Hindawi
Journal of Sensors
Volume 2017, Article ID 8513949, 25 pages
https://doi.org/10.1155/2017/8513949

Research Article

Optimum Pipeline for Visual Terrain Classification Using Improved Bag of Visual Words and Fusion Methods

Hang Wu,[1] Baozhen Liu,[1] Weihua Su,[1] Zihao Chen,[1] Wenchang Zhang,[2] Xudong Ren,[1] and Jinggong Sun[1]

[1]*Institute of Medical Equipment, Academy of Military Medical Science, Tianjin 300161, China*
[2]*The State Key Laboratory of Intelligent Technology and System, Computer Science and Technology School, Tsinghua University, Beijing 100084, China*

Correspondence should be addressed to Jinggong Sun; sunjg@vip.sina.com

Received 5 June 2016; Revised 17 October 2016; Accepted 19 January 2017; Published 29 March 2017

Academic Editor: Raymond Swartz

We propose an optimum pipeline and develop the hybrid representation to produce an effective and efficient visual terrain classification system. The bag of visual words (BOVW) framework has emerged as a promising approach and effective paradigm for visual terrain classification. The method includes four main steps: (1) feature extraction, (2) codebook generation, (3) feature coding, and (4) pooling and normalization. Recent researches have primarily focused on feature extraction in the development of new handcrafted descriptors that are specific to the visual terrain. However, the effects of other steps on visual terrain classification are still unknown. At the same time, fusion methods are often used to boost classification performance by exploring the complementarity of diverse features. We provide a comprehensive study of all steps in the BOVW framework and different fusion methods for visual terrain classification. Then, multiple approaches in each step and their effects are explored on the visual terrain dataset. Finally, the feature preprocessing technique, improved BOVW framework, and fusion method are used to construct an optimum pipeline for visual terrain classification. The hybrid representation developed by the optimum pipeline performs effectively and rapidly for visual terrain classification in the terrain dataset, outperforming those current methods. Furthermore, it is robust to diverse noises and illumination alterations.

1. Introduction

Technological advances allow more robots to be deployed in outdoor, off-road, and natural as well as unnatural environments [1]. Unlike the indoor-structured environment, there are a variety of terrain types. Certain flat and nonslippery terrain types allow the robot to traverse them at relatively high speed, but other terrain surfaces are loose, bumpy, or muddy, and the robot must traverse them slowly and carefully. The terrain surface itself could be a possible hazard to the outdoor mobile robot and is referred to as a nongeometric hazard [2]. The robot should be able to rapidly determine the nongeometric terrain type to avoid inappropriate motion strategies.

Two main approaches are used to recognize nongeometric terrain characteristics, that is, proprioceptive-based methods and appearance-based methods. Proprioceptive-based methods [1, 3, 4] learn the difficulty of traversing different terrain types by analyzing such inputs like vibrations, slips, and sinks, and so forth. The main drawback of these methods is the inability to classify the terrain type before the robot traverses it. Appearance-based methods [1, 5–18] project the problem into image-processing and classification realm. Visual sensors are widely used and cost effective to provide rich terrain information. The appearance-based methods have attracted the attention of many researchers.

However, obvious interclass similarity and significant intraclass variability make the problem more challenging. At the same time, visual terrain classification affects the movement strategies of the robot and requires high real-time performance if the robot moves rapidly. The question of how to effectively and efficiently complete the terrain classification task has become a hot topic.

To address this problem, many handcrafted descriptors have been used, such as color histograms [5, 13], Local Binary Pattern (LBP) [6, 14], GIST [15], scale-invariant feature transform (SIFT) [16, 17, 19], compact composite descriptors (CCDs) [18], fuzzy color and texture histograms (FCTH) [18], and joint composite descriptors (JCD) [18]. LBP is a very simple yet powerful texture descriptor. Color feature is an important attribute for various terrains and a direct choice for many people to deal with the terrain classification problem. GIST is a typical global feature based on Gabor filters. SIFT is a scale- and rotation-invariant detector and descriptor. CCDs, FCTH, and JCD composite the texture and color information of the image. In conclusion, those handcrafted descriptors use low-level color and texture information to develop global representations. However, their accuracies and robustness still cannot meet the increasing requirements for the visual terrain classification due to the semantic gap. Deep neural networks (DNNs) [20–23] can build and train deep architectures to capture graphical semantic information, achieving a large performance boost in many computer vision applications. However, it is computationally expensive to directly train effective DNNs for visual terrain classification. For a good trade-off between effectiveness and efficiency, the BOVW framework [17, 24, 25] is used to generate a compact semantic representation with low-level descriptors for visual terrain classification, obtaining good accuracy and nice robustness. This visual terrain classification algorithm has been successfully applied in the small quadruped robot *Littledog* as a necessary function module [24].

The pipeline of the BOVW in visual terrain classification consists of four main steps: (1) feature extraction, (2) codebook generation, (3) feature coding, and (4) pooling and normalization [26]. Recent researches have specifically focused on developing new handcrafted descriptors specific to the visual terrain. To the best of our knowledge, research on methods of other steps for visual terrain classification has not been reported. Those methods are also critical issues for the effectiveness and efficiency of visual terrain classification algorithms. *How to make decision in each step to construct an optimum pipeline for visual terrain classification* still remains unknown and needs to be extensively explored. In addition, none of the descriptors will exhibit the same discriminatory power for all terrain classes. Therefore, it is a natural choice to combine a set of diverse and complementary features for better classification performance. Many researches [27, 28] have been committed to fusing multiple descriptors to improve performance. Typical fusion methods include early fusion and late fusion. Early fusion is performed in low-level feature space, that is, descriptor space, where multiple descriptors would be concatenated into a single one. While, the late fusion works in midlevel future space using the kernel fusion methods. For visual terrain classification, the question of *how to use fusion methods to develop a hybrid representation to realize effective and efficient visual terrain classification* is well worthy of detailed investigation.

Unlike previous methods, our study uses *off-the-shelf* descriptors and focuses on designing an optimum pipeline for visual terrain classification. We present a review of existing methods from a new perspective (Sections 2 and 3) and then evaluate various methods to design an optimum pipeline for visual terrain classification (Sections 4 and 5). This contribution can be described by the following three objectives:

(i) Providing a comprehensive study of each step in BOVW framework and different fusion methods. We summarized the various methods of each step from a new perspective and analyzed their roles for visual terrain classification.

(ii) Presenting a comparison of different BOVW frameworks and fusion methods for visual terrain classification on the terrain dataset. Specifically, we explore two types of local descriptors, eight types of coding methods, six types of pooling methods, eight types of normalization methods, and two types of fusion methods.

(iii) Designing an optimum pipeline and developing the hybrid representation to produce an effective and efficient visual terrain classification system. This proposed hybrid representation improves the performance significantly with a certain margin compared to current methods. Furthermore, it is robust to diverse noises and illumination alterations.

The remainder of this paper is organized as follows. In Section 2, we provide a comprehensive study of each step in BOVW framework. Section 3 introduces selected fusion methods in detail. In Section 4, an empirical study of the optimum pipeline and the proposed hybrid representation for visual terrain classification are performed on a challenging dataset. Finally, Section 5 provides further analysis on several important attributes of the proposed methods. Section 6 is the conclusion.

2. Framework of BOVW for Visual Terrain Classification

In this section, we provide a comprehensive study of each step in the BOVW framework, which transforms the low-level features into midlevel features with stronger discriminability. For now, the framework of BOVW has emerged as a promising approach and the effective paradigm for visual terrain classification [6, 11, 16, 17, 24]. As shown in Figure 1, the robot obtains the terrain visual images through a camera. Then, the low-level local features are extracted directly from the images to describe the texture and color characteristics of various terrain. Coding methods reexpress these low-level local features using the pretraining or online-training codebook. Pooling and normalization methods aggregate the local features into a global representation. The choices of methods in each step are critical to the discriminability of final midlevel feature. Those methods in BOVW framework are worthy of careful study and evaluation. The version of our preliminary work has appeared in [29].

This BOVW framework consists of four steps: (1) feature extraction, (2) codebook generation, (3) feature coding, and (4) pooling and normalization. Let $\mathbf{X}$ be a set of D-dimensional local descriptors extracted from a terrain image

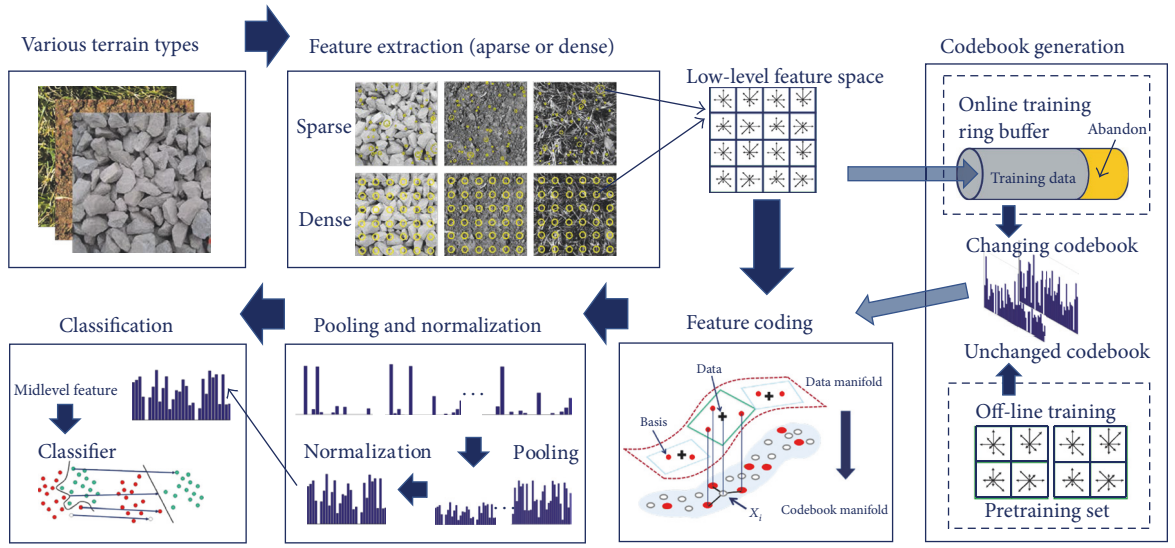

FIGURE 1: Pipeline for obtaining midlevel features for terrain classification composed of four main steps: (1) feature extraction, (2) codebook generation, (3) feature coding, and (4) pooling and normalization.

$\mathbf{X} = [x_1, x_2, \ldots, x_k] \in \mathbb{R}^{D\times K}$. K is the number of the local descriptors. Through clustering, a codebook is formed with M entries $\mathbf{B} = [b_1, b_2, \ldots, b_M] \in \mathbb{R}^{D\times M}$, where b_i ($b_i \in \mathbb{R}^D$) denotes a codeword. The codebook is used to express each descriptor and develop the feature coding result $\mathbf{D}$, and pooling and normalization methods are subsequently used to produce the image-level representation, that is, a midlevel feature $\mathbf{F}$. In the end, the midlevel feature $\mathbf{F}$ is fed into a linear or nonlinear classifier such as Support Vector Machine (SVM) for terrain classification.

The current visual terrain classification methods [5, 6, 11–17] use the primitive BOVW framework and focused primarily on developing new handcrafted low-level feature or applying sophisticated classifier. By contrast, our study concentrated on improving BOVW framework to design an optimum pipeline. Considering the focus of our study and the efficiency of evaluations, this paper only used the common low-level feature and classifier, that is, SIFT and SVM.

2.1. Descriptors and Quantization

2.1.1. Feature Extraction. Feature extraction acquires low-level feature information from the terrain images, and this process consists of two steps: extracting patch (detector) and representing patches (descriptor) [30]. The detector of the global descriptor is the entire image. According to different detection methods, local descriptors can be divided into sparse descriptors and dense descriptors. Sparse descriptors typically select scale-extreme points as detectors in Difference of Gaussian (DOG) filtered images, whereas dense descriptors are much simpler and apply a dense grid sample on one or several scales.

Many handcrafted descriptors are used to solve the visual terrain classification issue (e.g., ColorHist [5, 13], LBP [6, 14], GIST [15], SIFT [16, 17, 19], CCDs [18], FCTH [18], and JCD [18]). Among those descriptors, the scale-invariant feature transform (SIFT) and its variants are widely used due to their ease of use and good performance. SIFT is a milestone low-level feature proposed by Lowe in 1991 and perfected in 2004 and is a scale- and rotation-invariant detector and descriptor [31]. This method is of many variants such as SURF [32] and RootSIFT [33]. SURF can be categorized as an accelerated version of SIFT, and RootSIFT is equivalent to SIFT vectors with a Hellinger kernel and is implemented with simple algebraic manipulation based on SIFT without the need for additional storage space. These features can be unified and referred to as SIFT-various descriptors.

Feature extraction is not the focus of our study, and, thus, we use the off-the-shelf descriptors SIFT as the low-level descriptors representation $\mathbf{X} = [x_1, x_2, \ldots, x_K] \in \mathbb{R}^{D\times K}$ to produce midlevel features. Both sparse descriptors and dense descriptors, that is, SIFT and DSIFT, are studied in our work, which represent two different types of low-level descriptions that might exhibit different properties with respect to variations of the pipeline.

2.1.2. Feature Preprocessing. The raw input local descriptors $X_i \in \mathbb{R}^D$ are usually high dimensional and strongly correlated, which create great challenges in the subsequent codebook generation [34]. Principal component analysis (PCA) [35] is a statistical procedure that uses orthogonal transforms to map a raw input descriptor into a much lower dimensional descriptor while incurring notably little error, resulting in dimensional reduction. PCA is commonly used in conjunction with a whitening technique, and the goal of whitening is to render the input less redundant, that is, less correlated and having the same variance. The transform formula for PCA-whitening is

$$X_i' = \Lambda U^T X_i, \tag{1}$$

where $X_i \in \mathbb{R}^D$ is the raw input, $X_i' \in \mathbb{R}^N$ is the PCA-whitening result, $U \in \mathbb{R}^{D\times N}$ is the dimension reduction matrix from PCA, Λ is the diagonal whitening matrix $\text{diag}(\Lambda) = [1/\sqrt{\lambda_1}, 1/\sqrt{\lambda_2}, \ldots, 1/\sqrt{\lambda_N}]$, and λ_i is the ith largest eigenvalue of the covariance matrix.

In our evaluation, we found that this step is of considerable necessity for improving visual terrain classification performance; however, many previous terrain classification approaches have ignored this step. We add this step into the pipeline to decorrelate the descriptor, reduce the dimension, and normalize the variance. Moreover, this approach should be more necessary for dense features because adjacent pixel values are more highly correlated.

2.1.3. Codebook Generation. The BOVW framework is based on the idea of using overcomplete basis vectors to encode the local descriptors. These basis vectors are also known as *codewords*, and a collection of those codewords is referred to as a *codebook*. The codebook is computed on the training set and used for the descriptors of all images. The codewords are considered to be characteristically representative of the image descriptors [36]. Typically, two types of approaches are often used for codebook generation:

(i) *K-means clustering*: partitioning the local descriptor space into informative regions (codewords), each of which is represented by its center.

(ii) *Gaussian Mixture Model clustering*: using the generative models to capture the probability distribution of the local descriptors.

K-Means Clustering. K-means clustering is probably the most common way of constructing a codebook. Given a set of T training descriptors $X_{\text{train}} = [x_1, x_2, \ldots, x_T] \in \mathbb{R}^{D\times T}$, K-means seeks M basis vectors $[b_1, b_2, \ldots, b_M] \in \mathbb{R}^{D\times M}$ and data-to-means assignments $[q_1, q_2, \ldots, q_T] \in \{1, 2, \ldots, M\}$ to minimize the cumulative approximation error $\sum_{i=1}^{T} \|x_i - b_{q_i}\|^2$. Usually, we perform optimization using an iterative procedure. The details of this algorithm can be found in [37].

GMM Clustering. A Gaussian Mixture Model (GMM) $p(x \mid \theta)$ represents the probability density on training descriptors $X_{\text{train}} = [x_1, x_2, \ldots, x_T] \in \mathbb{R}^{D\times T}$.

$$p(x \mid \theta) = \sum_{m=1}^{M} P(x \mid u_m, \Sigma_m)\, \pi_m, \tag{2}$$

where M is the mixture number (codebook size) and $\theta = (\pi_1, \mu_1, \Sigma_1, \ldots, \pi_M, \mu_M, \Sigma_M)$ are the model parameters. The parameters include the prior probability value $\pi_m \in \mathrm{R}_+$, the mean $\mu_m \in \mathrm{R}^D$, and the diagonal covariance matrix $\Sigma_m \in \mathrm{R}^{D\times D}$ of each Gaussian component. The expectation maximization (EM, [38]) is used to learn those parameters from the training descriptors X_{train}.

The EM algorithm is sensitive to initial values. We use K-means to determine the initial values, thus improving the performance of GMM. Compared with the K-means algorithm, which collects only descriptor ascription information, GMM provides a more comprehensive description of the characteristics of the descriptor space, which contains not only the means information but also the shape of their distribution. Moreover, the GMM clustering algorithm defines the soft descriptors-to-codewords assignment compared with the Hard Assignment in the K-means algorithm. It should be noted that some potential alternative clustering algorithms, for example, HCS [39], DBSCAN [40], or UPGMA [41], can also be used to generate codebook.

2.2. Feature Coding. In this section, different types of coding methods are detailed, discussed, and studied. Coding is a core step in the BOVW framework for visual terrain classification. The coding step uses the codebook $\mathbf{B} = [b_1, b_2, \ldots, b_M] \in \mathbb{R}^{D\times M}$ to map the descriptor space $\mathbf{X} = [x_1, x_2, \ldots, x_k] \in \mathbb{R}^{D\times K}$ to the coding space $\mathbf{D}$. Unlike traditional views [30], in our study, we find that the essential difference in different coding methods is the way in which information is obtained from the descriptor space. Different methods of obtaining information construct different coding spaces and produce different discrimination representations. As shown in Figure 2, we divide the coding methods into two different types: *activation-based encoding methods* and *difference-based encoding methods*.

Activation-Based Encoding Methods. These use the activation concept to obtain information from the descriptor space.

(1) The code space is composed of diverse codewords. The concerns in these methods are which codewords will be activated and to what extent they will be activated. Different coding methods develop different activation rules.

(2) The information used by activation-based encoding methods is the 0-order statistics of the distribution of descriptors. The coding result reflects the information on affiliation of the local descriptors to the codewords.

(3) Each descriptor is encoded independently in the input image, and each has a respective coding result. The follow-up pooling step is required to obtain the image-level representation. The length of the image-level representation is the size of codebook.

Typical encoding methods in this category include Hard Assignment (HA) [24], Soft Assignment (SA) [36], Local Soft Assignment (LSA) [42], Sparse Coding (SC) [43], Local Coordinate Coding (LCC) [44], and Locality-constrained Linear Coding (LLC) [45].

Difference-Based Encoding Methods. These use the difference concept to obtain information on the descriptor space.

(1) The code space is built using the differences between descriptors and codebook. Different coding methods record various types of differences, and the core of those coding methods is to establish rules of representing the difference.

(2) The difference-based encoding methods use multi-dimensional information (0th, 1st, and 2nd) from

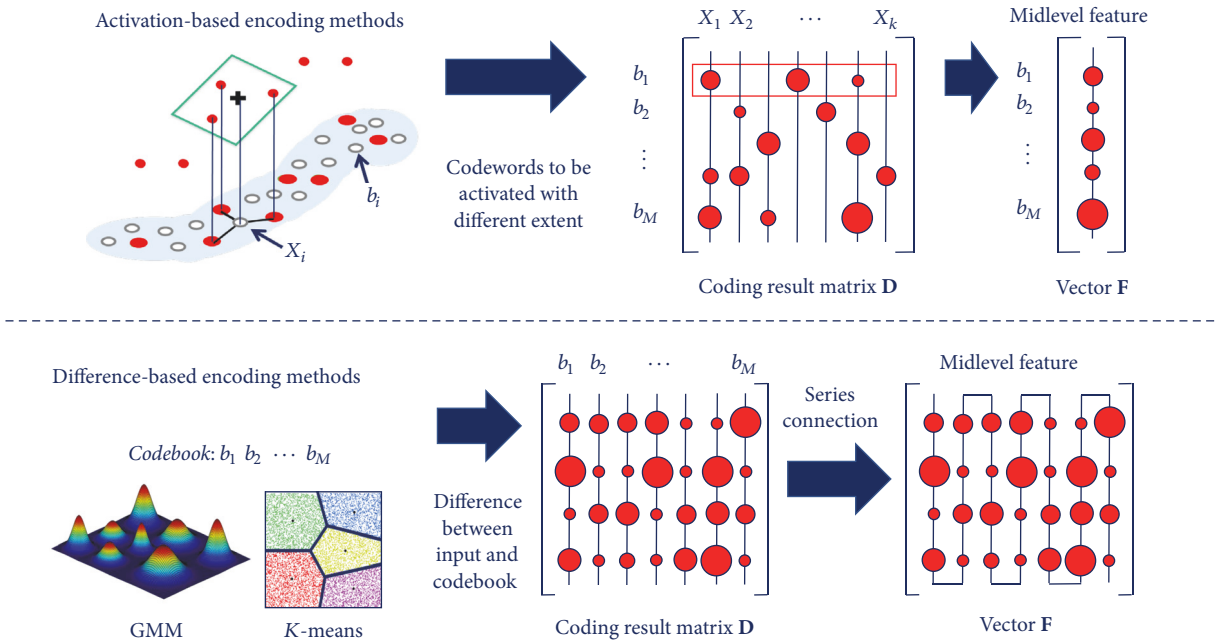

FIGURE 2: Two different types of coding methods: activation-based encoding methods and difference-based encoding methods. The essential difference in different coding methods is the way in which information is obtained from the descriptor space.

the descriptor space. Because these coding methods retain much richer information on the descriptor space, significantly fewer codewords are required compared with the activation-based encoding methods.

(3) All of the descriptors in the input image are encoded as a whole. The coding methods record the differences between the descriptors space and codebook, and the midlevel features are developed by connecting those differences together in series. The pooling step is simply the series connection.

Difference-based encoding methods typically include Fisher Vector (FV) [25, 46], Vector of Locally Aggregated Descriptors (VLAD) [47], Local Tangent-based Coding (LTC) [48], and Super Vector Coding (SVC) [49].

2.2.1. Activation-Based Encoding Methods. Activation-based encoding methods use the activation concept to obtain information on the descriptor space, and the core issue is to decide which codewords will be activated and to what extent they will be activated. The coding result is $D \in \mathbb{R}^{D\times M}$. Depending on different activation strategies, the methods can be subdivided into voting-based encoding methods and Sparse Coding methods.

(1) Voting-Based Encoding Methods. Voting-based encoding methods are designed from the perspective of activation based on similarity. The codewords similar to the coding descriptor are considered "close." Methods activate closer codewords with stronger responses. Typical methods include Hard Assignment (HA) [24], Soft Assignment (SA) [36], and Local Soft Assignment (LSA) [42].

For Hard Assignment (HA), the descriptor only activates the nearest codeword. The coding representation of descriptor $x_i \in \mathbb{R}^D$ is

$$\text{HA: } d_j = \begin{cases} 1, & \text{if } j = \underset{j}{\operatorname{argmin}} \left\| x_i - b_j \right\|_2, \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

Certain descriptors might have zero, one, or multiple candidate codewords in the codebook, and hard quantization will cause information loss. To solve this problem, Soft Assignment (SA) [42] chooses to activate all codewords and uses the kernel function of distance as the coding representation instead of simple 1 or 0 responses:

$$d_j = \frac{\exp\left(-\beta \widehat{e}\left(x_i, b_j\right)\right)}{\sum_{l=1}^{M} \exp\left(-\beta \widehat{e}\left(x_i, b_l\right)\right)}, \tag{4}$$
$$\text{SA: } \widehat{e}\left(x_i, b_j\right) = \left\| x_i - b_j \right\|^2,$$

where β is the smoothing factor that controls the softness of assignment and the Euclidean distance $\widehat{e}$ is used. Considering the manifold structure in data, Local Soft Assignment (LSA) activates its k-nearest codewords and suppresses the remaining codewords.

$$\text{LSA: } \widehat{e}\left(x_i, b_j\right) = \begin{cases} \left\| x_i - b_j \right\|^2, & \text{if } b_j \in N_k\left(x_i\right), \\ \infty, & \text{otherwise,} \end{cases} \tag{5}$$

where $N_k(x_i)$ denotes the k-nearest neighbors of x_i defined by the Euclidean distance $\|x_i - b_j\|^2$.

(2) Sparse Coding Methods. Research on image statistics clearly reveals that image patches are sparse signals [43],

and sparse code can capture more salient properties of the images. Sparse Coding methods activate a small number of codewords and seek a linear combination of those codewords in codebook $\mathbf{B} = [b_1, b_2, \ldots, b_M] \in \mathbb{R}^{D\times M}$ to reconstruct local descriptors $\mathbf{X} = [x_1, x_2, \ldots, x_k] \in \mathbb{R}^{D\times K}$ [30]. The coefficients $\mathbf{D} = [d_1, d_2, \ldots, d_K] \in \mathbb{R}^{M\times K}$ are used as the coding result. The typical Sparse Coding methods include Sparse Coding (SC) [43], Local Coordinate Coding (LCC) [44], and Locality-constrained Linear Coding (LLC) [45]. The unified representation of Sparse Coding methods is formulated in a least-square framework with a regularization term:

$$\underset{D}{\operatorname{argmin}} \sum_{i=1}^{K} \left(\left\| x_i - Bd_i \right\|^2 + \lambda \psi \left(d_i \right) \right), \tag{6}$$

where the least-square term $\|x_i - Bd_i\|^2$ pursues accurate reconstruction (i.e., the descriptors can be described by a small reconstruction error), the regularization term $\psi(d_i)$ limits the solution space to ensure that the activated codewords are representative and discriminating, and λ is a weight factor used to balance those two terms. The regularization term can produce a resulting representation with high intraclass and low interclass similarity [50]. Different Sparse Coding methods have different regularization terms, which can be considered as different rules used to define which type of codewords to be discriminating.

For SC, the regularization term is conducted using the L_1-norm:

$$\text{SC: } \psi \left(d_i \right) = \left\| d_i \right\|_1, \tag{7}$$

where the L_1-norm can limit the number of codewords activated per descriptor (referred to as sparsity), which can be adjusted using λ. Further studies found that the locality constraint plays a more important role [44]. To model this constraint, LCC defines a new regularization term:

$$\begin{aligned} &\text{LCC: } \psi \left(d_i \right) = \left\| \widehat{e_i} \odot \left| d_i \right| \right\|_1, \\ &\text{s.t.} \quad 1^T d_i = 1, \quad \forall i, \end{aligned} \tag{8}$$

where $\odot$ denotes the element-wise multiplication, $\widehat{e_i}$ is the locality adaptor that gives the weights for each codeword, and the weight is proportional to its similarity to the input descriptor x_i:

$$\widehat{e_i} = \left[\widehat{e} \left(x_i, b_1 \right), \ldots, \widehat{e} \left(x_i, b_m \right), \ldots, \widehat{e} \left(x_i, b_M \right) \right]^T, \tag{9}$$

where $\widehat{e}(x_i, b_m)$ is the Euclidean distance between x_i and b_m. The computational cost of LCC is high because its solution relies on iterative optimization. To address this problem, a practical coding scheme known as Locality-constrained Linear Coding (LLC) is designed, which adopts a new regularization term:

$$\begin{aligned} &\text{LLC: } \psi \left(d_i \right) = \left\| E_i \odot d_i \right\|^2, \\ &\text{s.t.} \quad 1^T d_i = 1, \quad \forall i, \end{aligned} \tag{10}$$

where E_i is the exponentiation of $\widehat{e_i}$.

$$E_i = \exp \left(\frac{\widehat{e_i}}{\sigma} \right), \tag{11}$$

where σ is used for adjusting the weight decay speed for the locality adaptor [45]. With the new regularization term designed as shown in (12), the solution of LLC can be derived analytically by

$$\widetilde{d_i} = \left(C_i + \lambda \operatorname{diag} \left(E \right) \right) \setminus 1, \tag{12}$$

$$d_i = \frac{\widetilde{d_i}}{1^T \widetilde{d_i}}, \tag{13}$$

where $C_i = (B^T - 1x_i^T)(B^T - 1x_i^T)^T$ denotes the data covariance matrix. The analytical solution greatly reduces the computational cost.

In practice, even faster approximation of LLC can be used to further speed the encoding process. This approach directly performs a K-nearest-neighbor search for each descriptor x_i to form a local coordinate system and only minimizes the least-square term within this much simpler linear system, and the codewords beyond the system are suppressed. This strategy further reduces the computation complexity.

2.2.2. Difference-Based Encoding Methods. Difference-based coding methods use the difference concept to obtain information on the descriptors space. This type of encoding method describes the difference between the distribution of descriptors in an input image and that fitted to the descriptors of all training images [51]. Different coding methods in this work use different types of difference. Typical methods include Fisher Vector (FV) [25, 46], Vector of Locally Aggregated Descriptors (VLAD) [47], Local Tangent-based Coding (LTC) [48], and Super Vector Coding (SVC) [49].

The Fisher Vector (FV) approach is based on the Fisher kernel, which combines the benefits of generative and discriminative approaches. The FV coding method uses the GMM codebook and fits the GMM to the descriptors in the input image, leading to a representation that captures the Gaussian mean (1st) and variance (2nd) differences between the descriptors and each codeword:

$$\begin{aligned} d_m^{(1)} &= \frac{1}{K\sqrt{w_m}} \sum_{p=1}^{K} \alpha_p \left(m \right) \left(\frac{x_p - \mu_m}{\sigma_m} \right), \\ d_m^{(2)} &= \frac{1}{K\sqrt{2w_m}} \sum_{p=1}^{K} \alpha_p \left(m \right) \left(\frac{\left(x_p - \mu_m \right)^2}{\sigma_m^2} - 1 \right). \end{aligned} \tag{14}$$

In this work, $\{w_m, \mu_m, \sigma_m\}_k$ are the respective mixture weights, means, and diagonal covariance of the GMM codebook $\mathbf{B} \in \mathbb{R}^{D\times M}$, $\alpha_p(m)$ is the Soft Assignment weight of the pth descriptor x_p to the mth Gaussian, and $D \in \mathbb{R}^{2D\times M}$ is obtained by stacking the first and second differences:

$$\text{FV: } D = \left[d_1^{(1)}, d_1^{(2)}, d_2^{(1)}, d_2^{(2)}, \ldots, d_M^{(1)}, d_M^{(2)} \right]. \tag{15}$$

The Vector of Locally Aggregated Descriptors (VLAD) can be viewed as a simplified nonprobabilistic version of the FV with only 1st-order statistics [47]. VLAD associates each local descriptor x_t with its nearest visual word $NN(x_t)$ in the K-means codebook. For each codeword b_i, the differences $(x_t - b_i)$ of the descriptors x_t assigned to codeword b_i are accumulated, and the VLAD coding result $D \in \mathbb{R}^{D\times M}$ is obtained:

$$\text{VLAD: } D = [d_1, d_2, \ldots, d_M], \quad d_i = \sum_{x_t : NN(x_t)=i} (x_t - b_i). \tag{16}$$

However, the salience of each codeword will be different for the descriptor space in the input images. Local Tangent-based Coding (LTC) and Super Vector Coding (SVC) modify the differences d_i used in VLAD by a weight factor θ_i. The weight factor θ_i and the amended difference $\theta_i d_i$ are recorded for each codeword b_i. The LTC and SVC coding result $D \in \mathbb{R}^{(D+1)\times M}$ are

$$\text{LTC/SVC: } D = [\alpha\theta_1, \theta_1 d_1, \alpha\theta_2, \theta_2 d_2, \ldots, \alpha\theta_M, \theta_M d_M], \tag{17}$$

where α is a positive scaling factor used to balance the two terms. The difference between LTC and SVC is used in the method to define the weight factors $\theta = [\theta_1, \theta_2, \ldots, \theta_M] \in \mathbb{R}^M$. LTC uses the LCC methods to obtain the weight factors θ, whereas SVC uses the LSA. LTC and SVC capture 0th-order and 1st-order statistics over the descriptor space.

2.3. Pooling and Normalization Methods. Pooling methods aggregate the coding result into a single vector **F** of the fixed length, thus achieving greater invariance to image transformations, more compact representations, and better robustness to noise and clutter [50]. The coding result, that is, matrix **D**, cannot be fed into the classifier to obtain a final classification result without pooling. The pooling method involves series connection for the difference-based encoding methods. For the activation-based encoding methods, typical pooling methods include average (Avg) [52], maximum (Max) [42], L_P-norm (L_P) [53], theoretical expectation of max pooling (MaxExp) [54], at least one codeword b_m present in image (ExaPro) [54], and Approximate Pooling (AxMin) [50]. Those pooling methods can be divided into two types: classical pooling methods and likelihood-based pooling methods.

Classical Pooling Methods. Classical pooling methods, that is, Avg, Max, and L_P pooling methods, calculate a suitable activation description for each codeword. The Avg pooling method is expressed as the average over the responses to visual word b_m.

$$\text{Avg: } F_m = \frac{1}{|K|}\sum_{k=1}^{K} d_{km}, \tag{18}$$

where d_{km} is the response of the kth descriptor x_k to mth codeword b_m and the input image includes N descriptors. Avg has been widely applied due to its direct and simple mathematical operations. The major disadvantage of Avg is that the resulting representation is strongly influenced by frequent yet often uninformative descriptors but only weakly influenced by rare yet potentially highly informative ones [55]. Frequently occurring uninformative descriptors, for example, background, will reduce the discriminatory ability for Avg pooling. It should be noted that some technology, for example, TF-IDF [56], can be used to further fine-tune the midlevel feature F_m and solve this dilemma effectively, while max pooling goes to the other extreme, and only the strongest response is taken into account.

$$\text{Max: } F_m = \max\left(\{d_{km}\}_{k\in K}\right). \tag{19}$$

The L_P pooling proposed in [53] represents a trade-off between average and max pooling. This method uses an L_P-norm with parameter p that varies the solution between Avg and max pooling for $p = 1$ and $p \to \infty$, respectively:

$$L_P\text{: } F_m = \left(\frac{1}{|K|}\sum_{k=1}^{K} |d_{km}|^p\right)^{1/p}. \tag{20}$$

Likelihood-Based Pooling Methods. Likelihood-based pooling methods are designed from the perspective that the activation characterizes the occurrence probability of the codeword on the input image. Methods assume the coding result of different descriptors follow an i.i.d. Bernoulli distribution and calculate diverse probabilities expression as the pooling result. Likelihood-based pooling methods generally include MaxExp, MaxPro, and AxMin. It is worth noting that the @n strategy is usually applied in this type of pooling method to suppress leakage [50]. The @n strategy selects n strongest responses Φ_{mn} per codeword b_m and feeds Φ_{mn} to those pooling methods.

MaxExp and MaxPro calculate the probability of at least one codeword b_m being present in the input image. The two methods have the same goal but use different methods of probabilistic mathematics.

$$\text{MaxExp: } F_m = 1 - \left(1 - \frac{1}{n}\sum_{k=1}^{n}\Phi_{km}\right)^n, \quad \text{MaxPro: } F_m = 1 - \prod_{k=1}^{n}(1 - \Phi_{km}). \tag{21}$$

Due to the overlap between neighboring descriptors, the probability based on the i.i.d. (independent and identically distributed) assumption will be overestimated. AxMin is designed to address this problem with a parameter β that accounts for the interdependence of descriptors.

$$\text{AxMin: } F_m = \min(1, \beta p) = \min\left(1, \beta\frac{1}{n}\sum_{k=1}^{n}\Phi_{km}\right), \quad 1 \le \beta \le n. \tag{22}$$

Normalization is used to cancel out the effect of the different number of extracted local descriptors in different input

images. With normalization, the midlevel features of input images can be cast on the same scale. Typical normalization methods include L_1-normalization [43], L_2-normalization [52], power-normalization [57], and intranormalization [58]. For L_1-normalization or L_2-normalization, the midlevel feature $\mathbf{F}$ is divided by its L_1-norm or L_2-norm, respectively.

$$L_1\colon F = \frac{F}{\|F\|_1}, \qquad L_2\colon F = \frac{F}{\|F\|_2}. \tag{23}$$

Power-normalization can be applied as the preprocess for L_1 or L_2. With power-normalization, the distribution of elements in the midlevel feature will be smoother.

$$\text{Power: } F_i = \operatorname{sign}(F_i)\,|F_i|^{\alpha}, \tag{24}$$

where F_i is the ith element in the midlevel feature $\mathbf{F}$ and $0 \le \alpha \le 1$ is the smoothing factor of normalization. The intranormalization method can be only applied for difference-based coding methods. This method uses L_1 or L_2 normalization operation in a block-by-block manner, where each block denotes the vector attached to one codeword.

$$\text{Intra: } F = \left[\frac{F^1}{\|F^1\|}, \frac{F^2}{\|F^2\|}, \ldots, \frac{F^m}{\|F^m\|}, \ldots, \frac{F^M}{\|F^M\|}\right], \tag{25}$$

where F^m is a vector related to the codeword b_m. Intranormalization is also used as the preprocess for L_1 or L_2. After intranormalization, L_1/L_2-normalization should be applied to the entire vector.

2.4. Relations of Different Coding Methods. Coding methods use the codebook $\mathbf{B} = [b_1, b_2, \ldots, b_M] \in \mathbb{R}^{D\times M}$ to map the descriptor space $\mathbf{X} = [x_1, x_2, \ldots, x_k] \in \mathbb{R}^{D\times K}$ to the coding space $\mathbf{D}$, and the pooling methods aggregate the coding result $\mathbf{D}$ into a single vector $\mathbf{F}$ of fixed length as the image-level representation. Those typical coding and corresponding pooling methods are summarized in Table 1.

Some relations of different coding methods are worthy of attention.

(1) Choice of Pooling Methods. Obviously, the choice of pooling methods creates an intuitive difference between the two types of coding methods, as shown in Table 1. The difference-based encoding methods record the difference between the descriptor space and codebook and connect those differences together in series to develop the midlevel feature, thus incurring notably little error. The pooling step is simply the series connection. In comparison, the activation-based encoding methods require many more codewords. Connecting the coding result in series directly results in an unacceptable increase in length of the midlevel feature. Thus, for activation-based encoding method, diverse pooling methods have been designed to generate an image-level representation with a reasonable vector length.

(2) Complexity and Dimensionality. The complexity and dimensionality as well as classification accuracy of various coding methods are affected by the codebook size to a great extent. The codebook size is a critical parameter for coding methods and should be determined critically with good trade-off between efficiency and effectiveness. Different types of coding methods need different sizes of codebooks, depending on the different coding theories. Generally, the difference-based coding methods have a smaller complexity compared with the activation-based coding methods due to much smaller codebook. It brings great convenience in steps of codebook generation and coding. However, the series connection pooling method of the difference-based coding methods will lead to a high dimension of the midlevel feature, while the midlevel features of activation-based coding methods fix their dimension in the same size as the size of codebook. Furthermore, the computational complexity increases significantly for the three coding methods, that is, SC, LCC, and LTC, due to their iteration optimization phase. Moreover, the SA may require a bit more coding time than LSA or LLC because it needs to calculate the activation of much more codewords.

(3) Assignment and Locality. The relations between different coding methods can also be found in the assignment strategy and locality. For the assignment strategy, two typical transformation strategies are used in those coding methods, namely, Hard Assignment and Soft Assignment. Hard Assignment associates the descriptor with only one codeword, whereas multiple codewords are associated with one descriptor in the Soft Assignment strategy. Considering the codeword uncertainty and plausibility, Soft Assignment leads to a more expressive model with a small coding error that improves classification performance. This Soft Assignment skill is used in several coding methods: SA versus HA and FV versus VLAD. Meanwhile, locality is another important aspect. This attribute is more essential than sparsity [45]. Using the local codewords, the descriptors in the same neighborhood tend to share codewords, whereas descriptors in different neighborhoods tend to have different codewords. Thus, similar descriptors will share the similar codewords. The locality makes the coding results of similar descriptors more similar and that of different descriptors more different, thereby delivering higher intraclass and lower interclass similarity. This locality factor can be found in several coding methods: LSA versus SA and LCC versus SC.

Using the same viewpoints, we can also extend VLAD to VLAD-k, LTC to LTC-k, and SVC to SVC-k by modifying the residual statistics φ_i from only the nearest visual word $NN(x_t)$ to the kth nearest visual words $NN_K(x_t)$. The difference φ_i of the VLAD-k and SVC-k coding methods is defined by

$$\varphi_i' = \sum_{x_t : NN_K(x_t) = i} (x_t - b_i). \tag{26}$$

There are a total of ten coding methods summarized in Table 1. HA is the common coding method used in the BOVW framework for terrain classification issues [16–18, 24]. We set the HA coding method as the baseline method in our study, while the computational costs of SC, LCC, and LTC are much more than other coding methods due to the iterative optimization process, which is obviously not suitable

Table 1: List of different encoding and pooling methods and their formulations.

	Coding method	Formulation	Coding Dim	Pooling Dim	Pooling method	Formulation
Activation-based coding methods	HA	$d_j = 1, \quad \text{if} j = \underset{j}{\operatorname{argmin}} \left\|x_i - b_j\right\|_2$	$M \times K$	M	Avg	$F_m = \frac{1}{\|K\|}\sum_{k=1}^{K} d_{km}$
	SA	$d_j = \frac{\exp(-\beta \widehat{e}(x_i, b_j))}{\sum_{l=1}^{M} \exp(-\beta \widehat{e}(x_i, b_l))}$ $\widehat{e}(x_i, b_j) = \|x_i - b_j\|^2$	$M \times K$	M	Max	$F_m = \max\left(\{d_{km}\}_{k \in K}\right)$
	LSA	$d_j = \frac{\exp(-\beta \widehat{e}(x_i, b_j))}{\sum_{l=1}^{M} \exp(-\beta \widehat{e}(x_i, b_l))}$ $\widehat{e}(x_i, b_j) = \|x_i - b_j\|^2 \quad \text{if } b_j \in N_k(x_i)$	$M \times K$	M	Lp	$F_m = \left(\frac{1}{\|K\|}\sum_{k=1}^{K} \|d_{km}\|^p\right)^{1/p}$
	SC	$D = \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_j \\ \vdots \\ d_K \end{bmatrix}$ $\underset{D}{\operatorname{argmin}} \sum_{i=1}^{K} (\|x_i - Bd_i\|^2 + \lambda \psi(d_i))$ $\psi(d_i) = \|d_i\|_1$	$M \times K$	M	MaxExp	$F_m = 1 - \left(1 - \frac{1}{n}\sum_{k=1}^{n} \Phi_{km}\right)^n$
	LCC	$\underset{D}{\operatorname{argmin}} \sum_{i=1}^{K} (\|x_i - Bd_i\|^2 + \lambda \psi(d_i))$ $\psi(d_i) = \|\widehat{e}_i \odot \|d_i\|\|_1, \quad \text{s.t. } 1^T d_i = 1, \forall i$ $\widehat{e}_i = [\widehat{e}(x_i, b_1), \ldots, \widehat{e}(x_i, b_m), \ldots, \widehat{e}(x_i, b_M)]^T$	$M \times K$	M	MaxPro	$F_m = 1 - \prod_{k=1}^{n} (1 - \Phi_{km})$
	LLC	$\underset{D}{\operatorname{argmin}} \sum_{i=1}^{K} (\|x_i - Bd_i\|^2 + \lambda \psi(d_i))$ $\psi(d_i) = \|E_i \odot d_i\|^2, \quad \text{s.t. } 1^T d_i = 1, \forall i$ $E_i = \exp(\frac{\widehat{e}_i}{\sigma})$ $\widehat{e}_i = [\widehat{e}(x_i, b_1), \ldots, \widehat{e}(x_i, b_m), \ldots, \widehat{e}(x_i, b_M)]^T$	$M \times K$	M	AxMin	$F_m = \min(1, \beta p) = \min\left(1, \beta \frac{1}{n}\sum_{k=1}^{n} \Phi_{km}\right)$

Table 1: Continued.

	Coding method	Formulation	Coding Dim	Pooling Dim	Pooling method	Formulation
Difference-based coding methods	FV	$D = [d_1^{(1)}, d_1^{(2)}, d_2^{(1)}, d_2^{(2)}, \ldots, d_M^{(1)}, d_M^{(2)}]$ $d_m^{(1)} = \frac{1}{K\sqrt{w_m}} \sum_{p=1}^{K} \alpha_p(m) \left(\frac{x_p - \mu_m}{\sigma_m} \right)$ $d_m^{(2)} = \frac{1}{K\sqrt{2w_m}} \sum_{p=1}^{K} \alpha_p(m) \left(\frac{(x_p - \mu_m)^2}{\sigma_m^2} - 1 \right)$	$D \times 2M$	$2DM$		Concatenation
	VLAD	$D = [d_1, d_2, \ldots, d_M]$ $d_i = \sum_{x_t : NN(x_t) = i} (x_t - b_i)$	$D \times M$	DM		
	LTC	$D = [\alpha\theta_1, \theta_1 d_1, \alpha\theta_2, \theta_2 d_2, \ldots, \alpha\theta_M, \theta_M d_M]$ $d_i = \sum_{x_t : NN(x_t) = i} (x_t - b_i)$ $\theta = [\theta_1, \theta_2, \ldots, \theta_M]$ defined by LCC	$(D+1) \times M$	$(D+1)M$		
	SVC	$D = [\alpha\theta_1, \theta_1 d_1, \alpha\theta_2, \theta_2 d_2, \ldots, \alpha\theta_M, \theta_M d_M]$ $d_i = \sum_{x_t : NN(x_t) = i} (x_t - b_i)$ $\theta = [\theta_1, \theta_2, \ldots, \theta_M]$ defined by LSA	$(D+1) \times M$	$(D+1)M$		

for terrain classification application. Thus, we select another six coding methods for evaluation, that is, three activation-based encoding methods (i.e., SA, LSA, and LLC) and three difference-based encoding methods (i.e., VLAD, SVC, and FV), to design an optimum pipeline for better visual terrain classification performance.

3. Fusion Methods

Fusion methods are often used to boost the performance of the classification system in various scenarios. Due to high intraclass variability and interclass similarity, visual terrain classification is a challenging task. Various descriptors are designed for effective classification, but it is clear that none of the descriptors will have the same discrimination power for all terrain classes. Therefore, it is a natural choice to combine a set of diverse and complementary features for better classification performance.

Fusion methods generally include early fusion and late fusion [27, 28]. Early fusion is performed in low-level feature space, that is, descriptor space, where multiple descriptors would be concatenated into a single one, and it is subsequently fed into the BOVW framework to obtain image-level visual terrain representation. Before combination, the diverse descriptors must be normalized into the same scale. It is worth noting that early fusion is only for local descriptors because its purpose is to develop a new low-level descriptor. However, late fusion can fuse local and global descriptors by performing in midlevel future space. Different descriptors are inputted into the BOVW framework separately, and the results can be fused as a single representation or fused with global descriptors. The midlevel features and global descriptors must be normalized into the same scale before fusion. Late fusion uses kernel methods that make use of kernel functions to define a measure of similarity between pairs of instances [28]. A kernel corresponds to a feature, and the kernel fusion method is used to generate the final representation. Typical kernel fusion methods include the averaging kernel, product kernel, multiple kernel learning (MKL), LP-β, and LP-B. However, averaging and product kernels yield competitive results and outperform other sophisticated fusion methods [27]. Considering effectiveness and efficiency, we only evaluate two types of late fusion methods for visual terrain classification in this study, that is, averaging linear kernel and product linear kernel.

4. Empirical Study

In this section, we describe a detailed empirical study of different BOVW framework and fusion methods designed to extensively explore their effects on visual terrain classification and subsequently design an optimum pipeline and develop the hybrid representation to produce an effective and efficient visual terrain classification system. First, the datasets used for evaluation and experimental setups are introduced. Next, the key parameter, codebook size, is determined for the six coding methods and then we optimize their framework to ensure which preprocessing, pooling, and normalization approaches to be applied are appropriate. Then, those coding methods are evaluated with two fusion methods. Through the detailed empirical study, we construct an optimum pipeline and develop the hybrid representation for visual terrain classification. Finally, the performance of our hybrid representation is compared with that of the baseline method using the dataset DS1.

4.1. Experimental Setup. Due to the lack of an available terrain image benchmark, we created the dataset DS1 with eight different terrain classes: *asphalt*, *dirt*, *grass*, *floor*, *gravel*, *rock*, *sand,* and *wood chips*. This dataset contains 2400 images, and each terrain class contains 300 samples of the same size, that is, 256 pixels × 256 pixels. Certain images are captured with a camera under different weather conditions, and the others come from *Google Image Search*. Most of the images in DS1 are collected with the camera facing downward to the ground, similar to those in [24]. To the best of our knowledge, dataset DS1 is the largest visual terrain dataset created thus far. Selected typical examples of different terrain classes in DS1 are illustrated in Figure 3.

In our experiments, we evaluated sparse descriptors (SIFT) and dense descriptors (DSIFT) simultaneously. These categories represent two different types of low-level descriptors and exhibit different properties with respect to variations of the pipeline. The intuitive differences in the midlevel features between different descriptors (SIFT versus DSIFT) of three different typical coding methods can be found in Figure 4.

For all three coding methods, the visual expression of midlevel features between SIFT and DSIFT is quite different. Much overlapping information of DSIFT descriptors may generate some uninformative codewords in the codebook through the clustering. For activation-based encoding methods, for example, HA and LLC, this results in the decreasing number of activated codewords. For difference-based encoding methods, for example, FV, the assignment weight $\alpha_p(m)$ becomes zero to some codewords. Due to this factor, the midlevel features of DSIFT appear to be of more portion in blank or much sparser than that of SIFT as shown in Figure 4. Thus, it is well worth exploring both SIFT and DSIFT and analyzing their characteristics for visual terrain classification.

In the paper, half of the images in each category of the dataset DS1 are fixed for training and the rest for testing. The accuracy is defined as the proportion of the correct predictions made by the model against the total test data. We use the 10-fold average accuracy as the evaluation index and the confusion matrices to visualize the results of terrain classification. The SIFT and DSIFT descriptors are computed using the code (https://sites.google.com/site/handsonbow/) released on the website of Lamberto Ballan, and DSIFT is extracted from patches densely located according to every 6 pixels on the terrain image under a single scale of 32 × 32. With respect to codebook generation, 700,000 local descriptors are randomly sampled from the descriptor space in the training set for building the K-means or GMM codebook. The number of local descriptors to build codebook is chosen according to the result of preexperiment for a trade-off between effectiveness and efficiency. The K-means

Figure 3: Terrain samples in dataset DS1. This dataset contains 2400 images grouped into eight terrain types, that is, "asphalt"; "dirt"; "grass"; "floor"; "gravel"; "rock"; "sand"; and "wood chips."

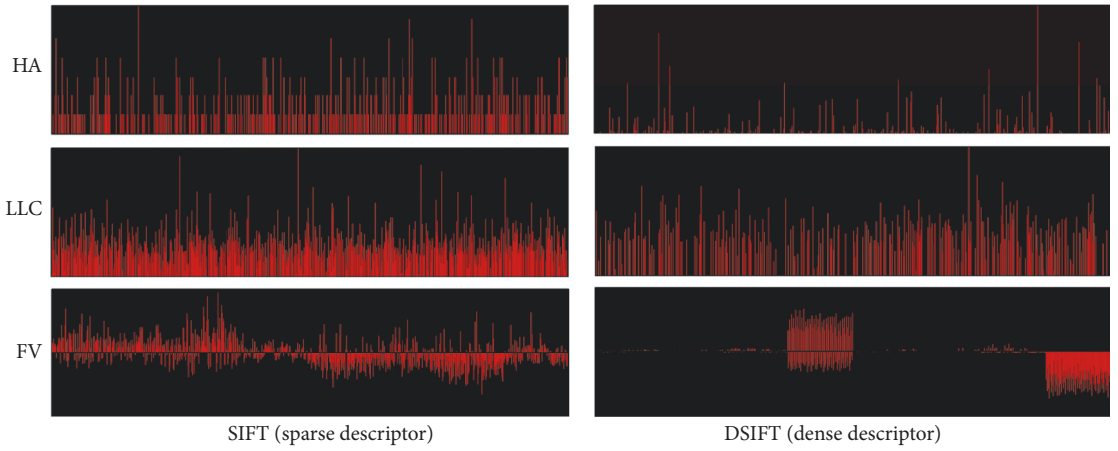

Figure 4: Midlevel features of different coding methods based on SIFT and DSIFT from the gravel class of dataset DS1.

codebook size ranges from 100 to 3200, and the GMM codebook size ranges from 2 to 64.

Finally, we choose the Support Vector Machine (SVM) as our classifier, and, specifically, we use the implementation of LIBSVM [59]. The nonlinear kernels (e.g., RBF kernel, intersection kernel, and chi-square kernel) are also tested in our evaluation. However, the efficiencies of these methods are much worse compared with the linear kernel. Due to efficiency and scalability, we applied our experiment together with the linear kernel only, and C value is selected via 10-fold cross-validation.

4.2. Exploration of Coding Methods. In this section, the terrain classification performances of different coding methods are compared and analyzed to determine the key parameter, that is, codebook size. Six coding methods, that is, three activation-based encoding methods (i.e., SA, LSA, and LLC) and three difference-based encoding methods (i.e., VLAD, SVC, and FV), are selected for evaluation. For each coding method, we fix the other parameter settings (the following pooling and normalization strategy), similar to the approaches in previous papers [42, 45, 49, 58]. PCA-whitening technology is also applied for each coding method. The experimental results of SIFT and DISFT on DS1 are illustrated in Figure 5.

Several observations can be concluded from these results:

(1) The terrain classification performance of all six coding methods increases with a larger codebook and reaches a plateau when the codebook size exceeds a threshold. A small codebook will result in a large quantization loss and the large codebook can cause overpartitioning in the descriptor space, which means that no terrain descriptor might fall into those codewords. Above all, for a good trade-off between effectiveness and efficiency, sizes of 800 and 16 are

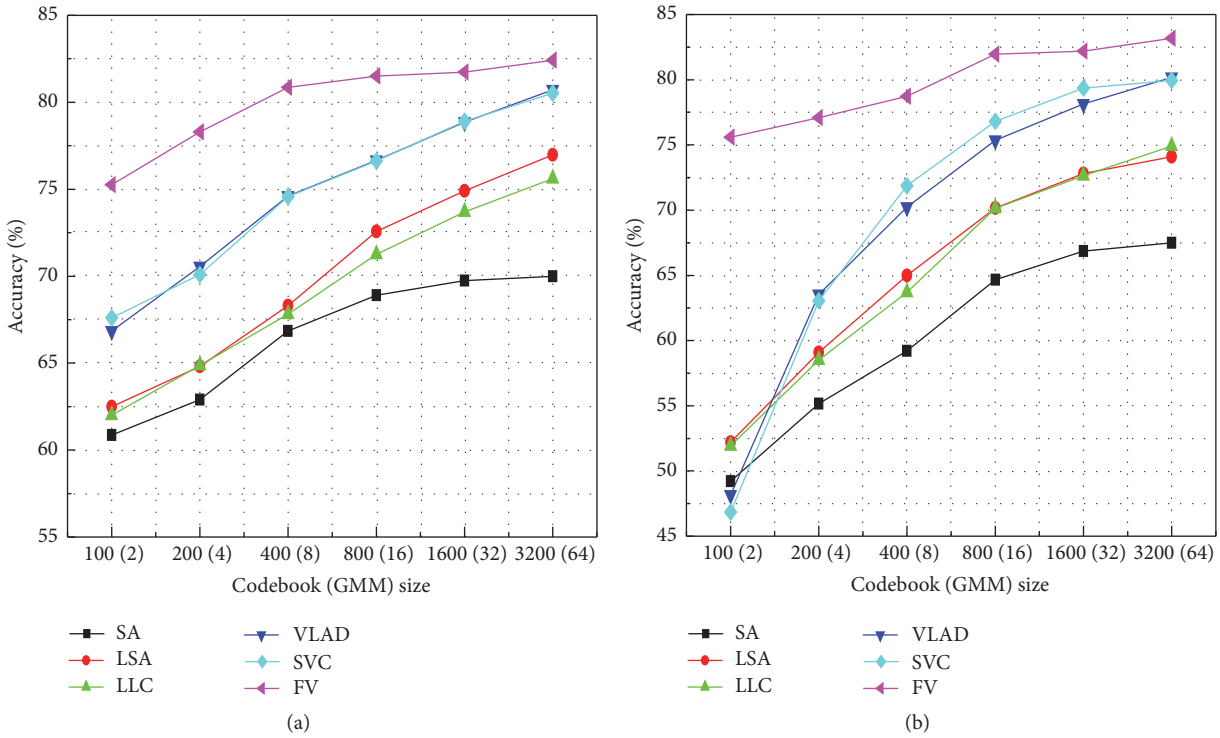

Figure 5: Performance of different encoding methods with varying codebook sizes on the DS1 dataset for SIFT and DSIFT descriptors. (a) SIFT (sparse descriptor); (b) DSIFT (dense descriptor). The numbers in x-axis outside (inside) the parenthesis indicate the codebook size for activation (difference)-based coding methods.

good choices for activation-based encoding methods and difference-based encoding methods, respectively.

(2) For both SIFT and DSIFT, the performance of difference-based encoding methods has an edge over that of activation-based encoding methods for visual terrain classification on dataset DS1. Difference-based encoding methods use multidimensional information (0th, 1st, and 2nd) from the descriptor space. This type of method can retain much richer information on the visual terrain descriptors. For the activation-based encoding methods, LLC and LSA outperform SA due to the locality constraint. In the difference-based encoding methods, FV is better than VLAD and SVC, especially with a small codebook. These two factors are worth noting: (1) compared with VLAD (1st statistics) and SVC (0th and 1st statistics), FV retains both 1st and 2nd statistics, providing much richer and more discriminating visual terrain classification. Especially with a small codebook, FV can extract sufficient information from notably limited material. (2) FV softly assigns each descriptor to codewords, whereas VLAD and SVC use Hard Assignment. Soft Assignment reduces the information loss. In short, difference-based encoding methods with much richer statistics of descriptor space might be more suitable for obtaining better visual terrain classification accuracy.

(3) Sparse (SIFT) and dense (DSIFT) descriptors represent two different types of low-level descriptors. In general, these methods are similar for certain major trends, for example, trends with the change in codebook size, and difference-based encoding methods outperform others. However, different aspects can be noted between SIFT and DSIFT. The most obvious aspect is the sensitivity to codebook size. The dense descriptor (DSIFT) is more sensitive to a small codebook, as attributed to the stronger correlation of DSIFT descriptors, which makes it more difficult to extract sufficient useful information if the codebook size is small. However, FV is an exception that provides consistent performance for both SIFT and DSIFT due to its powerful method of obtaining information. In conclusion, the type of descriptor is a necessary factor for consideration in choosing the codebook size and coding methods.

Efficiency is also an important aspect in visual terrain classification. The testing time is compared in Figure 6. Specifically, we randomly sample 150 images of each terrain category from DS1 and record the total encoding time, while, due to the demand of online-training for visual terrain classification applications [60, 61], the training time is also taken into account and illustrated in Figure 7. It consists of codebook generation time and coding time for the train set. Our codes are all implemented in MATLAB 2014a and run

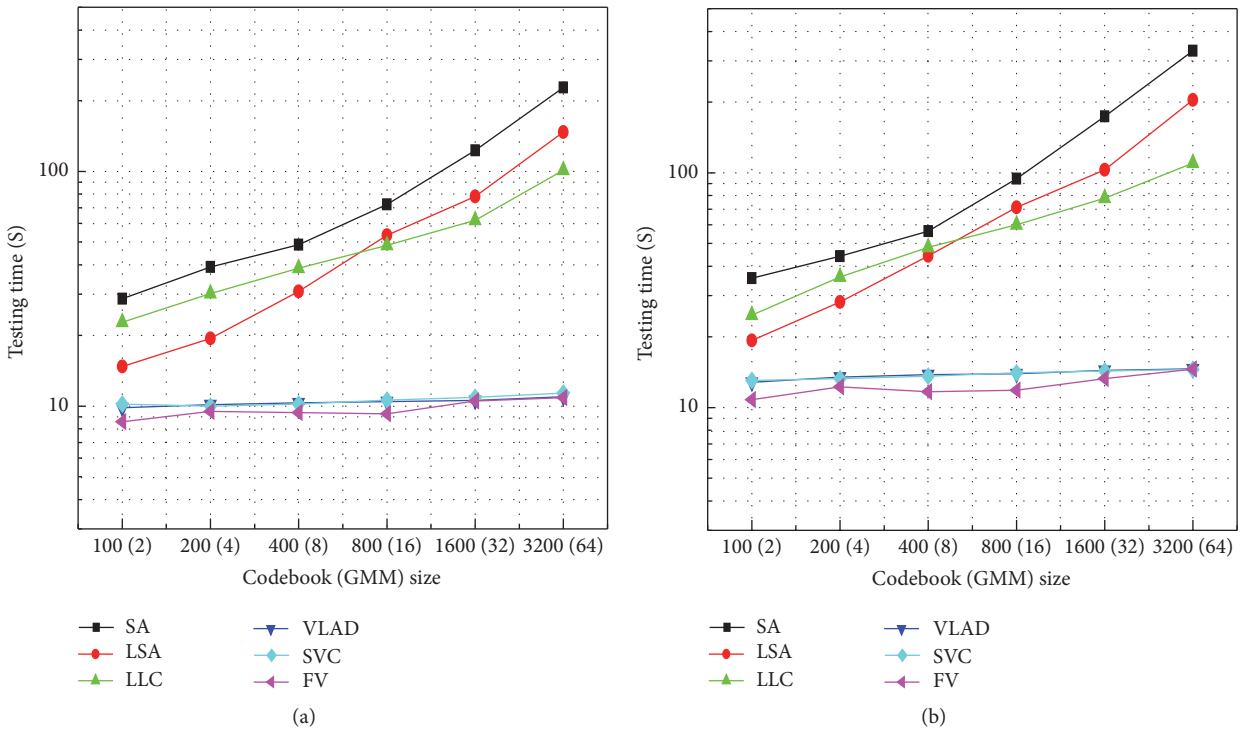

Figure 6: The testing time of 1200 terrain images (256 pixels × 256 pixels) for different encoding methods with varying codebook size on the DS1 dataset for SIFT and DSIFT descriptors. (a) SIFT (sparse descriptor); (b) DSIFT (dense descriptor). The numbers in x-axis outside (inside) the parenthesis indicate the codebook size for activation (difference)-based coding methods.

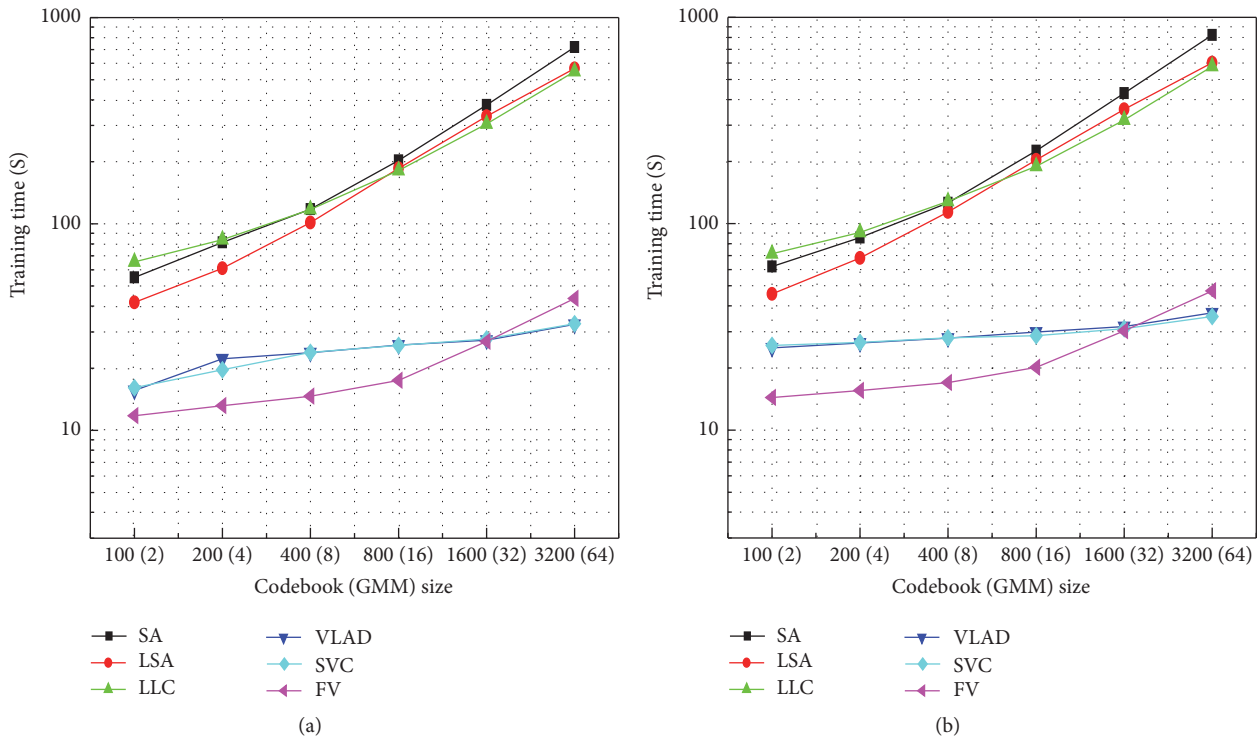

Figure 7: The training time (training set: 1200 terrain images) for different encoding methods with varying codebook size on the DS1 dataset for SIFT and DSIFT descriptors. (a) SIFT (sparse descriptor); (b) DSIFT (dense descriptor). The numbers in x-axis outside (inside) the parenthesis indicate the codebook size for activation (difference)-based coding methods.

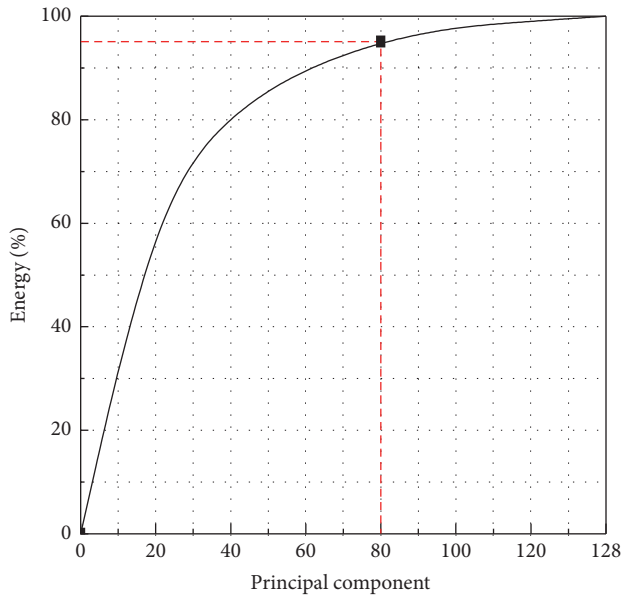

Figure 8: Energy distribution over principal components in DS1 dataset.

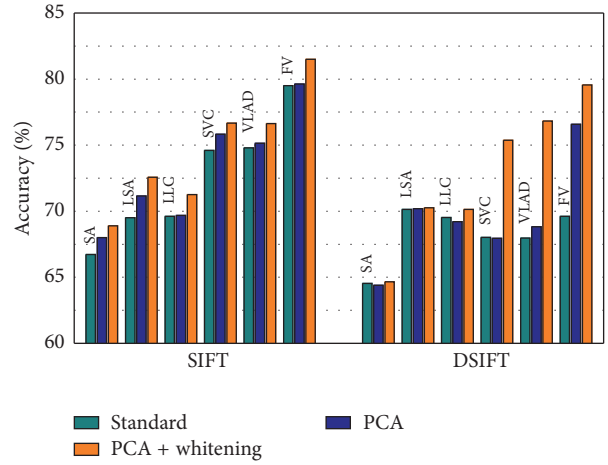

Figure 9: Comparison of PCA-whitening, PCA, and standard (without PCA-whitening) techniques for different encoding methods on the DS1 dataset. The codebook sizes are chosen as 800 (SA, LSA, and LLC) and 16 (SVC, VLAD, and FV).

on a computer with an Intel(R) Xeon(R) CPU E5-2620 v2 @ 2.1 GHZ and 32 G RAM in a 64-bit Win7 operation system.

For activation-based encoding methods, SA requires the most testing time because it needs to calculate the activation of each codeword in the codebook, and the training times for the three coding methods are nearly the same. For difference-based encoding methods, SVC, VLAD, and FV display nearly the same testing time. The computational cost of difference-based encoding methods is more than several times lower than that of activation-based encoding methods. The similar phenomenon also appears in the training phase. Difference-based encoding methods can be implemented much faster mainly due to the much smaller codebook size. Moreover, it can be observed that DSIFT spend a bit more time on both test and train phase because there are more descriptors to be processed compared to SIFT.

Based on the above analysis, difference-based encoding methods may be more promising for good performance and fast implementation in visual terrain classification.

4.3. Principal Component Analysis and Whitening Techniques. In this section, the Principal Component Analysis (PCA) and whitening preprocessing techniques are explored. With the PCA technique, the terrain descriptors of SIFT and DSIFT are reduced from 128-dimension to 80-dimension terrain descriptors, reducing space costs by 37.5% while retaining 95% of the energy, that is, the percentage of variance retained. The storage cost is an even more important aspect of the visual terrain classification application for the robot. The result is shown in Figure 8.

Three activation-based encoding methods (i.e., SA, LSA, and LLC) and three difference-based encoding methods (i.e., VLAD, SVC and FV) are chosen for elevation of the PCA and whitening technique. The max pooling is used for activation-based encoding methods, and we unified the use of L2-normalization. A codebook size of 800 is chosen for activation-based encoding methods, and a size of 16 is used for difference-based encoding methods. The result is illustrated in Figure 9.

For both the SIFT and DSIFT descriptors, the PCA-whitening preprocessing technique boosts the performance of different encoding methods. Particularly, the performance of FV is significantly enhanced for DSIFT, improving by 10.4 (from 69.2 to 79.6). The FV maintains both the 1st- and 2nd-order statistical information of the descriptor space, and the correlations of those descriptors have a much greater effect. PCA-whitening technology can aid in FV extraction of much more effective difference information. Meanwhile, no significant performance degradation can be found for six different coding methods with PCA technique because most of information of the descriptors is retained. Furthermore, it can be observed that whitening has a stronger effect on the DSIFT descriptors, which are extracted from dense patches on the terrain images. Due to the correlated adjacent pixel value, a large number of similar descriptors could be repeatedly extracted, and the correlation of those descriptors would be stronger. Thus, the decorrelation technique, that is, whitening, is a more important step for DSIFT descriptors.

Many previous terrain classification approaches have ignored this preprocessing step. However, we find that PCA-whitening is highly powerful technique for boosting the performance of terrain classification while reducing the space burden. We add this step into the pipeline. In the following portion of the evaluation, we use PCA to reduce the SIFT and DSIFT descriptors to 80-dimension ones and apply the whitening technique to decorrelate the visual terrain descriptors.

4.4. Exploration of Pooling Methods. In this section, we compare and analyze the performance of different pooling methods for visual terrain classification. For the difference-based encoding methods, the pooling method is series

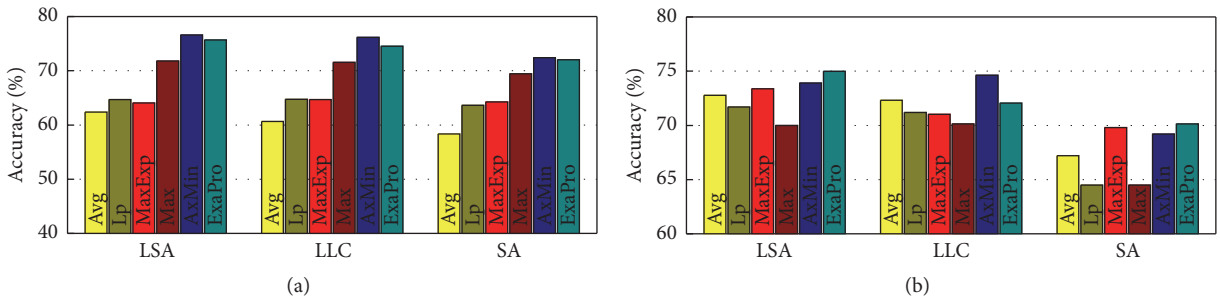

Figure 10: Comparison of different pooling methods on the DS1 dataset for SIFT and DSIFT descriptors. (a) SIFT (sparse descriptor); (b) DSIFT (dense descriptor).

connection. We focus on pooling methods for the activation-based encoding methods, that is, LSA, LLC, and SA. For this type of encoding methods, series connection will lead to the millions of dimensions of midlevel feature due to their much more codewords, causing the curse of dimensionality. Six pooling methods are selected, including Avg, Max, L_P, MaxExp, ExaPro, and AxMin. Based on the analysis in the previous section, the codebook size is chosen as 800. For each pooling method, we fix other parameters at values that match those of previous papers [42, 50, 53]. The experimental results of SIFT and DISFT on DS1 are illustrated in Figure 10.

Several observations can be concluded from these results:

(1) The choice of pooling method is critical for the activation-based encoding methods, a key step in developing the image-level representation. Different selections lead to dramatic performance differences; for example, the classification accuracy is improved from 60.7 (Avg) to 76.2 (AxMin) for LLC coding method with a sparse descriptor (SIFT), which is an increase of 15.5 and relative growth of almost 26%.

(2) For visual terrain classification, max pooling is better for sparse descriptors, whereas Avg pooling is more suitable for dense descriptors. L_P represents a theoretical trade-off between those two methods, and, thus, the performance falls between those of the Max and Avg pooling methods. Likelihood-based pooling methods are designed from the perspective of probability to describe codewords on the input terrain image. Those methods display good performance for both the sparse descriptors (SIFT) and dense descriptors (DSIFT).

(3) An appropriate pooling method can reduce the performance gap between different coding methods and allow the activation-based encoding methods to obtain a performance similar to that of difference-based encoding methods. In particular, discriminative combinations of the best performance for visual terrain classification include SA-AxMin, LSA-AxMin, and LLC-AxMin for the sparse descriptor (SIFT) and SA-ExaPro, LSA-AxMin, and LLC-ExaPro for dense descriptors (DSIFT).

Based on the above analysis, likelihood-based pooling methods are more promising for good performance in the activation-based encoding methods. The discriminative combinations are used in the following evaluation.

4.5. Exploration of Normalization Methods. In this section, we investigate the influence of different normalization methods on visual terrain classification accuracy. We explore eight normalization methods specified as with or without power-normalization, with or without intranormalization, and final L_1- or L_2-normalization. The eight coding methods used are FV, SVC, SVC_K, VLAD, VLAD_K, SA, LSA, and LLC, where the SVC_K and the VLAD_K are the variants of SVC and VLAD. The codebook size and pooling strategies are set as that in the previous analysis. The experimental results of SIFT and DISFT on the DS1 dataset are illustrated in Figure 11.

Normalization is a key step in generating the midlevel features. Figure 11 shows that the choice of normalization methods has a great impact on the visual terrain classification accuracy. Several observations can be concluded from these results:

(1) For different coding methods and local descriptors, L_1- or L_2-normalization is the most critical choice. For both SIFT and DSIFT, L_2-normalization generally outperforms L_1-normalization for almost all eight coding methods. For instance, SC increased 15.1 from 56.7 (L_1) to 71.8 (L_2) with SIFT, and SVC_K has increased 8.6 from 68.6 (L_1) to 77.2 (L_2) with DSIFT. L_2-normalization can remove the influence between the background image-independent and the image-specific components [57].

(2) Power-normalization reduces the difference between different codewords, which makes the representation smoother. This smoothing effect can reduce the influence of "strong" codewords on the kernel calculation and improve the influence of "weak" codewords. For visual terrain classification, the resulting representation for difference-based encoding methods is at times quite sharp and unbalanced due to feature burst, and the smoothing operation has a positive effect. However, the representation is usually not as sharp with activation-based encoding methods, and thus power-normalization might create side effects.

(3) Intranormalization is used to balance the weight of different codewords for difference-based encoding

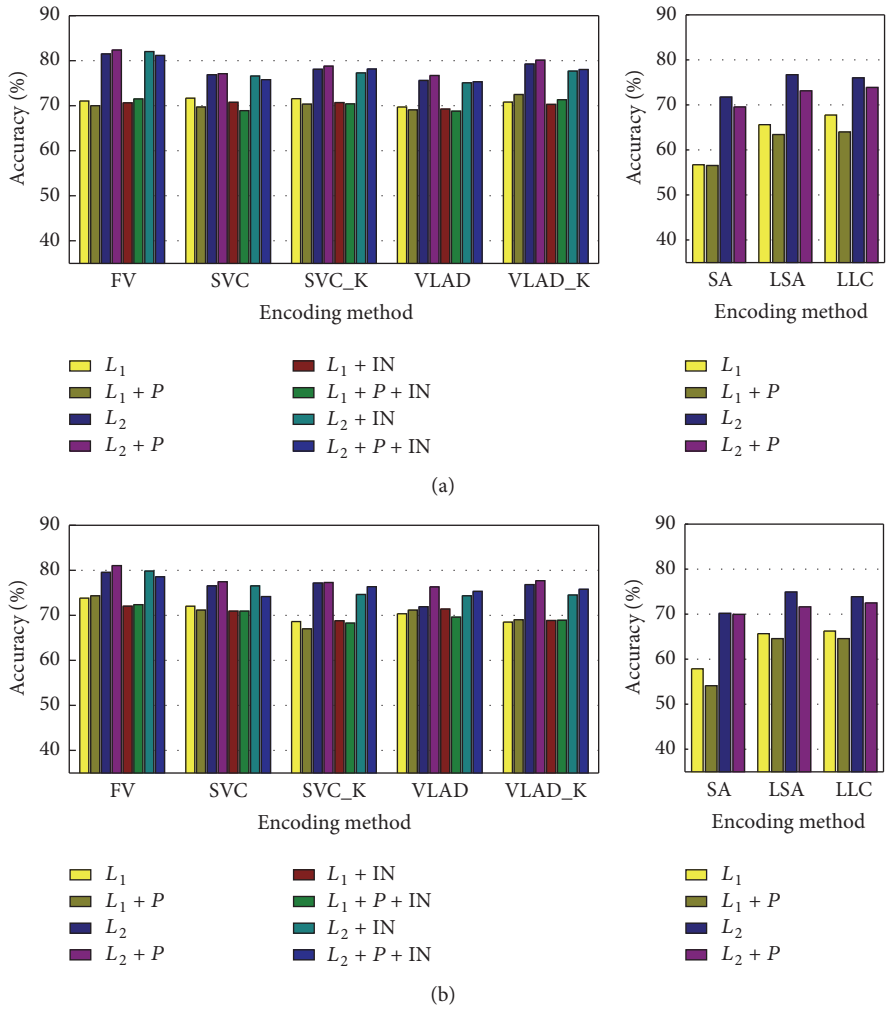

Figure 11: Comparison of different normalization methods on the DS1 dataset for SIFT and DSIFT descriptors. Intranormalization can be used only in difference-based encoding methods, and, hence, eight types of normalization methods are evaluated for difference-based encoding methods (left) and four types are evaluated for activation-based encoding methods (right). (a) SIFT (sparse descriptor); (b) DSIFT (dense descriptor).

methods. Intranormalization performs L_1- or L_2-normalization operation in a block-by-block manner in which each block denotes the vector attached to one codeword. This method develops a more balanced expression between codewords; however, it might also amplify background noise and weaken the expression of discriminative information. Judging from the experimental results, this method can have a positive or negative effect for different coding methods.

In conclusion, normalization is a key step in generating midlevel features and has a great effect on the visual terrain classification accuracy. L_1- or L_2-normalization is the most critical choice. The power operation plays a positive role in the difference-based encoding methods and has a negative effect on the activation-based encoding methods. Intranormalization has little to do with the performance of visual terrain classification. Thus, we choose power-L_2-normalization for different-based encoding methods and L_2-normalization for activation-based encoding methods.

4.6. Exploration of Fusion Methods. Visual terrain includes both the global and local information, and global and local features are naturally complementary. We choose the typical global feature GIST [62] to add global information to the midlevel features to boost the performance of visual terrain classification. The GIST can provide eligible and stable global description under different environmental conditions. Its robustness is better than other descriptors, which is verified in the following section. Late fusion is used because the fusion runs in the midlevel feature space. In this section, we primarily analyze the influence of different fusion methods on the final terrain classification performance.

For coding methods, the same eight approaches are chosen as in the previous sections. For difference-based

Table 2: Comparison of different fusion methods for the coding methods for SIFT on DS1. The average classification accuracy (%) is set as the evaluation index.

Methods	SA	LSA	LLC	SVC	SVC_K	VLAD	VLAD_K	FV
SIFT	72.17	76.13	76.05	76.97	79.23	76.50	80.13	**82.38**
GIST					**73.55**			
Averaging	83.17	86.12	85.60	85.20	87.70	85.72	86.45	**88.50**
Product	75.14	**81.00**	80.36	67.36	75.78	67.50	77.28	74.28

Table 3: Comparison of different fusion methods for the coding methods for DSIFT on DS1. The average classification accuracy (%) is set as the evaluation index.

Methods	SA	LSA	LLC	SVC	SVC_K	VLAD	VLAD_K	FV
SIFT	70.17	74.68	74.38	76.03	77.18	75.43	77.10	**81.02**
GIST					**73.66**			
Averaging	84.32	86.07	85.98	85.63	86.65	85.60	86.47	**87.72**
Product	76.53	**81.33**	78.72	68.42	74.11	68.64	76.42	73.06

encoding methods, the codebook size is set to 16, and we use power-L_2-normalization. For activation-based encoding methods, the codebook size is set to 800, the discriminative pooling combinations are used as previously stated, and L_2-normalization is applied. For fusion methods, averaging and product kernels yield competitive results and outperform other sophisticated methods [27]. Considering effectiveness and efficiency, we only evaluate the two types of late fusion methods for visual terrain classification, that is, the averaging linear kernel and product linear kernel. The experimental results of SIFT and DSIFT are shown in Tables 2 and 3, respectively.

It can be found from the experiment results that the appropriate fusion method is a highly important component for handling a combination of global and local information in the visual terrain classification system. For sparse or dense descriptors, the performance of all coding methods improves with the addition of global information via the averaging kernel fusion method. Among these, the best hybrid representation combines GIST and the midlevel features based on FV using the average kernel. To further illustrate the effect of the fusing method, the preclass classification accuracy corresponding to different features and fusion methods is shown in Figure 12.

In Figure 12, it can be observed that the performance improvement primarily relies on the complementarity of different features. For sparse or dense descriptors, GIST produces stronger discrimination in asphalt and grass. Compared with the discriminative midlevel featured, the accuracy of GIST is 10–20% better. In other terrain types without such obvious global characteristics, discriminative midlevel features can obtain much better performance than GIST. In conclusion, strong complementarity exists between these two representations.

At the same time, we find that the average kernel is much better than the product kernel; that is, the average kernel is more suitable for visual terrain classification. The average kernel can synthesize the advantages of different methods, and the advantages of one method can compensate for the shortcomings of another method, thus obtaining better results. In contrast, the product kernel easily imports the defects of any given method into the final result for visual terrain classification.

4.7. Optimum Pipeline for Visual Terrain Classification. Based on the above analysis, we find that every step is crucial in contributing to the final classification performance, and an improper choice in one step will greatly weaken the effectiveness and efficiency of the visual classification system as a whole. We use the feature preprocessing technique, the improved BOVW framework, and the fusion method to construct an optimum pipeline for visual terrain classification. The PCA-whitening technique is used to apply preprocessing and we improve the traditional BOVW framework using FV coding method and power-L_2-normalization. Then, the midlevel features and global features are combined with the average kernel fusion method. In the end, the hybrid representation is developed through the optimum pipeline, completing the terrain classification task effectively and efficiently. Previous visual terrain classification methods [5, 6, 11–17] always use primitive BOVW framework, that is, HA coding-average pooling-L_1-normalization and choose to construct new handcrafted low-level feature or apply sophisticated classifier. In contrast, our method focuses on improving BOVW framework to design an optimum pipeline for visual terrain classification. Taking into account the efficiency of evaluations, our study uses the *off-the-shelf* low-level feature and classifier, that is, SIFT and SVM. Those previous approaches are orthogonal to ours and better results will be achieved by a combined setup.

We chose the algorithm developed by Filitchkin and Byl in 2012 as the baseline method and their visual terrain classification algorithm (https://code.google.com/archive/p/opencv-visclass/) has been successfully applied in the small quadruped robot *Littledog* [17]. The baseline method uses the HA coding methods, average pooling, and L_1-normalization. The confusion matrices of the hybrid representation and the baseline method for SIFT are shown in Figure 13. For sparse descriptors, hybrid representation improves the visual terrain classification mean accuracy from

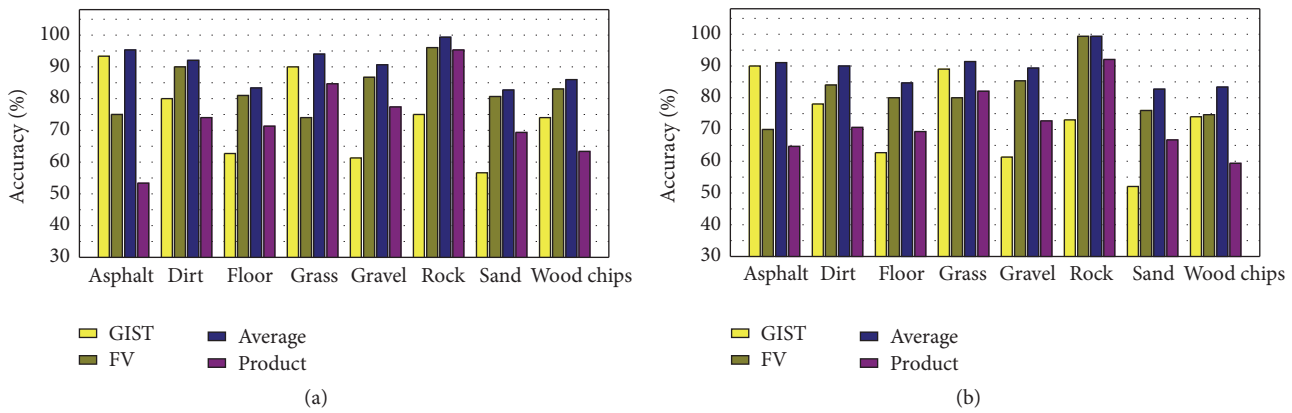

Figure 12: Preclass classification accuracies corresponding to different features, that is, GIST and discriminative midlevel feature, and different fusion methods, that is, average kernel and product kernel. (a) SIFT (sparse descriptor); (b) DSIFT (dense descriptor).

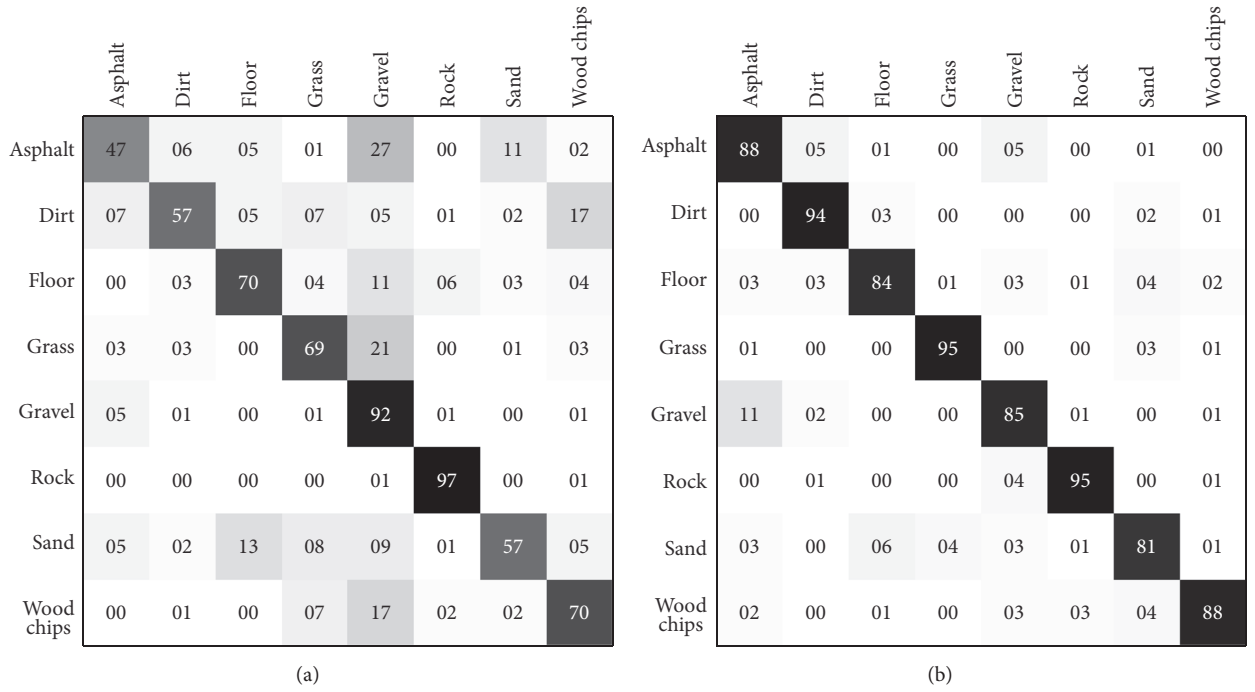

Figure 13: Confusion matrix of the hybrid representation and the baseline method for SIFT on the DS1 dataset. (a) Baseline method; (b) hybrid representation. The average classification accuracy (%) is set as the evaluation index.

69.5 (baseline method) to 88.7, an increase of 19.2, and relative growth of almost 28%. At the same time, the terrain classification ability for almost all eight terrain types is also enhanced, especially for asphalt (from 47 to 88, a growth of 41%), dirt (from 57 to 94, a growth of 37%), and sand terrain (from 57 to 81, a growth of 24%).

For dense descriptor DSIFT, the confusion matrices of the hybrid representation and baseline method are shown in Figure 14. Hybrid representation improves the visual terrain classification mean accuracy from 65.6 (baseline method) to 87.7, an increase of 22.1, and relative growth of almost 34%. The classification accuracy increases for almost all eight terrain types, especially for sand (from 37 to 83, a growth of 46%), grass (from 44 to 96, a growth of 52%), and wood chips terrain (from 66 to 89, a growth of 23%).

Efficiency is another key role in visual terrain classification. For different methods, we also tested the whole running time, including *feature extraction*, *encoding*, *pooling*, *normalization,* and *fusion*. The test environment is the same as in the previous sections. The running time of 1200 testing images in DS1 and the average classification accuracies are illustrated in Figure 15. For sparse descriptors, the classification accuracy increases from 69.5 to 88.7, a relative growth of almost 28%, and the running time decreases from 518 to 454, a relative reduction of 12.3%. For dense descriptors, the classification accuracy increases from 65.6

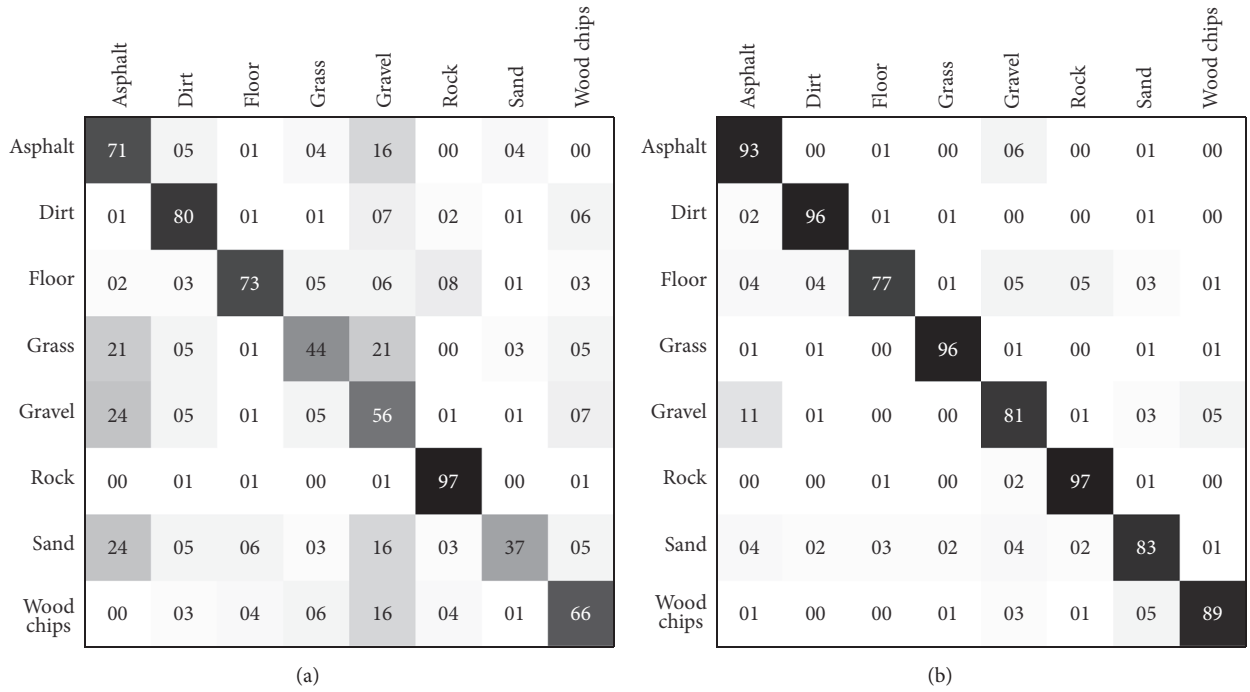

Figure 14: Confusion matrix of the hybrid representation and the baseline method for DSIFT on the DS1 dataset. (a) Baseline method; (b) hybrid representation. The average classification accuracy (%) is set as the evaluation index.

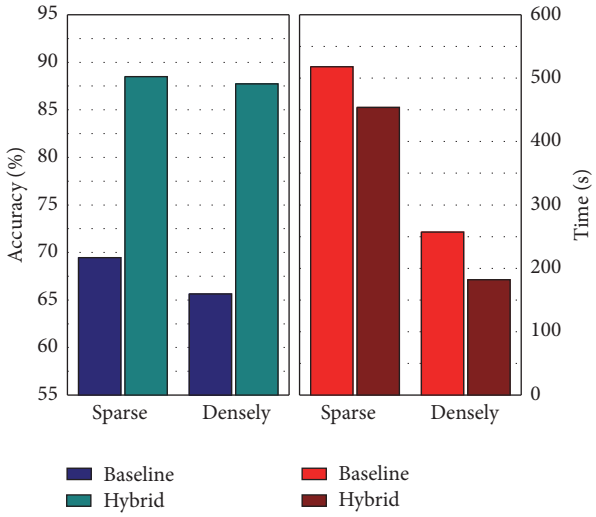

Figure 15: Comparison of the visual terrain average classification accuracy and running time (1200 terrain images) between the hybrid representation and baseline method for SIFT and DSIFT on the DS1 dataset.

to 87.7, a relative growth of almost 34%, and the running time decreases from 257to 182, a relative reduction of 29.1%.

Hybrid representation uses additional global descriptors that are more time-consuming in feature extraction step, and the fusion step also increases the running time. However, the improved BOVW frameworks greatly improve the efficiency. As a whole, the running time of hybrid representation is much reduced.

In these tests, the hybrid representation performs effectively and rapidly, greatly improving the accuracy and running speed compared with the baseline method. This approach can effectively and efficiently complete the visual terrain classification task.

5. Discussion

In this section, we perform further analysis on several important components of the proposed hybrid representation. First, we analyze the contribution of each component in the optimum pipeline to the performance improvement, and, second, we evaluate the impacts of noise and illumination alteration. Finally, we study the effect of training samples on the hybrid representation.

5.1. Performance Improvement Analysis. In the previous section, we showed that hybrid representation performs much better than the baseline method. To further analyze the performance gains, in this section, we evaluate the individual performance improvement provided by each component in the optimum pipeline using the *PCA-whitening technique*, *improved BOVW framework*, and *average kernel fusion method*. The joint activation of the three steps leads to eight different configurations for which the performance of the final representation is evaluated. The results are shown in Table 4. Each configuration is tested 10 times on DS1 dataset for sparse (SIFT) and dense (DSIFT) descriptors, and accuracy is measured in terms of MAP (in %).

Table 4: Impact of the three proposed improvements from baseline method to hybrid representation. The first line (no modification applied) corresponds to the baseline method, and the last line (three modifications applied) corresponds to the hybrid representation developed by the optimum pipeline. Accuracy is measured in terms of MAP (in %).

Method	PCA-whitening	Improved BOVW framework	Fusion methods	Sparse descriptor	Dense descriptor
Baseline	No	No	No	69.45	65.63
	No	Yes	No	80.20	71.73
	Yes	No	No	71.75	67.47
	Yes	Yes	No	82.13	80.65
	No	No	Yes	77.27	76.57
	No	Yes	Yes	83.75	84.62
	Yes	No	Yes	80.60	79.27
Hybrid representation	Yes	Yes	Yes	88.70	87.92

Table 4 shows the influence of each component in the optimum pipeline individually or combined together. The most important single improvement is *improved BOVW framework* for sparse descriptors and *average kernel fusion* for dense descriptors. Combinations of two components improve over a single component, and the combination of all three components delivers an additional increase. We also used a factorial analysis of variance (ANOVA) [63, 64] method to quantify the relative impact of each component. With 95% confidence, for the sparse descriptor (SIFT), the *improved BOVW framework* contributes almost 55.9% of the improvement, whereas the *average kernel fusion* and *PCA-whitening technique* are responsible for 31.5% and 6.9% improvement, respectively. For a dense descriptor (DSIFT), the largest impact is clearly the *average kernel fusion*, which is responsible for 51% improvement. Due to the stronger correlation of descriptors, the contribution of the *PCA-whitening technique* increases to 10.8%, and the *improved BOVW framework* explains almost 35.9%.

In conclusion, the contribution ranking for sparse descriptors is *improved BOVW framework* > *fusion methods* > *PCA-whitening*, and the contribution ranking for dense descriptors is *fusion methods* > *improved BOVW framework* > *PCA-whitening*.

5.2. Robustness. Visual terrain classification must address different environmental conditions, which typically include diverse noises and illumination alterations. In this section, we evaluate the impact of noise and illumination alteration on five classification methods, that is, the baseline method, ColorHist, LBP, GIST, and hybrid representation. The LBP is a very simple yet powerful texture descriptors and demonstrates the effectiveness in visual terrain classification [6]. Color feature is an important attribute for various terrains and a direct choice for many people to deal with the terrain classification problem. GIST is a typical global feature used in the fusion phase of our pipeline. For noise, we test Gaussian noise (mean: 0, variance: 0.01), speckle noise (mean: 0, variance: 0.04), and Poisson noise in this experiment. Illumination alteration ranges from −50% to 50% with 20% increments. The test results with SIFT on DS1 dataset are illustrated in Figures 16 and 17. The DSIFT descriptors obtained similar results.

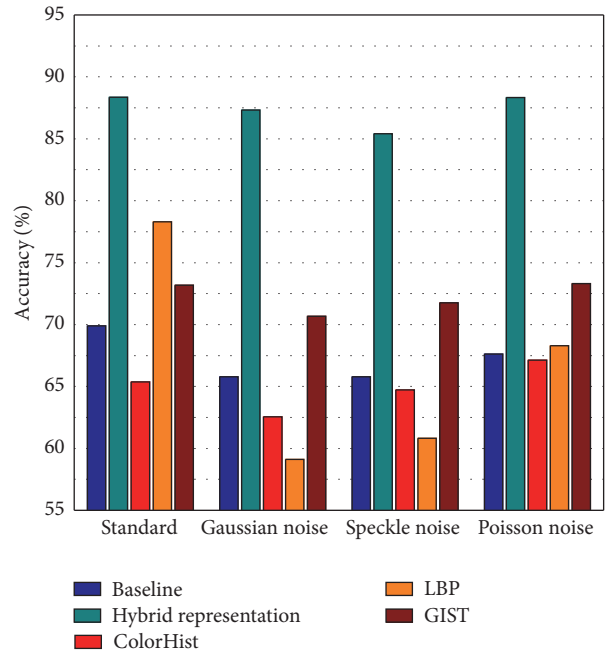

Figure 16: Comparison of performance with different methods of diverse noises, including Gaussian noise (mean: 0, variance: 0.01), speckle noise (mean: 0, variance: 0.04), and Poisson noise.

Several observations can be observed from these results:

(1) Our hybrid representation improves the performance significantly with a certain margin compared to current methods under various conditions, whether it is a standard environment, diverse noises, or illumination alteration. It demonstrates the effectiveness of our proposed optimum pipeline and hybrid representation.

(2) Our hybrid representation is highly robust, and its classification accuracy changes minimally for different types of noises and illumination conditions. By contrast, the texture feature LBP is sensitive to various noises. Its classification accuracy deceases 19.1 from 78.2 (standard) to 59.1 (Gaussian noise), a relative reduction of 24.5%. Under speckle noise and Poisson

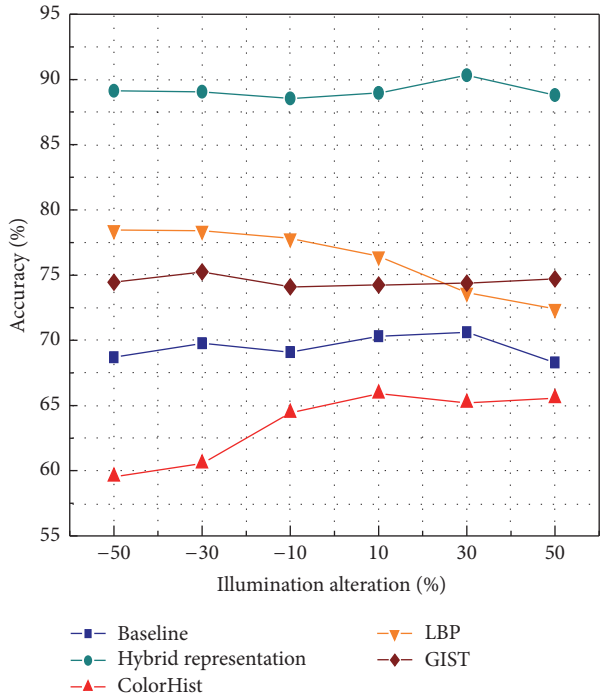

Figure 17: Comparison of performance with different methods of illumination alteration ranging from −50% to 50% with 20% increments.

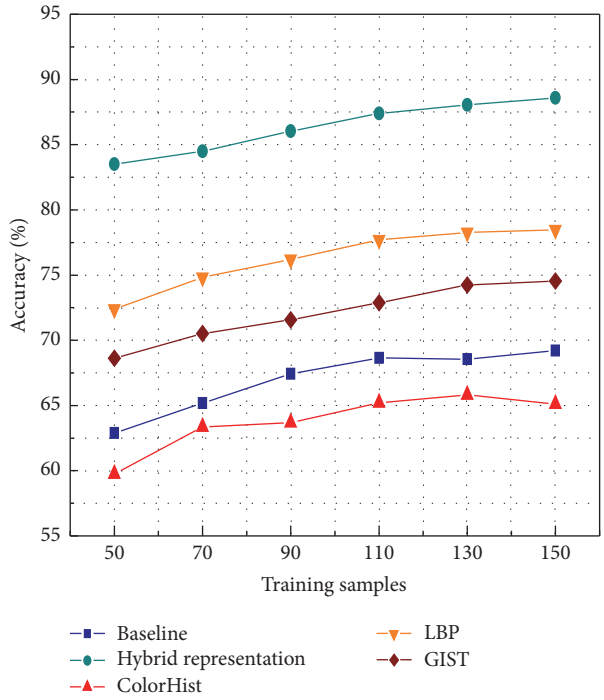

Figure 18: Comparison of efficiency between the hybrid representation and baseline method with different training samples ranging from 50 to 150 with increments of 20.

noise, the classification accuracies of LBP are just 60.8 and 68.3, respectively. Under strong or weak illumination, hybrid representation also provides stable high classification accuracy. Compared with hybrid representation, LBP is sensitive to strong illumination (deceasing to 72) and ColorHist is sensitive to weak illumination (deceasing to 59).

(3) The GIST is also robust to different environmental conditions, obtaining an eligible and stable performance under various noises and illumination alterations. It is an important factor for us to choose GIST in the fusion phase of our pipeline. Meanwhile, it can be found that the baseline method is also insensitive to noises and illumination alteration due to the same BOVW framework with hybrid representation.

In summary, the BOVW framework can make the final representation more robust, enhancing its ability to work in harsh environments. The proposed hybrid representation can provide stable and good performance for terrain classification under different noise and illumination conditions.

5.3. Sensitivity to Training Sample. In the previous section, our experiment used 150 training images and 150 test images per class on DS1 dataset. In this section, we test the performance of five classification methods, that is, the baseline method, ColorHist, LBP, GIST, and hybrid representation, with different sizes of the training set. The size of test set is still unchanged. The results with SIFT descriptors are illustrated in Figure 18. The DSIFT descriptors obtained similar results.

In Figure 18, it can be found the hybrid representation greatly outperforms the other methods with all different sizes of the training set. Although the accuracy decreases, the speed of hybrid representation is also slightly less than that of other methods with fewer training samples.

6. Conclusions

In this paper, we propose an optimum pipeline and uncover selected good practices to produce an effective and efficient visual terrain classification system. First, we provide a comprehensive study of all the steps in the BOVW framework and different fusion methods for visual terrain classification. Second, we explore multiple approaches in each step and study the effects of different methods and parameters, finding that every step is crucial and one improper choice greatly weakens the final performance. Finally, the feature preprocessing technique, improved BOVW framework, and fusion method are used to construct an optimum pipeline for visual terrain classification. The PCA-whitening technique is used for feature preprocessing. The improved BOVW framework includes the FV coding method and power-L2-normalization. The midlevel features and global features are combined by the average kernel fusion method. The hybrid representation generated by the optimum pipeline performs effectively and rapidly on visual terrain classification dataset under various conditions.

Conflicts of Interest

The authors declare no conflicts of interest.

Authors' Contributions

Hang Wu and Baozhen Liu contributed equally to this work and should be considered co-first authors.

Acknowledgments

The authors thank Professor Weihai Chen and Dr. Xiaojiang Peng for valuable discussion. This work is supported by the Science and Technology Pillar Program, Tianjin, China, under Project 16YFZCSF00590.

References

[1] P. Papadakis, "Terrain traversability analysis methods for unmanned ground vehicles: a survey," *Engineering Applications of Artificial Intelligence*, vol. 26, no. 4, pp. 1373–1385, 2013.

[2] B. H. Wilcox, "Non-geometric hazard detection for a mars microrover," in *Proceedings of the NASA Conference Publication*, pp. 675–675, NASA, 1994.

[3] M. Bajracharya, A. Howard, L. H. Matthies, B. Tang, and M. Turmon, "Autonomous off-road navigation with end-to-end learning for the LAGR program," *Journal of Field Robotics*, vol. 26, no. 1, pp. 3–25, 2009.

[4] F. L. Garcia Bermudez, R. C. Julian, D. W. Haldane, P. Abbeel, and R. S. Fearing, "Performance analysis and terrain classification for a legged robot over rough terrain," in *Proceedings of the 25th IEEE/RSJ International Conference on Robotics and Intelligent Systems (IROS '12)*, pp. 513–519, October 2012.

[5] A. Angelova, L. Matthies, D. Helmick, and P. Perona, "Fast terrain classification using variable-length representation for autonomous navigation," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '07)*, 8, 1 pages, June 2007.

[6] Y. N. Khan, *Visual Terrain Classification for Outdoor Mobile Robots*, Universität Tübingen, 2013.

[7] A. Howard, H. Seraji, and E. Tunstel, "A rule-based Fuzzy Traversability Index for mobile robot navigation," in *Proceedings of the IEEE International Conference on Robotics and Automation*, vol. 4, pp. 3067–3071, Seoul, South Korea, May 2001.

[8] A. Howard and H. Seraji, "Vision-based terrain characterization and traversability assessment," *Journal of Robotic Systems*, vol. 18, no. 10, pp. 577–587, 2001.

[9] Y. Guo, A. Song, J. Bao, and H. Zhang, "Optimal path planning in field based on traversability prediction for mobile robot," in *Proceedings of the International Conference on Electric Information and Control Engineering (ICEICE '11)*, pp. 563–566, IEEE, Wuhan, China, April 2011.

[10] D. Kim, S. M. Oh, and J. M. Rehg, "Traversability classification for UGV navigation: a comparison of patch and superpixel representations," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '07)*, pp. 3166–3173, November 2007.

[11] Y. N. Khan, P. Komma, and A. Zell, "High resolution visual terrain classification for outdoor robots," in *Proceedings of the IEEE International Conference on Computer Vision Workshops (ICCV Workshops '11)*, pp. 1014–1021, November 2011.

[12] J.-S. Lee, M. R. Grunes, E. Pottier, and L. Ferro-Famil, "Unsupervised terrain classification preserving polarimetric scattering characteristics," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 42, no. 4, pp. 722–731, 2004.

[13] K. Walas, "Terrain classification and negotiation with a walking robot," *Journal of Intelligent & Robotic Systems*, vol. 78, no. 3-4, pp. 401–423, 2015.

[14] N. Anantrasirichai, J. Burn, and D. Bull, "Terrain classification from body-mounted cameras during human locomotion," *IEEE transactions on cybernetics*, vol. 45, no. 10, pp. 2249–2260, 2015.

[15] I. Gheorghe, W. Li, T. Popham, A. Gaszczak, and K. J. Burnham, "Key learning features as means for terrain classification," in *Advances in Systems Science*, pp. 273–282, Springer, 2014.

[16] S. Zenker, E. E. Aksoy, D. Goldschmidt, F. Worgotter, and P. Manoonpong, "Visual terrain classification for selecting energy efficient gaits of a hexapod robot," in *Proceedings of the IEEE/ASME International Conference on Advanced Intelligent Mechatronics: Mechatronics for Human Wellbeing (AIM '13)*, pp. 577–584, IEEE, Wollongong, Australia, July 2013.

[17] P. Filitchkin and K. Byl, "Feature-based terrain classification for LittleDog," in *Proceedings of the 25th IEEE/RSJ International Conference on Robotics and Intelligent Systems (IROS '12)*, pp. 1387–1392, October 2012.

[18] Y. Zou, W. Chen, L. Xie, and X. Wu, "Comparison of different approaches to visual terrain classification for outdoor mobile robots," *Pattern Recognition Letters*, vol. 38, no. 1, pp. 54–62, 2014.

[19] Y. Yang and S. Newsam, "Bag-of-visual-words and spatial extensions for land-use classification," in *Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems (GIS '10)*, pp. 270–279, ACM, San Jose, Calif, USA, November 2010.

[20] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proceedings of the 26th Annual Conference on Neural Information Processing Systems (NIPS '12)*, pp. 1097–1105, December 2012.

[21] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[22] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks," in *Proceedings of the 20th Annual Conference on Neural Information Processing Systems (NIPS '06)*, pp. 153–160, December 2006.

[23] H. Wu, B. Liu, W. Su, W. Zhang, and J. Sun, "Deep filter banks for land-use scene classification," *IEEE Geoscience and Remote Sensing Letters*, vol. 13, pp. 1895–1899, 2016.

[24] P. Filitchkin, *Visual terrain classification for legged robots [M.S. thesis]*, University of California, Santa Barbara, Calif, USA, 2011.

[25] H. Wu, B. Liu, W. Su, W. Zhang, and J. Sun, "Hierarchical coding vectors for scene level land-use classification," *Remote Sensing*, vol. 8, no. 5, article 436, 2016.

[26] K. Chatfield, V. Lempitsky, A. Vedaldi, and A. Zisserman, "The devil is in the details: an evaluation of recent feature encoding methods," in *Proceedings of the 22nd British Machine Vision Conference (BMVC '11)*, September 2011.

[27] P. Gehler and S. Nowozin, "On feature combination for multiclass object classification," in *Proceedings of the IEEE 12th International Conference on Computer Vision*, pp. 221–228, IEEE, September-October 2009.

[28] A. Vedaldi, V. Gulshan, M. Varma, and A. Zisserman, "Multiple Kernels for object detection," in *Proceedings of the 12th International Conference on Computer Vision (ICCV '09)*, pp. 606–613, IEEE, Kyoto, Japan, October 2009.

[29] H. Wu, B. Z. Liu, W. H. Su, W. C. Zhang, and J. G. Sun, "Bag of words for visual terrain classification: a comprehensive study,"

Journal of Image and Graphics, vol. 21, no. 10, pp. 1276–1288, 2016.

[30] Y. Huang, Z. Wu, L. Wang, and T. Tan, "Feature coding in image classification: a comprehensive study," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 3, pp. 493–506, 2014.

[31] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[32] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-up robust features (SURF)," *Computer Vision and Image Understanding*, vol. 110, no. 3, pp. 346–359, 2008.

[33] R. Arandjelović and A. Zisserman, "Three things everyone should know to improve object retrieval," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '12)*, pp. 2911–2918, IEEE, Providence, RI, USA, June 2012.

[34] X. Peng, L. Wang, X. Wang, and Y. Qiao, "Bag of visual words and fusion methods for action recognition: Comprehensive Study and Good Practice," *Computer Vision and Image Understanding*, vol. 150, 2016.

[35] H. Abdi and L. J. Williams, "Principal component analysis," *Wiley Interdisciplinary Reviews: Computational Statistics*, vol. 2, no. 4, pp. 433–459, 2010.

[36] J. C. Van Gemert, C. J. Veenman, A. W. M. Smeulders, and J.-M. Geusebroek, "Visual word ambiguity," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 7, pp. 1271–1283, 2010.

[37] D. Arthur and S. Vassilvitskii, "K-means++: the advantages of careful seeding," in *Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA '07)*, pp. 1027–1035, SIAM, January 2007.

[38] G. McLachlan and D. Peel, *Finite Mixture Models*, John Wiley & Sons, New York, NY, USA, 2004.

[39] E. Hartuv and R. Shamir, "Clustering algorithm based on graph connectivity," *Information Processing Letters*, vol. 76, no. 4-6, pp. 175–181, 2000.

[40] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining (KDD '96)*, pp. 226–231, Portland, Ore, USA, August 1996.

[41] F. Murtagh, "Complexities of hierarchic clustering algorithms: state of the art," *Computational Statistics Quarterly*, vol. 1, pp. 101–113, 1984.

[42] L. Liu, L. Wang, and X. Liu, "In defense of soft-assignment coding," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV '11)*, pp. 2486–2493, IEEE, Barcelona, Spain, November 2011.

[43] J. Yang, K. Yu, Y. Gong, and T. Huang, "Linear spatial pyramid matching using sparse coding for image classification," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPR '09)*, pp. 1794–1801, June 2009.

[44] K. Yu, T. Zhang, and Y. Gong, "Nonlinear learning using local coordinate coding," in *Proceedings of the 22nd International Conference on Neural Information Processing Systems (NIPS '09)*, pp. 2223–2231, British Columbia, Canada, December 2009.

[45] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong, "Locality-constrained linear coding for image classification," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '10)*, pp. 3360–3367, June 2010.

[46] J. Sánchez, F. Perronnin, T. Mensink, and J. Verbeek, "Image classification with the fisher vector: theory and practice," *International Journal of Computer Vision*, vol. 105, no. 3, pp. 222–245, 2013.

[47] H. Jégou, F. Perronnin, M. Douze, J. Sánchez, P. Pérez, and C. Schmid, "Aggregating local image descriptors into compact codes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 9, pp. 1704–1716, 2012.

[48] K. Yu and T. Zhang, "Improved local coordinate coding using local tangents," in *Proceedings of the 27th International Conference on Machine Learning (ICML '10)*, pp. 1215–1222, IMLS, Haifa, Israel, June 2010.

[49] X. Zhou, K. Yu, T. Zhang, and T. S. Huang, "Image classification using super-vector coding of local image descriptors," in *Proceedings of the 11th European Conference on Computer Vision (ECCV '10)*, pp. 141–154, Springer, Crete, Greece, September 2010.

[50] P. Koniusz, F. Yan, and K. Mikolajczyk, "Comparison of mid-level feature coding approaches and pooling strategies in visual concept detection," *Computer Vision and Image Understanding*, vol. 117, no. 5, pp. 479–492, 2013.

[51] K. Simonyan, O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Fisher vector faces in the wild," in *Proceedings of the 24th British Machine Vision Conference (BMVC '13)*, September 2013.

[52] Y. Lin, F. Lv, S. Zhu et al., "Large-scale image classification: fast feature extraction and SVM training," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '11)*, pp. 1689–1696, June 2011.

[53] Y.-L. Boureau, J. Ponce, and Y. LeCun, "A theoretical analysis of feature pooling in visual recognition," in *Proceedings of the 27th International Conference on Machine Learning (ICML '10)*, pp. 111–118, Haifa, Israel, June 2010.

[54] Y.-L. Boureau, *Learning Hierarchical Feature Extractors for Image Recognition*, New York University, New York, NY, USA, 2014.

[55] N. Murray and F. Perronnin, "Generalized max pooling," in *Proceedings of the 27th IEEE Conference on Computer Vision and Pattern Recognition (CVPR '14)*, pp. 2473–2480, June 2014.

[56] G. Chowdhury, *Introduction to Modern Information Retrieval*, Facet Publishing, London, UK, 2010.

[57] F. Perronnin, J. Sánchez, and T. Mensink, "Improving the fisher kernel for large-scale image classification," in *Computer Vision—ECCV 2010*, pp. 143–156, Springer, Berlin, Germany, 2010.

[58] R. Arandjelovic and A. Zisserman, "All about VLAD," in *Proceedings of the 26th IEEE Conference on Computer Vision and Pattern Recognition (CVPR '13)*, pp. 1578–1585, June 2013.

[59] C. J. Lin, C.-W. Hsu, and C.-C. Chang, "A practical guide to support vector classification," National Taiwan University, 2003 http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf.

[60] R. Hadsell, A. Erkan, P. Sermanet, M. Scoffier, U. Muller, and Y. LeCun, "Deep belief net learning in a long-range vision system for autonomous off-road driving," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '08)*, pp. 628–633, September 2008.

[61] R. Hadsell, P. Sermanet, J. Ben et al., "Learning long-range vision for autonomous off-road driving," *Journal of Field Robotics*, vol. 26, no. 2, pp. 120–144, 2009.

[62] A. Oliva and A. Torralba, "Modeling the shape of the scene: a holistic representation of the spatial envelope," *International Journal of Computer Vision*, vol. 42, no. 3, pp. 145–175, 2001.

[63] R. Jain, *The Art of Computer Systems Performance Analysis*, John Wiley & Sons, 2008.

[64] S. Avila, N. Thome, M. Cord, and A. D. A. Araújo, "Pooling in image representation: the visual codeword point of view," *Computer Vision and Image Understanding*, vol. 117, no. 5, pp. 453–465, 2013.