Parametric L-System Description of the Retina with Combined Evolutionary Operators

Gabriella Kókai
Dept. of Computer Science
Friedrich-Alexander Univ. of Erlangen-Nürnberg
Martensstr. 3. D-91058 Erlangen, Germany
e-mail: kokai@informatik.uni-erlangen.de

Róbert Ványi, Zoltán Tóth
Institute of Informatics
József Attila University
Árpád tér 2, H-6720 Szeged, Hungary
e-mail: h531774|h531714@stud.u-szeged.hu

Abstract

In this paper the *GREDEA* system is presented. The main idea behind it is that with the help of evolutionary algorithms a grammatical description of the blood circulation of the human retina can be inferred. The system uses parametric *Lindenmayer systems* as description language. It can be applied on patients with diabetes who need to be monitored over long periods of time.

1 INTRODUCTION

In this paper the *GREDEA* (*Grammatical Retina Description with Evolutionary Algorithms*) system is presented. The main idea behind *GREDEA* was to develop patient-specific monitoring programs for examining the blood circulation of the human retina. The system can be used for on patients with diabetes who need to be monitored over long periods. It is essential to check the eyesight of patients with this disease, because the deterioration of the vascular tree caused by diabetes has a direct effect on the vision quality.

In recent years some systems (such as [Georgiadis 1995], [Hipwell 1997], [Zana 1997]) have been developed but these systems employed computer graphic methods. They require comparison of different images taken in regular intervals. To make a diagnosis based on these images is a difficult task and needs a lot of expertise. Another attempt to use a syntactic approach is the system developed by Hesse et al. [Hesse et al. 1993] Here fractal modelling was applied but the authors concentrated on simulating the typical growth pattern of the retinal artery and so this work has no real diagnostic benefit.

In *GREDEA* an individual description of the blood circulation of the human retina is created for each patient. To obtain the description the process starts from fundus images which are taken with scanning laser ophthalmoscope (*SLO*) and are preprocessed. We assume that background is black (0 values) and the vascular tree is white (1 values). These images are referred to later as *destination images*.

Then a parametric *Lindenmayer system* (*L-system*) is developed which creates the pattern closest to the vascular tree of the patient. This *L-system* can be stored (needs less storage than storing a picture) and used later to make comparisons. *L-systems*

are parallel grammatical rewriting systems. They contain an initial axiom and one or more rules. With the help of such a grammar complex, objects can be described starting from the axiom and using the productions [Rozenberg, Salomaa 1985]. When choosing the representation form, the fact that *L-systems* can be evolved using evolutionary algorithms [Koza 1992] was taken into consideration.

Since parametric L-systems are evolved, two kinds of evolutionary algorithms are applied: *genetic algorithms* [Goldberg 1989] are used on the rewriting rules of the evolved L-system, and *evolution strategies* [Rechenberg 1973] are applied on the parameters of the L-systems.

We proceed as follows. First in Section 2 and 3 a brief introduction to the recognition of the vascular tree in the retina and some details about the L-systems is given. The evolution process is presented in Section 4. In Section 5 an example for the experiments is discussed. Finally Section 6 contains a summary and some comments on future work.

2 THE RETINA

The retina is the light sensitive surface at the back of the eye [Esperance 1983]. The retina contains a vast number of specialised nerve cells called rods and cones that convert light rays into nerve signals. The cones provide color vision, and are most concentrated at the fovea. The fovea can be seen in Figure 1 as the region on the side without any blood vessels crossing it. Because of their dense concentration, the cones in the fovea provide the finest resolution vision.

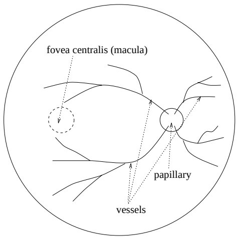

Figure 1: The Structure of the Retina

The rods are much more sensitive to light than the cones but they only provide black and white vision. All of the rods and cones connect with nerves whose signals exit from the eye via the optic nerve. The optic nerve along with the retinal artery, joins the eye at the optic disk. The optic disk is visible in Figure 2.

There are no rods or cones in the optic disk which creates a small blind spot on the retina. The vessel system of the eyebackground consists of two types of blood vessels, arteries and veins. The part where the vessels (arteries and veins) enter the eye is called papillary. Arteries and veins may have some crosspoints in the eye background, but arteries never cross arteries and veins never cross veins. The blood vessels can be described with a binary tree, because from a knot a vessel can branch off only in two directions. The vessels starting from the papillary surround the macula. The macula does not contain any vessels.

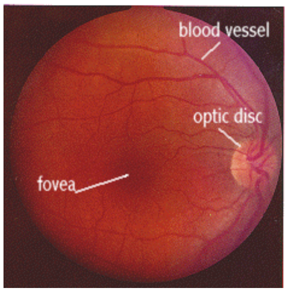

Figure 2: Fundus image of the retina

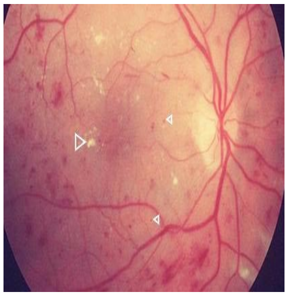

Figure 3: Diabetic retinopathy

When diabetes affects the eyes, it often affects the retina lining the back of the eye, specifically the small blood vessels (see in Figure 3) which nourish this membrane. This condition is called diabetic retinopathy. Diabetic retinopathy often has no symptoms in its early, most treatable stages. Improving awareness of the importance of early detection and treatment is imperative if one is to reduce visual loss from this disease. A minimal evaluation of the diabetic retina consists of regular examinations by a qualified practitioner though dilated pupils.

3 THE LINDENMAYER SYSTEMS

Lindenmayer introduced special kinds of descriptive grammars in order to describe the natural development processes of plants [Lindenmayer 1968]. These are called Lindenmayer grammars or *Lindenmayer systems* (*L-system*s). The most important characteristic of *L-systems* compared with Chomsky grammars is *parallel rewriting*. In Lindenmayer systems rewriting rules are applied in parallel and simultaneously replace all letters in a given word.

Different *L-system* classes are introduced, such as deterministic or nondeterministic *L-systems*. The distinction can be made between context free and context sensitive *L-systems*. In this paper only *deterministic context free L-systems* (*D0L-systems*) are used. To enlarge the number of described objects, *L-systems* can also be associated with parameters.

3.1 PARAMETRIC D0L-SYSTEMS

With the idea of parametric *L-systems*, the concept of parallel rewriting to parametric words instead of strings of symbols, can be applied. Following the notation of Prusinkiewicz [Prusinkiewicz, Hanan 1989] the word "module" as a synonym for "letter with associated parameters" is used.

The alphabet is denoted by Σ, and the set of parameters is the set of real numbers $\boldsymbol{R}$. A module with letter $a \in \Sigma$ and parameters $p_1, p_2, \ldots, p_n \in \boldsymbol{R}$ is denoted by $a(p_1, p_2, \ldots, p_n)$.

The real-valued *actual* parameters appearing in words have a counterpart in the *formal* parameters which may occur in the specification of *L-system* productions. Let Π be the set of formal parameters. The combination of formal parameters and numeric constants using the arithmetic operators $(+,-,*,/)$ the relational operators $(<,<=,>,>=,==)$, the logical operators $(!,\&\&,||)$, and parentheses $(())$ generates the set of all correctly constructed logical and arithmetic expressions with parameters from Π. There are noted $C(\Pi)$ and $\mathcal{E}(\Pi)$. A *parametric D0L-system* is an ordered quadruple $G = (\Sigma, \Pi, \alpha, P)$, where

- $\Sigma = s_1, s_2, \ldots, s_n$ is the alphabet of the system,
- Π is the set of formal parameters,
- $\alpha \in (\Sigma \times \boldsymbol{R}^*)^+$ is a nonempty parametric word called the axiom,
- $P \subset (\Sigma \times \Pi^*) \times C(\Pi) \times (\Sigma \times \mathcal{E}(\Pi)^*)^*$ is a finite set of productions.

A production in a *parametric D0L-system* is given in the form *pred* | *cond* → *succ*, where *pred* $\in \Sigma \times \Pi^*$ denotes the *predecessor*, *succ* $\in ((\Sigma \times \mathcal{E}(\Pi)^*)^*$ is the *successor* and *cond* $\in C(\Pi)$ describes the *conditions*. If the letter in the module and the letter in the production predecessor are the same, the number of actual parameters in the module is equal to the number of formal parameters in the production predecessor and the condition evaluates to true then the production *matches* a module

If a production is matching a module then it can be *applied* in order to create a string of modules specified by the production successor. The actual parameter values are substituted for the formal parameters according to their position.

There is no essential difference between *D*0*L-systems* that operate on strings with or without brackets. In our implementation brackets are used. However, in this case productions involving brackets are restricted to the following forms:

- *pred* | *cond* → *succ*, where *pred* and *cond* are as before, but *succ* $\in ((\Sigma \times \mathcal{E}(\Pi)^* \cup \{[,]\})^*$, and *succ* is well nested;
- [→ [, or] →].

A string S is called well-nested if and only if for all S' prefix of S the number of '[' symbols is not less than the number of ']' symbols, and the number of these symbols is equal in S. More details and examples of parametric *D*0*L-systems* can be found in [Prusinkiewicz, Hanan 1989].

3.2 DISPLAYING IMAGES DEFINED BY L-SYSTEMS

To display an image defined by an *L-system* the well-known turtle graphic is used [Abelson, diSessa 1982]. The turtle is an object in space such that its position is defined by a vector and its orientation is defined by a coordinate-system. The coordinate-system determines the heading($\vec{h}$), the left($\vec{l}$) and the up($\vec{u}$) directions. Using these vectors the turtle can be moved in space. There are several turtle operations, also known as turtle commands. The most important ones are drawing ('F'), and turning left and right ('$+$' and '$-$'). There are special turtle operations, called stack operations. These are denoted by '[' and ']', and used to save and restore turtle position and orientation. Using stack operations branching structures can be produced.

To use turtle graphics with *L-system*, turtle commands as the elements of the alphabet are applied.

4 THE EVOLUTION PROCESS

If an image is given and one wants to find the grammar that describes it, this is called the inverse problem of L-systems. Evolutionary algorithms are used to solve this problem and are described here in detail.

4.1 INITIAL POPULATION

The first step is to define the structure of the initial population and the individuals themselves. For the first examination the initial population is be randomly created, but if the patient returns for a check-up, the previously generated L-system is be used to create the initial population. This makes the convergence speed greater and makes comparison easier.

As was mentioned before, the individuals are represented by *parametric L-systems*. According to the definition parameters and constant values may appear in the rewriting rules. We consider the *constant values* as *parameters* of the whole L-system. For example if an $X \to Y(10)$ is present, it is assumed that an $X \to Y(A)$ rule is used and the L-system has a parameter called 'A' with a current value of 10. It is also assume that the axiom is S.

Because the shape of the generated image depends both on the rewriting rules and on the parameters, the representation can be divided into two important parts. The right sides of the rules are represented as strings while the parameters are described as vectors of real values. Usually it is more important to find the appropriate right sides. Changing the parameters is used for the fine-tuning of the image.

L-systems with the same rules are considered as an *L-system group*. In an *L-system group* the rules are constant, but the parameters can be varied. To find the best parameters *evolution strategies* (*ES*) are used inside the *L-system group*. Therefore an *L-system* group works as a *subpopulation*. A subpopulation can be represented by rewriting rules. Because the number of the rules, and also the left sides are constant, an *L-system group* can be represented by an array of strings. To find therewriting rules, *genetic algorithm* (*GA*) is applied on the string arrays representing an *L-system group*. So the *L-system groups* make up the population. The structure of the population can be seen in Figure 4.

4.1.1 The Structure of the Population

To calculate the fitness value, the images described by the L-systems are created, then the distance between these generated images and the destination image is calculated with a suitable function. This function will be discussed in section 4.2. Using this fitness value *evolution strategies* are applied on the parameter vectors several times. The number of the evolution steps (denoted by S_{ES}) is defined at the very beginning of the process. When the ES steps are finished, the best individuals in each subpopulation are selected. The parameters and the fitness value of these *L-systems* will act as

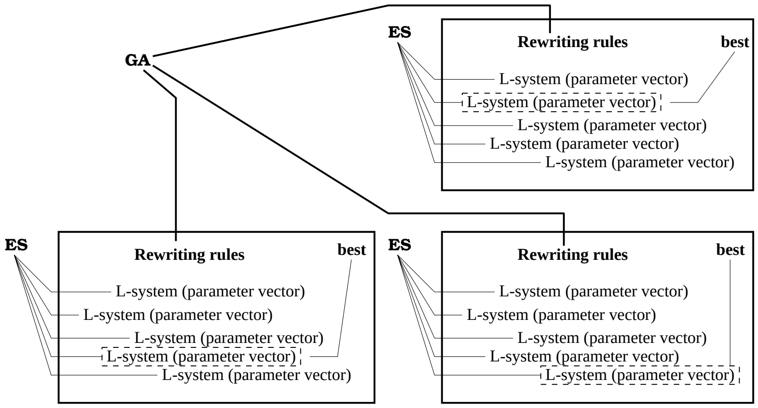
GA
ES
Rewriting rules
best
L-system (parameter vector)
L-system (parameter vector)
L-system (parameter vector)
L-system (parameter vector)
L-system (parameter vector)
ES
Rewriting rules
best
L-system (parameter vector)
L-system (parameter vector)
L-system (parameter vector)
L-system (parameter vector)
L-system (parameter vector)
ES
Rewriting rules
best
L-system (parameter vector)
L-system (parameter vector)
L-system (parameter vector)
L-system (parameter vector)
L-system (parameter vector)

Figure 4: The structure of the population

the parameters and the fitness value for the whole subpopulation. Using this fitness value several subpopulation can be selected and can then be used to create new subpopulation with *genetic algorithms* (*GA*). The detailed description of the applied ES and GA operators can be found later in section 4.3. Earlier we mentioned that ES is usually applied on real valued vectors, however GA is used with words over a finite alphabet. Therefore in this process both of them are used.

4.1.2 The Form of an Individual

The base *L-system* used as an individual is the following:

alphabet: $\Sigma = \{S, F, +, -, [,]\}$
axiom: $\alpha = S$
rewriting rule: $S \rightarrow u$, where $u \in \Sigma^*$ is a well-nested string.

Using this approach it can be noticed that this type of *L-system* sometimes generates *unnecessarily long word*s (*ul-words*). For example the image described by the word $F{+}{-}{-}{+}{+}F{-}{+}$ can be also generated by the word $F{+}F$, which is much shorter. To solve this problem, sequences of symbols can be defined, such that these sequences always start with a drawing symbol and do not contain unnecessary turtle movements (e.g. $+{-}{-}+$). If *regulated evolution* is used to *preserve sequences* those ul-words can be avoided [Jacob 1994]. During regulated evolution, sequences cannot be broken, so genetic operators must be applied on substrings and not on single symbols. But this makes the genetic algorithm more difficult. The other solution employs the specialities of the problem. Examining the blood vessel in the eye, a better *L-system* can be defined which itself includes the sequences:

$$\Sigma = \{S, F, +, -, [,], B, Y, T\}$$
$$\alpha = S$$

and rewriting rules:
$S \rightarrow v$, where $v \in \{S, B, Y, T, [,]\}^*$ is a well-nested string.
$B \rightarrow F{+}S$
$Y \rightarrow F[+S][-S]$
$T \rightarrow F[+S]S$

Note that the vascular tree can contain only the following parts: *bending* (B), *branching* (Y) and *forking* (T). Using this *L-system*, only the first rule will be modified during the evolution process. The initial F symbols of the other three rules ensure that ul-words will not be created. So this *L-system* will produce better words, just like the *L-system* with sequences, without making the use of genetic operators difficult.

The turtle commands F,$+$ and $-$ all have one parameter to define the length of the line, or the angle of rotation respectively. To give these parameters we have to slightly modify our L-system. So the form of an individual is the following *parametric L-system*:

$\Sigma = \{S, F(a), +(a), -(a), [,], B(a,b), Y(a,b,c), T(a,b)\}$
$\alpha = S$
and rewriting rules:
$S \rightarrow v$, where $v \in \{S, B(a,b), Y(a,b,c), T(a,b), [,]\}^*$
is a well-nested string, with constant parameters.
$B(a,b) \rightarrow F(a){+}(b)S$
$Y(a,b,c) \rightarrow F(a)[+(b)S][-(c)S]$
$T(a,b) \rightarrow F(a)[+(b)S]S$

The form of the first rule is like the following:

$$S \rightarrow B(A,B)Y(C,D,E)Y(F,G,H)T(I,J)B(K,L).$$

In this case the L-system has twelve parameters: from 'A' to 'L'.

4.2 THE FITNESS FUNCTION: THE DISTANCE OF TWO IMAGES

There are many possibilities to define the distance between two images. A very simple method to calculate the distance

d_q between two images I_1 and I_2 is the quadratic error of the images. During the definition we assume that the sizes of both pictures are the same: $N \times M$.

The distance is the sum of quadratic errors of two corresponding pixels:

$$d_q(I_1,I_2) = \sum_{x=0}^{N-1}\sum_{y=0}^{M-1}(I_1(x,y) - I_2(x,y))^2 \qquad (1)$$

For binary images, where $I_1^B(x,y)$ and $I_2^B(x,y)$ are two pixels the quadratic error is:

$$d_p(I_1^B(x,y), I_2^B(x,y)) = \begin{cases} 1 & \text{if } I_1^B(x,y) \neq I_2^B(x,y) \\ 0 & \text{otherwise} \end{cases} \qquad (2)$$

So let the distance between two binary pictures be defined like this:

$$d_q(I_1,I_2) = \sum_{x=0}^{N-1}\sum_{y=0}^{M-1} d_p(x,y) \qquad (3)$$

In this paper an extended version of this definition is used because in our evolutionary process we wanted to make a distinction between the following four cases for an arbitrary (x,y) point:

- The point (x,y) is 0 on both images, it is usually not important for calculating the fitness.
- The point (x,y) is 1 on both images. In this case a match is found; this means positive fitness.
- The point (x,y) is 0 on the destination image, but 1 on the generated image. This means that a point that is not part of the vascular tree is covered.
- The point (x,y) is 1 on the destination image, but 0 on the generated image. So there is a missing point.

Let us denote the pixels with $p_d = I_d^B(x,y)$ and $p_g = I_g^B(x,y)$, where p_d means the pixel value on the destination image I_d and p_g means the value on the pixel in the generated image I_g. In this case the weighted distance of two pixels can be defined:

$$d'_p(x,y) = \begin{cases} W_0 & \text{, if } p_d = p_g = 0 \\ W_1 & \text{, if } p_d = p_g = 1 \\ -W_2 & \text{, if } p_d < p_g \\ -W_3 & \text{, if } p_d > p_g \end{cases} \qquad (4)$$

So the fitness function φ is the weighed distance between two images:

$$\varphi = d'(I_1,I_2) = \sum_{x=0}^{N-1}\sum_{y=0}^{M-1} d'_p(x,y) \qquad (5)$$

Using this definition a missing point will cost more than an unnecessary one if W_3 is set to greater than W_2. These parameters can also be used to fine-tune the fitness function. Note that W_0 is usually 0.

4.2.1 Calculating the Fitness

Comparing two images pixel by pixel is too slow, even with the fast computers used nowadays. Well, it is not important if we have to do this only once. But during the evolution process fitness has to be calculated many times. So it is important to have a fast fitness calculating procedure. Notice that black pixels (0 values) do not count in the fitness value. So leaving the black pixels can increase calculation speed. The best way to do this is combining the turtle with the fitness calculation, resulting in an object called *fitness-calculating turtle* (*FCT*).

When the FCT draws an object, it also examines the corresponding pixels of the destination image. According to the result, the pixel is classified as *fpixel* if the pixel is present in the destination image or *upixel* if not. The number of these pixels is denoted by N_f and N_u. Before the evolution process, the number of black pixels (N_0) and the number of white pixels (N_1) can be calculated. Using these values we can determine the number of pixels which appear in the destination image but not in the generated image. We call this kind of pixels *mpixel*s. The number of these pixels is denoted by N_m and can be calculated in the following way:

$$N_m = N_1 - N_f \qquad (6)$$

The not examined pixels are called *npixel*s. The number of these pixels is:

$$N_n = N_0 - N_u \qquad (7)$$

Using these names the four cases of a pixel is npixel, fpixel, upixel and mpixel according to the list already mentioned in section 4.2. With these classes the fitness calculation can be modified:

$$\begin{aligned} \varphi &= d'(I_1,I_2) = \textstyle\sum_{x=0}^{N-1}\sum_{y=0}^{M-1} d'_p(x,y) \\ &= \textstyle\sum_{npixel} d'_p + \sum_{fpixel} d'_p + \sum_{upixel} d'_p + \sum_{mpixel} d'_p \\ &= \textstyle\sum_{npixel} W_0 + \sum_{fpixel} W_1 + \sum_{upixel} W_2 + \sum_{mpixel} W_3 \\ &= W_0N_n + W_1N_f - W_2N_u - W_3N_m \end{aligned} \qquad (8)$$

Namely instead of comparing each pixel we can use weighed sum of the numbers of pixels in each class:

$$\varphi = W_0N_n + W_1N_f - W_2N_u - W_3N_m \qquad (9)$$

Applying equation 6 and 7, the modified fitness function is the following:

$$\begin{aligned} \varphi &= W_0N_n + W_1N_f - W_2N_u - W_3N_m \\ &= W_0N_n + W_1N_f - W_2N_u - W_3(N_1 - N_f) \\ &= W_0N_n + (W_1 - W_3)N_f - W_2N_u - W_3N_1 \\ &= W_0(N_0 - N_u) + (W_1 - W_3)N_f - W_2N_u - W_3N_1 \\ &= W_0N_0 + (W_1 - W_3)N_f - (W_0 + W_2)N_u - W_3N_1 \end{aligned} \qquad (10)$$

Assuming that $W_0 = 0$ and denoting the constant values with c_i we get an easy to use formula:

$$\varphi = c_{13}N_f - c_{02}N_u - c \qquad (11)$$

Modifying the turtle to calculate this function makes turtle slower. The comparison of the normal drawing turtle, and the fitness calculating turtle can be seen in Figure 5. On the x axis the image resolution can be seen in pixels. We used equal sided images, so the number of the pixels is the square of the x value. On the y axis the running time is displayed in microsecundums.

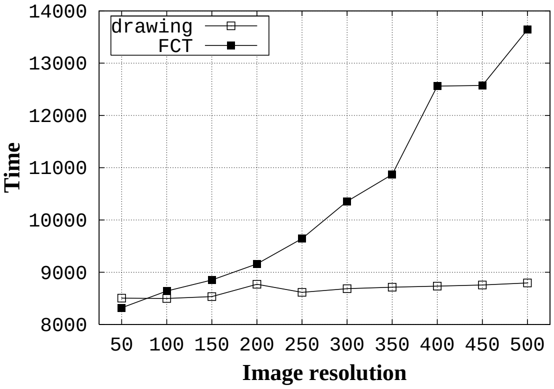

Figure 5: Turtle running times

However FCT also does fitness calculation. When using the normal drawing turtle fitness calculation must be made separately. The time required to compare images by pixels using equation 5 is displayed in Figure 6.

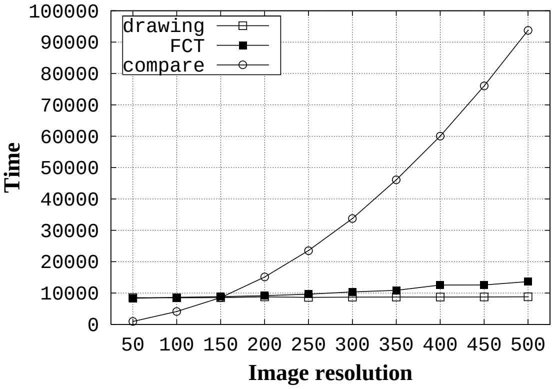

Figure 6: Fitness calculation times

One can notify that the time needed to calculate the original fitness function depends on the size of the picture. But the modified fitness calculation takes only time depending on the size of the vascular tree area. For example if the vascular tree is only the 10% of the image, the modified procedure can be 10 times faster.

4.3 THE USED OPERATORS

Since parametric L-systems are evolved, two kinds of evolutionary operators are applied. Genetic algorithms are used on the rewriting rules of the evolved L-system and evolution strategies are applied on the parameters of the rules. In this section these are described.

4.3.1 The GA Operators

Let us denote two strings with $u = u_1u_2\dots u_n$ and $v = v_1v_2\dots v_n$. During the evolution of strings two GA operators are used:

- *GA mutation.* When vector u is mutated, the resulting vector will be u' where $u'_k = u_k$ if $1 \leq k < i$ or $i < k \leq n$ and $u'_i = T$ for a randomly selected i position, and a randomly selected T character.
- *GA recombination.* When the u and v vectors are recombined, the resulting vector will be w, where $w_i = u_i$, if $i \leq k$ and $w_i = v_i$, if $k < i$ for a randomly selected k.

4.3.2 The ES Operators

Let us denote two parameter vectors with $u = u_1u_2\dots u_n$ and $v = v_1v_2\dots v_m$. To evolve these parameter vectors two different ES operators are used, similar to the operators used at GA. These operators are:

- *ES mutation.* When vector u is mutated, the resulting vector will be u' where $u'_k = u_k$ if $1 \leq k < i$ or $i < k \leq n$ and $u'_i = u_i + \xi(R_{ES})$ for a randomly selected i. $\xi(R_{ES})$ represents the Gaussian derivation with deviation R_{ES}. This R_{ES} parameter is called the *radius* of the mutation, and is defined before the evolution process.
- *ES recombination* When the u and v vectors are recombined, the resulting vector will be w, where

$$w_i = \frac{(u_i + v_i)}{2}.$$

4.4 HALTING CRITERIA

Because our image is a finite set of discrete points there is a rough upper bound for the fitness value: $(N \times M) \cdot W_1$. The maximum fitness can be also given: $\varphi_{max} = N_1W_1$. Therefore after a finite number of steps there will be a population fitness which is identical with the previous one, if decreasing fitness is not allowed. The process can be stopped if the maximum fitness is reached, or nearly reached, or the fitness value was unchanged during the previous S_{GA} steps.

4.5 CHOICE OF THE EVOLUTION PARAMETERS

There are several parameters that can be changed before the process and also during the evolution. The *GREDEA* system can currently be used to find L-systems for simpler retina images. An example is presented in Section 5. To find L-systems for more complex retina images, the starting parameters of the process must be set more carefully. In order to do this, more retina images must be analised. This analysis currently being carried out is our project.

- *General parameters.*: The user can change the number of the subpopulation and the number of individuals in a population.
- *EA parameters.*: To control the evolution strategy parameters are defined as previously mentioned: S_{ES}, W_1, W_2, W_3 and R_{ES}.
- *GA parameters.*: The genetic algorithm has only one parameter: S_{GA}.

5 EVOLUTION RESULT

Here we present an example of finding a Lindenmayer-system that describes a given retina image. In Figure 7 a *SLO* recognized retina image can be seen. We got this image from Albert Szent-Györgyi Medical University, Szeged, Hungary. On the original image several simple image processing algorithms were applied (see in Figure 8), such as edge detection, median filtering, thresholding to get the second image. But also some noise can be seen caused by the background. When background substraction defied by Watson ([Watson et al. 1981]) is applied the image showed in Figure 9 can be generated

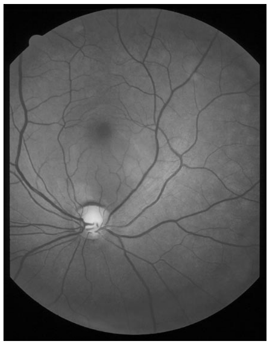

Figure 7: Original image

Since Watson's method is applied on myocardial images, there are several problems with retina images. For example there can be blood vessels, lighter than the background. This is usually caused by occlusion of an artery (*occlusio arteriae*). This and other problems can be eliminated by modifying the algorithm but it is not the topic of the paper therefore we do not go into the detail.

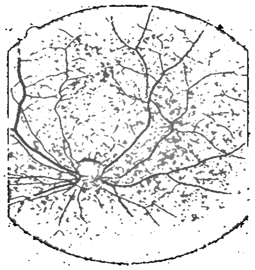

Figure 8: Preprocessed images

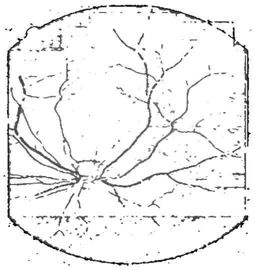

Figure 9: The result of using Watson's method

To demonstrate how the system works and to test it we take only the top-right part of this preprocessed image. It was used as destination image for a randomly generated population of L-systems. After several evolution steps an L-system is generated that describes the blood vessel.

The rules of the above L-system are:

- S→;(2)B(37,0) [B(0,38) B(16,0) [B(0,27) B(11,14) B(12,6) B(19,0) [B(0,47) B(19,0)] B(0,-6) B(17,0)] B(0,-14) B(22,37) B(19,0) [B(0,-6) B(12,34) B(18,32) B(14,0)] B(0,-43) B(24,0)] B(37,0) [B(0,-7) B(8,43) B(24,0)] B(0,-14) B(26,0)
- B(a,b)→ F(a)+(b)

We worked with simple images (continuous lines without crossing) with only one branching part. The size of the population was set to 5, the size of the subpopulations was 10. We examined the best fitness value after 1, 10, 50 and 100 steps. The results can be seen in Table 1. The parameters were set according to the following:

$$S_{ES} = 10 \quad W_1 = 20 \quad W_2 = W_3 = -1 \quad R_{ES} = 1.5$$

$S_{GA}$was not used

evolution steps:	**1**	**10**	**50**	**100**
default	-2569	-2336	-2286	-2285
$S_{ES} = 0$	-2952	-2910	-2835	-2835
$S_{ES} = 20$	-2598	-2288	-2087	-2066
without GA	-2737	-2474	-2308	-2228
$W_1 = 1$	-2883	-2510	-2364	-2364
$W_1 = 20$	-2209	-1879	-1808	-1807
$W_3 = 2$	-4969	-4746	-4698	-4697

Table 1: The Maximal Fitness Value

6 Conclusion

The program *GREDEA* developed by us can be applied in diabetic retinopathy for diabetic retinal disease. It solves the problem of describing the blood vessels of the eye using parametric *Lindenmayer system*s (*L-system*s). To find the most suitable description, evolutionary algorithms are applied.

Some modifications were made:

- To get a better solution, we combined genetic algorithms and evolution strategies.
- To increase convergence speed, special L-Systems including sequences were used.
- To make the algorithm faster, the fitness calculation was also improved.

According to the experimental results one can notify the followings:

- Using ES, the convergence speed is increased. Using more ES steps also increases convergence speed, but it also increases the calculation time.
- Without GA fitness may be acceptable, but there are images that cannot be described without changing the rewriting rules.
- Changing other parameters (like weights) may also influence the convergence speed. (Note that when changing the weights, the bounds of the fitness value are also changed, but still can be used to mesure convergence speed.)

It is among our future plans, to find the appropriate parameters, and make our system suitable for describing more complex retina images.

Later this evolved description can be used to establish the change in a patient's eye: The patient comes again for a checkup and pictures are recorded again. From these, *L-systems* are generated also and the previous grammars can be compared automatically with the new evolved grammars. For the comparison different algorithms may be used, however the detailed description of this process beyond the subject of this paper.

Another possibility of the application is that if the picture generated from the *SLO* was imperfect the incomplete part of the vascular system can be reconstructed using the description grammar.

References

[Abelson, diSessa 1982] Abelson, H., diSessa, A. A.: *Turtle Geometry*, MIT Press, Cambridge, 1982.

[Georgiadis 1995] Georgiadis, C.: *Segmentation of the Vascular Tree in Retinal Images*, Department of Computer Science, University of Crete at Heraklion, Greece, September 1995.

[Goldberg 1989] Goldberg, D.E.: *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, Redwood City, 1989.

[Hesse et al. 1993] Hesse, L., Chofflet, J. and Le Mer, Y.: *Simulating of the growth pattern of the central retinal artery using fractal modeling technique* In German Journal of Opthalmology 2:116-118 1993

[Hipwell 1997] Hipwell, J.: *Digital Angiography in Ophthalmology* Ph.D research at the Department of BioMedical Physics at Aberdeen University, 1997.

[Jacob 1994] Jacob, C.: *Genetic L-system Programming* PPSN III - Parallel Problem Solving from Nature, International Conference on Evolutionary Computation, Lecture Notes in Computer Science 866, Springer-Verlag, Berlin, 1994, pp. 334 - 343.

[Koza 1992] Koza, J.R.: *Genetic Programming: On the Programming of Computers by Means of Natural Selection* The MIT Press, Cambridge Massachusetts, 1992.

[Esperance 1983] L'Esperance, F. A. Jr: *Ophthalmic Lasers: Photocoagulation, Photoradiation and Surgery.* 2nd ed. The C. V. Mosby Company St. Louis. 1983

[Lindenmayer 1968] Lindenmayer, A.: *Mathematical models for cellular interaction in development* In: Journal of Theoretical Biology 18: 280-315 1968.

[Rozenberg, Salomaa 1985] Rozenberg. G., Salomaa A.:*The book of* $\mathcal{L}$ Springer Verlag 1985

[Prusinkiewicz, Hanan 1989] Prusinkiewicz, P., Hanan. J.: *Lindenmayer systems, fractals and plants* Lecture Notes in Biomathemathics vol 79. Springer-Verlag, 1989.

[Rechenberg 1973] Rechenberg, I.: *Evolutionsstrategie '94, Werkstatt Bionik und Evolutionstechnik* Band 1, Fromman-Holzboog, Stuttgart, 1973.

[Watson et al. 1981] Watson, D. D., Campbell, N. P., Read, E. K., et al: *Spatial and temporal quantitation of plane thallium myocardial images* In: Journal of Nuclear Medicine 22:577, 1981.

[Zana 1997] Zana, F.: *Registration of Retinal Images with Occlusion under a Weak Affine Hypothesis, Multimodal and/or Temporal Registration.* In the Proc. The 13th International Conference on Digital Signal Processing (DSP97), on the island of Santorini, Greece, July 2-4, 1997