Performance Study of a Regularization-Based Deformable Handwritten Recognition Approach

Yoshiki Mizukami, Shinya Nakanishi, and Katsumi Tadamura

Yamaguchi University, 2-16-1 Tokiwadai, Ube 755-8611, Japan
{mizu,s028vm,tadamura}@yamaguchi-u.ac.jp

Abstract. This study clarifies the accuracy performance of a deformable handwritten recognition approach (DHRA) for digit characters. The deformable approach consists of regularization-based displacement computation, coarse-to-fine strategy, distance measurement and k-nearest neighborhood method. We focus on several conditions for investigating the accuracy and the sensitivity, that is, the definition of averaging area in regularization process, regularization parameters and the number of k for k-nearest neighborhood method. According to the simulation results, it was shown that the proposed method has the error rate of 0.42% for MNIST handwritten digit database, resulting in the top-group of the performances reported until now.

Keywords: deformable handwritten recognition, MNIST digit database, regularization.

1 Introduction

Handwritten character recognition is one of the interesting topics in the field of computer vision and machine intelligence, since it does not only relate with the applications of optical character recognition but also the functionality of the human brain for reading many kinds of characters [1]. In order to compare the accuracy performances of different recognition approaches, several evaluation databases have been provided, and modified-NIST (MNIST) database is frequently employed [2].

Neural network approaches have been successfully applied to MNIST database. LeCun et al. proposed convolutional neural networks (CNN) [2–4], which are specifically designed to deal with the variability of 2D shapes. LeNet-5, a convolutional network, comprises 8 layers and has feature mapping planes where a set of the units shares identical weight values for an efficient learning. Under the hypothesis that more training data would improve the accuracy, they artificially generated more samples by randomly distorting the original training samples. For the distortion operation, planar affine transformation or elastic transformation was utilized. These ideas of deep convolutional neural network (DNN), earlier introduced by [5] and sophisticated by [2–4], were more developed by Ciresan et al. [6, 7]. They takes advantage of the recent parallel computation devices

A. Petrosino (Ed.): ICIAP 2013, Part I, LNCS 8156, pp. 300–310, 2013.

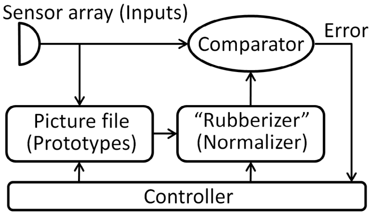


Fig. 1. Rubber mask technique for pattern recognition

(graphics processing units; GPUs) for their powerful learning process, and employed several effective strategies, that is, only winner-neurons are trained and several DNN columns were combined to form a multi-column DNN (MCDNN). The error rate of MCDNN was reported as 0.23%.

The support vector machine (SVM) is an extremely economical way of representing complex surfaces in high-dimensional spaces [8], which has been applied to various problems in pattern recognition. LeCun et al. investigated the application of SVM to MNIST [2], and then DeCoste and Scholkopf reported the error rate of their SVM as 0.56% [9]. Lin et al. reported their performance of 0.42% by using 8-direction gradient features and RBF kernel [10]. Recently, Niu and Suen presented a hybrid CNN-SVM classifier [11] and clarified the error rate of 0.19%, which is the top performance for MNIST database right now.

Deformable approach with k-nearest neighborhood method seems to be an alternative possibility. Compared with the above-mentioned approaches, deformable approach does not need the training process in advance, and the variability of 2D shapes is dealt with by deforming one of an input and prototype pattern into the other. The idea of the deformable approach can go back to Widrow's "rubber mask" technique [12], which was inspired by Gregory's insights [13],

> ⋯ perception is not determined simply by the stimulus patterns; rather it is a dynamic searching for the best interpretation of the available data ...

Figure 1 illustrates a recognition process of the rubber mask technique. An input is captured on the sensor array and the prototypes, which might be image features, are deformed in the "rubberizer" under the controller. Finally, both of them are compared in the comparator. The obtained error is referred by the controller to update the deformation. Many deformable approaches were proposed until now, and Belongie et al. applied their deformable approach to MNIST database earlier [14]. They defined shape context (image feature) at the contour pixels so as to represent the distribution of the surrounding contour pixels and utilized the shape context as image feature for elastically corresponding two character shapes with sub-pixel displacement. The error rate was reported as 0.63%. Keysers et al. proposed a novel approach [15], where the horizontal and vertical gradients with 3 x 3 pixel size are extracted as local context (image feature) and warping algorithm [16] is employed for the minimization in pixel-wise correspondence. They reported the error rate of 0.52%.

This study focuses on the accuracy performance of deformable approaches motivated by above-mentioned insights of Gregory [13] and Widrow [12]. A regularization-based deformable approach, which was originally proposed by Mizukami et al. [17–19], is employed for investigating the accuracy performance. Although they reported the error rate of 0.57% in 2010 [19], their study focused on reducing the computation time and did not succeed in investigating the fundamental accuracy performance. This study reviews their proposed deformable approach, conducts several simulations for clarifying the fundamental performance including the accuracy and the sensitivity to the parameters, and demonstrates how the approach deals with the variability of 2D shapes.

Section 2 describes the algorithm of the deformable recognition approach, Section 3 describes the simulation and Section 4 gives the conclusion.

2 Algorithm

The framework of regularization theory has been successfully applied to many early vision problems including optical flow, shape from shading and so on [20]. Inspired by a very simple but powerful regularization-based stereo correspondence method [21], Mizukami et al. proposed a deformable handwritten character recognition approach [17–19]. This section overviews the procedures in the proposed approach.

2.1 Regularization-Based Displacement Computation

The two-dimensional correspondence problem between pixels of an input pattern f and a prototype pattern g is ill-posed, and then it is necessary to introduce some adequate constraints to solve it. Figure 2 illustrates the input pattern f and the prototype g, where $u(x, y)$ and $v(x, y)$ indicates the horizontal and vertical displacements at the coordinate of (x, y) on the prototype g. In order to obtain the optimal displacement function for corresponding these two patterns, a cost function to be minimized is introduced,

$$E(u, v) = P(u, v) + \lambda S(u, v), \tag{1}$$

$$P(u, v) = \iint (f(x + u, y + v) - g(x, y))^2 dxdy, \tag{2}$$

$$S(u, v) = \iint (u_x^2 + u_y^2 + v_x^2 + v_y^2) dxdy, \tag{3}$$

where P is the Euclidean distance with considering the computed displacement, S is a stabilizing functional which imposes a smoothness constraint on (u, v), and λ is a so-called regularization parameter controlling the effect of S. The subscripts, x and y, are the derivative operation of horizontal and vertical directions, respectively.

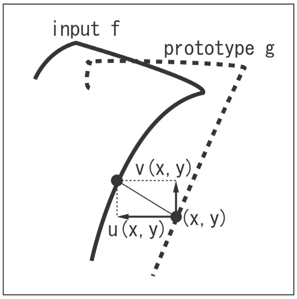


Fig. 2. Displacement function (u, v)

By applying calculus of variations to Eq. (1), the following iterative equations are obtained,

$$u^{[t+1]}(x,y) = \bar{u}^{[t]} - \frac{1}{4\lambda} f_x(x+\bar{u}^{[t]}, y+\bar{v}^{[t]})(f(x+\bar{u}^{[t]}, y+\bar{v}^{[t]}) - g(x,y)), \quad (4)$$

$$v^{[t+1]}(x,y) = \bar{v}^{[t]} - \frac{1}{4\lambda} f_y(x+\bar{u}^{[t]}, y+\bar{v}^{[t]})(f(x+\bar{u}^{[t]}, y+\bar{v}^{[t]}) - g(x,y)), \quad (5)$$

where the superscript t is the number of iteration $(1 \leq t \leq T)$, and $\bar{u}$ and $\bar{v}$ are the average of u and v over the predefined neighborhoods of (x,y), respectively. In order to make the computation more stable, $\bar{u}$ and $\bar{v}$ are employed not only in the first term of the right hand but also in the second term instead of u and v [17]. The first term smooths the displacement and the second term gives the deforming force so as to overlap two images based on the derivatives of f and the difference between the corresponding pixels on f and g.

2.2 Coarse-to-Fine Strategy with Distance Maps

For the fast convergence and preventing from being trapped in local minima, a coarse-to-fine strategy with multi-resolution images is utilized. In this study, the number of stage was set to 3. The original size of the pattern images is assumed as 32 × 32 pixels and then the n-th stage deals with the size of $2^{n+2} \times 2^{n+2}$ $(1 \leq n \leq 3)$. The displacement function obtained at the n-th stage will be used for preparing the initial displacement at the $n+1$-th stage.

Since the derivatives of the pattern images are zero in the background area, most of the pixel area on the image will not have the deforming force, resulting in an inefficient displacement computation. To overcome this problem, the binary images are generated by a threshold processing and then they are translated to the distance maps whose pixel value indicates the distance to the nearest foreground pixel as shown in Fig. 3. Instead of the original pattern images, these distance maps are used for computing the displacement [19].

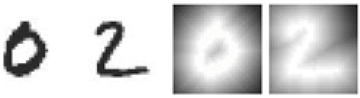


Fig. 3. Gray pattern images and distance maps

2.3 Prototype-Parallel Displacement Computation

Although the iterative equations seem to be efficiently implemented on GPUs due to its locally-parallel computation style, the sizes of the pattern images are too small to bring out the most latent strength of GPUs. To overcome this problem, prototype-parallel displacement computation (PPDC) is employed [18]. Figure 4 describes the diagram of PPDC. Three types of large plates, ($U_n^{[0]}$, $V_n^{[0]}$), F_n, and G_n, are generated by arranging displacement functions ($u_{n,l}^{[0]}$,$v_{n,l}^{[0]}$), input images f_n and prototypes $g_{n,l}$ at regular intervals respectively on the host memory, and then these plates are transferred collectively to the device memory, where l is the index of the prototype ($1 \leq l \leq L$). GPU computes the displacement for multiple pairs of input and prototype images, and updates the displacement function plate. Finally the plates of the computed displacement function ($U_n^{[T]}$, $V_n^{[T]}$) are transferred back from the device memory to the host memory.

2.4 Distance Measurement and Classification

Since the shape of a pattern will be deformed so as to be fitted to the other pattern shape by considering the computed displacement, there is a risk that the shapes of two patterns in different classes also become similar, resulting in very small distance. To avoid this problem, the local shape information of the original contours such as straight line or curvature is utilized in measuring the distance. Therefore, the distance between two pattern images is measured by applying the computed displacement function to eight-directional derivative images of two patterns, f^d and g^d ($1 \leq d \leq 8$; see Fig.5),

$$D(u,v) = \sum_{x,y}\sum_{d=1}^{8}(f^d(x+u,y+v) - g^d(x,y))^2. \qquad (6)$$

After the distances of the input pattern to L prototypes $\{f_l^d\}$ are measured, the input pattern is classified to one of the classes by the terms of k-nearest neighborhood method. The previous studies [18, 19] employed gradual prototype elimination (GPE) for reducing the computation time, where the candidate prototypes are gradually eliminated through the coarse-to-fine strategy. On the other hands, this study did not employ GPE for the purpose of investigating the fundamental accuracy performance, since GPE might eliminate several prototypes which are helpful for classifying the given input pattern.

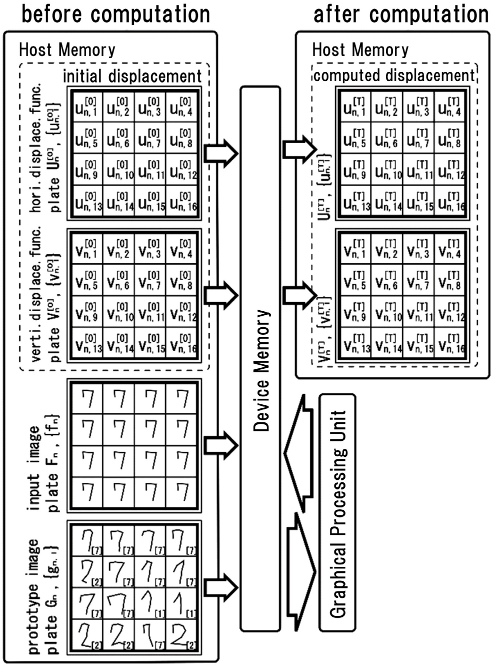

before computation
after computation
Host Memory
initial displacement
hori. displace. func. plate $U_n^{[0]}$, $\{u_n^{[0]}\}$
verti. displace. func. plate $V_n^{[0]}$, $\{v_n^{[0]}\}$
input image plate F_n, $\{f_n\}$
prototype image plate G_n, $\{g_{n,i}\}$
Host Memory
computed displacement
$U_n^{[T]}$, $\{u_n^{[T]}\}$
$V_n^{[T]}$, $\{v_n^{[T]}\}$
Device Memory
Graphical Processing Unit

Fig. 4. Prototype-parallel displacement computation

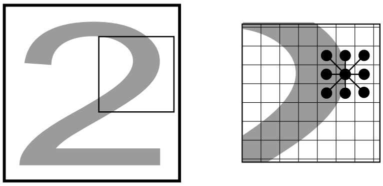


Fig. 5. Eight-directional derivatives

3 Simulation

This section investigates the effect of the regularization parameters, and the effect of k on the accuracy performance, and discusses how patterns are deformed. The deformable approach was implemented on a computer with Intel Core$^{\text{TM}}$ i7 CPU-2600 (3.40GHz) and Nvidia Geforce GTX590. The programming environments were Microsoft Visual C++ 2010 and CUDA Toolkit 4.0 on Microsoft Windows 7. MNIST handwritten digit database consists of 10,000 input patterns and 60,000 prototypes. The regularization parameter λ_1 was set to 6.0, while λ_2 and λ_3 were set to [0.8:1.2]. The computation time for recognizing an input pattern was about 40 sec, where the displacement functions are computed for all of 60,000 prototypes.

Table 1. Error rates using 4-pixel average

λ_3	λ_2				
	0.8	0.9	1.0	1.1	1.2
0.8	$\underline{0.45}$	$\underline{0.43}$	$\underline{0.43}$	$\underline{0.43}$	$\underline{0.44}$
0.9	$\underline{0.45}$	$\underline{0.44}$	$\underline{\underline{0.42}}$	$\underline{\underline{0.42}}$	$\underline{0.43}$
1.0	$\underline{0.44}$	$\underline{0.44}$	$\underline{0.45}$	$\underline{0.43}$	$\underline{0.44}$
1.1	$\underline{0.45}$	0.47	0.47	$\underline{0.45}$	$\underline{0.44}$
1.2	0.48	0.50	0.49	0.46	0.46

Table 2. Error rates using 5-pixel average

λ_3	λ_2				
	0.8	0.9	1.0	1.1	1.2
0.8	0.49	0.49	0.49	0.48	0.48
0.9	0.46	$\underline{0.45}$	$\underline{0.45}$	$\underline{0.45}$	$\underline{0.45}$
1.0	$\underline{0.43}$	$\underline{\underline{0.42}}$	$\underline{\underline{0.42}}$	$\underline{0.43}$	$\underline{0.43}$
1.1	0.46	0.46	$\underline{0.45}$	$\underline{0.44}$	$\underline{0.44}$
1.2	0.46	$\underline{0.45}$	$\underline{0.44}$	$\underline{0.44}$	$\underline{0.44}$

Table 1 and 2 show the error rates of the deformable approach by using 4-pixel and 5-pixel averages, where 4-pixel average means that the four-pixel displacements surrounding the coordinate of (x,y) were used for obtaining $\bar{u}$ and $\bar{v}$, while 5-pixel average means that the surrounding four-pixel displacements and the displacement at (x, y) were used. The value of k was set to 3. In both case of 4 and 5-pixel averages, the minimum error rate was 0.42% as underlined by double line. The error rates, which are equal to 0.45% or less, are underlined by single line. It can be noticed that the accuracy is not so sensitive for averaging ways and the value of both λ_2 and λ_3.

Figure 6 illustrates the effect of k on the accuracy performance, where both λ_2 and λ_3 were set to 1.0 and 5-pixel average was used. The minimum error 0.42% was obtained by using $k = 3$. It was also confirmed that the use of $k = 3$ was adequate for other values of λ_2 and λ_3.

Figure 7 shows several input patterns and prototypes deformed to each input pattern. Fig.7(a) illustrates 10 prototypes, while Fig.7(b,c,d) show input patterns of '7', '2' and '1' and the 10 deformed prototypes. The numbers under the deformed prototypes mean the distance to the corresponding input pattern.

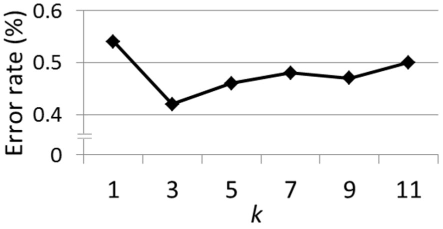
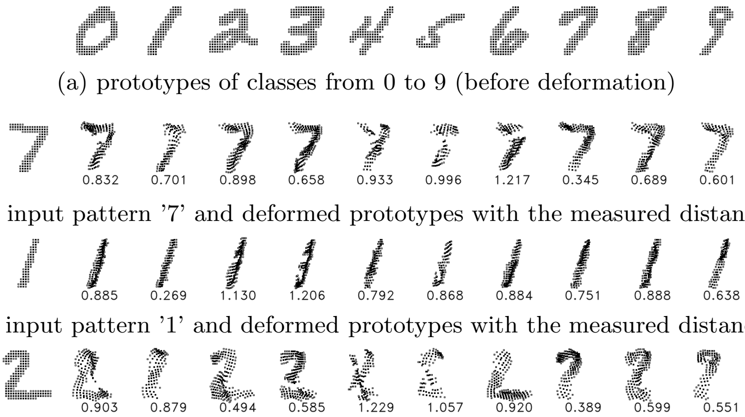


Fig. 6. Value of k versus accuracy



(a) prototypes of classes from 0 to 9 (before deformation)

(b) input pattern ’7’ and deformed prototypes with the measured distance

(c) input pattern ’1’ and deformed prototypes with the measured distance

(d) input pattern ’2’ and deformed prototypes with the measured distance

Fig. 7. Input patterns and deformed prototypes

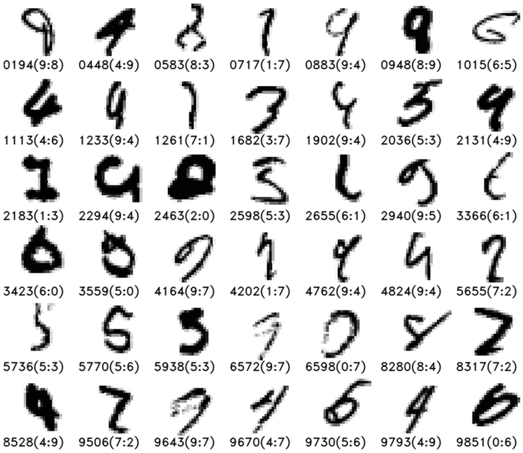


Fig. 8. Misclassified patterns with the index, the label and the misclassified label

It can be noticed that, in Fig.7(b), the deformation procedure also made the shapes of different-class prototypes of ’1’, ’2’, ’3’, ’8’ and ’9’ very similar to the shape of the input pattern ’7’, but the prototype ’7’ gave very smaller distance 0.345 than other prototypes. The same observation can be applied to the input pattern ’1’ in Fig.7(c). On the other hand, Fig.7(d) shows an interesting situation where the prototype ’7’ gave the smallest distance to the input pattern ’2’ due to the topological shape difference between the input pattern ’2’ and the prototype ’2’. These results indicate that the quality or variation of the prepared prototypes should be important for the accurate recognition.

Figure 8 shows all the misclassified input patterns with their indices, labels and misclassified labels. These patterns seem to be very challenging since they were written in rough ways and some of them lack a part of the shape due to the pale pixel value.

4 Conclusion

This study clarified the accuracy performance of a deformable handwritten recognition approach for digit characters. The deformable approach consists of regularization-based displacement computation, coarse-to-fine strategy, distance measurement and k-nearest neighborhood method. We focused on several conditions for investigating the recognition accuracy and the sensitivity, that is, the definition of averaging area in regularization process, regularization parameters and the number of k for k-nearest neighborhood method. According to the simulation results, it was shown that the proposed method has the error rate of 0.42% for MNIST handwritten digit database, resulting in the top-group of the performances reported until now. It was also shown that the accuracy performance is not so sensitive to the condition including the parameter settings. The simulation results illustrated how the prototypes are deformed to the input pattern and explained that even though the prototype becomes deformed very similar to the input pattern, the proposed approach gives adequate distances, and that the topologically-different prototype will not help the accurate recognition.

Although the obtained error rate of 0.42% is still inferior to the previous record of 0.23% obtained by multi-column deep convolutional neural network (MCDNN) [6, 7] or the best record of 0.19% by a hybrid CNN-SVM classifier [11], it should be noticed that deformable approaches are totally different from both of them. They deal with the variability of 2D shapes by generating a lot of artificial training patterns and employing very sophisticated concepts of network learning or SVM, while the deformable approaches deal with the variability by flexibly fitting the prototypes in the memory to the input pattern. As pointed out by Gregory [13] and Widrow [12], deformable approaches have some aspects which resemble to the recognition model of human brain and then there seems to be still a possibility that the deformable approaches will give further performance in the future studies.

References

1. Plamondon, R., Srihari, S.N.: On-Line and off-Line handwriting recognition: a comprehensive survey. IEEE Trans. Pattern Anal. Mach. Intell. 22(1), 63–84 (2000)
2. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proceedings of the IEEE 86(11), 2278–2324 (1998)
3. Simard, P.Y., Steinkraus, D., Platt, J.C.: Best practices for convolutional neural networks applied to visual document analysis. In: 7th International Conference on Document Analysis and Recognition, pp. 958–963 (2003)

4. Ranzato, M., Boureau, Y.-L., LeCun, Y.: Sparse feature learning for deep belief networks. In: Advances in Neural Information Processing Systems, vol. 20, pp. 1–8 (2007)
5. Fukushima, K.: Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. Biological Cybernetics 36(4), 193–202 (1980)
6. Ciresan, D., Meier, U., Gambardella, L.M., Schmidhuber, J.: Convolutional neural network committees for handwritten character classification. In: 11th International Conference on Document Analysis and Recognition, pp. 1250–1254 (2011)
7. Ciresan, D., Meier, U., Schmidhuber, J.: Multi-column deep neural networks for image classification. In: International Conference on Computer Vision and Pattern Recognition, pp. 3642–3649 (2012)
8. Vapnik, V.N.: The Nature of Statistical Learning Theory. Springer (1995)
9. DeCoste, D., Scholkopf, B.: Training invariant support vector machines. Machine Learning 46, 161–190 (2002)
10. Liu, C., Nakashima, K., Sako, H., Fujisawa, H.: Handwritten digit recognition: benchmarking of state-of-the-art techniques. Pattern Recognition 36(10), 2271–2285 (2003)
11. Niu, X.X., Suen, Y.: A novel hybrid CNN-SVM classifier for recognizing handwritten digits. Pattern Recognition 45(4), 1318–1325 (2012)
12. Widrow, B.: The 'Rubber-Mask' Technique - I. Pattern Measurement and Analysis. Pattern Recognition 5(3), 175–198 (1973)
13. Gregory, R.L.: Eye and brain, the Psychology of Seeing. World University Library. McGraw-Hill, New York (1966)
14. Belongie, S., Malik, J., Puzicha, J.: Shape matching and object recognition using shape contexts. IEEE Trans. Pattern Anal. Mach. Intell. 24(4), 509–522 (2002)
15. Keysers, D., Deselaers, T., Gollan, C., Ney, H.: Deformation models for image recognition. IEEE Trans. Pattern Anal. Mach. Intell. 29(8), 1422–1435 (2007)
16. Uchida, S., Sakoe, H.: A monotonic and continuous two-dimensional warping based on dynamic programming. In: 14th International Conference on Pattern Recognition, vol. 1, pp. 521–524 (1998)
17. Mizukami, Y., Koga, K.: A handwritten Character recognition system using hierarchical displacement extraction algorithm. In: 13th International Conference on Pattern Recognition, vol. 3, pp. 160–164 (1996)
18. Mizukami, Y., Tadamura, K.: GPU implementation of deformable pattern recognition using prototype-parallel displacement computation. In: International Workshop on Image Registration in Deformable Environments - DEFORM 2006, vol. 1, pp. 71–80 (2006)
19. Mizukami, Y., Tadamura, K., Warrell, J., Li, P., Prince, S.: CUDA implementation of deformable pattern recognition and its application to MNIST handwritten digit database. In: 20th International Conference on Pattern Recognition, vol. 1, pp. 2001–2004 (2010)
20. Poggio, T., Torre, V., Koch, C.: Computational vision and regularization theory. Nature 317, 314–319 (1985)
21. March, R.: Computation of stereo disparity using regularization. Pattern Recognition Letters 8(3), 181–188 (1988)