

Automatic Construction of Invariant Features Using Genetic Programming for Edge Detection

Wenlong Fu¹, Mark Johnston¹, and Mengjie Zhang²

¹ School of Mathematics, Statistics and Operations Research

² School of Engineering and Computer Science

Victoria University of Wellington, PO Box 600, Wellington, New Zealand

Abstract. This paper investigates automatic construction of invariant features using Genetic Programming (GP) for edge detection. Generally, basic features for edge detection, such as gradients, are further manipulated to improve detection performance. In order to improve detection performance, new features are constructed from different local features. In this study, GP is proposed to automatically construct invariant features based on basic invariant features from gradients, image quality (means and standard deviations), and histograms of images. The experimental results show that the invariant features constructed by GP combine advantages from the basic features, reduce drawbacks from basic features alone, and also improve the detection performance.

Keywords: Genetic Programming, Edge Detection, Image Analysis, Feature Construction.

1 Introduction

Edge detection is a well developed area of image analysis, but it is a subjective task [16,19]. Features in edge detection are functions of raw pixel values in an image relative to a local point and are used in the process of classifying pixels as edge points or not. Since there are no formulae to definitely describe the problem of edge detection, various approaches have been developed to extract features for detecting edges [16,4,19].

Generally, one feature for edge detection is not sufficient to fully identify the edges in an image, and multiple features are useful to improve detection performance. For instance, features based on image gradients are not good to detect texture edges [1,16,19]. Since different advantages exist in different features [19], a combination of features may possibly bring the advantages of each basic feature together. Two ways of combining a set of features for edge detection are: (1) to combine different edge detectors to construct features using a fixed model, such as a logistic regression model [16] or the combination of voting consensus ground truth based on a set of features [6]; or (2) to discriminate edge points based on different responses on different parameters in a method, such as multi-scale approaches [1,18]. From these methods, a set of features can improve detection performance [1,16,18,19], but, e.g., the performance of Boosted Edge Learning

using approximately 50000 features for natural images is only close to a contour detector Pb proposed in [16] with nine local features [4]. In [16], different learned models present very similar detection performance. Therefore, how to efficiently and effectively combine features still needs to be investigated.

Genetic Programming (GP) has been employed for edge detection since at least 1996 [20,12]. In our previous work, GP is employed to evolve low-level edge detectors (based on raw pixels selected automatically with their graylevels) [8,9]. The existing work for constructing edge detectors mainly focuses on low-level edge detectors via choosing raw pixels [24], or a combination of image operators [14]. The work on GP for edge detection for constructing low-level edge detectors shows that GP can evolve good edge detectors [24,14,8], and it is promising to use GP for automatically constructing features for edge detection based on existing features.

Goals. The overall goal of this paper is to investigate automatically constructing invariant features for edge detection using GP from a set of basic features so that the constructed features improve the detection accuracy. Here, invariant features mean that they are not affected by image rotation. The image gradients and histogram gradients [16] are popularly used to train learning contour detectors, so they are used as basic features. In order to enrich the set of features for edge detection, a measure of image quality is also used as a basic feature; it is the first time to use this measure for edge detection. Based on these three basic features, composite features will be constructed automatically by GP. Since invariant features can be directly evaluated, this study is an initial investigation on the construction of invariant features. Specifically, we would like to investigate the following research objectives.

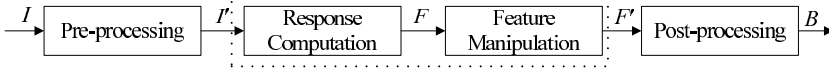
- Whether the features constructed by GP can improve the detection performance, compared with each basic invariant feature alone.
- Whether the constructed features are better than the features constructed by a simple Bayes model.
- What differences between the features constructed by GP and the basic features exist, from an analysis of the characteristics of the detected images.

In the remainder of the paper, Section 2 briefly describes some relevant background. Section 3 presents how GP can be used to construct invariant features for edge detection. After presenting the experimental design in Section 4, Section 5 describes the results with discussion. Section 6 gives conclusions and future work directions.

2 Background

2.1 Edge Detection

Edge detection usually contains three stages: pre-processing, feature extraction and post-processing [17,19]. The pre-processing stage mainly focuses on noise reduction and texture suppression while preserving edges and not blurring boundaries between different areas. The feature extraction stage is the main and necessary stage in edge detection. The purpose of extracting features is to use them

**Fig. 1.** General edge detection flow

to classify pixels as edge points or not. Post-processing mainly focuses on marking edge points, thinning edges and linking broken edges points. In general, the post-processing techniques can usefully follow most feature extraction approaches. Therefore, the estimation of edge responses is sensitive to the detection performance. The evaluation of the features in edge detection is also important [17] although the performance evaluation usually focuses on final edge maps.

Feature extraction mainly works on local features for the sake of simplicity and ease of implementation. Local features mainly come from gradient computation, such as image gradient [3,11] or local histogram gradient [16]. Based on the difference from different directions, local features are categorised as invariant features and variant features. Variant features are related to a direction, such as the horizontal and vertical derivatives. Invariant features are not affected when the image rotates, such as the outputs of the Laplacian edge detector. However, the local features from gradient computation contain high responses on non-edge points affected by noise or textures. Techniques for manipulating these local features are useful to improve the detection performance. For instance, surround suppression can reduce some texture responses on image gradients [11]. In [11], the features based on gradients and differences of Gaussian filters are used to construct a new feature that suppresses texture responses. Therefore, construction of features from local features has shown promise in improving the detection performance.

Figure 1 shows a general edge detection process flow. For an image I , an intermediate result I' will be obtained after pre-processing. The feature extraction stage is divided into two phases, namely response computation and feature manipulation. In the response phase, the computation can come from gradients, and also statistics [15], and a set of features F is obtained. Note that, some edge detectors combine the pre-processing and response computation together, such as the image gradients after filtering noise by a Gaussian filter in the Canny detector [3]. In the feature manipulation phase, feature selection [4] and further feature construction [16,11] are included, and the output is a set of features F' . After post-processing, a final edge map B is obtained.

2.2 Related Work to GP for Edge Detection

There has not been much previous work done using GP for edge detection. Harris and Buxton [12] designed approximate response detectors in one-dimensional signals by GP, but this is based on a theoretical analysis of the ideal edge detector with raw pixels and the corresponding properties. Poli [20] suggested to use four macros for searching a pixel's neighbours in image process using GP, and Ebner [5] used four shift functions and other functions to approximate the Canny

detector. The Sobel detector was approximated by hardware design [13] with the relationship between a pixel and its neighbourhood as terminals. Bolis et al [2] simulated an artificial ant to search edges in image. Zhang and Rockett [24] evolved a 13×13 window filter for comparison with the Canny edge detector. A 4×4 window was employed to evolve digital transfer functions (combination of bit operators or gates) for edge detection by GP [10]. Our previous work [8,9] used GP for edge detection based on ground truth and without using windows. All these GP edge detectors are based on raw pixels and their output can be considered as F in Figure 1.

Also, some image operators have been used to extract features in GP. Wang and Tan [23] used linear GP to find binary image edges, inspired by morphological operators (erosion and dilation) as terminals [21] for binary images. Variant features constructed by image filters with different directions using GP and texture gradients are combined to train a logistic regression classifier for boundary detection [14]. However, only one solution (combined with texture gradients) in their work was presented to compete with other edge and contour detectors. Therefore, their work belongs to the response computation phase, and the detection performance is also dependent on the texture gradients.

To sum up, the existing work has little research for constructing further features based on existing features. Although only one solution in [14] is found in the response computation phase, it still makes automatic feature construction in the feature manipulation phase appealing for edge detection.

3 Constructing Invariant Features Using GP

3.1 Terminal Set

Variant features are dependent on the direction to do extraction, and the number of features are generally large. For further constructing features by GP, the basic features in the terminal set of a GP system only contain invariant features, so that the GP system can easily choose these features to construct a new invariant feature. In this study, only three invariant features are used to construct new features. The three features are image Gaussian filter gradients T_{gg} (approximated by the horizontal and vertical derivatives [3]), a new invariant feature T_{sd} based on image quality (normalised standard deviations [22]), and histogram gradients T_{hg} [16]. Since the three features are totally different, it is possible to construct new features for improving detection performance. Therefore, the terminal set only contains T_{gg} , T_{sd} , T_{hg} , and random constants rnd in the range of $[-100, 100]$. Here, only image grayscales are used.

Image Gaussian Filter Gradients T_{gg} . The Canny edge detector [3] is a very popular edge detector. The features extracted in the Canny edge detector can be represented by the horizontal and vertical derivatives. The Canny detector puts the pre-processing and feature extraction stages together, so the horizontal and vertical derivatives for a Gaussian filter are described in formulae (1) and (2), and the Gaussian filter gradient is defined in formula (3), where u and v are the

position relative to a center pixel, and σ is the parameter for Gaussian filter. After doing convolution ($\otimes$) between image $I(x, y)$ with the Gaussian gradient $\nabla g(u, v)$, the image Gaussian filter gradients are obtained (see formula (4)).

$$g_u(u, v) = -\frac{u}{2\pi\sigma^4} \exp\left(-\frac{u^2 + v^2}{2\sigma^2}\right) \quad (1)$$

$$g_v(u, v) = -\frac{v}{2\pi\sigma^4} \exp\left(-\frac{u^2 + v^2}{2\sigma^2}\right) \quad (2)$$

$$\nabla g(u, v) = \sqrt{g_u^2(u, v) + g_v^2(u, v)} \quad (3)$$

$$T_{gg}(x, y) = \nabla g(u, v) \otimes I(x, y) \quad (4)$$

Local Normalised Standard Deviations T_{sd} . Normalised standard deviations are useful for image quality [22], but the normalised standard deviation is seldom used as a feature for edge detection in the literature. The local normalised standard deviation is introduced in this paper, and it is extracted based on a small window as a local invariant feature. The local normalised standard deviation T_{sd} is defined in formula (5), where $SD(x, y)$ and $Mean(x, y)$ are the standard deviation and mean of the pixel (x, y) intensities in a local area around, respectively.

$$T_{sd}(x, y) = \frac{SD(x, y)}{Mean(x, y)} \quad (5)$$

Image Local Histogram Gradients T_{hg} . Local image histogram gradients have shown good performance for detecting edges [16]. The image local histogram gradients are extracted based on different directions. In this paper, the local histogram gradients T_{hg} are combined with the two direction local histogram gradients as one invariant feature. The local histogram gradient in the direction θ is defined in Equation (6), where pixels around pixel (x, y) in a local area are divided into two groups based on the boundary with direction θ , and $l_{\theta, i}$ and $r_{\theta, i}$ are the occurrences for the pixels located in the bin i from the two groups, respectively. Since $h_{\theta}(x, y) \geq 0$, the local histogram gradient T_{hg} based on all possible directions θ is defined in Equation (7). Being different from T_{gg} (constructed based on the horizontal and vertical directions), T_{hg} is constructed based on $\theta = 45^\circ, 135^\circ$. The other reason for only using the two directions is that our previous work [7] showed that the diagonal derivatives are better than the horizontal and vertical derivatives for detecting edges. For the test image dataset in the Berkeley Segmentation Dataset (BSD) [16], the performance based on T_{hg} with $\theta = 45^\circ, 135^\circ$ is almost the same as T_{hg} based on $\theta = 0^\circ, 45^\circ, 90^\circ, 135^\circ$.

$$h_{\theta}(x, y) = \frac{1}{2} \sum \frac{(l_{\theta, i} - r_{\theta, i})^2}{l_{\theta, i} + r_{\theta, i}} \quad (6)$$

$$T_{hg} = \sum_{\theta} h_{\theta}(x, y) \quad (7)$$

3.2 Primitive Functions

The function set contains the four algebraic functions $\{+, -, \times, \div\}$ and three logical operators $\{IF, <, >\}$. Here, $\div$ is protected division, producing a result of 1 for a 0 divisor. IF contains three arguments, the first one is a boolean. IF will return the second argument with a real number when the first is true, otherwise will return the third argument with a real number. It is possible that a feature is better than another feature for some edge responses, but worse than the latter for other responses. Based on the logical operators, two features are combined together, and GP will automatically choose partial responses from one feature. That is why the logical operators are used.

3.3 Fitness Function

We treat the edge detection task as a balanced binary classification task (with the edge pixels as the main class) in the evolutionary training process. For the output (o) of a program, we do not use a threshold for marking edge points, but instead the output is directly evaluated by a simple Bayes model. Since the prior probabilities for the distribution of edge points and non-edge points are the same, the Bayes model is *simplified*, and formula (8) presents the weight value p_j for each class ($j = 0$ for non-edge points, and $j = 1$ for edge points) based on the output o , the estimated mean(s) $\hat{\mu}_j$ and estimated standard deviation(s) $\hat{\sigma}_j$. When p_1 is larger than p_0 , the output is considered as an edge point (in a soft edge map), otherwise, a non-edge point. The fitness function is based on the output T_{GP} without post-processing, following the suggestion from [17].

$$p_j = \frac{1}{\hat{\sigma}_j} \exp \left(-\frac{(o - \hat{\mu}_j)^2}{2\hat{\sigma}_j^2} \right) \quad (8)$$

$$T_{GP} = \begin{cases} \frac{p_1}{p_0 + p_1} & \text{if } p_1 > p_0 \\ 0 & \text{otherwise} \end{cases} \quad (9)$$

The aim of new features constructed by GP is that they should detect the number of true edge points as much as possible, so recall r (the number of pixels on the edges correctly detected as a proportion of the total number of pixels on the edges) is a very important indicator. When p_1 is larger than p_0 , the discriminated pixel is considered as an edge point ($T_{GP} > 0$). When p_1 is not larger than p_0 , the output T_{GP} is considered as very unlikely to be an edge point ($T_{GP} = 0$). However, it is possible that non-edge points are incorrectly detected as edge points, therefore, specificity s (the number of pixels not on the edges correctly detected as a proportion of the total number of pixels not on the edges) is also considered in the fitness function. Therefore, we adopt the fitness function Fit as defined in formula (10) in the training phase.

$$Fit = \frac{2rs}{r + s} \quad (10)$$

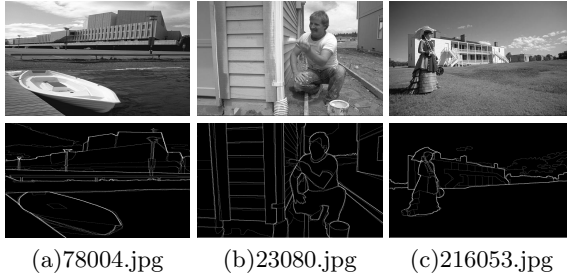


Fig. 2. Training images from BSD dataset and their ground truth

4 Experimental Design

The BSD consists of natural images (of size 481×321 pixels) with ground truth provided. All images are independent and are taken throughout the world. The training dataset contains **200** images and test dataset has **100** images. For fairness of judgement of edges, the ground truth are combined from five to ten persons as graylevel images. Fig. 2 shows three training images and their ground truth. For simplicity, we sample image pixels with the same number of edge points and non-edge points as our training data. Approximately 250 edge points and 250 non-edge points are randomly sampled from each training image. Therefore, the training data is a balanced binary classification dataset, including approximately 100,000 cases and the three invariant features. The window sizes for T_{gg} , T_{sd} and T_{hg} are 7×7 , 3×3 and 13×13 , respectively.

The parameter values for GP are: population size 200; maximum generations 200; maximum depth (of a program) 7; and probabilities for mutation 0.15, crossover 0.80 and elitism 0.05. These values are chosen based on common settings and initial experiments. There are 30 independent runs.

The evaluation is directly based on one feature, without post-processing. To measure the performance of these features constructed by GP, the F -measure is used in the testing phase [16,4]. The F -measure (used in [16,4] as $F = \frac{2rp}{r+p}$) is the combination of recall r and precision p (the number of pixels on the edges correctly detected as a proportion of the total number of pixels detected as edges). In the F -measure evaluation system, pixels are discriminated as edge points based on the value of their features larger than a threshold, and the predicted edges are simply thinned by the thinning operator [16]. After obtaining thinned prediction, an optimal matching operator will be used to match the prediction and the ground truth. Based on different threshold level indices $k = 0, 1, \dots, 51$, a maximum F_{max} ($F_{max} = \max\{F_k\}$) will be considered as the measurement for the feature, where F_k is the F value when the threshold level index k is used.

For fair comparison, the F_{max} values of the three basic features (T_{gg} , T_{sd} , T_{hg}) also are given without post-processing, so their values are different from the final performance evaluation in [16]. In the feature performance evaluation, the Sobel edge detector is the same result as the final detection result evaluated in [16], because the Sobel edge detector does not contain special post-processing techniques.

Table 1. Comparison of F_{max} values among constructed features by GP, Image Gaussian Gradients T_{gg} , Normalised Standard Deviations T_{sd} , Histogram Gradients T_{hg} , Sobel Edge Detector and a Bayes Model for the BSD Test Images

	F_{max}
GP	0.5728 ± 0.0292
T_{gg}	0.5153
T_{sd}	0.4968
T_{hg}	0.5434
Sobel	0.4832
Bayes	0.5302

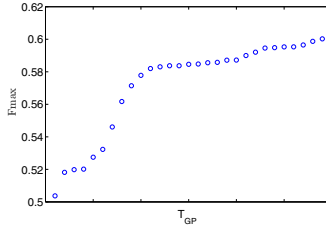


Fig. 3. F_{max} values for the 30 features constructed by GP

5 Results and Discussion

5.1 Overall Results

Table 1 presents the mean and standard deviation of F_{max} values from the 30 features constructed by GP, and F_{max} values from T_{gg} , T_{sd} , T_{hg} and the Sobel edge detector. The training time for each constructed feature is around 3 hours, but the test time can be ignored because the time to extract histogram gradients is far longer than executing a program. Also, an estimated Bayes model based on the sampling dataset is used to extract features (using formulae (8) and (9)) for the test images when T_{gg} , T_{sd} , T_{hg} are considered as independent variables. The bold font means that the results from GP are significantly better than the others using the one sample t -test with significance level 0.05. From the test results, it shows that the features constructed by GP significantly improve the detection performance. However, the combination of the three features by the Bayes model does not improve the detection performance. Therefore, we can see that GP is effective for automatic construction of invariant features.

5.2 Comparison among GP, T_{gg} , T_{sd} and T_{hg}

Figure 3 shows the F_{max} values of the 30 features constructed by GP. The maximum F_{max} is 0.6009, and the lowest F_{max} is 0.5037. From Figure 3, the worst feature in the 30 features is an outlier, and it is worse than T_{gg} and T_{hg} (see Table 1). The number of the 30 features with higher F_{max} than T_{hg} is 24, and

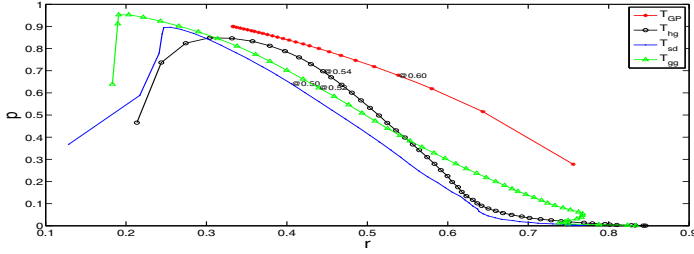


Fig. 4. Details for r and p of T_{gg} , T_{sd} , T_{hg} and the best feature constructed by GP

two thirds of these features are higher than 0.58. Therefore, most of the features constructed by GP are better than the basic features although the training data is not large.

Figure 4 shows the details for r and p with different threshold levels. Here, “@” is the position for the F_{max} . Compared with the three basic features T_{gg} , T_{sd} and T_{hg} , the curve for the best feature constructed by GP (T_{GP}) is obviously better than the three basic curves from T_{gg} , T_{sd} and T_{hg} . Therefore, the r vs p curves also show that GP can construct good features.

5.3 Detected Images

Figure 5 shows four detected images from the BSD test image dataset by T_{gg} , T_{sd} , T_{hg} , the best constructed feature T_{GP} , the Sobel edge detector, and the feature constructed by the simple Bayes model (see formulae (8) and (9)), where “GT” is ground truth. The detected images from T_{gg} , T_{sd} , T_{hg} might be hard to read in print version because of low contrast edge responses. The reason for selecting the four images is that they have edges which are difficult to detect: the graylevels of the top of sail in the first image are similar to the background (dark sky); the texture of the body in the second image is similar to the background (grass covered ground); the middle of the background in the third image has plants with slightly irregular texture in different spaces and their reflection in the water; and the last image includes clouds in different spaces, the graylevels of the left middle of the animal body are almost the same as the clouds, and irregular stone textures.

There are interesting observations from the three basic features. From the detected images, T_{gg} is very good at finding obvious edges without textures, but has only a weak response in low contrast areas, such as the top of the sail in the first image. T_{sd} improves the responses on the edges in low contrast areas (see the detected edges from the sail in the first image), but gives stronger responses on the irregular textures, such as the stone area in the fourth image, compared with T_{gg} . Compared with T_{gg} and T_{sd} , T_{hg} suppresses the responses in the texture areas, however it gives too stronger responses on the discontinuities, such as the responses on the edges for the clouds in different spaces, even the water wave.

The detected images by GP in Figure 5 reveal that the feature constructed by GP weakens the response problems (in T_{gg} , T_{sd} and T_{hg} for some non-edge areas)

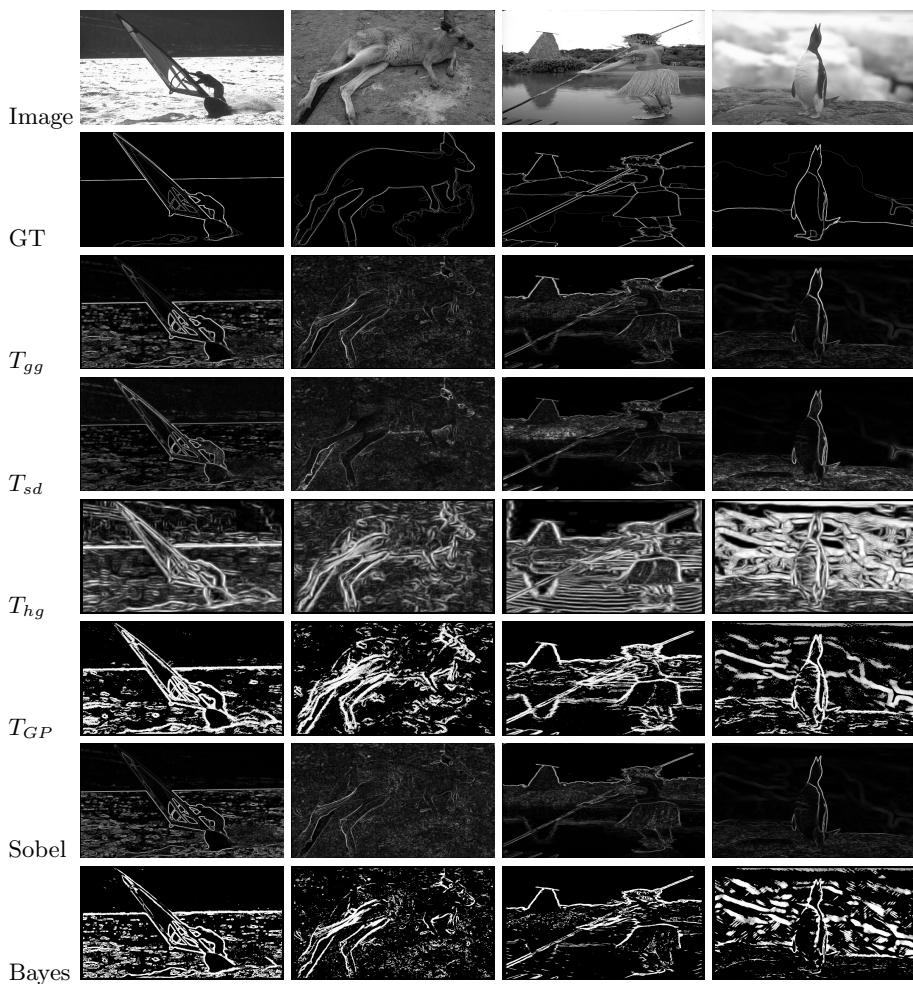


Fig. 5. Detected images based on the different features

and gives clear responses on edges. From the four detected images, it seems that the constructed feature suppresses most of the textures in the four images. From the first image and fourth image, the constructed feature has a good ability to detect edges in low contrast areas, such as the edges of the sail, and the response magnitudes for the boundaries of the clouds (weaker than the response for the contour of the animal). From the third image, the constructed feature avoids the strong response on the water wave. Therefore, the constructed feature combines the advantages from image Gaussian gradients T_{gg} , local normalised standard deviations T_{sd} and image local histogram gradients T_{hg} , and weakens the inappropriate responses existing in the three basic features.

The detected results by the simple Bayes model still have the problem of high responses on low contrast areas, such as the water waves in both sides of the

third image and the boundaries of the clouds in the fourth image. The feature constructed by the Bayes model is still not good for texture suppression, such as the water surface in the first image and the grass in the second image. Therefore, these examples confirm the effectiveness of the GP method.

6 Conclusions

The goal of this paper was to investigate using GP to construct invariant features for edge detection to improve the detection performance. Based on the experimental results of the features constructed by GP with three basic features, namely image Gaussian gradients, local normalised standard deviations and image local histogram gradients, the goal was successfully achieved. The constructed features combine the advantages from the three basic features, and reduces their disadvantages. Also, the comparison between GP and a simple Bayes model shows that GP has ability to find a way of efficiently combining different features together.

For future work, we will test this technique on various features and analyse constructed features in order to find useful rules (functions) for combining features to improve detection performance. In addition, other machine learning algorithms will be used to compare with GP. Post-processing techniques will be employed to obtain the final solutions, and the final edge maps will be compared to state of the art edge and contour detectors.

References

1. Basu, M.: Gaussian-based edge-detection methods: a survey. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews* 32(3), 252–260 (2002)
2. Bolis, E., Zerbi, C., Collet, P., Louchet, J., Lutton, E.: A GP Artificial Ant for Image Processing: Preliminary Experiments with EASEA. In: Miller, J., Tomassini, M., Lanzi, P.L., Ryan, C., Tetamanzi, A.G.B., Langdon, W.B. (eds.) *EuroGP 2001*. LNCS, vol. 2038, pp. 246–255. Springer, Heidelberg (2001)
3. Canny, J.: A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 8(6), 679–698 (1986)
4. Dollar, P., Tu, Z., Belongie, S.: Supervised learning of edges and object boundaries. In: *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2, pp. 1964–1971 (2006)
5. Ebner, M.: On the edge detectors for robot vision using genetic programming. In: *Proceedings of Horst-Michael Groß, Workshop SOAVE 1997 - Selbstorganisation von Adaptivem Verhalten*, pp. 127–134 (1997)
6. Fernández-García, N., Carmona-Poyato, A., Medina-Carnicer, R., Madrid-Cuevas, F.: Automatic generation of consensus ground truth for the comparison of edge detection techniques. *Image and Vision Computing* 26(4), 496–511 (2008)
7. Fu, W., Johnston, M., Zhang, M.: Analysis of diagonal derivatives in edge detectors evolved by genetic programming. In: *Proceedings of the Twenty-Sixth International Conference on Image and Vision Computing*, New Zealand, pp. 345–350 (2011)

8. Fu, W., Johnston, M., Zhang, M.: Genetic programming for edge detection: a global approach. In: *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 254–261 (2011)
9. Fu, W., Johnston, M., Zhang, M.: Genetic programming for edge detection based on accuracy of each training image. In: *Proceedings of the 24th Australasian Joint Conference on Artificial Intelligence*, pp. 301–310 (2011)
10. Golonek, T., Grzechca, D., Rutkowski, J.: Application of genetic programming to edge detector design. In: *Proceedings of International Symposium on Circuits and Systems*, pp. 4683–4686 (2006)
11. Grigorescu, C., Petkov, N., Westenberg, M.A.: Contour and boundary detection improved by surround suppression of texture edges. *Image and Vision Computing* 22(8), 609–622 (2004)
12. Harris, C., Buxton, B.: Evolving edge detectors with genetic programming. In: *Proceedings of the First Annual Conference on Genetic Programming*, pp. 309–314 (1996)
13. Hollingworth, G., Smith, S., Tyrrell, A.: Design of highly parallel edge detection nodes using evolutionary techniques. In: *Proceedings of the Seventh Euromicro Workshop on Parallel and Distributed Processing*, pp. 35–42 (1999)
14. Kadar, I., Ben-Shahar, O., Sipper, M.: Evolution of a local boundary detector for natural images via genetic programming and texture cues. In: *Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation*, pp. 1887–1888 (2009)
15. Lim, D.H., Jang, S.J.: Comparison of two-sample tests for edge detection in noisy images. *Journal of the Royal Statistical Society. Series D (The Statistician)* 51(1), 21–30 (2002)
16. Martin, D., Fowlkes, C., Malik, J.: Learning to detect natural image boundaries using local brightness, color, and texture cues. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 26(5), 530–549 (2004)
17. Moreno, R., Puig, D., Julia, C., Garcia, M.: A new methodology for evaluation of edge detectors. In: *Proceedings of the 16th IEEE International Conference on Image Processing (ICIP)*, pp. 2157–2160 (2009)
18. Papari, G., Campisi, P., Petkov, N., Neri, A.: A biologically motivated multiresolution approach to contour detection. *EURASIP Journal on Applied Signal Processing*, 119 (2007)
19. Papari, G., Petkov, N.: Edge and line oriented contour detection: state of the art. *Image and Vision Computing* 29, 79–103 (2011)
20. Poli, R.: Genetic programming for image analysis. In: *Proceedings of the First Annual Conference on Genetic Programming*, pp. 363–368 (1996)
21. Quintana, M.I., Poli, R., Claridge, E.: Morphological algorithm design for binary images using genetic programming. *Genetic Programming and Evolvable Machines* 7, 81–102 (2006)
22. Rezai-Rad, G., Larijani, H.H.: A new investigation on edge detection filters operation for feature extraction under histogram equalization effect. In: *Proceedings of the Geometric Modelling and Imaging*, pp. 149–153 (2007)
23. Wang, J., Tan, Y.: A novel genetic programming based morphological image analysis algorithm. In: *Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation*, pp. 979–980 (2010)
24. Zhang, Y., Rockett, P.I.: Evolving optimal feature extraction using multi-objective genetic programming: a methodology and preliminary study on edge detection. In: *Proceedings of the Conference on Genetic and Evolutionary Computation*, pp. 795–802 (2005)