

## Eye Location Using Genetic Algorithm

Jeffrey Huang and Harry Wechsler  
Department of Computer Science  
George Mason University  
Fairfax, VA 22030  
{rhuang, wechsler}@cs.gmu.edu  
<http://www.chagall.gmu.edu/>

### Abstract

*While there are many eye location methods our technique is the first to approach this location task using navigational routines and to automate the derivation of such routines using learning and evolution rather than manually handcrafting them. The adaptive eye location approach seeks first where salient things are and then what their identity is. Specifically, eye location involves (i) the derivation of the saliency attention map, and (ii) the possible classification of salient locations as eye regions. The saliency ('where') map is derived using consensus between navigation routines encoded as finite state automata (FSA) exploring the facial landscape and evolved using genetic algorithms (GAs). The classification ('what') stage is concerned with the optimal selection of features and the derivation of DT (decision trees) for confirmation of eye classification ('location') using genetic algorithms (GAs). Experimental results, using 20 unseen test face images show the feasibility of our approach, and its possible use for the location of facial landmarks and their annotation during the derivation of shape-free images.*

**Key Words:** Active Perception, Decision Trees (DT), Genetic Algorithms (GAs), Saliency Map, Shape-Free Images, Visual Routines (VR).

### 1. Introduction

Automated face recognition has become important for applications such as biometrics and forensics, telecommunication and HDTV, and medicine. There are several related (face recognition) subproblems: (i) detection of a pattern as a face (in the crowd), (ii) detection of facial landmarks, (iii) identification of the faces, and (iv) analysis of facial expressions (Samal et. al., 1992). Face recognition starts with the detection of face patterns in sometimes cluttered scenes, proceeds by normalizing the face images to account for geometrical and illumination changes, possibly using information about the location and appearance of facial landmarks, identifies the faces using appropriate

classification algorithms, and post processes the results using model-based schemes and logistic feedback. The ability to detect salient facial landmarks is an important component of any face recognition system, in particular for the derivation of shape-free images, normalization purposes, and for the extraction of image features to be used later on by the face classifiers. As both the position of the eyes and the interocular distance are relatively constant for most people, locating the eyes serves first of all as an important role in face normalization and thus facilitates further localization of facial landmarks. It is eye detection that allows one to focus attention on salient facial configurations, to filter out structural noise, and to achieve eventual face recognition.

Exploring the facial landscape involves assessing the saliency of each region in terms of the likelihood that it covers one of the two eyes such that only salient regions are tested using specific classification rules for actual eye location. The approach is conceptually similar to Active Perception (AP) (Bajcsy, 1988) and it applies as well to other exploratory tasks carried out over some unknown terrain landscape. Active perception has advanced the widely-held belief that intelligent data collection rather than image recovery and reconstruction is the goal of visual perception in general, and vision routines in particular. Active perception involves a large degree of adaptation and it provides a mobile and intelligent observer, either human or animat, with the capability to decide *where* to seek information, *what* information to pick up, and *how* to process it.

Moghaddam and Pentland (1995) have recently suggested that reactive behavior be implemented in terms of "perceptual intelligence" so the sensory input is directly coupled to a (probabilistic) decision-making unit for the purpose of control and action. An autonomous agent is then essentially a Finite-State Automata (FSA) whose feature inputs are chosen by sensors connected to the environment and/or are derived from it, and whose actions operate on the same environment. The automata decides its actions based on inputs and/or features it 'forages', while the behavior of the controller is learned. The visual routine described in this paper attempts to implement the same "perceptual intelligence" ability with respect to eye location. The important question then is how to automati-

cally craft such visual routines and how to integrate their outputs. Early attempts, developed using manual design, involved simulations lacking low-level (base) representations and operating on bitmaps only. Ramachandran (1985), an advocate of the utilitarian theory of perception, has suggested as an alternative that one could craft such visual routines by evolving a “bag of perceptual tricks” whose survival is dependent on functionality and fitness. This approach, which can be directly traced to the earlier “Neural Darwinism” theory of neuronal group selection as a basis for higher brain function (Edelman, 1987), suggests natural selection as the major force behind the automatic design of visual routines and their integration. In order to scale up to more complex behavior and greater robustness, one should look for machine learning and genetic algorithms to develop (‘learn’) and evolve such behavioral routines to specific task functionalities such as those supporting perceptual intelligence as described earlier.

Our adaptive eye location approach seeks first *where* salient things are and then *what* their identity is. Specifically, eye location involves (i) the derivation of the saliency attention map, and (ii) the possible classification of salient locations as eye regions. The saliency (‘where’) map is derived using consensus between navigation routines encoded as finite state automata (FSA) exploring the facial landscape and evolved using genetic algorithms (GAs). The classification (‘what’) stage is concerned with the optimal selection of features and the derivation of DT (decision trees) for confirmation of eye classification using genetic algorithms (GA). As the design of eye location routines addresses both attention (‘where’) mechanisms and recognition (‘what’) schemes, one has thus to address the twin problems of optimal feature selection and classifier design. As searching non-linear space is both complex and expensive this paper introduces a novel and hybrid adaptive methodology for developing eye location routines drawing on both learning and evolutionary components. The experimental results obtained using the standard FERET facial image data support our conceptual approach.

## 2. Eye Location

There are two major approaches for automated eye location. The first approach, the *holistic* one, conceptually related to template matching, attempts to locate the eyes using global representations. Characteristic of this approach is connectionist methods such as principal component analysis (PCA) using eigen-representations (Pentland et al, 1994). Although location by matching raw images has been successful under limited circumstances, it suffers from the usual shortcomings of straightforward correlation-based approaches, such as sensitivity to eye orientation, size, variable lighting conditions, and noise. The reason for this vulnerability of direct matching methods lie in

their attempt to carry out the required classification task in a space of extremely high dimensionality. To overcome the curse of dimensionality contextual information should be employed as well. As an example, Samaria (1994) employs stochastic modeling, using Hidden Markov Models (HMMs), to holistically encode facial landscape information. When the frontal facial images are sampled using top-bottom scanning, the natural order in which the facial landmarks would appear is encoded using a HMM. The HMM leads to the efficient detection of eye strips but the task of locating on the eyes still requires further processing. As the eye location methods lack size invariance, they either assume a prespecified face size or they have to operate at several multiscale grid resolutions. The use of visual routines in an iterative fashion could address this problem and implicitly achieve size invariance.

The second approach for eye detection, the *abstractive* one, extracts (and measures) discrete local features, while standard pattern recognition techniques are then employed for locating the eyes using these measurements. Yuille (1991) describes a complex but generic strategy, characteristic of the abstractive approach, for locating facial landmarks such as the eyes, using the concept of deformable templates. The template is a parametrized geometric model of the face or part of it (mouth and/or eyes) to be recognized, together with a measure of how well it fits the image data, where variations in the parameters correspond to allowable deformations. Lam and Yan (1996) extended Yuille’s method for extracting eye features by using corner locations inside the eye windows which are obtained by means of average anthropometric measures after the head boundary is first located. Deformable templates and the closely related elastic snake models are interesting concepts, but using them can be difficult in terms of learning and implementation, not mentioning their computational complexity. Eye features could also be detected using their symmetrical characteristics. As an example, Yeshurun and his colleagues (1991) developed a generalized symmetry operator for eye detection. Most recently, an attempt has been made to speed up the optimization process involved in symmetry detection using evolutionary computation. Gofman et al (1996) have introduced a global optimization approach for detecting the local reflectional symmetry in gray level images using 2D Gabor decomposition via soft windowing in frequency domain. Our approach draws from both the holistic and abstractive eye location approaches and it develops corresponding visual routines for eye location using both learning and evolution.

## 3. Evolutionary Computation and Genetic Algorithms (GAs)

Natural evolution uses selection while testing different adaptive strategies for their fitness similar to closed-loop

control. Evolutionary Computation (EC) in general, and Genetic Algorithms (GAs) in particular, could mimic nature to computationally emulate the same 'survival of the fittest' paradigm for difficult problems such as those encountered in location tasks. GAs are fully defined when one provides the strategy for deriving the offsprings and the composition of the next generation. Simulated breeding is the usual strategy used when offsprings are selected according to their fitness. Note that simulated breeding is conceptually similar to stochastic search in general, and to simulated annealing in particular, for the case when the size of the offspring population is limited to only one individual.

GAs (Goldberg, 1989) are nondeterministic methods that employ cross-over and mutation operators for deriving offsprings. GAs work by maintaining a constant-sized population of candidate solutions known as individuals ('chromosomes'). The power of a genetic algorithm lies in its ability to exploit, in a highly efficient manner, information about a large number of individuals. The search underlying GAs is such that breadth and depth - exploration and exploitation - are balanced according to the observed performance ('fitness') of the individuals evolved so far. By allocating more reproductive occurrences to above average individual solutions, the overall effect is to increase the population's average fitness.

Robust pattern analysis and classification requires the integration of various learning processes in a modular fashion. Adaptive systems that employ several strategies can potentially offer significant advantages over single-strategy systems. Since the type of input and acquired knowledge are more flexible, such hybrid systems can be applied to a wider range of problems. As an example, Hinton and Nowlan (1987) advocate the hybrid use of evolution and learning, with learning possibly easing the computational burden on evolution. Evolution (genotype adaptation) only has to get close to the goal; (phenotype) learning can then finely tune the behavior (Mühlenbein and Kinderman, 1989).

#### 4. Learning and Decision Trees

Pattern recognition, a difficult but fundamental task for intelligent systems, depends heavily on the particular choice of the features used by the classifier. One usually starts with a given set of features and then attempts to derive an optimal subset of features leading to high classification performance. A standard approach involves ranking the features of a candidate feature set according to some criteria involving second order statistics (ANOVA) and/or information theory based measures such as "infomax", and then deleting lower ranked features. Ranking by itself is usually not enough because the criteria used do not capture possible non-linear interactions among the features, nor does it measure the effectiveness of the features selected on the actual recognition task itself.

Performance ('fitness') functions for feature subsets, based on information content, orthogonality, etc., however, can still provide baseline feedback for stochastic search methods resulting in enhanced performance. As an example, GAs can search the space of all possible subsets of a large set of candidate discrimination features, capture important non-linear interactions, and thus improve on the quality of the feature subsets produced by using ranking methods only. The effectiveness of the features selected on the actual pattern recognition task can be then assessed by learning appropriate classifiers and measuring their observed performance. Learning has the ability to smooth the fitness landscape, thus facilitating evolution, and eventually what is learned becomes clamped ('phenotype rigidity').

The basic aim of any concept-learning symbolic system supporting pattern recognition and classification is to construct rules for classifying objects given a *training set* of objects whose class labels are known. The objects belong to only one class and are described by a fixed collection of attributes, each attribute with its own set of discrete values. The classification rules can be derived using C4.5, the most commonly used algorithm for the induction of decision trees (DT) (Quinlan, 1986). The C4.5 algorithm uses the entropy as an information-theoretical discriminating measure for building the decision tree. The entropy is a measure of uncertainty ('ambiguity') and characterizes the intrinsic ability of a set of features to discriminate between classes of different objects. The entropy  $E$  for a feature set  $\{f\}$  is given by

$$E(f) = \sum_{k=1}^n \sum_{i=1}^{m_j} \left[ -x_{i,k}^+ \log_2 \left( \frac{x_{i,k}^+}{x_{i,k}^+ + x_{i,k}^-} \right) - x_{i,k}^- \log_2 \left( \frac{x_{i,k}^-}{x_{i,k}^+ + x_{i,k}^-} \right) \right]$$

where  $n$  is the number of classes and  $m_j$  is the number of distinct values that feature  $f$  can take on, while  $x_{i,k}^+$  is the number of positive examples in class  $k$  for which feature  $f$  takes on its  $i^{\text{th}}$  value. Similarly  $x_{i,k}^-$  is the number of negative examples in class  $k$  for which feature  $f$  takes on its  $i^{\text{th}}$  value. C4.5 determines in an iterative fashion the feature which is most discriminatory and then it splits the data into two sets of classes as dichotomized by this feature. The next significant feature of each of the subsets is then used to further split them and the process is repeated recursively until each of the subsets contain only one kind of labeled ('class') data. The resulting structure is called a decision tree, where nodes stand for feature discrimination tests while their exit branches stand for those subclasses of labeled examples satisfying the test.

#### 5. Eye Location Architecture

Active perception works by systematically organizing the visual tasks in such a manner that as visual processing progresses in time the volume of data to attend to is reduced and computing resources are focused only on salient

regions of the image. The adaptive eye location approach seeks first *where* salient things are and then *what* their identity is. Specifically, the eye location architecture, shown in Fig. 1, involves (i) the derivation of the saliency attention map, and (ii) the possible classification of salient locations as eye regions. The saliency ('where') map is derived using consensus between navigation routines encoded as finite state automata (FSA) exploring the facial landscape and evolved using genetic algorithms (GAs). The classification ('what') stage is concerned with the optimal selection of features and the derivation of DT (decision trees) for confirmation of eye classification using genetic algorithms (GA). The saliency ('where') and recognition ('what') components are briefly described in Sect. 5.1 and 5.2, respectively, while their specific implementation is deferred to Sect. 6.

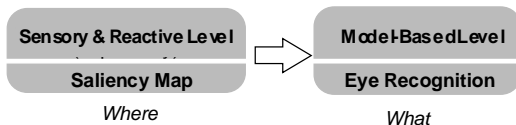


Figure 1. Architecture for Eye Location

## 5.1 Derivation of the Saliency Map

The derivation of the saliency map, shown in Fig. 2, is addressed in terms of tasks involved, computational models, and corresponding functionalities. The tasks involved include feature extraction, derivation of conspicuous ('paths') maps, and the integration of outputs from several visual routines. The corresponding computational model involves the mean, standard deviation and entropy as feature maps, finite state automata (FSA) evolved using GAs forage the feature maps and derive conspicuous paths, while consensus methods integrate the conspicuous paths into the final saliency map.

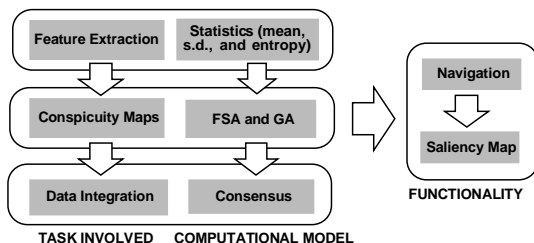


Figure 2. Derivation of the Saliency Map

## 5.2 Eye Recognition

Once the saliency map is derived, the eye recognition component has to decide if the most promising ('salient') corresponds to an actual eye location. Towards that end, the eye recognition component implements a pattern recognition classifier whose task is to make the binary deci-

sion between eye vs non-eye locations. The eye classifier is implemented as a decision tree whose adaptive derivation takes advantage of the feedback provided by evolution on how to choose an optimal feature subset. Specifically, GAs use the classification performance provided by the decision tree as the feedback needed to bias the search for an optimal feature subset and improve the population's ('DTs') average fitness.

## 6. Implementation

We first describe briefly the data preparation stage in terms of the database used and the face detection stage preceding the location of facial landmarks such as the eyes. We address then in detail the implementation for eye location consisting of two stages: (i) the 'where' stage for the derivation of the conspicuity and saliency attention maps, and (ii) the 'what' stage for eye recognition (see Fig. 2 and Sect. 5). The facial imagery used is drawn from the FERET database which has been developed over the last several years and has become a standard testbed for face recognition applications (Phillips et al, 1998). The FERET database consists of 1,564 sets comprising 14,126 images. It contains 1,199 individuals, possibly wearing glasses. Since large amounts of images were acquired during different photo sessions, the lighting conditions and the size of the facial images can vary. The diversity of the FERET database is across gender, race, and age and includes 365 duplicate sets taken at different times. Face processing starts with the detection of face patterns using FERET images taken at a resolution of 256x384. The approach used for face detection (Huang et. al, 1996) involves three main steps, those of *location*, *cropping*, and *post processing*.

### 6.1 'Where' Stage and the Saliency Map

The tasks of the 'where' stage involved include feature extraction, derivation of conspicuous ('paths') maps, and the integration of outputs from several visual routines. The corresponding computational model ('colony of animats') defines the mean, standard deviation and entropy as feature maps, finite state automata (FSA) evolved using GAs forage the feature maps and derive conspicuous paths, while consensus integrates the (conspicuous) paths into the final saliency map.

#### 6.1.1 Feature Maps

The input face images whose resolution is 192x192 using 8 bit gray levels. To account for illumination changes, the original images are processed using 5x5 Laplacian masks. The Laplacian filters out small changes due to illumination, and detects those image transitions usually associated with changes in image contents or contrast. Three feature maps corresponding to the mean, standard deviation (s.d.) and entropy are then computed over 6x6 non-overlapping

windows and then compressed using quantization to yield three 32x32 feature maps, encoded using four gray levels (2 bits) only. The original facial images are also resized to the resolution of 32x32 for display purposes. Examples of such feature maps are shown in Fig. 3 below.

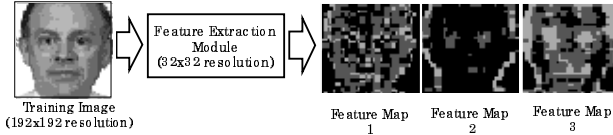
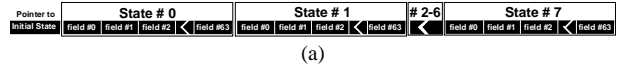


Figure 3 Feature Maps

### 6.1.2 Finite State Automata (FSA)

The FSA implements explores the previously derived 32x32 feature maps in order to generate trajectories consisting of conspicuous points on the path to salient eye locations. The FSA ('animat') starts to search its towards the eye from an initially defined initial point, the center of the bottom line of the resized (32x32) face box, corresponding approximately to the center of the chin region. The basic FSA encoding, string-like, resembles a chromosome, and the full structure of the FSA is evolved using GAs. If PS and NS stand for the present and next state, respectively, **INPUT** is drawn from the feature maps, and **ACTION** stands for some specific animat move, the FSA is then fully defined if one derives the transition function  $f: \{PS, INPUT\} \rightarrow \{NS, ACTION\}$ . The FSA is assumed to start from some initial state IS. The FSA, exploring the 32x32 facial landscape, the original face image and the corresponding three feature maps, consists of eight states. As the animat ('FSA') moves across the face, it senses ('forages') from the three pre-computed feature maps, whose composite range is encoded using 6 bits (3 features maps x 2 bits per feature = 64 level **INPUT** subfield). As measurements are taken, the animat decides on its next state and an appropriate course of action ('move'). As it is shown in Fig. 7a, both the present state (PS) and the composite feature being sensed are represented using 8 state fields <0> through <7> and 64 corresponding (feature) **INPUT** subfields <0> through <63>. The transition (FSA) function defines then the next state (NS) and move (see Fig. 4b). The animat is prevented from moving backwards by design - no backwards moves are allowed - and it can choose from five possible lateral or forward moves for a total of eight possible moves. Two of the moves are sideways (left and right), while two moves each are allocated to left 45°, straight on, and right 45°. The pointer to the initial state (Fig. 4a) and the shaded blocks from the state transition table (Fig. 4b), corresponding to the next state and move are subject to learning through evolution as described next. As learning an FSA is known as a difficult and complex computational problem we evolve FSAs using GAs. Specifically, evolution is driven by fitness, where fitness is defined as the ability of the left or right FSA to find their

way to the left or right eye within a limited number of moves (less than 64) and home on the eye within 2 pixels from the eye center. The GA component is implemented using GENESIS (Grefenstette et. al, 1991). The standard default parameter settings from the GA literature were used. This results in a constant population size of 50 FSA chromosomes, a crossover rate of 0.6 and a mutation rate of 0.001. As FSAs become more fit, the LEFT and RIGHT animats eventually learn to locate the corresponding eyes.



Present State (PS)	State #0 000				State #1 001				...	State #7 111			
Input field	0	1	...	63	0	1	...	63	...	0	...	...	63
ew state (NS)	001	101	...	000	001	111	...	110	...	100	...	...	000
Action (Move)	100	101	...	001	001	011	...	111	...	111	...	...	101

(b)

Figure 4 (a) Chromosome, (b) Transition Table,

### 6.2.3 Derivation of the Saliency Map Using Consensus

The goal for the derivation of the saliency map is to screen out most of the facial landscape as possible eye locations so the eye recognition stage can operate on less but most promising data. Note that the derivation of the saliency map takes place only during testing. For robustness purposes 20 LEFT and RIGHT animats (FSAs) search the face landscape in parallel while starting from 5 closed by points within the chin region. As a consequence, one has to collect and integrate their conspicuous ('paths') outputs to eventually determine the most salient eye locations. The motivation for the parallel search approach comes also from the fact that if one were to let loose trained animats it is likely that areas of increased traffic and subject to face modelling constraints, would correspond to the eye regions. To realize the parallel search approach, we trained different animats on the same task, LEFT and RIGHT eye location, using random seeds to initialize the FSA model described earlier. Once the (L and R) animats end their travel on the upper boundary of the face image, L and R traffic density across the facial landscape is collected, and one generates the (L - R) traffic with the expectation that the eyes will show up strongly thus indicating increased image saliency, the nose regions will cancel out, while other facial areas will show only insignificant strength. The consensus method consists then of the following steps. Left and Right path traffics are counted for a number of different Left and Right animats and several starting adjacent locations, the (L - R) traffic map is generated and its significant local maxima are then detected using hysteresis and thresholding. This procedure, stepwise illustrated in Fig. 5, shows how the colony of animats detects salient eye locations on an unseen face image using 20 Left and

20 Right trained animats starting at five close by the chin locations and consensus as described above.

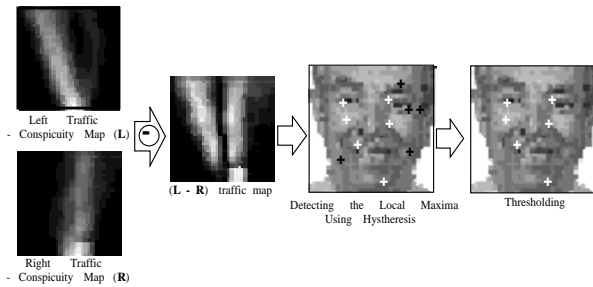


Figure 5 Saliency Derivation through Consensus

### 6.3 'What' Stage - Eye Recognition

The first step in applying GAs to the problem of feature selection for eye recognition is to map the pattern space into a representation suitable for genetic search. Since the main interest is in representing the space of all possible subsets of the original feature list, the simplest form for image base representations considers each feature as a binary gene. Each individual chromosome is then represented as a fixed-length binary string standing for some subset of the original feature list. A chromosome corresponds to an N dimensional binary feature vector X, where each bit represents the absence or inclusion of the associated feature. The advantage of this representation is that the classical GAs operators described earlier (crossover and mutation) can be easily applied to this representation without any modification. Eye recognition is activated now only on salient eye locations using the hybrid GA-DT architecture.

The strategy used to learn DT able to recognize the eyes is conceptually akin to cross-validation (CV) and bootstrapping, as it draws randomly from the original list of face region examples, both positive ('eye') and negative ('non-eye'), and it generates two labeled sets consisting of training and tuning data. The training data is used to induce trees, while the tuning data is used to evaluate the trees and generate appropriate fitness measures in terms of eye recognition accuracy and the number of eye features used. The fitness measure is feedback to the evolution module in order to generate the next subset of eye features. Once the evolution stops the best tree is frozen for use when animats are actuated on the salient eye locations found earlier. The whole process is initialized by randomly selecting some subset of eye features and making them available to the decision tree evaluation module. In order to implement the Genetic Algorithm and Decision Tree architecture we use GENESIS (GENETic Search Implementation System (Grefenstette et. al, 1991), and C4.5 (Quinlan, 1993), respectively.

We now describe in detail how one learns the DT for eye recognition. The facial images are normalized to 64x64 resolution from the original face images of size 192x192 based on our determination that the 64x64 size is the minimum required to induce accurate learning. For consistency purposes the salient locations found earlier on 32x32 images are adjusted accordingly to fit the increased dimension where eye recognition takes place. The database for training the DTs consists of 120 positive (+) eye examples and 480 negative (-) non-eye examples, manually cropped and randomly drawn from across the facial landscape. The resolution for both the positive and negative eye examples is 24x16. The feature list for each 24x16 window consists initially of one hundred and forty seven features  $\{x_1, x_2, \dots, x_{147}\}$ . The set of features, each of them measured over 6x4 windows and using two pixels overlap, is:

- $x_1$  to  $x_{49}$  : means for each window
- $x_{50}$  to  $x_{98}$  : entropies for each window
- $x_{99}$  to  $x_{147}$  : means for each preprocessed window using the Laplacian

The set of six hundred examples, 120 (+) eye and 480 (-) non-eye examples, is divided into three equal subsets for cross validation (CV) training and tuning. Once each CV is completed its results are passed back as feedback to the corresponding GA module. Note that during each trial the DT will be induced using both different features and different examples. The corresponding error rates, including both false positive - false detection of eyes - and false negative - missing the eyes, on tuning data, are shown as fitness measures in Fig. 6 as a function of the trial number for each of the three CV rounds. The feature subset corresponding to the tree derived from the third CV round, which achieved the smallest - 4.87% - error rate after 1,000 trials, consists of only 60 of the original 147 features. This feature subset is the one used to evaluate the overall performance on the eye location task using all the candidates suggested by the saliency maps.

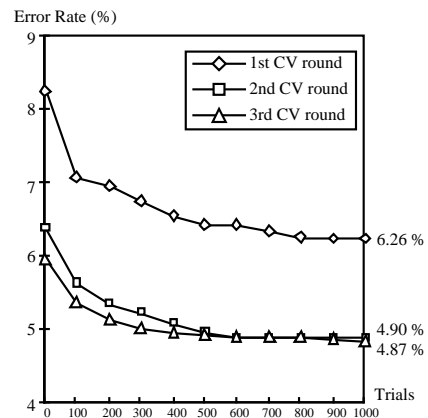


Figure 6. Fitness Measures

Due to the coarse resolution of the saliency map, the 'what' - eye recognition stage might have to process salient but adjacent locations and as a consequence multiple but redundant eye locations are found. To account for this effect, post processing using WTA (Winner Take All) is employed following the eye recognition stage. The WTA procedure clusters adjacent eye candidates, finds the cluster centers, and for robustness reasons filters out the eye locations that fail pairwise (horizontal and vertical) constrain ocular distances, like those holding between the two eyes.

## 7. Experimental Results

We use 10 face images drawn from the output of the face detection module to train 20 LEFT and 20 RIGHT Finite State Automata (FSA) for deriving the saliency ('where') map. It takes on the order of about 2,000 generations before evolution succeeds to generate such FSA, i.e., 100% performance in terms of animat trajectories starting from the chin area and passing by the eye within two pixels. Fig. 7 shows the paths followed by the animats searching for LEFT or RIGHT eye locations using some of the training images.

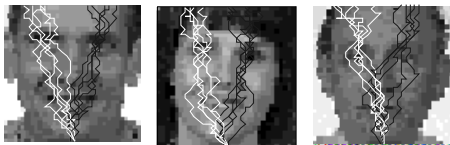


Figure 7. Conspicuous Paths Leading to the Left and Right Eye LocationFound during Training

During testing, the animats detect salient eye locations on an unseen but already detected face image using 20 Left and 20 Right trained FSAs starting at 5 close by chin locations and consensus. This goal, as it can be seen from Fig. 8, has been achieved to a large degree.

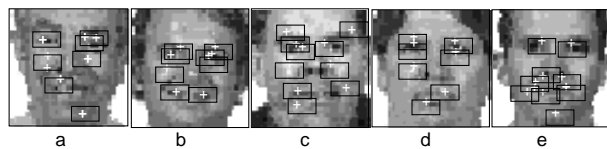


Figure 8. Salient Eye Locations

As it can be seen from Fig. 8 while all the eye locations are correctly identified as salient, several false positive eye locations have shown up as well. The 24x16 windows, centered on the salient eye locations, are now used to verify the actual presence of the eyes and possibly discard the false positive salient locations found earlier. The DT are learned using 120 eye and 480 non-eye examples drawn from 60 face images for training the eye ('what') recognition stage (see Sect. 6.3). The results of applying DT on

salient locations are shown in Fig. 9 and one can see that all the false positive salient locations have now been discarded. As discussed earlier, due to coarse resolution, multiple eye recognition can still take place at adjacent locations. Post processing using WTA as explained earlier show the final results in Fig. 10 and now one can see that all the eye locations have been correctly and uniquely identified. On the test data set of 20 face images we missed two eye locations only and no false positives were identified.

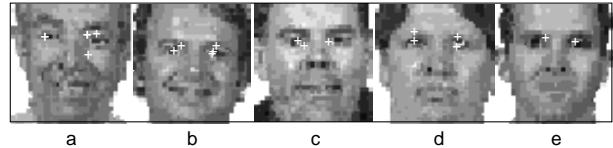


Figure 9. Eye Location Verification



Figure 10. Final Results for Eye Location

## 8. Conclusions

This paper uses eye location as a testbed for developing navigation routines implemented as visual routines (VR) driven by an adaptive behavior-based AI. While there are many eye location methods our technique is the first to approach such a location task using navigational routines and to automate the derivation of such routines using evolution and learning rather than manually handcrafting them. The adaptive eye location approach seeks first *where* salient things are and then *what* their identity is. Specifically, eye location involves (i) the derivation of the saliency attention map, and (ii) the possible classification of salient locations as eye regions. The saliency ('where') map is derived using consensus between navigation routines encoded as finite state automata (FSA) exploring the facial landscape and evolved using genetic algorithms (GAs). The classification ('what') stage is concerned with the optimal selection of features and the derivation of DT (decision trees) for confirmation of eye classification ('location') using genetic algorithms (GA). Experimental results, using facial image data, show the feasibility of our approach and suggest a novel approach for the adaptive development of task driven active perception and navigational mechanisms. The method described has the potential to find location routines scale invariant as the FSA would just further iterate on the same step when the scale is increased. One extension for future research would consider starting the animat from any location on the facial landscape and assigning it the task to locate the eye.

## Acknowledgements

This work was partially supported by the DoD Counterdrug Technology Development Program, with the U.S. Army Research Laboratory as Technical Agent, under contract DAAL01-97-K-0118.

## References

- Bajcsy, R. (1988), Active Perception, *Proc. IEEE*, Vol. 76(8), pp.996-1005.
- Edelman, G.M (1987), *Neural Darwinism*, Basic Books.
- Fukuda, T., et al. (1994), Optimization of group behavior on cellular robotic system in dynamic environment, *IEEE Int. Conference on Robotics and Automation*, pp.1027-1032.
- Gofman, Y. and N. Kiryati (1996), Detecting Symmetry in Gray Level Images: The Global Optimization Approach, *Proceeding of ICPR '96*, pp.889-894.
- Goldberg, D. E. (1989), *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley.
- Grefenstette, J. J., L. David and D. Cerys (1991), *Genesis and OOGA: Two Genetic Algorithms Systems*. Melrose, MA: TSP.
- Hinton, G. E. and S. J. Nolan (1987), How learning can guide evolution, *Complex Systems*, Vol. 1, pp.495-502.
- Huang, J., S. Gutta, and H. Wechsler (1996), Detection of Human Faces Using Decision Trees, *2nd International Conference on Automatic Face and Gesture Recognition (ICAFGR)*, Killington, VT
- Lam, K. M. and H. Yan (1996), Locating and Extracting the Eye in Human Face Images, *Pattern Recognition*, Vol.29, NO. 5, pp.771-779.
- Moghaddam, B. and A. Pentland (1995), Probabilistic visual learning for object detection, *5th Int. Conf. on Computer Vision*, Cambridge, MA.
- Mühlenbein, H., and J. Kinderman (1989). The dynamics of Evolution and Learning. Toward Genetic Neural Networks, in R. Pfeifer, Z. Schreter, F. Fogelman-Soulie, and L. Steels (Eds.), *Connectionism in Perspective*, Elsevier Science, pp. 173-197.
- Pentland, A.P., B. Moghaddam, and T. Starner (1994). "View-based and Modular Eigenspaces for Recognition", *Proceedings of Computer Vision and Pattern Recognition*, Seattle, USA.
- Phillips, J. P., H. Wechsler, J. Huang, and P. Rauss (1998), The FERET Database and Evaluation Procedure for Face-Recognition Algorithms, *Image and Vision Computing Journal*, 16 (5), pp. 295-306.
- Quinlan, J.R. (1986), The effect of noise on concept learning, in *Machine Learning: an Artificial Intelligence Approach*, edited by R.S. Michalski, J.G. Carbonell and T.M. Mitchell, Morgan Kaufmann, San Mateo, CA, pp.149-166.
- Quinlan, J. R. (1993). *C4.5: Programs for Machine Learning*, Morgan Kaufmann, San Mateo, CA.
- Ramachandran, V.S. (1985), Apparent motion of subjective surfaces, *Perception*, 14, pp.127-134.
- Rechenberg, I. (1973), *Evolution strategies: Optimierung Technischer Systeme nach Prinzipien der Biologischen Evolution*, Frommann-Holzboog Verlag.
- Samal, A., and P. Iyengar (1992), Automatic recognition and analysis of human faces and facial expressions: A Survey", *Pattern Recognition*, Vol. 25, pp. 65-77.
- Samaria, F. S., and A. C. Harter (1994), Parameterisation of a Stochastic Model for Human Face Identification, *2nd IEEE Workshop on Applications of Computer Vision*, Sarasota, Florida.
- Yeshurun, Y., D. Reisfeld, and H. Wolfson (1991),"Symmetry: A Context Free Cue for Foveated Vision", *Neural Networks for Pattern Recognition (Vol. 1.)*, H. Wechsler, ed., Academic Press.