Pattern recognition for high throughput zebrafish imaging using genetic algorithm optimization

Nezhinsky, A.E.; Verbeek, F.J.; Dijkstra, T.M.H.; Tsivtsivadze, E.; Marchiori, E.; Heskes, T.

Pattern Recognition for High Throughput Zebrafish Imaging Using Genetic Algorithm Optimization

Alexander E. Nezhinsky and Fons J. Verbeek

Imaging and Bioinformatics, Leiden University, Leiden Institute of Advanced Computer Science, Niels Bohrweg 1, 2333CA, Leiden, The Netherlands

Abstract. In this paper we present a novel approach for image based high–throughput analysis of zebrafish embryos. Zebrafish embryos can be made available in high numbers; specifically in groups that have been exposed to different treatments. Preferably, the embryos are processed in batches. However, this complicates an automated processing as individual embryos need to be recognized. We present an approach in which the individual embryos are recognized and counted in an image with multiple instances and in multiple orientations. The recognition results in a mask that is used in the analysis of the images; multichannel images with bright–field and fluorescence are used.

The pattern recognition is based on a genetic algorithm which is the base of an optimization procedure through which the pattern is found. The optimization is accomplished by a deformable template that is incorporated in the genetic algorithm. We show that this approach is very robust and produces result fast so that it becomes very useful in a high–throughput environment. The method is fully automated and does not require any human intervention. We have tested our approach on both synthetic and real life images (zebrafish embryos). The results indicate that the method can be applied to a broad range of pattern recognition problems that require a high–throughput approach.

1 Introduction

Retrieving location and contour of a shape is crucial to the analysis of large image datasets in many fields, such as optical character recognition (OCR) and bio–imaging. If the number of occurrences of an object under study is not known beforehand, or the object we are trying to locate is subject to slight deformations extra complications arise; this is typical for life–sciences. In addition we sometimes need to take into account partial occlusion and noise.

In the current practices segmentation is a first step to separate objects from the background and a variety of segmentation techniques is available [8], [15]. In this paper we want to address the problem of recognition of object localization under different conditions of strain stress. Our specific interest is in shapes of which a prior shape information is available. To that end we need to use an approach that depends on the inexact predefined shape and can be subject to deformations in the input image. A deformable template [1] approach gives us the possibility to represent an object that can be subject to certain deformations in a compact manner. In that case object localization should be performed by a process of matching the deformable template to the object shape in the input image.

One approach is the *free–form* class [1] with the Active Contour (a.k.a. active snake) model [9]. This method has no global template structure and needs a starting location in the image to evolve from. Shape edges need to be connected together for an active snake to perform correctly. If the starting location is not known or multiple objects are present in the image this algorithm might be hampered.

We consider the type of deformable template models of the *parametric* class ([1]); i.e. the template shape is predefined as a set of parameters. In most approaches [15], [3], [11] deformable template representation of a shape is a set of points approximating the outline as obtained from a priori knowledge. In this manner a user defined input template is represented and we adopted this representation for our approach.

Matching a deformable template to an image can be seen as optimization problem with some possible global maxima – in our case best solution, being the best shape. This process is computationally expensive. A possible solution is therefore to reduce the search space [16] by focusing only on areas containing certain intensity (color). However, we want to base our algorithm on shape characteristics only, since color information might not always be available or subject to large variation. Genetic Algorithms (GA's) [4], [14] are typically suitable for solving global optimization problems, in particular if the solution space is very large. Therefore, we consider a Genetic Algorithm for optimization.

In this paper we introduce a slice representation model (SR) of a deformable template. Instead of considering the outline of a shape, we simplify the shape representation by considering only certain characteristic horizontal slices and use these for template matching. The proposed SR model is made advantageous for efficient optimization with a Genetic Algorithm. [6], [12] also used a GA for low parameter shape templates (circle and ellipse detection).

In the approach discussed in this paper we will use binary images as input. Binary images are successfully used for template and polygon matching in [10] and [7]). Whenever the images are presented as RGB or gray scale, they are thresholded to binary, cf [13].

In addition we want to address an important feature of automatic retrieval of multiple deformed shapes from a single image and counting the amount of shapes.

The paper is organized as follows. In section 2 a more detailed overview of our deformable template approach is given. Section 3 addresses the proposed GA used for shape recognition. In Section 4 we propose the application of the algorithm to retrieve multiple shapes in a single image. In section 5 experimental results for an application to a real life problem are shown and they are discussed in section 6.

2 Deformable Template

We propose a template representation which captures both boundary and interior of the object and thereby describes the average structure of a predefined shape. The grid of the search space is determined by the discrete pixels in a $K \times L$ image matrix M. We take a binary image as starting point. The binary image is obtained by thresholding. Thereby we assume a background pixel has the value 0 and a foreground pixel value 1. In the following subsections we will describe the template representation in more detail as well as the deformations that a template can undergo.

2.1 Slice Representation Model

The prototype template T_0 we propose is represented by the following vector:

$$T_0 = (s_0, s_1, .., s_n, d), \tag{1}$$

where n is the number of slices; d is the distance between two slices in the horizontal direction. Initially T_0 is represented in a X(horizontal)–Y(vertical) space in the horizontal direction, with slices being parallel to the Y–axis. This is done for ease in the representation. A template slice itself is then represented by the vector:

$$s_i = (w_i, b_i, \varsigma_i), \tag{2}$$

where w_i is the fixed width in pixels in the vertical direction, preferably $w_i = 1$; where b_i is the fixed width in pixels in the vertical direction of the image, preferably $b_i = 0$ and are located below and above the area of w_i. The b_i are required to define that the area surrounding the shape is preferably empty. ς_i is the seed point of a slice; ς is needed to be able to define non symmetrical shapes as input. This point will be used as a reference for allowed slice shifts. In Fig. 1(a) an example slice is shown; a slice is a sample of the template always perpendicular to the length axis.

In Fig. 1(b), 1(c), 1(d) two examples of templates are shown. The template length is fixed according to $\ell(v) = n * d$. For templates with horizontal symmetry axis, ς is simply chosen as the center of each slice (cf Fig. 1(c)). For templates having no horizontal symmetry axis, ς is chosen in such a way that all ς_i are located on the same horizontal line (cf Fig. 1(e)).

2.2 Deformations

This representation was chosen as we assumed the slices will be able to move vertically along the template to match a deformed (slightly shifted) shape. At template shift or slight rotation the global shape is still captured within the template. The total length of the template is fixed in the proposed representation.

A deformed template T is derived from the prototype template T_0 and is represented as $T(T_0, I)$. A deformation I will then be encoded by a state-sequence as follows:

$$I = (x, y + \delta_0, y + \delta_1, .., y + \delta_n), \tag{3}$$

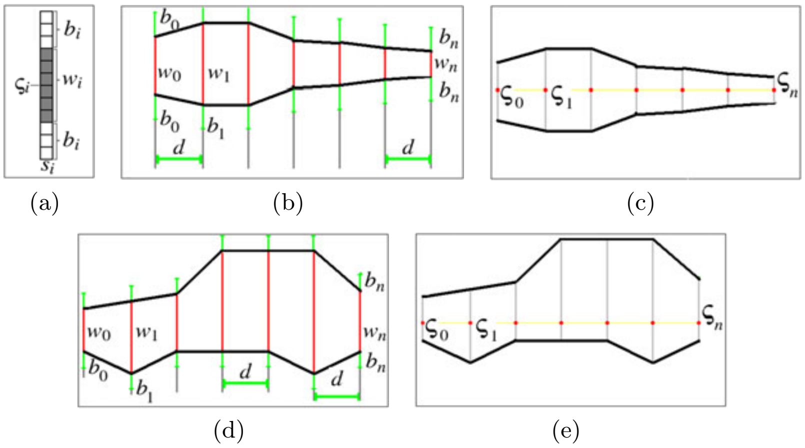

Fig. 1. a) A single slice. The slice is always parallel to the Y–axis b) Template for a fish shape c) Location of ς in a fish shape template d) Template for a car shape e) Location of ς in a car shape template.

where x is the shift in the X-axis direction of ς_0; $y+\delta_i$ is the translation measured in the Y–axis direction of the ς_i of slice i.

We assume the maximal vertical deformation between two slices is 45 degrees (cf Fig. 2(a). As a result of this constraint $|\delta_{i-1} - d| < |\delta_i| < |\delta_{i-1} + d|$ applies for two starting points of consecutive slices i and $i + 1$. Then, each ς_i can then be shifted vertically within the following boundaries:

$$\max\{(\varsigma_{i-1} - d, \varsigma_{i+1} - d)\} < \varsigma_i < \min\{(\varsigma_{i-1} + d, \varsigma_{i+1} + d)\}. \qquad (4)$$

In Fig. 2(b) we demonstrate a possible deformation of a template representing a fish.

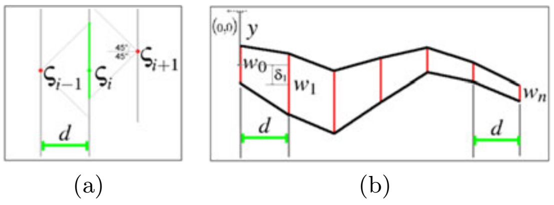

Fig. 2. a)Vertical shift margins allowed for slice starting point ς_i, relative to ς_{i-1} and ς_{i+1}. b)A possible deformation for a fish template.

2.3 Energy Function

The energy function Φ, is a fitness function that specifies to what extent an identified shape in the image matches the deformed template. In the literature the probability of a deformed template being located over a shape is referred to as the likelihood [1].

Let us now introduce the details of the computation of Φ:

Consider the fitness ϕ_i of a single slice i. In order to find the optimum we wish to maximize matching values (0 or 1) between pixels covered by the slice, i.e. pixels with value 1 contained in w_i and pixels (at top and bottom of the slice) with value 0 in b_i. $S_i[j]$ represents the j–th element (pixel) of slice S_i, as counted from the top of the slice.

$$\phi_i = \frac{1}{w_i + b_i * 2} \sum_{j=0}^{w_i+2*b_i} H_j, H_j = \begin{cases} 1, & \text{if } M(x, j + y + \delta_i) = S_i[j] \\ 0, & \text{otherwise} \end{cases}$$

In Fig. 3 an example of a matching is given. The location of proposed matching is denoted in pattern. For this example:

$$\phi_i = \frac{1+0+0+1+1+1+1+1+0+1+1+1}{12} = 0.75$$

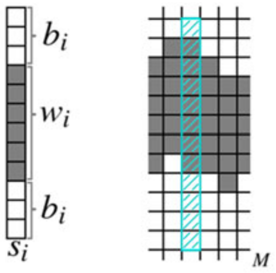

Fig. 3. Example of matching a slice S_i on to a location in matrix M

The deformation I consists of multiple slice shifts. Total fitness of all n slices is given by:

$$\Phi = \frac{1}{n} \sum_{i=0}^{n} \phi_i. \tag{5}$$

3 Genetic Algorithm

In this section we discuss the major components of a Genetic Algorithm (GA), i.e., population, evaluation, selection, crossover and mutation respectively. In a GA, a population P (of size m) of candidate solutions is evolved toward better solutions by introducing computer analogues for recombination, mutation and selection.

A candidate solution is also referred to as an *individual.* The outline of a generic GA pseudo code reads:

```
1:  $t = 0$
2:  Initialize $P(t)$
3:  Evaluate $P(t)$
4:  **while** not terminate **do**
5:      $P'(t)$ = SelectMates $(P(t))$
6:      $P''(t)$ = Crossover $(P'(t))$
7:      $P'''(t)$ =Mutate $(P''(t))$
8:      Evaluate $P'''(t)$
9:      $P(t+1) = P'''(t)$
10:     $t = t + 1$
11: **end while**
```

The termination criterion can differ, depending on the problem at hand. The details of the operators are dealt with in more detail in the following subsections.

3.1 Representation of Individuals

A shape I based on the template T_0, also referred to as *individual* is then represented in the image as, and consists of the following genomes:

$$I = (x, y + \delta_0, y + \delta_1, .., y + \delta_n). \tag{6}$$

The individuals are initialized with randomly valued genomes. According to common practice in the GA research, a population is generated with random genome values, covering the entire range of possible solutions. For finding shapes in images, this means random shapes are initialized on random location in the target image with a random deformation according to Eq. 4.

3.2 Evaluation

The fitness function determines the quality of an individual and depends on the problem at hand. Function Φ (cf. Eq.5) will serve as a fitness function for the genome values of an individual; Φ is then used to evaluate the candidate solutions in the selection step.

3.3 Selection

For the selection operation the, so called, tournament selection scheme [2] is used in order to prevent premature convergence. Tournament size, i.e. k_{size}, was determined through empirical testing. Consequently, comparison of high and low k_{size} is given in Fig. 4. We have established $k_{size} = 7$ to be a good value for our experiments.

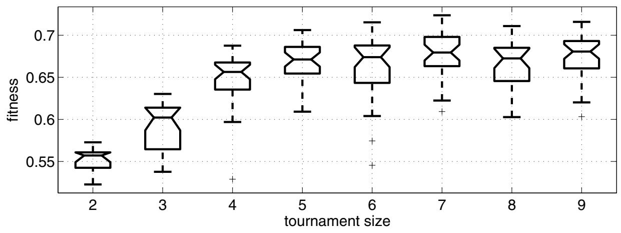

Fig. 4. Fitness comparison for different threshold values in 30 runs

3.4 Crossover

A crossover is applied with single crossover point. A random point $p \in (0, n)$ is chosen where n denotes the length of the genome. After crossover of two individuals I_J and I_K the resulting individual I_L has the following form:

$$I_L = (x_K, y_K, \delta_{0_K}, .., \delta_{p_K}, \delta_{p+1_J} + a, .., \delta_{n_J} + a). \tag{7}$$

Variable a is needed to make sure Eq. 4 holds and is determined by:

$$a = \begin{cases} \delta_{p_K} - d, & \text{if}(\delta_{p_K} - \delta_{p+1_J} > d) \\ \delta_{p_K} + d, & \text{if}(\delta_{p_K} - \delta_{p+1_J} < -d) \\ 0, & \text{otherwise} \end{cases}$$

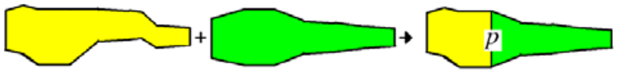

Fig. 5. Graphical representation of crossover function as derived from the data. Typical example in zebrafish imaging.

3.5 Mutation

For the mutation operator we use standard settings commonly used for GA's. That is a uniform mutation, where each genome g has the probability to mutate: $1/n$ [14], where n is the genome length. For each genome:

$$g\ U(\bar{g}, \underline{g})$$

To make sure a uniform mutation is used we allow every slice center to mutate anywhere within the image space. Since the constrain $\max\{(\varsigma_{i-1} - d, \varsigma_{i+1} - d)\} < \varsigma_i < \min\{(\varsigma_{i-1} + d, \varsigma_{i+1} + d)\}$ holds, subsequent slice centers are not allowed to be more separated then distance d.

4 Multiple Object Recognition

The shape under study can have multiple slightly different (deformed) instances in one and the same image. The complete search space in this image is denoted as M_0. First the best matching shape S_0 (with highest fitness) is localized. To decrease the search space for finding the next shape instance, we set all the elements contained within $S_0 \in M_0$ to 0 and name the resulting image M_1.

This process is repeated by iteration and in that manner M_i is reduced for each next identified shape i. No shapes are found if fitness of the found optimum $F(S_i)$ drops drastically; under a predefined threshold value r. The value of r can be determined empirically from a test on similar images (acquired under the same conditions). In Figure 6 an example of such drastic fitness drop in fitness growth is depicted. This is a fitness plot for an image containing 3 fish shapes (cf Fig. 8(a)).

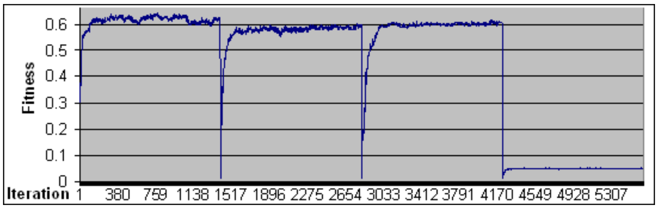

Fig. 6. Fitness evolution of 3 fish shapes found in an image. Every 1500 generations the found shape S_i is extracted and the algorithm is restarted on M_{i+1}. After finding S_0, S_1 and S_2 with fitnesses over 0.5, at the fourth run the maximum fitness can not get over 0.1. This is what we consider a fitness drop. For this type of images r should be chosen somewhere between 0.1 and 0.5, e.g. 0.4 is a good value.

The pseudo code for finding multiple shapes in an image can be written as:

```
1: $i = 0$
2: $S_0 = \mathrm{GA}(M_0)$
3: Evaluate $P(t)$
4: **while** $F(S_i) > r$ **do**
5:     save $S_i$ as found shape
6:     $M_i = M_{i-1} - S_i$
7:     $i = i + 1$
8:     $S_i = \mathrm{GA}(M_i)$
9:     $P(t+1) = P'''(t)$
10:    $t = t + 1$
11: **end while**
```

5 Experiments

To evaluate the performance of our template representation and GA optimization we have designed the task of finding objects based on predefined slice templates in images with different type of content. An experiment was performed with templates i.e. in synthetic as well as microscope images. With a simple interface the user selects both input images as well as the template shape for analysis.

5.1 Testing with Synthetic Images

We have used 20 synthetic binary images of 388 × 291 pixels. We have generated the images with different shapes located at different locations in images, slightly rotated, skewed and missing pixel data. Random noise (drawing debris) is introduced. The images randomly contain up to 3 instances of an object.

First template used for the synthetic shapes is a template of an animal figure presented in Fig. 7(a). We have created a synthetic binary image containing one deformed animal shape. The result of the application of the GA optimization is depicted in Fig. 7(c).

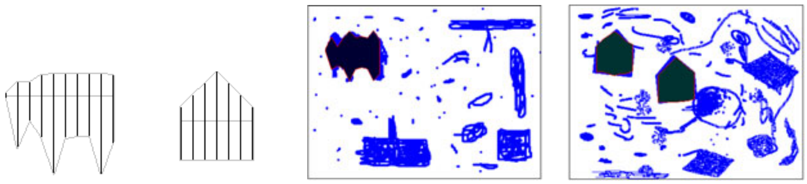

Fig. 7. a)Very simple template of an animal shape (head, legs and body) b) Very simple template of an house shape c) Result of shape localization in a synthetic image containing a simple animal shape d) Result of shape localization in a synthetic image containing two simple house shapes

Second template used for the synthetic shapes is a template of a house figure presented in Fig. 7(b). We have created a synthetic binary image containing two figures that have a deformed house shape. The result of the algorithm (implementation done in Delphi) is shown in Fig. 7(d).

5.2 Testing with Zebrafish Images

To evaluate the performance of our algorithm in a real–world imaging application we have chosen the task of finding zebrafish embryo shapes. The task at hand concerned using a High Throughput (HT) segmentation technique for retrieving the location and number of zebrafish embryo objects within images [5]. An additional requirement for this application is the need for automatic recognition of head, body and tail of each embryo. A typical binary image as presented for localization is shown in Fig. 8(a). This image was converted from color scale images to binary in a preprocessing step. We use a straightforward gradient operator (Sobel) followed by an iso–data threshold. In that way strong edge pixels were extracted for a binary representation [13].

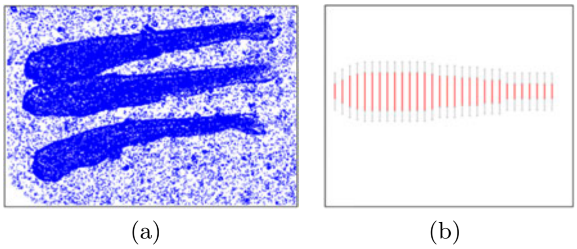

Fig. 8. a) Typical binary image of zebrafish embryos in a resolution of 388 × 291 pixels. b) The template T_0 in a graphical representation that has been used on 100 tested images. Red lines represent w_i and gray represent b_i within slices. Some results are shown in Fig. 9.

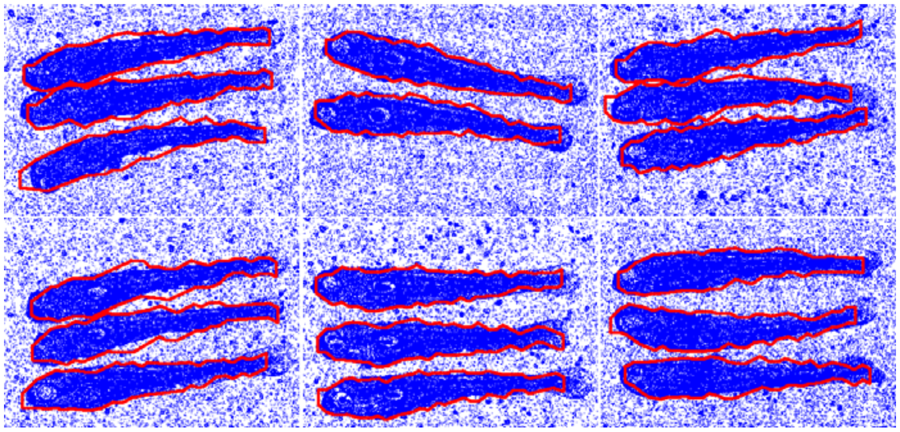

Fig. 9. Some results of shape counting and localization by our algorithm (Delphi implementation) on binary images. Rotated, slightly overlapping and bended objects could be retrieved. The algorithm needed about $2s$ CPU time for the retrieval of one shape on a Intel Dual Core 2.66Ghz, 1.00Gb.

Each image contains multiple zebrafish embryo shapes. The number of shapes in an image is not known in advance, the shapes might be overlapping. The shapes in all the images are assumed to be located approximately horizontal with a maximum angle of 45 degrees; so our approach could be used. We have applied the algorithm for a database of 100 images with the same settings for the GA (m = 200, k_{size} = 7, r = 0.5). For 87 images the amount of embryos in the image and the approximation of their shape could be retrieved correctly. In the cases where the algorithm failed, it was mostly due to large occlusion overlap or shapes were much longer or shorter than the proposed template. For the small and medium occlusions and overlap the algorithm performed correctly (cf Fig. 9). In Fig. 8(b) the template of a zebrafish used in these results is shown in a graphical representation.

6 Conclusions and Discussion

In this paper we have illustrated an application of a GA to optimize a deformable template approach. This method can be used in different fields as the template can represent different shapes. Our approach was designed for an application in the HT screening of zebrafish embryos [5]. The optimization of template parameters is done through a Genetic Algorithm, which provides the possibility to search for an optimal solution in large search spaces. Our approach also allows to retrieve multiple instances of a certain object in a single image. Results indicate that this approach has a low error rate while computational performance is manageable and fast, such is, of course, very suitable for HT applications. Future work is directed towards a further generalization of the approach and making the template representation scalable.

Acknowledgments

This work was supported by the Smartmix program. Special thanks for providing the images to Drs. E. Stoop and Dr. A. vd Sar and for the suggestions to Dr. M. Emmerich and Drs. J.Kruisselbrink.

References

1. Jain, A.K., Zhong, Y., Lakshmanan, S.: Object matching using deformable templates. IEEE Tran. on Pattern Analysis and Machine Intell. 18(3) (1996)
2. Miller, B.L., Goldberg, D.E.: Genetic algorithms, tournament selection, and the effects of noise. Complex Systems (1995)
3. Kervrann, C., Heitz, F.: A hieraerchial statical framework for the segmentation of deformable objects in image sequences. IEEE Comput. Vision Pattern Recogn. (1994)
4. Goldberg, D.E.: Genetic algorithms in search, optimization and machine learning. Kluwer Academic Publishers, Dordrecht (1989)
5. Stoop, E., et al.: Zebrafish embryo screen for mycobacterial genes involved in granuloma formation reveals a novel esx-1 component (submitted) (2010)
6. Yao, J., et al.: Fast robust ga-based ellipse detection. ICPR 2(2) (2004)
7. Krolupper, F., Flusser, J.: Polygonal shape detection for recogn. of partially occluded objects. Pattern Recogn. Lett. 28, 1002–1011 (2007)
8. Shapiro, L., Stockman, G.: Comput. Vision. Prentice-Hall, Englewood Cliffs (2002)
9. Kass, M., Witkin, A., Terzopoulos, D.: Snakes: Active contour models. Int. Journal of Comput. Vision. 1(4) (1987)
10. Funobiki, N., Isogai, M.: An eye-contour extraction algorithm from face image using deformable template matching. Mem. of the Fac. of Eng. 40, 78–87 (2006)
11. Nohre, R.: Deformed template matching by the viterbi algorithm. Pattern Recogn. Lett. 17(14) (1996)
12. Ramirez, A.: Circle detection on images using genetic algorithms. Pattern Recogn. Lett. 27(6) (2006)

13. Gonzales, R., Woods, R.: Digital Image Process, 2nd edn. Addison-Wesley, London (2001)
14. Bäck, T.: Evol. Algorithms in Theory and Practice: Evol. strategies, Evol. Programming, Genetic Algorithms. Oxford Univ. Press, Oxford (1996)
15. Cootes, T.F., Taylor, C.J., Cooper, D.H., Graham, J.: Active shape models - their training and application. Comput. Vision and Image Understanding 61(1) (2009)
16. Zhong, Y., Jain, A.K.: Object localization using color, texture and shape. Pattern Recogn. 33 (2000)