Supplementary Material for ICCV 2015 Paper #1007
RIDE: Reversal Invariant Descriptor Enhancement

Lingxi Xie[1]* Jingdong Wang[2] Weiyao Lin[3] Bo Zhang[4] Qi Tian[5]
[1,4]LITS, TNList, Dept. of Comp. Sci. & Tech., Tsinghua University, Beijing, China
[1]Department of Statistics, University of California, Los Angeles, Los Angeles, LA, USA
[2]Microsoft Research, Beijing, China
[3]Department of Electronic Engineering, Shanghai Jiao Tong University, Shanghai, China
[5]Department of Computer Science, University of Texas at San Antonio, San Antonio, TX, USA
[1]198808xc@gmail.com [2]jingdw@microsoft.com
[3]wylin@sjtu.edu.cn [4]dcszb@mail.tsinghua.edu.cn [5]qitian@cs.utsa.edu

Abstract

This document is the supplementary material of the ICCV paper #1007 [5]. We provide the proof of gradient estimation of SIFT, and the generalization of RIDE to deal with more types of reversal and rotation invariance.

1. Orientation Estimation of Dense SIFT

In this section, we aim at proving an approximated estimation of SIFT orientation based on its local gradient values. The approximation is used in Section 3.3 of the main article.

1.1. Implementation of SIFT

The implementation of SIFT is based on the original paper [2]. In the following paragraphs, we briefly review the process of orientation assignment and descriptor representation.

First let us assume that the assignment of descriptor scale is finished, which fits the case of dense sampling [1] where all the descriptors have the same, fixed window size. Denote an image as $\mathbf{I} = [L(x,y)]_{W \times H}$. The gradient magnitude, $m(x,y)$, and orientation, $\theta(x,y)$, is pre-computed for each pixel:

$$m(x,y) = \left[(L(x+1,y)-L(x-1,y))^2 + (L(x,y+1)-L(x,y-1))^2\right]^{1/2} \quad (1)$$

$$\theta(x,y) = \arctan\left[(L(x,y+1)-L(x,y-1)) \,/\, (L(x+1,y)-L(x-1,y))\right] \quad (2)$$

The magnitude and orientation on each pixel are then used to estimate the dominant orientation of that descriptor. An orientation histogram is formed from gradient orientation of the pixels within a region around the keypoint. Each sample added to the histogram is weighted by its gradient magnitude and by a Gaussian-weighted circular window with a σ that is 1.5 times that of the scale of the keypoint. Peaks in the orientation histogram correspond to **dominant** orientations of local gradients. The highest peak in the histogram is detected, and then any other local peak that is within 80% of the highest peak is used to also create a keypoint with that orientation. Therefore, for locations with multiple peaks of similar magnitude, there will be multiple keypoints created at the same location and scale but different orientations.

The above method works well on image matching and retrieval [2], but we do not need to assign multiple orientations for a descriptor in the classification tasks. As an alternation, it is also suggested to estimate a unique **accumulated** orientation

*This work was done when Lingxi Xie was an intern at Microsoft Research.

using the following method. Every gradient magnitude is decomposed along both x and y axes, *i.e.*,

$$\begin{aligned} m_x(x,y) &= m(x,y) \times \cos\theta(x,y) & (3) \\ m_y(x,y) &= m(x,y) \times \sin\theta(x,y) & (4) \end{aligned}$$

and all the decomposed components are accumulated on x and y axes, respectively:

$$\begin{aligned} G_x(x,y) &= \sum_{x,y} m_x(x,y) & (5) \\ G_y(x,y) &= \sum_{x,y} m_y(x,y) & (6) \end{aligned}$$

Finally we get a 2-D vector indicating the orientation of that descriptor.

Of course, we can also follow the orientation assignment of original SIFT implementation [2]. In practise, we have implemented RIDE with both **dominant** and **accumulated** orientations, and found that the latter works slightly better. Another reason why we prefer the **accumulated** orientation is that it is a continuous value in $[0, 2\pi)$, which makes it easier for us to design the RIDE-8 algorithm.

In descriptor representation, we inherit $m(x,y)$ and $\theta(x,y)$ values of each pixel. The implementation of dense SIFT [4] does not rotate the descriptor region. The region of a descriptor is partitioned into 4×4 grids, and an 8-bin orientation histogram is constructed in each grid. The central orientation value of the t-th bin is $\theta_t = t\pi/4$, $t = 0, 1, \ldots, 7$. Then the gradient magnitude of each pixel is then trilinearly quantized onto at most two bins. By trilinear we mean that if the orientation of a pixel, $\theta(x,y)$, is closest to two standard orientation, say, $\theta_a < \theta(x,y) < \theta_b$, then the coefficients assigned to the bins are:

$$m_a = m(x,y) \times \frac{\theta_b - \theta(x,y)}{\theta_b - \theta_a} \tag{7}$$

$$m_b = m(x,y) \times \frac{\theta(x,y) - \theta_a}{\theta_b - \theta_a} \tag{8}$$

An 8-dimensional orientation histogram is thereafter obtained in each of the 4×4 grids. Finally, the descriptor vector is constructed by concatenating the histogram vectors from all 4×4 grids.

1.2. Orientation Estimation

The main goal of this part is to prove the next approximation theorem:

Theorem: Given a densely sampled SIFT descriptor $\mathbf{d} = (d_k, \theta_k)_{k=1,2,\ldots,128}$, where d_k and θ_k are the gradient value and the histogram orientation for the k-th dimension, respectively. Its **accumulated** orientation θ approximately satisfies:

$$\tan\theta = \frac{G_y(x,y)}{G_x(x,y)} = \frac{\sum_{x,y} m_y(x,y)}{\sum_{x,y} m_x(x,y)} \approx \frac{\sum_k d_k \sin\theta_k}{\sum_k d_k \cos\theta_k} \tag{9}$$

For this, we only need to prove the following lemma:

Lemma: When a gradient value (m, θ) with an arbitrary orientation is quantized as (m_a, θ_a) and (m_b, θ_b) $(\theta_a < \theta < \theta_b)$ with the trilinear interpolation, *i.e.*, using(7) and (8):

$$m_a = m \times \frac{\theta_b - \theta}{\theta_b - \theta_a} \tag{10}$$

$$m_b = m \times \frac{\theta - \theta_a}{\theta_b - \theta_a} \tag{11}$$

its impacts on SIFT descriptor representation, before and after quantization, are approximately the same, *i.e.*,

$$\begin{aligned} m\cos\theta &\approx m_a \cos\theta_a + m_b \cos\theta_b & (12) \\ m\sin\theta &\approx m_a \sin\theta_a + m_b \sin\theta_b & (13) \end{aligned}$$

Proof: we only prove (12), since the proof of (13) is very similar.

Using (10) and (11) to substitute m_a and m_b in (12) yields:

$$\begin{aligned} & m_a \cos\theta_a + m_b \cos\theta_b \\ = \; & m \times \frac{\theta_b - \theta}{\theta_b - \theta_a} \times \cos\theta_a + m \times \frac{\theta - \theta_a}{\theta_b - \theta_a} \times \cos\theta_b && (14) \\ = \; & m \times \left(\frac{\theta_b - \theta}{\theta_b - \theta_a} \times \cos\theta_a + \frac{\theta - \theta_a}{\theta_b - \theta_a} \times \cos\theta_b \right) && (15) \end{aligned}$$

Now, let us make the approximation that:

$$\begin{aligned} \frac{\theta_b - \theta}{\theta_b - \theta_a} & \approx \frac{\sin(\theta_b - \theta)}{\sin(\theta_b - \theta_a)} && (16) \\ \frac{\theta - \theta_a}{\theta_b - \theta_a} & \approx \frac{\sin(\theta - \theta_a)}{\sin(\theta_b - \theta_a)} && (17) \end{aligned}$$

and (15) becomes:

$$\begin{aligned} & m_a \cos\theta_a + m_b \cos\theta_b \\ = \; & m \times \frac{\theta_b - \theta}{\theta_b - \theta_a} \times \cos\theta_a + m \times \frac{\theta - \theta_a}{\theta_b - \theta_a} \times \cos\theta_b && (18) \\ \approx \; & m \times \left[\frac{\sin(\theta_b - \theta)}{\sin(\theta_b - \theta_a)} \times \cos\theta_a + \frac{\sin(\theta - \theta_a)}{\sin(\theta_b - \theta_a)} \times \cos\theta_b \right] && (19) \\ = \; & \frac{m \times [\sin(\theta_b - \theta) \cos\theta_a + \sin(\theta - \theta_a) \cos\theta_b]}{\sin(\theta_b - \theta_a)} && (20) \\ = \; & \frac{m \times [(\sin\theta_b \cos\theta - \cos\theta_b \sin\theta) \cos\theta_a + (\sin\theta \cos\theta_a - \cos\theta \sin\theta_a) \cos\theta_b]}{\sin(\theta_b - \theta_a)} && (21) \\ = \; & \frac{m \times (\sin\theta_b \cos\theta \cos\theta_a - \cos\theta_b \sin\theta \cos\theta_a + \sin\theta \cos\theta_a \cos\theta_b - \cos\theta \sin\theta_a \cos\theta_b)}{\sin(\theta_b - \theta_a)} && (22) \\ = \; & \frac{m \times (\sin\theta_b \cos\theta \cos\theta_a - \cos\theta \sin\theta_a \cos\theta_b)}{\sin(\theta_b - \theta_a)} && (23) \\ = \; & \frac{m \times \cos\theta \times (\sin\theta_b \cos\theta_a - \cos\theta_b \sin\theta_a)}{\sin(\theta_b - \theta_a)} && (24) \\ = \; & m \cos\theta && (25) \end{aligned}$$

We provide a discussion on the approximation (16) and (17). Given that $\theta_b - \theta_a = \pi/4$, the maximum relative error of the approximation is less than 11%. Let us define $f(x) = \frac{\sin x}{x}$. Since $\lim_{x \to 0} f(x) = 1$ and $f(x)$ is a monotonically increasing function, large errors of (16) and (17) appear when $\theta_b - \theta$ or $\theta - \theta_a$ is quite small, in which case the m_a or m_b is also quite small thus the absolute estimation error is ignorable. Therefore, we can conclude that (16) and (17) are reasonable.

2. Generalized RIDE: RIDE-4 and RIDE-8

In this section, we provide a detailed discussion of generalizing RIDE to deal with more types of reversal and rotation invariance. It is an extension to Section 3.5 of the main article.

2.1. RIDE-2, RIDE-4 and RIDE-8

Recall that we have computed a 2-D global gradient vector $\mathbf{G} = (G_x, G_y)^\top$, in which G_x and G_y estimate the horizontal and vertical orientation of a descriptor, respectively. If it is constrained that $G_x \geqslant 0$ holds for a descriptor $\mathbf{d}$, we need to generate a left-right reversed version of that descriptor, $\mathbf{d}^{\mathrm{R}}$, and select the one in $\mathbf{d}$ and $\mathbf{d}^{\mathrm{R}}$ that satisfies $G_x \geqslant 0$. Such a descriptor, denoted as $r_2(\mathbf{d})$, is left-right reversal invariant. If $G_x = 0$ for $\mathbf{d}$, then both versions satisfy the condition. In such cases, we choose the one with the larger alphabetical order.

If we need to achieve upside-down reversal invariance, the value G_y should also be constrained, *i.e.*, $G_y \geqslant 0$. We then generate 3 other versions of a descriptor $\mathbf{d}$, namely $\mathbf{d}_0$, $\mathbf{d}_1$, $\mathbf{d}_2$ and $\mathbf{d}_3$, in which $\mathbf{d}_0$ is just $\mathbf{d}$, $\mathbf{d}_1$ is the left-right reversed version of $\mathbf{d}$, $\mathbf{d}_2$ is the upside-down reversed version of $\mathbf{d}$, and $\mathbf{d}_3$ is the left-right and upside-down reversed version of $\mathbf{d}$.

Algorithm	**Aircraft-100**-1	**Aircraft-100**-2	**Aircraft-100**-4	**Aircraft-100**-8
ORIG	$\mathbf{58.75}$	48.52	39.33	25.11
RIDE-2	55.22	$\mathbf{55.22}$	43.20	29.71
RIDE-4	47.44	47.44	$\mathbf{47.44}$	35.41
RIDE-8	43.47	43.47	43.47	$\mathbf{43.47}$

Table 1. Classification accuracy ($\%$) of different versions of RIDE on different versions of the **Aircraft-100** dataset.

Obviously, there exists at least one of them that satisfies both $G_x \geqslant 0$ and $G_y \geqslant 0$. If more than one candidates satisfy the conditions, we choose the one with the largest sequential lexicographic order. Such a descriptor, denoted as $r_4(\mathbf{d})$, is both left-right and upside-down invariant.

A final type of variant comes from rotating the descriptor by 90°. Adding a 90°-rotation option into left-right and upside-down reversals obtains up to 8 descriptor versions. We generate all these variants and select one from them by constraining $G_x \geqslant G_y \geqslant 0$, *i.e.*, $G_x \geqslant 0$, $G_y \geqslant 0$ and $G_x \geqslant G_y$. If more than one candidates satisfy the conditions, we choose the one with the largest sequential lexicographic order. Such a descriptor, denoted as $r_8(\mathbf{d})$, is invariant through all the reversal and rotation operations.

We provide an intuitive explanation of RIDE-2, RIDE-4 and RIDE-8 algorithms. All the reversal and rotation operations change the orientation of a descriptor correspondingly. RIDE-2, in which $G_x \geqslant 0$, limits the orientation to falling into an interval of a 180° range. This range is further shrunk into 90° in RIDE-4, and 45° in RIDE-8. A descriptor with **any** orientation could be aligned into the range with one or a few reversal or rotation variations, and in this way we cancel out the reversal and rotation operations and achieve the desired invariance.

2.2. Experiments

We evaluate the original descriptors with RIDE-2, RIDE-4 and RIDE-8 on the **Aircraft-100** dataset [3]. We use four different versions of the dataset. The **aligned** version, denoted as **Aircraft-100**-1, is the one in which all the objects are manually aligned to the right. Other three versions, denoted as **Aircraft-100**-2, **Aircraft-100**-4 and **Aircraft-100**-8, are generated by randomly assigning one of 2, 4 and 8 image transformations to each image in the aligned dataset. Here, 2 transformations include unchanged and the left-right reversal, 4 transformations are constructed by adding the option of upside-down reversal to 2 transformations, and 8 transformations are constructed by adding the option of 90° rotation to 4 transformations. The property of **Aircraft-100**-2 is very similar to the original (unaligned) version of the **Aircraft-100** dataset.

The basic setting follows that in Section 4.1 of the main article. We only use SIFT descriptor, and do not use spatial pyramids in the following experiments. The classification results are summarized in Table 1. One can observe that on the **Aircraft-100**-1 dataset, the system with original descriptors (**ORIG**) works best. After original descriptors are processed by RIDE, classification accuracy drops dramatically. The underlying reason is that RIDE harms the descriptive power of original descriptors by performing a one-of-many selection. The more candidates generated for selection, the heavier accuracy drop is observed.

However, in the case of **Aircraft-100**-2, **RIDE-2** works better than **ORIG**. This implies that **RIDE-2** captures the left-right reversal invariance. Although the descriptive power of SIFT is reduced, the benefit of reversal invariance is larger than the loss in descriptive power. However, when we use **RIDE-4** and **RIDE-8**, the descriptive power continues to drop but we do not obtain any new invariance, resulting in an accuracy drop from **RIDE-2** to both **RIDE-4** and **RIDE-8**. Similar results are also observed in the **Aircraft-100**-4 dataset, *i.e.*, **RIDE-4** is just enough to capture left-right and upside-down reversal. In **Aircraft-100**-8 dataset, all the reversal and rotation variance might be encountered, therefore **RIDE-8** produces the highest accuracy.

The above experiments verify that RIDE increases the robustness of descriptors but harms the descriptive power. According to Table 1, one type of reversal/rotation variance, if **not** captured, causes about 10% accuracy drop, meanwhile performing RIDE to capture an unnecessary invariance causes about 5% accuracy drop. Therefore it is not good to cover all types of invariance: **the best strategy is to take what we need.**

Consequently, we do not use **RIDE-4** and **RIDE-8** in all the experiments presented in the main article, since all the evaluated datasets, either on fine-grained object recognition or scene understanding, often do not contain upside-down reversed or 90°-rotated objects. **RIDE-2** works best in such cases.

References

[1] A. Bosch, A. Zisserman, and X. Munoz. Scene Classification via pLSA. *International Conference on Computer Vision*, 2006.

[2] D. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal on Computer Vision*, 60(2):91–110, 2004.

[3] S. Maji, E. Rahtu, J. Kannala, M. Blaschko, and A. Vedaldi. Fine-Grained Visual Classification of Aircraft. *arXiv preprint, arXiv: 1306.5151*, 2013.

[4] A. Vedaldi and B. Fulkerson. VLFeat: An Open and Portable Library of Computer Vision Algorithms. *ACM Multimedia*, 2010.

[5] L. Xie, J. Wang, W. Lin, B. Zhang, and Q. Tian. RIDE: Reversal Invariant Descriptor Enhancement. *International Conference on Computer Vision*, 2015.