Lazy Nonlinear Diffusion Parameter Estimation

Daniel Thuerck[1] and Arjan Kuijper[1,2]

[1] TU Darmstadt,
[2] Fraunhofer IGD, Darmstadt,
Germany

Abstract. Perona–Malik diffusion is a well-known type of nonlinear diffusion that can be used for image segmentation and denoising. The process itself needs an parameter k to decide which edges will be retained and which can be blurred and a stopping time t_S. Although there have been investigations on how to set these parameters, especially for regularized diffusion models, as well as different criteria for the optimal stopping time have been suggested, there is yet no quick and conclusive way to estimate both parameters – or to reduce the search space at least. In this paper, we show that Gaussian noise characteristics of an image and the diffusion parameters for an optimal optical result can be estimated based on the image histogram. We demonstrate the effectiveness of lazy learning in this area and develop a custom feature weighting algorithm.

1 Introduction

In 1990, Perona and Malik proposed nonlinear diffusion as a model mainly for image segmentation in a scale-space process [10]. It has been shown that this model can also be used for denoising. The model itself is expressed through a partial differential equation (PDE): $I_t = \operatorname{div}(c(||\nabla I||)\nabla I)$ with $I(x,0) = I_0(x)$, where $\Omega \subset \mathbb{R}^2_+$ denotes the picture domain, $I : \Omega \to \mathbb{R}_+$ the intensity mapping for gray scale images, $x \in \Omega$, and appropriate boundary condition are taken. Furthermore, Perona and Malik introduced *conduction coefficient functions* expressed by c. The proposed explicit functions are $c_1(||\nabla I||) = \exp\left(-\left(||\nabla I||/k\right)^2\right)$ and $c_2(||\nabla I||) = \left(1 + \left(||\nabla I||/k\right)^2\right)^{-1}$.

Of utmost importance is the parameter k. It can best be described as an *gradient threshold*. When we investigate the derivative of the *flux function*

$$\Phi(\nabla I) = \nabla I \cdot c(||\nabla I||), \tag{1}$$

we notice that with c_1 for edges with $||\nabla I|| > k$, we experience *backwards diffusion*, for $||\nabla I|| \leq k$ *forward diffusion*. On the one hand the backwards diffusion leads to edge enhancement [10,4,2,6,13], but also creates the problem of ill-posedness and image segmentation, which is not desirable for denoising.

2 Related Work

Different methods to estimate or determine k and t_S have been investigated. For k, Perona and Malik proposed in [10] a method: in each iteration, take the Quantile $p_{0.9}$ of the

A. Petrosino (Ed.): ICIAP 2013, Part I, LNCS 8156, pp. 211–220, 2013.

gradient histogram as k. Of course, this approach was meant for image segmentation, not denoising.

In the total variation image processing domain, the constrained optimization approach of Rudin et al. [12] avoids the problem of stopping time estimation – at the cost of a Lagrangian multiplier that has to be chosen. Convergence of the PDE then yields the desired minimal energy and optimized image [6,5].

Although the choice of k is such an important issue in nonlinear diffusion, not much research has been done in the area lately. Today, there are mostly simple histogram-based and morphological approaches. Histogram-based methods include the mentioned Canny algorithm which changes k in each iteration or a simple quadratic approximation of k by the image's average gradient as carried out in [14]. This is an interesting and yet simple approach; however, the formula is only based on two images. A quick experiment using our training set with ca. 130 images reveals that the method cannot be generalized.

In [16], Voci et al. proposed estimating the gradient threshold by the gradient histogram's p-Norm. For this, they introduced a new parameter σ for weighting the p-Norm. In effect, we get an histogram-based method again. Their morphological method, in contrast, compares the image opening and closing, thus estimating the average noise quantity and in the end, k.

An interesting approach, similar to the Canny method can be found in [7]. This method does not take the full histogram into account. Instead, it ranks blocks of a given window size w by a defined homogeneity and only regards the most inhomogeneous blocks for parameter estimation.

Weickert introduced his coherence enhancing diffusion and discussed the choice of the parameters - like a suitable stopping time t_S of the process [17]. Mrazek and Navara developed a time-selection strategy for iterative image restoration techniques: the stopping time is chosen such that the correlation of signal and noise in the filtered image is minimized [8]. Gilboa et al. used SNR Analysis [1] which was shown to be quite effective with respect to maximizing the SNR, but the SNR is per se not a good indicator for the subjective quality of an image as perceived by a human. This issue has been investigated by Wang et al. in [9] who developed the *structural similarity index* SSIM. The resulting index is a good approximation of the human visual system, as discussed by Ndajah et al. [9]. In this paper, we will concentrate on estimating parameters which will maximize the SSIM.

Different to the mentioned authors, we try to give *a priori* estimates for the parameters, not an *a posteriori* stopping criterion.

3 Parameter Estimation

In this section, we present for estimating the mentioned parameters. The first method is reducing the parameter space such that brute force parameter search becomes feasible. When an explicit model, on the other hand, is required, we show a simple machine learning method that is able to give reasonable estimates for a given input image.

3.1 Parameter Space Reduction

In [15], we present a new model for nonlinear diffusion that is constructed for efficient denoising and well-posedness. Although it requires again the parameter pair (k, t_S),

we could show that the range of k values is significantly smaller than with the original Perona and Malik model. All images in our test set had the optimal parameter k in the range of $[6, 12]$ which results in acceptable denoising for a constant value $k = 9$.

3.2 Machine Learning Methods

Machine learning methods are usually categorized into *lazy learning* and *eager learning*. When applying lazy learning, an agent usually works with a training set of classified instances and compares them with a given, unclassified instance. The main methods in this field are nearest neighbour-based algorithms.

In eager learning, the learning algorithm usually gets the database as an input once and derives an algorithm for classifying from this data. Afterwards, a given example can be classified with respect to the learned model. While eager learning has the advantages of needing less memory and being able to reduce the influence of noise in the data, lazy learning techniques are especially useful when the data has no simple underlying model that can be learned by common machine learning algorithms.

In the remainder of this paper we show that for the estimation of k, no easy model can be derived. Hence we are restricted to lazy learning techniques. In contrast to that, when estimating the stopping time t_S methods from both classes work almost equally well.

3.3 Data Structure

For all machine learning experiments, we need to create input data from a set of images as training or test data. A prerequisite for this process is to define the feature set for the input training data set. In contrast to [17], we focus on a histogram approach: Each gray scale image is transformed in a *intensity histogram* and a *gradient magnitude histogram*. After that, we interpret both histograms as a statistical distribution and use these standard descriptors for characterizing the distribution:

- Quantiles $\left\{p_{\frac{1}{8}}, p_{\frac{1}{4}}, p_{\frac{1}{2}}, p_{\frac{3}{4}}\right\}$
- Average value
- Median value
- Standard deviation

These result in a $(14 + 1)$-dimensional space. Hence, each dimension is called an *attribute* or *feature* of an *instance*, which is a n-tuple of values, one for each attribute and one for the parameter k or t_S. For later sections, we will denote the set of all attributes by $\mathcal{A}$. The 15th dimension is the dimension of the value that shall be estimated, e.g. k. Our used training set contains 128 images in different categories, mainly from the Berkeley Segmentation database. The test set contains 10 images (see Fig. 3) not in the training set.

Interestingly, in most publications, the authors conclude that structural information is required to estimate good parameter values. In contrast to that, our results show that the simple histogram approach does a reasonable job while being relatively simple.

3.4 Estimating Noise Variance

In this paper, we assume a Gaussian noise model with zero mean as underlying noise model for our images. For our numerical experiments, the noise variance will be discretized in a set of classes: $\{0.001, 0.005, 0.01, 0.05, 0.1\}$. To the human vision system, a higher noise variance is commonly perceived as *stronger noise*. For denoising purposes, this usually means that more iterations of the denoising process must be executed. Hence, the estimation of an image's noise variance is a prerequisite for estimating t_S.

For noise estimation, we used a simple regression tree algorithm with alternating growing and pruning, using randomly chosen $\frac{2}{3}$ of the data for growing in each iteration. The error measurement is carried out using the mean squared error metric. A leaf's value is then the average variance of all instances in the leaf.

When using the resulting model for classifying the test set presented in the next sections, we can estimate the correct value in 90% of all cases. Of course, this might be due to the discretized classes. This weakness can be improved by interpolating between classes or by using more complex regression tree algorithms like *M5P* [11] which create linear functions for each leaf. However, a 90% success rate is sufficient for our following tasks. The value can be used as a 16th dimension in the parameter estimation process or for inducing convergence in the denoising process (see [15]).

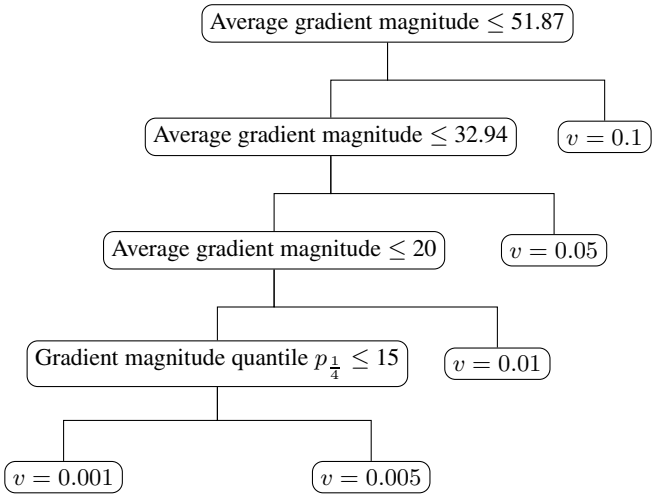


Fig. 1. Example regression tree for estimating noise variance v for a given image, trained on our training set

3.5 Estimating *k*

Unfortunately, the estimation of k is not quite as easy as the variance. If we plot k against each one of our features, most plots are likely to resemble normal distributions. Therefore, there is no usable information which could be used with regression algorithms. Experiments with our regression program verify that claim: The data is at most

distributed in 2 leafs which would result in a binary choice for k. This is also the reason why the approach of Shao and Zou cannot work: There is no quadratic interpolation model for the correlation of average gradient, we can only recognize the different variance classes in a plot of k over the average gradient. This fact can be proven by using an regression algorithm based on a quadratic model. Additionally, additive regression or other ensemble techniques as boosting do not work. In fact, none of the classifiers provided by the WEKA [3] framework deliver a satisfying model.

This insight leads to the necessity of using a lazy learning technique, for example a simple nearest neighbor (NN) method. In this algorithm, all features or attributes $a \in \mathcal{A}$ for a given image are calculated. Afterwards, we compute the distance between two instances by an modified euclidean metric

$$d(x,y) = \sqrt{\sum_{a \in \mathcal{A}} (a_x - a_y)^2}. \tag{2}$$

as a combination of the distances of two instances' attributes. However, as most attributes usually have different orders of scale - e.g. the average gradient may be around 32, while the intensity Quantile $pi_{0.75}$ is 160 - thus leading to an unwanted implicit attribute weight in the naive approach. A first solution here is to normalize all attributes to $[0,1]$. Normalized attributes will be denoted by $\hat{a}$, leading to the metric

$$d(x,y) = \sqrt{\sum_{a \in \mathcal{A}} (\hat{a_x} - \hat{a_y})^2}. \tag{3}$$

where all attributes share the implicit weight 1. Equation (6) shows the metric that is used in simple nearest neighbor methods in our experiments („1NN“ and „4NN“ where x in xNN is the number of nearest neighbors whose values are included in calculating the estimation).

Up to here, our technique uses equal weights for all attributes of a histogram. As explained before, some attributes are normally distributed when plotting their relation to k. Hence, those attributes deliver less information in the sense of entropy than non-normally distributed attributes. This motivates our attribute weighting method: Attributes which contain more information about the distribution of k receive a higher weight than others. We will measure the amount of transferred information by the discrete entropy.

After Shannon, the entropy of a discrete random variable X over an alphabet $Z = \{z_1, z_2, ...\}$ is defined as $H(X) = -\sum_{z \in Z} P(X = z) \cdot \log_2 P(X = z)$.In the application, we discretize the values of each attribute in 256 values, resulting in the alphabet $\{0, 1, ..., 255\}$ and calculate the attribute's entropy. Afterwards, the entropy value is divided by the entropy of a normal distribution, which is in this case $\log_2 256 = 8$. The derived formula for the weights is now

$$w_a = 1 - \frac{H(a)}{8} \tag{4}$$

which results in our *entropic distance metric*

$$d(x,y) = \sqrt{\sum_{a \in \mathcal{A}} w_a (\hat{a_x} - \hat{a_y})^2}. \tag{5}$$

The metric can be explained intuitively: An attribute that has a lower entropy value will usually reveal more regularities in contrast to the normal distribution. Hence, an attribute with lower entropy should be weighted stronger in the distance measure. The comparison with the normal distribution's entropy is only for normalizing purposes.

After calculating the distance to all images in the database, those images are ranked after their distance, low distances first. To estimate k, the values of the first n nearest neighbors is simply averaged. Here again, we can introduce weights in the average to penalize higher distances. In our experiments, a weight $\frac{1}{d^2}$ with d being the distance, delivered the best results.

This entropic measure will be called *nENN* with n as the number of instances that are involved in the averaging process.

3.6 Estimating t_S

Having k fixed, we usually get a development for the image quality as depicted in Fig. 2. The image quality (independent of used metric) has a single global maximum at a given time t_S which we call *optimal stopping time*. For all $t > t_S$, the image quality decreases; convergence is not guaranteed. Apart from estimating k, the next task is now estimating the ideal stopping time.

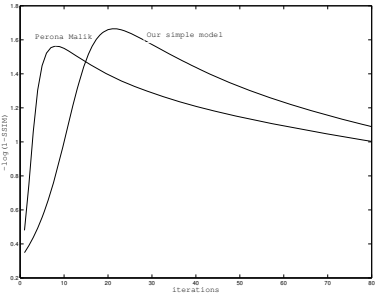


Fig. 2. Typical image quality development (SSIM [9]) in time when applying a diffusion model (here from [15])

The problem is very similar to the problem explained above: Scatter plots show that there also is no trivial model for t_S. Hence we need to use our learning techniques from above again; the technique can be applied without further changes. Interestingly, experiments show that the eager learner M5P delivers results of nearly the same quality for the test set as the entropic lazy learning techniques. For M5P, a linear model with an underlying regression tree of 2 classes was used. As a minimum occupancy per node, we chose $n = 100$; the model was created using the machine learning framework *WEKA* [3].

4 Numerical Experiments and Comparison

All learning algorithms mentioned above were tested on a test set of 10 images, none of those contained in the training set (see Fig. 3). We tried to concentrate on pictures

which were not taken for testing purposes, as this shows the performance of our method in practical use. All images were scaled to 512 pixels on the larger side and saved as binary PGM.

The variance levels used for noise on the test set were $\{0.005, 0.05\}$. For each picture, we determined optimal parameters (k, t_S) for each image and variance levels using brute force to optimize the SSIM. The metric used for image quality measure is the SSIM [9]. The SSIM offers an human vision system-based quality metric that penalizes noise as well as blur [9]; just notice that the SSIM is normalized to $[0, 1]$ and equal images result in an SSIM of 1.

The Perona Malik filter itself is discretized as explained in [10] as a simple finite differences *lattice* model and implemented in pure C. The resulting programs are connected to MATLAB via the MEX interface. For testing purposes, we benchmarked the algorithms 1ENN, 4ENN, 1NN, 4NN, M5P and Shao [14], with abbreviations explained above.

Fig. 3. Images in test set in the order of their reference number

4.1 Estimation Performance for k and t_S

As the short statistical evaluation for k (Table 1) and t_S (Table 2) show, for the ks in an interval of [32,111], all methods deliver a quite good estimation. Although the average performance is comparable, the standard nearest neighbour methods 1NN and 4NN tend to have outliers, e.g. Fig. 5. The ENN methods, on the other hand, present a constant estimation performance, preventing extremely wrong results.

When comparing the number of data points used for estimation, the entropic method improves in performance with a higher n. Further experiments have shown that increasing n beyond 4 does not yield a significant improvement in the estimation performance. In the light of this result, we recommend two combinations for parameter setting:

1. 1ENN for k, 4ENN for t_S
2. 4ENN for k, 4ENN for t_S

as 1ENN and 4ENN are likewise suited for k estimation. Both methods and their SSIM results are compared in the next section.

4.2 Denoising with Estimated Parameters

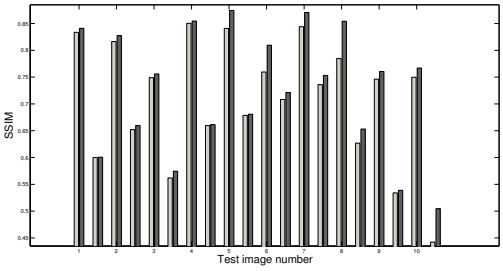
After evaluating the estimated parameters in contrast to the optimised parameters, we shall now proceed to investigate the effects of estimation errors on the image denoising

Table 1. Statistical evaluation of k estimation methods

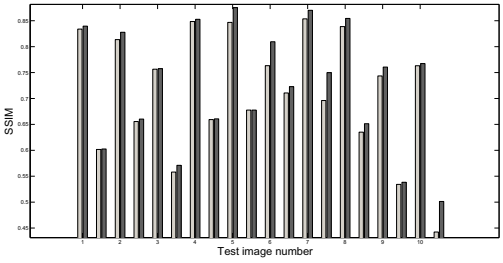
Quantity	1ENN	4ENN	1NN	4NN	M5P	Shao
average absolute error	11.15	11.19	17.55	21.6	51.45	35.57
average relative error	0.18	0.19	0.36	0.48	0.8	0.58
absolute error span	0-30	2-27	3-79	2-91	14-95	19-60
relative error span	0-0.3	0-0.39	0-1.6	0-2.9	0.65-1	0.53-0.65

Table 2. Statistical evaluation of t_S estimation methods

Quantity	1ENN	4ENN	1NN	4NN	M5P
average absolute error	5.3	4.5	7.7	5.79	5.67
average relative error	0.52	0.39	0.71	0.52	0.19
absolute error span	0-24	0-13	0-27	0-24	2-44
relative error span	0-3.5	0-1.43	0-3.5	0-1.4	0.34-0.56



(a) 1ENN for k, 4ENN for t_S



(b) 4ENN for both k and t_S

Fig. 4. SSIM results for denoising with 2 parameter estimation methods. For each test images, we present two bar groups: One for variance 0.005 and one for variance 0.05, where in turn each group consists two bars, denoting the result of the application of the estimated and optimal parameters.

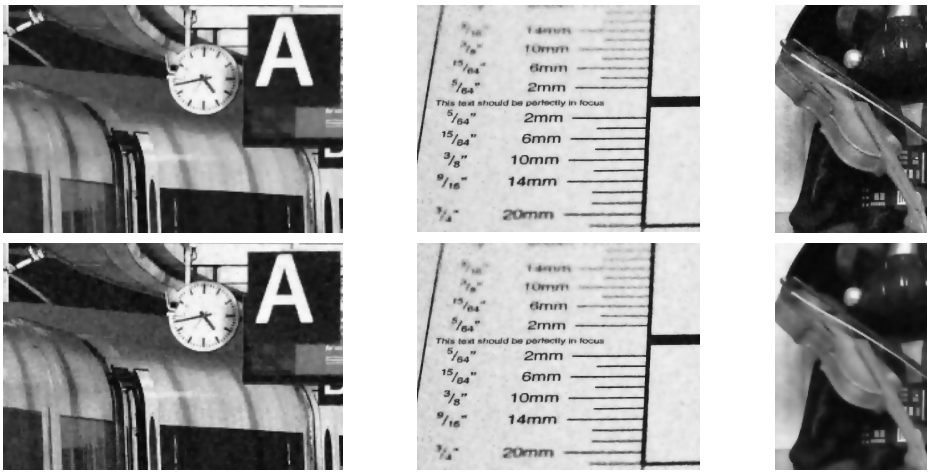


Fig. 5. Testing images #8,#7,#5: two typical cases and #5 as worst case, i.e. worst estimation. Top row: optimal parameters, bottom row: estimated parameters.

process. As mentioned above, the denoising success is measured by an increase of the denoised image's SSIM. When looking at bar charts Fig. 4 (a) for method 1 and (b) for method 2, we can see that the errors from estimation result in only small SSIM deficits. Altogether, the estimated values provided almost equal denoising as the optimised parameters. It is obvious that higher estimation errors lead to higher SSIM deficits, but for a quick estimation, our methods perform well. If one wants to optimise the SSIM, our estimates can be used as bounds for reducing the parameter search space by over 70%.

5 Conclusion

This paper has investigated the use of different machine learning techniques for Perona and Malik parameter estimation. An entropy-based distance measure was developed and used for estimation, which resulted in parameter combinations that led to denoised images of a quality comparable to the image denoised with optimal parameters. Different estimation methods were compared, with 4ENN winning the overall comparison. Its underlying database can be gradually improved by adding new images, making it possible to create even better approximations. The methods itself can also be applied to other denoising processes.

Acknowledgements. We thank Annemarie Sonnenberg for the clearance to use test pictures #1 and #5 as well as flickr users joachim_s_mueller, hotgear, mitch-in-wanderlust, sigsegv, 34053291@N05 and tuxdriver. We also thank Oren Halvani from CASED for his help in refining the manuscript.

References

1. Gilboa, G., Sochen, N.A., Zeevi, Y.Y.: Estimation of optimal pde-based denoising in the SNR sense. IEEE Transactions on Image Processing 15(8), 2269–2280 (2006)
2. Guo, Z., Sun, J., Zhang, D., Wu, B.: Adaptive Perona-Malik model based on the variable exponent for image denoising. IEEE TIP 21(3), 958–967 (2012), http://dx.doi.org/10.1109/TIP.2011.2169272
3. Halland, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The WEKA data mining software: An update. In: ACM SIGKDD Explorations, vol. 11, pp. 10–18 (2009)
4. Kichenassamy, S.: The Perona-Malik method as an edge pruning algorithm. Journal of Mathematical Imaging and Vision 30, 209–219 (2008)
5. Kuijper, A.: p-Laplacian driven image processing. In: 14th International Conference on Image Processing, ICIP 2007, vol. V, pp. 257–260 (2007)
6. Kuijper, A.: Geometrical PDEs based on second order derivatives of gauge coordinates in image processing. Image and Vision Computing 27(8), 1023–1034 (2009)
7. Monteil, J., Beghdadi, A.: A new interpretation and improvement of the nonlinear anisotropic diffusion for image enhancement. IEEE TPAMI 21(9), 940–946 (1999)
8. Mrázek, P., Navara, M.: Selection of optimal stopping time for nonlinear diffusion filtering. International Journal of Computer Vision 52(2-3), 189–203 (2003)
9. Ndajah, P., Kikuchi, H., Yukawa, M., Watanabe, H., Muramatsu, S.: An investigation on the quality of denoised images. Circuits, Systems and Signal Processing 5, 423–434 (2011)
10. Perona, P., Malik, J.: Scale-space and edge detection using anisotropic diffusion. IEEE Transactions on Pattern Analysis and Machine Intelligence 12, 629–639 (1990)
11. Quinlan, J.R.: Learning with continuous classes. In: Proceedings of the Australian Joint Conference on Artificial Intelligence, pp. 343–348 (1992)
12. Rudin, L.I., Osher, S., Fatemi, E.: Nonlinear total variation based noise removal algorithms. Physica D 60(259-268) (1992)
13. Schwarzkopf, A., Kalbe, T., Bajaj, C., Kuijper, A., Goesele, M.: Volumetric nonlinear anisotropic diffusion on gpus. In: Bruckstein, A.M., ter Haar Romeny, B.M., Bronstein, A.M., Bronstein, M.M. (eds.) SSVM 2011. LNCS, vol. 6667, pp. 62–73. Springer, Heidelberg (2012)
14. Shao, H., Zou, H.: Threshold estimation based on Perona-Malik model. In: Int. Conf. on Computational Intelligence and Software Engineering, pp. 1–4 (2009)
15. Thuerck, D., Kuijper, A.: Cosine-driven non-linear denoising. In: Kamel, M., Campilho, A. (eds.) ICIAR 2013. LNCS, vol. 7950, pp. 245–254. Springer, Heidelberg (2013)
16. Voci, F., Eiho, S., Sugimoto, N., Sekibuchi, H.: Estimating the gradient in the Perona-Malik equation. IEEE Signal Processing Magazine 21(3), 39–65 (2004), http://dx.doi.org/10.1109/MSP.2004.1296541
17. Weickert, J.: Coherence-enhancing diffusion of colour images. Image Vision Comput. 17(3-4), 201–212 (1999)