1998

Steady State Memetic Algorithm for Partial Shape Matching

Ender Ozcan
Syracuse University

Chilukuri K. Mohan
Syracuse University, ckmohan@syr.edu

Steady State Memetic Algorithm for Partial Shape Matching

Ender Ozcan and Chilukuri K. Mohan
eozcan/mohan@top.cis.syr.edu

2-120 Center for Science and Technology
Department of Electrical Engineering and Computer Science
Syracuse University, Syracuse, NY 13244-4100, U.S.A.

Abstract. Shape matching techniques are important in machine intelligence, especially in applications such as robotics. Currently, there are three major approaches to shape recognition: statistical, syntactic and neural approaches. This paper presents a fourth approach: evolutionary algorithms. A steady state memetic algorithm is shown to be successful in matching shapes even when they are partially obscured, and even in the presence of noise in the input image.

1 Introduction

Many intelligent applications, such as VLSI design and part inspection, use shape matching algorithms to identify the *model* shapes whose instances are present in an *input* shape . This task is computationally expensive when objects in the input image overlap, touch, or occlude one another. Existing approaches to solve the shape recognition problem ([3, 1, 4, 7, 8, 18, 17]) do not perform well in such situations. We show that appropriate evolutionary algorithms perform extraordinarily well for such problems.

In related work, Bala and Wechsler [2] apply genetic algorithms (GAs) to develop morphological operators for shape classification, not directly for shape matching. Di Ianni [6] uses genetic algorithms (GAs) for matching shapes but the results obtained were not encouraging, possibly because of using raw pixel arrays for the representation of shapes rather than image features.

In our earlier work, we obtained preliminary results on a small set of shapes, showing that GAs can be used for shape matching[11], and that they perform better than than simulated annealing[12]. This paper shows that even better results are obtained using a steady state algorithm: we present results for a large set of shapes, and with noisy perturbation of input shapes. Our algorithm gave robust matching results for the test shapes, providing translation, rotation and size independence. The new algorithm is introduced in Section 2. Section 3 describes experimental results, and the conclusions are presented in Section 4.

2 Steady State Memetic Algorithm for Shape Matching

2.1 Memetic Algorithms

Evolutionary algorithms are population-based search procedures drawing inspiration from the biological processes of genetics and evolution. Many researchers, such as Moscato [9] and Radcliffe *et al.* [13] have pointed out the usefulness of hill climbing and local search operators in evolutionary algorithms. Our research applies such a *Memetic Algorithm (MA)* that invokes hill climbing after generating offspring using evolutionary operators. This approach has already been applied successfully to several problems such as the Traveling Salesman Problem (*Moscato et al.* [9]).

Dawkins [5] coined the word *meme* to refer to a "contagious" piece of information. If a person is infected by a meme, that person processes the meme; understands it, adapts it and passes it on, whereas genes get inherited unchanged. This adaptation process resembles local refinement, hence the use of the term "memetic algorithm" for evolutionary algorithms that make extensive use of local search.

2.2 Features

The results of shape matching depend significantly on the features chosen to represent the shapes. For specialized problems such as face recognition, problem-specific features may lead to best results. For the general problem, however, we need a description of each shape in terms of generic features (such as line segments), that are also easy to extract using well-known algorithms. Furthermore, the choice of the representation is crucial if size-invariant and rotation-invariant shape recognition is desired.

To meet these requirements, we have chosen *attributed strings*[14, 15] to represent shapes. Each shape is considered to be a polygon, defined by a string of features $(x_1, x_2, ..., x_i, ..., x_n)$. Each feature $x_i = (l_i, \theta_i)$ is a set of attributes belonging to the i^{th} line segment on shape x: The length l_i of the corresponding line segment, and the relative angle (turn angle) θ_i it forms with the preceding line segment x_{i-1}. The choice of these attributes provides invariance under translation and rotation transformations. Normalization of lengths provides a reliable scale invariant measure (Figure 1), and the following functions are used by our algorithm:

$$l(l_i) = l_i / l_{i-1},\ \theta(\theta_i) = \theta_i.$$

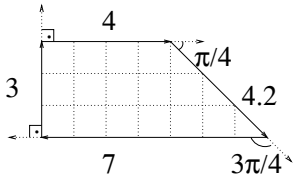

Fig. 1. After the normalization, the quadrilateral representation becomes $((1.33, \frac{\pi}{2}), (1.05, \frac{\pi}{4}), (1.67, \frac{3\pi}{4}), (0.43, \frac{\pi}{2}))$.

2.3 Representation

In the MA for shape matching, each individual maps each input shape feature to one of the model shape features. Each individual is represented as a list of lists in which each entry has two slots, one showing the matching model shape and the other showing its corresponding feature. We use the following notation:

- Input shape $I = (I_1, I_2, ..., I_p, ..., I_n)$. Size of input shape $|I| = n$, the number of features in I.
- The model shapes are $M_1, M_2, ..., M_j, ..., M_S$, where $M_j = (M_{j,1}, ..., M_{j,m_j})$. Size of the jth model shape $|M_j| = m_j$.
- Each individual $P = (P_1, P_2, ..., P_k, ..., P_n)$ corresponds to a mapping μ_P from input shape features to model shape features such that $P_k = \mu_P(I_k) = M_{j,i}$, where $1 \leq k \leq n$, $1 \leq j \leq S$, and $1 \leq i \leq m_j$.

The initial population is a set of randomly chosen individuals. The shape recognition problem now reduces to multiple substring matching. The search space is immense, since multiple partial instances of the same model shape may be present in the input shape

2.4 Fitness

The fitness of an individual describes how well each feature of the input shape matches with the model shape feature to which it is matched. Fitness also depends on the degree of consistency between model features to which neighboring shape features (I_j, I_{j+1}) are mapped.

Fitness is calculated by testing the compatibility of the input shape features and the corresponding model features to which an individual maps them. The difference (dissimilarity) between input shape feature I_k and model feature $f_k = \mu_P(I_k)$ is measured by means of a distance function $d(I_k, \mu_P(I_k))$, defined below.

$$d(I_k, \mu_P(I_k)) = \begin{cases} d_\theta(I_k, \mu_P(I_k)) + d_l(I_k, \mu_P(I_k)) & \text{if } \mu_P(I_{k-2}) = M_{i,j-2}, \\ & \mu_P(I_{k-1}) = M_{i,j-1}, \\ & \mu_P(I_k) = M_{i,j} \text{ and} \\ & \mu_P(I_{k+1}) = M_{i,j+1} \\ & \text{for some } i, j \\ d_\theta(I_k, \mu_P(I_k)) & \text{if } \mu_P(I_{k-1}) = M_{i,j-1} \text{ and} \\ & \mu_P(I_k) = M_{i,j} \\ & \text{for some } i, j \\ \infty & otherwise. \end{cases}$$

This measure has angle and length components. The first component, from angle measurements, is defined as follows:

$$d_\theta(I_k, \mu_P(I_k)) = c_\theta \text{ abs}(\theta(I_k) - \theta(\mu_P(I_k)))$$

The constant c_θ is chosen in our experiments so that differences up to $\pi/18$ are considered negligible. For angle information (d_θ) to be useful, it is necessary

for two successive input shape features to be mapped to two successive features of the same model shape. The length component of the distance measure compares the normalized feature lengths as follows:

$$d_l(I_k, \mu_P(I_k)) = |(l(I_k) - l(\mu_P(I_k)))/\max(l(I_k), l(\mu_P(I_k))|$$

This measure is invoked only if four successive input shape features to be mapped to four successive features of the same model shape. This is because normalized length information for the kth feature is reliable only if the $(k-1)$th feature' s length is known, and the latter information is unreliable if the $(k-2)$th input feature is not matched to the corresponding feature of the same model. Also notice that overlapping may occur at kth feature, mapping the $(k+1)$th input feature to a different model shape feature, making the length information (d_l) unreliable. For example, a rectangle and a hexagon are overlapped to form an

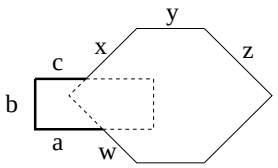

Fig. 2. Overlapped rectangle and a hexagon as an input shape.

input shape in Figure 2, whose features include w, x, y and z. For the most appropriate matching, only d_θ must be used to calculate the distance of features w and y, whereas both d_θ and d_l can be used for z. Neither d_θ nor d_l is reliable for x, since x and its preceding feature c belong to different model shapes.

The distance between two features is compared with a threshold value. If the distance is small, then the corresponding primitive is marked as matched. Otherwise, the features are not considered to have matched:

$$Matched(I_k, \mu_P(I_k)) = \begin{cases} 1 \text{ if } d(I_k, \mu_P(I_k)) < threshold \\ 0 \; otherwise \end{cases}$$

The fitness function penalizes the number of partially matched objects in the input shape to which features are mapped by an individual. Fitness is calculated using the following formula:

$$Fitness = -(\text{No. of partial shapes} + \text{No. of unmatched input shape features})$$

2.5 Selection and Crossover

Steady State evolutionary algorithms apply one crossover or recombination step at a time, then apply selection. One point crossover (1PTX) was used in all experiments reported in this paper. Experiments showed that 1PTX performs as well as two point crossover. Each application of 1PTX produced two offspring from two parents selected randomly for mating. The best two among these four (parents and offspring) were chosen to survive in the population. This process was iterated until either the population converges to a relatively unchanging state, or until computational limitations were exceeded.

2.6 Mutation

Definition: An input shape feature and a model shape feature are considered to be *Similar* when the error for each of the next two successive turn angles is less than $\pi/18$:

$$Similar(I_k, \mu_P(I_k)) = \begin{cases} 1 \text{ if } d_\theta(I_{k+1}, \mu_P(I_{k+1})) \text{ and } d_\theta(I_{k+2}, \mu_P(I_{k+2})) < \pi/18 \\ \quad \text{for some k} \\ 0 \; otherwise \end{cases}$$

Definition: *Similarity List* is an array of lists, where the size of the array indicates the size of the input shape and each list consists of features of the model shapes that are similar to each feature of the input shape.

We have used a mutation operator that replaces a subsequence with a fixed length of 3, from an individual by an equally long model shape subsequence. Each allele is mutated with a probability of $1/n$ The start feature for the subsequence is chosen randomly from the similarity list.

2.7 Hill Climbing

Hill climbing is applied, primarily to improve the mappings obtained at the borders between feature sequences mapped to different model shapes. Each hill climbing step attempts to improve the fitness of an individual by shifting the "intersection point" (between feature sequences mapped to different model shapes) first in one direction, then in the opposite direction, replacing the relevant component by the most appropriate feature from the model to which neighboring shape features are mapped. For instance (Figure 3), if $\mu_P(I_k) = M_{j,i}$ and

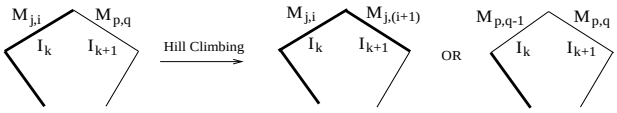

Fig. 3. Hill climbing in action.

$\mu_P(I_{k+1}) = M_{p,q}$, hill climbing first attempts to change $\mu_P(I_k)$ to $M_{p,q-1}$. If this attempt does not improve the fitness, hill climbing attempts to change $\mu_P(I_{k+1})$ to $M_{j,i+1}$. The change is not implemented if the fitness does not improve.

3 Experimental Results

The matching *threshold* is a nonlinear function of $t = \max(l(I_k), l(\mu_P(I_k)))$, allowing less error for high values, e.g. 0.2 for $t > 0.5$, and higher error for lower values, e.g. 0.9 for $t < 0.005$. In our MA experiments, we used a population size twice the number of features of the input shape. Each MA run was terminated when the correct solution was reached, or if the number of crossovers equals 500,000. Each test was repeated 100 times for all input shapes on a Sun workstation. 100 model shapes were used ($s0 - s99$), subset of which are shown in

Figure 4. The rest of the model shapes can be found in [12]. All of the input shapes($j0-j15$) were obtained by overlapping two or more model shapes (Figure 5). In the tables, "$fr.$" refers to the frequency of matching, i.e., how often the correct result was obtained.

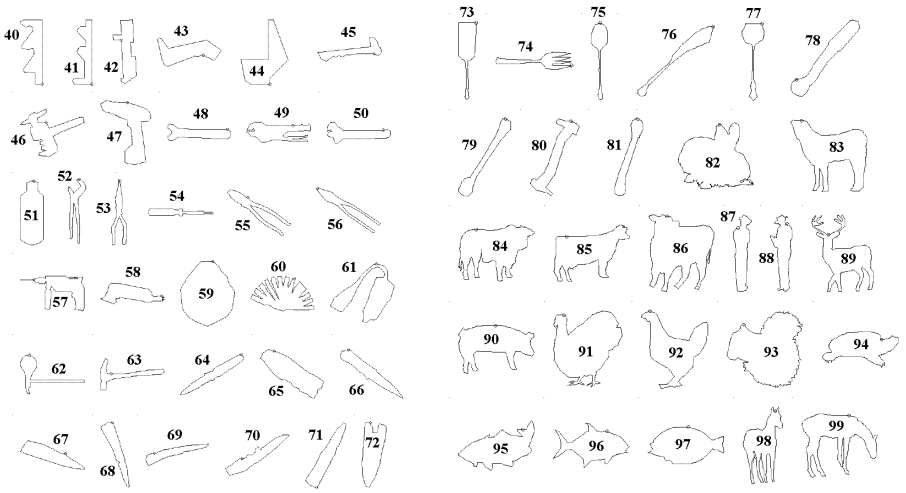

Fig. 4. Subset of normalized model shapes

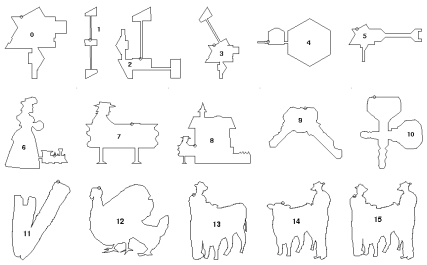

Fig. 5. Normalized input shapes

In the experiments reported here, we utilized steady state MA with the best mutation operator from [10]. A larger database of model shapes was used for steady state MA experiments. In the recent experiments, even though the total number of model features increase by a factor of 3.41, hence expanding the search space by a factor of 3.41^n, the number of states visited increases at most 50%.

Initial experiments were conducted using $j0-j10$ and $s0-s39$ to determine the efficacy of various operators. Performance became poorer if hill climbing was omitted, i.e., when a GA was used instead of MA (Table 1). The algorithm works best if all of the three operators (crossover, mutation and hill climbing) are used.

Table 1. Steady state MA test results using mutation and crossover (without hill climbing): Averages (μ), and standard deviations (σ) are based on over 100 experiments for each input shape in which the correct solution is found within 500,000 attempts of crossovers.

Shape label	$fr.$	No. of attempts		Time (sec.)	
		μ	σ	μ	σ
$j0$	0.46	110,614.44	130,619.29	61.79	72.85
$j1$	0.96	34,263.77	83,897.43	11.24	27.49
$j2$	0.00	500,000.00	0.00	293.05	4.13
$j3$	0.38	196,686.42	145,355.65	126.07	93.51
$j4$	0.73	37,469.96	76,595.18	17.66	36.49
$j5$	0.57	61,420.72	109,533.77	24.09	43.00
$j6$	0.83	70,718.06	97,432.07	130.24	181.15
$j7$	0.48	149558.77	154531.94	173.90	179.90
$j8$	0.14	431837.66	169001.91	606.30	237.35
$j9$	0.02	24,053.00	8,210.00	25.77	8.87
$j10$	0.30	165,728.73	132,817.56	242.95	195.21

Table 2 shows the results obtained by applying the GA to noisy versions of $j4$ against $s0 - s39$. Locations of 5%-25% of the input image vertices were randomly perturbed (higher noise levels may completely alter a shape). Our algorithm successfully found the correct (expected) matching results in almost all runs for all input shapes at different noise levels.

Table 2. Test results for input image $j4$: Success rates are averages of over 100 experiments, for different noise levels (fraction of features perturbed).

Noise Levels	0.05	0.10	0.15	0.20	0.25
Success Rate	1.00	1.00	1.00	0.93	1.00

Definition: *Visibility ratio* ($v.r.$) is the ratio of total number of input shape features and the total number of features of each composing shape.

Several experiments were conducted to observe the behavior of our algorithm as the number of occluded features increases, using input shapes shown in Figure 6. Shapes $s26$ and $s27$ are overlapped forming $r0 - r6$ and shapes $s21$, $s26$ and $s27$ are overlapped forming $r7 - r15$ with different visibility ratios. The last shapes are overlapped keys where the number of partial shapes increases with label number; the average visibility ratio is 0.89.

Visibility ratio test results (Table 3) show that decrease in the visibility ratio causes a decrease in the number of states visited. Meanwhile as the number of

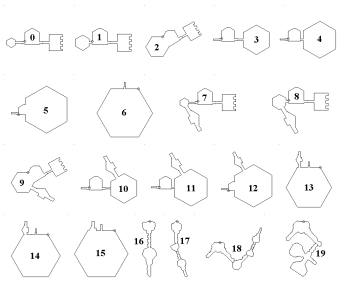

Fig. 6. Overlapped shapes $r0-r19$ with different visibility ratios.

partial shapes forming an input shape increases, the number of states visited increases as well. Still, MA found the correct matching result for visibility ratio tests in all runs. Experiments were performed using all operators and input and

Table 3. Steady state MA test results while varying visibility ratio

Shape label	$v.r.$	$fr.$	No. of Crossovers		No. of HC steps		Time (sec.)	
			μ	σ	μ	σ	μ	σ
$r0$	1.00	1.00	537.96	288.96	12.60	3.11	1.19	0.50
$r1$	0.91	1.00	783.08	566.33	10.61	3.15	1.45	0.87
$r2$	0.77	1.00	189.69	121.64	16.71	6.81	0.42	0.24
$r3$	0.64	1.00	175.70	106.82	11.69	3.55	0.35	0.19
$r4$	0.55	1.00	192.48	207.23	10.22	2.99	0.33	0.29
$r5$	0.43	1.00	136.29	93.08	9.34	5.28	0.21	0.13
$r6$	0.32	1.00	85.10	59.12	7.70	2.47	0.11	0.07
$r7$	1.00	1.00	881.83	387.98	11.98	2.36	2.43	0.90
$r8$	0.93	1.00	1,173.69	665.35	10.31	2.59	2.88	1.40
$r9$	0.83	1.00	400.97	136.11	14.64	2.97	1.10	0.34
$r10$	0.73	1.00	478.96	213.73	10.93	1.84	1.18	0.47
$r11$	0.66	1.00	378.14	246.28	11.50	2.22	0.89	0.48
$r12$	0.58	1.00	480.81	237.88	8.73	1.45	0.98	0.42
$r13$	0.49	1.00	238.61	131.05	10.40	2.54	0.46	0.22
$r14$	0.46	1.00	255.67	148.78	9.34	2.31	0.47	0.24
$r15$	0.39	1.00	243.68	132.39	8.19	1.88	0.41	0.20
$r16$	1.00	1.00	716.62	327.07	12.91	2.06	2.57	0.92
$r17$	0.81	1.00	1,617.98	377.26	11.07	1.35	7.60	1.55
$r18$	0.89	1.00	4,216.15	1,742.46	9.96	1.07	27.74	9.38
$r19$	0.88	1.00	4,539.13	1,903.92	8.29	0.74	26.94	9.62

model shape database, demonstrating the success of memetic algorithm (Table 4). The execution times were not affected by the enlargement of database. As results show, steady state MA found the correct matching for all input shapes (except $j2$ with $fr.$ 0.99).

Table 4. Steady state MA test results using all operators: Averages (μ), and standard deviations (σ) are based on over 100 experiments for each input shape in which the correct solution is found within 500,000 crossovers.

Shape label	$fr.$	No. of crossovers		No. of HC steps		Time (sec.)	
		μ	σ	μ	σ	μ	σ
$j0$	1.00	236.82	157.69	12.68	7.03	0.47	0.25
$j1$	1.00	55.45	32.26	16.61	7.93	0.09	0.04
$j2$	0.99	17,974.36	65,989.43	7.12	2.22	29.56	106.79
$j3$	1.00	634.21	430.21	10.18	2.54	1.40	0.80
$j4$	1.00	192.48	207.23	10.22	2.99	0.33	0.29
$j5$	1.00	129.96	112.26	14.18	4.42	0.27	0.19
$j6$	1.00	722.80	268.93	21.34	2.99	3.45	0.97
$j7$	1.00	3,860.49	3,332.53	8.38	4.10	8.27	6.64
$j8$	1.00	353.71	101.68	25.40	3.62	1.54	0.38
$j9$	1.00	501.71	271.52	12.00	1.95	1.70	0.74
$j10$	1.00	2,290.88	3,684.49	8.62	1.38	8.86	11.46
$j11$	1.00	8,188.33	7,998.94	6.55	1.13	29.47	26.15
$j12$	1.00	1,381.74	1,699.11	15.05	3.22	8.23	6.69
$j13$	1.00	2035.80	642.78	16.44	2.71	18.95	4.77
$j14$	1.00	4,346.20	3,751.22	12.85	2.54	37.73	23.5
$j15$	1.00	7,145.50	3,268.92	10.65	1.68	73.16	25.45

4 Conclusions

We have used a steady state memetic algorithm for shape matching, utilizing attributed string representations. Outline features of shapes are represented using attributed strings. Each line segment is associated with a feature of two attributes: length and angle. Relative lengths and angles are used for size invariance. The algorithms we propose have many advantages:

- They are much more computationally efficient than exhaustive search algorithms.
- They are space-efficient compared to neural networks, with much smaller memory requirements.
- The algorithms are fast, and explore a relatively small number of elements of the search space.
- The results obtained are better than a traditional GA without hill climbing.
- Operators used by MA help avoid getting stuck in locally optimal solutions.
- Steady state MA performs better than transgenerational MA, even under noise.
- Steady state MA performs better than simulated annealing.

If multiple instances of the same model shape are overlapped to form an input shape, or if two model shapes are almost identical, MA might get stuck in

a locally optimum solution as in the case of input shape $j2$. Overall, experimental results show that the memetic algorithm is successful for partial shape matching, even with a large database of shapes.

References

1. N. Ansari and E. J. Delp: *Partial shape recognition: A landmark-based approach* **IEEE Trans. on PAMI**, vol 12., pp. 489-497, 1990.
2. J. Bala, and H. Wechsler: *Shape Analysis Using Morphological Processing and Genetic Algorit ms*, **Int. Conf. on Tools for AI**, pp.130-137, 1991.
3. D. H. Ballard: *Generalizing the Hough transform to detect arbitrary shapes*, **Pattern Recognition**, vol. 13, pp. 111-122, 1981.
4. C. C. Chang, S. M. Hwang, and D. J. Buehrer: *A shape recognition scheme based on relative distances of feature points from the centroid*, **Pattern Recognition**, vol. 24, no. 11, pp. 1053-1063, 1991.
5. R. Dawkins: *The Selfish Gene*, Oxford University Press, 1976.
6. M. Di Ianni: *Simulated Annealing and Genetic Algorithms for Shape Detection*, **Polish J. Control and Cybernetics**, to appear.
7. M. W. Koch and R. L. Kashyap: *Using polygons to recognize and locate partially occluded objects*, **IEEE Trans. on PAMI**, vol. 9, pp. 483-494, July 1987.
8. N. M. Nasrabadi, and W. Li: *Object recognition by a Hopfield Neural Network*, **IEEE Trans. on Sys., Man, and Cybernetics**, vol. 21/6: 1503-1535, 1991.
9. P. Moscato, and M. G. Norman: *A Memetic Approach for the TSP, Implementation of a Computational Ecology for Combinatorial Optimization on Message-Passing Systems*, **Proc. of IC on Parallel Comp. and Transp. App.**, Ed. IOS Press, Amsterdam, pp. 187-194, 1992.
10. E. Ozcan, C. K. Mohan: *Partial Shape Matching Using Genetic Algorithms*, **Pattern Recognition Letters 18**:987-992, 1997.
11. E. Ozcan, and C. K. Mohan: *Shape Recognition Using Genetic Algorithms*, **Proc. of IEEE International Conference on Evolutionary Computation**, pp. 414-420, May 1996.
12. E. Ozcan, C. K. Mohan: *Simulated Annealing and Genetic Algorithms for Partial Shape Matching,* **Proc. of IEEE XI Intl. Symp. on Computer and Information Sciences**, pp. 173-181, Nov. 1996.
13. N. J. Radcliffe, and P. D. Surry: *Formal Memetic Algorithms*, **Evolutionary Computing: AISB Workshop** (Ed: T. Fogarty, Springer- erlag), 1994.
14. W. -H. Tsai and K. S. Fu: *Attributed grammar - a tool for combining syntactic and statistical approaches to pattern recognition*, **IEEE Tran. on SMC**, vol. 10, Dec 1980.
15. W. -H. Tsai and S.-S. Yu: *Attributed string matching with merging for shape recognition*, **IEEE Tran. on PAMI**, vol. 7, pp 453-462, July 1985.
16. J. L. Turney, T. N. Mudge and R. A. Voltz: *Recognizing partially occluded parts*, **IEEE Trans. on PAMI**, pp. 410-421, July 1985.
17. M. C. Yang, K. Mehrotra, C. K. Mohan, S. Ranka: *A New Algorithm for Shape Matching*, **Proc. Artificial Neural Networks in Eng. Conf.**, pp.523-528, Nov 1993.
18. S. S. Young, P. D. Scott and N. Nasrabadi: *Object Recognition Using Multi-layer Hopfield Neural Network*, **Conference on Computer Vision and Pattern Recognition**, pp. 417-422, June 21-23, 1994.

This article was processed using the LaTeX macro package with LLNCS style