Available online at www.elixirpublishers.com (Elixir International Journal)

Advanced Engineering Informatics

Elixir Adv. Engg. Info. 33 (2011) 2130-2133

ISSN: 2229-712X

A narrative approach for human face detection using ant colony optimization and genetic algorithm

S. Venkatesan[1] and S. Srinivasa Rao Madane[2]

[1]Dept of Computer Science & Engg , Anna University of Technology, Coimbatore,
[2]Dept. of Computer Science &Engg, Priyadarshini college of Engg, Vaniyambadi.

ARTICLE INFO

Article history:
Received: 16 February 2011;
Received in revised form:
16 March 2011;
Accepted: 25 March 2011;

Keywords
Image Enhancement,
Histogram Equalization,
Genetic Searching,
Fitness Function,
Cross-over,
Mutation,
ACO,
Genetic algorithm,
System theory.

ABSTRACT
This paper presents a novel method for detecting human faces in an image with complex backgrounds. The approach is based on visual information of the face from the template image and is commenced with the estimation of the face area in the given image. As the genetic algorithm is a computationally expensive process, the searching space for possible face regions is limited to possible facial features such as eyes, nose, mouth, and eyebrows so that the required timing is greatly reduced. In addition, the lighting effect and orientation of the faces are considered and solved in this method. Experiments demonstrate that this face detector provides promising results for the images of individuals which contain quite a high degree of variability in expression, pose, and facial details. Hybrid algorithm is proposed to solve combinatorial optimization problem by using Ant Colony and Genetic programming algorithms. Evolutionary process of Ant Colony Optimization algorithm adapts genetic operations to enhance ant movement towards solution state. The algorithm converges to the optimal final solution, by accumulating the most effective sub-solutions.

Introduction

Face detection has many real world applications, like human-computer interface, surveillance systems, video-conferencing, forensic applications, pedestrian detection, image databases, etc. However, the development of a reliable system for human face detection in a complex scene is very difficult due to variation in illumination, variability in scale, location, orientation (up-right, rotated) and pose (frontal, profile). Furthermore, facial expression, occlusion and lighting conditions change the overall appearance of face. The face detection problem has been faced up with various approaches over the last few decades: neural network, principal components, independent components, skin color based methods [1], [2].Each of them imposes some constraints: frontal view, expressionless images, limited variations of lighting conditions, hairstyle dependence, uniform background, and so on. Yokoo *et al.* [3] proposed a face detection method by searching an ellipse using the genetic algorithm. As the performance of edge detection is quite dependent on lighting conditions, so an ellipse may not occur in a true face. Yang *et al.* [4] utilized a hierarchical knowledge-based system which consists of three levels of detection. Level 1 and level 2 are based on mosaic images, so the method is difficult to locate face regions accurately. Furthermore, these methods are unable to detect a rotated human face. The main objective of this work is to propose a reliable method to detect human faces rotated at any angle. Possible face regions are selected by means of the Genetic Algorithm (GA) [5] and the fitness function is based on their projections on the template image. Experimental results indicate that the system is capable of detecting human faces in a complex scene with a high degree of variability in expression, pose, and facial details.

Face Detection Methodology

Face detection is concerned with determining which part of an image contains face. This is the first step of face recognition which requires both high- and low-level visual and geometric information processing. This work presents genetic searching for detecting human faces in a complex background. Face detection is achieved by employing template matching between a known face image and the input image. The main steps employed for the face detection process are shown in Fig. 1.

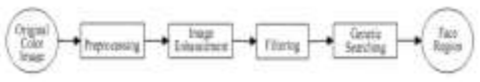

Fig. 1 Fundamental steps employed for face Detection

Preprocessing

The original image is obviously a color image. It is first converted into gray scale image. While registering images, the eyes, tip of the nose, and the corners of the mouth of each face is labeled. These points are then used to normalize each face to same scale, orientation and position. The normalization is performed by mapping the facial features to some fixed locations in an *MXN* image. Each normalized image is then subjected to some image processing operations like image enhancement and filtering to account for different lighting conditions and contrast.

Image Enhancement

The face images may be of poor contrast because of the limitations of the lighting conditions. So histogram equalization

Tele:
E-mail addresses: selvamvenkatesan@gmail.com, raomadane@gmail.com

is used to compensate for the lighting conditions and improve the contrast of the image [6]. Let the histogram of a digital face image consists of the color bins in the range , where is the *i*-th color bin, is the number of pixels in the image with that color bin and *n* is the total number of pixels in the image. For any *r* in the interval [0, 1], the cumulative sum of the bins provides with some scaling constant. Histogram equalization is performed transforming the function which produces the mapping with the allowed range of pixel values, i.e., a level *s* for every pixel value *r* in the original image [7], [8] and as shown in Fig.2

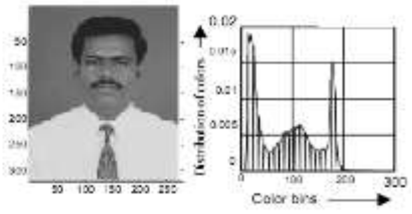

Fig. 2

Filtering

Various sources of noise may exist in the input image. The fine details of the image represent high frequencies which mix up with those of noise. So low-pass filters are used to obliterate some details in the image. In this experiment, Prewitt filter is used to suppress the noise.

ACO Genetic Algorithm (ACO)

A combinatorial optimization problem is a problem defined over a set

C = c*1*, , c *n* of basic *components*.

A subset S of components represents a *solution* of the problem; F ⊆ 2*C* is the subset of *feasible solutions*, thus a solution S is feasible,

if and only if S ∈ F.

A *cost function z* is defined over the solution domain, z : 2*C* *X* R, the objective being to find a minimum cost feasible solution S.

They move by applying a stochastic local decision policy based on two parameters, called *trails* and *attractiveness*. By moving, each ant incrementally constructs a solution to the problem.

The ACO system contains two rules:

1.Local pheromone update rule, which applied whilst constructing solutions.

2. Global pheromone updating rule, which applied after all ants construct a solution. Furthermore, an ACO algorithm includes two more mechanisms: trail evaporation and, optionally, daemon actions. Trail evaporation decreases all trail values over time, in order to avoid unlimited accumulation of trails over some component. Daemon actions can be used to implement centralized actions which cannot be performed by single ants, such as the invocation of a local optimization procedure, or the update of global information to be used to decide whether to bias the search process from a non-local perspective [6][10] At each step, each ant computes a set of feasible expansions to its current state, and moves to one of these in probability.

The probability distribution is specified as follows. For ant k, the probability of moving from state t to state n depends on the combination of two values the attractiveness of the move, as computed by some heuristic indicating the priori desirability of that move the trail level of the move, indicating how proficient it has been in the past to make that particular move it the computational effort is computed in this way, first determining the number of independent runs R needed to yield a success with a certain probability. Second, multiply R by the amount of Processing required for each run, that is. The number of independent runs R required to satisfy the success predicate by generation i with a probability z which depends on both z and P (M, i), where z is the probability of satisfying the success predicate by generation i at least once in R runs defined by:

z = 1 - [1- P (M, i)]R ……Eq.3

Represents therefore an a posteriori indication of the desirability of that move. An ACOG is differ from that algorithm given in reference, it use genetic programming (GP) to enhance performance. It consists of two main sections: *in*itialization and a main loop, where GP is used in the second sections. The main loop runs for a user defined number of iterations. These are described below:

♦Initialization:

a. Set initial parameters that are system: variable, states, function, input, output, input trajectory, output trajectory.

b. Set initial pheromone trails value.

c. Each ant is individually placed on initial state with empty memory.

♦While termination conditions not meet do

a. Construct Ant Solution: Each ant constructs a path by successively applying the transition function the probability of moving from state to state depend on: as the attractiveness of the move, and the trail level of the move.

b. Apply Local Search

c. Best Tour check:

If there is an improvement, update it.

d. Update Trails: Evaporate a fixed proportion of the pheromone on each road. For each ant perform the “ant-cycle” pheromone update. Reinforce the best tour with a set number of “elitist ants” performing the “ant-cycle”

e. Create a new population by applying the following operation, based on pheromone trails. The operations are applied to individual(s) *I is*

The Performance of Genetic Process

Genetic generation process involves probabilistic steps, because of these probabilistic steps, non convergence and premature convergence, i.e. convergence to a globally sub-optimal result problems become inherent features of genetic generation process. To minimize the effect of these problems, multiple independent runs of a problem must be made.

Best-of-run individual l from a ll such multiple independent runs can then be designated as the result of the group of runs.

If every run of GPG were successful in yielding a solution, the computational effort required to get the solution would depends primarily on four factors: population size, M, number of generation that are run, g, (g must be less than or equal to the maximum number of generation G) the amount of processing required for fitness measure over all fitness cases, and the amount of processing required for test phase e, we assume that the processing time to measure the fitness of an individual is its run time, P.

If success occurs on the same generation of every run, then the computational effort E would be computed as follows:

E= M • g • β • e ……..E q.1

Since the value of e is too small with respect to other factors, we shall not consider it. However, in most cases, success occurs on a different generations in different runs, then the computational effort E would be computed as follows:

E=M•gavr• β …….Eq.2

P (M, i) is the cumulative probability of success for all the generations between generation 0 and generation i. P(M, i) is computed after experimentally obtaining an estimate for the instantaneous probability Y (M, i) that a particular run with a population size M yields, for the first time, on a specified generation i, an individual is satisfying the success predicate for the problem]. This experimental measurement of Y(M, i) usually requires a substantial number of runs. After taking logarithms for equation 4 .

The computational effort E, is the minimal value of the total number of individuals that must be processed to yield a solution for the problem with z probability (ex: z = 99%):

$$E= M \bullet (\bullet g + 1) \bullet \beta \bullet R \text{ ....Eq.5}$$

Where •g is the first generation at which minimum number of individual evaluation is produced, it is called best generation. •g value is incremented by one since generation •g must also run to reach the solution. From equation (5), computational effort depends on the particular choices of values for M, G, P (M, i), and the effort required for fitness evaluation, hence, the value of E is not necessarily the minimum computational effort possible for the problem.

Genetic Searching

Genetic algorithm is a blind search technique which is used for searching possible facial regions in an image. Each generated solution for a problem is called a chromosome which is defined by four parameters to specify a face region. The parameters are the location (x, y), the face size, and the angle of rotation, as illustrated in Fig.3. In this method, a population of possible face regions of different locations, sizes, and slopes is generated randomly. The algorithm starts with an initial set of random solutions called the population. Each individual in the population, known as chromosome, is assigned a fitness value depending on how good its solution to the problem is. After fitness allotment, the natural selection is executed and the 'survival of the fittest chromosome' can prepare to breed for the next generation. A new population is then generated by means of genetic operations: cross-over and mutation. This evolution process is iterated until a near-optimal solution is obtained or a given number of generations is reached. However, different steps employed in the genetic algorithm are shown in Fig. 4.To apply GA for face detection, a template of the face image obtained from averaging the gradation level of pixels of a number of similar looking face images of several persons is constructed as shown in Fig. 5. The template face image is then moved through the whole image to find the location where the most suitable match exists. The genetic algorithm and different genetic operations are given below.

G.Fitness Function

To determine where a selected region is a face or not, a function needs to assign a degree of fitness to each chromosome in every generation. The fitness of a chromosome is defined as the function of the difference between the intensity value of the input image and that of the template image measured for the expected location of the chromosome. That is, for each chromosome *n*, fitness function is defined as [9], [10]:

$$f(n) = 1 - \frac{\sum_{(x,y)\in W} \left| f(x,y) - f_{n,t}(x,y) \right|}{B_{\max} \times xSize \times ySize} \qquad (1)$$

where is the maximum brightness of the image, *xSize* and *ySize* are the number of pixels in the horizontal max *B* and vertical directions of the template image, *W* is the window, *f* and are the intensity values of the original image and the template image when it is justified for the *n*th position of the chromosome, respectively.

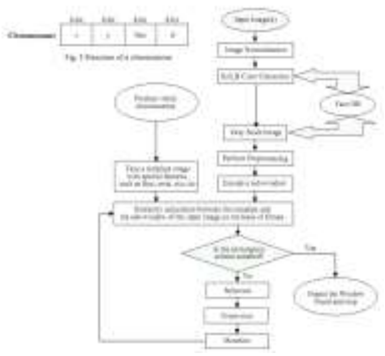

Fig. 3 Flowchart of the GA based face detection

Selection

The knowledge to breed for the next generation. Here we use conventional elitist selection scheme to select an elitist chromosome with the highest fitness value, which is copied directly into the new population of next generation. The other chromosomes are selected by a roulette-wheel selection process, where the selection probability of each individual is proportional to its fitness value.

Cross-over

Cross-over operator randomly chooses a crossover point where two parent chromosomes 'break', and then exchanges the chromosome parts after that point. As a result, two offspring are generated by combining the partial features of two chromosomes. If a pair of chromosomes does not cross over, then the chromosome cloning takes place, and the offspring are created as exact copies of each parent. This research employs single point cross-over, two point crossover and uniform cross-over operators. The cutting points are selected randomly within the chromosome for exchanging the contents. *Mutation* .Mutation operator alters a randomly selected gene of chromosome with a very low probability, *P*. For each chromosome, generate a random value between[0,1].

Table I Parameter Settings

Chromosome length	32 bits
Population size	150
Number of generation	300
Cross-over probability	0.7
Mutation probability	0.01

The basic steps of the genetic algorithm are as follows:

Step 1 : Initial Population: Generate randomly a population of chromosomes of size *N*: . Assign the crossover probability *P* and the mutation probability *P*

Step 2 : Evaluation Function: Evaluate the fitness function for each chromosome in the population.

Step 3 : Selection: Select a pair of chromosomes for mating. Use the roulette wheel selection procedure, where each chromosome is given a slice of a circular roulette wheel. The area of the slice within the wheel is equal to the chromosome fitness ratio. Obviously, the highly fit chromosomes occupy the largest areas, where the chromosomes with least fit have much smaller segments in the wheel. To select a chromosome for mating, a random number is generated in the interval [0, 100], and the chromosome whose segment spans the random number is selected.

Step 4 : Cross-over: Produce two off-springs from two parent chromosomes.

Step 5 Mutation: Apply the conventional mutation operation to the population with a mutation rate *P*

Step 6 : Termination Test: If a predefined termination condition is satisfied, go to Step 7, else go to Step 2. Step 7 Preservation: Keep the best chromosome and Stop.

Experimental Results

The effectiveness of this approach is justified using different images with various kinds of expressions. When a complex image is subjected in the input, the face detection result highlights the facial part of the image, as shown in Fig.6. For multiple faces, the system finds the dominant face only. Images of different persons are taken at their own working places and at different environments both in shiny and gloomy weather. A total of 260 images including more than 70 different persons are used to investigate the capacity of the proposed algorithm. Among them only 8 faces are found false. This demonstrates that the success rate of the system is approximately 97%. The main reason behind the failure of those images in finding face regions is the occlusion. The genetic algorithm is examined using single point and uniform cross-over with different population sizes and the results are illustrated graphically in Fig. 7 and Fig. 8respectively. Fig. 7 reveals that larger population size offer better performance because of the larger pool of diverse schemata available in the chromosome but the inertia of larger population also boils down a problem of poorer initial. Fig. 8 shows that smaller population size is better for uniform cross-over. So, a trade off is always taken between population size and the way of cross-over. Therefore, this research adopts single point cross-over with a population size of 150 during face detection process. The Experiments are performed on a Pentium IV 1.70GHz PC using Visual C++. The processing time for locating faces in a picture of size 320×240 is 2s to10s.Some experimental results are shown in g. 6.

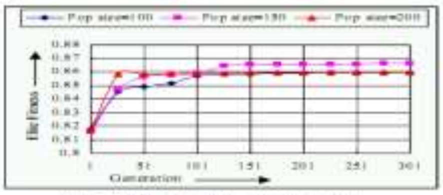

Fig. 7 Elite fitness versus generation (Single point cross-over)

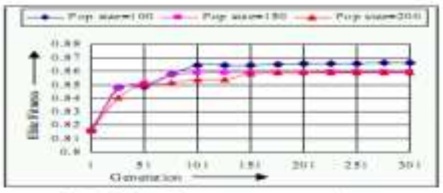

Fig. 8 Elite fitness versus generation (Uniform cross-over)

Conclusion

Detection of faces and facial features using machine vision techniques has many useful applications. Though human beings accomplish these tasks countless times a day, they are still very challenging for machine vision. Most of the researchers attack this kind of problem with face localization and feature selection with frontal view faces and without facial expressions and normal lighting conditions although the variation between the images of the same face is too large due to facial expression, hair style, pose variation, lighting conditions, make-up, etc. In this paper, face detection has been implemented using Ant colony optimization and genetic algorithm to search for the face of a particular individual in an image. The effectiveness of the face detection algorithm has been tested both in simple and complex backgrounds for different types of face and non-face images of 320×240 resolution. The algorithm is capable of detecting the faces in the images with different backgrounds. A rotated human face can also be detected, even if the face is under shadow, wearing glasses, or under bad lighting conditions.

References

1. Abraham: Evolutionary Computation: from Genetic Algorithms to Genetic Programming, Studies in Computational Intelligence (SCI) 13, 1–20 (2006)

2. Grosan and A. Abraham: Hybrid Evolutionary Algorithms: Methodologies, Architectures, and Reviews, Studies in Computational Intelligence75,1–17(2007),

3. Hector A Montes and Jeremy L Wyatt ," Cartesian genetic Programming for Image Processing Tasks",

4. Introductory Tutorial and a Survey of Techniques &applications",Technical ReportCES-475ISSN:1744-80502007.

5. M. Dorigo, M. Birattari, and T. Stitzle, "Ant Colony Optimization: Arificial Ants as a Computational Intelligence Technique, IEEE computational intelligence magazine, November, 2006.

6. M. Dorigo, G. Di Caro, and L.M. Gambardella, "Ant algorithm for discrete optimization", Artificial

7. Nada M. A. AL-salami, Saad Ghaleb Yaseen, "Ant Colony Optimization", IJCSNS International Journal of Computer Science and Network Security, VOL.8 No.6, pp 351-357, June, 2008.

8. Simon Harding, Julian F. Miller, Wolfgang Banzhaf, "Self-Modifying Cartesian Genetic Programming", GECCO'07, July 7–11, 2007, ACM 978-1-59593-697-4/07/0007, pp: 1021-1028.Nada M.A. AL-Salami, "System Evolving using Ant Colony Optimization Algorithm ", Journal of Computer Science 5 (5): 380-387, 2009, ISSN 1549-3636..

9. Talay, A. Cagatay, An approach for eye detection using parallel genetic algorithm, V.S. Sunderam et al. *(Eds.)*: ICCS 2005, LNCS 3516. p. 1004-1007, 2005

Authors

S.Venkatesan Pursuing Research in Computer Science and Engineering in Anna University of Technology Coimbatore, Tamilnadu India. His area of interest includes Image Processing, Soft Computing, Pattern Recognition, and Optimization Techniques.

Dr.S.Srinivasa Rao Madane Doctorate in Computer Science and Engineering, currently working as Professor & Principal in Priyadarshni Engineering College. Vaniyambadi Tamilnadu India, His area of interest includes Neural Networks, Fuzzy logic, Analog and Digital Communication, and Image Processing.