Evolving a task specific image operator

Marc Ebner and Andreas Zell

Eberhard-Karls-Universität Tübingen, Wilhelm-Schickard-Institut für Informatik
Arbeitsbereich Rechnerarchitektur, Köstlinstraße 6, 72074 Tübingen, Germany
{ebner,zell}@informatik.uni-tuebingen.de

Abstract. Image processing is usually done by chaining a series of well known image processing operators. Using evolutionary methods this process may be automated. In this paper we address the problem of evolving task specific image processing operators. In general, the quality of the operator depends on the task and the current environment. Using genetic programming we evolved an interest operator which is used to calculate sparse optical flow. To evolve the interest operator we define a series of criteria which need to be optimized. The different criteria are combined into an overall fitness function. Finally, we present experimental results on the evolution of the interest operator.

1 Motivation

A large number of standard image processing operators are available to solve a particular problem. In general, the required operators depend on the current task and environmental conditions. In our work we are trying to evolve image processing operators which perform optimal for the task and the given environmental conditions. To evolve the image operators we have chosen genetic programming [14, 15, 2] because it allows the evolution of hierarchical structures which are often required to solve image processing tasks. The sample problem which we address here is the evolution of an interest operator which is used to compute sparse optical flow. We show how an interest operator can be evolved which is optimal according to multiple criteria which are specific to the application. Before we present our experimental results we briefly discuss related work of using evolutionary methods for image processing tasks.

2 Background

A number of researchers have used evolutionary algorithms for image processing tasks. The methods used range from evolutionary programming [3], structure evolution [18] a variant of an evolution strategy, to genetic algorithms [25, 26, 13, 4]. A growing number of researchers are using genetic programming.

Tackett [30] evolved a symbolic expression for image classification based on image features. Koza [15] and Andre [1] evolved character detectors using genetic programming. Johnson et al. [11] evolved Ullman's visual routines [32] using genetic programming to locate the left and right hand in an image showing the silhouette of a person. Poli [21] applied genetic programming to the task of image segmentation. Daida et al. [5] used genetic programming to extract pressure-ridges from satellite images of arctic sea ice. Harris and Buxton [9] used genetic programming to evolve one-dimensional edge detectors. Poli and Cagnoni [22] evolved algorithms for image enhancement using interactive program evolution. Winkeler and Manjunath [33] used genetic programming for face detection.

Considerable work has been done in the area of feature extraction and tracking. A match between interesting points extracted from an image sequence or from a pair of stereo images can be established easily [36, 34]. Knowledge about point correspondences may be used to establish a three-dimensional model of the world. A number of different methods have been developed to extract interesting points from an image. Moravec [20] developed an interest operator which extracts points with a high variance of pixel values in four directions: horizontal, vertical and both diagonals. Smith [29] developed a corner finder which extracts points where the size of the region belonging to the current pixel in a small neighborhood is a local minimum. Other methods range from using the determinant of the Hesse matrix to find regions of high curvature [27, 19], corner detection [27], difference of Gabor filters [36], detection of symmetry [24, 35] to the use of entropy [12]. Shi and Tomasi [28] argue that good features are those for which the tracker works best. Extracted features (textured regions) are monitored by calculating a dissimilarity measure computed from an affine model of image motion. Features with a high dissimilarity measure should be abandoned. Lew et al. [17] developed an adaptive method for feature selection. From a set of features they select a subset which maximizes the error distance between the correct match and other possible matches.

Genetic programming has so far been rarely used for the construction of image processing operators. Ebner [6] used genetic programming to evolve operators to extract edges from digitized images and evolved an approximation to the Moravec interest operator [7]. In difference to the previous work no existing operator is used for computing the fitness of an evolved individual. In this paper, we only specify the desired properties of the operator and integrate them into a measure of the individual's fitness.

3 Evolving an interest operator

As a sample application we have chosen to evolve an interest operator. The points extracted by the operator are used to calculate sparse optical flow. Optical flow is calculated by establishing corresponding points between the previous image and the current image. It is assumed here that the optical flow can be quite large. This might occur if the camera moves very fast or, equivalently, if delays between subsequent images are long. In this case the calculation of optical flow is simplified by focusing the search only on interesting points in the image. Correspondences are established by comparing the pixels in a small area around the interesting points. The goal is to extract only those points which can be localized accurately in the next image. To achieve this goal we are trying to optimize a number of different properties of the operator. We start by describing the different properties qualitatively which are formalized later. The following properties were used here.

a) The number of established matches should be large. If only a single point is extracted for every image, localizing the point is easy. However, obtaining a dense flow field is usually desirable.
b) The quality of the match should be good. If some error measure describes the difference between pixel values in a small area surrounding the two matched points then this measure should be small.
c) A threshold is used to determine when a match can be considered adequate. Thus a match is not found for every extracted point. Therefore the ratio between matched points and number of extracted points should be high, that is, a match should be established for most of the extracted points. Otherwise it would be possible to extract almost every point and let the search procedure weed out the unnecessary points. However, this is precisely the task the operator should perform.
d) The matches should be unambiguous. For each point all other points are considered as a possible match. Therefore the difference between the error measure of the best match and the second-best match should be large indicating clearly which of the possible matches is the correct one.
e) The optical flow field should be smooth with only a few discontinuities. That is, nearby flow vectors should have approximately the same direction.
f) The density of the flow field may be regulated by introducing a term that tries to achieve a flow field with a maximum density that is distributed over the image.

We now formalize the different optimization criteria. Let $I(t)$ be the image taken at time t. First, the evolved operator is applied to this image. Non-local maxima are suppressed and all points where the pixel value is larger than a threshold ϵ_1 are extracted. Let $F(t)$ be the extracted interesting points of image $I(t)$. Given two images $I(t_1)$ and $I(t_2)$ taken at times t_1 and t_2, respectively, a correspondence between the points in $F(t_1)$ and $F(t_2)$ is established. Given a point $(x_1, y_1) \in F(t_1)$ we calculate the following error measure e for every point $(x_2, y_2) \in F(t_2)$ that is within a specified distance of the original point.

$$e(x_1, y_1, x_2, y_2) = \sqrt{\frac{1}{wh} \sum_{-\frac{w}{2} \leq i < \frac{w}{2}} \sum_{-\frac{h}{2} \leq j < \frac{h}{2}} \left(\tilde{I}(x_1 + i, y_1 + j) - \tilde{I}(x_2 + i, y_2 + j)\right)^2} \tag{1}$$

where $\tilde{I}(t)$ is obtained by smoothing image $I(t)$ with a Gaussian filter and w and h specify the width and height of the patch which is used to calculate the error measure. The point (x_2, y_2) for which the error measure is minimal is chosen as the corresponding point. In addition a threshold is used to reject bad matches. Therefore a match is only established provided that e is less then a threshold ϵ_2. Let $F_m(t)$ be the points for which a match could be established. Let n_p be the number of points in $F(t_1)$ and let n_m be the number of points for which a match could be established. Then the following measures of operator quality were used for our experiments.

a) Number of matches:

$$m_1 = n_m \tag{2}$$

b) Quality of matches:

$$m_2 = \frac{1}{n_m} \sum_{(x,y)\in F_m(t_1)} \frac{1}{1+e_{\min}(x,y)} \tag{3}$$

where $e_{\min}(x_1, y_1) = \min_{(x,y)\in F(t_2)} e(x_1, y_1, x, y)$ is the minimum of the error measure e. The measure m_2 is analogous to Pratt's figure of merit which is used to judge the performance of edge detectors [10].

c) Match percentage:

$$m_3 = \frac{n_m}{n_p} \tag{4}$$

d) Match ambiguity:

$$m_4 = \frac{1}{n_m} \sum_{(x,y)\in F_m(t_1)} \frac{e_{\text{next}}(x,y) - e_{\min}(x,y)}{e_{\max}(x,y) - e_{\min}(x,y)} \tag{5}$$

where $e_{\max}(x_1, y_1) = \max_{(x,y)\in F(t_2)} e(x_1, y_1, x, y)$ is the maximum of the error measure e. Let (x_m, y_m) be the point for which the error measure is minimal. Then the value of the error measure for the second-best match is defined as $e_{\text{next}}(x_1, y_1) = \min_{(x,y)\in F(t_2)\setminus(x_m,y_m)} e(x_1, y_1, x, y)$.

e) Flow smoothness:

$$m_5 = \frac{1}{n_p} \sum_{(x,y)\in F(t_1)} s(x,y) \tag{6}$$

where s is a smoothness measure calculated for a small neighborhood around the point. Let $F_{N(x,y)}$ be the points inside the neighborhood of point (x, y).

$$F_{N(x,y)}(t) = \{(x', y') \in F(t) | \sqrt{(x'-x)^2 + (y'-y)^2} < \epsilon_3\} \tag{7}$$

Then the smoothness measure is calculated as

$$s(x,y) = \frac{1}{2|F_{N(x,y)}(t_1)|} \sum_{(x',y')\in F_{N(x,y)}(t_1)} 1 + \frac{\Delta x \Delta x' + \Delta y \Delta y'}{\sqrt{\Delta x^2 + \Delta y^2}\sqrt{\Delta x'^2 + \Delta y'^2}} \tag{8}$$

where $(\Delta x, \Delta y)$ is the computed optical flow of point (x, y).

f) Maximum flow field density:

$$m_6 = \frac{1}{n_p} \sum_{(x,y)\in F(t_1)} \min\left\{\frac{d_{\min}(x,y)}{d_{\text{des}}}, 1.0\right\} \tag{9}$$

where $d_{\min}(x, y)$ is the distance in pixels between point (x, y) and its nearest point and d_{des} is the desired minimum distance between the extracted points.

4 Using genetic programming to evolve image operators

Genetic programming is especially suited to combine simple elementary functions into a complex hierarchical image processing operator. To apply genetic programming to the evolution of an image processing operator we have to define the set of terminal symbols, the set of primitive functions and a suitable fitness measure. We now describe each of these in turn.

4.1 Terminal symbols

The input image I was our only terminal symbol. The pixel values were normalized to the range $[0, 1]$.

4.2 Primitive functions

As primitive functions we used the following set of unary and binary functions. Let I_R be the image that results from the application of a primitive function to an input image I in the case of an unary function and two input images I_1 and I_2 in the case of a binary function. Image coordinates are denoted with x and y. Unary primitive functions:

- Square root (`Sqrt`): $I_R(x,y) = \sqrt{|I(x,y)|}$
- Square (`Square`): $I_R(x,y) = I(x,y) \cdot I(x,y)$
- Gabor filters (`Gabor0`,...,`Gabor7`):
  $I_R(x,y) = |\int \Psi(x',y',f,\theta_j) I(x-x', y-y') dx'dy'|$
  with $\Psi(x,y,f,\theta) = \exp(i(fx\cos\theta + fy\sin\theta) - \frac{f^2(x^2+y^2)}{2\sigma^2})$,
  $\sigma = \pi$, $f = \frac{\pi}{2}$ and $\theta_j = \frac{\pi j}{8}$ with $j \in \{0, ..., 7\}$ (as defined in [16]).
- Average (`Avg3x3`): $I_R(x,y) = \frac{1}{9}\sum_{-1 \le i,j \le 1} I(x+i, y+j)$
- Median filter (`Median3x3`): $I_R(x,y) = \text{Median}\{I(x+i, y+j) | -1 \le i,j \le 1\}$
- Gaussian filter (`Gauss`):
  $I_R(x,y) = \int e^{\frac{x'^2+y'^2}{2\sigma^2}} I(x-x', x-y') dx'dy'$ with $\sigma = 1.0$.
- Derivative of Gaussian in x direction (`GaussDx`):
  $I_R(x,y) = \frac{1}{\sqrt{2\pi\sigma^3}} \int x e^{-\frac{1}{2\sigma^2}(x'^2+y'^2)} I(x-x', y-y') dx'dy'$ with $\sigma = 1.0$.
- Derivative of Gaussian in y direction (`GaussDy`):
  $I_R(x,y) = \frac{1}{\sqrt{2\pi\sigma^3}} \int y e^{-\frac{1}{2\sigma^2}(x'^2+y'^2)} I(x-x', y-y') dx'dy'$ with $\sigma = 1.0$.

Binary primitive functions:

- Addition (`+`): $I_R(x,y) = I_1(x,y) + I_2(x,y)$
- Subtraction (`-`): $I_R(x,y) = I_1(x,y) - I_2(x,y)$
- Multiplication (`*`): $I_R(x,y) = I_1(x,y) \cdot I_2(x,y)$
- Protected division (`/`): $I_R(x,y) = \begin{cases} 1 & \text{if} \quad I_2(x,y) = 0 \\ I_1(x,y)/I_2(x,y) & \text{otherwise} \end{cases}$

Figure 1 shows how some of the primitive functions could be used to build an operator which calculates the determinant of the Hesse matrix.

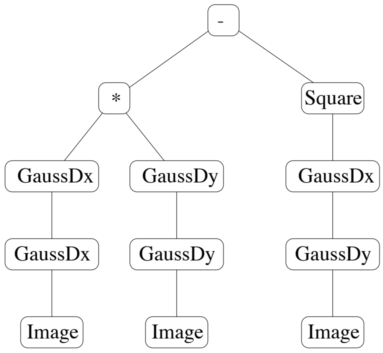


Fig. 1. Example of an existing operator which was manually constructed from the set of primitive functions.

4.3 Fitness measure

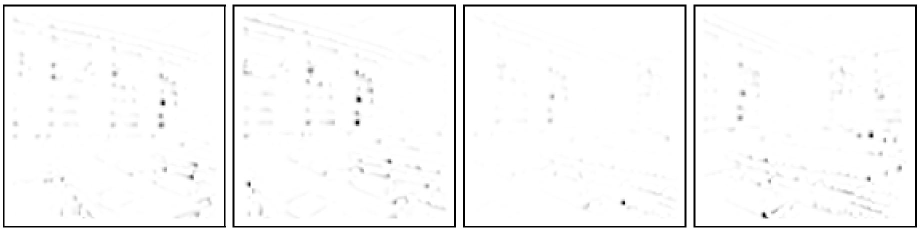
The different criteria have to be integrated into one fitness measure. We have to do multi-objective optimization to evolve a detector which is optimal according to all of the criteria. An overview about multi-objective optimization is given by Fonseca and Fleming [8]. To integrate the different measures into one we calculate the average of each measure over all fitness cases. Let $\bar{m}(i)$ be the average of the measure m for the individual i. Next, we normalize them across all individuals in the population. This gives us a selection probability $p_c(i) = \frac{\bar{m}_c(i)}{\sum_j \bar{m}_c(j)}$ for each criterion c and individual i. The selection probabilities were combined into a single fitness function $\mathrm{f} = \prod_i p_i$. The combined fitness reaches its maximum value only if all of the different selection probabilities have a large value. The normalization is not necessary for the multiplicative contribution of the different measures. We normalize them, because in other experiments an additive contribution was used.

Name of operator	$\bar{m}_1$	$\bar{m}_2$	$\bar{m}_3$	$\bar{m}_4$	$\bar{m}_5$	$\bar{m}_6$	Absolute fitness
Kitchen-Rosenfeld [27]	54.33	0.9850	0.4669	0.6710	0.9294	0.6154	9.592
Det(H_I) [27, 19]	51.67	0.9853	0.5386	0.7396	0.9635	0.6568	12.83
Moravec [20]	49.33	0.9848	0.5242	0.6912	0.9359	0.7293	12.01
SUSAN [29]	77.00	0.9867	0.5426	0.5318	0.9195	0.6042	12.18
Diff. of Gabor filters[36]	64.00	0.9865	0.5967	0.5803	0.9348	0.6891	14.08
Evolved	131.0	0.9871	0.7814	0.5504	0.9170	0.6620	33.77

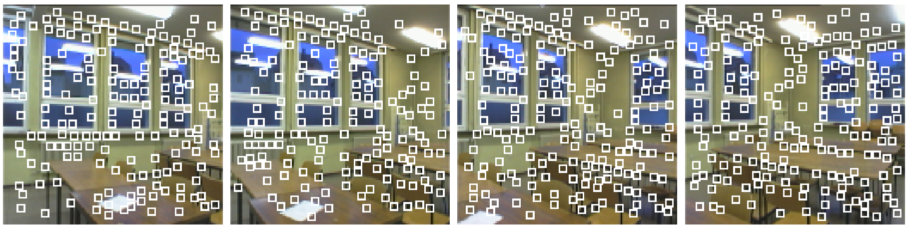
Table 1. Comparison between different interest operators and the evolved interest operator. Absolute fitness is computed as fitness $= \Pi_i \bar{m}_i$ which is used as an absolute measure to compare the different operators.

Fig. 2. Image sequence which was used during evolution.

Response of the operator:



Extracted points:



Sparse optical flow:

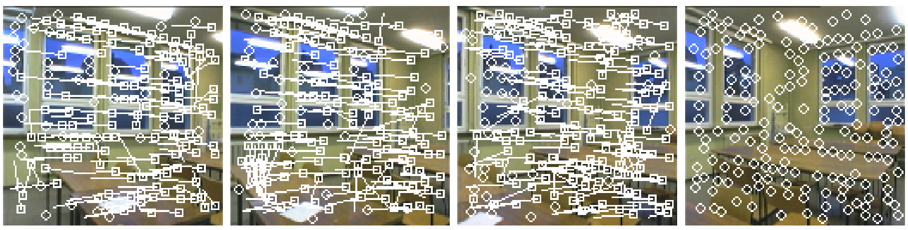
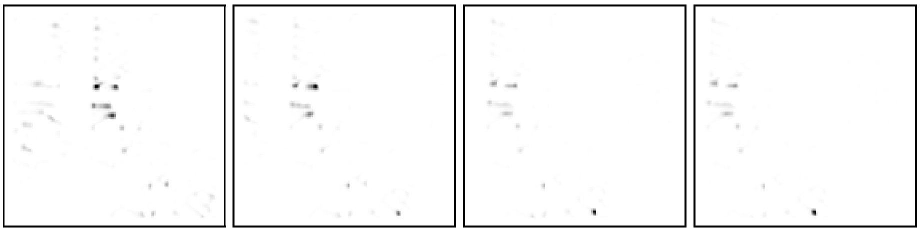


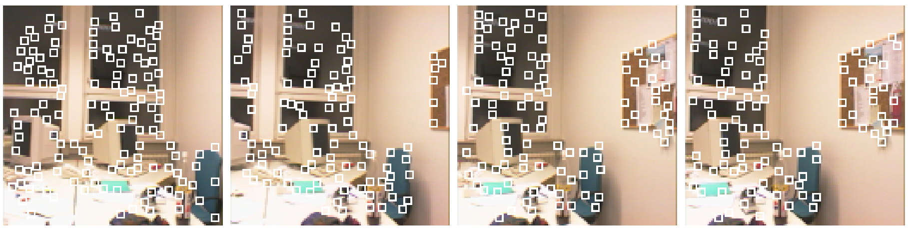
Fig. 3. Best individual from generation 50. The first row shows the response of the evolved operator. The second row shows the extracted interesting points. The third row shows the computed sparse optical flow.

Fig. 4. Image sequence which was used to test the evolved operator.

Response of the operator:



Extracted points:



Sparse optical flow:

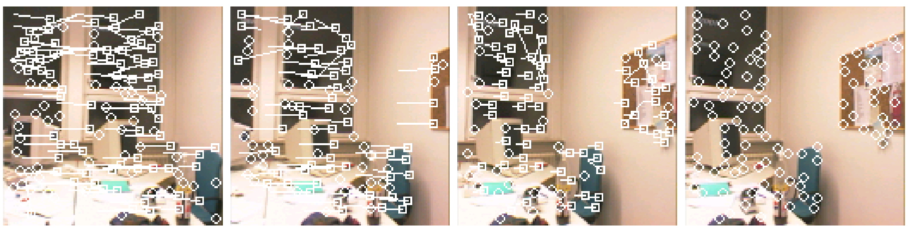
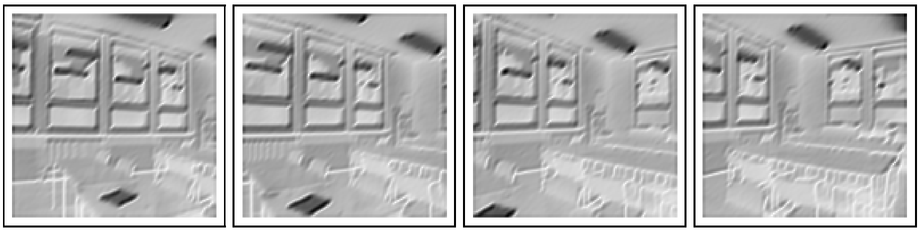
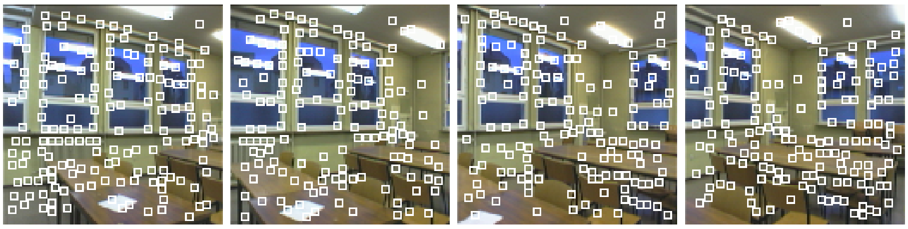


Fig. 5. Results of the evolved operator on a test sequence.

Response of the operator:



Extracted points:



Sparse optical flow:

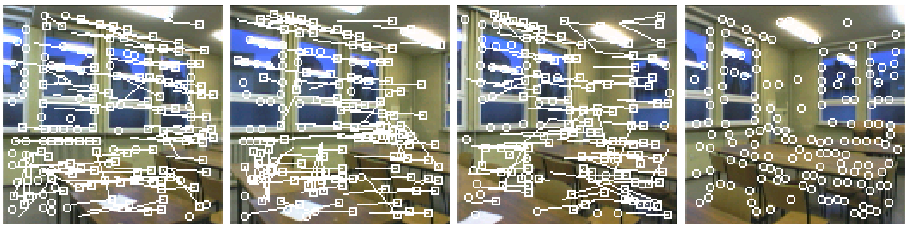
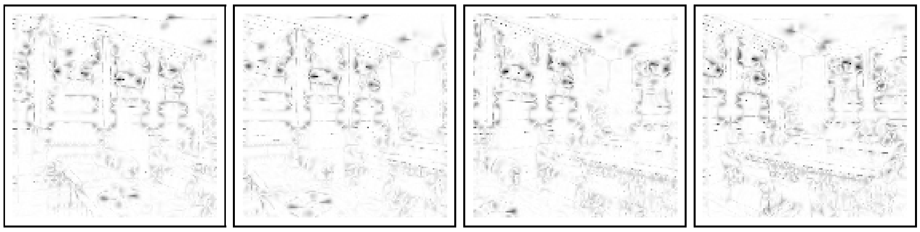


Fig. 6. Best individual from the first generation.

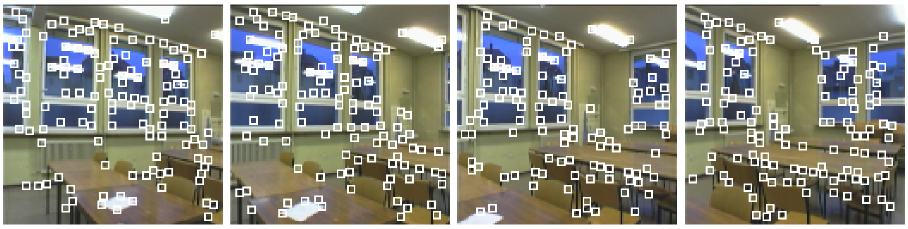
4.4 Results

With the above representation we evolved an interest operator. We used a sequence of 4 images with sizes 128×128 shown in Figure 2. The major parameters of the run were as follows. We used a population size of 500 individuals. The experiment was run for 50 generations. A limit of 1000 nodes and a maximum possible depth of the trees of 17 was used. Tournament selection with size 7 was used and crossover, reproduction and mutation probabilities were set to 85%, 10% and 5%, respectively. The results of the experiment are displayed in Figure 3. The first row shows the response of the best evolved operator from generation 50. The second row shows the extracted interesting points and the third row shows the computed sparse optical flow. The evolved operator was tested on an additional image sequence shown in Figure 4. The results achieved with the evolved operator on the test sequence is shown in Figure 5. The response of the best operator which was found in the first generation of the experiment applies the `Gabor2` operator twice. It extracts edges which are oriented in direction $\frac{\pi}{4}$

Response of the operator:



Extracted points:



Sparse optical flow:

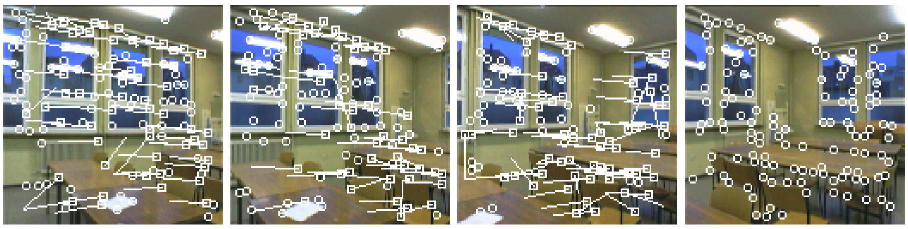
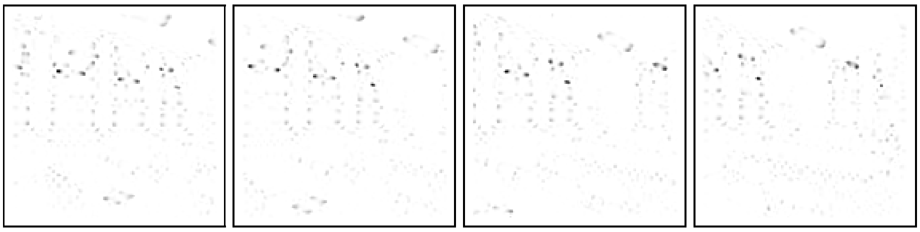


Fig. 7. Results achieved with the Kitchen-Rosenfeld corner detector [27].

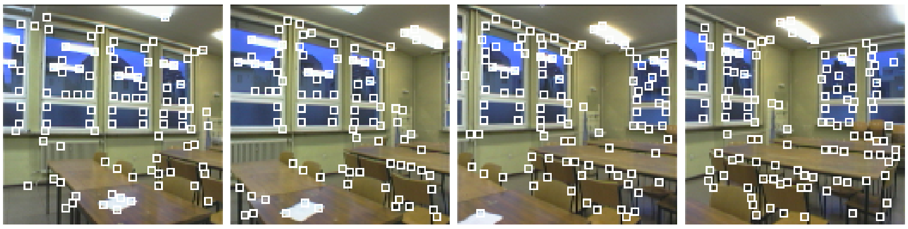
(Figure 6). Table 1 shows the performance of the evolved operator in comparison to the Kitchen-Rosenfeld corner detector [27], the determinant of the Hesse matrix [27,19], the Moravec operator [20], the SUSAN operator [29], and the difference of Gabor filters [36]. The results of these operators are shown in Figure 7, Figure 8, Figure 9, Figure 10 and Figure 11 respectively. For some quality measures the evolved operator performed better than the other operators whereas for others it performed worse. Selection, however, is done according to the overall fitness. The evolved operator clearly outperformed the existing operators in terms of the overall fitness.

As can be seen the evolved operator highlights regions in the image that are of particular interest for the calculation of sparse optical flow. Some wrong matches are also produced. This is due to the fact that the operator combines different possibly contradicting measures. For instance the number of points extracted should be high and at the same time the matches should be unambiguous. Fitness statistics for the experiment can be found in Figure 12. Except for the Gaussian filter all of the available functions occurred in the evolved individual. The division

Response of the operator:



Extracted points:



Sparse optical flow:

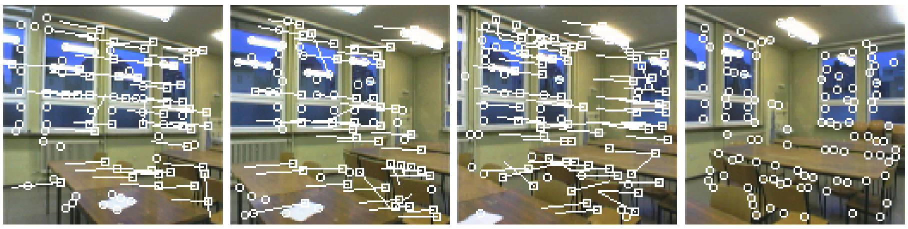


Fig. 8. Results achieved with the determinant of the Hesse matrix [27, 19].

operation, derivative of the Gaussian in y direction, average, Gabor filters, the square root and the square function were used several times.

5 Conclusion

We have shown that task specific image operators may be evolved using genetic programming. Different criteria are used to evolve operators which are optimal for the task at hand. As a sample task we evolved an interest operator for the computation of sparse optical flow. The following criteria were used to evolve the interest operator. a) A large number of matches should be produced. b) The quality of the matches should be good. c) The relation of matched points to unmatched points should be high. d) Matches should be unambiguous, e) the flow field should be rather smooth and f) have a maximum density. These criteria led to the evolution of an operator which can be used to extract interesting points from an image.

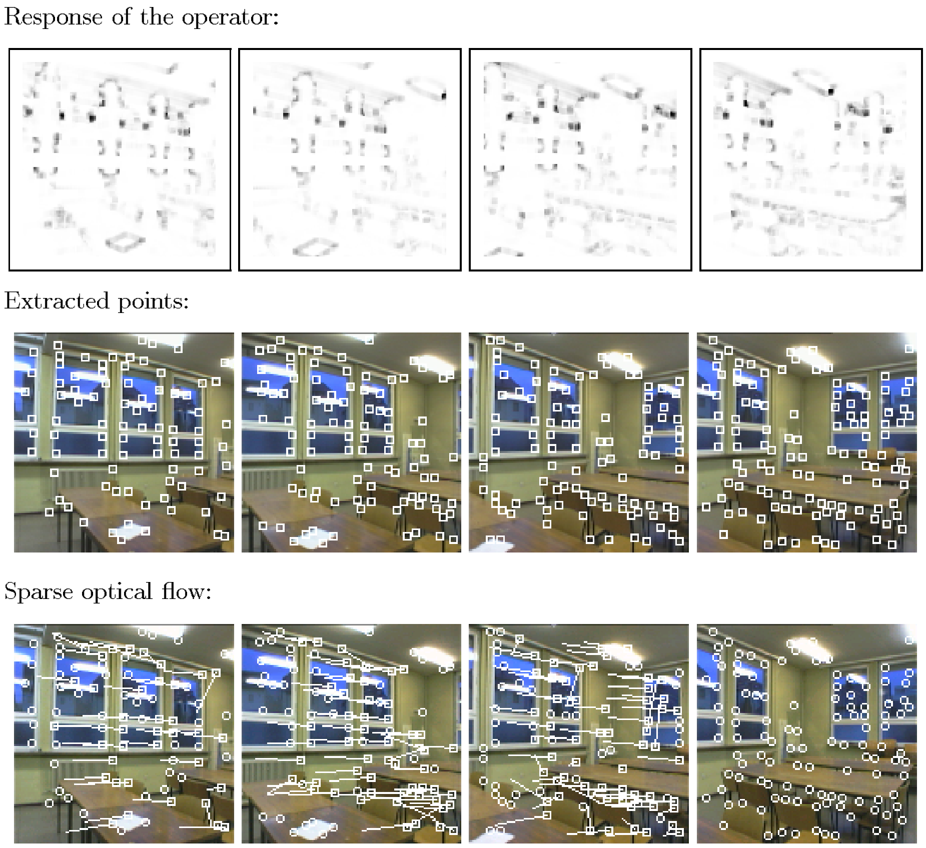


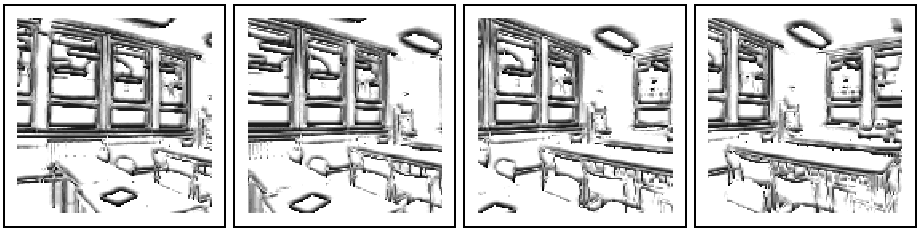
Fig. 9. Results achieved with the Moravec operator [20].

Provided that the fitness evaluation can be done fast enough it might be possible to construct adaptive vision systems which are able to adapt themselves to changing environmental conditions. Just as the pupil's diameter adapts to changing brightness conditions [31] an artificial visual system might evolve optimal or near optimal image processing operators on the fly. At present, however, the evolution is performed offline and evolution of an operator from scratch takes several days to complete on a single PC.

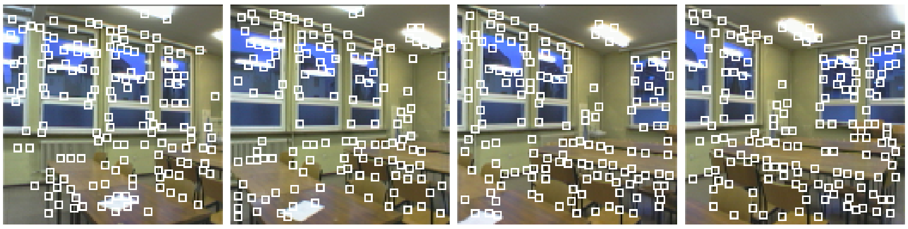
6 Acknowledgements

This work was supported in part by a scholarship to the first author according to the Landesgraduiertenförderungsgesetz. For our experiments we used the lil-gp Programming System [37]. For image processing we used the Vista software environment [23].

Response of the operator:



Extracted points:



Sparse optical flow:

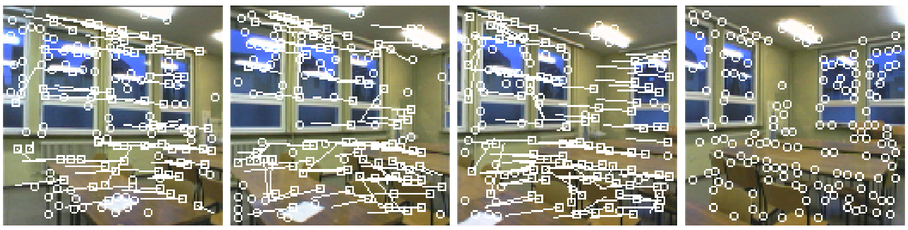
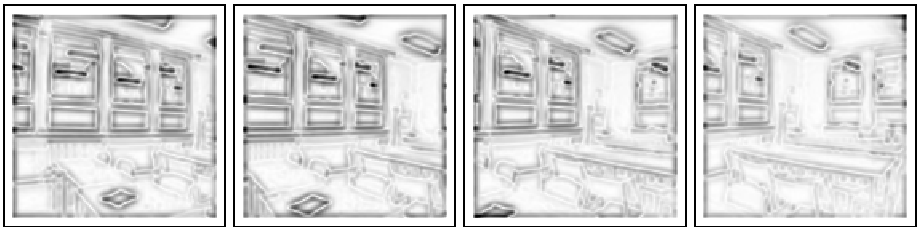


Fig. 10. Results achieved with the SUSAN operator [29].

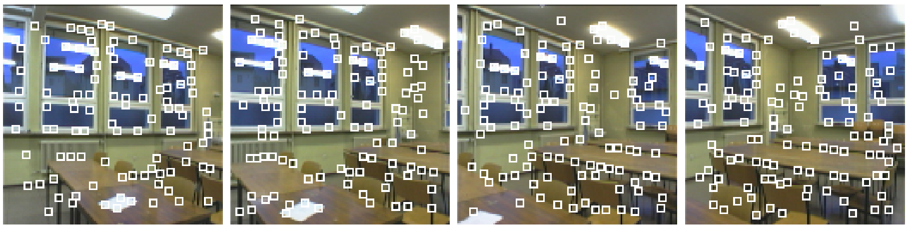
References

1. D. Andre. Automatically defined features: The simultaneous evolution of 2-dimensional feature detectors and an algorithm for using them. In K. E. Kinnear Jr., editor, *Advances in Genetic Programming*, pp. 477–494, Cambridge, Massachusetts, 1994. The MIT Press.
2. W. Banzhaf, P. Nordin, R. E. Keller, and F. D. Francone. *Genetic Programming - An Introduction: On The Automatic Evolution of Computer Programs and Its Applications*. Morgan Kaufmann Publishers, San Francisco, California, 1998.
3. A. K. Bhattacharjya and B. Roysam. Joint solution of low, intermediate, and high-level vision tasks by evolutionary optimization: Application to computer vision at low SNR. *IEEE Transactions on Neural Networks*, 5(1):83–95, January 1994.
4. R. R. Brooks, S. S. Iyengar, and J. Chen. Automatic correlation and calibration of noisy sensor readings using elite genetic algorithms. *Artificial Intelligence*, 84:339–354, 1996.
5. J. M. Daida, J. D. Hommes, T. F. Bersano-Begey, S. J. Ross, and J. F. Vesecky. Algorithm discovery using the genetic programming paradigm: Extracting low-contrast curvilinear features from sar images of arctic ice. In P. J. Angeline and K.

Response of the operator:



Extracted points:



Sparse optical flow:

Fig. 11. Results achieved with the difference of Gabor filters [36].

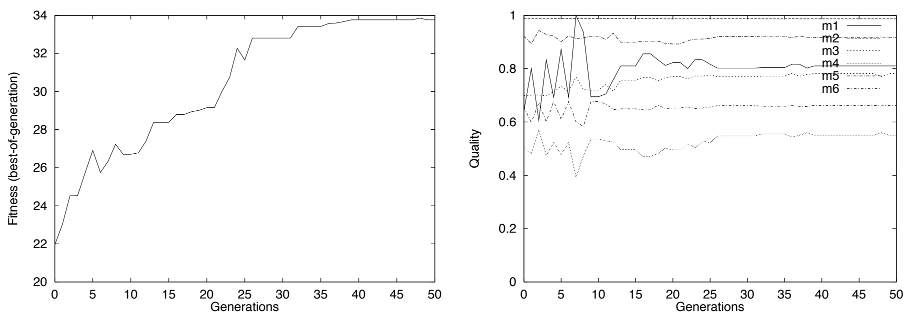


Fig. 12. The absolute fitness (best of generation) is shown on the left. Absolute fitness is calculated as fitness $= \Pi_i \bar{m}_i$. The different quality measures which belong to the individual with the highest fitness are shown on the right. The first quality measure (number of matches) was normalized to the range of [0,1] to integrate the measure into the same diagram.

E. Kinnear, Jr., editors, *Advances in Genetic Programming Volume II*, pp. 417–442, Cambridge, Massachusetts, 1996. The MIT Press.
6. M. Ebner. On the evolution of edge detectors for robot vision using genetic programming. In H.-M. Groß, editor, *Workshop SOAVE '97 - Selbstorganisation von Adaptivem Verhalten, VDI Reihe 8 Nr. 663*, pp. 127–134, 1997. VDI Verlag.
7. M. Ebner. On the evolution of interest operators using genetic programming. In R. Poli, W. B. Langdon, M. Schoenauer, T. Fogarty, and W. Banzhaf, editors, *Late Breaking Papers at EuroGP'98: the First European Workshop on Genetic Programming*, pp. 6–10, Paris, France, 1998. The University of Birmingham, UK.
8. C. M. Fonseca and P. J. Fleming. An overview of evolutionary algorithms in multiobjective optimization. *Evolutionary Computation*, 3(1):1–16, 1995.
9. C. Harris and B. Buxton. Evolving edge detectors with genetic programming. In J. R. Koza, D. E. Goldberg, D. B. Fogel, and R. L. Riolo, editors, *Genetic Programming 1996, Proceedings of the First Annual Conference*, pp. 309–314, Cambridge, Massachusetts, 1996. The MIT Press.
10. R. Jain, R. Kasturi, and B. G. Schunck. *Machine Vision*. McGraw-Hill, NY, 1995.
11. M. P. Johnson, P. Maes, and T. Darrell. Evolving visual routines. In R. A. Brooks and P. Maes, editors, *Artificial Life IV, Proceedings of the Fourth International Workshop on the Synthesis and Simulation of Living Systems*, pp. 198–209, Cambridge, Massachusetts, 1994. The MIT Press.
12. T. Kalinke and W. von Seelen. Entropie als Maß des lokalen Informationsgehalts in Bildern zur Realisierung einer Aufmerksamkeitssteuerung. In B. Jähne, P. Geißler, H. Haußecker, and F. Hering, editors, *Mustererkennung 1996, 18. DAGM-Symposium, Heidelberg*, pp. 627–634, Berlin, 1996. Springer-Verlag.
13. A. J. Katz and P. R. Thrift. Generating image filters for target recognition by genetic learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(9):906–910, September 1994.
14. J. R. Koza. *Genetic Programming, On the Programming of Computers by Means of Natural Selection*. The MIT Press, Cambridge, Massachusetts, 1992.
15. J. R. Koza. *Genetic Programming II, Automatic Discovery of Reusable Programs*. The MIT Press, Cambridge, Massachusetts, 1994.
16. J. Lampinen and E. Oja. Distortion tolerant pattern recognition based on self-organizing feature extraction. *IEEE Transactions on Neural Networks*, 6(3):539–547, May 1995.
17. M. S. Lew, T. S. Huang, and K. Wong. Learning and feature selection in stereo matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(9):869–881, September 1994.
18. R. Lohmann. Selforganization by evolution strategy in visual systems. In H.-M. Voigt, H. Mühlenbein, and H.-P. Schwefel, editors, *Evolution and Optimization '89*, pp. 61–68. Akademie-Verlag, 1990.
19. H. A. Mallot. *Sehen und die Verarbeitung visueller Information, Eine Einführung*. Vieweg, Braunschweig, 1998.
20. H. P. Moravec. Towards automatic visual obstacle avoidance. In *Proc. of the 5th International Joint Conference on Artificial Intelligence*, Vision–1: p. 584, 1977.
21. Riccardo Poli. Genetic programming for image analysis. In J. R. Koza, D. E. Goldberg, D. B. Fogel, and R. L. Riolo, editors, *Genetic Programming 1996, Proceedings of the First Annual Conference*, pp. 363–368, Cambridge, Massachusetts, 1996. The MIT Press.
22. R. Poli and S. Cagnoni. Genetic programming with user-driven selection: Experiments on the evolution of algorithms for image enhancement. In J. R. Koza, K.

Deb, M. Dorigo, D. B. Fogel, D. B. Fogel, M. Garzon, H. Iba, and R. L. Riolo, editors, *Genetic Programming 1997, Proceedings of the Second Annual Conference*, pp. 269–277, 1996. Morgan Kaufmann Publishers.

23. A. R. Pope and D. G. Lowe. Vista: A software environment for computer vision research. In *Proceedings of the 1994 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 768–772. IEEE, 1994.
24. D. Reisfeld, H. Wolfson, and Y. Yeshurun. Detection of interest points using symmetry. In *Proceedings of the International Conference on Computer Vision, Osaka, Japan*, pp. 62–65. IEEE, December 1990.
25. M. M. Rizki, L. A. Tamburino, and M. A. Zmuda. Evolving multi-resolution feature-detectors. In D. B. Fogel and W. Atmar, editors, *Proceedings of the Second American Conference on Evolutionary Programming*, pp. 108–118. Evolutionary Programming Society, 1993.
26. G. Roth and M. D. Levine. Geometric primitive extraction using a genetic algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(9):901–905, September 1994.
27. M. A. Shah and R. Jain. Detecting time-varying corners. *Computer Vision, Graphics, and Image Processing*, 28:345–355, 1984.
28. J. Shi and C. Tomasi. Good features to track. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 593–600, 1994.
29. S. Smith. A new class of corner finder. In *Proceedings of the 3rd British Machine Vision Conference 1992*, pp. 139–148, 1992.
30. W. A. Tackett. Genetic programming for feature discovery and image discrimination. In S. Forrest, editor, *Proceedings of the Fifth International Conerence on Genetic Algorithms*, pp. 303–309. Morgan Kaufmann, 1993.
31. M. J. Tovée. *An introduction to the visual system.* Cambridge University Press, Cambridge, 1996.
32. S. Ullman. Visual routines. In M. A. Fischler and O. Firschein, editors, *Readings in Computer Vision: Issues, Problems, Principles, and Paradigms*, pp. 298–328, Los Altos, California, 1987. Morgan Kaufmann Publishers.
33. J. F. Winkeler and B. S. Manjunath. Genetic programming for object detection. In J. R. Koza, K. Deb, M. Dorigo, D. B. Fogel, D. B. Fogel, M. Garzon, H. Iba, and R. L. Riolo, editors, *Genetic Programming 1997, Proceedings of the Second Annual Conference*, pp. 330–335, San Francisco, California, 1997. Morgan Kaufmann Publishers.
34. M. Xie. Automatic feature matching in uncalibrated stereo vision through the use of color. *Robotics and Autonomous Systems*, 21:355–364, 1997.
35. H. Zabrodsky, S. Peleg, and D. Avnir. Symmertry as a continuous feature. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(12):1154–1165, 1995.
36. Q. Zheng and R. Chellappa. Automatic feature point extraction and tracking in image sequences for arbitrary camera motion. *International Journal of Computer Vision*, 15:31–76, 1995.
37. D. Zongker and B. Punch. *lil-gp 1.01 User's Manual (support and enhancements Bill Rand).* Michigan State University, March 1996.