GENETIC ALGORITHMS FOR ADAPTIVE IMAGE SEGMENTATION

Bir Bhanu and Xing Wu
University of California at Riverside

Sungkee Lee
Kyungpook National University

ABSTRACT

Image segmentation is an extremely important and difficult low-level task. The difficulty arises when the segmentation performance needs to be adapted to the changes in image quality which is affected by variations in environmental conditions, imaging devices, time of day, etc. In this Chapter, we describe an adaptive image segmentation system that incorporates a feedback loop consisting of a machine learning subsystem, an image segmentation algorithm, and an evaluation component which determines segmentation quality. The machine learning component is based on genetic adaptation and uses separately a pure genetic algorithm and a combination of genetic algorithm and hill climbing. We present experimental results which demonstrate learning and scalability of the technique with the number of parameters to adapt the segmentation performance in outdoor color imagery.

1. INTRODUCTION

Image segmentation is an old and difficult problem. It refers to the grouping of parts of an image that have "similar" image characteristics. All subsequent interpretation tasks including object detection, feature extraction, object recognition, and classification rely heavily on the quality of the segmentation process. The difficulty arises when the segmentation performance needs to be adapted to the changes in image quality. Image quality is affected by variations in environmental conditions, imaging devices, time of day, etc. Despite the large number of segmentation techniques presently available [7, 13], no general methods have been found that perform adequately across a diverse set of imagery, i.e., no segmentation algorithm can automatically generate an "ideal" segmentation result in one pass (or in an open loop manner) over a range of scenarios encountered in practical applications. Any technique, no matter how "sophisticated" it may be, will eventually
yield poor performance if it cannot adapt to the variations in real-world scenes. The following are the key characteristics of the image segmentation problem:

- When presented with a new image, selecting the appropriate set of algorithm parameters is the key to effectively segmenting the image. Most segmentation techniques contain numerous control parameters which must be adjusted to obtain optimal performance, i.e., they are to be learned. The size of the parameter search space in these approaches can be prohibitively large, unless it is traversed in a highly efficient manner.

- The parameters within most segmentation algorithms typically interact in a complex, non-linear fashion, which makes it difficult or impossible to model the parameters' behavior in an algorithmic or rule-based fashion.

- The variations between images cause changes in the segmentation results, the objective function that represents segmentation quality varies from image to image. The search technique used to optimize the objective function must be able to adapt to these variations.

- The definition of the objective function itself can be a subject of debate because there are no universally accepted measures of image segmentation quality.

Hence, a need exists to apply an adaptive technique that can efficiently search the complex space of plausible parameter combinations and locate the values which yield optimal results. The approach should not be dependent on the particular application domain nor should it have to rely on detailed knowledge pertinent to the selected segmentation algorithm. Genetic algorithms (GA), which are designed to efficiently locate an approximate global maximum in a search space, have the attributes described above and show great promise in solving the parameter selection problem encountered in the image segmentation task.

The next section of this Chapter argues about the genetic algorithms as the appropriate optimization technique for the segmentation problem. Section 3 describes the adaptive image segmentation algorithm. Section 4 presents the experimental results on a sequence of outdoor images. Section 5 presents the adaptive segmentation results when we scale the number of parameters in a scheme that uses genetic algorithms and hill climbing. Finally, Section 6 provides the conclusions of this Chapter.

2. IMAGE SEGMENTATION AS AN OPTIMIZATION PROBLEM

Fig. 1 provides an example of an objective function that is typical for the image segmentation process. The figure depicts an application in which only two segmentation parameters (maxmin and abscoore) are being varied, and the corresponding segmentation quality obtained for any pair of algorithm parameters.
Because the algorithm parameters interact in complex ways, the objective function is multimodal and presents problems for many commonly used optimization techniques. Further, since the surface is derived from an analysis of real-world imagery, it may be discontinuous, may contain significant amounts of noise, and cannot be described in closed form. The derivation of this surface will be described in Section 3, where we discuss the segmentation evaluation process.

![Segmentation Quality Surface](image)

**Figure 1:** Segmentation quality surface.

The conclusion drawn from an analysis of many segmentation quality surfaces that we have examined is that we must utilize a highly effective search strategy which can withstand the breadth of performance requirements necessary for the image segmentation task.

Various commonly used search techniques for functional optimization exist. These include (a) exhaustive techniques (random walk, depth first, breadth first, enumerative), (b) calculus-based techniques (gradient methods, solving systems of equations), (c) partial knowledge techniques (hill climbing, beam search, best first, branch and bound, dynamic programming, A*), and (d) knowledge-based techniques (production rule systems, heuristic methods). The limitations of these methods are given in [3, 15, 24]. There are other search techniques such as genetic algorithms, simulated annealing and hybrid or integrated methods [3]. To address the characteristic of image segmentation problem as discussed earlier, we have selected genetic algorithms and hybrid methods for adaptive image segmentation.

2.1. **Genetic Algorithms**

Genetic algorithms were pioneered at the University of Michigan by John Holland and his associates [6, 10, 14]. The term genetic algorithm is derived from the fact that its operations are loosely based on the mechanics of genetic adaptation in biological systems. Genetic algorithms can be briefly characterized by three main concepts: a Darwinian notion of fitness or strength which determines
an individual's likelihood of affecting future generations through reproduction; a reproduction operation which produces new individuals by combining selected members of the existing population; and genetic operators which create new offspring based on the structure of their parents.

A genetic algorithm maintains a constant-sized population of candidate solutions, known as individuals. The initial seed population from which the genetic process begins can be chosen randomly or on the basis of heuristics, if available for a given application. At each iteration, known as a generation, each individual is evaluated and recombined with others on the basis of its overall quality or fitness. The expected number of times an individual is selected for recombination is proportional to its fitness relative to the rest of the population. Intuitively, the high strength individuals selected for reproduction can be viewed as providers of "building blocks" from which new, higher strength offspring can be constructed. New individuals are created using two main genetic recombination operators known as crossover and mutation. Crossover operates by selecting a random location in the genetic string of the parents (crossover point) and concatenating the initial segment of one parent with the final segment of the second parent to create a new child. A second child is simultaneously generated using the remaining segments of the two parents. The string segments provided by each parent are the building blocks of the genetic algorithm. Mutation provides for occasional disturbances in the crossover operation by inverting one or more genetic elements during reproduction. This operation insures diversity in the genetic strings over long periods of time and prevents stagnation in the convergence of the optimization technique.

The individuals in the population are typically represented using a binary notation to promote efficiency and application independence of the genetic operations. Holland [14] provides evidence that a binary coding of the genetic information may be the optimal representation. Other characteristics of the genetic operators remain implementation dependent, such as whether both of the new structures obtained from crossover are retained, whether the parents themselves survive, and which other knowledge structures are replaced if the population size is to remain constant. In addition, issues such as the size of the population, crossover rate, mutation rate, generation gap, and selection strategy have been shown to affect the efficiency with which a genetic algorithm operates [12].

The inherent power of a genetic algorithm lies in its ability to exploit, in a highly efficient manner, information about a large number of individuals. By allocating more reproductive occurrences to above average individuals, the overall net effect is an upward shift in the population's average fitness. Since the overall average moves upward over time, the genetic algorithm is a "global force" which shifts attention to productive regions (groups of highly fit individuals) in the search space. However, since the population is distributed throughout the search space, genetic algorithms effectively minimize the problem of converging to local maxima.

To date, genetic algorithms have been applied to a wide diversity of problems.
They have been used in combinatorial optimization [16], gas pipeline operations [9, 17] and machine learning [15]. With regards to computer vision applications, Gadiyanda et. al [18] have used genetic algorithms for image registration, Gillies [8], and Roth and Levine [22] for feature extraction, and Ravichandran [21] for object recognition.

3. GENETIC LEARNING FOR ADAPTIVE IMAGE SEGMENTATION

Genetic algorithms can be used in several different ways to provide an adaptive behavior within a computer vision system [3]. The simplest approach is to allow the genetic system to modify a set of control parameters that affect the output of an existing computer vision program. By monitoring the quality of the resulting program output, the genetic system can dynamically change the parameters to achieve the best performance. In this paper, we have adopted this strategy for adaptive image segmentation.

The block diagram of our approach is shown in Fig. 2. After acquiring an input image, the system analyzes the image characteristics and passes this information, in conjunction with the observed external variables, to the genetic learning component. Using this data, the genetic learning system selects an appropriate parameter combination, which is passed to the image segmentation process. After the image has been segmented, the results are evaluated. If the quality of segmentation ("fitness") is acceptable, an update to long-term population is made. If the quality is unacceptable, the process of new parameter selection, segmentation and evaluation continues until a segmentation result of acceptable quality is produced.

The termination criteria are satisfied.

3.1. IMAGE CHARACTERISTICS

A set of characteristics of the image is obtained by computing specific properties of the image itself as well as by observing the environmental conditions in which the image was acquired. Each type of information encapsulates knowledge that can be used to determine a set of appropriate starting points for the parameter adaptation process. For the experiments described here, we compute twelve first order properties for each color component (red, green, and blue) of the image. These features include mean, variance, skewness, kurtosis, energy, entropy, x intensity centroid, y intensity centroid, maximum peak height, maximum peak location, interval set score, and interval set size [17, 23]. The last two features measure histogram properties used directly by the PHOENIX segmentation algorithm used in this research and provide useful image similarity information. Since we use a gray scale image to compute edge information and object contrast during the evaluation process, we also compute the twelve features for the Y (luminance component) image as well. Combining the image characteristic data from these four components yields a list of 48 elements. In addition, we utilize two external variables, time of day and weather conditions to characterize each image. The
external variables are represented symbolically in the list structure (e.g., time = 9am, 10am, etc. and weather conditions = sunny, cloudy, hazy, etc.). The distances between these values are computed symbolically when measuring image similarity. The two external variables are added to the list to create an image characteristic list of 50 elements. The representation of an individual knowledge structure of the genetic population is shown in Fig. 3, where I is the number of image statistics, J is the number of external variables and N is the number of segmentation parameters.

3.2. GENETIC LEARNING SYSTEM

Once the image statistics and external variables have been obtained, the genetic learning component uses this information to select an initial set of segmentation algorithm parameters. A knowledge-based system is used to represent the image characteristics and the associated segmentation parameters. The image statistics and external variables shown in Fig. 3 form the condition portion of the knowledge structure, C1 through Cj+k, while the segmentation parameters indicate the actions, A1 through AN, of the knowledge structure. The fitness, W, which ranges in value from 0.0 to 1.0, measures the quality of the segmentation parameter set. Note that only the fitness value and the action portion of the knowledge structure are subject to genetic adaptation; the conditions remain fixed for the life of the knowledge structure.

When a new image is provided to the genetic learning system, the process begins by comparing the image characteristics of the new image (Fig. 2) with
Figure 3: Representation of a knowledge structure used by the genetic learning system. The image characteristics (image statistics and external variables), segmentation parameters, and the image quality or fitness of the parameter set are stored in each structure.

the knowledge structures in the long-term population (also called global population, Fig. 3). The long-term population represents the accumulated knowledge of the adaptive system obtained through previous segmentation experience. The algorithm computes a ranked list of individuals in the population that have characteristics similar to the new image. Ranking is based on the normalized Euclidean distance between the image characteristic values as well as the fitness of the knowledge structure. The normalized distance between images A and B is computed using

\[
dist_{AB} = \sum_{i=1}^{I+J} W_i \left| \frac{C_{iA} - C_{iMIN}}{C_{iMAX} - C_{iMIN}} - \frac{C_{iB} - C_{iMIN}}{C_{iMAX} - C_{iMIN}} \right|
\]

where \(C_{iMIN}\) is the minimum value of the \(i\)th numeric or symbolic feature in the global population, \(C_{iMAX}\) is the maximum value of the \(i\)th feature in the global population, and \(W_i\) is the weight attached to the \(i\)th feature. For the results presented in this paper, the ranges are normalized and the \(W_i\) values have been set to 1 so that each feature contributes equally to the distance calculation.

When the distance between an image and several members of the global population are the same (e.g., if a previous image contributed multiple individuals to the global population), fitness values are used to select the best individuals from the population. Temporary copies of the highest ranked individuals are used to create the initial or seed population for the new image.

Once the initial or seed population is available, the genetic adaptation cycle begins. (The seed population is the same as the initial population, when the genetic algorithm begins its search operation.) The segmentation parameter set in each member of the seed population is used to process the image. The quality of the segmented results for each parameter set is then evaluated. If the maximum segmentation quality for the current population is above a predefined threshold of acceptance or other stopping criteria are satisfied, the cycle terminates and the high quality members of the current image population are used to update the
global population. Less fit members of the global population are discarded in favor of higher strength individuals obtained from processing the current image. In this manner, the system is able to extend the knowledge of the adaptive segmentation system by incorporating new experience into the knowledge database.

Alternatively, if after segmenting and evaluating the performance of the current or local (also called short-term) population, the system has not achieved acceptable segmentation quality and any other termination criteria are not satisfied, the genetic recombination operators are applied to the members of the current population. The crossover and mutation operators are applied to the high strength individuals in the population, creating a new set of offspring which will theoretically yield better performance [3, 14]. The new population is supplied back to the image segmentation process, where the cycle begins again. Each pass through the loop (segmentation-evaluation-recombination) is known as a generation. The cycle shown continues until the maximum fitness achieved at the end of a generation exceeds some threshold or other termination criteria are satisfied. The global population is updated and the system is then ready to process a new image.

3.3. Segmentation Algorithm

Since we are working with color imagery in our experiments, we have selected the PHOENIX segmentation algorithm developed at Carnegie-Mellon University and SRI International [17, 19, 23]. The PHOENIX algorithm is a recursive region splitting technique. An input image typically has red, green, and blue image planes, although monochrome images, texture planes, and other pixel-oriented data may also be used. Each of the data planes is called a feature or feature plane. The algorithm recursively splits nonuniform regions in the image into smaller subregions on the basis of a peak/valley analysis of the histograms of the red, green, and blue image components simultaneously. Segmentation begins with the entire image, considered to be a single region, based on histogram and spatial analyses. If the initial segmentation fails, the program terminates; otherwise, the program fetches each of the new regions in turn and attempts to segment them. This process terminates when the recursive segmentation reaches a predefined depth, or when all the regions have been segmented as finely as various user-specified parameters permit.

PHOENIX contains seventeen different control parameters [17], fourteen of which are used to control the thresholds and termination conditions of the algorithm. There are about $10^{40}$ conceivable parameter combinations using these fourteen values. For the outdoor image sequence that we have used, these parameters can be divided into three groups according to their effect on segmentation results.
Group I: Essential PHOENIX Parameters.

<table>
<thead>
<tr>
<th>Parameter (default)</th>
<th>Description</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hsmooth (9)</td>
<td>The width of the averaging window used to smooth each feature histogram.</td>
<td>1 – 100</td>
</tr>
<tr>
<td>Maxmin (160)</td>
<td>The minimum acceptable ratio of apex height to higher shoulder.</td>
<td>100 – 10^4</td>
</tr>
</tbody>
</table>

Group II: Important PHOENIX Parameters.

<table>
<thead>
<tr>
<th>Parameter (default)</th>
<th>Description</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Absscore (70)</td>
<td>The lowest interval set score that will be passed to the threshold phase.</td>
<td>0 – 1000</td>
</tr>
<tr>
<td>Splitmin (4)</td>
<td>Direct manipulation of the segmentation queue, for which fetched regions are to be segmented further</td>
<td>1 – 200</td>
</tr>
<tr>
<td>Noise (10)</td>
<td>The size of the largest area that is to be considered noise</td>
<td>0 – 10^4</td>
</tr>
<tr>
<td>Height (20)</td>
<td>The minimum acceptable apex height as a percentage of the second highest apex</td>
<td>0 – 100</td>
</tr>
</tbody>
</table>

Group III: Less Important PHOENIX Parameters

The rest of the parameters have relatively much less influence on the segmentation result.

To minimize the problem complexity, four parameters have been selected for GA to search for the combination that gives best segmentation result using PHOENIX. Thirty two values are sampled for each of these four parameters. This results in a search space whose size is about one million. The parameters are shown in Table 1, together with the formula by which they are sampled, and the associated test range for each. In Section 4, we will present results using the first two parameters (hsmooth and maxmin). In Section 5, we show scaling results when we adapt all the four parameters.

3.4. Segmentation Evaluation

After the image segmentation process has been completed by the PHOENIX algorithm, we must measure the overall quality of the segmented image. There are a large number of segmentation quality measures [2] that have been developed in the past, although none has achieved widespread acceptance as a universal measure of segmentation quality. In order to overcome the drawbacks of using only a single quality measure, we have incorporated an evaluation technique that
Table 1: PHOENIX parameters used for adaptive image segmentation.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Sampling Formula</th>
<th>Test Range</th>
</tr>
</thead>
</table>
| $H_{smooth}$:  
$h_{index} \in [0 : 31]$ | $h_{smooth} = 1 + 2 \cdot h_{index}$ | 1 — 63 |
| $Maximin$:  
$m_{minindex} \in [0 : 31]$ | $ep = \log(100) + 0.05 \cdot m_{minindex}$  
$ma_{zmin} = \exp(ep) + 0.5$ | 100 — 471 |
| $Splitmin$:  
$s_{minindex} \in [0 : 31]$ | $splitmin = 9 + 2 \cdot s_{minindex}$ | 9 — 71 |
| $Height$:  
$h_{index} \in [0 : 31]$ | $height = 4 + 2 \cdot h_{index}$ | 4 — 66 |

uses five different quality measures to determine the overall fitness for a particular parameter set. In the following, boundary pixels refer to the pixels along the borders of the segmented regions, while the edges obtained after applying an edge operator are called edge pixels. The five segmentation quality measures that we have selected are,

1. **Edge-Border Coincidence**: Measures the overlap of the region borders in the image acquired from the segmentation algorithm relative to the edges found using an edge operator. In this quality measure, we use the Sobel operator to compute the necessary edge information. The original, unthinned Sobel edge image is used to maximize overlap between the segmented image and the edge image. Edge-border coincidence is defined as follows (refer to Fig. 4(a)).

Let $E$ be the set of pixels extracted by the edge operator after thresholding and $S$ be the set of pixels found on the region boundaries obtained from the segmentation algorithm:

$$E = \{p_1, p_2, \cdots, p_E\} = \{(x_{p1}, y_{p1}), (x_{p2}, y_{p2}), \cdots, (x_{pE}, y_{pE})\} \quad \text{and}$$
$$S = \{q_1, q_2, \cdots, q_S\} = \{(x_{q1}, y_{q1}), (x_{q2}, y_{q2}), \cdots, (x_{qS}, y_{qS})\}, \quad \text{then}$$

$$\text{Edge-border Coincidence} = \frac{n(E \cap S)}{n(E)}$$

$$E \cap S = \{(x_k, y_k), k = 1, \cdots, m, \text{where} (x_y, y_k) \in E \text{ and } S\}$$

and

$n(A) = \text{the number of elements in set } A$.

2. **Boundary Consistency**: Similar to edge-border coincidence, except that region borders which do not exactly overlap edges can be matched with each
Figure 4: Illustration for the quality measures used in the adaptive image segmentation system. (a) Edge-border coincidence, (b) Boundary consistency, (c) Pixel classification, (d) Object overlap. Object contrast is defined by using the symbols shown in the center figure in (a) and the left most figure in (c).
other. In addition, region borders which do not match with any edges are used to penalize the segmentation quality. The Roberts edge operator is used to obtain the required edge information. As with the edge-border coincidence measure, the Roberts edge image is not thinned to maximize the overlap between images. Boundary consistency is computed in the following manner (see Fig. 4(b)).

The first step is to find neighboring pixel pairs in the region boundary and edge results. For each pixel in the segmented image region boundary results, $S$, a neighboring pixel in the edge image, $E$, that is within a distance of $d_{\text{max}}$ is sought. A reward for locating a neighbor of the $i$th boundary pixel is computed using

$$R_i = \frac{d_{\text{max}} - d_i}{d_{\text{max}}}$$

where $d_{\text{max}} = 10$, and $d_i =$ the distance to the nearest edge pixel.

Thus, if the pixels had overlapped, $R_i = (10 - 0)/10 = 1$. Pixels that do not directly overlap contribute a reward value that is inversely related to their distance from each other. As matching pairs of pixels are identified, they are removed from the region boundary and edge images ($S$ and $E$). The total reward for all matching pixel pairs is obtained using

$$R_{\text{TOTAL}} = \sum_i R_i$$

Once all neighboring pixel pairs have been removed from $E$ and $S$, the remaining (i.e., non-overlapping and non-neighboring) pixels correspond to the difference between the two images. The average number of these pixels is used to compute a penalty

$$P = \frac{n(\text{all remaining pixels in } E \text{ and } S)}{2}.$$

Finally, since the value of boundary discrepancy must be positive, we define an intermediate value, $M$, as $M = (R_{\text{TOTAL}} - P)/n(E)$, then

$$\text{Boundary Consistency} = M, \text{ if } M \geq 0, \text{ and zero otherwise.}$$

3. **Pixel Classification**: This measure is based on the number of object pixels classified as background pixels and the number of background pixels classified as object pixels. Let $G$ be the set of object pixels in the groundtruth image and $R$ be the set of object pixels in the segmented image (see Fig. 4(c)). Formally, we have

$$G = \{p_1, p_2, \cdots, p_A\} = \{(x_{p1}, y_{p1}), (x_{p2}, y_{p2}), \cdots, (x_{pA}, y_{pA})\} \quad \text{and}$$

$$R = \{q_1, q_2, \cdots, q_B\} = \{(x_{q1}, y_{q1}), (x_{q2}, y_{q2}), \cdots, (x_{qB}, y_{qB})\}.$$
Since pixel classification must be positive, we define the intermediate value $N$ as follows

$$ N = 1 - \left[ \frac{(n(G) - n(G \cap R)) + (n(R) - n(G \cap R))}{n(G)} \right] $$

where $G \cap R = \{(x_k, y_k), k = 1, \ldots, m, \quad \text{where} \ (x_k, y_k) \in G \text{ and } R\}$

Using the value of $N$, pixel classification can then be computed as

Pixel Classification $= N$, if $N \geq 0$, and zero otherwise.

4. **Object Overlap**: Measures the area of intersection between the object region in the groundtruth image and the segmented image, divided by the object region. As defined in the pixel classification quality measure, let $G$ be the set of object pixels in the groundtruth image and $R$ be the set of object pixels in the segmented image (Fig. 4(d)). Object overlap can be computed as

$$ \text{Object Overlap} = \frac{n(G \cap R)}{n(G)} $$

where $G \cap R = \{(x_k, y_k), k = 1, \ldots, m, \quad \text{where} \ (x_k, y_k) \in G \text{ and } R\}$

5. **Object Contrast**: Measures the contrast between the object and the background in the segmented image, relative to the object contrast in the groundtruth image. Let $G$ be the set of object pixels in the groundtruth image and $R$ be the set of object pixels in the segmented image, as shown in Fig. 4(a). In addition, we define a bounding box ($X$ and $Y$) for each object region in these images. These boxes are obtained by enlarging the size of the minimum bounding rectangle for each object ($G$ and $R$) by 5 pixels on each side. The pixels in regions $X$ and $Y$ include all pixels inside these enlarged boxes with the exception of the pixels inside the $G$ and $R$ object regions. We compute the average intensity for each of the four regions ($G$, $R$, $X$, and $Y$) using the equation $I_L = \sum_{j=1}^{L_{max}} I(j)/L_{max}$, where $I(j)$ is the intensity of the $j$th pixel in some region $L$ and $L_{max}$ is the total number of pixels in region $L$. The contrast of the object in the groundtruth image, $C_{GT}$, and the contrast of the object in the segmented image, $C_{SI}$, can be computed using

$$ C_{GT} = \left| \frac{I_G - I_X}{I_G} \right|, \quad C_{SI} = \left| \frac{I_R - I_Y}{I_R} \right|.$$

The object contrast quality measure is then computed as

$$ \text{Object Contrast} = \begin{cases} \frac{C_{SI}}{C_{GT}}, & \text{if } C_{GT} \geq C_{SI} \\ \frac{C_{GT}}{C_{SI}}, & \text{if } C_{GT} < C_{SI}. \end{cases} $$
The maximum and minimum values for each of the five segmentation quality measures are 1.0 and 0.0, respectively. The first two quality measures are global measures since they evaluate the segmentation quality of the whole image with respect to edge information. Conversely, the last three quality measures are local measures since they only evaluate the segmentation quality for the object regions of interest in the image. When an object is broken up into smaller parts during the segmentation process, only the largest region which overlaps the actual object in the image is used in computing the local quality measures. In the experiments described in this chapter, we combine the five quality measures into a single, scalar measure of segmentation quality using a weighted sum approach. Each of the five measures is given equal weighting in the weighted sum. Elsewhere we have investigated a more complex vector evaluation approach that provides multidimensional feedback on segmentation quality [3, 4].

4. SEGMENTATION RESULTS

4.1. SEGMENTATION USING GENETIC ALGORITHM

The adaptive image segmentation consists of the following steps:

1. Compute the image statistics.
2. Generate an initial population.
3. Segment the image using initial parameters.
4. Compute the segmentation quality measures.
5. WHILE not <stopping conditions> DO
   5a. select individuals using the reproduction operator
   5b. generate new population using the crossover and mutation operators
   5c. segment the image using new parameters
   5d. compute the segmentation quality measures
5. END
6. Update the knowledge base using the new knowledge structures.

We have tested the performance of the adaptive image segmentation system on a time sequence of outdoor images. The outdoor image database consisted of twenty frames captured using a JVC GXF700U color video camera. The images were collected approximately every 15 minutes over a 4 hour period. A representative subset of these images is shown in Fig. 5. The original images were digitized to be 480 x 480 pixels in size but were subsequently subsampled (average of 4 x 4 pixel neighborhood) to produce 120 x 120 pixel images for the segmentation experiments. Weather conditions in our image database varied from bright sun to overcast skies. The changing environmental conditions caused by movement of the sun also created varying object highlights, moving shadows, and many subtle contrast changes between the objects in the image. Also, the colors of most objects in the image are subdued. The auto-iris mechanism in the camera was functioning, which causes a similar appearance of the background foliage.
throughout the image sequence. Even with the auto-iris capability built into the
camera, there was still a wide variation in image characteristics across the image
sequence. This variation required the use of an adaptive segmentation approach
to compensate for these changes.

Figure 5: Sample outdoor images used for adaptive segmentation experiments.

The car in the image is the object of interest for the pixel classification, object
overlap, and object contrast segmentation quality measures. The groundtruth
image for the car was obtained by manual segmentation of Frame 1 only for
the image sequence. The Sobel and Roberts edge operator results, which are
used in the computation of the edge-border coincidence and boundary consistency
measures respectively, are obtained from the gray scale image (Y component of
the YIQ image set) for each frame [5]. For the results presented in this section, the
maxmin and hsmooth parameters of the PHOENIX algorithm were used to control
the segmentation quality and the segmentation quality surfaces were defined for
preselected ranges of these two parameters as shown in Table 1. All the parameters
that were not optimized were set at the default PHOENIX parameter values.
These parameters remain fixed throughout all the experiments. By selecting 32
discrete values (5 bits of resolution) for each of these parameter ranges, the search
space contained 1024 different parameter combinations. Fig. 6 presents the five
individual segmentation quality surfaces and the combined surface for Frame 1
of the database. Notice that the surfaces are complex and hence, would pose
significant problems to traditional optimization techniques.

The genetic component used a local or seed population size of 10, long-term
population size of 100, a crossover rate of 0.8, and mutation rate of 0.01. A
crossover rate of 0.8 indicates that, on average, 8 out of 10 members of the popula-
tion will be selected for recombination during each generation. The mutation rate
of 0.01 implies that on average, 1 out of 100 bits is mutated during the crossover
operation to insure diversity in the local population. The stopping criterion for
the genetic process contains three tests. First, since the global maximum for each
segmentation quality surface was known a priori (the entire surface was precom-
Figure 6: Segmentation quality surfaces for Frame 1. (a) Edge-border Coincidence, (b) Boundary Consistency, (c) Pixel Classification, (d) Object Overlap, (e) Object Contrast, (f) Combined Segmentation Quality.
uted to evaluate results), the first test is the location of a parameter combination that produces quality of 95% or higher. In experiments where the entire surface is not precomputed, this test would be discarded. Second, the process terminates if three consecutive generations produce a decrease in the average population fitness for the local population. Third, if five consecutive generations fail to produce a new maximum value for the average population fitness, the genetic process terminates. If any one of these three conditions is met, the processing of the current image is stopped and the maximum segmentation quality currently in the local population is reported.

Numerous experiments [3, 5] were performed for training and testing to measure the optimization capabilities of the genetic algorithm and to evaluate the reduction in effort achieved by utilizing previous segmentation experience. In the following we present some of these results.

4.2. PERFORMANCE COMPARISON WITH OTHER TECHNIQUES

![Segmentation Comparison](image)

Figure 7: Segmentation of Frame 1 (a–c) and Frame 11 (d–f) for the adaptive technique, default parameters, and the traditional approach.

Since there are no other known adaptive segmentation techniques with a learning capability in both the computer vision and neural networks fields to compare
our system with, we measured the performance of the adaptive image segmentation system relative to the set of default PHOENIX segmentation parameters [17, 23] and a traditional optimization approach. The default parameters have been suggested after extensive amounts of testing by researchers who developed the PHOENIX algorithm [17]. The parameters for the traditional approach are obtained by manually optimizing the segmentation algorithm on the first image in the database and then utilizing that parameter set for the remainder of the experiments. This approach to segmentation quality optimization is currently a standard practice in state-of-the-art computer vision systems. Fig. 7 illustrates the quality of the segmentation results for Frames 1 and 11 using the default parameters and the traditional approach and contrasts this performance with our adaptive segmentation technique. By comparing the extracted car region in each of these images, as well as the overall segmentation of the entire image, it is clear that the adaptive segmentation results are superior to the other methods. For the 20 frames the average segmentation quality for the adaptive segmentation technique is 95.8%. In contrast, the performance of the default parameters is only 55.6% while the traditional approach has a 63.2% accuracy. The size of the search space in these experiments is 1024, since each of the two PHOENIX parameters are represented using 5 bits. The price paid for achieving consistent higher quality of segmentation is the average number of times (2.5) one has to go through the genetic loop. Thus, only 2.4% of the search space is explored to achieve the global maximum. Many additional tests, including the comparison with random walk approach, are performed to demonstrate the effectiveness of the reproduction and crossover operators [3].

4.3. Demonstration of Learning Behavior

The above experiments were conducted in a parallel fashion, i.e., all training and all testing was performed without the aid of previous segmentation experience. Although the testing experiments used the knowledge acquired during training, the tests were still performed in parallel. None of the segmentation experience obtained during testing was applied to subsequent testing images. The following multiple day experiment shows that experience can be used to improve the segmentation quality over time. The test simulates a four day scenario where the frequency of image acquisition decreases to approximately one hour. The order of the images in this test is 1, 5, 9, 12, 16, 20, 3, 7, 11, 14, 18, 2, 6, 10, 13, 17, 4, 8, 15, 19. Each group of images in the sequence of Frames (1, 5, 9, 12, 16, 20), (3, 7, 11, 14, 18), (2, 6, 10, 13, 17), or (4, 8, 15, 19) was designed to represent a collection of images acquired on a different day.

The genetic population of the first frame in the image sequence was randomly selected. Once the segmentation performance for that frame was optimized by the genetic algorithm, the final population from that image was used to create the initial global population. This global population was then used to select the seed population for subsequent frames in the image sequence. The global population
size was set to 100 for these experiments to insure a diversity of segmentation experience in the population. While the size of the global population remained below 100 members (prior to processing 10 frames), the final populations for each image were merely added to the current global population. After the size of the global population reached 100 individuals, the final populations from each successive image had to compete with the current members of the global population. This competition was based on the fitness of the individuals; highly fit members of a new local population replaced less fit members of the global population, thus keeping the size of the global population constant. Fig. 8 presents the performance results achieved by the adaptive image segmentation system during each of the three sequential tests. The images in the first “day” (frames 1, 5, 9, 12, 16,

![Figure 8: Performance of the adaptive image segmentation system for a multiple day sequential test.](image)

show a continually decreasing level of computational effort. When the second sequence (frames 3, 7, 11, 14, 18) is encountered, the effort increases temporarily as the adaptive process fills in the knowledge gaps present as a result of the differences between the images in each sequence. The image sequence for the third “day” (frames 2, 6, 10, 13, 17) was handled with almost no effort by the genetic learning. Finally, the fourth image sequence (frames 4, 8, 15, 19) requires no effort by the genetic learning at all; each image is optimized by the information stored in the global population. Twelve of the twenty frames in this test were optimized using the global population.

5. SCALING THE NUMBER OF PARAMETERS

For the results presented in Section 4, we selected only two (hsMOOTH and mazmin) parameters of the PHOENIX algorithm. In this section, we present details when we select four parameters (hsMOOTH, mazmin, splitmin and height) for adaptive image segmentation. In this case the size of the search space is about 1 million. Table 1 shows the parameter values. As the number of segmentation parameters for adaptation increases, the number of points to be visited on the surface will also increase. However, genetic algorithms offer a number of advantages over
other search techniques. These include parallel search from a set of points with the expectation of achieving the global maximum. Unlike the Hough transform, which is essentially an exhaustive search technique commonly used in Computer Vision, it is expected that the genetic algorithm will visit only a small percentage of the search space to find an adequate solution that is sufficiently close to the global maximum.

5.1. Search Space and GA Control Mechanism

Visualization of the Search Space: Visualization of the search space allows one to understand its complexity—the number and distribution of local peaks and the location of global maximum. But this 5-dimensional space (four parameters plus the fitness or quality of image segmentation) is difficult to be visualized with traditional methods. So we project this 5-dimensional data into a 4-dimensional space by slicing it into 32 pieces along the Height axis. Fig. 9 shows the 3-D volume

![3D Volume Representation](image)

(a) Projection with height = 10

![Coordinate Axes](image)

(b) Coordinate axes

Figure 9: Volume representation of segmentation parameter search space. (a) The original 5-dimensional data (hsmooth, splitmin, maxmin, height, segmentation quality) is projected along height axis, where the color represents the fitness or segmentation quality value. (b) The coordinate system.

representation of this 4-dimensional data using the brick and slice visualization technique, where the x, y, z axes are maxmin, hsmooth, and splitmin respectively, and the color associated with each point represents the combined segmentation quality for a given parameter set. Blue color represents segmentation quality of 0, while the red color represents 100% quality.
GA Control Mechanism: GA learning requires 3 operations: selection, crossover, and mutation. In our approach, a chromosome is formed by combining the 4 segmentation parameters together. Using our method of crossover point selection, the ordering of these parameters within the chromosome does not affect the search process. Tests are carried out to select the best control parameters for GA, which include the number of crossover points, crossover rate, method of selection, population size, and quality threshold. The results given below are averaged over 1000 independent tests.

Crossover Rate: Table 2 shows the number of segmentations that are needed for frame 1 for different crossover rates. The threshold for minimum acceptable segmentation quality is 95%, population size varies from 50 to 200. We can see that a lower crossover rate leads to smaller number of total segmentations.

Table 2: Number of segmentations under varying population size and crossover rate. The threshold for minimum acceptable segmentation quality was set at 95%.

<table>
<thead>
<tr>
<th>Population</th>
<th>Crossover Rate</th>
<th>2-Point Crossover</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>80%</td>
<td>9439</td>
</tr>
<tr>
<td></td>
<td>50%</td>
<td>6077</td>
</tr>
<tr>
<td>100</td>
<td>80%</td>
<td>5805</td>
</tr>
<tr>
<td></td>
<td>50%</td>
<td>4675</td>
</tr>
<tr>
<td>200</td>
<td>80%</td>
<td>7548</td>
</tr>
<tr>
<td></td>
<td>50%</td>
<td>5068</td>
</tr>
</tbody>
</table>

Population Size and Number of Crossover Points: Table 3 shows the number of segmentations required for different population sizes and crossover points. The threshold for acceptance of segmentation quality is 95% and the crossover rate is set at 80%. From the results we can see that using more crossover points and larger population size, the total number of required segmentations can be reduced. This experiment also showed that the total number of segmentations will not reduce further when population size is greater than 500. A complete scenario for crossover operation using four points is shown in Fig. 10.

Segmentation Quality Threshold: Table 4 shows how different thresholds for minimum acceptable segmentation quality affect the total number of required
Figure 10: Genetic algorithm crossover operation. (a) Scheme for doing 4-point crossover with each chromosome containing four parameters. (b) A complete scenario for one crossover operation.
Table 3: Number of segmentations under varying population size and crossover points (Segmentation quality threshold = 95%, Crossover rate = 80%).

<table>
<thead>
<tr>
<th>Population</th>
<th>1-Point Crossover</th>
<th>2-Point Crossover</th>
<th>4-Point Crossover</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>7102</td>
<td>6553</td>
<td>5941</td>
</tr>
<tr>
<td>100</td>
<td>4960</td>
<td>5805</td>
<td>5528</td>
</tr>
<tr>
<td>200</td>
<td>4131</td>
<td>3939</td>
<td>3900</td>
</tr>
<tr>
<td>500</td>
<td>3575</td>
<td>3332</td>
<td>2878</td>
</tr>
</tbody>
</table>

Segmentations. The difference is not significant between 90% and 95% because these segmentation qualities are quite close.

Table 4: Number of segmentations under varying threshold (Population = 500, Crossover rate = 80%).

<table>
<thead>
<tr>
<th>Threshold</th>
<th>1-Point Crossover</th>
<th>2-Point Crossover</th>
<th>4-Point Crossover</th>
</tr>
</thead>
<tbody>
<tr>
<td>95%</td>
<td>3575</td>
<td>3332</td>
<td>2878</td>
</tr>
<tr>
<td>90%</td>
<td>2943</td>
<td>2788</td>
<td>2325</td>
</tr>
</tbody>
</table>

The results presented for Frame 1 in Tables 2-4 show that the number of points that are visited on the surface varies from 0.9% to 0.3% for 95% quality of segmentation. In the best case only 0.28% of the search space is visited to achieve 99.89% (threshold is 95%) quality of segmentation.

5.2. Genetic algorithms and hill climbing

Integrated search techniques have the potential for improved performance over single optimization techniques since these can exploit the strengths of the individual approaches in a cooperative manner [1, 4]. One such scheme which we describe in this section combines a global search technique (genetic algorithms) with a specialized local search technique (hill climbing). Hill climbing methods are not suitable for optimization of multimodal objective functions, such as the segmentation quality surfaces, since they only lead to local extrema. The integrated scheme provides performance improvements over the genetic algorithm alone by taking advantage of both the genetic algorithm's global search ability and the hill climbing's local convergence ability. In a sense, the genetic algorithm first finds
the hills and the hill climber climbs them.

The search through a space of parameter values using hill climbing consists of the following steps: (1) Select a starting point; (2) Take a step in each of the fixed set of directions; (3) Move to the best alternative found; and (4) Repeat until a point is reached that is higher than all of its adjacent points. An algorithmic description of the hill climbing process is given below.

1a. Select a point \( x_c \) at random.
1b. Evaluate the criterion function, i.e., obtain \( V(x_c) \).
2a. Identify points \( x_1, \ldots, x_n \) adjacent to \( x_c \).
2b. Evaluate the criterion function, i.e., obtain \( V(x_1), \ldots, V(x_n) \).
3. Let \( V(x_m) \) be the maximum of \( V(x_i) \) for \( i = 1, \ldots, n \).
3a. If \( V(x_m) > V(x_c) \) then
   set \( x_c = x_m, V(x_c) = V(x_m) \).
   goto Step 2.
3b. Otherwise, stop.

In this algorithm, a set of points that are "adjacent" to a certain point can be defined in two ways. First, it can denote the set of points that are a Euclidean distance apart from the given point. Thus, the adjacent points are located in the neighborhood of the given point. Second, "adjacent" points can denote the set of points that are unit Hamming distance apart from the given point pair. Each point in this set differs by only one bit value from the given point in binary representation of points. It defines the set of points with varying step size from the given point. The set of Hamming adjacent points was used in this research. Hamming adjacent points have an advantage over Euclidean adjacent points in our implementation because all the segmentation parameter values are represented as binary strings when using the GA. The set of Hamming adjacent points also represents the set of points which can be generated by a genetic mutation operator from the given point.

A conventional hill climbing approach, as described above, finds the largest \( V(x_m) \) from \( V(x_i), i = 1, \ldots, n \), and the search moves to its corresponding point, \( x_m \). For a space of \( n \) adjacent points, it requires \( n \) function evaluations to make each move. To reduce the cost of evaluating all the adjacent points before making each move, the approach is designed to try alternatives only until an uphill move is found. The first uphill move is undertaken without checking whether there are other (higher) possible moves. After the hill climbing process has examined all the adjacent points by flipping each bit in the binary representation of the current point, in turn, without finding an uphill move, the current point is taken as a local maximum. The algorithmic description of the hill climbing process used in the search scheme is as follows:

1. Select a starting point \( x_c \) with fitness value \( V(x_c) \) from the genetic population.
2. Set $i = 0$.
   Set $j = i$.
4a. Generate an adjacent point $x_a$ by flipping the $i$th bit in $x_c$.
4b. Obtain $V(x_a)$. Set $i = (i + 1) \mod n$.
5. If $V(x_a) > V(x_c)$ then
   set $x_c = x_a$.
   goto Step 3.
Else if $i < j$ then
   goto Step 4.
Otherwise, pass the control to the GA.

5.3. Experimental results

There are several possibilities in which genetic algorithms and hill climbing can be used. In one case the control moves back and forth between GA and hill climbing [3, 4]. In this approach when GA finds a new maximum, hill climbing is used to keep climbing until local maximum or termination condition is satisfied. If a local maximum is found then GA is again used to find a new maximum. For the experiments presented in this Chapter this approach is used for the first frame only. Specifically, the integrated technique used is given below:

1. Perform GA and hill climbing search for frame 1 using a population size of 10 (chosen from available hardware consideration) and 4 point crossover operation with a crossover rate of 0.8 (same as in Section 4). The goal here is to use small population size to achieve the desired segmentation quality with a minimum number of segmentations.

2. For frame 2 to frame 20 perform hill climbing with accumulated knowledge structures. The parameter set generated from previous frames is used to hill climb. The best result obtained for the current frame is kept as a new knowledge structure and added to the parameter set for hill climbing for the next frame.

After we are done with frame 20, a total of 29 knowledge structures are accumulated, with 19 of them generated by hill climbing.

The experimental results for frame 1 are shown in Table 5. The results show that for 95% threshold for image segmentation quality, the technique helps to reduce the required number of segmentations by almost half. For low segmentation quality threshold (90%), this effect is not dramatic.

Fig. 11 summarizes the performance of the technique for frames 1 to 20, and compares it with the performance of the default parameter set of the PHOENIX algorithm [17]. The performance corresponds to the parameter set in the population that has the highest fitness. The average performance improvement for the technique over the default parameter set is about 50%, performance improvement over the technique that uses the parameter set generated by GA plus hill
Table 5: *Performance comparison between pure GA and GA with hill climbing (crossover points = 4, crossover rate = 80%, mutation rate ≈ 1%).*

<table>
<thead>
<tr>
<th>Population = 10</th>
<th>Genetic w/o hill climbing</th>
<th>Genetic with hill climbing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Threshold = 95%</td>
<td>5941</td>
<td>3340</td>
</tr>
<tr>
<td>Threshold = 90%</td>
<td>1720</td>
<td>1631</td>
</tr>
</tbody>
</table>

climbing learning for frame 1 only (no subsequent hill climbing) is also significant. This shows that learning from frame 1 does provide a good starting point for hill climbing for subsequent frames. The maximum improvement over the default parameter set shown in Fig. 11 is 107.8%.

Fig. 12 shows the sample segmentation results obtained by using the default parameter set and the parameter set generated by the technique. Using the default parameter set, it is seen that the car does not show up at all in the segmentation result of frame 16, but the corresponding result using GA and hill climbing is quite good. The overall results show that by combining genetic search and hill climbing techniques the performance improvement is significant when the search space is large.

6. CONCLUSIONS

The goal of this research was to perform adaptive image segmentation and evaluate the convergence properties of the closed-loop system using outdoor data. In this Chapter we have provided sample results. Using the outdoor data we have shown in [3, 4, 5] that the performance improvement provided by the adaptive system was consistently greater than 30% over the traditional approach or the default segmentation parameters [17, 23].

The adaptive image segmentation system can make use of any segmentation technique that can be controlled through parameter changes. The adaptive segmentation system is only as robust as the segmentation algorithm that is employed. It may be possible to keep multiple segmentation algorithms available and let the genetic process itself dynamically select the appropriate algorithm based on image characteristics. Further, it is possible to define various evaluation criteria which can be automatically selected and optimized in a complete vision system. In a complete computer vision system, the segmentation evaluation component can be replaced by the object recognition component (for example, see [20]). In our adaptive image segmentation system, the focus is the image segmentation component. Therefore, we supplied the manually generated groundtruth image to the segmentation evaluation component and used local and global measures.
Figure 11: Performance comparison for techniques based on (a) default parameters (+), (b) GA plus hill climbing to generate the best parameter set for frame 1 only (*), and (c) integrated technique, (parameter set generated for frame 1 in the same manner as in (b) and hill climbing for all subsequent frames (o)).

Elsewhere, we have optimized both global and local measures in a multi-objective optimization framework [4]. In the future we plan to use a data set with dramatic environmental variations and we will utilize several segmentation algorithms. Ultimately, we will incorporate the adaptive segmentation component into a learning integrated object recognition system.

ACKNOWLEDGEMENTS
This material is based upon work supported by the ARPA/AFOSR under award no F49620-93-1-0624.
Figure 12: Segmentation performance comparison using default and learned parameters. (a) and (b) Frame 2, (c) and (d) Frame 3, (e) and (f) Frame 16.
REFERENCES


