Genetic Search for Object Identification

Sushil J. Louis George Bebis Satish Uthiram Yaakov Varol

Department of Computer Science
University of Nevada
Reno - 89557
sushil@cs.unr.edu

Abstract. We attack the problem of recognizing real, planar objects from two-dimensional, intensity images taken from arbitrary viewpoints using genetic algorithms. More specifically, we use genetic algorithms to search for a geometric mapping that brings subsets of points comprising the model and subsets of points comprising the scene into alignment. The genetic algorithm searches the image space and we compare different encodings and operators on a set of three increasingly complex scenes. Our preliminary results are promising with exact and near exact matches being found reliably and quickly.

1 Introduction

Recognizing objects from images is one of the most important and challenging problems in computer vision with many practical ramifications in modern manufacturing and autonomous navigation. During the last two decades, there have been a variety of proposed approaches to tackle this problem. The most successful approach has been **model-based** object recognition[2] (MBR) which relies on a rather constrained environment and the existence of a set of predefined model objects. Given an unknown scene, model-based recognition has two goals: (1) identify a set of features from the unknown scene which approximately match a set of features from a known view of a model object and (2) recover the geometric transformation that the model underwent (called pose recovery). In practical applications, recognition is considerably complicated due to (1) occlusions – where some model features have no corresponding image features and (2) unknown objects in the scene – where some image features have no corresponding model features. Moreover, the number of possible correspondences between model and image features grows exponentially with the number of features and there is usually little *a priori* knowledge to limit this number.

Genetic algorithms (GAs) are stochastic, parallel search algorithms based on the mechanics of natural selection, the process of evolution [5, 4]. GAs were designed to efficiently search large, nonlinear, poorly-understood search spaces where expert knowledge is scarce or difficult to encode and where traditional optimization techniques fail. As such, GAs appear well suited for searching the large, poorly-understood spaces that arise in object recognition. This paper uses a set of three increasingly complex recognition problems to test the efficacy of genetic algorithms at finding these correspondences and recognizing real, planar

objects from arbitrary viewpoints. Canonical genetic algorithms are usually good at improving average fitness over time We thus expected the GA to be able to perform a rough alignment of the model with the scene. In fact, our experimental results demonstrate that the genetic algorithm finds almost exact matches in the case of scenes with little occlusion while it finds near-exact matches when scenes with more occlusion are considered. Although near-exact matches might not solve the recognition problem completely, they are very useful in reducing the search space to a limited domain. Then, a local optimization technique can be used for finding an exact match. The preliminary stage of rough alignment may help prevent such local methods from reaching a local minimum instead of the global one.

The next section provides a short introduction to model-based object recognition and can be skipped by those already familiar with this area. Subsequently, Section 3 describes our experiments and the genetic algorithm encoding, operators, and parameters. Section 4 presents our results and analyses. The last section provides conclusions and directions for future work.

2 Model-Based Object Recognition

Recognition approaches usually operate in one of the following two spaces: the *image space* or the *transformation space*. Image-space based techniques proceed by first extracting a collection of image features. Then, a correspondence between these features and a set of previously extracted model features is hypothesized. The hypothesis determines the position and orientation of the model. Then the hypothesis is verified by projecting the model onto the scene using the computed transformation to find how well the model maps onto the scene. We are usually interested in finding the largest pairing of model and image features for which there exists a single geometric transformation, mapping each model feature to its corresponding image feature. Since there is often no *a priori* knowledge of which model features correspond to which scene features, recognition through matching can be too computationally expensive, even for relatively simple scenes. Assuming m model features and n image features, we have $p = m \times n$ possible pairs, which implies that searching for the largest pairing of model and image features is exponential in p.

Transformation-space based techniques deal with the space of possible geometric transformations (poses) between model and image features. The objective is to determine a transformation in the space of possible transformations which would maximize the number of model points aligned with image points. Again, a problem with this approach is the computational cost due to the large number of possible transformations. We do not consider transformation space techniques in this paper.

2.1 Affine Transformations

Under the assumption of weak perspective projection (i.e., orthographic projection plus scale), two different views of the same planar object are related

through an affine transformation [6, 1]. Specifically, let us assume that each model is characterized by a set of "interest" points $(p_1, p_2, p_3, \ldots, p_m)$, corresponding, for example, to curvature extrema or curvature zero-crossings of the model's boundary contour[8]. Let us now consider two images of the same planar object obtained from different viewpoints, and a point $p = (x, y)$ in the first image with its corresponding point $p' = (x', y')$ in the second image. Then, the coordinates of p' can be expressed in terms of the coordinates of p as follows:

$$p' = Ap + b$$

where A is a non-singular 2×2 matrix and b is a two-dimensional vector. Figure 1(b)-(d) show affine transformed views of the planar object shown in Figure 1(a). These views were generated by transforming the known view using the affine transformations shown in Table 1.

Table 1. Affine transformation parameters for Figure 1

Parameters	Fig 1(b)	Fig 1(c)	Fig 1(d)
a_{11}, a_{12}, b_1	0.992, 0.130, -0.073	-1.010, -0.079, 1.048	0.860, 0.501, -0.255
a_{21}, a_{22}, b_2	-0.379, -0.878, 1.186	0.835, -0.367, -.253	0.502, -0.945, 0.671

Assuming planar objects, we need at least three matches between model and scene points to derive a transformation that aligns the model with the scene. With M model points and S image points, the maximum number of possible alignments is of the order of $O(M^3S^3)$. Errors due to noise however, imply that we usually need more than three matches to accurately compute the affine transformation with least squares approaches used to reduce errors.

We used genetic algorithms to search the space of possible matches between model and scene points with different encodings and operators. We describe our encodings and methodology in the next section.

3 Methodology

For testing, we captured three scenes, Scene1, Scene2, and Scene3, with the object to be recognized being increasingly occluded in each of the three scenes. Scene2 and Scene3 are shown in Figure 2. Scene1 was the model itself and is shown in Figure 5. Our selection strategy was cross generational. Assuming a

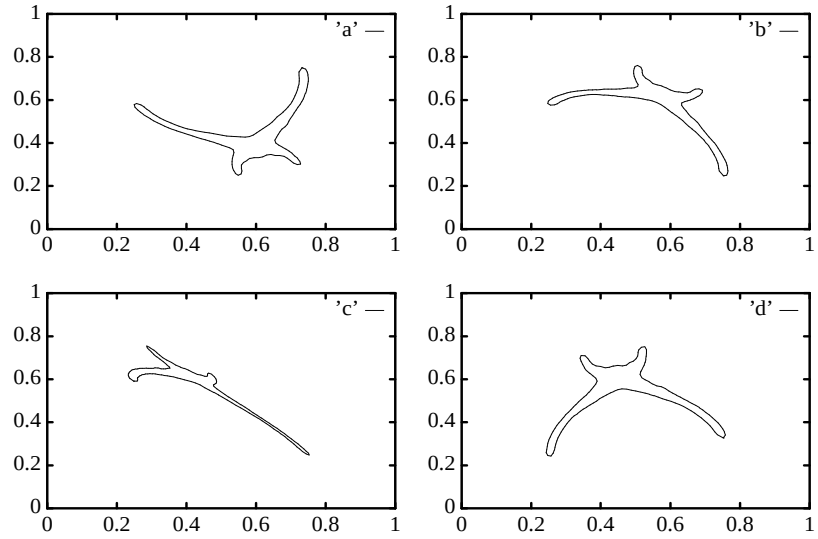

Fig. 1. (a) a known view of a planar object. (b)-(d) Affine transformed views of the same object generated through the affine transformations parametrized in Table 1. The two axes are the x and y coordinate axes

population of size N, the offspring double the size of the population and we select the best N individuals from the combined parent-offspring population for further processing[3]. This kind of selection does well with small populations and leads to quick convergence (sometimes prematurely). We also linearly scale fitnesses to try maintain a constant selection pressure.

3.1 Encoding

A simple encoding scheme where the identity of the points to be matched made up the alleles in the genotype gave good results on all three scenes. The parameters are provided in the next section. Our first encoding is shown in Figure 3. Since three pairs of points are needed to compute an affine transformation, the chromosome contains the binary encoded identities of the three pairs of points. The model or object to be recognized was defined by 19 points requiring $\lceil log_2 19 \rceil = 5$ bits per point while the scene had between 19 and 45 points and required either 5 or 6 bits per point. We did not check for repeated points and used simple two-point crossover and point mutation.

We also used a second encoding where the chromosome represents a vector whose indices corresponded to scene points and whose contents identify matching model points. Figure 4 pictorially describes this encoding and shows that the first scene point matches the fifth model point, the second scene point matches the 19^{th} model point, while the third scene point is not matched by any model point.

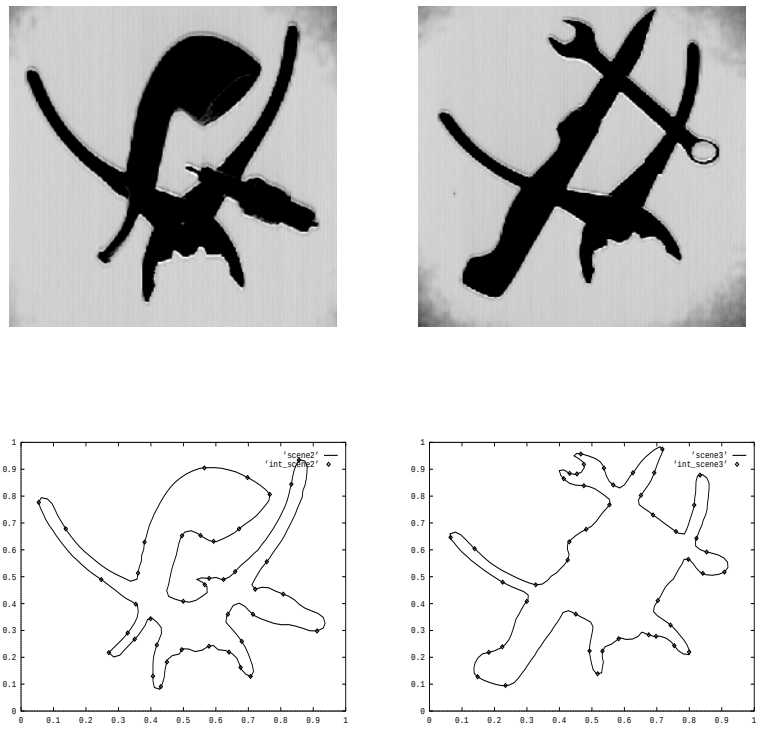

Fig. 2. Scenes used in our recognition experiments along with their extracted boundaries and interest points.

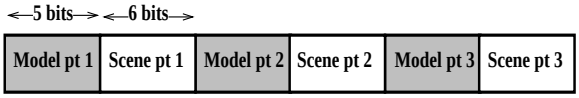

Fig. 3. Encoding 1 where the chromosome contains the binary encoded points to be matched between the model and scene

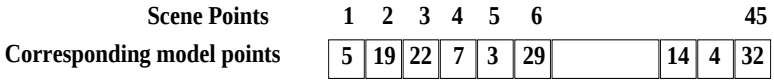

Fig. 4. Encoding 2 where the chromosome is a vector whose indices identify scene points and whose contents define matching model points.

Since the number of scene points (S) is usually greater than the number of model points, M, we use allele values greater than M to indicate non-matched scene points. Thus the third scene point does not match any model point (22 is greater than 19). This sequential representation is very similar to that for the traveling salesperson problem and allows us to use the PMX crossover operator and swap mutation [7]. Note that in this sequential encoding we only consider the first three matches in computing the transformation.

3.2 Fitness Evaluation

We evaluate fitness of individuals by computing the backprojection error (BE) between the model and scene. That is, to evaluate the goodness of the match specified by an individual, we first compute the parameters of the affine transformation which maps the encoded model points to the encoded scene points. Recall that we only consider the first three matches in computing the transformation. Alternatively, a transformation can also be computed using all point matches represented in the chromosome using a least-squares scheme, however, we have restricted ourselves to only three point matches since the search space is much smaller in this case.

After the transformation has been computed, we apply it to all the points of the model in order for us to backproject the model onto the scene. Finally, we compute the error, BE, between the backprojected model and the scene. To compute BE, for every model point we find the closest scene point and we compute the distance dj between these two points. Then, the backprojection error is

$$BE = \sum_{i=1}^{M} (dj)^2$$

Since we need to maximize fitness but minimize the error, our fitness function is

$$\text{Fitness} = 10000 - BE$$

which changes the minimization problem to a maximization problem for the GA.

4 Results

All genetic algorithm parameters were identical except for the population size and running time. We used a crossover probability of 0.95, a mutation probability of 0.05, and a scaling factor of 1.2. Population sizes were set to 100, 400, and 500 for scenes Scene1, Scene2, and Scene3 respectively. We ran the GA ten (10) times with different random seeds. Performance plots indicate that the GA very quickly gets close to the correct mapping and then spends most of its time making little progress.

Figure 5 shows a solution found by the GA for the simple test problem of trying to find an identity mapping (the scene and model are the same). Since the GA found the optimal mapping, the transformed model and the scene overly each

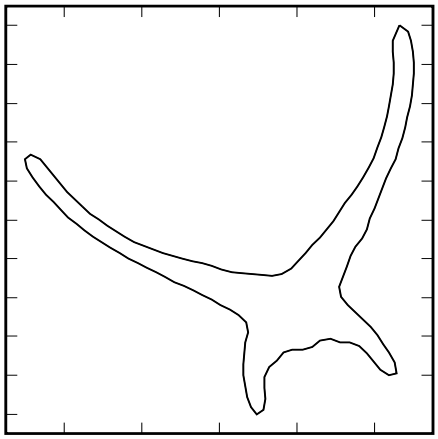

Fig. 5. Solution found by the GA on scene Scene1

other and present a single outline. Both encodings always found exact mappings.

Figure 6 (left) shows the best and worst solutions found by the GA on Scene2 for the simple binary encoding with two point crossover. Although an exact match was not found, the GA reliably finds acceptable matches. Figure 6 (right) displays best and worst solutions for the sequential encoding with PMX crossover and indicates little qualitative difference in results.

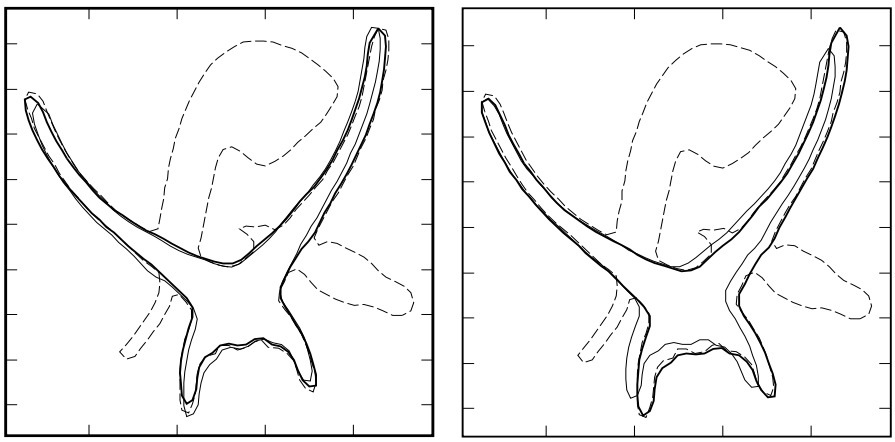

Fig. 6. Best and worst solutions found by the GA on Scene2 with a *simple binary* encoding and 2-point crossover (left) and with a *sequential* encoding and PMX (right).

Scene3 results are displayed in Figure 7. This scene is a little deceptive in

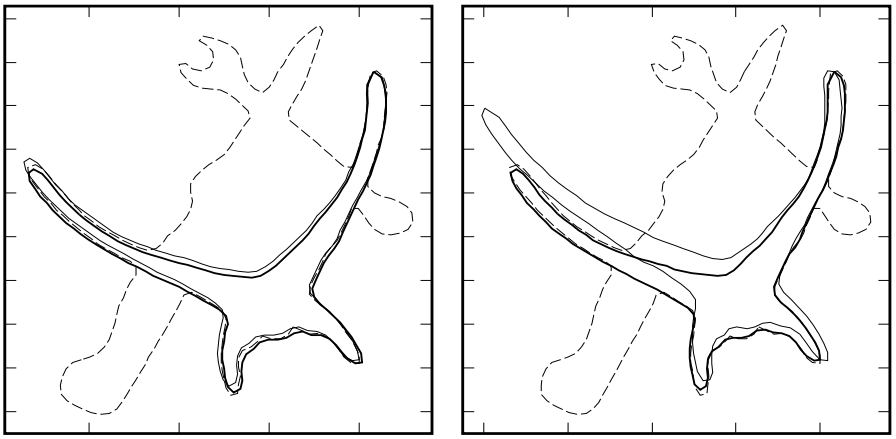

Fig. 7. Best and worst solutions found by the GA on Scene3 with a *simple binary* encoding and 2-point crossover (left) and with a *sequential* encoding and PMX (right).

that there is more that one possible place that matches the business end of the pliers. The GA with sequential encoding converged to a wrong mapping twice out of the ten runs. Figure 8 shows these solutions. The binary encoded GA always converged around the right mapping.

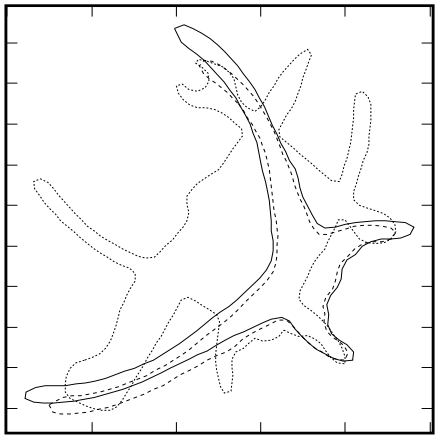

Fig. 8. Deceptive mapping found by the GA with sequential encoding on scene Scene3

Table 2 provides a summary of our results. The first column specifies the scene. The second, third, and fourth columns help describe the size of the test problems. These columns list (in order) the number of scene points (S), the number of possible combinations taking three scene points at a time (S_3), and

the size of the search space. The last two columns indicate GA effort in terms of the number of matches explored and the corresponding fraction in the searched space.

In our experiments the number of model points, $M = 19$. Thus the number of possible triplets, M_3

$$M_3 = \binom{19}{3} = 969$$

The order of points matters in computing the total number of possible matches between model and scene points and is thus given by the expression

$$\text{Total number of matches} = 3! \times M_3 \times S_3$$

Referring again to Table 2, the third column computes S_3, the fourth column lists the total number of possible matches, and the last two columns indicate the average number of matches the GA searched through ($GA_{matches}$), for each encoding. The number within parenthesis calculate (Total number of matches/$GA_{matches}$) which is the fraction of the space searched by the GA.

Table 2. Summary of results

Scene	Scene Points	$S_3 = \binom{S}{3}$	Problem Size $3!M_3S_3$	Binary Encoding $GA_{matches}$ (fraction)	Sequential Encoding $GA_{matches}$ (fraction)
Scene1	19	969	5,633,766	1800 (0.0003)	980 (0.0001)
Scene2	40	9880	57,442,320	47800 (0.0008)	45600 (0.0008)
Scene3	45	14190	82,500,660	133250 (0.0016)	71318 (0.0009)

Summarizing, although both encodings allow the GA to find good solutions reliably and quickly, the binary encoded GA seems to provide better quality. The sequential encoding takes less time but sometimes converges to the wrong solution.

5 Conclusions and Future Work

In this paper, we considered using genetic algorithms for recognizing real, planar, objects assuming that the viewpoint is arbitrary. Our experimental results

demonstrate that genetic algorithms are a viable tool for efficiently searching the space of possible matches between model and scene points.

One limitation of the current approach is that we do not consider geometric constraints. For example, interest points which correspond to curvature zero-crossings can be classified as negative-to-positive or positive-to-negative zero crossings depending on how the curvature changes sign. We can use this information to bias genetic algorithm search. Another limitation is that we only look at recognition problems that entail the recognition of a single model in a number of scenes. However, in practical model-based object recognition applications, more than one model may need to be recognized in scenes. In future research we plan to deal with the above issues, compare performance when searching in transformation space, and work on recognizing real three dimensional objects.

6 Acknowledgements

This material is based upon work supported by the National Science Foundation under Grant No. 9624130.

References

1. G. Bebis, M. Georgiopoulos, N. da Vitoria Lobo, and M. Shah. Learning affine transformations of the plane for model-based object recognition. In *Proceedings of the 13th International Conference on Pattern Recognition (ICPR-96)*, pages 60–64, 1996.
2. R. Chin and C. Dyer. Model-based recognition in robot vision. *Computing Surveys*, 18(1):67–108, 1986.
3. Larry J. Eshelman. The chc adaptive search algorithm: How to have safe search when engaging in nontraditional genetic recombination. In Gregory J. E. Rawlins, editor, *Proceedings of the Foundations of Genetic Algorithms Workshop - 1*, San Mateo, CA, 1990. Morgan Kauffman.
4. D. E. Goldberg. *Genetic Algorithm in Search, Optimization, and Machine Learning.* Addison-Wesley, Reading, MA, 1989.
5. J. Holland. *Adaptation In Natural and Artificial Systems.* The University of Michigan Press, Ann Arbor, 1975.
6. Y. Lamdan, J. Schwartz, and H. Wolfson. Affine invariant model-based object recognition. *IEEE Transactions on Robotics and Automation*, 6(5):578–589, 1990.
7. Sushil J. Louis and Gong Li. Augmenting genetic algorithms with memory to solve traveling salesman problems. In P. Wang, editor, *Proceedings of the Joint Conference on Information Sciences*, pages 108–111. Duke University Press, 1997.
8. F. Mokhtarian and A. Mackworth. A theory of multiscale, curvature-based shape representation for planar curves. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(8):789–805, 1992.