Local Tensor Descriptor from Micro-deformation Analysis

Bangsheng Cheng
Biomedical Engineering Department & Interdisciplinary Lab of Physics Department
Zhejiang University, Hangzhou 310027, China
chengbs@zju.edu.cn

Abstract

This paper proposes a novel method called micro-deformation analysis to analyze and describe local image structures. This method is a general analytic tool and can be applied to any high-dimensional scalar or vector functions. We derive the tensor matrix from this method as the descriptor to represent the information within local image patches. Our experimental results suggest that we can design low-dimensional local tensor descriptors with performance comparable to the popular SIFT descriptor, which is the state-of-the-art feature descriptor used for object recognition and categorization.

1. Introduction

Representing local features by compact and distinctive descriptors is important for many computer vision tasks. Traditional techniques, such as local jets [6] and Gabor filters [1], analyze and represent local geometric structures by convolving images with certain filter banks or receptive fields. Current techniques improve the descriptors' performance by summarizing these filtered responses over a local image region. SIFT (Scale Invariant Feature Transform) descriptor proposed by Lowe [7, 8] provides a mechanism to construct distinctive descriptors by summarizing the gradient orientations to a histogram with certain spatial configuration. SIFT-like descriptors [14, 10] have proved to be the state-of-the-art feature descriptors for object recognition and categorization. But their high dimensions make the computational cost high.

Unlike the aforementioned methods, we propose a different method to analyze and describe local image structures. When we apply a deformation to an image patch, the change caused by this deformation depends on the geometric structure within this image patch. If we apply a group of deformations on this patch, the changes caused by them define a function on the deformation space, and this function characterize the geometric structure within the patch. So we can use this function to identify the local image structure.

When we constraint these deformations to be micro-deformations with infinitesimal parameters, the function becomes a tensor matrix, and this leads to the idea of using this tensor matrix to represent the information contained in the image patch.

In this paper, we formulate this micro-deformation analysis method, and derive the tensor matrix from it as the descriptor to represent local image structures. We propose a method to normalize these tensor matrixes to make them isotropic in their distance space. We also investigate the tensor descriptors' properties by experiments and compare them with SIFT descriptor.

This paper is organized as follows: in section 2, we develop the micro-deformation analysis method and derive the normalized tensor descriptors. In section 3, we investigate the properties of local tensor descriptors by experiments, and compare them with SIFT descriptor. In the last section, we conclude this paper and discuss some future works.

1.1. Related works

Investigating local image properties by applying small deformations to local patches and studying the changes caused by these deformations can be traced back to the works of Moravec [12] and Hannah [2]. This line of research is the basis of interest point detection. The original deformations applied to local patches are translations in different directions. Based on this simple deformation model, the structure tensor matrix (second moment matrix) [3] was deduced for corner detection. Translational model was extended to affine deformation model later and a tensor matrix based on this model was derived for general corner detection[5, 13]. All these works were restricted to derive the properties of the second moment matrix's eigenvalues for corner detection.

2. Tensor matrix as local descriptor

2.1. Micro-deformation analysis

The main idea of our method can be described as follows: An image patch is sent into a machine which applies

978-1-4244-2243-2/08/$25.00 ©2008 IEEE

various small deformations to this patch and the changes caused by these deformations are measured. The relationship between the image changes and the deformations captures the structure information contained in this patch, so we can use this relationship to identify the content of this image patch.

We can parameterize the deformations by $\mathbf{p} = (p_1, \cdots, p_n)^T$ to make small $||\mathbf{p}||$ value corresponds to small deformation, and $\mathbf{p} = \mathbf{0}$ corresponds to no deformation. The relationship between the patch change E and the deformation $\mathbf{p}$ can be represented by a function:

$$E = f(\mathbf{p}) \tag{1}$$

The value of E may be interpreted as the energy needed to accomplish this deformation. For small deformation parameterized by $\delta\mathbf{p}$, this function can be approximated by its Taylor series:

$$E = f(\mathbf{0}) + \nabla_{\mathbf{p}}^T f(\mathbf{0})\delta\mathbf{p} + \frac{1}{2}\delta\mathbf{p}^T H_{\mathbf{p}} f(\mathbf{0})\delta\mathbf{p} + \cdots \tag{2}$$

where $\nabla_{\mathbf{p}}$ and $H_{\mathbf{p}}$ denote the gradient and Hessian matrix with respect to $\mathbf{p}$. Because $f(\mathbf{0}) = 0$ and $E \geq 0$, so $\nabla_{\mathbf{p}}^T f(\mathbf{0}) = 0$, and we get:

$$E = \frac{1}{2}\delta\mathbf{p}^T H_{\mathbf{p}} f(\mathbf{0})\delta\mathbf{p} + o(||\delta\mathbf{p}||^2) \tag{3}$$

So we can use the Hessian matrix $H_{\mathbf{p}} f(\mathbf{0})$ as the descriptor to describe the structure within an image patch.

Micro-deformation analysis is a general analytic tool which can be applied to high-dimensional scalar or vector functions. But in this paper, we restrict our discussions on two dimensional gray-value images.

2.2. Tensor matrix from micro-deformations

A deformation applied to an image patch W can be represented by a displacement field $\delta\mathbf{x} = (\alpha(\mathbf{x};\mathbf{p}), \beta(\mathbf{x};\mathbf{p}))^T$ which reflects the displacement of each pixel $\mathbf{x} = (x, y)^T$ in this patch. Where $\mathbf{p} = (p_1, \cdots, p_n)^T$ denotes the parameters used to control the deformation. If small $||\mathbf{p}||$ value corresponds to small deformation, then $\alpha(\mathbf{x};\mathbf{0}) = 0$ and $\beta(\mathbf{x};\mathbf{0}) = 0$. For a small deformation with parameters $\delta\mathbf{p}$ we get:

$$\begin{aligned} \delta\mathbf{x} &= \begin{pmatrix} \alpha(\mathbf{x};\delta\mathbf{p}) \\ \beta(\mathbf{x};\delta\mathbf{p}) \end{pmatrix} \\ &= \begin{pmatrix} \alpha(\mathbf{x};\mathbf{0}) + \nabla_{\mathbf{p}}^T \alpha(\mathbf{x};\mathbf{0})\delta\mathbf{p} + o(||\delta\mathbf{p}||) \\ \beta(\mathbf{x};\mathbf{0}) + \nabla_{\mathbf{p}}^T \beta(\mathbf{x};\mathbf{0})\delta\mathbf{p} + o(||\delta\mathbf{p}||)) \end{pmatrix} \\ &\approx \begin{pmatrix} \nabla_{\mathbf{p}}^T \alpha(\mathbf{x};\mathbf{0}) \\ \nabla_{\mathbf{p}}^T \beta(\mathbf{x};\mathbf{0}) \end{pmatrix} \delta\mathbf{p} \\ &= M_\delta(\mathbf{x})\delta\mathbf{p} \end{aligned} \tag{4}$$

where $\nabla_{\mathbf{p}}$ denotes the gradient with respect to the parameter $\mathbf{p}$ and $M_\delta(\mathbf{x}) = (\nabla_{\mathbf{p}}\alpha(\mathbf{x};\mathbf{0}), \nabla_{\mathbf{p}}\beta(\mathbf{x};\mathbf{0}))^T$. We call $M_\delta(\mathbf{x})$ as the deformation matrix. There are two ways to define this deformation of interest: We can define two functions $\alpha(\mathbf{x};\mathbf{p})$ and $\beta(\mathbf{x};\mathbf{p})$, and then derive $M_\delta(\mathbf{x})$ from these functions as formula (4) did. Or we can define $M_\delta(\mathbf{x})$ directly.

According to formula (4), we can decompose $\delta\mathbf{x}$ as follows:

$$\delta\mathbf{x} = \sum_{k=1}^{n} \delta p_k \begin{pmatrix} \alpha_k(\mathbf{x};\mathbf{0}) \\ \beta_k(\mathbf{x};\mathbf{0}) \end{pmatrix} \tag{5}$$

where $\alpha_k(\cdot)$ and $\beta_k(\cdot)$ $(1 \leq k \leq n)$ denote the partial derivatives of $\alpha(\mathbf{x};\mathbf{p})$ and $\beta(\mathbf{x};\mathbf{p})$ with respect to the k-th parameter component p_k. We can regard $(\alpha_k(\mathbf{x};\mathbf{0}), \beta_k(\mathbf{x};\mathbf{0}))^T$ as the deformation bases defined on the image patch. So we can define n deformation bases $B_1(\mathbf{x}), \cdots, B_n(\mathbf{x})$, and generate a linear micro-deformation space as follows:

$$\delta\Omega_D = \{\sum_{k=1}^{n} \delta p_k B_k(\mathbf{x}) | \delta p_k \in R\} \tag{6}$$

These deformation bases constitute a deformation matrix:

$$M_\delta(\mathbf{x}) = (B_1(\mathbf{x}), \cdots, B_n(\mathbf{x})) \tag{7}$$

We can use the sum-of-squared-difference (SSD) to measure the change E produced by the deformation $\delta\mathbf{x}$:

$$E = \sum_W w(\mathbf{x})(I(\mathbf{x} + \delta\mathbf{x}) - I(\mathbf{x}))^2 \tag{8}$$

where $w(\mathbf{x})$ is a kernel function used to weight the image patch. We set $w(\mathbf{x})$ be the Gaussian kernel with the deviation proportional to the size of the image patch.

Because the displacement $\delta\mathbf{x}$ is very small for each pixel, we can approximate the change of each pixel as follows:

$$I(\mathbf{x} + \delta\mathbf{x}) - I(\mathbf{x}) \approx \nabla_{\mathbf{x}}^T I \delta\mathbf{x} \tag{9}$$

where $\nabla_{\mathbf{x}}$ denotes the spatial gradient. Hence we obtain the following approximation:

$$E \approx \sum_W w(\mathbf{x})\delta\mathbf{x}^T(\nabla_{\mathbf{x}} I \nabla_{\mathbf{x}}^T I)\delta\mathbf{x} \tag{10}$$

Substitute formula (4) into (10), we get:

$$\begin{aligned} E &= \delta\mathbf{p}^T \sum_W (w(\mathbf{x}) M_\delta^T(\mathbf{x})(\nabla_{\mathbf{x}} I \nabla_{\mathbf{x}}^T I) M_\delta(\mathbf{x}))\delta\mathbf{p} \\ &= \delta\mathbf{p}^T M_d \delta\mathbf{p} \end{aligned} \tag{11}$$

where $M_d = \sum_W w(\mathbf{x}) M_\delta^T(\mathbf{x})(\nabla_\mathbf{x} I \nabla_\mathbf{x}^T I) M_\delta(\mathbf{x})$. M_d is the tensor matrix that reflects the relationship between the change E of the patch W and the small deformation parameterized by $\delta\mathbf{p}$. M_d is a semi-definite symmetric matrix and captures the structure information of this patch, so we can use it to describe local image structures.

2.3. Tensor matrix normalization

The components of the deformation parameter $\mathbf{p} = (p_1, \cdots, p_n)^T$ reflect different factors controlling the amount of displacement of each pixel in the image patch. These factors are generally not equivalent, e.g. translational displacement is not equivalent to rotational angle, so a unit change of different components may cause different amount of total displacement of the image patch. This makes the components of the tensor matrix M_d are not equivalent for distance calculation. In order to make the tensor matrix isotropic in its distance space, we need to normalize it to make a unit change of each parameter component p_i $(1 \leq i \leq n)$ produces the same unit displacement of the image patch.

If we set the k-th component $(1 \leq k \leq n)$ be unit change ϵ, and other components be 0, Then $\delta p_k = \epsilon$ and $\delta p_i = 0$ $(i \neq k)$. ϵ is a small value. Then based on formula (5), we get:

$$\delta \mathbf{x}_k = \epsilon \begin{pmatrix} \alpha_k(\mathbf{x};\mathbf{0}) \\ \beta_k(\mathbf{x};\mathbf{0}) \end{pmatrix} \tag{12}$$

The total displacement of the image patch W is:

$$\begin{aligned} \delta_k &= \sum_W w(\mathbf{x}) ||\delta \mathbf{x}_k|| \\ &= \epsilon \sum_W w(\mathbf{x}) \sqrt{\alpha_k^2(\mathbf{x};\mathbf{0}) + \beta_k^2(\mathbf{x};\mathbf{0})} \\ &= \epsilon N_k \end{aligned} \tag{13}$$

where $N_k = \sum_W w(\mathbf{x}) \sqrt{\alpha_k^2(\mathbf{x};\mathbf{0}) + \beta_k^2(\mathbf{x};\mathbf{0})}$. So we can normalize the k-th components of $M_\delta(\mathbf{x})$ by dividing them by N_k for image patch W. Then a unit change of the k-th parameter component p_k produces a unit displacement of the image patch W. The normalized deformation matrix M_δ^n is:

$$M_\delta^n = \begin{pmatrix} \frac{\alpha_1(\mathbf{x};\mathbf{0})}{N_1} & \cdots & \frac{\alpha_k(\mathbf{x};\mathbf{0})}{N_k} & \cdots & \frac{\alpha_n(\mathbf{x};\mathbf{0})}{N_n} \\ \frac{\beta_1(\mathbf{x};\mathbf{0})}{N_1} & \cdots & \frac{\beta_k(\mathbf{x};\mathbf{0})}{N_k} & \cdots & \frac{\beta_n(\mathbf{x};\mathbf{0})}{N_n} \end{pmatrix} \tag{14}$$

And the normalized tensor matrix can be calculated as follows:

$$M_d^n = \sum_W w(\mathbf{x}) (M_\delta^n)^T(\mathbf{x})(\nabla_\mathbf{x} I \nabla_\mathbf{x}^T I) M_\delta^n(\mathbf{x}) \tag{15}$$

In order to make the tensor descriptor be invariant to affine intensity transformation, we can normalize the above tensor matrix M_d^n by $\sum_W w(\mathbf{x}) \nabla_\mathbf{x}^T I \nabla_\mathbf{x} I$. So the final local tensor descriptor d_M for an image patch W is:

$$d_M = \frac{\sum_W w(\mathbf{x})(M_\delta^n)^T(\mathbf{x})(\nabla_\mathbf{x} I \nabla_\mathbf{x}^T I) M_\delta^n(\mathbf{x})}{\sum_W w(\mathbf{x})(\nabla_\mathbf{x}^T I \nabla_\mathbf{x} I)} \tag{16}$$

2.4. Polynomial deformation models and derived tensor descriptors

The deformation model can be defined by the functions $\alpha(\mathbf{x};\mathbf{p})$ and $\beta(\mathbf{x};\mathbf{p})$ which determine the displacement field of the image patch. We define the n-th order polynomial deformation model as follows:

$$\alpha(\mathbf{x};\mathbf{p}) = \sum_{i \geq 0, j \geq 0, i+j \leq n} p_{ij}^\alpha x^i y^j \tag{17}$$

$$\beta(\mathbf{x};\mathbf{p}) = \sum_{i \geq 0, j \geq 0, i+j \leq n} p_{ij}^\beta x^i y^j \tag{18}$$

where $\mathbf{p} = (\cdots, p_{ij}^\alpha, \cdots, p_{ij}^\beta, \cdots)^T$. The parameters p_{ij}^α related to $\alpha(\mathbf{x};\mathbf{p})$ are different from the parameters p_{ij}^β related to $\beta(\mathbf{x};\mathbf{p})$, and this makes $\alpha(\mathbf{x};\mathbf{p})$ and $\beta(\mathbf{x};\mathbf{p})$ independent. The above deformation model makes the displacement of each pixel is a polynomial function of its coordinates, and this defines a general class of deformations used to probe the structure within an image patch. Then the deformation matrix is:

$$M_\delta(\mathbf{x}) = \begin{pmatrix} \cdots & x^i y^j & \cdots & \mathbf{0} & 0 & \mathbf{0} \\ \mathbf{0} & 0 & \mathbf{0} & \cdots & x^i y^j & \cdots \end{pmatrix} \tag{19}$$

According to formula (14), we can normalize this deformation matrix for patch W as follows:

$$M_\delta^n(\mathbf{x}) = \begin{pmatrix} \cdots & \frac{x^i y^j}{N_{ij}} & \cdots & \mathbf{0} & 0 & \mathbf{0} \\ \mathbf{0} & 0 & \mathbf{0} & \cdots & \frac{x^i y^j}{N_{ij}} & \cdots \end{pmatrix} \tag{20}$$

where $N_{ij} = \sum_W w(\mathbf{x}) |x^i y^j|$. If we denote $\mathbf{a} = (\cdots, \frac{x^i y^j}{N_{ij}}, \cdots)^T$, then

$$M_\delta^n(x) = \begin{pmatrix} \mathbf{a}^T & \mathbf{0} \\ \mathbf{0} & \mathbf{a}^T \end{pmatrix} \tag{21}$$

Substitute formula (21) to formula (15), the normalized tensor matrix for n-th order polynomial deformation model can be calculated as follows:

$$\begin{aligned} M_d^n &= \sum_W w(\mathbf{x}) \begin{pmatrix} \mathbf{a} & \mathbf{0} \\ \mathbf{0} & \mathbf{a} \end{pmatrix} (\nabla_\mathbf{x} I \nabla_\mathbf{x}^T I) \begin{pmatrix} \mathbf{a}^T & \mathbf{0} \\ \mathbf{0} & \mathbf{a}^T \end{pmatrix} \\ &= \sum_W w(\mathbf{x}) \begin{pmatrix} A I_x^2 & A I_x I_y \\ A I_x I_y & A I_y^2 \end{pmatrix} \end{aligned} \tag{22}$$

where $A = \mathbf{a}\mathbf{a}^T$ is a symmetrical matrix.

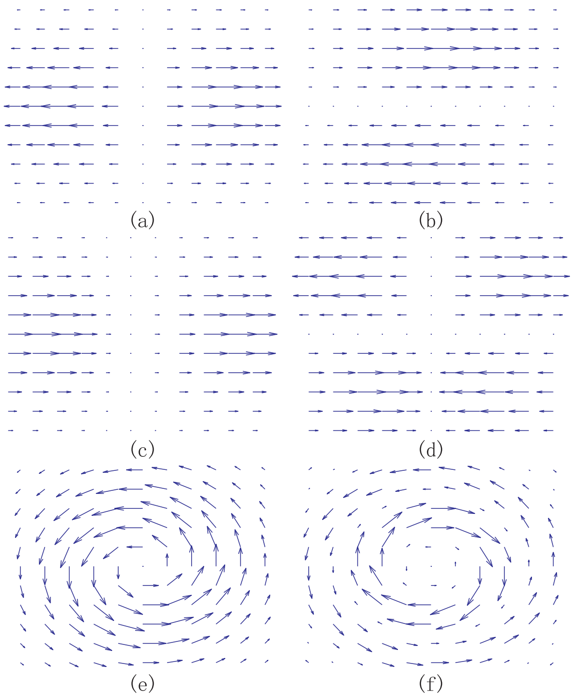


Figure 1. Gaussian weighted displacement fields for some normalized deformation bases. (a) $\delta\mathbf{x} = (x, 0)^T$. (b) $\delta\mathbf{x} = (y, 0)^T$. (c) $\delta\mathbf{x} = (x^2, 0)^T$. (d) $\delta\mathbf{x} = (xy, 0)^T$. (e) $\delta\mathbf{x} = (-y, x)^T$. (f) $\delta\mathbf{x} = (-y\sin(wr), x\sin(wr))^T$.

2.4.1 Zero-th order deformation: translational deformation

For the zero-th order polynomial deformation model, $\alpha(\mathbf{x}; \mathbf{p}) = p_1$ and $\beta(\mathbf{x}; \mathbf{p}) = p_2$, where $\mathbf{p} = (p_1, p_2)^T$. This is the translational deformation with displacement $(p_1, p_2)^T$ independent of the coordination $\mathbf{x}$. The deformation matrix $M_\delta(\mathbf{x})$ is the 2×2 identity matrix. The tensor matrix is the second moment matrix:

$$\begin{aligned} M_d &= \sum_W w(\mathbf{x}) \nabla_{\mathbf{x}} I \nabla_{\mathbf{x}}^T I \\ &= \sum_W w(\mathbf{x}) \begin{pmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{pmatrix} \end{aligned} \tag{23}$$

2.4.2 First order deformation: affine deformation

For the first order polynomial deformation model, $\alpha(\mathbf{x}; \mathbf{p}) = p_1 x + p_2 y + p_3$ and $\beta(\mathbf{x}; \mathbf{p}) = p_4 x + p_5 y + p_6$, this is the affine deformation parameterized by $\mathbf{p} = (p_1, p_2, \cdots, p_6)^T$ as follows:

$$\delta\mathbf{x} = \begin{pmatrix} p_1 & p_2 \\ p_4 & p_5 \end{pmatrix} \mathbf{x} + \begin{pmatrix} p_3 \\ p_6 \end{pmatrix} \tag{24}$$

So the deformation matrix is

$$M_\delta = \begin{pmatrix} x & y & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x & y & 1 \end{pmatrix} \tag{25}$$

According to formula (21), we have

$$\mathbf{a} = \left(\frac{x}{N_1}, \frac{y}{N_2}, \frac{1}{N_3} \right) \tag{26}$$

where $N_1 = \sum_W w(\mathbf{x})|x|$, $N_2 = \sum_W w(\mathbf{x})|y|$ and $N_3 = \sum_W w(\mathbf{x})$.

Then we can calculate $A = \mathbf{a}^T\mathbf{a}$ and the normalized tensor matrix M_d^n according to formula (22). M_d^n is a 6×6 symmetric matrix. Because A is a 3×3 symmetric matrix, M_d^n only has 18 different elements.

2.4.3 Second order deformation

For the second order polynomial deformation model, $\alpha(\mathbf{x}; \mathbf{p}) = p_1 x^2 + p_2 xy + p_3 y^2 + p_4 x + p_5 y + p_6$ and $\beta(\mathbf{x}; \mathbf{p}) = p_7 x^2 + p_8 xy + p_9 y^2 + p_{10} x + p_{11} y + p_{12}$, where $\mathbf{p} = (p_1, \cdots, p_{12})^T$. The deformation matrix is

$$M_\delta = \begin{pmatrix} x^2 & \cdots & 1 & 0 & \cdots & 0 \\ 0 & \cdots & 0 & x^2 & \cdots & 1 \end{pmatrix} \tag{27}$$

and

$$\mathbf{a} = \left(\frac{x^2}{N_1}, \frac{xy}{N_2}, \frac{y^2}{N_3}, \frac{x}{N_4}, \frac{y}{N_5}, \frac{1}{N_6} \right)^T \tag{28}$$

where $N_1 = \sum_W w(\mathbf{x})|x|^2$, $N_2 = \sum_W w(\mathbf{x})|xy|$, $N_3 = \sum_W w(\mathbf{x})|y|^2$, $N_4 = \sum_W w(\mathbf{x})|x|$, $N_5 = \sum_W w(\mathbf{x})|y|$ and $N_6 = \sum_W w(\mathbf{x})$.

We can compute $A = \mathbf{a}\mathbf{a}^T$ and the normalized tensor matrix M_d^n according to formula (22). M_d^n is a 12×12 symmetric matrix. Because A is a symmetric 6×6 matrix, M_d^n has 63 different elements.

Figure 1 (a-d) show the Gaussian weighted normalized displacement fields for some polynomial deformation bases.

2.5. Other example deformation models

This section gives some deformation fields not included in the aforementioned polynomial deformation models to demonstrate the flexibility of micro-deformation analysis.

2.5.1 Translational + rotational deformation

If we deform an image region by small rotation angle $\delta\theta$ around its center and small translational displacement $(\delta p_1, \delta p_2)^T$, then the displacement $\delta\mathbf{x}$ for pixel $\mathbf{x}$ is:

$$\begin{aligned}\delta\mathbf{x} &= \begin{pmatrix} 1-\cos(\delta\theta) & -\sin(\delta\theta) \\ \sin(\delta\theta) & 1-\cos(\delta\theta) \end{pmatrix}\mathbf{x} + \begin{pmatrix} \delta p_1 \\ \delta p_2 \end{pmatrix} \\ &\approx \begin{pmatrix} 0 & -\delta\theta \\ \delta\theta & 0 \end{pmatrix}\mathbf{x} + \begin{pmatrix} \delta p_1 \\ \delta p_2 \end{pmatrix} \qquad (29)\end{aligned}$$

The deformation parameter is $\delta\mathbf{p} = (\delta\theta, \delta p_1, \delta p_2)^T$, and the deformation matrix is $M_\delta(\mathbf{x}) = \begin{pmatrix} -y & 1 & 0 \\ x & 0 & 1 \end{pmatrix}$.

The normalized deformation matrix is:

$$M_\delta^n(\mathbf{x}) = \begin{pmatrix} \frac{-y}{N_1} & \frac{1}{N_2} & 0 \\ \frac{x}{N_1} & 0 & \frac{1}{N_3} \end{pmatrix} \qquad (30)$$

where $N_1 = \sum_W w(\mathbf{x})\sqrt{x^2+y^2}$ and $N_2 = N_3 = \sum_W w(\mathbf{x})$.

Figure 1(e) shows the displacement field for the deformation base $(-y, x)^T$. This deformation model is not included in the polynomial deformation models because the parameter $\delta\theta$ is related to both $\alpha(\mathbf{x}; \delta\mathbf{p})$ and $\beta(\mathbf{x}, \delta\mathbf{p})$.

2.5.2 Non-polynomial deformation model

According to formula (7), we can construct a deformation matrix by define a set of deformation bases for the image patch, and any function can be selected as a deformation base. Here we present the periodic rotational deformation functions as an example:

$$B_i(\mathbf{x}) = \begin{pmatrix} -y\sin(iwr) \\ x\sin(iwr) \end{pmatrix} \qquad (31)$$

where $r = \sqrt{x^2+y^2}$, $i \in N$, $w = \frac{2\pi}{R}$ and R is the size of the normalized image patch. Figure 1 (f) shows the displacement field for $B_1(\mathbf{x}) = (-y\sin(wr), x\sin(wr))^T$.

2.6. Distance metrics for descriptor matching

Matrix norms can be used to compute the distance between two tensor descriptors. Four kinds of norms have been defined for matrixes, they are the maximum absolute column sum norm $||M||_1$, the spectral norm $||M||_2$, the maximum absolute row sum norm $||M||_\infty$ and the Frobenius (Euclidean) norm $||M||_F$:

$$\begin{aligned} ||M||_1 &= \max_j \sum_{i=1}^n |a_{ij}| && (32) \\ ||M||_2 &= (\max_i \lambda_i)^{1/2} && (33) \\ ||M||_\infty &= \max_i \sum_{j=1}^n |a_{ij}| && (34) \\ ||M||_F &= (\sum_{i=1}^n \sum_{j=1}^n |a_{ij}|^2)^{1/2} && (35) \end{aligned}$$

where M is a $n \times n$ matrix with a_{ij} $(1 \le i, j \le n)$ as its elements, λ_i $(1 \le i \le n)$ is the eigenvalues of M^*M, and M^* is M's conjugate transpose. Because tensor matrix is symmetric, so $||M||_1 = ||M||_\infty$.

We also consider to convert each tensor descriptor to a vector consists of only its different elements, and use this vector's Euclidean norm as the distance metric. We denote this norm as $||M||_V$.

Figure 2. Example images used in our experiments.

3. Experiments

We investigate the following topics by experiments: (1) What is the suitable distance metric for tensor descriptor matching. (2) Does matrix normalization proposed in section 2.3 effective. (3) The influence of different deformation bases on tensor descriptors' performance. (4) Comparing tensor descriptors with SIFT descriptor.

3.1. Method

We experiment local tensor descriptors on a dataset of 64 images including indoor and outdoor scenes. some example images are shown in figure 2. Each image is normalized to be size 512×512. We extract the 1000 most salient keypoints from each image by the Harris scale invariant keypoint detector. We do not adapt the keypoints' locations and scales in the scale space. This introduces more noise to the keypoints' locations and scales. For each keypoint, we extract the circular image region around it with radius proportional to the keypoint's scale. We compute the dominant orientations for each image patch by the method proposed by Lowe [7, 8] and normalize each image patch to size 21×21 with respect to the dominant orientations. Then we compute local descriptors for each normalized image patch.

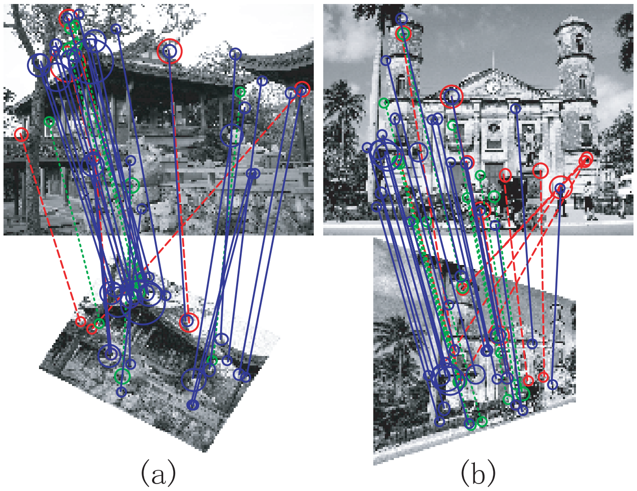


Figure 3. Image transformations and the feature correspondences recovered by matching the second-order polynomial tensor descriptors (LTD2). (The 100 most salient keypoints are extracted from each image. Solid lines indicate the recovered true correspondences. Dashed lines indicate the recovered false correspondences. Dotted lines indicate the missed true correspondences.) (a) Affine transformation with Gaussian noise corruption (denoted as T1). (b) Homography transformation with Gaussian noise corruption (denoted as T2).

3.1.1 Harris scale invariant keypoint detector

Harris scale invariant keypoint detector is based on the scale-adapted tensor matrix (second moment matrix) derived from the zero-th order polynomial micro-deformation analysis:

$$\mu(\mathbf{x},\sigma_d,\sigma_i) = \sigma_d^2 g(\sigma_i) * \begin{pmatrix} I_x^2(\mathbf{x},\sigma_d) & I_x I_y(\mathbf{x},\sigma_d) \\ I_x I_y(\mathbf{x},\sigma_d) & I_y^2(\mathbf{x},\sigma_d) \end{pmatrix} \tag{36}$$

where $g(\sigma_i)$ is a Gaussian kernel with the integration scale σ_i, $I_x(\mathbf{x},\sigma_d)$ and $I_y(\mathbf{x},\sigma_d)$ are the image gradients computed at the differentiation scale σ_d. To build a scale space representation of an image, we set $\sigma_d = k\sigma_i$. where k is a parameter used to determine the ratio between σ_d and σ_i. The Harris cornerness then can be computed as:

$$C_H = \det(\mu) - \alpha \times tr^2(\mu) \tag{37}$$

where $\det(\cdot)$ and $tr(\cdot)$ calculate the determinant and trace of a matrix respectively. α is a parameter generally set to be $0.04 \leq \alpha \leq 0.06$. The local maxima of C_H responses over scale space are candidate keypoints for feature detection.

Although C_H responses rarely attain maxima over scale space when we set $\sigma_d = 0.7\sigma_i$ as Mikolajczyk and Schmid suggested [9], our experiments show that the increase of k can greatly increase the number of C_H's local maxima in the image scale space. When $k = 1.25$, the number of keypoints extracted by Harris scale invariant detector is comparable to DoG (Difference of Gaussian) keypoint detector [8]. Experimental results also show Harris scale invariant detector gets better repeatability than DoG detector [7] and Harris-Laplace detector [9]. For SIFT descriptor [8], the keypoints extracted by Harris scale invariant detector is also better than DoG detector and Harris-Laplace detector.

So in our experiments, we use Harris scale invariant detector for feature detection, and set $\sigma_d = 1.25\sigma_i$. For each detected keypoint, we extract its surrounding circular region with radius $R_p = \beta s_k$, where s_k is the keypoint's scale and β is a parameter used to control the ratio between R_p and s_k. Large β value means large patches extracted from each keypoint, so makes them contain more image information. But large patch size also makes the extracted features easily be disturbed by 3D transformations, occlusions and objects' global variations. In our experiments, we experiment the descriptors for patches with $\beta = 6$. For each extracted image patches, we normalize them to be size 21×21.

3.1.2 Recall-precision criterion

We use the areas of recall-precision curves as the criteria [4, 11] to evaluate the descriptors' performance. For an image I_1 and its transformation $I_2 = T \cdot I_1$ under the transformation T, we can establish the correspondences between the features detected in I_1 and the features detected in I_2 by ignoring some errors. We use surface error proposed by Mikolajczyk and Schmid [9, 11] to measure the corresponding errors between two features F_1 and F_2. The surface error between them is

$$\epsilon_s = 1 - \frac{|T \cdot A_1 \cap A_2|}{|T \cdot A_1 \cup A_2|} \tag{38}$$

where A_1 and A_2 are image patches corresponding to F_1 and F_2 respectively. $T \cdot A_1 \cup A_2$ and $T \cdot A_1 \cap A_2$ are their union and intersection under I_2 coordinate frame. In our experiments, two features correspond if their surface error ϵ_s is less than 0.4.

Two descriptors are considered matched if their distance is below a threshold. The match is correct if their corresponding features correspond under the transformation T. If the number of true matches is N_t, the number of false matches is N_f, the total number of corresponding features is N_c, and the number of recovered correspondences from the true matches is N_r, then *recall* and *precision* are defined as follows:

$$recall = \frac{N_r}{N_c} \tag{39}$$

$$precision = \frac{N_t}{N_t + N_f} \tag{40}$$

In our experiments, we add 0.5% Gaussian noise to the images and then transform them by two kinds of transformations. The first one (denoted as T1) transforms images by an affine transformation. The second one (denoted as T2) transforms images by a homography transformation. Figure 3 shows the transformed images and the recovered feature correspondences by matching the second-order tensor descriptors (LTD2).

3.2. Results

3.2.1 Results for different distance metrics

We use Norm-F, Norm-1, Norm-2 and Norm-V to denote the matrix norms $||\cdot||_F$, $||\cdot||_1$, $||\cdot||_2$ and $||\cdot||_V$ respectively. For tensor descriptors, we have $||\cdot||_1 = ||\cdot||_\infty$. Polynomial tensor descriptors with order 1 (LTD1) and order 2 (LTD2) are experimented on images under the transformations T1 and T2, and the results are shown in table 1. Frobenius norm $||\cdot||_F$ gets the best results. So $||\cdot||_F$ is the most suitable distance metric for tensor descriptor matching. Frobenius norm is also very efficient for calculation. So in the following experiments, we use Frobenius norm to calculate the distance between tensor descriptors.

	LTD1		LTD2	
Norms	T1	T2	T1	T2
Norm-F	0.52(0.19)	**0.31**(0.14)	**0.66**(0.14)	**0.45**(0.13)
Norm-1	**0.53**(0.17)	**0.31**(0.13)	0.65(0.11)	0.43(0.10)
Norm-2	0.52(0.17)	0.30(0.12)	0.62(0.11)	0.40(0.09)
Norm-V	0.50(0.18)	0.28(0.13)	0.64(0.14)	0.42(0.12)

Table 1. Performance results for different distance metrics. (Descriptors' performances are indicated by the average recall-precision areas. Standard deviations are shown in the brackets.)

3.2.2 Normalized vs. non-normalized tensor descriptors

We experiment the first order (LTD1) and the second order (LTD2) polynomial tensor descriptors and their non-normalized forms (LTD1-N and LTD2-N) on the images under transformations T1 and T2 to show the effectiveness of the matrix normalization proposed in section 2.3. Figure 4(a) shows the results. It is evident that the matrix normalization improve the performance of tensor descriptors. We can get the same conclusion from table 2. The performance of LTD2 increase more greatly than LTD1 after normalization.

3.2.3 Polynomial tensor descriptors with different orders

We experiment the polynomial tensor descriptors from order 1 to order 5 on the images under transformations T1 and T2. These descriptors are denoted as LTD1 to LTD5 respectively. Table 2 shows the results. The third order polynomial tensor descriptor (LTD3) gets the best performance. The addition of the polynomial deformation bases with order greater than 3 decreases the descriptors' performance. But the addition of non-polynomial deformation base $B_1(\mathbf{x}) = (-y\sin(wr), x\sin(wr))^T$ defined in formula (31) improve LTD1's performance greatly (The improved version of LTD1 is denoted as LTD1+ in table 2). So selecting good deformation bases is important for tensor descriptor design.

3.2.4 Comparison with SIFT descriptor

We implement SIFT descriptor by the method proposed by Lowe [8]. Figure 4(b) shows the results of comparing SIFT descriptor with the third order polynomial tensor descriptor (LTD3) and the improved version of LTD1 (denoted as LTD1+). Table 2 shows the performance results of SIFT descriptor and different tensor descriptors. It is clear that LTD2, LTD3 and LTD4 all outperform SIFT descriptor. LTD1+ gets performance comparable to SIFT descriptor.

LTD1+ has only 25 independent dimensions. Our results demonstrate that we can design low-dimensional tensor descriptors with performance comparable to SIFT descriptor, which has 128 dimensions.

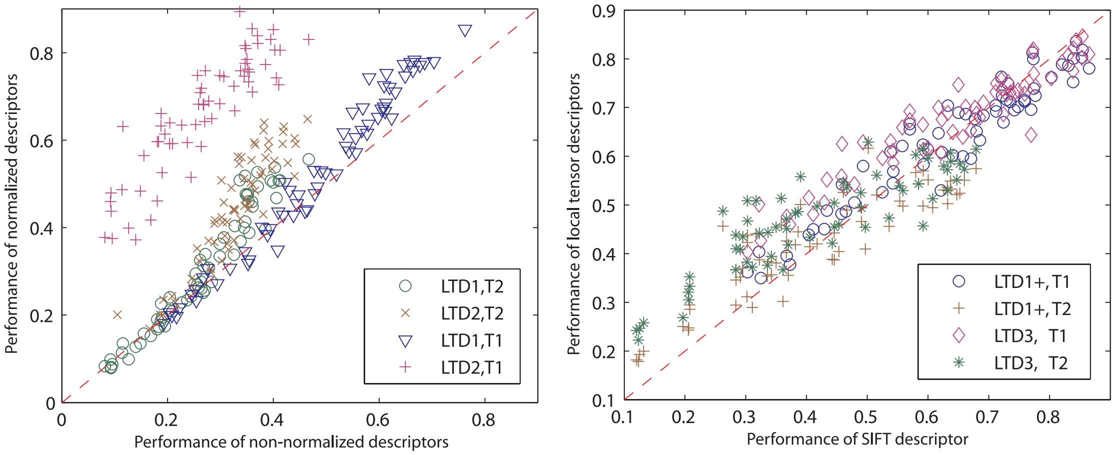
4. Discussion and conclusions

Micro-deformation analysis proposed in this paper is a new method different from traditional methods to analyze and describe local image structures. It is also a general analytic tool can be applied to high-dimensional scalar or vector functions. Instead of analyzing image patches directly, we apply micro-deformations to the input image patch, and monitor the relationship between these deformations and the changes caused by them. Tensor matrixes can be derived from micro-deformation analysis to represent the geometric structure within an image patch.

Our experimental results demonstrate that we can design low-dimensional tensor descriptors with performance comparable to SIFT descriptor, which is the state-of-the-art feature descriptor for object recognition and categorization.

Good deformation bases are important to derive good tensor descriptors. For polynomial deformation bases, the third-order tensor descriptor gets the best performance. Higher order deformation bases do not improve the tensor descriptor's performance. Some non-polynomial deformation bases can improve tensor descriptors' performance greatly. How to select optimal micro-deformation bases to improve tensor descriptors' performance remains to be answered.

In this paper, we use SSD (Sum-of-Squared-Difference) as the function to measure the image changes caused by the micro-deformations. But SSD function can be replaced



(a) (b)

Figure 4. (a) Normalized (LTD1 and LTD2) vs. non-normalized (LTD1-N and LTD2-N) tensor descriptors. (b) SIFT descriptor vs. improved version of LTD1 (LTD1+) and LTD3. (The performances are indicated by recall-precision areas.)

Descriptors (dimensions)	LTD1 (18)	LTD1-N (18)	LTD1+ (25)	LTD2 (63)	LTD2-N (63)	LTD3 (165)	LTD4 (360)	LTD5 (693)	SIFT (128)
T1	0.52(0.19)	0.48(0.15)	0.63(0.13)	0.66(0.14)	0.55(0.11)	**0.67**(0.11)	0.64(0.10)	0.59(0.09)	0.64(0.15)
T2	0.31(0.14)	0.27(0.10)	0.42(0.11)	0.45(0.13)	0.33(0.07)	**0.47**(0.10)	0.43(0.08)	0.39(0.07)	0.41(0.16)

Table 2. Performance results for different descriptors under transformations T1 and T2. The descriptors are polynomial tensor descriptors from order 1 to order 5 (denoted as LTD1 to LTD5), non-normalized version of LTD1 and LTD2 (denoted as LTD1-N and LTD2-N), improved version of LTD1 (denoted as LTD1+), and SIFT descriptor.

by other suitable functions. Investigating good functions to measure the changes caused by the micro-deformations is also an interesting topic for future research.

It is also possible to learn the deformation bases and the change measurement functions from data for special tasks (e.g. face recognition). But how to formulate this learning problem remains to be investigated.

The basis of Harris keypoint detector is the zero-th order polynomial micro-deformation model. This means that micro-deformation analysis can provide a common mechanism for feature detection and feature description.

References

[1] D. Gabor. Theory of communication. *Journal I.E.E.*, 3(93):429–457, 1946.

[2] M. Hannah. *Computer matching of areas in stereo images*. PhD thesis, Stanford University, 1974.

[3] C. Harris and M. Stephens. A combined corner and edge detector. In *Proc. of the 4th Alvey Vision Conference*, pages 147–151, University of Manchester, England, 1988.

[4] Y. Ke and R. Sukthankar. PCA-SIFT: A more distinctive representation for local image descriptors. In *ICCV*, volume 1, pages 511–517, June 2004.

[5] C. Kenney, B. Manjunath, M. Zuliani, G. Hewer, and A. V. Nevel. A condition number for point matching with application to registration and post-registration error estimation. *IEEE PAMI*, 25(11):1437–1454, 2003.

[6] J. Koenderink and A. van Doorn. Representation of local geometry in the visual system. *Biological Cybernetics*, 55:367–375, 1987.

[7] D. Lowe. Object recognition from local scale-invariant features. In *ICCV*, pages 1150–1157, Corfu, Greece, 1999.

[8] D. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004.

[9] K. Mikolajczyk and C. Schmid. Scale & affine invariant interest point detectors. *IJCV*, 60(1):63–86, 2004.

[10] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *IEEE PAMI*, 27(10):1615–1630, 2005.

[11] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. V. Gool. A comparison of affine region detectors. *IJCV*, 65(1/2):43–72, 2005.

[12] H. Moravec. Visual mapping by a robot rover. In *IJCAI*, pages 598–600, 1979.

[13] B. Triggs. Detecting keypoints with stable position, orientation, and scale under illumination changes. In *ECCV*, volume 4, pages 100–113, 2004.

[14] S. A. Winder and M. Brown. Learning local image descriptors. In *CVPR*, pages 1–8, 2007.