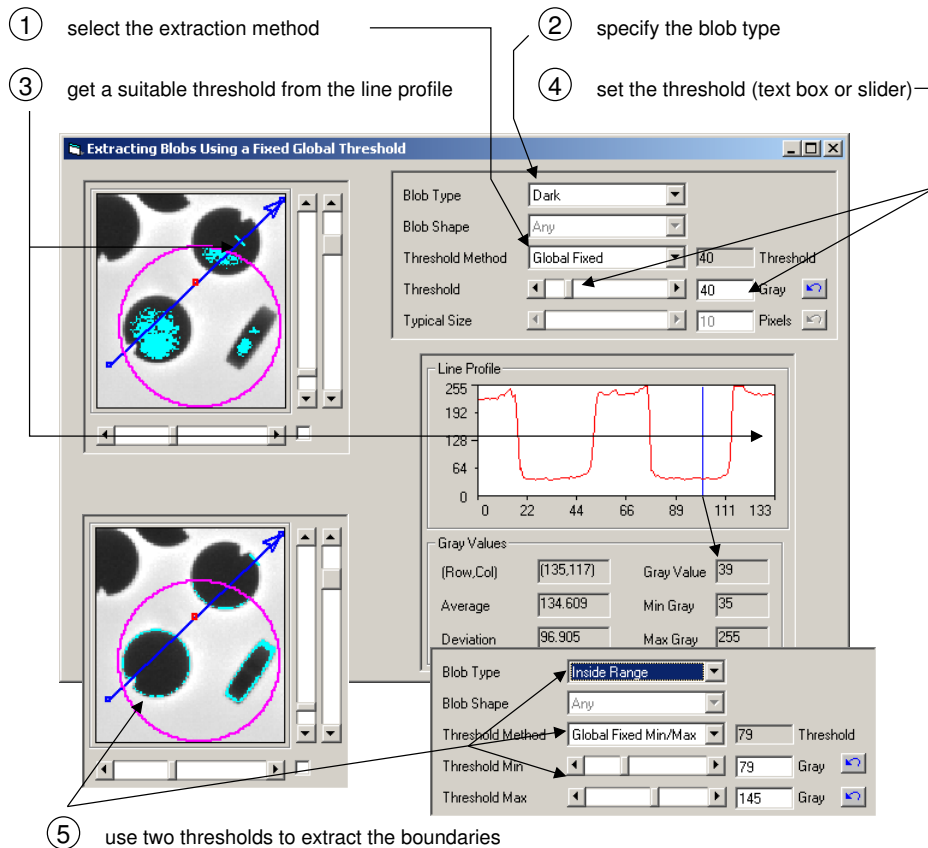


Figure 2.3: Using a fixed global threshold.

To get a first taste of the automatic thresholding described in the following chapter, select 'Global Auto' in the combo box Threshold Method. The computed threshold is displayed in the text box Threshold beside the combo box.

- ⑤ If you select 'Global Fixed Min/Max' in the combo box Threshold Method, you can work with two thresholds, a lower and upper one. In the combo box Blob Type you can now specify whether the gray values inside or outside the two thresholds correspond to the objects. Figure 2.3 shows how to extract the boundaries of the pills this way.

Note that all ROIs share the selected threshold(s). To use different thresholds for different ROIs you must create multiple instances of AVTBlobFinder.

### 2.3.2 Extracting Blobs Using Automatic Thresholding

ActivBlobFinder provides a method that automatically determines a suitable threshold for a region of interest by analyzing its content. As described in [chapter 1](#) on page 1, the underlying assumption is that two gray values dominate in the image, the gray value of the background and that of the blobs. This has two implications: First, ROIs must be placed with care; they must contain enough background and foreground to fulfill the assumption. Secondly, for each new image ActivBlobFinder will analyze the content of ROIs again; if the content does not change, this will waste processing time. In such a case it might be better to let ActivBlobFinder determine the threshold once and then use it as a fixed threshold.

On the other hand, automatic thresholding is very useful if the illumination changes from image to image. This will be illustrated in the following with an image sequence containing two images of a bar code under different illumination.

#### Visual Basic Example

Preparation for the following example:

- If you worked on the previous example, you may continue using this project.
- Otherwise, open the project `extract_auto\blob_extract_auto.vbp` and execute it (Run > Start or via the corresponding button).
- Open AVTViewFG by clicking into AVTView with the right mouse button and selecting Image Acquisition in the popup menu. Select the image sequence `barcode\barcode2.seq` in the combo box Input File. If necessary, create an ROI and place it over the bar code.

The following steps are visualized in [figure 2.4](#).

- ① Select 'Global Auto' in the combo box Threshold Method.
- ② In the combo box Blob Type, select whether the blobs to be extracted are darker or lighter than the background; in our example, select 'Dark'.
- ③ Automatically, a suitable threshold is computed and displayed in the text box Threshold. If AVTBlobFinder failed to determine a threshold, the value defaults to 128 and is marked in red. Experiment by modifying the ROI.
- ④ You can modify the threshold by specifying an offset with the slider Offset or by specifying a value in the text box beside it.

Note that the offset is valid for all ROIs, whereas AVTBlobFinder computes a suitable threshold for each ROI separately. The text box Threshold displays the actual threshold, i.e., computed threshold plus offset, for the active ROI.

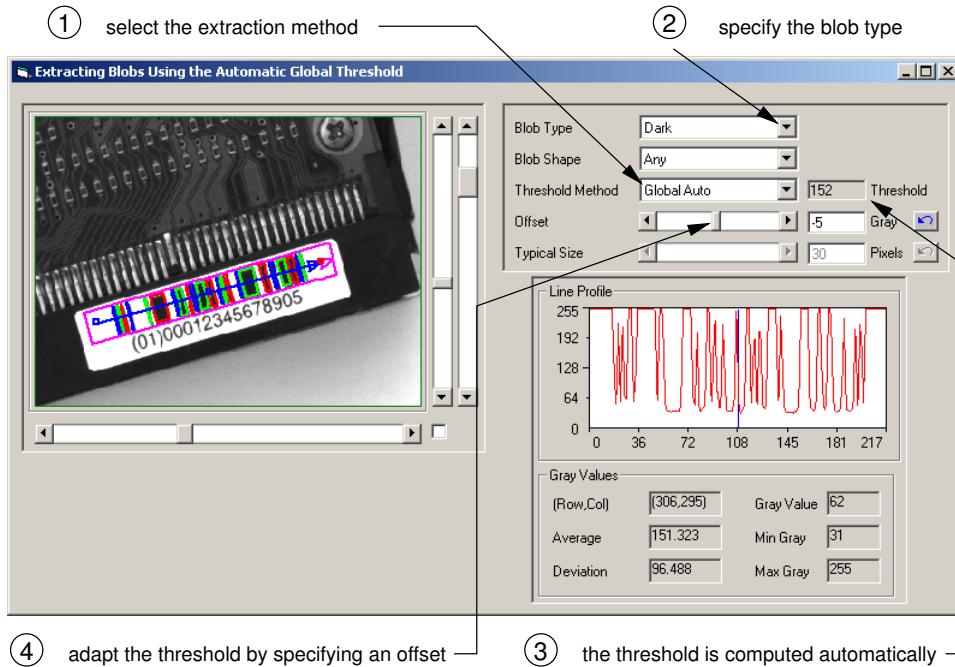


Figure 2.4: Using automatic thresholding.

Now step through the image sequence by clicking `Single` in `AVTViewFG`. The threshold will be adapted automatically. For a closer look at what's happening create a line-shaped ROI for `AVTLineProfile` and inspect the gray value profile.

As described in [chapter 1](#) on page 1, a slightly different method is used when characters are to be extracted: It “only” determines the dominating gray value corresponding to the background leaving it to you to determine a suitable offset. To experiment with this method, select the image `misc\characters_01` in the combo box `Input File` and place an ROI over some of the characters. Then, select 'Characters' in the combo box `Blob Shape` of `AVTBlobFinder`. Vary offset and ROI and watch what happens.

### 2.3.3 Extracting Blobs Using a Local Threshold

Both extraction methods described in the previous sections will fail when pixels cannot be classified by their gray value alone. This is the case, e.g., when the gray value of the background changes over the image because of a unregular illumination. Such a situation is depicted in [figure 2.5](#); see the gray value profile corresponding to the horizontal line.

However, if the blobs differ *locally* from the background (like the characters in the example), they still can be extracted using the local thresholding method.

#### Visual Basic Example

Preparation for the following example:

- If you worked on the previous example, you may continue using this project.
- Otherwise, open the project `extract_local\blob_extract_local.vbp` and execute it (Run ▷ Start or via the corresponding button).
- Open `AVTViewFG` by clicking into `AVTView` with the right mouse button and selecting Image Acquisition in the popup menu. Load the image `misc\characters_02`. If necessary, place an ROI over the lower line of characters.

The following steps are visualized in [figure 2.5](#).

- ① Select 'Local Neighborhood' in the combo box Threshold Method.
- ② In the combo box Blob Type, select whether the blobs to be extracted are darker or lighter than the background; in our example, select 'Dark'. Furthermore, specify their shape in the combo box Blob Shape (in the example: 'Characters').
- ③ Now, create a line-shaped ROI for `AVTLineProfile` and place it vertically over a character. Inspect the gray value profile and determine the *local contrast* of the characters by moving the cursor over it and measuring the difference of gray values between background and character (see [figure 2.5](#)). Enter this value in the text box beside the slider Contrast.
- ④ Similarly, determine the height of a character (see [figure 2.5](#)) and enter a slightly larger value in the text box beside the slider Typical Height.

Experiment with different values using the sliders and watch the effect. The local thresholding extracts all parts in the ROI which show a local contrast larger than the value you specified. The value you specified for the typical height influences the maximum size of the extracted blobs. From larger blobs, only the border is extracted.

If you want to extract blobs that are approximately line-shaped, i.e., have an almost constant



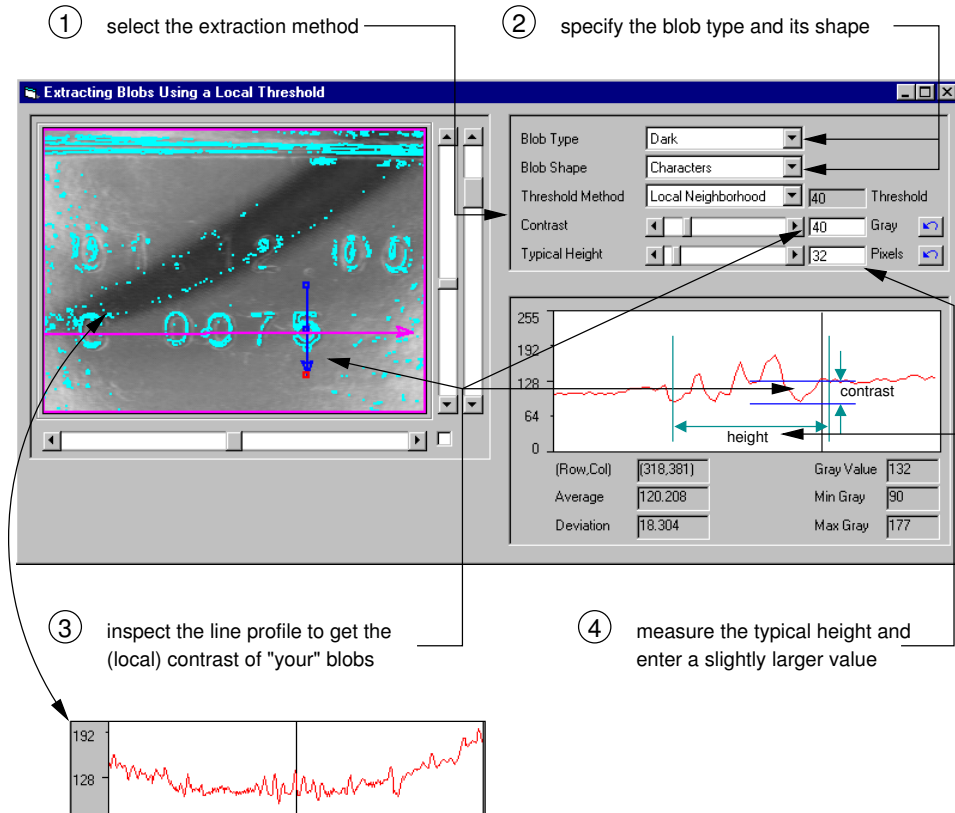


Figure 2.5: Using a local threshold.



width, select 'Lines' in the combo box Blob Shape and enter a value slightly larger than their width in the text box beside the slider now labelled Typical Width. If the blobs do not resemble lines or characters, select 'Any' and specify their maximum extent.

## 2.4 Processing Extracted Blobs Using


Using AVTBlobFinderProcess, you can process the extracted blobs further, e.g., fill up holes or smooth their contour. Suitable parameters can again be determined using an *inspection tool*, in this case ActivZoom.

### Visual Basic Example

#### Preparation for the following example:

- ❑ If you worked on the previous example, you may continue using this project. At design time, delete AVTLineProfile and add AVTBlobFinderProcess and AVTZoom by double-clicking  and .
- ❑ Otherwise, open the project processing\blob\_processing.vbp.
- ❑ Execute the application (Run > Start or via the corresponding button). Open AVTViewFG by clicking into AVTView with the right mouse button and selecting Image Acquisition in the popup menu. Load the image misc\characters\_03. If necessary, place an ROI over the characters.

The following steps are visualized in [figure 2.6](#). Please note that the purpose of the example is to demonstrate the processing capabilities, not to show how to extract the characters.

- ① You can combine all extracted blobs into a single one by checking  Combine to Single Region. This can be useful, e.g., to calculate features like the overall area.
- ② You can discard blobs by specifying a minimum and/or maximum size. The default value for the maximum size, 100001, signals that there is no upper boundary for the size.
- ③ To inspect a part of the image, point the mouse at it. In AVTZoom, you can then inspect the gray value of single pixels (text box Gray Value) or “measure” lengths or areas by counting pixels in the display or by consulting the position displayed in the text boxes Row and Column.
- ④ In AVTBlobFinderProcess, you can change parameters via the sliders or by entering values in the corresponding text boxes.
- ⑤ To reset parameters to their default value click .
- ⑥ To fill gaps between blobs, measure their width and enter a slightly larger value in the text box beside the slider Fill Gaps. Note that by closing a gap blobs are fused into a single blob. As a side effect, holes that are smaller than the specified width are closed as well.

① combine extracted blobs into a single one      ② specify an minimum and maximum size

③ inspect pixel values or "measure" lengths or sizes by counting pixels

④ change parameter

⑤ reset to default value

⑥ **filling gaps:** gap width = ca. 6 pixels => set parameter to '7'

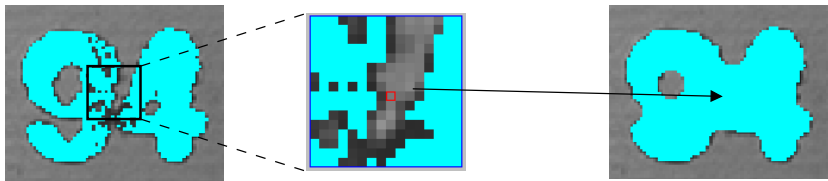


Figure 2.6: Processing blobs.

We continue with the example on the next double page.

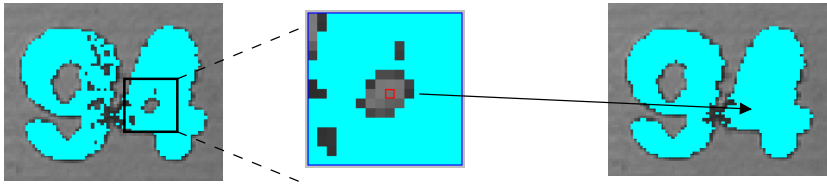
## (Processing Extracted Blobs, continued)

The following steps are visualized in [figure 2.7](#).

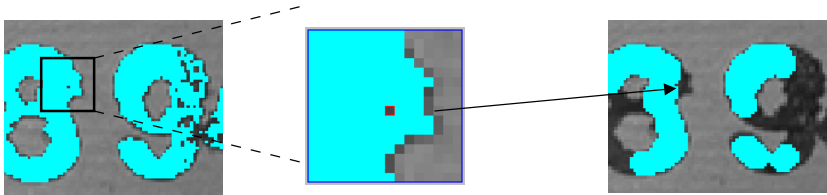
- ⑦ To close holes, measure their size (area) and enter a slightly larger value in the text box beside the slider `Fill Holes`.
- ⑧ Using the slider `Eliminate Bulge` you can remove bulges from the contour. The value specified for this parameter corresponds to the width of a bulge.  
Note that large values can have dramatic effects, e.g., remove whole parts of a blob. If a blob has a hole, bulges are removed from the inner contour as well.
- ⑨ Complementary to removing bulges you can also fill indentations on the contour via the slider `Fill Indentation`. This operation is similar to filling gaps with one difference: Gaps between blobs are not closed, i.e., blobs are not fused at this stage. Note that blobs can be differentiated easily if displayed with alternating colors (see [section 4.1](#) on page 34).

Note that all ROIs share the selected parameter values. To apply different processing steps in different ROIs you must create multiple instances of `AVTBlobFinder`.

- ⑦ **filling holes:** size (area) of hole = ca. 20 pixels => set parameter to '21'



- ⑧ **removing bulges:** width = ca. 6 pixels => set parameter to '7'



- ⑨ **filling indentations:** gap width = ca. 6 pixels => set parameter to '7'

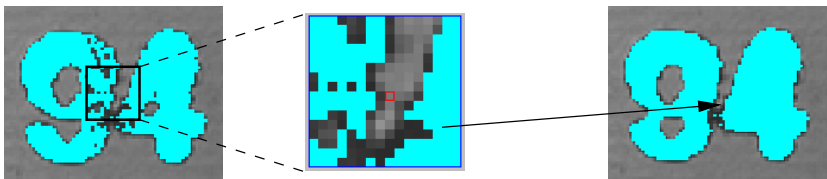


Figure 2.7: Processing blobs (cont.).



## Chapter 3

# Combining ActivBlobFinder with other ActivVisionTools

---

While the previous chapter explained how to extract the blobs you look for, this chapter focuses on how to further analyze them and how to evaluate and output the results using other ActivVisionTools.

In the corresponding example Visual Basic projects, the task is to inspect chips and check the presence of certain elements.

<b>3.1 Analyzing Blobs</b> . . . . .	<b>26</b>
<b>3.2 Evaluating Results</b> . . . . .	<b>28</b>
<b>3.3 Output of Results</b> . . . . .	<b>30</b>

### 3.1 Analyzing Blobs Using

The previous chapter showed how to extract blobs. In fact, this is all you can do with ActivBlobFinder. To analyze the extracted blobs you need another ActivVisionTool: ActivFeatureCalc. With this tool, you can select various features to be calculated for each blob; please refer to the [User's Manual for ActivFeatureCalc](#) for detailed information. These numeric results can then be further evaluated using ActivDecision, as will be described in the following section.

The task is to extract and analyze the four black squares at the corners of the chip.

#### Visual Basic Example

##### Preparation for the following example:

- We recommend to open the project `analyzing\blob_analyzing.vbp`, as it is already setup suitably to extract the squares.
- Execute the application (Run ▷ Start or via the corresponding button). Open AVTViewFG by clicking into AVTView with the right mouse button and selecting Image Acquisition. Load the image `chip\chip_02`.



**There is one thing to note on AVTFeatureCalc**, the master tool of ActivFeatureCalc: This ActiveX control does not have a graphical user interface; thus, it is represented by its icon at design time and invisible at run time. If you forget to add it to the form and only add the support tools, they are disabled (and do not calculate any features).

The following steps are visualized in [figure 3.1](#).

- ① While experimenting with parameters for the blob extraction itself it is useful to switch off the update of results in AVTDataView via the corresponding check box to speed up the processing.
- ② You can select features to be calculated by checking their box.
- ③ To view the results calculated by ActivFeatureCalc in ActivDataView, you first have to tell ActivDataView whose results to display by selecting the corresponding tool name in the combo box `ActivVisionTool1`. If there is only one tool, as in our example, it is selected automatically.
- ④ Automatically, a list of the ROIs of AVTFeatureCalc1 appears below. Select the ROIs whose results you want to examine by checking their box. Their results are then displayed in a table, the columns corresponding to the selected features. As you can see, the dark blob in the middle of the chip can be distinguished clearly from the four squares.



1 switch off updating while experimenting with parameters!

2 select features to be calculated

3 select AVTFeatureCalc1 in the combo box

4 select the ROIs you are interested in

Object Id	Area	Major Extent	Minor Extent	Compactness
0	464.000	23.000	22.000	1.989
1	362.000	22.000	20.000	2.288
2	421.000	44.000	25.000	11.217
3	456.000	22.000	21.000	1.572
4	478.000	22.000	22.000	1.720

Figure 3.1: Calculating and displaying blob features.


## 3.2 Evaluating Results Using

In the former examples, you have employed ActivVisionTools to extract and display blobs and additional features. Using ActivDecision, you can evaluate these results by formulating *conditions* the results have to meet in order to be “okay”. For a detailed description of ActivDecision please consult the [User’s Manual for ActivDecision](#).

We continue with the example task of checking chips for the presence of the four black squares. To be accepted as a square, the major extent of a blob has to lie between 20 and 25, the minor extent between 18 and 23.

### Visual Basic Example

#### Preparation for the following example:

- If you worked on the previous example, you may continue using this project. At design time, add AVTDecision to the form by double-clicking  with the left mouse button. You may delete AVTDataView and AVTFeatureCalcBasic; they can be opened via the right mouse button of AVTBlobFinder.
 

Otherwise, open the project decisions\blob\_decisions.vbp.
- Execute the application and load the image sequence chip\chip1.seq.

The following steps are visualized in [figure 3.2](#) (not shown: AVTView and AVTBlobFinder).

- ① The main functionality of ActivDecision is presented by its *support tools*, which can be opened via clicking on AVTDecision with the right mouse button. The *master tool* displays the overall evaluation of the application.
- ② To view the current feature and evaluation results check  Enable Update in AVT-DecisionViewResults.
- ③ ActivDecision lets you compare the value of an individual object, ROI, or tool feature with two boundary values, a minimum and a maximum value. You can formulate conditions for features by specifying values in the columns Min and Max and selecting a comparison mode in the column Operation. If you select None, the feature is not evaluated. [Figure 3.2](#) shows suitable conditions for the example task.
- ④ Those features which meet their condition appear in green, the others in red. If at least one feature is “not okay”, the whole object, ROI, or tool is evaluated as “not okay” as well. Analogously, the overall evaluation of application, which is visualized by AVT-Decision, depends on the tool evaluations.

Step through the image sequence and examine the evaluations. In some of the images, an additional blob is extracted which does not meet the condition; in other images, the

① open the support tools via the context menu (right mouse click)

② first, enable the update of results

③ formulate conditions for objects, ROIs or tools

④ the evaluations are displayed immediately

⑤ specify default conditions

⑥ check this box to show the used parameters

Tool	ROI	Object	Name	Value	Min	Max	Interpretation	Operation
AVTFeatureCalc1			Objects Tool	5	0	10000	Number	None
			Good Objects Tool	4	0	10000	Number	None
			Bad Objects Tool	1	0	10000	Number	None
			Good ROIs	1	0	10000	Number	None
			Bad ROIs	0	0	0	Number	= Max
			Objects ROI	5	4	10000	Number	>= Min
			Good Objects ROI	4	4	4	Number	= Max
			Bad Objects ROI	1	0	0	Number	None
			Major Extent	23.000	20.000	25.000	Number	Inside
			Minor Extent	22.000	18.000	23.000	Number	Inside

Tool	ROI	Object	Name	Min	Max	Interpretation	Operation
AVTFeatureCalc1			Objects ROI	4	10000	Number	>= Min
			Good Objects ROI	4	4	Number	= Max
			Bad Objects ROI	0	0	Number	None
			Major Extent	20.000	25.000	Number	Inside
			Minor Extent	18.000	23.000	Number	Inside
			Major Extent	Default	Default	Default	Default
			Minor Extent	Default	Default	Default	Default

Figure 3.2: Formulating conditions to evaluate results.

squares are missing.


- ⑤ In the example application, all squares should meet the same condition. Instead of specifying the same conditions for each object, you can specify default conditions using `AVTDecisionViewDefaults`. Defaults can be set per tool or per ROI; ROI defaults override tool defaults, and individual conditions override defaults.
- ⑥ If you check  `Substitute Default`, the entries marked `Default` are substituted by their actual content.

### 3.3 Output of Results Using

Using ActivFile, you can write the results and the evaluations to a log file. How to access results and evaluations via the programming interface is described in the User's Manual for ActivFeatureCalc, [section 4.2](#) on page 34, how to output them via a serial interface or a digital I/O board in the [User's Manual for ActivSerial](#) and the [User's Manual for ActivDigitalIO](#), respectively.

#### Visual Basic Example

##### Preparation for the following example:

- If you worked on the previous example, you may continue using this project. At design time, add AVTOutputFile by double-clicking  .  
Otherwise, open the project output\blob\_output.vbp.
- Execute the application and load the image sequence chip\chip1.seq.

The following steps are visualized in [figure 3.3](#) (not shown: AVTView, AVTBlobFinder, AVT-Decision).

- ① By clicking on **Select**, you can open a file selector box to choose a file name for the log file, which will then appear in the text field beside the button. By pressing **Clear File**, you can clear the content of the selected file.
- ② By checking  **Enable Writing** you enable the writing mode.
- ③ You can open the ActivFile's two dialogs **DialogFileOptions** and **DialogOutputDataSelect** by clicking **File Options** and **Data Selection**, respectively.
- ④ In **DialogFileOptions**, you can choose between two file formats: Standard text files (suffix .txt) and the so-called *comma-separated values* (suffix .csv) which can be used as an input to Microsoft Excel. Furthermore, you can select a delimiter.
- ⑤ In the same dialog you can limit the size of the log file in form of the number of *cycles* that are to be recorded. A cycle corresponds to one processing cycle from image input to the evaluation and output of results. If you use this option, ActivFile creates two log files and switches between them, thus assuring that you can always access (at least) the results of the last  $N$  cycles,  $N$  being the specified number of cycles.
- ⑥ By pressing **Estimate**, you can let ActivFile estimate the size of one cycle. Note that you must first select the output data in order to get meaningful results!
- ⑦ In the left part of **DialogOutputDataSelect**, you can navigate through the result hierarchy similarly to ActivDecision.

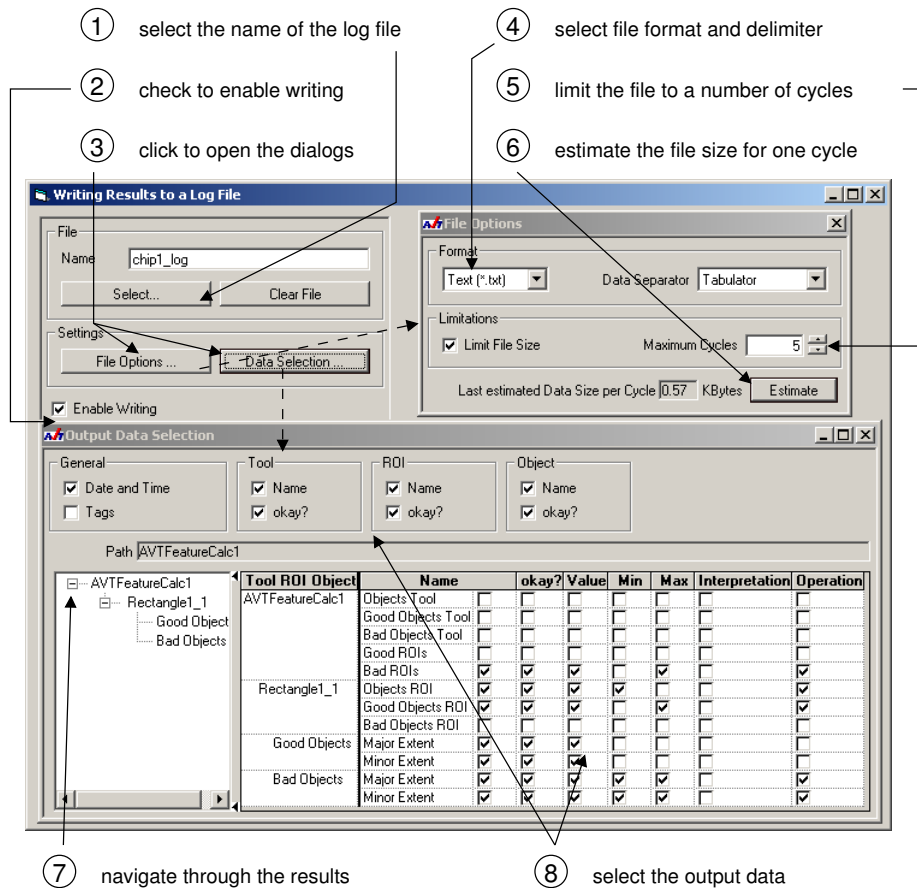


Figure 3.3: Customizing log files.

- ⑧ In the right part of DialogOutputDataSelect, choose the output data by checking the corresponding boxes. You may output different items depending on the evaluation of an object. By clicking on the column labels with the right mouse button you can check or uncheck all boxes in the column; similarly, you can check or uncheck whole rows or all rows of a certain tool.

If you now step through the image sequence by clicking `Single` in AVTviewFG, the log file is created. Figure 3.4 shows part of an example log file.

```

09/12/04 18:57:58
AVTFeatureCalc1 yes
  Bad ROIs          yes  0  0  = Max
  Rectangle1_1     yes
    Objects ROI     yes  5  4  >= Min
    Good Objects ROI yes  4  4  = Max
  0 yes
    Major Extent    yes  24.000000
    Minor Extent    yes  23.000000
  1 yes
    Major Extent    yes  23.000000
    Minor Extent    yes  21.000000
  2 no
    Major Extent    no  45.000000  20.000000  25.000000  Inside
    Minor Extent    no  26.000000  18.000000  23.000000  Inside
  3 yes
    Major Extent    yes  23.000000
    Minor Extent    yes  22.000000
  4 yes
    Major Extent    yes  22.656562
    Minor Extent    yes  22.229080

09/12/04 18:57:59
AVTFeatureCalc1 no
  Bad ROIs          no  1  0  = Max
  Rectangle1_1     no
    Objects ROI     no  0  4  >= Min
    Good Objects ROI no  0  4  = Max

```

Figure 3.4: Part of an example log file.

# Chapter 4

## Tips & Tricks

---

This chapter contains additional information that facilitates working with `ActivBlobFinder`, e.g., how to modify the graphical display of results and how to customize the appearance of an `ActivBlobFinder` application in the two execution modes.

<b>4.1</b>	<b>Adapting the Display of Results</b>	<b>34</b>
<b>4.2</b>	<b>Configuring the Two Execution Modes</b>	<b>36</b>

## 4.1 Adapting the Display of Results Using

You can adapt the way results and ROIs are displayed using `AVTViewDisplayModes`, which is a *support tool* of `ActivView`.

### Visual Basic Example

#### Preparation for the following example:

- If you worked on the example in the previous chapter, you may continue using this project.  
Otherwise, open the project `display/blob_display.vbp` and execute it (Run > Start or via the corresponding button).
- Load the image sequence `chip\chip1.seq` and create additional ROIs; `AVTView-FG` and `AVTViewROI` can be opened at run time via a right mouse button click on `AVTView`.

The following steps are visualized in [figure 4.1](#). Experiment by choosing different settings for the display parameters and watching the result.

- ① Open `AVTViewDisplayModes` by clicking on `AVTView` with the right mouse button and selecting `Display Modes` in the popup menu.
- ② Change the color of the selected pick point. Select another pick point by clicking into its vicinity.
- ③ Change the color of the currently selected ROI. Select another ROI by clicking into its vicinity. Change the color of the other, not selected ROIs of the currently selected tool. Change the color of the ROIs of the other, not selected tools. Select another tool in the combo box `ActivVisionTool` of `AVTViewROI` (if this box contains more than one item). If you use `ActivDecision` to evaluate results, it can mark ROIs evaluated as “not okay” in a special color.
- ④ Change the LUT (look-up table) that is used to display the image.
- ⑤ Change the line width or the ROIs (or of the blob margin, see below).
- ⑥ Change the color of the blobs in the text box `Group I`.
- ⑦ Change the color `ActivDecision` uses to mark objects evaluated as “not okay”.
- ⑧ Change the color used for highlighting objects if you click on them in `ActivDataView` or in the tree of `ActivDecision`.
- ⑨ Change the way the extracted blobs are displayed (filled or margin only).



① open AVTViewDisplayModes via a click on AVTView with the right mouse button

The screenshot shows the 'ActivViewDisplayModes' dialog box with the following settings:

- Display Region of Interest:** Active ROI (Blue), Error ROI (Red), Selected Tool (Yellow), Inactive Tools (Gray), Special (Magenta), Pick Point (Red), Line Width (1 Pixel).
- Display Image:** LUT (default).
- Display Results:** Group I (Green), Group II (Cyan), Error (Red), Selected (Yellow), Display Mode (Margin), Line Width (2 Pixels), Text (Blue).
- Other Settings:** MS Sans Serif (12pt), Text Background (unchecked), Contrasting Contours (unchecked).

② color of the selected pick point

③ color of the different types of ROIs

④ look-up table for image display

⑤ line width of the ROIs or of the contour of the blobs

⑥ color of the blobs

⑦ color of objects that are not okay

⑧ color of selected objects

⑨ drawing mode for the blobs



Figure 4.1: Adapting the display of ROIs and results.

## 4.2 Configuring the Two Execution Modes Via and

In an ActivVisionTools application you can switch between two execution modes: the *configuration mode* and the *application mode*. The former should be used to setup and configure an application, the latter to run it. ActivView's *support tools* AVTViewExecute and AVTViewConfigExec allow you to switch between the two modes and to customize the behavior of an ActivVisionTools application in the two execution modes, e.g., display live images only in the *configuration mode* to setup your application, but then switch it off in the *application mode* to speed up the application. A third sub-tool, AVTViewExecuteSimple, provides a single button to start/stop the application. With the help of AVTViewStatus, another *support tool* of ActivView, you can inspect the current status of your application.

### Visual Basic Example

#### Preparation for the following example:

- If you worked on the previous example, you may continue using this project. At design time, add AVTViewExecuteSimple and AVTViewStatus to the form by double-clicking the icons  and  with the left mouse button.  
Otherwise, open the project usermodes\blob\_usermodes.vbp.
- Execute the application and load the image sequence chip\chip1.seq.

The following steps are visualized in [figure 4.2](#) (not shown: AVTBlobFinder).

- ① Open AVTViewExecute and AVTViewConfigExec by clicking on AVTView with the right mouse button and selecting Execution and Execution Parameters.
- ② Switch between the two execution modes via AVTViewExecute's combo box Mode.
- ③ To execute one cycle, press . With the other two buttons you can let the application run continuously and stop it again.

By default, AVTViewExecuteSimple starts and stops an application; how to change its behavior to a single-step button is described in the User's Manual for ActivView, [section 3.4](#) on page 34.

- ④ For each of the two execution modes, you can choose what is to be displayed by checking the corresponding boxes in AVTViewConfigExec. Furthermore, you can specify if images can be dragged to the image window and whether ROIs can be modified in the two modes; by default, this is disabled in the *application mode* to prevent you from accidentally moving or deleting an ROI.
- ⑤ In AVTViewStatus, an icon indicates the current execution mode of the application.

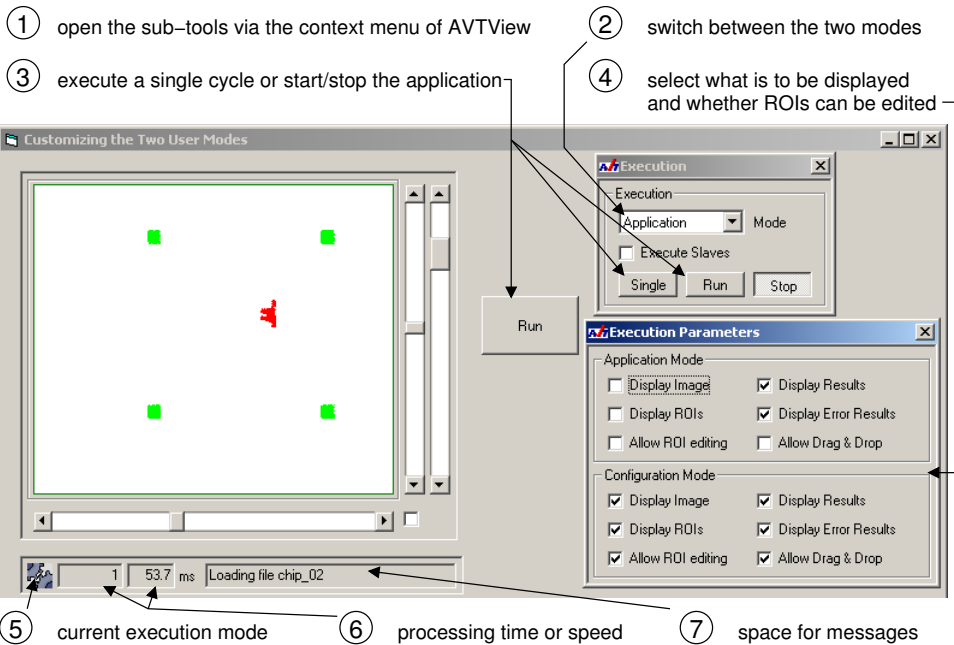



Figure 4.2: Customizing and switching between the two execution modes.

In the mode , the application does not perform any processing and waits for your interaction. If you start the continuous mode the cogwheels rotate; any interaction on your part is stored in the event queue and processed after the current cycle is finished. If the cursor gets “busy” the ActivVisionTools have started a particularly time-consuming operation, e.g., connecting to an image acquisition device. Any interaction on your part is then deferred to the end of this operation.

- ⑥ AVTViewStatus also shows the number of processed cycles and the time needed for the last processing cycle.
- ⑦ AVTViewStatus display two types of messages: Informative messages describe, e.g., what the application is doing while it is “busy”, while error messages indicate errors that prevent the application from working correctly, e.g., the failure to connect to an image acquisition device. If AVTViewStatus is not added to an application, error messages are displayed in popup dialogs.

More information about AVTViewStatus, e.g., how to modify its appearance, can be found in the User’s Manual for ActivView, [section 3.3](#) on page 32.