Design of Image Exploring Agent using Genetic Programming[*]

Mario KÖPPEN, Bertram NICKOLAY

Fraunhofer Institute IPK-Berlin
Pascalstr. 8-9, 10587 Berlin, Germany
Phone: (++49)(0)30 390 06 200 Fax: (++49)(0)30 391 75 17
E-mail: {mario.koeppen|bertram.nickolay}@ipk.fhg.de

Key Word: image processing, genetic programming, agent design, image exploring agent, image understanding, genetic optimization

Abstract

An essential goal of image processing is the detection of image structures like objects, object boundaries, shapes, regions or surface defects. This goal can be reached by a combination of detection algorithms and localization algorithms. Detection algorithms are able to recognize the physical evidence of a structure according to a given localization in the image.
Localization algorithms cope with the position of image structures in images. Most of them are tracking algorithms. Because of the limited recognition abilities the main goal of the foregoing detection algorithm is to enhance an image object. The image structure must be transformed into a kind of information which is suitable for the localization algorithm.
However, in many fields of image processing this approach does not give satisfying results.
This paper introduces a new approach by replacing the conventional localization algorithm by an image exploring agent and by adapting its functional design by means of genetic programming.

1. GENETIC PROGRAMMING

Genetic programming, a new discipline in the field of genetic algorithms, was introduced to overcome some problems concerning the use of conventional genetic algorithms [1]. The main problem addressed was the encoding of the optimization problem into a genom. The problem to solve does not directly take part in the evolutionary optimization. The optimal solution, as it is served by a genetic algorithm, is a genom, that represents the solution. It is not a program that solves the problem. Genetic programming serves with the problem solving program itself as result of the evolutionary optimization. This is achieved by optimizing expression trees rather than optimizing genoms. Standard genetic operators (without the mutation operator) can be used in a similar manner than in conventional genetic algorithms. The crossover operator e.g. is established by the exchange of subtrees of the expression trees that are the operands of the crossover operation (see figure 1). Finally, the optimal expression tree realizes the optimal solution of the problem and also a program for the application of the solution.
This approach offers new possibilities in the field of evolutionary computation. The aspect used in our paper is the possibility to design algorithms by the goal of its application, but not, as usual, by the needs of the optimization procedure.

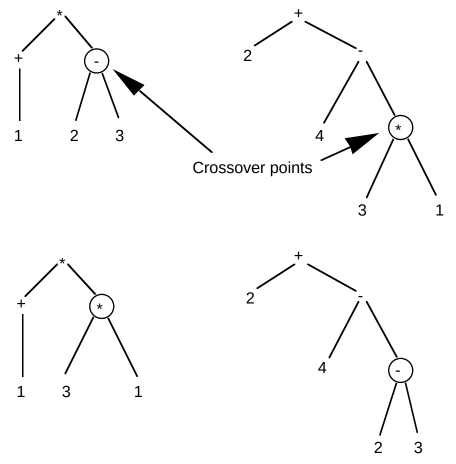

Fig.1 *Crossover of expression trees*

[*] Proceedings of IIZUKA'96, Iizuka, Japan, pp. 549-552

2. IMAGE EXPLORING AGENTS

An image exploring agent is assigned a specified position and an orientation in the image. It consists of two subparts, an evaluating part and a replacing part. The agent works as follows (see figure 2):

Initialization:

Place the agent at a fixed position x_0, y_0 in the image.

Action:

1. Calculate the value of the evaluation function using the values of the image function in a definite neighborhood of the current agent position x,y.
2. Change position and/or orientation of the agent according to the result of 1.
3. Repeat steps 1 and 2 until a given number of steps is reached.

The emphasis here is that the specification of the evaluation function can be adapted by genetic optimization of its expression tree. The fitness value of the evaluation function can be determined by letting the agent follow its program for a fixed number of steps. After that, the agent has completed a track in the image. This track gives the suitability of the agent for the operation goal, e.g. to find a certain position, to meet as many dark greyvalues as possible or to find an object boundary.

3. EXAMPLE: A CRACK DETECTOR

The methodology will be illustrated. The problem to solve was to detect cracks in a greyscale image of a textured surface. The evaluating function of the designed agent is a calculation using the greyvalues in a local, 11x11-neighborhood of the current agent position, that results in a real from [0,1]. If the evaluation result is a value below 0.5, the agent changes its orientation by turning clockwise. If the value is greater than 0.5, the agent goes one step „forward“, i.e. along the current orientation. Care is taken about keeping the agent inside the image boundaries. Also if the agent gets „stuck“ by repeating four turns (it will meet the same situation as at the beginning and will continue turning) the agent is turned to a random orientation and moved forward.

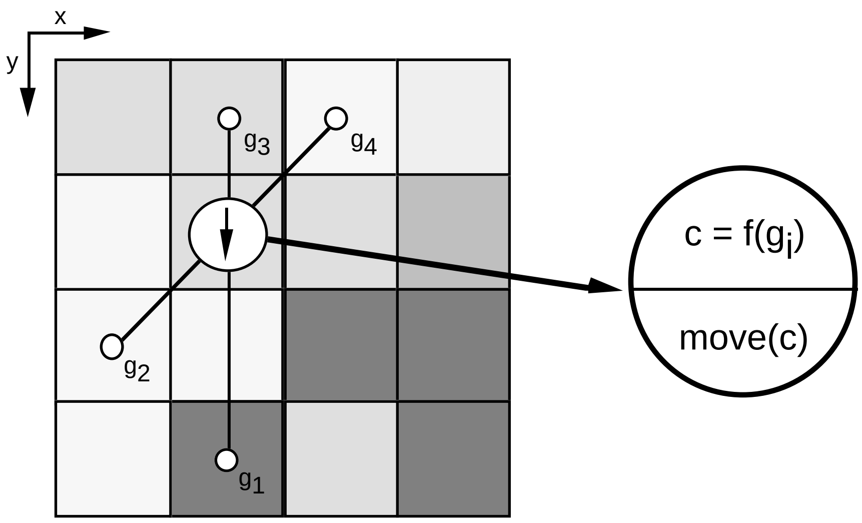

Fig.2 *Functionality of the image exploring agent. Localized at a definite image position it has access to some neighboring greyvalues g_i. The agent calculates a value c from the g_i and changes position and orientation according to the value of c. The function f can be adapted by genetic programming.*

This simple approach will assign a track in the image to every setting of the evaluation function and every choice of a starting point. E.g. if the evaluation function f is of kind:

$$f(x,y) = \frac{g(x,y)}{\max_g}$$

with g(x,y) the greyvalue of the image at (x,y) and $\max_g$ the maximum possible greyvalue, the agents track will try to stay in regions with low greyvalues.

The fitness of the agent was defined by the amount of dark greyvalues met along its track for 200 steps. While performing 200 steps every step was counted with „penalty“ value 1 which fulfilled one of the two following conditions:

1. It was a turn (to force the agent to move through the image).
2. It moved from a position with a greyvalue greater than 100.

The fitness is the ratio of penalty counts to the number of steps. Due to the above conditions the fitness 0.0 is not reachable, but a low value (about 0.3) can be expected. The longer the track followed and the more greyvalues along its track are beyond greyvalue 100 the better the agent. This can be used as a crack finding agent.
For applying genetic programming it is necessary to define terminals and the function set. This is for assigning a meaning to the nodes of the expression tree.
As terminals of the trees neighboring positions of the current agent position were chosen by randomly fixing offset vectors (i,j). Also randomly fixed real constants from [-1,1] were selected. The term „randomly fixed“ here means the values were selected by random number generator if referenced for the first time while initializing the population, after that they remained constant for the following applications of genetic operators.

The function set consisted of standard arithmetic functions modified in a manner to ensure the domain of the operations to be [-1,1]. These functions were:

divide:	$o = \mathrm{sign}(i_1 i_2)\ \min(\lvert i_1/i_2\rvert, \lvert i_2/i_1\rvert)$ if $i_1 i_2 \neq 0$, 1.0 otherwise
harmonic:	$o = 2i_1 i_2/(i_1+i_2)$ if $i_1+i_2 \neq 0$, 1.0 otherwise
ifgte:	$o = i_3$ if $i_1>i_2$, else i_4
iflte:	$o = i_3$ if $i_1<i_2$, else i_4
max:	$o = \max(i_1,i_2)$
min:	$o = \min(i_1,i_2)$
minus:	$o = (i_1-i_2) / 2$
negate:	$o = -i_1$
plus:	$o = (i_1+i_2) / 2$
squared:	$o = i_1^2$
times:	$o = i_1 i_2$

with o the output of the node and i_k, k=1,..,4 resp. the input values of the preceding nodes or terminals.
After that, a randomly initialized population of expression trees was genetically optimized by repeated execution of the following genetic operators:

- Tournament selection
- Crossover
- Fitness proportionate selection

For our experiment a population of 500 expression trees was optimized over 1000 generations. Due to different runs different optimal solutions resulted. The most interesting one was the expression shown in figure 3.
This expression offered a surprising ability to detect dark regions in images. Some runs of this agent are shown in the pictures of figure 4. The design of this expression is very different from what a human would have designed to construct the agent.

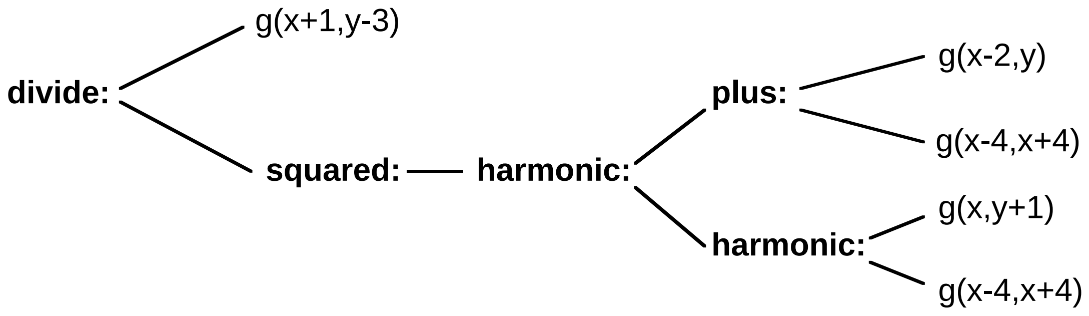

Fig.3 *Best expression of a sample run of genetic programming*

4. SUMMARY

A new approach for the design of image exploring agents by means of genetic programming has been introduced. The approach is based on tracking image positions forced by greyvalue calculations of the local neighborhood. Depending on the result of the calculation a turn or a step forward is performed. The fitness goal of the genetic programming is to find image regions with given properties. An example was

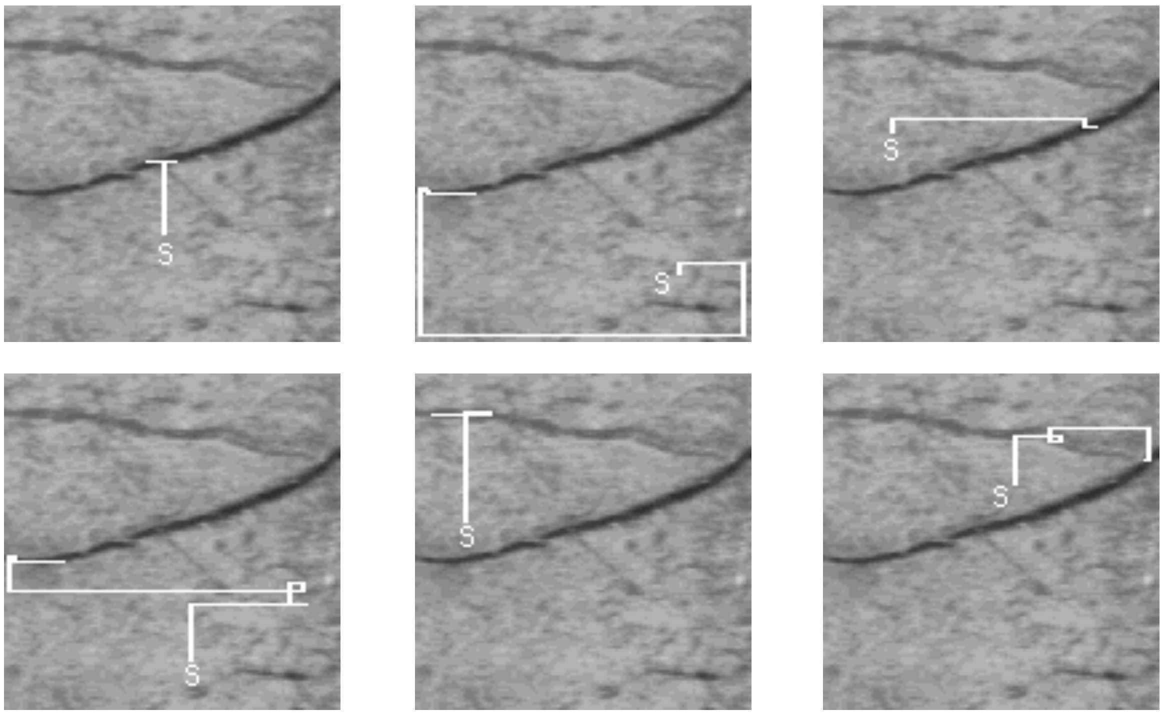

Fig.4 *Track of the agent for performing 1000 steps. The agent starts at a random position S and tries to find dark regions. All image positions it walked over are painted white in the figures.*

given where the goal was to find dark regions of an image which might comprise a crack.
The results are motivating to expand this agent approach to the design of agents with other abilities, e.g. to detect object boundaries, to count objects in images or to find characters and read them.

References

[1] Koza, J.R., „Genetic Programming - On the Programming of Computers by Means of Natural Selection“, MIT Press, Cambridge, London, 1992