19.1 Introduction

Enzyme linked immunospot assay (ELISPOT) (Gebauer et al., 2002; Hricik et al., 2003) is a powerful technique used as a non-invasive diagnostic tool for the prediction of long term renal allograft function and the early detection of chronic graft rejection process markers (Koscielska-Kasprzak et al., 2009). It focuses on the detection and quantification of antigen specific immunological responses at a single cell level. The ELISPOT experiment (Fig. 19.1) is performed in a membrane bottomed 96-well plates with each well bottom (6 mm diameter) coated with antibodies specific against each of the cytokines analyzed.

During the experiment the suspension with a known number of recipient lymphocytes is placed in the wells. To selected wells inactivated donor splenocytes are added. Cytokines secreted by activated recipient lymphocytes are captured in the immediate vicinity of the cells by the specific antibody bound to the membrane. After the removal of the cells, the presence of the exuded cytokine is detected with the use of the enzyme immunoassay reaction. As a result of the enzyme-catalyzed conversion of chromogen to an insoluble stain, spots, seen in the image, are formed in places prior to test the presence of cytokine-secreting cells.
ELISPOT assay can be used for the simultaneous analysis of two different cytokine secretions, resulting in the two-color images consisting of spots of the following colors:

- purple-blue, as a result of the presence of the reaction of the substrate BCIP/NBT catalyzed by alkaline phosphatase;
- bricky-red derived from the oxidation of the AEC substrate by peroxidase in the presence of H₂O₂;
- transient color.

The procedures for preparing cells for two-color ELISPOT test did not differ from those for a standard single color tests. Special optimization requires only an examination, which is based largely on commercial sets of antibodies and reagents.
19.2 Characteristics of the images

Viewed under a microscope, an image was photographed with one of the two available acquisition devices:

- a digital camera Nikon E4500 giving a resolution of 2272×1704 pixels and JPEG image format;
- a motion digital camera Nikon DS5M-U1 giving a resolution of 2560×1920 pixels for BMP and JPEG formats.

For both devices, acquiring the images entails some inconvenience that may have a significant effect on the correct operation of the image processing algorithm.
A still camera allows recording only in JPG format, which, despite a fairly high-resolution, introduces errors due to lossy compression.

![Image](image1.png)

**Fig. 19.5** Visualization of distortions caused by lossy compression.

In Fig. 19.5 the result of processing of the fragment of an image with an edge filter is depicted. Consequences of JPEG compression are visible as horizontal and vertical brighter lines. This distorts the process of detecting spots’ shapes and boundaries, which affects the accuracy of measurement of their size. Another disadvantage is the unequal exposure of the image because the device was not designed for this type of lighting. This defect may have an adverse effect on the proper identification of dichroic colors in images. The image has a relatively good resolution thanks to the effective focal length of the lens control. However, the depth of field is very poor, which means that it is not possible to set the correct focus around the area of interest. Fragments further away from the center are darker and slightly blurred.

A motion camera captures an image evenly exposed, with the correct white balance, uniform sharpness and allows it to record uncompressed images. However, due to the optical system used, an effective image resolution is twice as low as that of a still camera (3600 DPI vs 7200 DPI for a still camera). This means that the spot of a diameter of 0.03 mm has approximately 5 pixels. The segmentation algorithm must be sufficiently sensitive to detect objects of this size.

Other observable defects of the images are related to the production of ELISPOT preparations. Blue pigments can produce a much better contrast and color saturation than the brown pigment which hinders the simultaneous detection of both stainings.
19.3 Challenges of the image processing algorithm

The image processing algorithm should:
- detect the bounds of the bottom of the well as a circle which will be the region of interest in the image. Knowing that its diameter is 6 mm it is possible to obtain the scale
If we define a color image $I$ as a weighted graph:

$$I = (D, E, f) \text{ where } D: [1,..., M] \times [1,..., N] \subset N \times N$$

(19.1)

where $E$ defines the neighbourhood of its points and $f : D \rightarrow N \times N \times N$, the region of interest is defined as:

$$\text{ROI} = \{(x, y) : (x, y) \in I \land (x_0 - x)^2 + (y_0 - y)^2 \leq R^2 \}$$

(19.2)

- distinguish from the image the spots of types A, B and C (three colors) as connected components, and the background

$$B_g, I_{A1}, I_{A2}, ..., I_{B1}, I_{B2}, ..., I_{C1}, I_{C2}, ..., I_{C}$$

$$I = \bigcup_{k=1}^{s} I_k \cup B_g$$

(19.3)

$$\forall i, j \in \{1, ..., s\}, i \neq j \Rightarrow I_i \cap I_j = \emptyset$$

- the spots are approximately round ($AR > 0.4$) where $AR$ is defined as follows:

$$AR_i = \frac{\min_{p \in \text{int}(I_i)} \{\rho(C_i, p)\}}{\max_{p \in \text{int}(I_i)} \{\rho(C_i, p)\}}; \quad 0 < AR_i \leq 1;$$

(19.4)

$C$ – “gravity center” of the component

- the area of the individual spot is in the range $[0.03 \text{ mm}^2 - 0.1 \text{ mm}^2]$.

The segmentation may be hampered by the fact that the spots have fuzzy edges. Below (Fig. 19.6) are graphs of intensity distribution of the dye in the cross-section of the spot.

![Graphs of intensity distribution of the dye in the cross-section of the spot.](Q=1.00 \text{ sec}^{-1} Q=0.50 \text{ sec}^{-1} Q=0.25 \text{ sec}^{-1} Q=0.10 \text{ sec}^{-1})

Fig. 19.6 Exemplary sizes of the spots and distributions of dye intensity.
The subject of image analysis, and more specifically – quantitative analysis – is to obtain the area distribution of individual spots in the visual field and present the result in the form of histograms, separately for spots A, B and C. The area of the spot is an approximation of quantity of the secreted dye, but not the only one. To improve the accuracy, an additional measure, called the spot weight was introduced. Below is the definition for type A.

Let \( \{I_{A1}, I_{A2}, \ldots, I_{Ak}\} \) be the set of all segmented A-type spots. Denote by \( I \) the dye intensity of the image pixel. For monochrome images, it will be an inverse of the luminance, and for color images e.g. the color saturation. Let us denote \( L_{\min} = \min_{(x,y)\in I} \{I(x,y)\} \) and \( L_{\max} = \max_{(x,y)\in I} \{I(x,y)\} \) as extreme values of the intensity. Then the spot weight is defined by:

\[
W_k := \sum_{(x,y)\in I_k} \frac{L_{\max} - I(x,y)}{L_{\max} - L_{\min}} = \frac{n_k \cdot L_{\max} - \sum I(x,y)}{L_{\max} - L_{\min}}
\]

(19.5)

The equation allows one to scale the value in this way, which would be comparable to the area of the spot (\(W=S\) in the case of even intensity and perfectly sharp edges) (Fig. 19.7).

![Spot area distribution B5](image)

Fig. 19.7 Spot area vs. weighted area distributions for a sample image.

### 19.4 Preprocessing algorithms

#### 19.4.1. White balance correction

This operation is very important in the case of digital images. Color space transformation is linear, so there is no loss of image information. Let \( I=(I_R, I_G, I_B) \) be an input color image. Denote:

\[
\overline{R} = \{\overline{p} : p \in I_R\}, \quad \overline{G} = \{\overline{p} : p \in I_G\}, \quad \overline{B} = \{\overline{p} : p \in I_B\}
\]

(19.6)
Various Approaches to Processing and Analysis of Images

\[ \text{Max} R = \max \{ p \in I_R \}, \text{Max} G = \max \{ p \in I_G \}, \text{Max} B = \max \{ p \in I_B \} \]

\[ \text{Min} R = \min \{ p \in I_R \}, \text{Min} G = \min \{ p \in I_G \}, \text{Min} B = \min \{ p \in I_B \} \]

\[ \text{Max} I = \max \{ \text{Max} R, \text{Max} G, \text{Max} B \}, \text{Min} I = \min \{ \text{Min} R, \text{Min} G, \text{Min} B \} \]

We have the following cases. If \( R = \min \{ R, G, B \} \) then

\[
WR = \frac{256}{1 + \text{Max} R - \text{Min} I} ; \quad WG = \frac{256 \cdot R}{(1 + \text{Max} R - \text{Min} I) \cdot G} ; \\
WB = \frac{256 \cdot R}{(1 + \text{Max} R - \text{Min} I) \cdot B} .
\]

(19.7)

If \( G = \min \{ R, G, B \} \) then

\[
WR = \frac{256 \cdot G}{(1 + \text{Max} G - \text{Min} I) \cdot R} ; \quad WG = \frac{256}{1 + \text{Max} G - \text{Min} I} ; \\
WB = \frac{256 \cdot G}{(1 + \text{Max} G - \text{Min} I) \cdot B} .
\]

(19.8)

If \( B = \min \{ R, G, B \} \) then

\[
WR = \frac{256 \cdot B}{(1 + \text{Max} B - \text{Min} I) \cdot R} ; \quad WG = \frac{256 \cdot B}{(1 + \text{Max} B - \text{Min} I) \cdot G} ; \\
WB = \frac{256}{1 + \text{Max} B - \text{Min} I} .
\]

(19.9)

Otherwise (if none of the listed cases applies), all the mean values are equal:

\[
WR = WG = WB = \frac{256}{1 + \text{Max} I - \text{Min} I} .
\]

(19.10)

The output image \( I' = (I'_R, I'_G, I'_B) \) is generated as follows:

\[
I'_R = \{ p' = WR \cdot p - \text{Min} I; p \in I_R \} ; \quad I'_G = \{ p' = WG \cdot p - \text{Min} I; p \in I_G \} ; \\
I'_B = \{ p' = WB \cdot p - \text{Min} I; p \in I_B \} .
\]

(19.11)

19.4.2 Red color correction

The aim of the operation is the correction of the intensity of red spots, because its intensity is worse than in the case of blue spots. The input parameters will be
reference hue of red $H_0$, the tolerance of the hue $T_H$ and the gain amplitude $A$.

The steps are as follows:

Convert the image to HSV color space:

$$U : (I_R, I_G, I_B) \rightarrow (I_H, I_S, I_V)$$

Select pixels which are not background pixels, which means:

$$I_T = \{ p \in (I_H, I_S, I_V) : S > 0.28 \land V < 0.8 \}$$

For each pixel $p$ denote:

$$dH = \begin{cases} 
\frac{|H - H_0|}{360 - |H - H_0|} & \text{if}\; |H - H_0| < 180 \\
\frac{360}{360 - |H - H_0|} & \text{otherwise}
\end{cases}$$

if $dH < T_H$ then

$$V' = V \cdot \left(1 - \exp\left(-\frac{dH^2}{10 \cdot T_H}\right) \cdot \frac{A}{10}\right)$$

Convert the image back to RGB. For the parameters set experimentally $H_0 = 5$, $T_H = 30$ and $A = 3$ the algorithm corrects the red spots. It will not affect the hue of any pixel of the image.

### 19.4.3 Uneven exposure correction

The algorithm requires that the round region of interest (2) is yet specified. The steps of the algorithm:

Paint the rest of the image black

$$I_1 = \{ p : p \in \text{ROI} \cup (0,0,0) : p \in I \setminus \text{ROI} \}$$

For the obtained image do the averaging (a convolution filter using square mask $M$ with all 1s):

$$I_2 = I_1 * M$$

Make the subtraction of the images:

$$I_3 = I_2 - I_1$$

Paint the region outside the ROI white:

$$I_4 = \{ p : p \in \text{ROI} \cup (255,255,255) : p \in I_3 \setminus \text{ROI} \}.$$
The size of the square mask $M$ is set experimentally to $0.25 \cdot R$ (one fourth of the ROI radius).

19.5 Segmentation of monochrome and color images using color thresholding

The simplest method of segmentation of ELISPOT images is the selection of image pixels that belong to spots from all the image pixels. This type of segmentation is called pixel classification. The classification may be carried out by comparison of its color, intensity, brightness or any other property to a threshold value $\tau$. (Fig. 19.8). For instance, after thresholding the gray level image is converted to a binary one. There exist algorithms that use more than one threshold values (multithresholding), which enables one to assign pixels to one of a few classes instead of only two. Threshold value(s) may be entered manually or automatically. A thorough survey of automated image thresholding selection can be found in (Mehmet and Bulent, 2004). For the purpose of our research we selected three methods for the automatic selection of $\tau$ value for ELISPOT images. Originally intended for grayscale images, the method was adapted for color images.

The first of them is Bernsen’s method (Bersen, 1986), while two others are based on the analysis of a pixel color histogram. Bernsen’s method was applied with only slight modifications, while histogram-based methods had to be reimplemented.
19.5.1 Bernsen algorithm

In the Bernsen algorithm each pixel \((i, j)\) is considered with its surrounding (usually square) window \(W\)

\[
W_{ij} = \{(x, y) \in I : |y - i| < b \land |x - j| < b\} \quad (19.19)
\]

As a local threshold the mean value of the maximum and the minimum pixel intensity within \(W\) is taken:

\[
T(i, j) = 0.5[\max_w(I(i + m, j + n)) + \min_w(I(i + m, j + n))] = 0.5[I_{\text{high}}(i, j) + I_{\text{low}}(i, j)] \quad (19.20)
\]

Additionally, the contrast for the search window for pixel \((i, j)\) is defined:

\[
C(i, j) = I_{\text{high}}(i, j) - I_{\text{low}}(i, j) \quad (19.21)
\]

If \(C(i, j)\) is not sufficient (the experimentally set threshold value – usually for images of global contrast 255, it is set as 15) the threshold value from (20) is replaced by a global value \(\tau_0\). This value must be set a priori. Although it may be set manually, we decided to obtain it by simple image statistics (SIS) (Kittler et al., 1985), the algorithm for adaptive threshold selection.

Let us assume that the perfect image presents objects with intensity \(a\) over the background with intensity \(b\), and because of some light and material non-uniformity, the intensity values are to some extent distorted – the noise is introduced to image pixels. Despite this, the best threshold that discriminates the objects from the background is

\[
\tau = (a + b)/2.
\]

For each image point \(p(i, j)\) with intensity \(l(i, j)\), let us define its gradient module:

\[
e(i, j) = |\nabla p| = \max \{||p(i, j) - l(i + 1, j)||, ||p(i, j) - l(i, j + 1)||\} \quad (19.22)
\]

Then we may set the optimal threshold value as

\[
\tau_0 = \frac{\sum_{i=1}^{m} \sum_{j=1}^{n} ||p(i, j) - e(i, j)||}{\sum_{i=1}^{m} \sum_{j=1}^{n} e(i, j)} \quad (19.23)
\]

The Bernsen algorithm has some drawbacks. If the image contains some impulse noise (Fig. 19.9) (isolated pixels of a different color) applying formula (19.20) can give unexpected results.
In such a case the estimated contrast may be rather high and the region under the moving window is supposed to be nonhomogeneous. In fact, all the pixels of this fragment of the image belong to the background. But the Bernsen algorithm considers this bright pixel as a background, and its neighborhood as an object. To avoid this phenomenon, the standard deviation was used as a measure of color homogeneity.

\[
\sigma = \sqrt{\frac{\sum_{i,j} ((l(i,j) - \mu)^2)}{N}} \quad (19.24)
\]

Regarding the manner of local threshold selection, Bernsen’s method has another significant drawback. The window \(W\) size should fit the size of expected objects (shapes of fairly uniform intensity) in order to perform the correct segmentation; thus it must be set experimentally before running the actual Bernsen routine. If the image contains objects of a wide size range or the object scale is unknown, the segmentation fails.

With too large a window the method works slowly and tends to misclassify small objects with low contrast to the background, especially when they are located near bigger objects with higher contrast. The small window range can cause incorrect border detection of large objects and incorrect identification of large objects.

To eliminate this phenomenon in our research, we introduced a multipass version. In each iteration, the size of \(W\) is increased so that both small and larger spots are detected.
19.5.2. **Histogram analysis**

Both examined histogram-based methods that have been tested against ELISPOT images analyze the extreme values of a brightness distribution function, which are maxima (peaks) and minima (valleys).

![Maxima and minima in the brightness distribution function.](image1)

The algorithm searches for a sequence of three points: maximum - minimum - maximum. The location of such groups suggests that there are two classes of objects with different brightness and gives a hint how to set a threshold value, for instance:

$$\tau = \frac{\max_1 + \max_2}{2}$$  \hspace{1cm} (19.25)

The first of these methods is called “Peaks & Valleys (valley)” because the value of the threshold is set as a point where a local minimum is found. In the second method “Peaks & Valleys (half)”, the threshold is set according to formula (25).

![An examined fragment of the image.](image2)

In Fig. 19.11 a fragment of an exemplary ELISPOT image is presented and in Fig. 19.12 the histograms of its brightness.
For this fragment of the image none of peaks & valleys variant is effective.

A solution may be filtering the image using algorithms from section 4.

19.5.3. Tests of the algorithms

The results were developed for all three methods and for three images. The images test1, test2 and test3 are filtered with an uneven exposure correction filter.
For all three algorithms the window size was set to 15 pixels, 75 pixels and a two-pass version was run.
All three methods (if appropriate parameters have been set) give very similar and satisfactory results. The resulting images helped to see that the use of a small window (15x15) is suitable for small spots, while the bigger ones are of poor quality. The use of a larger window (75x75) resulted in a significant improvement of this deficiency; however, this window coped worse with the detection of very small objects. A trade-off is to carry out two runs with two windows. Unfortunately, this operation is time-consuming.

19.6 Segmentation of color images using color clustering

Clustering (Hanson and Riseman, 1978) is a technique which requires defining a feature space and mapping the image pixels to vectors in this space. With the use of statistical methods, the feature space is split into connected regions (clusters), which implies grouping the pixels of similar color to a number of classes.

One of the most commonly used clustering algorithms is $k$-means (MacQueen, 1967). The most important parameter for this algorithm is $k$, the number of clusters in a feature space (here: color space) to be found. A fixed value of $k$ may be a drawback of the algorithm if the predicted number of classes is unknown. The algorithm must start from initializing $k$ centers of the clusters – the initial values may be set manually or chosen randomly. In each iteration, each point is attracted to the nearest centroid and the centroids are updated (as a mean value of all points belonging to one class). The algorithm presented below uses clustering and thresholding techniques.

The algorithm operates in four phases: initialization (Fig. 19.17), teaching, grouping and finalization.

1. Get segmentation params and the coordinates of ROI
2. Initialize the class matrix for image pixels
3. Load the image
4. Set the ROI

Fig. 19.17 The initialization phase.

An image to be processed is divided into square regions using specified masks (Fig. 19.18).
This helps to cope with uneven lighting and varying focal conditions in the entire image. The mask for teaching phase is bigger. Teaching and grouping phases work separately on the square regions.

The following parameters are set by the user at the start:

- the number of classes, diameters of scanning masks (separately for teaching and grouping phases);
- the gray-level thresholds (the level of homogeneity which controls the sensitivity of the algorithm and the level of spot which affects the spot sharpness);
- the method of initialization of centroids;
- optionally – post segmentation median filtering.

In this stage, a class matrix for the image pixels is allocated.

In the next – teaching phase – the gravity centers for each class are initialized and pixels are initially classified. Those centers are generated on the basis of one of three methods: global gray-level, global feature level, local feature level.

The global gray level is a method where the centers are initialized based on the maximum and minimum levels found in the whole region of interest (Table 19.1).
Various Approaches to Processing and Analysis of Images

Table 19.1  Centroid initialization – global gray level.

<table>
<thead>
<tr>
<th>centroid</th>
<th>R</th>
<th>G</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>maxgray</td>
<td>maxgray</td>
<td>maxgray</td>
</tr>
<tr>
<td>1</td>
<td>maxgray</td>
<td>mingray</td>
<td>mingray</td>
</tr>
<tr>
<td>2</td>
<td>mingray</td>
<td>mingray</td>
<td>maxgray</td>
</tr>
<tr>
<td>3</td>
<td>mingray</td>
<td>maxgray</td>
<td>mingray</td>
</tr>
<tr>
<td>4</td>
<td>maxgray</td>
<td>maxgray</td>
<td>mingray</td>
</tr>
<tr>
<td>5</td>
<td>mingray</td>
<td>maxgray</td>
<td>maxgray</td>
</tr>
<tr>
<td>6</td>
<td>maxgray</td>
<td>mingray</td>
<td>maxgray</td>
</tr>
</tbody>
</table>

The global feature level method (Table 19.2) assumes that the initial positions of centroids are constructed of feature values (R, G, B) of a pixel with the darkest (min) and the brightest (max) within the region of interest.

The last variant, the local feature level (Table 19.3), is analogous in operation to the previous one, but the values are evaluated individually for each square region.

Table 19.2. Centroid initialization – global feature level.

<table>
<thead>
<tr>
<th>centroid</th>
<th>R</th>
<th>G</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>max</td>
<td>max</td>
<td>max</td>
</tr>
<tr>
<td>1</td>
<td>max</td>
<td>min</td>
<td>min</td>
</tr>
<tr>
<td>2</td>
<td>min</td>
<td>min</td>
<td>max</td>
</tr>
<tr>
<td>3</td>
<td>min</td>
<td>max</td>
<td>min</td>
</tr>
<tr>
<td>4</td>
<td>max</td>
<td>max</td>
<td>min</td>
</tr>
<tr>
<td>5</td>
<td>min</td>
<td>max</td>
<td>max</td>
</tr>
<tr>
<td>6</td>
<td>max</td>
<td>min</td>
<td>max</td>
</tr>
</tbody>
</table>

Table 19.3. Centroid initialization – local feature level.

<table>
<thead>
<tr>
<th>centroid</th>
<th>R</th>
<th>G</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>maxR</td>
<td>maxG</td>
<td>maxB</td>
</tr>
<tr>
<td>1</td>
<td>maxR</td>
<td>minG</td>
<td>minB</td>
</tr>
<tr>
<td>2</td>
<td>minR</td>
<td>minG</td>
<td>maxB</td>
</tr>
<tr>
<td>3</td>
<td>minR</td>
<td>maxG</td>
<td>minB</td>
</tr>
<tr>
<td>4</td>
<td>maxR</td>
<td>maxG</td>
<td>minB</td>
</tr>
<tr>
<td>5</td>
<td>minR</td>
<td>maxG</td>
<td>maxB</td>
</tr>
<tr>
<td>6</td>
<td>maxR</td>
<td>minG</td>
<td>maxB</td>
</tr>
</tbody>
</table>

The thresholds: homogeneity level and spot level are used as follows.

The homogeneity level (hl) is a factor used in the expression of homogeneity of the analyzed square:

$$(localMaxGray - localMinGray) > (Maxgray - Mingray) \cdot hl \cdot 0.01$$  \hspace{1cm} (19.26)
where Maxgray and Mingray are the levels defined in Table 19.1, while localMaxgray and localMinGray are the minimum and maximum pixel levels within an individual quare.

The spotlevel \((sl)\) is a factor in the decision rule if the pixel is assigned as a spot pixel. The rule is:

\[
\text{pixelGray} > \text{MinGray} + |\text{MaxGray} - \text{MinGray}| \cdot sl \cdot 0.01
\]  

(19.27)

As a result of the teaching phase, the matrix of pixel classes is filled. The algorithm is presented in Fig. 19.19.

1. Initialize the matrix of gravity centers (for each mask region)
2. If the region is homogenous (homogeneity level is not achieved) set all region pixel class as background and proceed to the next region. Otherwise, go to 3.
3. Get the next pixel in the region.
4. If the gray-level is less than the spot level, set the pixel class as background, otherwise assign the pixel class to the nearest class and go to 5.
5. If all region pixels are set, go to 6 else go to 3.
6. Update the matrix of gravity centers.
7. if the gravity center matrix has been changed go to step 3, else proceed to the next region.

Fig. 19.19 The teaching phase.

In the grouping phase the region of interest is split using a mask of a smaller diameter. This is the final step of pixel classification; all the pixels are assigned to some class. For each mask region:

1. Initialize the gravity center matrix based on the class matrix.
2. Evaluate the mean gray-level in the region.
3. Get the next pixel.
4. If the gray level of the pixel is less than the mean value, assign the pixel as a background and go to 6 else assign it to the nearest class and go to 5.
5. If all pixels are classified go to 6 else go to 3.
6. Update the gravity center matrix.
7. Once again assign all pixels to nearest centers.

Fig. 19.20 The grouping phase.
In the last – finalization – phase, the image of classes is generated. The part which is not in the Region of Interest is colored gray, the background pixels are white, and the other pixels have the color corresponding to the adequate gravity center value. If the median filter option is on, a filtering procedure to remove the image artifacts and to smooth the ragged object edges is carried out.

The algorithm was tested to prove the reliability and high performance of the segmentation. The following aspects were tested:

- the impact of the number of gravity centers upon the convergence of the algorithm (Tables 19.4–19.5),
- the impact of mask sizes upon the speed and quality of the segmentation (Table 19.6),
- the impact of the homogeneity and sensitivity thresholds upon the segmentation accuracy (Table 19.7).

Table 19.4 Convergency tests for various centroid initialization methods (part 1).

<table>
<thead>
<tr>
<th>Centroid initialization method</th>
<th>Iteration 0 Centroids coords.</th>
<th>Iteration 1 Centroids coords.</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>R</td>
<td>G</td>
<td>B</td>
</tr>
<tr>
<td>Global gray level</td>
<td>169</td>
<td>169</td>
<td>169</td>
</tr>
<tr>
<td></td>
<td>52</td>
<td>52</td>
<td>169</td>
</tr>
<tr>
<td></td>
<td>169</td>
<td>52</td>
<td>52</td>
</tr>
<tr>
<td></td>
<td>52</td>
<td>169</td>
<td>52</td>
</tr>
<tr>
<td></td>
<td>169</td>
<td>169</td>
<td>52</td>
</tr>
<tr>
<td></td>
<td>52</td>
<td>169</td>
<td>169</td>
</tr>
<tr>
<td></td>
<td>169</td>
<td>52</td>
<td>169</td>
</tr>
<tr>
<td>Global feature level</td>
<td>172</td>
<td>172</td>
<td>172</td>
</tr>
<tr>
<td></td>
<td>35</td>
<td>35</td>
<td>172</td>
</tr>
<tr>
<td></td>
<td>172</td>
<td>35</td>
<td>35</td>
</tr>
<tr>
<td></td>
<td>35</td>
<td>172</td>
<td>35</td>
</tr>
<tr>
<td></td>
<td>172</td>
<td>172</td>
<td>35</td>
</tr>
<tr>
<td></td>
<td>35</td>
<td>172</td>
<td>172</td>
</tr>
<tr>
<td></td>
<td>172</td>
<td>35</td>
<td>172</td>
</tr>
<tr>
<td>Local feature level</td>
<td>180</td>
<td>177</td>
<td>174</td>
</tr>
<tr>
<td></td>
<td>35</td>
<td>54</td>
<td>174</td>
</tr>
<tr>
<td></td>
<td>180</td>
<td>54</td>
<td>55</td>
</tr>
<tr>
<td></td>
<td>35</td>
<td>177</td>
<td>55</td>
</tr>
<tr>
<td></td>
<td>180</td>
<td>177</td>
<td>55</td>
</tr>
<tr>
<td></td>
<td>35</td>
<td>177</td>
<td>174</td>
</tr>
<tr>
<td></td>
<td>180</td>
<td>54</td>
<td>174</td>
</tr>
</tbody>
</table>
The results shown in Tables 19.4–19.5 indicate that the global gray level approach is the most convergent and the initial positions of centroids correspond to the colors: red, green, blue, cyan, magenta, yellow and white. The other methods require more iterations to achieve the final positions but the initial position is set better, which may be an advantage when the maximum number of iterations is limited. All three methods are convergent to the same centroid positions.

<table>
<thead>
<tr>
<th>Centroid initialization method</th>
<th>Iteration 2</th>
<th>Last iteration</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Centroid coords.</td>
<td>Centroid coords.</td>
</tr>
<tr>
<td></td>
<td>R    G    B</td>
<td>R    G    B</td>
</tr>
<tr>
<td>Global gray level</td>
<td>162 155 146 4</td>
<td>163 157 149</td>
</tr>
<tr>
<td></td>
<td>65 81 107 7</td>
<td>70 85 110</td>
</tr>
<tr>
<td></td>
<td>133 91 83 18</td>
<td>140 104 94</td>
</tr>
<tr>
<td></td>
<td>0 0 0 0 14</td>
<td>0 0 0</td>
</tr>
<tr>
<td></td>
<td>152 127 115 20</td>
<td>157 141 129</td>
</tr>
<tr>
<td></td>
<td>118 127 136 6</td>
<td>123 130 138</td>
</tr>
<tr>
<td></td>
<td>0 0 0 0</td>
<td>0 0 0</td>
</tr>
<tr>
<td>Global feature level</td>
<td>161 154 145 5</td>
<td>163 157 149</td>
</tr>
<tr>
<td></td>
<td>62 78 105 12</td>
<td>70 85 110</td>
</tr>
<tr>
<td></td>
<td>130 87 80 24</td>
<td>140 104 94</td>
</tr>
<tr>
<td></td>
<td>0 0 0 0 15</td>
<td>0 0 0</td>
</tr>
<tr>
<td></td>
<td>150 123 111 26</td>
<td>157 141 129</td>
</tr>
<tr>
<td></td>
<td>115 124 135 10</td>
<td>123 130 138</td>
</tr>
<tr>
<td></td>
<td>0 0 0 0</td>
<td>0 0 0</td>
</tr>
<tr>
<td>Local feature level</td>
<td>162 155 146 4</td>
<td>163 157 149</td>
</tr>
<tr>
<td></td>
<td>66 81 108 1</td>
<td>67 82 108</td>
</tr>
<tr>
<td></td>
<td>135 96 87 13</td>
<td>141 105 95</td>
</tr>
<tr>
<td></td>
<td>0 0 0 0 17</td>
<td>0 0 0</td>
</tr>
<tr>
<td></td>
<td>154 130 118 17</td>
<td>157 142 130</td>
</tr>
<tr>
<td></td>
<td>119 127 137 3</td>
<td>121 129 137</td>
</tr>
<tr>
<td></td>
<td>0 0 0 0</td>
<td>0 0 0</td>
</tr>
</tbody>
</table>

Mask size adjustments affect the algorithm speed and the segmentation quality. The experiments carried out on the images showed that the mask in the teaching phase should be smaller than in the grouping with the ratio about 1:3. The suggested mask sizes for the processed resolution are 20 for teaching and 60 – 10 for grouping. Too small masks defect the weak fuzzy spot detection and too large masks dramatically increase the computation time.
Table 19.6 Segmentation quality for various segmentation mask sizes.

<table>
<thead>
<tr>
<th>Centroid initialization method</th>
<th>Mask size</th>
<th>Teaching phase</th>
<th>Grouping phase</th>
<th>Time [ms]</th>
<th>Fract. of properly detected spots [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Global gray level</td>
<td>5</td>
<td>5</td>
<td>1210</td>
<td>65</td>
<td></td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>1122</td>
<td>68</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>20</td>
<td>1109</td>
<td>80</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>1192</td>
<td>78</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>200</td>
<td>1333</td>
<td>81</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>1241</td>
<td>92</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>1728</td>
<td>43</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>200</td>
<td>1699</td>
<td>72</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>400</td>
<td>1953</td>
<td>62</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>800</td>
<td>2346</td>
<td>66</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Global feature level</td>
<td>5</td>
<td>1217</td>
<td>35</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>1240</td>
<td>38</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>20</td>
<td>1225</td>
<td>53</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>1364</td>
<td>37</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>200</td>
<td>1402</td>
<td>56</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>1204</td>
<td>66</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>1675</td>
<td>43</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>200</td>
<td>1807</td>
<td>64</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>400</td>
<td>1808</td>
<td>67</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>800</td>
<td>2246</td>
<td>64</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Local feature level</td>
<td>5</td>
<td>1240</td>
<td>67</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>1107</td>
<td>68</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>20</td>
<td>1273</td>
<td>80</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>1258</td>
<td>84</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>200</td>
<td>1214</td>
<td>96</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>1339</td>
<td>88</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>1653</td>
<td>43</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>200</td>
<td>1670</td>
<td>68</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>400</td>
<td>1778</td>
<td>64</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>800</td>
<td>2274</td>
<td>65</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The homogeneity and spot levels have a very high impact on the quality of segmentation. The values must be adjusted experimentally and may differ when the camera is replaced or the reagents are changed.

The experiments showed that the proper adjustment of the segmentation parameters enables segmentation of all the tested ELISPOT images. The method with local feature levels offers the highest segmentation quality. Despite its worst convergence, it is also the fastest because a segmentation mask may be relatively small. Its drawback is high sensitivity to threshold factor adjustment, which is a serious weakness for unattended processing.
Table 19.7 Segmentation quality as a function of thresholding factors.

<table>
<thead>
<tr>
<th>Centroid initialization method</th>
<th>hl [%]</th>
<th>sl [%]</th>
<th>Time [ms]</th>
<th>Spot count</th>
<th>False spot count</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Global gray level</strong></td>
<td>1</td>
<td>1302</td>
<td>91</td>
<td>*</td>
<td></td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>1331</td>
<td>86</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td></td>
<td>75</td>
<td>1578</td>
<td>92</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td></td>
<td>25</td>
<td>1175</td>
<td>93</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td></td>
<td>75</td>
<td>971</td>
<td>22</td>
<td>78</td>
<td></td>
</tr>
<tr>
<td><strong>Global feature level</strong></td>
<td>1</td>
<td>1640</td>
<td>95</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>1706</td>
<td>81</td>
<td>19</td>
<td></td>
</tr>
<tr>
<td></td>
<td>75</td>
<td>1763</td>
<td>83</td>
<td>17</td>
<td></td>
</tr>
<tr>
<td></td>
<td>25</td>
<td>1545</td>
<td>83</td>
<td>17</td>
<td></td>
</tr>
<tr>
<td></td>
<td>75</td>
<td>1235</td>
<td>23</td>
<td>77</td>
<td></td>
</tr>
<tr>
<td><strong>Local feature level</strong></td>
<td>1</td>
<td>1806</td>
<td>*</td>
<td>*</td>
<td></td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>1807</td>
<td>*</td>
<td>*</td>
<td></td>
</tr>
<tr>
<td></td>
<td>75</td>
<td>2719</td>
<td>*</td>
<td>*</td>
<td></td>
</tr>
<tr>
<td></td>
<td>25</td>
<td>2759</td>
<td>*</td>
<td>*</td>
<td></td>
</tr>
<tr>
<td></td>
<td>75</td>
<td>1663</td>
<td>96</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td></td>
<td>75</td>
<td>1220</td>
<td>24</td>
<td>76</td>
<td></td>
</tr>
</tbody>
</table>

* means very poor segmentation

The global feature level method gives high-quality segmentation without the user having to take excessive care of the parameters, but it requires large segmentation masks, which increases the computation cost.

The method of global gray level gives stable results without modifying the clustering parameters, but during the segmentation the weakest and smallest spots are often omitted.

19.7 Monochrome image segmentation using Hit-Miss transform

The hit-miss transform is a basic binary morphological operation. It is most frequently used to search for specific structures that form the object and the background pixels. Its idea was thoroughly described in (Fisher et al., 2004). In practice, with this transformation, it is possible to search for objects of various shapes, or of the same shape, but changing orientation. In this chapter, HMT has been adapted for greyscale and color images. Since the objects to be found are circular spots of certain colors and size ranges, it is possible to design suitable hit-and-miss conditions.
To detect spots of different diameters, a multi-pass algorithm is used. In each cycle, the image is scanned with a decreasing size of the mask. After the setup phase of the algorithm, the mask array is filled as shown in Fig. 19.21.

![Flowchart of the main function of the algorithm.](image)

**Fig. 19.22** The flowchart of the main function of the algorithm.
The array consists of the region representing the object (‘ones’) and the background (‘zeros’). After filling the mask, the main loop of the algorithm is run (Fig. 19.22)

In each pass of the function, a new size of the mask is set and the mask is filled. Then, scanning for each pixel of the image starts. The implementation uses a pointer to optimize the algorithm performance. Initially, for each inspected pixel it is checked whether there is a spot at these coordinates (possibly found earlier). If so, the point is omitted. If no, the neighborhood of the pixel determined by the mask is examined. Pixels which belong to the pink region (see Fig. 19.21) and marked “1” are supposed to belong to the object, while those which belong to the cyan region are supposed to belong to the background.

It was experimentally determined that the color of the background pixels differs significantly from the object pixels in each color component. In Table 19.8 there are model values of colors for spots and the background presented.

Table 19.8 Model colors for background and spot pixel.

<table>
<thead>
<tr>
<th>Color component</th>
<th>Background color</th>
<th>Spot color</th>
</tr>
</thead>
<tbody>
<tr>
<td>Red</td>
<td>249</td>
<td>167</td>
</tr>
<tr>
<td>Green</td>
<td>204</td>
<td>105</td>
</tr>
<tr>
<td>Blue</td>
<td>113</td>
<td>84</td>
</tr>
</tbody>
</table>

For each position of the mask we calculate:

- an average value of R, G, B for pixels marked pink;
- an average value of R, G, B for pixels marked cyan;

These two values are compared with threshold values to classify the examined point. The threshold values are calculated locally for each position of the mask using the calculated averages of R, G, and B. This mitigates the influence of unequal exposure of the image. In Table 19.9 threshold values are presented.

Table 19.9 Threshold values for the spot and the background.

<table>
<thead>
<tr>
<th>Value of S=avg(R+G+B)</th>
<th>Threshold value</th>
</tr>
</thead>
<tbody>
<tr>
<td>from to</td>
<td>Object</td>
</tr>
<tr>
<td>0 to 180</td>
<td>0</td>
</tr>
<tr>
<td>180 to 250</td>
<td>S - 12</td>
</tr>
<tr>
<td>250 to 300</td>
<td>S - 15</td>
</tr>
<tr>
<td>300 to 350</td>
<td>S - 17</td>
</tr>
<tr>
<td>350 to 400</td>
<td>S - 18</td>
</tr>
<tr>
<td>400 to 460</td>
<td>S - 20</td>
</tr>
<tr>
<td>460 to 510</td>
<td>S - 24</td>
</tr>
<tr>
<td>510 to 570</td>
<td>S - 22</td>
</tr>
<tr>
<td>570 to 630</td>
<td>S - 22</td>
</tr>
<tr>
<td>630</td>
<td>S - 24</td>
</tr>
</tbody>
</table>
Apart from comparing the sums of colors with the thresholds, we use specific control points. In Fig. 19.21 they are “1” marked cyan. The necessary condition of the positive detection of a spot is that the values of the pixels must be less than the threshold value increased by 6% (the margin for the control points).

If the detection is positive, a spot is registered, otherwise we move to the next pixel. The algorithm terminates when all the pixels have been examined.

**Algorithm benchmarks**

The algorithm was tested to evaluate its efficiency. Figures 19.23 and 19.24 present the fragments of segmented images from two different cameras and exposure conditions. An uneven exposure is, to a certain extent, not an obstacle in the operation of the algorithm.

![Fig. 19.23 Exemplary image and found spots (even exposure).](image)

The results from these images were obtained using low thresholds for the colors of the objects and the background and a rather high margin of the control points.

![Fig. 19.24 Exemplary image and found spots (uneven exposure).](image)
Although the sensitivity of the algorithm is sufficiently good to find all the actual spots it has a little drawback. The radii of the recognized spots are larger than in reality. (Fig. 19.25)

In Fig. 19.26 the results of segmentation with tuned parameters of the algorithm are presented. The original image (a) was initially segmented using the default values of the local thresholds and control points (b). The number of spots found is 85. In (c) the local thresholds were decreased and the control margin was decreased to 104%. The number of spots found is 94.

In (d) the result of segmentation with default thresholds and the margin of control points increased to 108% is shown. The number of spots found is 93 and the effect of too large a size of spots found is minimal.

The experiments show that decreasing the thresholds and increasing the margins enables catching spots of low contrast, but then the “strong” objects have a false size. Increasing thresholds but decreasing the control margin gives the result of losing small objects. The best results were obtained for the thresholds given in Table 19.9 and the control point margin 108% of the thresholds.
Simple image preprocessing (noise reduction and contrast enhancement) may significantly increase the efficiency of the method (Fig. 19.27).
The number of detected spots increased by 10% compared to the original image.

The range of variation of the scanning mask size has also a significant impact on the quality of segmentation. Figs. 19.28 and 19.29 show two different images processed: with large and small spots. For each image the range of the mask was set to 6–24 pixels and then to 1–10 pixels. In both cases the number of iterations was 10.

The range 6–24 appeared to be good only for the first image, with large spots. When the image consisted of many small spots, the algorithm misinterpreted clusters of small spots as larger spots.

For the mask range 1–10 pixels the results are much better for both images. It should also be noted that the computational complexity of the algorithm depends on the size of the mask. In the case of the exemplary images the algorithm execution time was three times faster for smaller masks.
Various Approaches to Processing and Analysis of Images

Fig. 19.29  Image consisting small spots segmented with different mask ranges.

19.8  Monochrome image segmentation using edge detection
and geometrical fitting of the shape

The routines examined in the previous chapters allow successful detection of rough location and size of the spots as well as recognition of their color.

But the main problem was how to precisely measure the amount of the staining, which translates to the area of individual spots. The reason why it is difficult to determine is blurry edges of the spots. All the approaches listed above use thresholding at least in their initial stage and thus are very sensitive to the selection of the threshold; in other words, changing the threshold slightly may severely change the measured area of a given object. In the current chapter we present an algorithm based on edge detection which appears to yield more stable results and the returned spot contours quite closely correspond to real ones.

Tracing edges on an image has a potential of being more accurate in returning local object boundaries, but there are two problems concerning it: an easy one is to fill the interior of the contour of the spot, and a harder one is to obtain a closed contour instead of several non-connected arcs. We follow this approach in the current description.

In general, the algorithm performs the operations outlined in Figure 19.30.
1. resolution normalization;
2. noise reduction using low-pass filtering;
3. boundary emphasize using sharpening filtering;
4. Canny curve detection;
5. circle-fitting procedure applied to arcs;
6. reconstruction of circular spot boundaries.

Fig. 19.30  The general outline of the algorithm.

We claim that the algorithm is less vulnerable to variable image properties: image resolution, out-of-focus zones and improper exposition (non-uniform lighting). Even if the detected spot borders are severely fragmented and incomplete, the fitting procedure can evaluate the spot circles.

**Image preprocessing**

In this phase we attempt to prepare the image for the Canny procedure, which performs best if the border to be traced is approximately 3–5 pixels wide. In the examined images obtained using a Nikon Camera (1182 x 1280) the ROI has approximately 750 pixels of diameter, and the diameter of the spots varies from 10 to 50 pixels. To obtain the best performance, the dimensions of the picture must be doubled (using bilinear interpolation).

On the resized (resampled) image two procedures are imposed:
- the Gaussian smoothing filter with a mask diameter of 5 pixels,
- the Laplacian as a sharpening filter with a mask diameter of 3 pixels,
- the linear adjustment of the image intensity histogram (for contrast equalization).

Fig. 19.31  Gaussian smoothing and Laplace sharpening masks.

**Boundary detection**

There are many approaches to edge and boundary detection. For this purpose the Canny procedure (Canny, 1986; Grigorescu, 2004) was chosen because of its benefits listed below:
- after the proper preprocessing it generates stable results,
- it is invulnerable to image rotation,
- thanks to the “threshold hysteresis”, it can detect and trace weak or fading lines.

The algorithm is carried out in three phases:
1. The image is filtered with a vertical and horizontal Sobel mask in order to estimate the gradient magnitude and angle.

\[
\begin{array}{ccc}
1 & 2 & 1 \\
0 & 0 & 0 \\
-1 & -2 & -1 \\
\end{array}
\quad
\begin{array}{ccc}
-1 & 0 & 1 \\
-2 & 0 & 2 \\
-1 & 0 & 1 \\
\end{array}
\]

Fig. 19.32 Sobel gradient masks.

2. If the intensity gradient magnitude is greater than the threshold value \( t_1 \), the procedure of line tracing is started, and continued until the line direction changes or its strength falls below the \( t_2 \) threshold (\( t_2 < t_1 \)).
3. The algorithm usually finds line zones (several perpendicular neighboring lines). The proper lines are extracted with the aid of the non-maximum suppression procedure.

The Canny procedure generates a binary image, which is subsequently segmented into connected components (individual lines), which are then regarded as candidates for arcs of the reconstructed circles.

Fig. 19.33 The input image and after Canny operator.
Arc approximation

In this stage each connected component is regarded as a tabulated function and its analytical representation has to be approximated. The circle-finding procedure introduced in (Gander, 1994) runs in two steps:

1. the fitting procedure using the linear least squares method (the implementation from (Press et al., 2002));
2. the iterative fine-fitting using geometric distance (the implementation from (Press et al., 2002));

Consider $x^T$ and $y^T$ as $m$-length vectors of coordinates of the component points. Let us denote:

$$
\begin{bmatrix}
  x_1^2 + y_1^2 & x_1 & y_1 & 1 \\
  x_2^2 + y_2^2 & x_2 & y_2 & 1 \\
  \vdots & \vdots & \vdots & \vdots \\
  x_m^2 + y_m^2 & x_m & y_m & 1 \\
\end{bmatrix}
$$

(19.28)

Assuming that $m$, the number of points in each connected component, is more than 4, the matrix $[A]$ is rectangular and singular. To deal with $[A] \cdot x = b$ linear equation in this case, the singular value decomposition method must be applied.

The idea of decomposition of the matrix $[A]$ (MxN) is the creation of a column orthogonal matrix $[U]$ (MxN), a diagonal matrix $[W]$ (NxN) with zero or non-zero elements and square, and an orthogonal matrix $[V]$ (NxN).

$$
[A] = [U] \cdot \begin{bmatrix}
  w_{11} & 0 & \cdots & 0 \\
  \vdots & \ddots & \ddots & \vdots \\
  0 & \cdots & w_{mm} \\
\end{bmatrix} \cdot [V]^T \quad \text{where} \quad [U]^T \cdot [U] = [V]^T \cdot [V] = [I] \quad (19.29)
$$

In the further processing of the fitting algorithm, the square matrix $[V]$ (4x4) is used and the circle parameters are calculated as follows:

$$
x_0 = \frac{-v_{24}}{2 \cdot v_{14}} ; \quad y_0 = \frac{-v_{34}}{2 \cdot v_{14}} ; \quad r_0 = \sqrt{\frac{v_{24}^2 + v_{34}^2}{4 \cdot v_{14}^2} - \frac{v_{44}}{v_{14}}} \quad (19.30)
$$

When initial $(x_0, y_0, r_0)$ values are calculated, the iterative procedure is launched.

The aim of the procedure is to minimize the square error

$$
\varepsilon = \sum \left( \| (x, y) - (x_0, y_0) \| - r \right)^2 \quad (19.31)
$$
Let us denote:

\[
\begin{bmatrix}
\sqrt{(x_1 - x_0)^2 + (y_1 - y_0)^2} - r_0 \\
\sqrt{(x_2 - x_0)^2 + (y_2 - y_0)^2} - r_0 \\
\vdots \\
\sqrt{(x_m - x_0)^2 + (y_m - y_0)^2} - r_0 
\end{bmatrix}
\]

(19.32)

and

\[
[J] = \begin{bmatrix}
\frac{x_1 - x_0}{\sqrt{(x_1 - x_0)^2 + (y_1 - y_0)^2}} & \frac{y_1 - y_0}{\sqrt{(x_1 - x_0)^2 + (y_1 - y_0)^2}} & 1 \\
\frac{x_2 - x_0}{\sqrt{(x_2 - x_0)^2 + (y_2 - y_0)^2}} & \frac{y_2 - y_0}{\sqrt{(x_2 - x_0)^2 + (y_2 - y_0)^2}} & 1 \\
\vdots & \vdots & \vdots \\
\frac{x_m - x_0}{\sqrt{(x_m - x_0)^2 + (y_m - y_0)^2}} & \frac{y_m - y_0}{\sqrt{(x_m - x_0)^2 + (y_m - y_0)^2}} & 1 
\end{bmatrix}
\]

(19.33)

The algorithm solves the equation:

\[
[J] \cdot h = f
\]

(19.34)

with the help of the QR decomposition method.

\[
[J] = [Q] \cdot [R] \Rightarrow [R] \cdot h = [Q]^T \cdot f
\]

(19.35)

where \([Q]\) is orthogonal (\([Q]^T \cdot [Q] = [I]\)) and \([R]\) is triangular (upper). In each iteration, the circle is updated

\[
(x_0, y_0, r_0) = (x_0, y_0, r_0) + h
\]

(19.36)

and the error

\[
\delta = \frac{||h||_{\infty}}{||(x_0, y_0, r_0)||_{\infty}}
\]

(19.37)

is estimated. The iteration repeats until \(\delta < \delta_0 = 10^{-5}\) or the number of iterations exceeds 100 (then, the algorithm is not convergent – the line is not an arc).
Spot border reconstruction

The Canny algorithm finds the following lines:

- fragments of the spot borders (sometimes more than 1 for each spot);
- fragments of the spot interiors (caused by non homogenous spot staining);
- the well boundaries;

Initially, we leave only the circles for which $r \in (0.05 \cdot D; 0.1 \cdot D)$ where $D$ is the diameter of ROI. Then, we delete the circles that cross the ROI circle.

It is very likely that the remaining connected components are parts of the spot contours.

It sometimes happens that the spot contour is fragmented. In this case, the same number of circles is produced for each spot. The circles which overlap, or in other words, for which the following condition is satisfied:

$$\rho((x_1, y_1), (x_2, y_2)) \leq \min(r_1, r_2),$$  \hspace{1cm}  (19.38)

represent the same contour and the respective arcs should be merged.

The merging algorithm works in three phases.

Phase 1: For each arc, find the neighboring arcs based on (38). Let us look at Fig. 19.35 (the artificial spots).
Fig. 19.35 The exemplary image: enumeration of found arcs.

For this example, the following neighbors are found (Table 19.10).

Table 19.10 Object parameters for the exemplary image.

<table>
<thead>
<tr>
<th>Object number</th>
<th>Circle ((x, y, r))</th>
<th>Found neighbors</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>((360.1, 256.1, 30.9))</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>((367.2, 254.2, 43.0))</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>((224.8, 260.7, 22.7))</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>((228.3, 259.8, 21.2))</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>((235.2, 406.5, 85.6))</td>
<td>6; 9; 11; 12</td>
</tr>
<tr>
<td>6</td>
<td>((248.5, 397.8, 71.1))</td>
<td>5; 9; 11; 12</td>
</tr>
<tr>
<td>7</td>
<td>((435.9, 391.3, 52.3))</td>
<td>8; 10</td>
</tr>
<tr>
<td>8</td>
<td>((444.0, 389.1, 47.1))</td>
<td>7; 10</td>
</tr>
<tr>
<td>9</td>
<td>((224.1, 398.4, 71.9))</td>
<td>5; 6; 11; 12</td>
</tr>
<tr>
<td>10</td>
<td>((438.8, 390.3, 55.8))</td>
<td>7; 8</td>
</tr>
<tr>
<td>11</td>
<td>((238.1, 390.4, 87.8))</td>
<td>5; 6; 9; 12</td>
</tr>
<tr>
<td>12</td>
<td>((236.4, 389.1, 87.7))</td>
<td>5; 6; 9; 11</td>
</tr>
</tbody>
</table>

Phase 2: After the process of traversing the adjacency graph, we build a list of clusters (Table 19.11).

Table 19.11 Clusters for the exemplary image.

<table>
<thead>
<tr>
<th>Cluster number</th>
<th>Arcs numbers</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1; 2</td>
</tr>
<tr>
<td>2</td>
<td>3; 4</td>
</tr>
<tr>
<td>3</td>
<td>5; 12; 11; 9; 6</td>
</tr>
<tr>
<td>4</td>
<td>7; 10; 8</td>
</tr>
</tbody>
</table>
In the last phase, the points of all the arcs for each cluster are submitted to the Find-Circle procedure to obtain fine-tuned circles.

**Results**

The experiment involves the comparison of three segmentation methods: Bernsen thresholding, Background equalization + adaptive thresholding, and Canny edge detector + arc merging. In Fig. 36 we can observe the borders of segmented objects with use of the three methods.

![Fig. 19.36 The results of image segmentation.](image)

For this image the basic geometry properties: area, perimeter, Aspect Ratio and Gamma coefficient were calculated. The results for individual spots are shown in Table 19.12. From Fig. 19.36 and Table 19.12 we can see that the adaptive thresholding missed some small spots. This happens because small spots have weak intensity. Both Bernsen’s algorithm and Canny’s operator were able to detect the spots. Comparing the area value obtained for the spots for different methods, we easily notice that edge detecting gives larger spots than thresholding (in this case). Using another thresholding algorithm or changing its parameters will yield give different areas. Changing the thresholds in Canny’s method may introduce some irrelevant lines (edges) or suppress some important fragments of the contours, but the contours will always appear in the same place. This guarantees that the sizes of the objects are more stable. Proper contour detection
enables accurate measurement of the object shape descriptors, which is also visible in Table 19.12.

Table 19.12  Object parameters for different segmentation types: A – adaptive, B – Bernsen, C – Canny.

<table>
<thead>
<tr>
<th>Obj. Area</th>
<th>Perimeter</th>
<th>AR</th>
<th>Gamma</th>
</tr>
</thead>
<tbody>
<tr>
<td>1290</td>
<td>117</td>
<td>123</td>
<td>132</td>
</tr>
<tr>
<td>1127</td>
<td>1283</td>
<td>62</td>
<td>61</td>
</tr>
<tr>
<td>1085</td>
<td>874</td>
<td>1021</td>
<td>61</td>
</tr>
<tr>
<td>2684</td>
<td>222</td>
<td>303</td>
<td>60</td>
</tr>
<tr>
<td>300</td>
<td>205</td>
<td>54</td>
<td>61</td>
</tr>
<tr>
<td>214</td>
<td>292</td>
<td>64</td>
<td>60</td>
</tr>
<tr>
<td>2503</td>
<td>3000</td>
<td>62</td>
<td>189</td>
</tr>
<tr>
<td>230</td>
<td>257</td>
<td>323</td>
<td>254</td>
</tr>
<tr>
<td>196</td>
<td>295</td>
<td>61</td>
<td>49</td>
</tr>
<tr>
<td>3892</td>
<td>4572</td>
<td>99</td>
<td>237</td>
</tr>
<tr>
<td>3862</td>
<td>472</td>
<td>99</td>
<td>254</td>
</tr>
<tr>
<td>733</td>
<td>720</td>
<td>147</td>
<td>93</td>
</tr>
<tr>
<td>216</td>
<td>294</td>
<td>203</td>
<td>52</td>
</tr>
<tr>
<td>1122</td>
<td>1313</td>
<td>1593</td>
<td>152</td>
</tr>
<tr>
<td>1373</td>
<td>2969</td>
<td>209</td>
<td>139</td>
</tr>
<tr>
<td>1926</td>
<td>2486</td>
<td>1642</td>
<td>172</td>
</tr>
<tr>
<td>2027</td>
<td>2655</td>
<td>3101</td>
<td>173</td>
</tr>
</tbody>
</table>

19.9 Conclusions

In this chapter we have shown that segmentation of ELISPOT images is not a trivial task and requires combination of standard and specially prepared image processing algorithms. A very important phase in the process of segmentation is proper image preprocessing, which allows the main algorithm to work smoothly.

Despite this, it is possible to use at least two main approaches for the segmentation process:

- pixel classification: the thresholding of the grey level or color, the clustering of the color space
- shape analysis: matching of the shapes and sizes of the objects or finding their contours.

All the described algorithms yielded accurate results on condition that their parameters were tuned. It is unlikely that the design of an algorithm running for each image completely without user supervision is possible.
References


