sensors

MDPI

Article

Frequency-Temporal Disagreement Adaptation for Robotic Terrain Classification via Vibration in a Dynamic Environment

Chen Cheng [1,2], Ji Chang [1], Wenjun Lv [1,*], Yuping Wu [3], Kun Li [1,4], Zerui Li [1,5], Chenhui Yuan [5,6] and Saifei Ma [5,6]

1 Department of Automation, University of Science and Technology of China, Hefei 230027, China; 2014107@ahiib.edu.cn (C.C.); cjchange@mail.ustc.edu.cn (J.C.); zkdlk@mail.ustc.edu.cn (K.L.); lzerui@mail.ustc.edu.cn (Z.L.)
2 School of Information Engineering, Anhui Institute of International Business, Hefei 231131, China
3 Key Laboratory of Industrial Computer Control Engineering of Hebei Province, Yanshan University, Qinhuangdao 066004, China; ypwu@stumail.ysu.edu.cn
4 Department of Research and Development, Anhui Etown Information Technology Co., Ltd, Hefei 230011, China
5 Institute of Artificial Intelligence, Hefei Comprehensive National Science Center, Hefei 230088, China; yuanch@stu.ahu.edu.cn (C.Y.); sfma@stu.ahu.edu.cn (S.M.)
6 School of Computer Science and Technology, Anhui University, Hefei 230601, China
* Correspondence: wlv@ustc.edu.cn

Received: 10 October 2020; Accepted: 14 November 2020; Published: 16 November 2020

Abstract: The accurate terrain classification in real time is of great importance to an autonomous robot working in field, because the robot could avoid non-geometric hazards, adjust control scheme, or improve localization accuracy, with the aid of terrain classification. In this paper, we investigate the vibration-based terrain classification (VTC) in a dynamic environment, and propose a novel learning framework, named DyVTC, which tackles online-collected unlabeled data with concept drift. In the DyVTC framework, the exterior disagreement (ex-disagreement) and interior disagreement (in-disagreement) are proposed novely based on the feature diversity and intrinsic temporal correlation, respectively. Such a disagreement mechanism is utilized to design a pseudo-labeling algorithm, which shows its compelling advantages in extracting key samples and labeling; and consequently, the classification accuracy could be retrieved by incremental learning in a changing environment. Since two sets of features are extracted from frequency and time domain to generate disagreements, we also name the proposed method feature-temporal disagreement adaptation (FTDA). The real-world experiment shows that the proposed DyVTC could reach an accuracy of 89.5%, but the traditional time- and frequency-domain terrain classification methods could only reach 48.8% and 71.5%, respectively, in a dynamic environment.

Keywords: autonomous robot; non-geometric hazards; terrain classification; dynamic environment; vibration

1. Introduction

Robotic terrain classification refers to the process of a mobile robot classifying the terrain, on which it is traversing or will traverse, as one of the predefined classes [1]. An accurate terrain classification method is of great importance to an autonomous robot performing field tasks which usually need to traverse a variety of terrains like sand, grass, gravel, or clay [2,3]. For example, if a wheeled robot decides to traverse the sandy ground, its wheels may sink into the sand; and therefore, the robot could only move at an extreme low speed or even be trapped. To prevent the robots from suffering from such non-geometric hazards, the mobile robots must have the ability of terrain classification [4]. For another example, the robotic pose estimation, calculated by the kinematics model, which includes the slip parameters decided by the traversing terrains, usually benefits a lot from robotic terrain classification, especially in the situation without reliable global positioning systems [5–8]. Apart from hazards avoidance and pose estimation, many existing works have demonstrated that the performances of many other robotic fundamental functions, such as energy savings, route planning, gait control, etc., can be improved significantly from an accurate terrain classification method [9–13]. Therefore, terrain classification and its relevant research have received great attention from the DARPA Grand Challenge and Mars Exploration Plan [14].

As a non-interactive approach, the visual terrain classification method can recognize not only the traversing terrain, but also the terrains traversed or that will be traversed [15–17]. However, it suffers from two issues: (i) vision cannot work in extreme illumination (glare or dark); (ii) vision may be confused by the covering materials, thus it cannot recognize the real terrain [18–20]. Therefore, the interactive terrain classification, which are often implemented by means of acoustics [21,22], haptics [23,24], or vibration [25,26], is becoming more and more promising in robotic environment perception. The acoustic terrain classification has not been intensively studied, because its robustness against environmental noises cannot be guaranteed [27]. The haptic terrain classification is usually realized by means of tactile sensor arrays mounted on the robot–terrain contact area, thus it is more suitable for legged robots [28]. More than sound and contact force, the robot–terrain interaction generated vibration provides sufficient information to discriminate different types of terrains [29]. The time series collected by an accelerometer is the mixture of vibration and gravitational acceleration. As a result that gravity is almost time-invariant, the vibration can be easily recovered from the accelerometer readings; and therefore, the vibration-based terrain classification (VTC) method has incomparable advantages over the acoustic one. Additionally, unlike the haptic one, the VTC can be applied to both wheeled and legged robots. Hence, this paper concentrates on the vibration-based terrain classification.

Although a large body of terrain classification methods based on VTC have been investigated, most of them are achieved by supervised learning without considering the unlabeled upcoming vibration data [25,30–34]. In fact, we cannot guarantee a sufficient sampling of training dataset, so it is nature to resort to the semi-supervised or unsupervised machine learning tools for VTC. This idea was first proposed for safely operation of planetary exploration rovers [35]. In their work, co- and self-training approaches are employed, and two modalities, vibration and vision, are used to constitute two independent views, thus enabling the vibration- and vision-based classifiers to learn and develop mutually. Meanwhile, the vision-based classifier learns by itself when visions are collected on the different patches of vibration. More work that concerns the semi-supervised or unsupervised learning applying to the field of terrain classification can be found in [36–38]. These methods could be used in static environments, where the offline training dataset and the online testing dataset are independent and identically distributed (iid). However, if the training dataset is obtained from a certain area of grassland and the testing dataset from another, it is highly possible that the two datasets are non-iid since the two areas differ in moisture, roughness, or some other aspects. Hence, the dynamic environment could give rise to a degradation in predicting incoming vibration samples by using the classifier learned from the offline training dataset [39].

In this paper, we propose a vibration-based terrain classification framework for autonomous robots working in a dynamic environment (named DyVTC), mainly to suppress the affect rendered by data drift, during the period that manual labels do not arrive. First, according to different feature extraction methods, we construct the time- and feature-domain classifiers from the vibration view. Second, considering the potential temporal correlation in the traversed terrain patches, we introduce the Bayesian filter to correct the terrain predictions output by the two classifiers. Third, in terms of the classifier- and filter-output terrain predictions of the two domains, we propose a novel disagreement-based learning algorithm, which can be read as the most valuable contribution of our work. In the learning algorithm, the concept of ex- and in-disagreement are introduced, which is verified to be powerful to extract key samples and label them in high accuracy.

The rest of the paper is organized as follows. Section 2 covers the framework description of the proposed terrain classification method, as well as the details of some key steps, including feature extraction, classification algorithm, Bayesian filter, domain fusion, and pseudo-labeling algorithm. Section 3 presents the experimental verification, including the description of the experimental robot and data collection, performance evaluation of classifier and Bayesian filter, and comparative study between the existing methods and ours. The paper is concluded in Section 4.

2. Methodology

The framework of the proposed DyVTC is shown in Figure 1. A single vibration point provides an extreme limited information, so we should use a vibration frame, which is composed of a certain number of successive vibration points, to extract its representative features. All vibration frames are transformed into samples both in the time and frequency domain. Based on the labeled time- and frequency-domain vibration samples, two classifiers are obtained by batch training, respectively. The above process is offline. When the mobile robot is operating outdoors, online-collected vibration samples are fed into the pre-trained classifiers; and then, the classifier-output terrain predictions are fed into Bayesian filter to yield a better terrain prediction. Meanwhile, the classifier- and filter-output terrain predictions are analyzed based on the mechanism of ex- and in-disagreement; and therefore, some key samples could be extracted and labeled in high accuracy. When these pseudo-labeled samples accumulate to some extent, they are used to re-train the classifiers incrementally. The rest of the section expatiates on some key steps in the DyVTC.

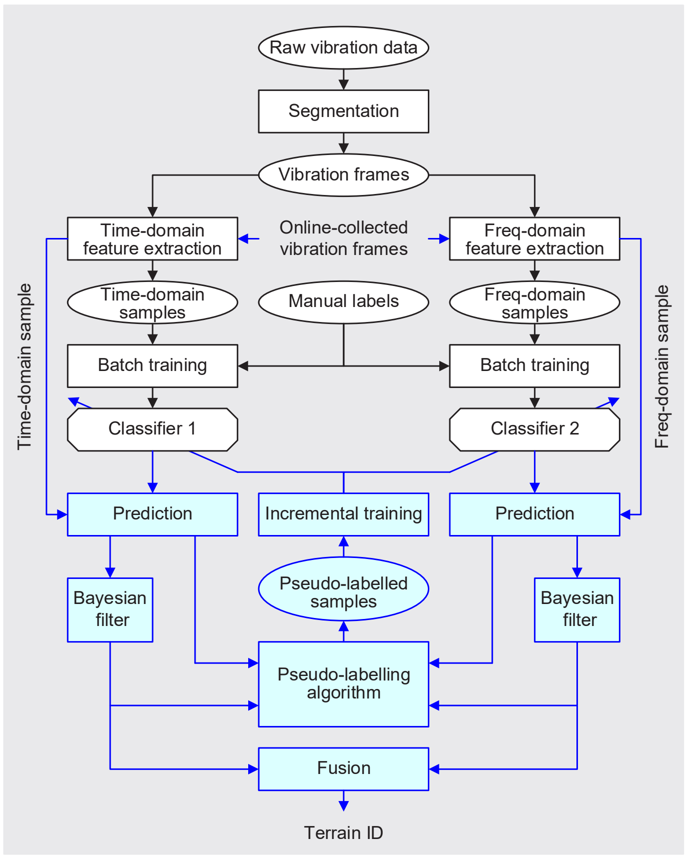


Figure 1. Framework of the proposed dynamic vibration-based terrain classification (DyVTC). The rectangular and elliptical blocks represent *operations* and *dataset*, respectively. The rectangles without corners represent *models*. We use the color blue to highlight the online parts.

2.1. Feature Extraction

We use an accelerometer to detect the acceleration along the vertical axis at 100 Hz, thus obtaining the acceleration time series. Due to the presence of gravity, the accelerometer does not detect a pure motion vibration, but the vertical acceleration mixed with gravitational acceleration. Hence, we subtract the gravitational acceleration constant from the acceleration time series, and therefore obtain the vibration time series. Furthermore, the vibration time series is split into vibration frames, each of which contains n vibration points. To guarantee a real-time terrain classification, each vibration frame overlaps the successive one by 50%. Define a vibration frame by $a = (a_1, a_2, \cdots, a_n)$. Now we are in the position to extract features from a in the frequency domain and time domain.

2.1.1. Frequency-Domain Features

The expression of time series in the frequency domain is usually beneficial to simplify the mathematical analysis and understand the signal components. The discrete Fourier transform (DFT) is such a powerful tool to yield the amplitude spectrum of the time series, thus being intensively used in the analysis of time series. The N-point DFT on the vibration frame a is defined by [40]

$$A_k = \sum_{i=0}^{N-1} a_i \mathrm{e}^{-\mathrm{j}\frac{2\pi ki}{N}}, k = 0, 1, ..., N-1, \tag{1}$$

where $\mathrm{j}^2 = -1$, k is the frequency. The implementation of DFT often employs an efficient algorithm, which is well known as fast Fourier transform (FFT). For an N-point FFT, the parameter N is typically specified as a power of 2 or a value that can be factored into a product of small prime numbers. In the case $N > n$, the vibration frame a should be padded using zeros; that is, the terms from a_{n+1} to a_N are specified as zeros.

The accelerometer usually work at a frequency of up to 100 Hz. If the terrain classification is desired to work at 1 Hz, which means the prediction should be given every second, then we use the 128-point FFT to transform the vibration frames into their spectrums. If treating the spectrum as the feature directly, the feature is a 128-dimensional vector. In order to reduce the feature dimension, we sample some entries uniformly from the spectral vector to constitute the feature.

2.1.2. Time-Domain Features

Other than the frequency domain, we also extract the features in the time domain directly. A 10-dimensional feature vector $\phi = (\phi_1, \phi_2, \cdots, \phi_{10})$ is obtained, and its entries are shown in Table 1. It is noted that ϕ_5 can be extended by setting $\tau = 1, 2, \cdots, n-1$. However, according to the Khintchine's law, it should be guaranteed that $\tau \ll n$ to bound the estimation error of ϕ_5. In this paper, we choose $\tau = 1$.

2.2. Support Vector Machine

Let $\{(x_1, y_1), ..., (x_m, y_m)\}$ denote the training set, where m is the size of the training set and $y_i \in \{\pm 1\}$. Support vector machine (SVM) aims to construct a separating hyperplane between two classes of points that maximizes the margin between the hyperplane and support vectors [41]. Usually the hyperplane cannot be found in the original sample space. For such a nonlinear classification task, kernel technique is applied to map the original data to a high-dimensional feature space by $\varphi : x \to \varphi(x)$. Inner product of points in feature space is then conducted implicitly by a kernel function. In our work, we use two common kernel functions that are linear kernel $\kappa(x_i, x_j) = x_i' x_j$, and Gaussian kernel $\kappa(x_i, x_j) = \exp(-\frac{\|x_i - x_j\|^2}{2\sigma^2})$, where σ denotes the width of the Gaussian kernel. Soft margin is used to regularize the trade-off between minimizing the training error and maximizing the margin. Therefore, an SVM can be described as the following optimization problem [42]

$$\min_{\omega,b,\xi} \frac{1}{2}\|\omega\|^2 + \lambda \sum_{i=1}^{m} \xi_i \tag{2}$$

$$\text{s.t.}\ \ y_i\left(\omega'\varphi(x_i)+b\right) \geq 1-\xi_i \tag{3}$$

$$\xi_i \geq 0,\ i=1,2,...,m \tag{4}$$

where ω is the vector normal to the hyperplane, b is a scalar bias, and λ is the soft margin parameter. Multi-class classification task of SVM can be performed using one-versus-one approach. The SVM model can be updated online using incremental SVM (i.e., [43]). As a result that only the support vectors participate in the learning process, the incremental SVM reduces the training time greatly and seldom loses accuracy.

Table 1. Entries of the time-domain feature.

Name	**Equation**	**Description**
Zero-crossing number (ZCN)	$\phi_1 = \sum_{i=2}^{n} \mathbb{I}(a_i a_{i-1} < 0)$	$\mathbb{I}(\cdot)$ is an indicator function, which outputs 1 if the expression in $(\cdot)$ holds, or 0 otherwise. This feature is an approximation of the frequency of a.
Mean	$\phi_2 = \frac{1}{n}\sum_{i=1}^{n} a_i$	Although the gravitational acceleration has been subtracted, the mean of a may considerably diverge from zero for some course terrains.
ZCN in $\bar{a}$	$\phi_3 = \sum_{i=2}^{n} \mathbb{I}(\bar{a}_i \bar{a}_{i-1} < 0)$	$\bar{a}_i = a_i - \phi_2$. ϕ_3 is a complement to ϕ_1, which avoids $\phi_1 \approx 0$ for even high-frequency vibration signal when the robot is traversing coarse terrains.
Variance	$\phi_4 = \frac{1}{n}\sum_{i=1}^{n}(a_i - \phi_2)^2$	Intuitively, the variance is higher when the terrain becomes coarser.
Autocorrelation	$\phi_5 = \frac{1}{(n-\tau)\phi_4}\sum_{i=1}^{n-\tau}(a_i - \phi_2)(a_{i+\tau} - \phi_2)$	$\tau < n$ is an integer indicating time difference. As a measure of non-randomness, ϕ_5 gets larger with a stronger dependency between a_i and $a_{i+\tau}$.
Maximum	$\phi_6 = \max(a)$	ϕ_6 indicates the biggest bump of the terrain.
Minimum	$\phi_7 = \min(a)$	ϕ_7 indicates the deepest puddle of the terrain.
ℓ_2-norm	$\phi_8 = \sqrt{\sum_{i=1}^{n}(a_i)^2}$	ϕ_8 reflects the energy of a. If $\phi_2 \to 0$, ϕ_9 has the similar function as ϕ_4. Instead, we can also use the ℓ_1-norm, i.e., $\phi_8^* = \sqrt{\sum_{i=1}^{n}\lvert a_i\rvert}$.
Impulse factor	$\phi_9 = n(\phi_6 - \phi_7)/\phi_8^*$	ϕ_9 measures the impact degree in a.
Kurtosis	$\phi_{10} = \frac{1}{n}\sum_{i=1}^{n}(a_i - \phi_2)^4/\phi_4^2 - 3$	ϕ_{10} measures the deviation degree of the a with Gaussian distribution.

2.3. Bayesian Filter

The recursive form of Bayesian filter can be seen in [44]. Define χ_t as the state at time t, c_t the measurement, and $C_t = \{c_1, c_2 \cdots, c_t\}$ the measurement set. The purpose is to acquire $\mathbb{P}(\chi_t|C_t)$, the *a posteriori* possibility distribution function (pdf) of χ_t conditioned on C_t. Given $\mathbb{P}(\chi_{t-1}|C_{t-1})$, we have

$$\mathbb{P}(\chi_t|C_{t-1}) = \int \mathbb{P}(\chi_t|\chi_{t-1})\mathbb{P}(\chi_{t-1}|C_{t-1})\mathrm{d}\chi_{t-1}, \tag{5}$$

$$\mathbb{P}(\chi_t|C_t) = \frac{\mathbb{P}(c_t|\chi_t)\mathbb{P}(\chi_t|C_{t-1})}{\int \mathbb{P}(c_t|\chi_t)\mathbb{P}(\chi_t|C_{t-1})\mathrm{d}\chi_t}, \tag{6}$$

where $\mathbb{P}(\chi_t|C_{t-1})$ denotes the *a priori* pdf of χ_t conditioned on C_{t-1}.

Define χ_t as the state at time t, c_t the measurement, and $C_t = \{c_1, c_2 \cdots, c_t\}$ the measurement set. The purpose is to acquire $\mathbb{P}(\chi_t|C_t)$, the *a posteriori* possibility distribution function (pdf) of χ_t conditioned on C_t. Generally speaking, analytic solutions to Equations (5) and (6) are unavailable in most cases, so the estimation problem for continuous state is seldom tackled by Bayesian filter. However, if the state is discrete and its number is not too large, the Bayesian filter is a practicable method to solve such a state estimation problem. In terrain classification, the state at time t is defined as $\chi_t \in \{1, 2, \cdots, \ell\}$ where $i = 1, 2, \cdots, \ell$ denotes the terrain ID. The measurement $c_t \in \{1, 2, \cdots, \ell\}$ is the classifier-output terrain predictions. Given $\mathbb{P}(\chi_{t-1}|C_{t-1})$, we have

$$\mathbb{P}(\chi_t = i|C_{t-1}) = \sum_{j=1}^{\ell} \mathbb{P}(\chi_t = i|\chi_{t-1} = j)\mathbb{P}(\chi_{t-1} = j|C_{t-1}), \tag{7}$$

$$\mathbb{P}(\chi_t = i|C_t) = \frac{\mathbb{P}(c_t = j|\chi_t = i)\mathbb{P}(\chi_t = i|C_{t-1})}{\sum_{i=1}^{\ell} \mathbb{P}(c_t = j|\chi_t = i)\mathbb{P}(\chi_t = i|C_{t-1})}, \tag{8}$$

where $\mathbb{P}(\chi_t|C_{t-1})$ denotes the *a priori* pdf of χ_t conditioned on C_{t-1}, $\mathbb{P}(\chi_t = i|\chi_{t-1} = j)$ denotes the probability that the mobile robot moves from terrain j to i at time t, and $\mathbb{P}(c_t = j|\chi_t = i)$ denotes the probability of the classifier outputting terrain j conditioned on terrain i. Meanwhile, we observe that the denominator of Equation (8) is a normalizer.

Applying the Bayesian filter to improve the terrain classification is on the premise of knowing $\mathbb{P}(\chi_0|C_0)$, $\mathbb{P}(c_t|\chi_t)$ and $\mathbb{P}(\chi_t|\chi_{t-1})$. First, the initial *a posteriori* pdf $\mathbb{P}(\chi_0|C_0)$, where C_0 denotes a set of no measurements, describes the distribution of the terrain at which the mobile robot locates initially. If the initial terrain is known, then we have $\mathbb{P}(\chi_0 = i|C_0) = 1$ and $\mathbb{P}(\chi_0 \neq i|C_0) = 0$ when locating at terrain i; otherwise, $\mathbb{P}(\chi_0|C_0)$ is assumed to be uniform distribution, namely, $\mathbb{P}(\chi_0 = i|C_0) = \frac{1}{\ell}$ for $i = 1, 2, \cdots, \ell$. Second, $\mathbb{P}(c_t|\chi_t)$, which is required during the measurement-update procedure, is determined by the confusion matrix. Third, $\mathbb{P}(\chi_t|\chi_{t-1})$, which is required during the time-update procedure, describes the correlation of the sampled terrain series. Given ℓ terrains, an $\ell \times \ell$ square matrix M with elements $m_{ij} = \mathbb{P}(\chi_t = i|\chi_{t-1} = j)$ is defined. The diagonal elements m_{ii} where $i = 1, 2, \cdots, \ell$ should be assigned a relatively large value not greater than 1, based on the heuristic that terrain is spatially continuous. The off-diagonal elements m_{ij} where $i \neq j$ can be determined by the terrain distribution in a map. For example, if terrain i possesses more area than terrain j, then $m_{ij} < m_{ji}$. It should be guaranteed that the sum of a row equals 1. A general and simple setup of M is that $m_{ii} = \mu$ for $i = 1, 2, \cdots, \ell$ and $m_{ij} = \frac{1-\mu}{\ell-1}$ for $i \neq j$.

2.4. Pseudo-Labeling Algorithm

The pseudo-labeling algorithm aims to extract key samples, and label them in a high accuracy. The term *key samples* is denoted as the unlabeled samples that cannot be correctly classified. Now we introduce a new term named *interior disagreement* (*in-disagreement*). For each domain, we have two terrain predictions at the same time. The classifier outputs are read as the *a priori* terrain predictions, while the filter outputs as the *a posteriori* terrain predictions. If the *a priori* and *a posteriori* terrain predictions of the same domain at a certain time are different, then this phenomenon is referred to as *in-disagreement*. The term *a priori ex-disagreement* means the *a priori* terrain predictions at a certain time of the two domains are different. Similarly to the *a priori ex-disagreement*, the *a posteriori ex-disagreement* is denoted by that the *a posteriori* terrain predictions at a certain time of the two domains are different. Based on the in- and ex-disagreement, we propose the following heuristic rules:

1. If one domain (denoted as the 1st domain) appears in-disagreement at a certain time, the sample is likely to be a key sample of the 1st domain.
2. Based on the first rule, if at the same time, the other domain (denoted as the 2nd domain) does not appear in-disagreement, and there is no *a posteriori* ex-disagreement between the two domains, then the 2nd-domain terrain prediction is likely to be a reliable label to the 1st-domain key sample.
3. If in-disagreement appears in both domains, but there is no *a posteriori* ex-disagreement, the filter-output terrain prediction can be used to label the samples from both domains.
4. If neither in-disagreement nor ex-disagreement appears at a certain time, the sample is likely to be classified correctly, thus not a key sample.

Now we present the algorithm in detail. Define $\gamma \in \{T, F\}$ as the domain type, where T stands for time domain, and F for frequency domain. In the γ domain, upon feeding a sample x_t^γ, the γ-domain classifier outputs the *a priori* terrain prediction c_t^γ; and then, the Bayesian filter outputs the *a posteriori* terrain prediction $\hat{c}_t^\gamma$. The pseudo-labeling algorithm is shown in Algorithm 1. As a result that the rules are proposed on the mechanism of in- and ex-disagreement, we name it in- and ex-disagreement-based pseudo-labeling (IE). The proposed IE is a sample that is an efficient method to extract and label key samples, which will be verified in Section 3.

Algorithm 1 In- and Ex-Disagreement-Based Pseudo-Labeling Algorithm (IE)

Input: The unlabeled samples x_t^T and x_t^F, the *a priori* terrain predictions c_t^T and c_t^F, the *a posteriori* terrain predictions $\hat{c}_t^T$ and $\hat{c}_t^F$, where $t = 1, 2, \cdots, K$.

Output: Pseudo-labeled sample sets L_T and L_F, for time and frequency domain, respectively.

```
1:  set $L_T, L_F \leftarrow \emptyset$
2:  for $t = 1$ to $K$ do
3:      if $c_t^T = \hat{c}_t^T$ and $c_t^F \neq \hat{c}_t^F$ and $\hat{c}_t^T = \hat{c}_t^F$ then
4:          $L_F \leftarrow L_F \cup (x_t^F, \hat{c}_t^T)$
5:      end if
6:      if $c_t^F = \hat{c}_t^F$ and $c_t^T \neq \hat{c}_t^T$ and $\hat{c}_t^T = \hat{c}_t^F$ then
7:          $L_T \leftarrow L_T \cup (x_t^T, \hat{c}_t^F)$
8:      end if
9:      if $c_t^T \neq \hat{c}_t^T$ and $c_t^F \neq \hat{c}_t^F$ and $\hat{c}_t^T = \hat{c}_t^F$ then
10:         $L_F \leftarrow L_F \cup (x_t^F, \hat{c}_t^T)$, $L_T \leftarrow L_T \cup (x_t^T, \hat{c}_t^F)$
11:     end if
12: end for
13: return $L_T$ and $L_F$
```

2.5. Fusion of Terrain Predictions

In ensemble learning, voting, including the majority, plurality, and weighted voting, are general schemes to fuse different predictions [45]. However, they cannot be used to our fusion task directly, since we only have two domains. Two dedicated schemes follow:

The 1st fusion scheme is

$$o_t^1 = \begin{cases} \hat{c}_t^T, & \text{if } \hat{p}_t^T > w\hat{p}_t^F, \\ \hat{c}_t^F, & \text{if } \hat{p}_t^T \leq w\hat{p}_t^F, \end{cases} \tag{9}$$

where o_t^1 denotes the fused terrain prediction using the 1st fusion scheme, $\hat{p}_t^\gamma$ denotes the confidence of $\hat{c}_t^\gamma$. The weight $w > 0$ assigns the two *a posteriori* terrain predictions different weights, which are often set as a number larger than 1 because the frequency domain usually outperforms the time domain.

The 2nd fusion scheme is

$$o_t^2 = \mathcal{M}\left\{\frac{\hat{v}_t^T + w\hat{v}_t^F}{1+w}\right\}, \tag{10}$$

where o_t^2 denotes the fused terrain prediction using the 2nd fusion scheme, $\hat{v}_t^\gamma$ denotes the confidence vector of the γ-domain Bayesian filtering at time t. The weight $w > 0$ should be a number larger than 1. The function $\mathcal{M}\{\cdot\}$ returns the index of the largest element in the vector. As a result that the terrain IDs correspond to the vector indices, $\mathcal{M}\{\cdot\}$ returns the terrain prediction.

The mentioned fusion schemes fuse the *a posteriori* terrain predictions, while they can be also used to fuse the *a priori* terrain predictions.

3. Experimental Verification

In this section, we first present the description of the experimental robot, the experimental terrains, and the details of the experimental data collection. Second, we demonstrate the performance of the traditional terrain classification methods when data drift exists. Thirdly, we exhibit how the Bayesian filtering improves the classification results. Finally, a comparative study is done to verify the effectiveness of the proposed DyVTC.

3.1. Experimental Data Collection

The experimental robot and its electronic system structure and signal flows are shown in Figure 2. The robot is 340 mm in length, 270 mm in width, 230 mm in height, and 2.6 kg in mass. The diameter and width of the wheels are 130 mm and 60 mm, respectively. With a power supply of 12 V, the robot could traverse coarse grounds at the speed of up to 1.5 m/s. An accelerometer–gyroscope–magnetometer integrated sensor (MPU9250) and an odometry constitute the sensor system. The main configurations of odometry, gyroscope, accelerometer, and magnetometer are exhibited in Table 2. The odometry is actually four incremental encoders which are directly mounted on the motor shafts to perceive the motor rotational speeds; and consequently, the odometry outputs the robot's moving speed. The accelerometer–gyroscope–magnetometer integrated sensor can be used to obtain the robot pose and the vibration. The micro control unit (MCU) reads the Z-axis accelerometer at 100 Hz. Meanwhile, the robot moving speed and pose are measured every second, which evaluates the robot motion modes. The robot is controlled with a smart phone, by sending commands to the robot via Bluetooth. The MCU is a development board of Arduino Mini Pro which is used to realize some simple and fundamental operations, such as data gathering, motor control, and command receiving. While the robot is working, all data are stored in the local memory (a T-Flash card); and next, the card is unplugged from the robot, connected, and transferred to a desktop computer (3.20 GHz, 8 GB RAM).

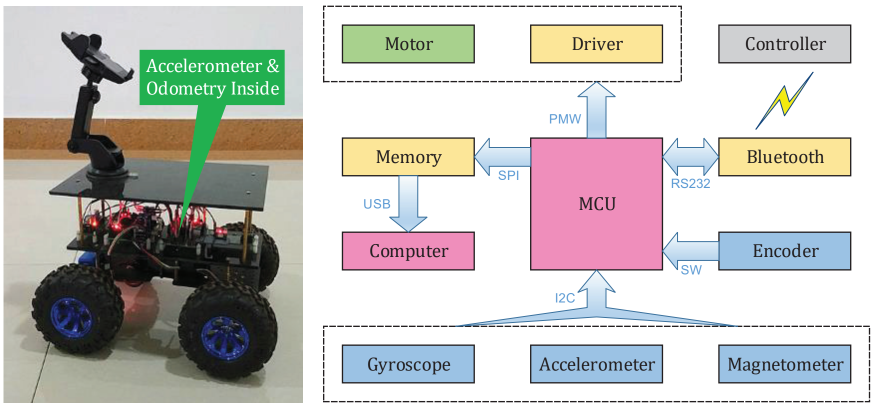


Figure 2. A four-wheeled mobile robot for experiment. The left figure shows the robot photograph, and the right one shows its electronic system structure and signal flows.

Table 2. Specifications of sensors.

Sensor	**Specifications**
Odometry	540 pulse per round; resolution: 0.67 deg.
Gyroscope	range: ±250 deg/s; initial ZRO * tolerance: ±5 deg/s; total RMS † noise: 0.1 deg/s.
Accelerometer	range: ±2 g; initial ZGO ‡ tolerance: ±80 mg; total RMS noise: 8 mg.
Magnetometer	range: ±4800 uT.

* zero-rate output; † root mean square; ‡ zero-gravity output.

All algorithms will be evaluated on the computer based on the gathered data. Among the terrains listed in [46], we select six terrains on which a robot is most likely to traverse to do the experiment. As shown in Figure 3, some of them are artificial terrains (e.g., asphalt road), while some are natural ones (e.g., natural grass). These terrains are different in rigidity, roughness, and flatness. The segments of vibration time series collected on the six terrains and the corresponding terrain photographs are shown in Figure 3. Compared with other terrains, it is observed that the interaction between the robot and the cobble path generates a highly distinguishable vibration. The vibration has higher frequency, larger magnitude, and weaker autocorrelation, because the cobble path is relatively rigid and irregular. The vibrations of the other five terrains may not be easy to discriminate intuitively because of their slight differences; however, they still can be found in terms of their variation tendency.

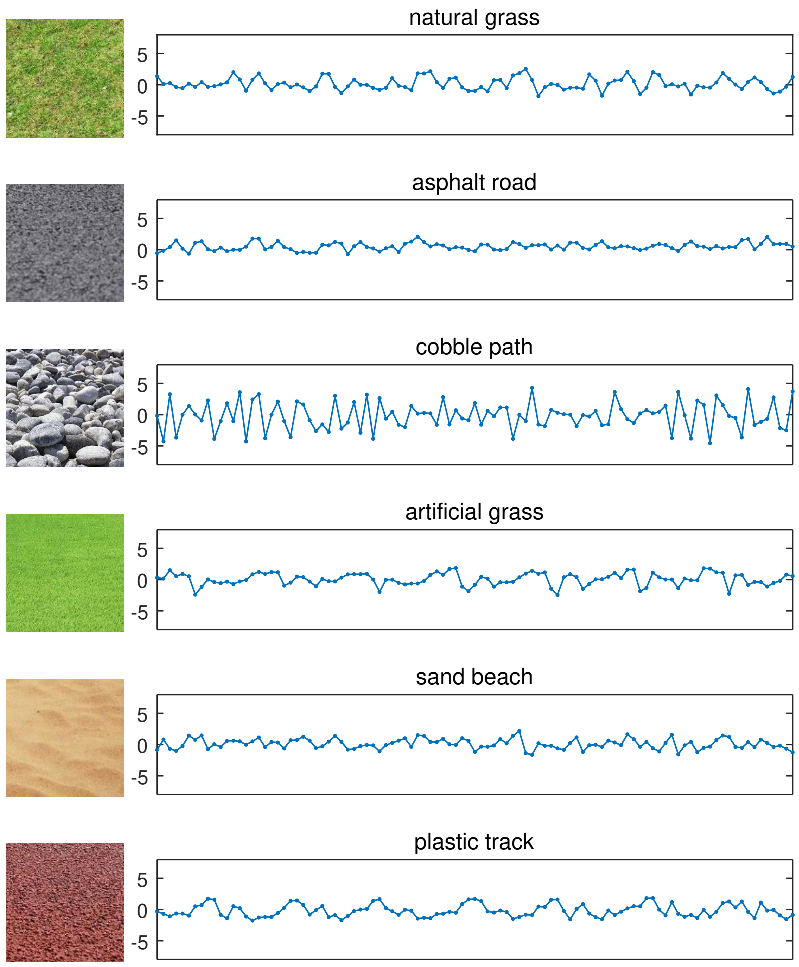
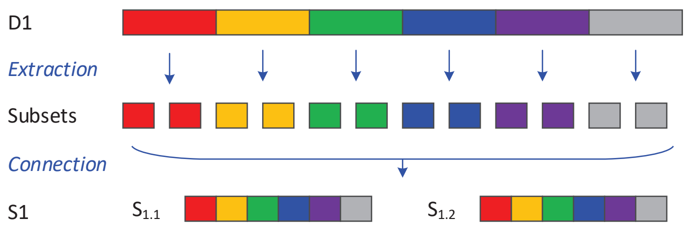
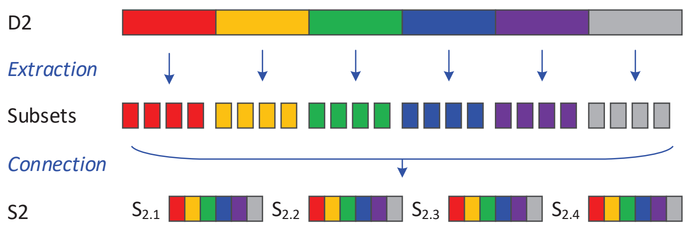


Figure 3. Photos of the traversed terrains and the corresponding segments of vibration time series. From top to bottom, the experimental terrains are: natural grass, asphalt road, cobble path, artificial grass, sand beach, plastic track, respectively. They are abbreviated as NG, AR, CP, AG, SB, and PT, respectively. The Y axis represents acceleration (m/s^2).

Different motion states might also cause the data drift, but it could be eliminated by the sufficient data collection, as the number of motion states are relatively limited. Hence, in our experiment of data collection, we control the experimental robot to wander on the six terrains at a speed ranging from the minimum speed (0.2 m/s) to the maximum speed (1.1 m/s) and in different motion modes (e.g., circular and linear motion), which avoids the data drift from an insufficient experiment. We collect the vibration data in two different environments, thus obtaining two vibration datasets: D_1 and D_2. lIntuitively speaking, (i) the grass in garden and roadside may be different in height, (ii) the natural grass gets harder under fine weather, while softer after raining, and (iii) the soil is harder at night than that in the daytime because of the lower temperature. Environment One and Environment Two both include the aforementioned 6 terrains, but are different in location, weather, and temperature. For each dataset, the vibration time series are segmented into vibration frames by every 100 points with 50% overlap, and therefore, D_1 and D_2 are transformed into S_1 and S_2 which are composed of vibration frames. As shown in Figure 4a, S_1 is divided into $S_{1.1}$ and $S_{1.2}$, each of which contains 3000 frames. Similarly, as shown in Figure 4b, S_2 is divided into $S_{2.1}$, $S_{2.2}$, $S_{2.3}$, and $S_{2.4}$, each of which contains 3000 frames. In addition, according to different feature extraction methods, $S_{1.1}$ is transformed into two sample sets, where $S^T_{1.1}$ and $S^F_{1.1}$ are derived by using time-domain features and frequency-domain features, respectively. Analogously, we have $(S^T_{1.2}, S^F_{1.2})$, $(S^T_{2.1}, S^F_{2.1})$, $(S^T_{2.2}, S^F_{2.2})$, $(S^T_{2.3}, S^F_{2.3})$, and $(S^T_{2.4}, S^F_{2.4})$.



(**a**) Constitution of $S_{1.1}$ and $S_{1.2}$.



(**b**) Constitution of $S_{2.1}$, $S_{2.2}$, $S_{2.3}$, and $S_{2.4}$.

Figure 4. Illustration of data constitution.

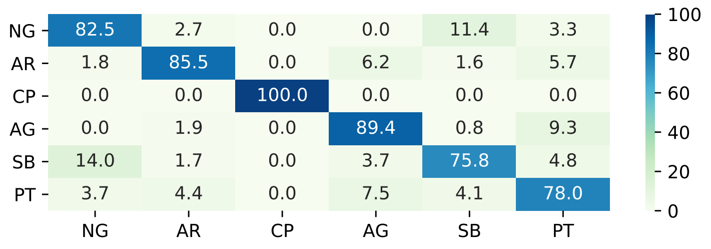
3.2. Performance Evaluation of Classifier

To evaluate the classifier performance in a static environment, i.e., the training and test data are both gathered in Environment One, we train two classifiers on $S^T_{1.1}$ and $S^F_{1.1}$, and test them on $S^T_{1.2}$ and $S^F_{1.2}$, respectively. The Gaussian kernel is employed for the time-domain classifier. As for the frequency-domain classifier, because the feature vector is of high dimension, we employ a linear kernel. We use the confusion matrix to show the classification performance. The rows of the confusion matrices represent the real terrains, while the columns represent the predicted terrains. The trained time- and frequency-domain SVM model can achieve the accuracies of 85.4% and 86.5%, which are acceptable to a field robot. It is observed that the main confusion exists between the terrains of natural gas (NG) and sand beach (SB). Compared with other terrain, NG and SB are both natural terrains, and have the similar rigidity and unevenness. In addition, the terrain of plastic track (PT) cannot be easily classified. The classifier $\mathbb{C}_T$ cannot distinguish PT and asphalt road (AR) perfectly, while $\mathbb{C}_F$ are confused in PT and artificial grass (AG). The terrains of PT, AR, and AG are all artificial terrains, which are made to enhance pedestrian or vehicle's traversability, so they usually have the similar characteristics in rigidity, roughness, and flatness.

To evaluate the classifier performance in a dynamic environment, we use the classifiers trained on $S^T_{1.1}$ and $S^F_{1.1}$ to predict $S^T_{2.1}$ and $S^F_{2.1}$, respectively. Due to the data drift, the accuracies on $S^T_{2.1}$ and $S^F_{2.1}$ could only reach 48.8% and 71.5%, respectively. As illustrated in Figure 5a,b, the data drift causes many confusions. In the time domain, only 0.6% of AG samples and 7.1% of PT samples could be classified correctly. Most AG samples are misclassified as NG, AR, and SB. In particular, 45.7% of the PT samples are misclassified as NG. Obviously, the SVM model trained on $S^T_{1.1}$ cannot distinguish AG from other terrains on S^T_2. In the frequency domain, as demonstrated in Figure 6a,b, the classifier performs much better under data drift, but only about 33% of NG and SB samples can be classified correctly. The performance degradation of SVM model is caused by data drift. In our experiment, NG is the most changeful terrain, hence becoming the main class that confuses the classifier.

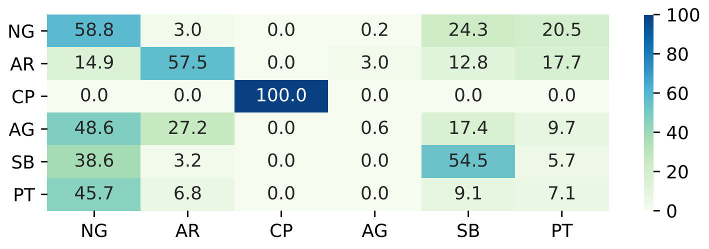
The fusion accuracies with different w are shown in Figure 7. It is observed that the fusion of the time- and frequency-domain classifiers could increase the classification accuracy slightly, with an appropriate w. The time-domain classifier performs much worse than the frequency-domain one, so the increase of fusion accuracy is not significant.

The offline terrain classification, which means the classifiers performing on $\mathcal{S}^T_1/\mathcal{S}^F_1$, could achieve a maximum accuracy of 92.7%. The offline classification accuracy is improved. However, the online terrain classification, which means the classifiers performing on $\mathcal{S}^T_2/\mathcal{S}^F_2$, does not see a significant improvement. The online classification accuracy can be increased by about only 1% if ω could be appropriately set. If we have no *a priori* knowledge on the two views and do not know which is better, then the coefficient ω is usually assigned by 1.



	NG	AR	CP	AG	SB	PT
NG	82.5	2.7	0.0	0.0	11.4	3.3
AR	1.8	85.5	0.0	6.2	1.6	5.7
CP	0.0	0.0	100.0	0.0	0.0	0.0
AG	0.0	1.9	0.0	89.4	0.8	9.3
SB	14.0	1.7	0.0	3.7	75.8	4.8
PT	3.7	4.4	0.0	7.5	4.1	78.0

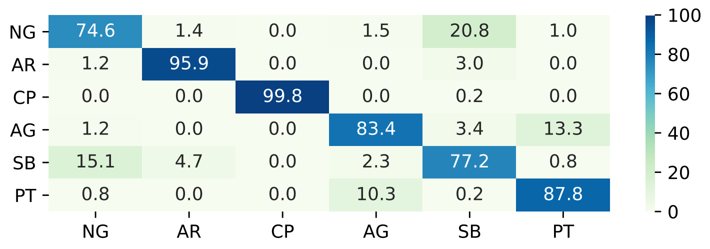
(**a**) Performance of time-domain classifier trained on $S^T_{1.1}$ testing on $S^T_{1.2}$. The accuracy is 85.4%.



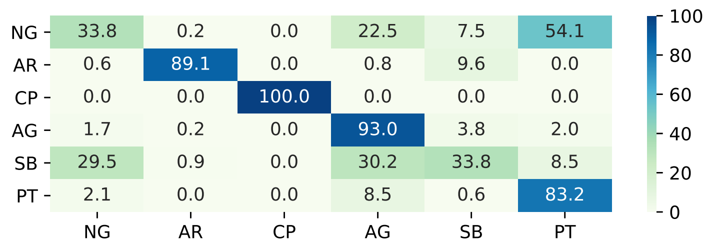
	NG	AR	CP	AG	SB	PT
NG	58.8	3.0	0.0	0.2	24.3	20.5
AR	14.9	57.5	0.0	3.0	12.8	17.7
CP	0.0	0.0	100.0	0.0	0.0	0.0
AG	48.6	27.2	0.0	0.6	17.4	9.7
SB	38.6	3.2	0.0	0.0	54.5	5.7
PT	45.7	6.8	0.0	0.0	9.1	7.1

(**b**) Performance of time-domain classifier trained on $S^T_{1.1}$ testing on $S^T_{2.1}$. The accuracy is 48.8%.

Figure 5. Normalized confusion matrices (in %) of SVM-based terrain classification on sample set S_2. NG, AR, CP, AG, SB, PT denote natural grass, asphalt road, cobble path, artificial grass, sand beach, plastic track, respectively.



(**a**) Performance of frequency-domain classifier trained on $S^F_{1.1}$ and tested on $S^F_{1.2}$. The accuracy is 86.5%.



(**b**) Performance of frequency-domain classifier trained on $S^F_{1.1}$ and tested on $S^F_{2.1}$. The accuracy is 71.5%.

Figure 6. Normalized confusion matrices (in %) of SVM-based terrain classification on a sample set S_2. NG, AR, CP, AG, SB, PT denote natural grass, asphalt road, cobble path, artificial grass, sand beach, plastic track, respectively.

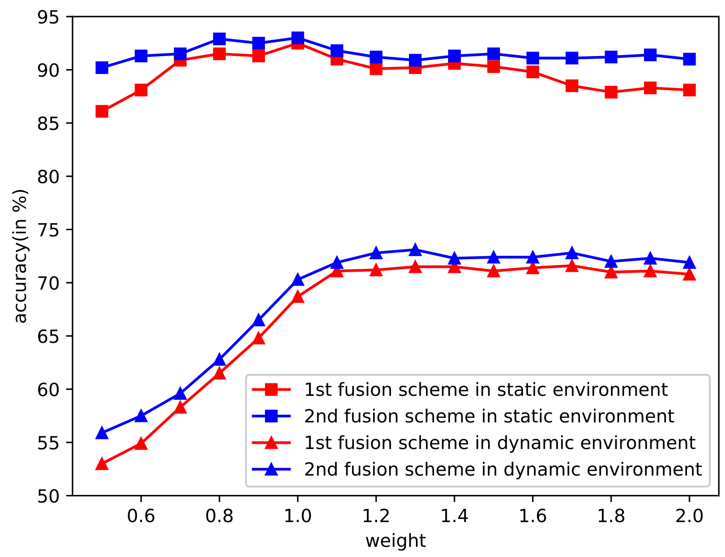
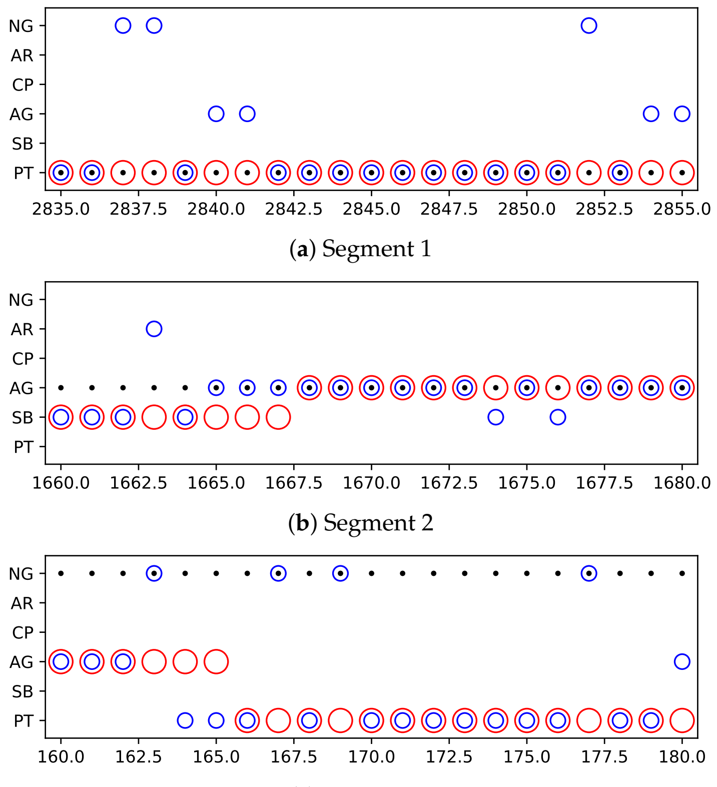


Figure 7. Fusion accuracies of the time- and frequency-domain classifiers with different weights.

3.3. Performance Evaluation of Bayesian Filter

Now we are in the position to evaluate the Bayesian filter improving the classifier-output terrain predictions. Here, we exhibit the details of the Bayesian filter correcting the classifier's outputs, as shown in Figure 8. Taking the temporal correlation in sample stream into consideration, the prediction of the current terrain is not only based on the current vibration frame any more, but a combination of the current vibration frame and the previous terrain prediction. Hence, as shown in Figure 8a,b, the incorrect predictions by the classifier at time 1674, 1676, 2837, 2838, 2840, 2841, 2852, 2854, 2855 can be corrected by the Bayesian filter. The Bayesian filter regards the classifier-output terrain predictions as observations. Due to the introduction of temporal correlation, which results in the lags in response to the variation of observations, the Bayesian filter outputs incorrect predictions at time 1663, 1665–1667, as shown in Figure 8b. Such lags can be found in Figure 8c as well. Furthermore, it is known that the Bayesian filter has the ability of tracking the observations. Therefore, as the Figure 8c demonstrated, the Bayesian filter fails if the classifier outputs incorrect terrain predictions continuously.



(**a**) Segment 1

(**b**) Segment 2

(**c**) Segment 3

Figure 8. Filtering details exhibition. • denotes real terrain, ○ denotes classifier-output terrain predictions, ○ denotes filter-output terrain predictions. NG, AR, CP, AG, SB, PT denote natural grass, asphalt road, cobble path, artificial grass, sand beach, plastic track, respectively.

Denote the sets $\mathcal{C}_1^T, \mathcal{C}_1^F, \mathcal{C}_2^T$, and $\mathcal{C}_2^F$ by the outputs of $\mathbb{C}_T(S_1^T), \mathbb{C}_F(S_1^F), \mathbb{C}_T(\mathcal{S}_2^T)$, and $\mathbb{C}_F(\mathcal{S}_2^F)$, where $S_1^T \subset \mathcal{S}_1^T$ and $S_1^F \subset \mathcal{S}_1^F$ denote the testing set of $\mathcal{S}_1^T$ and $\mathcal{S}_1^F$, respectively. Feeding these classifier's output set into the Bayesian filter, the classification results are increased by approximately 5% to 10%. The filtering accuracies with different μ are exhibited in Figure 9. It is observed that the accuracies almost reach 97% and 98% with the Bayesian filter performing on $\mathcal{C}_1^T, \mathcal{C}_1^F$, which means the offline classification accuracy increases by approximately 10%. On the other hand, filtering on $\mathcal{C}_2^T$ and $\mathcal{C}_2^F$ does not see such an effectiveness, which increases the classification by approximately 7% only.

The influences on Bayesian filtering and pseudo-labeling algorithm of different diagonal elements are shown in Figure 10. The term "*true*" means the number of the extracted samples that are not key samples (i.e., the samples that can be classified correctly), while "*false*" means the number of key samples (i.e., the samples that cannot be classified correctly). The term "*all*" means the number of the extracted samples. The terms "*true-positive*", "*false-positive*", and "*all-positive*" mean the numbers of the *true* samples, *false* samples, and *all* samples which could be correctly labeled by the proposed pseudo-labeling algorithm, respectively. It is observed that the Bayesian filter could increase the classification accuracy to some extent. In the time domain, the increasing accuracy varies from 1.6% to 4%, and peaks when the diagonal element exceeds 97%. Such an accuracy promotion could be found in the frequency domain more apparently. Furthermore, the pseudo-labeling algorithm could extract more *false* and *false-positive* samples with the diagonal element getting larger, both in the time and frequency domain. On the contrary, the number of *true* and *true-positive* samples does not increase significantly. Therefore, the pseudo-labeling algorithm could reach a high performance with a larger diagonal element.

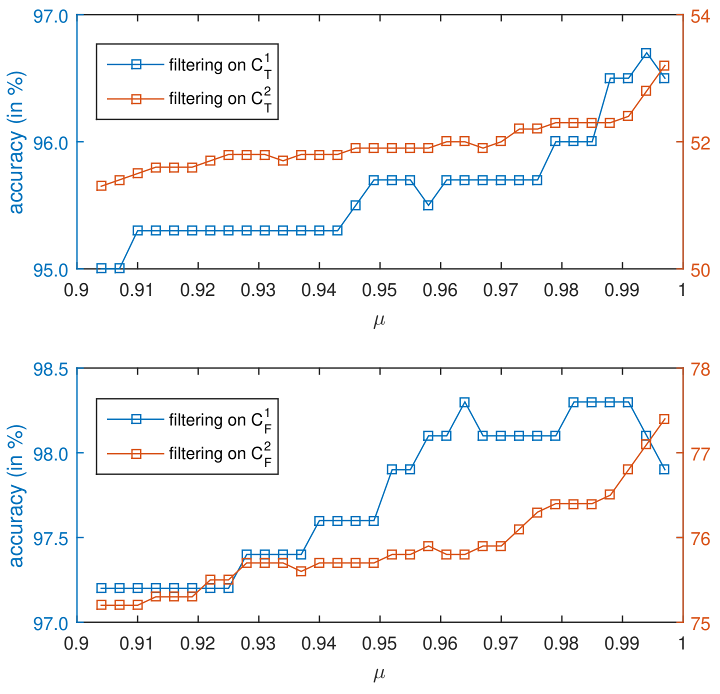


Figure 9. Accuracies of filtering results with different μ. The upper figure is for time domain, while the lower one for frequency domain.

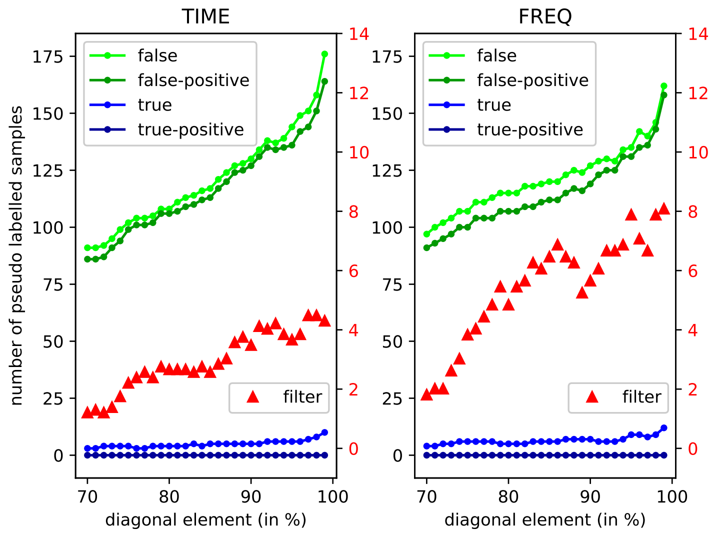


Figure 10. The influences on Bayesian filtering and pseudo-labeling algorithm of different diagonal elements. The right axis corresponds to *filter*. The left axis corresponds to *false*, *false-positive*, *true*, *true-positive*. The left figure is for time domain, while the right one for frequency domain.

3.4. Comparative Study of Adaptation in a Dynamic Environment

As shown above, the classifier trained on S_1 cannot achieve a high accuracy on S_2 for the presence of data drift. Now we are in the position to evaluate how DyVTC could retrieve the classification accuracy by incremental learning on the data chunks. As aforementioned, the terrain classification in a dynamic environment has rarely been investigated. We construct some terrain classification methods by applying the existing learning algorithms. The performances of the proposed DyVTC and those constructed ones will be evaluated. The 9 methods are shown as follows:

- *IE1:* The proposed DyVTC. IE is the abbreviation of *in- and ex-disagreement*.
- *IE2:* Similar to IE1, we use the *a priori* ex-disagreement, instead of *a posteriori* ex-disagreement.

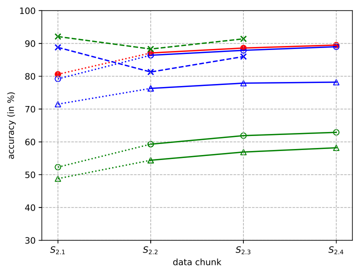
- *IE3:* Similar to IE1, we use both *a priori* and *a posteriori* ex-disagreement, which are combined using logical OR.
- *CT.95:* Using co-training algorithm (see [47]) to tackle such a terrain classification problem. The confidence threshold is 0.95.
- *CT.8:* Similar to CT.95, but the confidence threshold is 0.8.
- *ST.95:* Using the self-training algorithm for both domains. The similar idea can be found in [35,36]. The confidence threshold is 0.95.
- *ST.8:* Similar to ST.95, but the confidence threshold is 0.8.
- *KM.95:* Using an advanced fuzzy k means (see [48]) semi-supervised clustering algorithm to label the newly collected samples. The confidence threshold is 0.95.
- *KM.8:* Similar to KM.95, but the confidence threshold is 0.8.

The performances of pseudo-labeling algorithms are shown in Table 3. It is observed that the IE1 outperforms all the other algorithms in accuracy. The IE1 algorithm could only extract 100–200 samples from the whole 3000 samples, and the *true-positive* accuracy is 0. However, most of the extracted samples are key samples and these key samples could be labeled in an extremely high accuracy (over 95%). Hence, as shown in Figure 11, such a pseudo-labeling algorithm could increase the classification accuracy on S_2. The IE2 and IE3 are the variants of IE1. IE2 could extract many *true* samples and label them in 100% accuracy, but its *false-positive* accuracy is 0%. This indicates IE2 cannot bring valuable information, and thus cannot increase nor decrease the classification accuracy. All indices of IE3 are the sums of the corresponding indices of IE1 and IE2, and consequently, the performance of IE3 is between those of IE1 and IE2. We can also observe that the pseudo-labeling accuracies of IE2 and IE3 decrease at learning steps 2 and 3, but the classifier accuracy does not decrease. This is because IE2 and IE3 have high true-positive accuracy, which guarantee that the classifier accuracy does not decrease after update. In conclusion, it is the best to use *a posteriori* ex-disagreement in the pseudo-labeling algorithm.

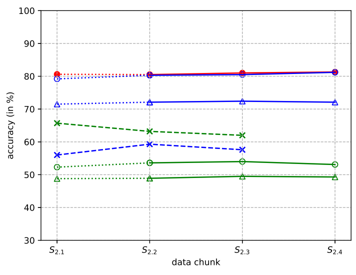
Table 3. Comparison of pseudo-labeling algorithms performing on $S_{2.1}$.

		Method								
		IE1	**IE2**	**IE3**	**CT.95**	**CT.8**	**ST.95**	**ST.8**	**KM.95**	**KM.8**
Time Domain	**True**	8	71	79	753	1005	716	1096	658	1542
	True-Positive	0	71	71	702	886	653	868	610	1071
	Accuracy	0%	**100%**	89.8%	93.2%	88.2%	91.2%	79.2%	92.7%	69.5%
	False	157	37	194	491	963	155	503	566	1653
	False-Positive	152	0	152	462	834	112	378	43	186
	Accuracy	**96.8%**	0%	78.4%	94.1%	86.6%	72.3%	75.1%	7.6%	11.3%
	All	165	108	273	1244	1968	871	1599	1224	3195
	All-Positive	152	71	223	1164	1720	765	1246	653	1257
	Accuracy	92.1%	65.7%	81.7%	**93.6%**	87.4%	87.8%	77.9%	53.3%	39.3%
Freq Domain	**True**	9	65	74	765	1232	1175	1691	5	2298
	True-Positive	0	65	65	645	881	708	862	3	1985
	Accuracy	0%	**100%**	87.8%	84.3%	71.5%	60.3%	51.0%	60.0%	86.4%
	False	146	51	197	98	354	75	260	0	910
	False-Positive	143	0	143	65	211	43	132	0	175
	Accuracy	**97.9%**	0%	72.6%	66.3%	59.6%	57.3%	50.8%	0.0%	19.2%
	All	155	116	271	863	1586	1250	1951	5	3208
	All-Positive	143	65	208	710	1092	751	994	3	2160
	Accuracy	**92.3%**	56.0%	76.8%	82.3%	68.9%	60.0%	50.9%	60.0%	67.3%

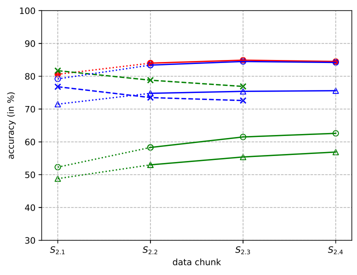
The CT.95 and CT.8 could increase the accuracy of the time-domain classifier but decrease that of the frequency-domain classifier, which is caused by the unequal performances of the two domains. The frequency-domain classifier performs much better, so it acts as a supervisor of the time-domain classifier. The ST.95 and ST.8 do not utilize a mutual learning mechanism, thus they have no effect on the classifier accuracy. The KM.95 and KM.8 only work under clustering assumption which is seldom satisfied when data drift occurs. Hence, the classifier accuracy decreases after updating using the KM methods. In conclusion, the IE methods could increase the classifier accuracy by incremental learning, but the others cannot work or even are counterproductive.



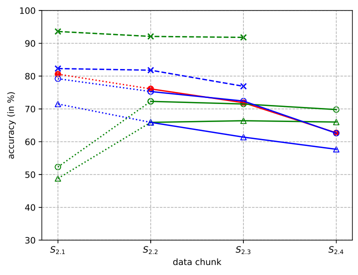
(**a**) IE1



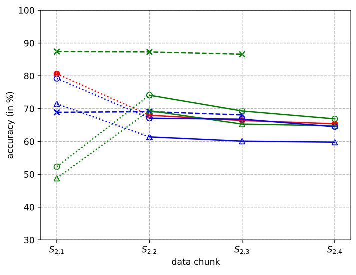
(**b**) IE2



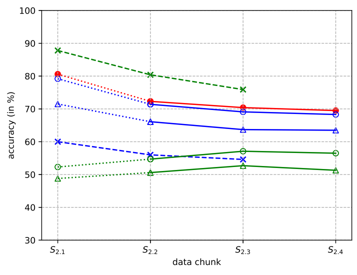
(**c**) IE3



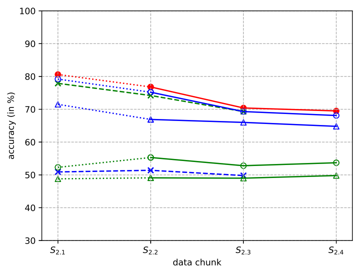
(**d**) CT.95



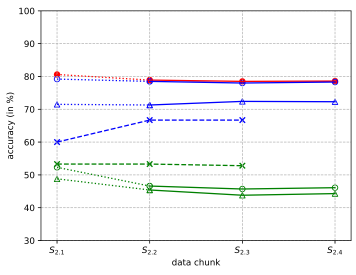
(**e**) CT.8



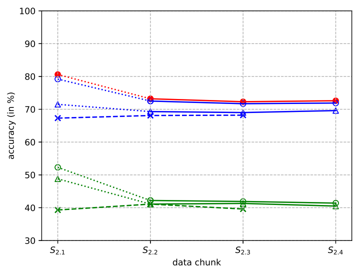
(**f**) ST.95



(**g**) ST.8



(**h**) KM.95



(**i**) KM.8

Figure 11. Accuracies of iterative incremental learning. The pseudo-labeling algorithm is conducted on $S_{2.1}$, $S_{2.2}$, and $S_{2.3}$, while the classifier is re-trained incrementally at the end of $S_{2.1}$, $S_{2.2}$, and $S_{2.3}$. The original classifier which is trained on $S_{1.1}$ is tested on $S_{2.1}$, while the updated classifiers are tested on $S_{2.2}$, $S_{2.3}$, and $S_{2.4}$. The fusion is based on the 2nd scheme. The marker definitions follow: ● denotes fusion accuracy; ○, △, and ∗ denote the frequency-domain filter, classifier, and pseudo-labeling accuracy, respectively; ○, △, and ∗ denote the time-domain filter, classifier, and pseudo-labeling accuracy, respectively.

The time cost is shown in Table 4. It can be observed that IE1, IE2, and IE3 take the shortest time to generate the pseudo-labeled sample set, while KM.95 and KM.8 is the most time-consuming. Unlike KM.95 and KM.8, which could only work after a data chunk is collected completely, IE1, IE2, IE3, CT.95, CT.8, ST.95, ST.8 could generate the pseudo-labeled samples at the time when a vibration frame prediction is finished, so the time cost of pseudo-labeling of these methods could be ignored. For the incremental learning part, compared with CT, ST, and KM, IE1 and IE3 use less pseudo-labeled samples to train the last classifier incrementally, but are the most time-consuming. This is because the majority of the pseudo-labeled samples generated by IE1 and IE3 are correctly-labeled key samples, which leads to the changing of classifier. Even so, IE1 and IE3 could be done within one second, which guarantees the real-time application.

Table 4. Time cost (in microsecond).

Method	Pseudo Labeling	Incremental Learning
IE1	12	584
IE2	13	152
IE3	16	651
CT.95	62	195
CT.8	147	237
ST.95	73	213
ST.8	196	281
KM.95	359	204
KM.8	838	323

4. Conclusions

In this paper, we propose a novel vibration-based terrain classification method for autonomous robots working in a dynamic environment, mainly to suppress the affect rendered by data drift, during the period that manual labels do not arrive. We mainly propose an ex- and in-disagreement-based learning algorithm, which is verified to be powerful to extract key samples and label them in high accuracy. In order to activate such a learning framework, we divide the vibration view into two domains, which may produce ex-disagreements; and introduce the Bayesian filter to correct the classification results, which may produce in-disagreements. The real-world experiment shows that the proposed DyVTC could reach an accuracy of 89.5%, which outperforms the existing VTC methods.

Author Contributions: Conceptualization, C.C. and W.L.; data curation, J.C.; formal analysis, J.C. and Z.L.; funding acquisition, W.L.; investigation, K.L.; methodology, C.C. and J.C.; project administration, Z.L.; resources, W.L. and Z.L.; software, Y.W., C.Y., and S.M.; validation, Y.W.; visualization, W.L.; writing—original draft, C.C.; writing—review and editing, W.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the National Natural Science Foundation of China (61903353), the 2020 Domestic Visiting Training Project for Young Key Teachers in Colleges and Universities (gxgnfx2020175), the SINOPEC Programmes for Science and Technology Development (PE19008-8), and the Fundamental Research Funds for the Central Universities (WK2100000013).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Song, W.; Cho, K.; Um, K.; Won, C.S.; Sim, S. Complete scene recovery and terrain classification in textured terrain meshes. *Sensors* **2012**, *12*, 11221–11237. [CrossRef] [PubMed]
2. Reina, G. Cross-coupled control for all-terrain rovers. *Sensors* **2013**, *13*, 785–800. [CrossRef] [PubMed]
3. Reinstein, M.; Kubelka, V.; Zimmermann, K. Terrain adaptive odometry for mobile skid-steer robots. In Proceedings of the 2013 IEEE International Conference on Robotics and Automation, Karlsruhe, Germany, 6–10 May 2013; pp. 4706–4711.
4. Brooks, C.A.; Iagnemma, K. Vibration-based terrain classification for planetary exploration rovers. *IEEE Trans. Robot.* **2005**, *21*, 1185–1191. [CrossRef]
5. Lv, W.; Kang, Y.; Zhao, Y. Self-tuning asynchronous filter for linear Gaussian system and applications. *IEEE/CAA J. Autom. Sin.* **2018**, *5*, 1054–1061. [CrossRef]
6. Pentzer, J.; Brennan, S.; Reichard, K. Model-based Prediction of Skid-steer Robot Kinematics Using Online Estimation of Track Instantaneous Centers of Rotation. *J. Field Robot.* **2014**, *31*, 455–476. [CrossRef]
7. Lv, W.; Kang, Y.; Qin, J. FVO: Floor Vision Aided Odometry. *Sci. China Inf. Sci.* **2019**, *62*, 12202. [CrossRef]
8. Lv, W.; Kang, Y.; Zhao, Y.B.; Wu, Y.; Zheng, W.X. A Novel Inertial-Visual Heading Determination System for Wheeled Mobile Robots. *IEEE Trans. Control Syst. Technol.* **2020**, 1–8. [CrossRef]

9. Pentzer, J.; Reichard, K.; Brennan, S. Energy-based path planning for skid-steer vehicles operating in areas with mixed surface types. In Proceedings of the 2016 American Control Conference, Boston, MA, USA, 6–8 July 2016; pp. 2110–2115.
10. Lee, J.; Lim, J.; Lee, J. Compensated Heading Angles for Outdoor Mobile Robots in Magnetically Disturbed Environment. *IEEE Trans. Ind. Electron.* **2018**, *65*, 1408–1419. [CrossRef]
11. Lv, W.; Kang, Y.; Qin, J. Indoor Localization for Skid-Steering Mobile Robot by Fusing Encoder, Gyroscope, and Magnetometer. *IEEE Trans. Syst. Man Cybern. Syst.* **2019**, *49*, 1241–1253. [CrossRef]
12. Chen, M. Disturbance attenuation tracking control for wheeled mobile robots with skidding and slipping. *IEEE Trans. Ind. Electron.* **2017**, *64*, 3359–3368. [CrossRef]
13. Lv, W.; Kang, Y.; Zhao, Y. FVC: A Novel Nonmagnetic Compass. *IEEE Trans. Ind. Electron.* **2019**, *66*, 7810–7820. [CrossRef]
14. Gonzalez, R.; Apostolopoulos, D.; Iagnemma, K. Slippage and immobilization detection for planetary exploration rovers via machine learning and proprioceptive sensing. *J. Field Robot.* **2017**, *35*, 231–247. [CrossRef]
15. Anantrasirichai, N.; Burn, J.; Bull, D. Terrain classification from body-mounted cameras during human locomotion. *IEEE Trans. Cybern.* **2015**, *45*, 2249–2260. [CrossRef] [PubMed]
16. Filitchkin, P.; Byl, K. Feature-based terrain classification for littledog. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, Vilamoura, Portugal, 7–12 October 2012; pp. 1387–1392.
17. Zhu, Y.; Luo, K.; Ma, C.; Liu, Q.; Jin, B. Superpixel segmentation based synthetic classifications with clear boundary information for a legged robot. *Sensors* **2018**, *18*, 2808. [CrossRef] [PubMed]
18. Dong, Y.; Guo, W.; Zha, F.; Liu, Y.; Chen, C.; Sun, L. A Vision-Based Two-Stage Framework for Inferring Physical Properties of the Terrain. *Appl. Sci.* **2020**, *10*, 6473. [CrossRef]
19. Zou, Y.; Chen, W.; Xie, L.; Wu, X. Comparison of different approaches to visual terrain classification for outdoor mobile robots. *Pattern Recognit. Lett.* **2014**, *38*, 54–62. [CrossRef]
20. Kurban, T.; Beşdok, E. A comparison of RBF neural network training algorithms for inertial sensor based terrain classification. *Sensors* **2009**, *9*, 6312–6329. [CrossRef]
21. Christie, J.; Kottege, N. Acoustics based terrain classification for legged robots. In Proceedings of the 2016 IEEE International Conference on Robotics and Automation, Stockholm, Sweden, 16–21 May 2016; pp. 3596–3603.
22. Valada, A.; Burgard, W. Deep spatiotemporal models for robust proprioceptive terrain classification. *Int. J. Robot. Res.* **2017**, *36*, 1521–1539. [CrossRef]
23. Wu, X.A.; Huh, T.M.; Mukherjee, R.; Cutkosky, M. Integrated Ground Reaction Force Sensing and Terrain Classification for Small Legged Robots. *IEEE Robot. Autom. Lett.* **2016**, *1*, 1125–1132. [CrossRef]
24. Walas, K. Terrain classification and negotiation with a walking robot. *J. Intell. Robot. Syst.* **2015**, *78*, 401–423. [CrossRef]
25. Otte, S.; Weiss, C.; Scherer, T.; Zell, A. Recurrent Neural Networks for fast and robust vibration-based ground classification on mobile robots. In Proceedings of the IEEE International Conference on Robotics and Automation, Stockholm, Sweden, 16–21 May 2016; pp. 5603–5608.
26. Bermudez, F.L.G.; Julian, R.C.; Haldane, D.W.; Abbeel, P.; Fearing, R.S. Performance analysis and terrain classification for a legged robot over rough terrain. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, Vilamoura-Algarve, Portugal, 7–12 October 2012; pp. 513–519.
27. Libby, J.; Stentz, A.J. Using sound to classify vehicle-terrain interactions in outdoor environments. In Proceedings of the 2012 IEEE International Conference on Robotics and Automation, Saint Paul, MN, USA, 14–18 May 2012; pp. 3559–3566.
28. Hoepflinger, M.A.; Remy, C.D.; Hutter, M.; Spinello, L.; Siegwart, R. Haptic terrain classification for legged robots. In Proceedings of the 2010 IEEE International Conference on Robotics and Automation, Anchorage, AK, USA, 3–7 May 2010; pp. 2828–2833.
29. Weiss, C.; Frohlich, H.; Zell, A. Vibration-based terrain classification using support vector machines. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, Beijing, China, 9–15 October 2006; pp. 4429–4434.
30. Mei, M.; Chang, J.; Li, Y.; Li, Z.; Li, X.; Lv, W. Comparative study of different methods in vibration-based terrain classification for wheeled robots with shock absorbers. *Sensors* **2019**, *19*, 1137. [CrossRef] [PubMed]

31. Dutta, A.; Dasgupta, P. Ensemble Learning With Weak Classifiers for Fast and Reliable Unknown Terrain Classification Using Mobile Robots. *IEEE Trans. Syst. Man Cybern. Syst.* **2016**, *47*, 2933–2944. [CrossRef]
32. Bai, C.; Guo, J.; Guo, L.; Song, J. Deep Multi-Layer Perception Based Terrain Classification for Planetary Exploration Rovers. *Sensors* **2019**, *19*, 3102. [CrossRef] [PubMed]
33. Wang, S.; Kodagoda, S.; Shi, L.; Wang, H. Road-Terrain Classification for Land Vehicles: Employing an Acceleration-Based Approach. *IEEE Veh. Technol. Mag.* **2017**, *12*, 34–41. [CrossRef]
34. Wang, C.; Lv, W.; Li, X.; Mei, M. Terrain Adaptive Estimation of Instantaneous Centres of Rotation for Tracked Robots. *Complexity* **2018**, *2018*, 4816712. [CrossRef]
35. Otsu, K.; Ono, M.; Fuchs, T.J.; Baldwin, I.; Kubota, T. Autonomous terrain classification with co-and self-training approach. *IEEE Robot. Autom. Lett.* **2016**, *1*, 814–819. [CrossRef]
36. Brooks, C.A.; Iagnemma, K. Self-supervised terrain classification for planetary surface exploration rovers. *J. Field Robot.* **2012**, *29*, 445–468. [CrossRef]
37. Shi, W.; Li, Z.; Lv, W.; Wu, Y.; Chang, J.; Li, X. Laplacian Support Vector Machine for Vibration-Based Robotic Terrain Classification. *Electronics* **2020**, *9*, 513. [CrossRef]
38. Lv, W.; Kang, Y.; Zheng, W.X.; Wu, Y.; Li, Z. Feature-temporal semi-supervised extreme learning machine for robotic terrain classification. *IEEE Trans. Circuits Syst. II Express Briefs* **2020**. [CrossRef]
39. Sheu, J.J.; Chu, K.T.; Li, N.F.; Lee, C.C. An efficient incremental learning mechanism for tracking concept drift in spam filtering. *PLoS ONE* **2017**, *12*, e0171518. [CrossRef]
40. Hostetter, G. Recursive discrete Fourier transformation. *IEEE Trans. Acoust. Speech Signal Process.* **1980**, *28*, 184–190. [CrossRef]
41. Li, Z.; Kang, Y.; Feng, D.; Wang, X.M.; Lv, W.; Chang, J.; Zheng, W.X. Semi-supervised learning for lithology identification using Laplacian support vector machine. *J. Pet. Sci. Eng.* **2020**, *195*, 107510. [CrossRef]
42. Chang, C.C.; Lin, C.J. LIBSVM: A library for support vector machines. *Acm Trans. Intell. Syst. Technol.* **2011**, *2*, 1–27. [CrossRef]
43. Cauwenberghs, G.; Poggio, T. Incremental and decremental support vector machine learning. In Proceedings of the Advances in Neural Information Processing Systems, Vancouver, BC, Canada, 3–8 December 2001; pp. 409–415.
44. Fang, H.; Tian, N.; Wang, Y.; Zhou, M.; Haile, M.A. Nonlinear Bayesian estimation: From Kalman filtering to a broader horizon. *IEEE/CAA J. Autom. Sin.* **2018**, *5*, 401–417. [CrossRef]
45. Kittler, J.; Hatef, M.; Duin, R.; Matas, J. On combining classifiers. *IEEE Trans. Pattern Anal. Mach. Intell.* **1998**, *20*, 226–239. [CrossRef]
46. Xue, J.; Zhang, H.; Dana, K. Deep texture manifold for ground terrain recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018; pp. 558–567.
47. Blum, A.; Mitchell, T. Combining labeled and unlabeled data with co-training. In Proceedings of the Eleventh Annual Conference on Computational Learning Theory, Madison, WI, USA, 24–26 July 1998; pp. 92–100.
48. Gan, H.; Nong, S.; Rui, H.; Tong, X.; Dan, Z. Using clustering analysis to improve semi-supervised classification. *Neurocomputing* **2013**, *101*, 290–298. [CrossRef]