We are IntechOpen,
the world's leading publisher of
Open Access books
Built by scientists, for scientists

4,800
Open access books available

122,000
International authors and editors

135M
Downloads

Our authors are among the

154
Countries delivered to

TOP 1%
most cited scientists

12.2%
Contributors from top 500 universities

CLARIVATE ANALYTICS
BOOK CITATION INDEX
INDEXED

WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index
in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Parametric Circle Approximation Using Genetic Algorithms

Victor Ayala-Ramirez, Raul E. Sanchez-Yanez,
Jose A. Gasca-Martinez and Sergio A. Mota-Gutierrez
Universidad de Guanajuato FIMEE
Salamanca, Mexico

1. Introduction

Pattern recognition is one of the main research areas in the computer vision community. The basic problem consists in detecting and recognizing one or several known patterns in a data stream. Patterns can be specified in the raw data space or in any feature space suitable for the analysis of the input data. In particular, visual pattern recognition deals with applications where the pattern description is specified in terms of visual information and where input data comes from any kind of visual sensor (Chen & Wang, 2005). A common classification scheme divides pattern recognition according to the way the pattern is defined. There are then structural and statistical techniques. With respect to the problem solving approach, object location techniques are divided into two types of methods: i) deterministic methods like Hough transform, e.g. (Yuen et al., 1990), geometric hashing and template or model matching, e.g. (Iivarinen et al., 1997; Jones et al., 1990) and ii) stochastic techniques, including RANSAC (Fischer & Bolles, 1981), simulated annealing and genetic algorithms (Roth & Levine, 1994).

Geometric shapes are very useful in a number of tasks because they are often present in human-made environments. They are also widely used as a part of man-designed symbols. To recognize this kind of shapes, many methods have been developed. In particular, circle and ellipse detection problems have been widely studied in the shape recognition community. Most approaches use Hough transform-based techniques. For instance, Lam and Yuen (Lam & Yuen, 1996) have proposed to use a hypothesis filtering approach to a Hough transform to detect circles in images. Yuen and Lo (Yuen & Lo, 1994) have posed the circle detection problem as a multi-resolution problem for the Hough transform. A traffic sign detector proposed by Mainzer uses a circle detector (Mainzer, 2002a,b). Shape classification using a soft computing approach is addressed by Rosin and Nyongesa (Rosin & Nyongesa, 2000).

Genetic Algorithms (GA), proposed by Holland (Holland, 1975) in the 60s, are a family of algorithms where we apply an artificial evolution process. The purpose of the evolution is to produce a computational individual which is the best fitted to solve a specific problem. GA have been extensively used to solve optimization problems. Genetic Algorithms (GA) have been already used for several pattern classification tasks. In order to apply them to a pattern recognition problem, we need to pose the pattern recognition problem as an optimization problem. We need then to associate key elements in both approaches. Bandyopadhay

(Bandyopadhay et al., 1995) have proposed a GA-based method to classify patterns by selecting the position of the decision boundaries in an N-dimensional feature space. They place H hyper planes in order to minimize misclassification of the sample points. They have also developed some methods to remove the redundant hyper planes originated by the overestimation of H (Pal, et al., 1998). Decision boundaries selected by the GA-based pattern classification method will approach those found by the Bayes maximum likelihood classification (Murthy, et al., 1996). Van Hove and Verschoren (van Hove & Verschoren, 1996) have also applied a GA-based method in the recognition of binary trees that represent bitmap images. They propose to use a two dimensional GA (2DGA) as a model to use GA in image processing tasks. Buckles et al. (Buckles, et al., 1996) apply a GA to label mesoscale features arising from satellite images. They use a combination between semantic nets and fuzzy predicates to improve labeling of satellite images.

We are interested in the problem of interacting with a computer by using graphical sketches to simplify its use by novice users. Sketch understanding is a research field where shape recognition plays a role to complete successfully the task (Notowidigdo & Miller, 2004). Hand-drawn input has been already used in a number of human-computer interface applications.

In this application context, we are interested in approximating geometrical shapes drawn by a human user to parametric ones to be handled by a computer user interface. As a first example, we show how to approximate circles drawn by a human user to ideal circles by using a GA-based approach. Using GAs to detect shapes in an image involves mainly the making of design choices for the solution elements in a genetic algorithms framework. We work on images containing one or several circles. The circles are searched through the edge image resulting from an image pre-processing step. A classical Sobel edge detector was used for this purpose. Our algorithm can detect one or many circles present in the image under analysis and estimate shape parameters for them. Parametric estimation is achieved with sub-pixel accuracy. In the rest of this paper, we will show how to pose the circle detection problem in terms of a genetic algorithm approach. Section 2 presents an overview of GA. We show how GA can be used to solve a circle detection task in Section 3. The test protocol and the performance evaluation results for the proposed approach are shown in Section 4. Conclusions arising from these results are presented in Section 5. Some extensions to this work are also depicted in this section.

2. Genetic algorithms

Genetic algorithms are a pseudo-random optimization technique that mimics natural evolution to find the solution of the problem being solved. The block diagram of the Simple Genetic Algorithm (SGA) is shown in Figure 1. As described by this diagram, to use a SGA, we need to initialize a population of candidate solutions. Each potential solution is represented as a computational individual where the genetic material is encoded in the form of on and off bits. We name computational chromosome to such a string of bits. There is also an encoding and decoding function that let us go both ways from the solution space to the chromosome space. Every computational individual is evaluated in order to find the best solutions among the population. We need to define the so called fitness function in order to assign a fitness value to each potential solution. The fitness value encodes the knowledge we have about the characteristics of the best solution and tries to reward potential solutions that best fit these features. Best solutions are kept and they are given the privilege of mating to generate a new population that will be analyzed in the same way, until a stopping condition is fulfilled. Genetic operators like mating and mutation operate directly on the bit string

representing the parent individuals in order to generate bit patterns that represent best-fitted solutions to the problem being solved.

Main advantage of such stochastic optimization is the possibility of escaping from local minima through the mutation, selection and mating operations. Main drawback is the need of definition of a good fitness function that reflects how well any bit pattern performs in the solution space. For the sake of completeness, we are going to present an overview of the key elements of a GA.

2.1 Encoding mechanism

The encoding scheme used in a GA has the function of representing in a unique way each solution in the problem search space by using one or several computational chromosomes. We understand by computational chromosome a concatenated string of symbols, where the symbols are chosen from a particular alphabet. Most often used representations are based in a binary alphabet. That is, every potential solution is represented as a string of bits. We can use binary strings to encode candidates in almost any optimization problem with good results. Many optimization problems require using real precision values as the computer implementation data type. We can use integer representations to simplify computational handling of the information. However, we use a mapping function to decode the integer index of the candidate into a real value.

2.2 Population initialization

GA work on an initial population searching for best fitted individuals. Each individual corresponds to a potential solution of the optimization problem. We can use any a priori procedure to populate the GA. Such initialization schemes could exploit a priori knowledge on the specific problem to be solved. Nevertheless, and as a mean of providing diversity to the GA population, this step is done by randomly selecting a number of individuals from the search space.

2.3 Fitness evaluation

We need to evaluate how well each candidate solution performs. To do this, we use a function $f(x)$ that is optimized when x assumes the value that solves the problem. In this function, we need to encode all the knowledge about the problem being solved. For each potential solution, this function returns a fitness score used to rank potential solutions. In this step, the bit string representing the solution (also called the genotype) is decoded to get the features associated to such a bit string (also called the phenotype). These parameters are used by the fitness function to compute the fitness score. Fitness function complexity can vary depending on the problem being solved. The fitness score is always the key to decide which individuals remain being part of the GA population.

2.4 Selection mechanism

The selection mechanism let us set up survival strategies in the evolutionary framework. One of the simplest survival strategies to be considered is the best fitted survival. In this method, individuals which have the better fitness score are kept to participate with its genetic code to the next iteration of the GA population. We need to define how many parent individuals will be selected to mate among them in order to generate a new population for the GA, and how much will each of them contribute to the next generation of the GA.
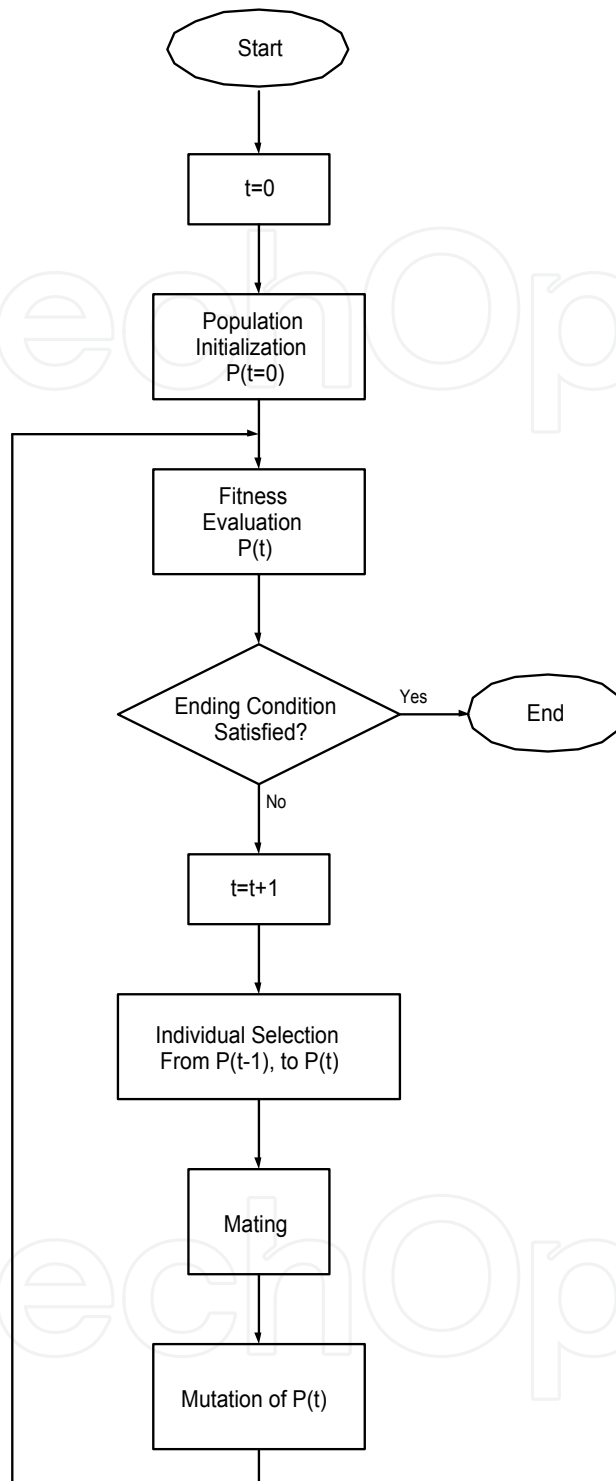
Fig. 1. A block diagram of the operations involved in a SGA.

2.5 Mating procedure

In order to generate a new population of candidate solutions, a mating operator is used. We choose the parents to be mated and how they will exchange genetic material to generate new individuals. We need to specify a crossover rate that defines the probability of a genetic

code exchange to occur. At each generation, there are a number of individuals that are kept as elite individuals. They possess the best fitness scores among the genotype population currently being analyzed, and they are kept for the fitness function to follow a monotonic behavior in its temporal evolution.

2.6 Mutation operator

The mutation operator is a random perturbation operator used to change in a slightly sort the genetic material of the computational chromosomes. The application of such an operator is intended to maintain the population diversity and to avoid the premature convergence of the algorithm in a local optimum. The operator is represented by a mutation probability value. This value is proportional to the easiness of a random change in each bit in the genotype. It is recommended to use small values for this probability.

3. Circle approximation using genetic algorithms

3.1 Related work

Chakraborty and Deb (Chakraborty & Deb, 1998) have proposed to use a GA in combination with the Randomized Hough Transform (RHT) to detect analytic curves in images. In their work, they extract straight lines from an image. They use GA to guide the random search associated to the RHT. Roth and Levine (Roth & Levine, 1994) have proposed to use minimal point subsets to define geometric primitives. Lutton and Martinez (Lutton & Martinez, 1994) have improved the shape parameters to reduce redundancy in the primitive representation. They show an implementation to detect several shapes in images including segments, circles and ellipses in synthetic and real images. Yuen and Ma (Yuen & Ma, 2000) have proposed to use a GA to detect shape templates in images. They detect shapes with up to 6 degrees of freedom using a multi-modal GA. A multi-population GA is used by Yao et al. (Yao et al., 2004) to detect ellipses in images. They use the coordinates of five points as the genes of the computational individuals. In the work by Ayala et al. (Ayala et al., 2006), it is proposed to arrange all the edge points in an array and to use only the index of the point in this array as the encoding value of the point. Their approach reduces dimensionality of the search space. A recent application (Moreno et al., 2008) uses a variant of the former approach for depth estimation of micro-sized particles. They use a SGA to identify the minimal intensity ring on an image generated by the diffraction of the light of a laser beam when micro-sized particles are illuminated. As said before, the characteristics of that ring (namely the central spot size, CSS) identify the depth of the small particle in a microscopic test setup. In order to apply a SGA to the CSS measurement problem, they have used a fitness function that concurrently optimizes the existence of low- intensity pixel points along the circumference of the candidate solution and the low intensity average of the individual under test.

3.2 Our approach

Our method uses the following steps:
i.   Firstly, a binary image is acquired as input. We consider this image as figure-background edge image. Such an image can contain any number of shapes; however, at this time we are only interested in detecting circles.
ii. We create a vector by considering only the figure points in the image under analysis. To decide which points are the figure points, we need to consider the application context.

According to the acquisition system convention, figure points could be black or white points in the input image.

iii. We set up a simple genetic algorithm to search for the best circle in the image under analysis. Circles are encoded as three integer vector indexes in the GA chromosome. These indexes define a circle in a continuous parametric space (x, y, r). Circle optimality is defined by using a fitness function that validates if the candidate circle is actually present in the test image. Presence of the candidate circle is verified in a set of points along the circumference of the candidate.

iv. The search procedure in the reduced sub-space leads to a circle finding algorithm fast enough to identify a circle in an input image. The best individual of the genetic population is considered as the best solution to generate the approximated circle. In the case of multiple circles present in the input image, the best circle is deleted from the input image and steps iii) and iv) are repeated.

3.1 GA design

Using GAs to detect shapes in an image involves mainly the making of design choices for the solution elements in a genetic algorithms framework. We work on images containing one or several circles. The circles are searched through the edge image obtained from an image pre-processing step. A classical Sobel edge detector is used for this purpose. In the following paragraphs we show how to pose the circle detection problem in terms of a genetic algorithm approach as stated by (Ayala et al., 2006).

3.2 Individual representation

Each individual C uses three edge points as chromosomes. Edge points are represented by their relative index in a list V of all the edge points resulting from the edge extraction step. Each individual represents then a feasible circle where their (x_0, y_0, r) parameters are defined as follows:

$$(x - x_0)^2 + (y - y_0)^2 = r^2 \tag{1}$$

with:

$$x_0 = \frac{\begin{vmatrix} x_j^2 + y_j^2 - (x_i^2 + y_i^2) & 2(y_j - y_i) \\ x_k^2 + y_k^2 - (x_i^2 + y_i^2) & 2(y_k - y_i) \end{vmatrix}}{4((x_j - x_i)(y_k - y_i) - (x_k - x_i)(y_j - y_i))} \tag{2}$$

$$y_0 = \frac{\begin{vmatrix} 2(x_j - x_i) & x_j^2 + y_j^2 - (x_i^2 + y_i^2) \\ 2(x_k - x_i) & x_k^2 + y_k^2 - (x_i^2 + y_i^2) \end{vmatrix}}{4((x_j - x_i)(y_k - y_i) - (x_k - x_i)(y_j - y_i))} \tag{3}$$

and

$$r = \sqrt{(x - x_0)^2 + (y - y_0)^2} \tag{4}$$

We can then represent the shape parameters (for the circle, *[x₀, y₀, r]*) as a transformation *T* of the edge vector indexes *i, j, k*.

$$[x_0, y_0, r] = T(i, j, k) \tag{5}$$

This approach enables us to sweep a continuous space for the shape parameters while keeping a binary string for the GA individual. We can then reduce the search space by eliminating unfeasible solutions.

3.3 Fitness evaluation

Each individual has a fitness value proportional to the number of actual edge points matching the locus generated by the parameters of the shape (x_0, y_0, r). In our practical implementation, we can not test for every point in the feasible circle so we perform a uniform sampling along the circumference. If we take N_s points, we construct an array of points $S_i = (x_i, y_i)$. Their coordinates are given by:

$$x_i = x_0 + r \cdot \cos \frac{2\pi i}{N_S} \tag{6}$$

$$y_i = y_0 + r \cdot \sin \frac{2\pi i}{N_S} \tag{7}$$

Fitness function *F(C)* accumulates the number of expected edge points (i.e. the points in the set S) that actually are present in the edge image $E(x_i, y_i)$. That is:

$$F(C) = \frac{\sum_{i=0}^{N_s - 1} E(x_i, y_i)}{N_S} \tag{8}$$

We also use some other factors to favour the context of specific applications for detection, including completeness of the circumference or a given size for the circles.

4. Tests and results

We present tests on images containing a single circle on several images. Our test set contains synthetic and hand-drawn images. Our method presents a good qualitative approximation over a set of test images.

4.1 Test protocol

Given that hand-drawn shapes cannot be standardized to some measure, we propose to measure the accuracy of our approach by using the following two quantitative tests: i) We have generated synthetic circles having a sinusoidal undulation effect along its circumference, and ii) We have also generated synthetic circles corrupted by additive Gaussian noise in the radius parameter. Figure 2 shows examples for both types of synthetic images. In both cases, we know the parameters of the perturbation functions and our goal is to measure how the results of our approach are related to the ground truth information.
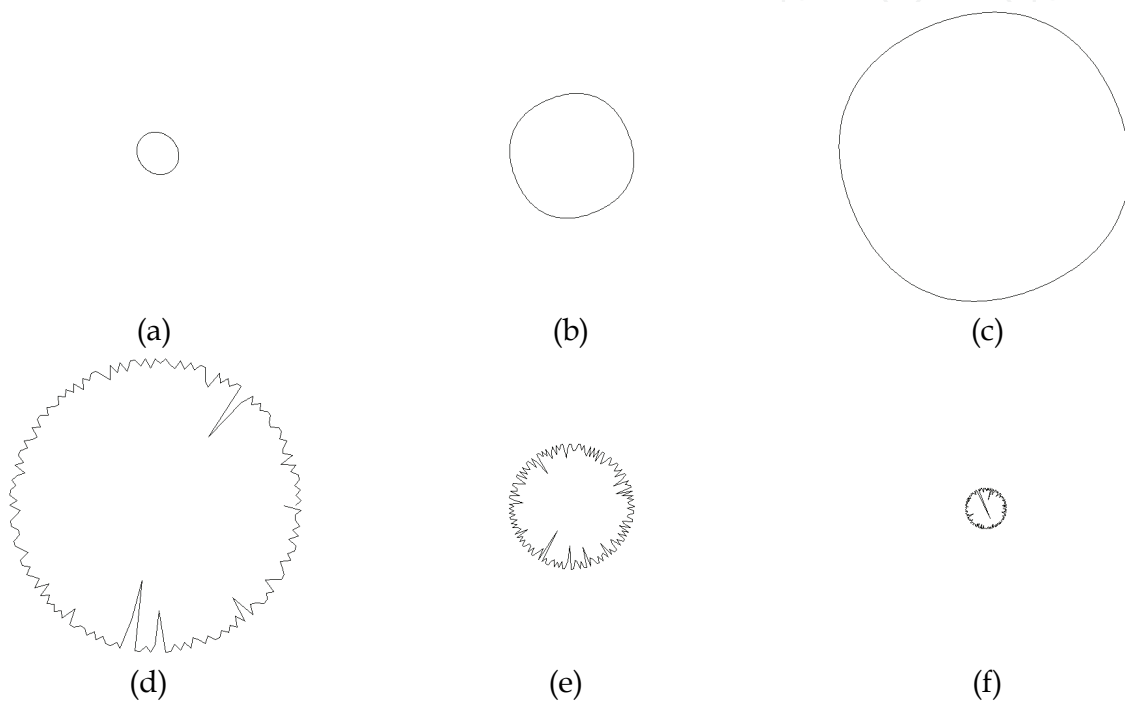
Fig. 2. Sample deformed circles: showing sinusoidal undulations along the circumference (a-c) and corrupted with noise in the radius parameter (d-f).

Circles perturbed by a sinusoidal undulation are described by the following equation in the polar plane (ρ, θ):

$$\rho = k + n \cdot sen\left(m\theta \right) \tag{9}$$

Where n is the amplitude of the sinusoidal perturbation and m is the number of periods that it will traverse along the circumference of the circle.

Noisy circles are constructed by computing a set of points modified with additive Gaussian noise in the radius parameter along the circumference of the circle. In the polar plane (ρ, θ), we have:

$$\rho = k + \eta\left(\mu, \sigma \right) \tag{10}$$

With μ being the mean value of the Gaussian pdf $\eta\left(\bullet \right)$ and σ being the standard deviation value for the noise pdf. In our tests, we have used $\mu=0$ in order to avoid biasing estimation.

Synthetic images were computed using VGA resolution (640×480pixels). Without loss of generality, we have placed the circle in the middle of the image.

A qualitative test was also performed, we have presented the systems with real human user drawn circles, and we have subjectively analyzed the results to gain insight on the advantages and inconveniences of the proposed method.

4.2 Test results
4.2.1 Circles corrupted by undulating sinusoids

A circle for each combination of the Cartesian product $k×n×m$, has been created, using k=*32, 96, 160, 224*; *n=2, 4*; and *m=2, 4, 8*, according to Equation 9. For each image, series of 30 estimations of the center *(x,y)* and radius *(k)* are performed and then the mean and standard deviation of such parameters are compared against those used for the ground truth construction. Just for comparison purposes, the circles estimated in one execution of the algorithm over those images in Figures 2 a-c, are shown as light gray overlays in Figures 3 a-c. Note in Figure 3a the circle detected for a figure containing two periods of the oscillating perturbation.

Results for the estimation of parameter *k* are given in Table 1. Here, each row corresponds to a *30* execution series. Mean and standard deviation values should be compared against the *n* value, which controls the deformation in the circles. We can see a small relative error of the estimated radius against the ground truth data except for some cases. In particular, let us consider the cases where *n=4*, *m=2, 4*. In these cases, the perturbed circle has lost its circular appearance, and it seems like a rounded rectangle. Our method tries to adjust the best circle in the image. So it gets in a random manner a circle inscribed on one of the four corners as the best result.

Parameters	k	μ_k	σ_k	e_k
$n = 2$, $m = 2$	32	29.1750	0.2711	2.8250
$n = 2$, $m = 4$	32	33.3615	0.8627	1.3615
$n = 2$, $m = 8$	32	33.3034	0.0761	1.3034
$n = 4$, $m = 2$	32	26.2145	0.1842	5.7855
$n = 4$, $m = 4$	32	29.4962	0.0661	2.5038
$n = 4$, $m = 8$	32	35.6446	0.0443	3.6446
$n = 2$, $m = 2$	96	92.5663	0.2959	3.4337
$n = 2$, $m = 4$	96	93.8680	0.1385	2.1320
$n = 2$, $m = 8$	96	97.7291	0.0577	1.7291
$n = 4$, $m = 2$	96	88.3233	0.3180	7.6767
$n = 4$, $m = 4$	96	67.4882	4.7472	28.5118
$n = 4$, $m = 8$	96	92.5621	0.0783	3.4379
$n = 2$, $m = 2$	160	156.5334	0.1239	3.4666
$n = 2$, $m = 4$	160	158.3205	0.0449	1.6795
$n = 2$, $m = 8$	160	161.6845	0.0413	1.6845
$n = 4$, $m = 2$	160	151.4321	0.3150	8.5679
$n = 4$, $m = 4$	160	155.5221	0.2715	4.4779
$n = 4$, $m = 8$	160	163.7453	0.0371	3.7453
$n = 2$, $m = 2$	224	220.2067	0.1070	3.7933
$n = 2$, $m = 4$	224	225.5948	0.0490	1.5948
$n = 2$, $m = 8$	224	222.5754	0.0719	1.4246
$n = 4$, $m = 2$	224	217.0377	0.1371	6.9623
$n = 4$, $m = 4$	224	227.4639	1.3441	3.4639
$n = 4$, $m = 8$	224	227.3497	0.0446	3.3497

Table 1. Results for the estimated radius.

(a)   (b)   (c)
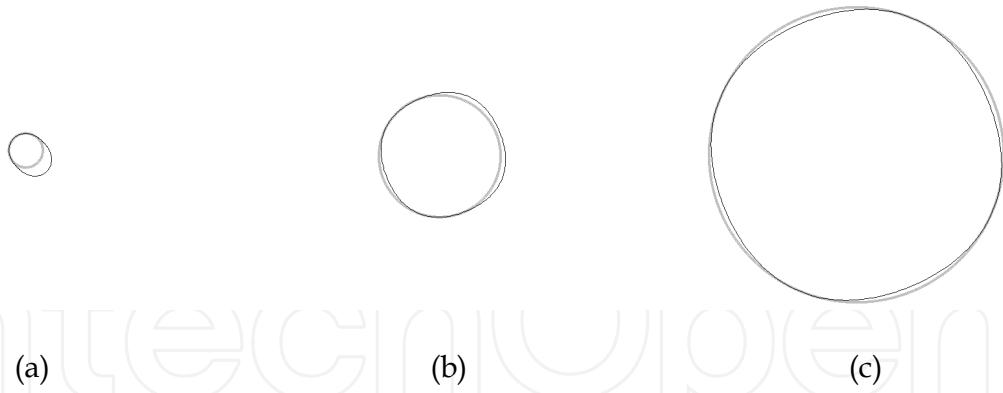
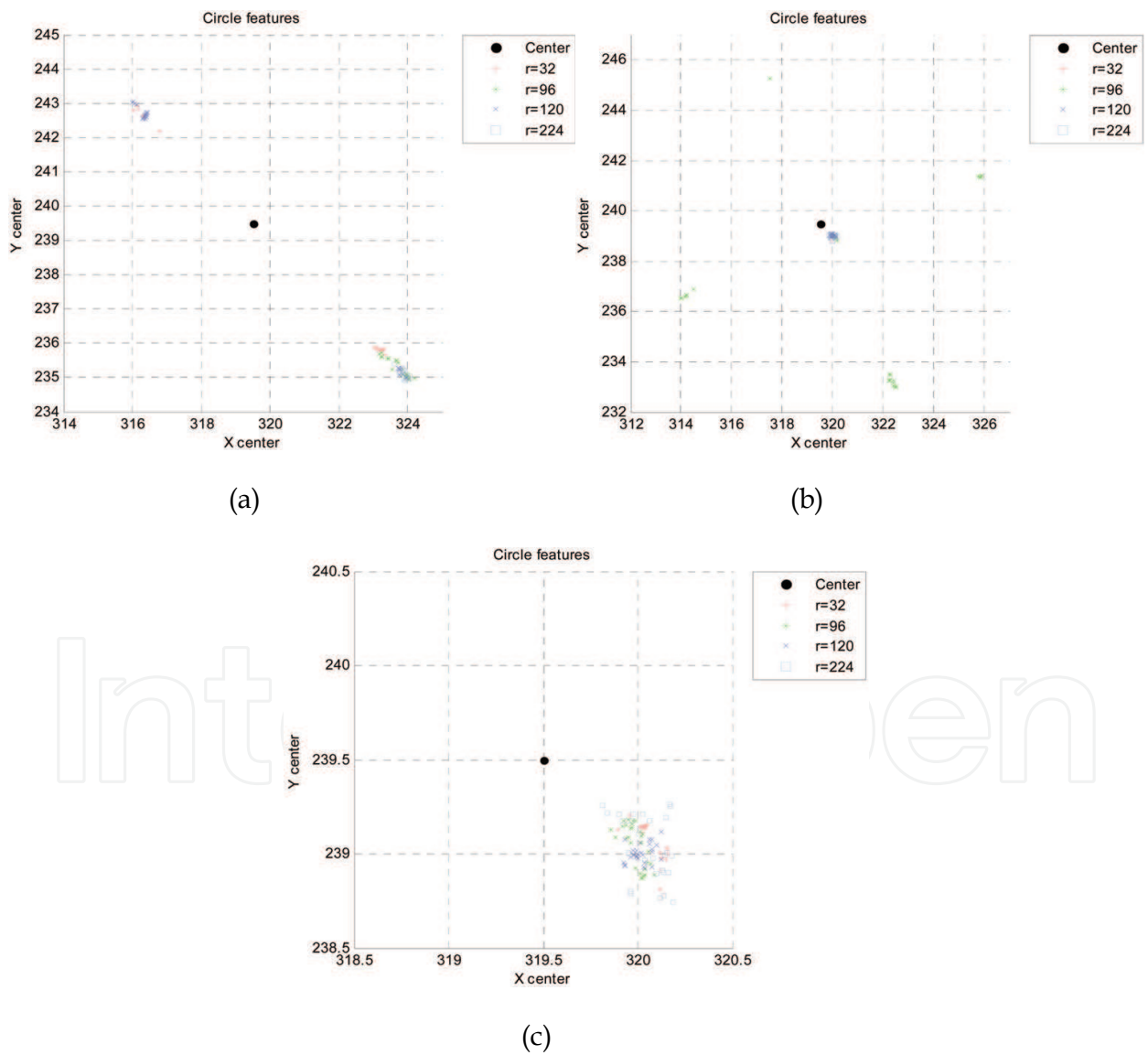Fig. 3. Deformed circles in Figure 2 a-c and the circles estimated using a GA.

(a)

(b)

(c)

Fig. 4. Center of circles estimations for $k=2$ and $m=2$ in (a), $m=4$ in (b), and $m=8$ in (c).
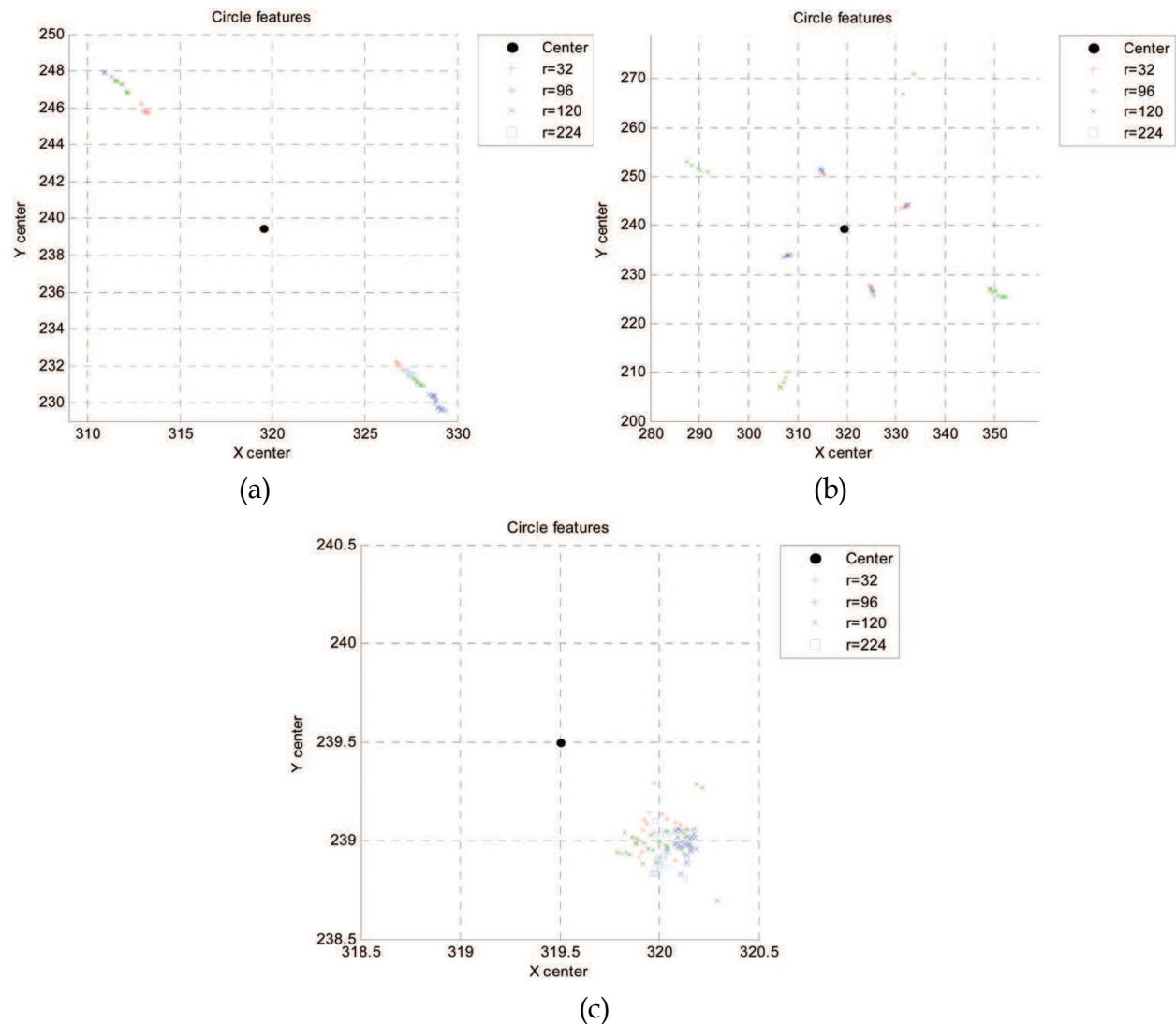
(a)

(b)

(c)

Fig. 5. Center of circles estimations for *k=4* and *m=2* in (a), *m=4* in (b), and *m=8* in (c).

Results for the center of circles estimations are given in Figures 4a-c for *k=2*, while Figures 5a-c show results for *k=4*. Observe in Figures 4a-b and 5a-b, the oscilating nature of the center estimation when we have *m=2* and *m=4* periods of the sinusoidal perturbation along the circumference of the circle.

We can observe on both figures, the cyclic nature of the estimated position of the center of the circle. As explained before, the center shift is provoked by the round polygon appearance of the perturbed circles. A difference with this trend is shown in Figures 4c and 5c, where the center estimation shows a sub-pixellic error (lower than 0.5 pixels). In this case, the perturbed circle seems more like a circle because of the low amplitude of the sinusoidal perturbation.

4.2.2 Circles corrupted by additive gaussian noise
A circle for each combination of the Cartesian product $k \times \sigma$, has been created, using k=*32, 96, 160, 224*, and σ=*4, 8, 16*, for the additive Gaussian noise perturbation, defined in Equation 10.

As for the circles distorted with sinusoidal waves, series of *30* estimations of the center *(x,y)* and radius *(k)* are performed for each image, and then the mean and standard deviation of such parameters are compared agaist those used for the ground truth construction.

Circles estimated in one execution of the algorithm over those images in Figures 2 d-f, are shown in Figures 6 a-c, in a light gray overlay. Note here that circle accuracy is higher than for the undulating distortion case. That happens because the distorted points follow a normal distribution in the radius coordinate in the polar plane.

| (a) | (b) | (c) |
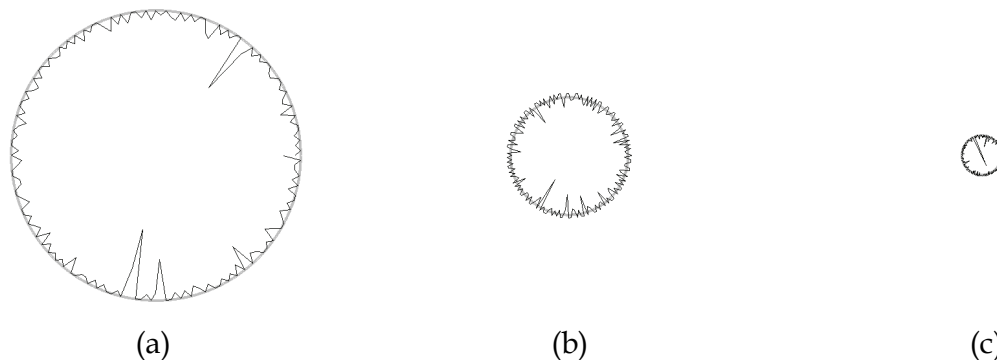
Fig. 6. Deformed circles in Figure 2 d-f and their corresponding circles estimated using a GA.

Results for the estimation of parameter *k* are given in Table 2, each row corresponding to a *30* execution series. Compare these results with those given previously in Table 1. In this kind of tests, error bound is lowered, as expected, with respect to the undulating circles.

Parameters	k	μ_k	σ_k	e_k
$\sigma =4$	32	31.6944	0.1056	0.3056
$\sigma =8$	32	30.4204	0.0813	1.5796
$\sigma =16$	32	24.9730	0.4624	7.0270
$\sigma =4$	96	95.7740	0.0293	0.2260
$\sigma =8$	96	95.1253	0.1399	0.8747
$\sigma =16$	96	91.0716	1.3534	4.9284
$\sigma =4$	160	159.6619	0.0189	0.3381
$\sigma =8$	160	159.2446	0.1209	0.7554
$\sigma =16$	160	159.7199	0.4473	0.2801
$\sigma =4$	224	223.6464	0.0277	0.3536
$\sigma =8$	224	222.8001	0.1105	1.1999
$\sigma =16$	224	223.6822	0.2805	0.3178

Table 2. Results for the estimated radius of noisy circles.

Plots of centers of circles estimated on tests are given in Figure 7. Note the high accuracy on parameter determination, yielding errors less than *1* pixel in magnitude.

4.2.3 Hand drawn circle approximation
Qualitative testing of our algorithm has been made by using as input images, circles drawn by a human user on a computer drawing program. Figure 8 shows eight hand-drawn circles by human users. A lighter gray overlay shows the best circle approximating the graphical sketch. As we can see in Figure 8, our approach results in good circle approximation. The

parametric circle found using our approach follows very closely the graphical sketch. As we can see in the results, the approximated circle sticks to the longer circular arcs present in the hand-drawn shape.
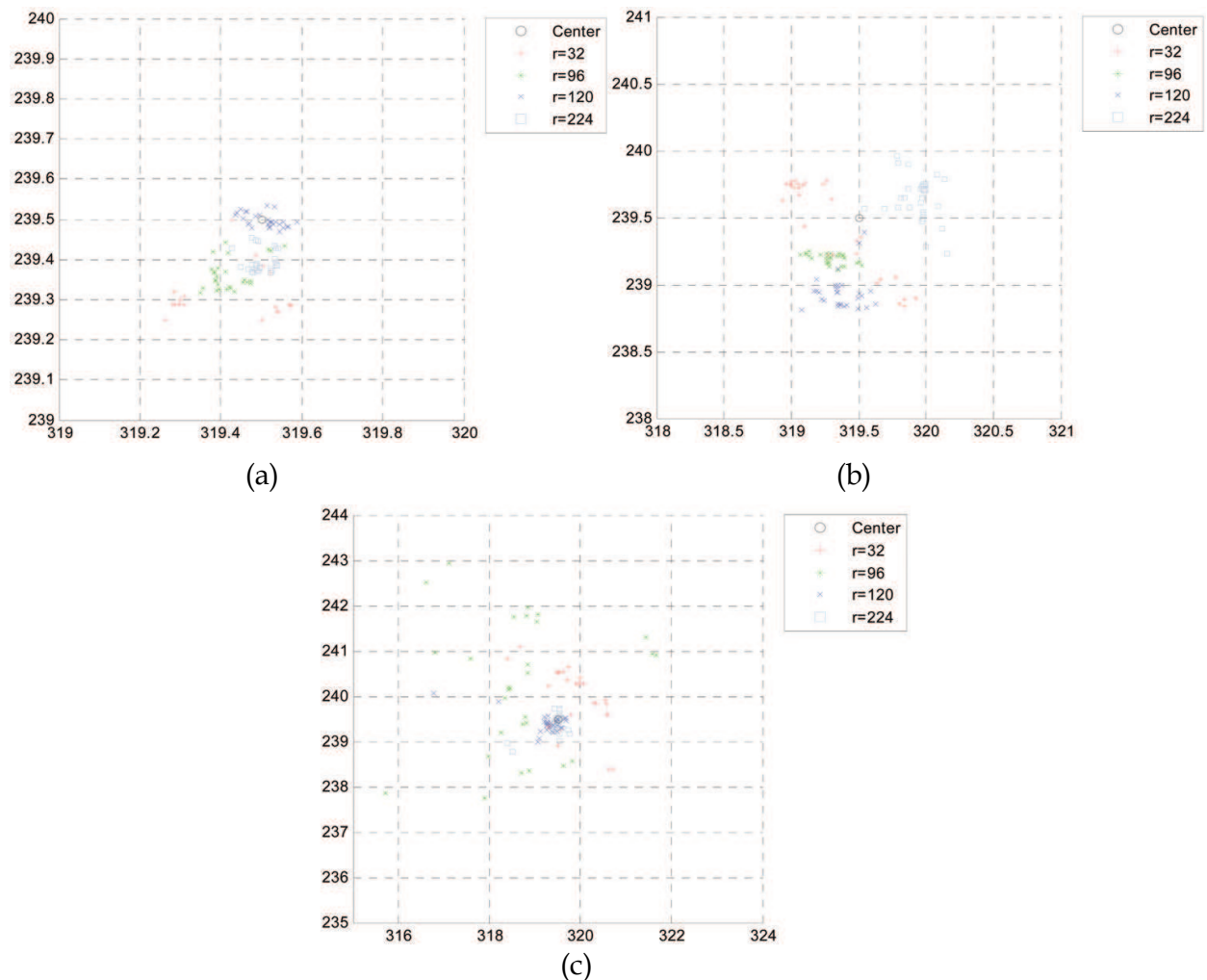

(a)   (b)

(c)

Fig. 7. Center of circles estimations for σ =4, 8,16 are shown in (a), (b), and (c), respectively.

4.3 Analysis and discussion

Tests have shown a good performance of our algorithm in both syntethic and real images using quantitative and qualitative tests. The undulation perturbation tests present to the algorithm smooth curves to be approximated by a circle. As expected, there is a bias in center position originated by the rounded corner effect generated by the sinusoidal waveform mounted on the circumference of the circle being detected. This effect is attenuated when the amplitude of the perturbating waveform is kept as a small fraction of the radius of the circle. That represents a more common case for circles drawn by human users.

With respect to the Gaussian noise test, we get accuracies in the sub-pixellic range because outlier points along the circumference generate candidate circles that are not supported by evidence. This leads the GA to choose as the best solution a circle bound by points in the ring where there is a larger density of points. In our case, the ring with the ground truth radius is very close to the found parameter.
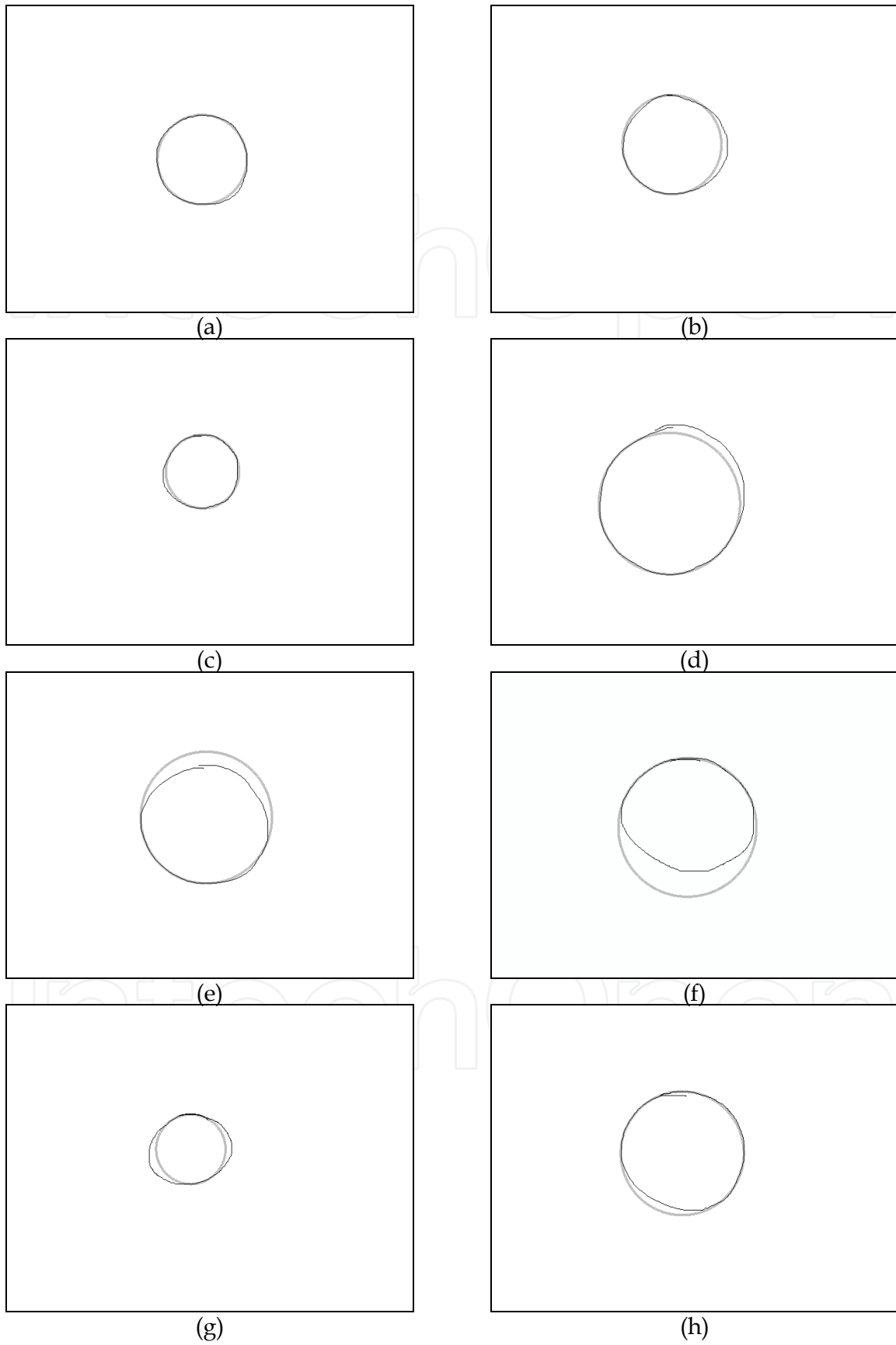
Fig. 8. Results for the circle detection using hand drawn shapes by human users.

5. Conclusions and perspectives

We have presented an algorithm that approximates circles by using a GA-based approach. Also, we have presented a brief overview of GA-based techniques and we have shown how to pose a pattern recognition problem to use GA for solving it. To know, we solve a circle approximation task.

We have executed two quantitative tests to evaluate the performance of the proposed method. We have also performed a subjective qualitative test over hand-drawn images. As a result, we have found in one hand, a high accuracy algorithm performance on circles corrupted by Gaussian noise. In the other hand, sinusoidal perturbations are better recognized if the amplitude of the perturbation is small with respect to the circle radius. For any other case, our algorithm will try to fit a circle to one of the rounded corners of the perturbed circle. We have also shown the result of a number of qualitative tests on human input. The results are promising enough to consider this as a building block for graphical sketch recognition.

We are developing at this moment other modules to recognize more geometrical shapes that can be expressed in parametric terms. We are developing modules to recognize ellipses, quadrilaterals and polygons in general.

6. Acknowledgements

7. References

Ayala-Ramirez, V.; Garcia-Capulin, C.H; Perez-Garcia, A. & Sanchez-Yanez, R.E. (2006). Circle detection on images using genetic algorithms, Pattern Recognition Letters, (27) pp. 652-657.

Bandyopadhyay, S.; Murthy, C.A. & Pal, S.K. (1995). Pattern classification Using Genetic Algorithm, Pattern Recognition Letters, 16(8), pp. 801-808.

Buckles, B.P.; Petry, F.E.; Prabhu, D. & Lybanon, M. (1996). Mesoscale Feature Labeling from Satellite Images, in Genetic Algorithms for Pattern Recognition, Pal, S.K. & P.P. Wang (Eds.), CRC Press, Boca Raton, FL., pp. 167-178.

Chakraborty, S. & Deb, K. (1998). Analytic Curve Detection from a Noisy Binary Edge Map Using Genetic Algorithm. 5th PPSN, Amsterdam, The Netherlands, LNCS (1498), pp. 129-138.

Chen, C.H. & Wang, P.S.P. (2005), Handbook of Pattern Recognition and Computer Vision, 3rd Ed., World Scientific, Singapore.

Fischer, M., Bolles, R., (1981) Random sample consensus: A paradigm to model fitting with applications to image analysis and automated cartography, *Comm. ACM*, Vol. 24, No. 6, pp. 381–395.

Holland, J. (1975). Adaptation in Natural and Artificial Systems, University of Michigan Press, Ann Arbor, MI.

Iivarinen, J.; Peura, M.; Sarela,J. & Visa, A. (1997). Comparison of combined shape descriptors for irregular objects, *Proc. 8th British Machine Vision Conf.*, pp. 430–439.

Yuen, H.; Princen, J.; Illingworth, J. & Kittler, J. (1990). Comparative study of Hough transform methods for circle finding, *Image Vision Comput.*, Vol. 8, No. 1, pp. 71–77.

Jones, G.; Princen, J.; Illingworth, J. & Kittler, J. (1990). Robust estimation of shape parameters. *Proc. British Machine Vision Conf.*, pp. 43–48.

Lam, W. & Yuen, S. (1996). Efficient techniques for circle detection using hypothesis filtering and Hough transform, IEEE Proc. Visual Image Processing, 143(5), pp.292-300.

Lutton, E. & Martinez, P. (1992). A Genetic algorithm for detection of 2D geometric primitives in images. Proc. of the IEEE Int. Conf. on Pattern Recognition (ICPR), (1) pp. 526-528.

Mainzer, T., (2002). Genetic algorithm for shape detection, Tech. Report DCSE/TR-2002-06, University of West Bohemia in Pilsen, Pilsen, Czech Republic.

Mainzer, T., (2002). Genetic algorithms for traffic sign detection, Proc. Int. Conf. on Applied Electronics, Pilsen, Czech Republic, pp. 129-132.

Moreno-Hernandez, D.; Ayala-Ramirez, V. & Guerrero-Viramontes, J.A. (2008) 3D positioning of micro-spherical particles by using genetic algorithms, Proc. LEOS Symposium on Photonics in Mexico (SPIM).

Murthy, C.A.; Bandyopadhyay, S. & Pal, S.K. (1996). Genetic algorithm-based pattern classification: relationship with bayes classifier, in genetic algorithms for pattern recognition, Pal, S.K. & P.P. Wang (Eds.), CRC Press, Boca Raton, FL., pp. 127-144.

Notowidigdo, N. & Miller, R.C. (2004) Off-line sketch interpretation. Proc. of AAAI Fall Symp. On Making Pen-Based Interaction Intelligent and Natural, pp. 120-126.

Pal, S.K.; Bandyophadyay, S. & Murthy, C.A. (1998), Genetic algorithm for generation of class boundaries. IEEE Trans. On Systems, Man and Cybernetics B, 28(6), pp. 816-828.

Rosin, P.L. and Nyongesa, H.O., (2000), Combining evolutionary, connectionist, and fuzzy classification algorithms for shape analysis, In Proc. EvoIASP, Real World Applications of Evolutionary Computing. pp. 87-96.

Roth, G. & Levine, M.D. (1994). Geometric primitive extraction using a genetic algorithm., *IEEE Trans. Pattern Anal. Mach. Intell.*, Vol. 16, No. 9, pp. 901–905.

Van Hove, H. & Verschoren A., (1996). Genetic Algorithms and Recognition Problems, in Genetic Algorithms for Pattern Recognition, Pal, S.K. & P.P. Wang (Eds.), CRC Press, Boca Raton, FL., pp. 167-178.

Yuen, K.S. and Lo, E.K. (1994). A coarse-to-fine approach for circle detection, in Proc. Int. Symp. On Speech, Image Processing and Neural Networks, Hong Kong, HK. pp. 523-526.

Yuen, S. Y. & Ma, C. H. (2000). Genetic algorithm with competitive image labelling and least square. Pattern Recognition 33(12), pp. 1949-1966.

Pattern Recognition Techniques, Technology and Applications

Edited by Peng-Yeng Yin

A wealth of advanced pattern recognition algorithms are emerging from the interdiscipline between technologies of effective visual features and the human-brain cognition process. Effective visual features are made possible through the rapid developments in appropriate sensor equipments, novel filter designs, and viable information processing architectures. While the understanding of human-brain cognition process broadens the way in which the computer can perform pattern recognition tasks. The present book is intended to collect representative researches around the globe focusing on low-level vision, filter design, features and image descriptors, data mining and analysis, and biologically inspired algorithms. The 27 chapters coved in this book disclose recent advances and new ideas in promoting the techniques, technology and applications of pattern recognition.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

INTECH
open science | open minds