Journal of Electrical Engineering, Vol. 76, No. 5, 2025, pp. 420-428
https://reference-global.com/journal/JEE

Vibration based terrain classification for mobile robot using machine learning methods

Mátyás Sátor-Érsek, Slavomír Kajan, Ladislav Körösi

Nowadays, mobile robots require accurate and reliable control algorithms. The parameters of these algorithms are strongly influenced by the characteristics of the terrain over which the robot moves. The type of terrain can be identified using machine learning approaches. One promising approach is classification using measurements from inertial sensors. This method provides effective classification without the need for complex models. This paper deals with the analysis of different machine learning models using the vibration of inertial sensors. We experimented with various manual and automatic feature extraction techniques, each combined with different classification algorithms. At the conclusion of the evaluation phase, the most successful method from each feature extraction category will be selected based on performance metrics. These selected models were further validated on a custom dataset to assess their generalization capabilities in real-world conditions.

Keywords: mobile robotics, vibration analysis, machine learning, terrain classification, inertial sensor

1 Introduction

Wheeled mobile robots are increasingly being adopted for a wide range of outdoor applications. As a result, these systems must be capable of operating smoothly across various terrain types. To achieve this level of robustness and adaptability, it is essential to classify the underlying terrain based on sensor data collected during the robot's operation.

Knowledge of the terrain type provides critical information for the design of control systems in mobile robotics [1]. By identifying the current terrain, it becomes possible to design compensation mechanisms within the control.

Terrain classification for mobile robots can be achieved using machine learning techniques. A common approach involves visual terrain classification using onboard cameras [2]. However, this method typically requires image segmentation or object detection as a pre-processing step to isolate the terrain from other elements in the scene [3]. Such visual-based approaches are often computationally expensive and come with several limitations, including dependence on adequate lighting conditions and the availability of a vision system – both of which may not be feasible in all robotic platforms or environments.

Other studies have explored audio-based methods, where terrain classification is performed using sound recordings captured by microphones [4]. While this approach can be effective in certain scenarios, it is highly sensitive to environmental noise, which can significantly degrade performance in real-world conditions. Additionally, some projects utilize LiDAR-based techniques [5], where terrain classification is achieved through point cloud processing. These methods often require complex data processing pipelines and can be computationally demanding, limiting their applicability on resource-constrained platforms.

This paper focuses on a vibration-based approach, where relevant information is extracted from inertial sensors such as accelerometers, gyroscopes and magnetometers [6]. A key advantage of this method is its broad applicability, as most wheeled mobile robots are already equipped with these types of sensors. Furthermore, the outputs of these sensors are one-dimensional time-series signals, which reduces the complexity of the required machine learning models and eliminates the need for computationally intensive processing structures.

A critical aspect of vibration-based terrain classification is the selection of an appropriate feature extraction method. Some studies extract features from the frequency domain [7] of the inertial sensor signals, leveraging transformations such as the Fast Fourier Transform to capture spectral characteristics. However, the most successful approaches to date have typically relied on features extracted directly from the time-domain signals [8], which often retain more localized and transient information relevant to terrain classification [9]. There are studies that propose adaptive models capable of accounting for variations in the robot's speed, thereby improving classification robustness under dynamic operating conditions [10].

Institute of Robotics and Cybernetics, Faculty of Electrical Engineering and Information Technology,
Slovak University of Technology, Bratislava, Slovakia
xsatorersek@stuba.sk, slavomir.kajan@stuba.sk, ladislav.korosi@stuba.sk

https://doi.org/10.2478/jee-2025-0044, Print (till 2015) ISSN 1335-3632, On-line ISSN 1339-309X

In addition to manual feature extraction methods, there are also architectures capable of automatically extracting informative features directly from raw sensor signals using various deep neural networks [13, 14]. In this study, we explore and compare multiple approaches involving both manual and automatic feature extraction techniques. The performance of each method is evaluated based on validation and testing accuracy, defined as the ratio of correctly classified samples to the total number of samples. This comparative analysis aims to identify the most effective and efficient strategies for vibration-based terrain classification.

The models will first be trained and evaluated on an open source dataset. Subsequently, the best-performing model from each group – manual and automatic feature extraction – based on validation and testing accuracy, will be selected for further evaluation using our custom-collected dataset.

The primary motivation behind this comparative study was to highlight the strengths and limitations of each approach, providing valuable insights that can help practitioners select the most suitable algorithm for their specific application requirements.

2 Methods

In this study, we employed machine learning techniques in combination with vibration analysis [13] for terrain classification. The classification algorithm consisted of two primary stages. In the first part we extracted the features from raw sensor data, which potentially contain discriminative information related to terrain types. The feature extraction can be performed with manual and automatic methods. In the second stage, the extracted features were utilized to train the classification models.

2.1 Manual feature extraction methods

At the beginning of our research, we experimented with manual feature extraction methods.

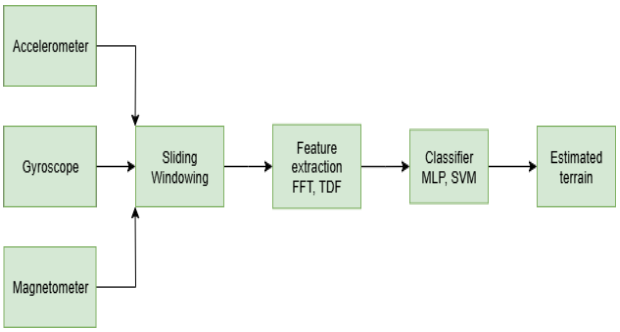

Fig. 1. Parts of the classification algorithm with manual feature extraction

In this case, first we computed certain features from the time windows, which later will be used for classification (Fig. 1).

2.1.1 Fast Fourier Transform

The Fast Fourier Transform (FFT) is a widely used algorithm in the field of signal processing. It operates on discrete signals and transforms a time-domain vector into its corresponding representation in the complex frequency domain. The resulting complex-valued output can be decomposed into magnitude and phase components [14]. For terrain classification in robotic applications, the magnitude spectrum is particularly informative and can be utilized as a set of input features for training classification models [15].

2.1.2 Time domain features

While certain methods extract features from the frequency domain, this study focused on techniques that extract features directly from time-domain segments. In this study we used 9 time-domain features (TDF) of sensor signals for terrain classification:

1. Mean absolute value (MAV)
2. Root mean squared (RMS)
3. Number of zero crossings (NZC)
4. Maximum value (MAX)
5. Minimum value (MIN)
6. Peak-to-Peak (PTP)
7. Number of slope changes (NSSC)
8. Willison amplitude (WAMP)
9. Waveform length (WL)

The definition of these features can be found in [8].

2.1.3 Classification models

As the classification model, first we employed a multilayer perceptron (MLP) network [18]. It is a machine learning model which can be trained with supervised learning methods. This model is widely used due to its effectiveness and the flexibility it offers in selecting hyperparameters. The MLP architecture is a forward network and consists of an input layer, multiple hidden layers, and an output layer. In each hidden layer, neurons compute a weighted sum of the inputs from the preceding layer, add a bias term, and pass the result through a nonlinear activation function. The output of this activation function then serves as input to the subsequent layer. Finally, a Softmax activation function is applied to the output layer to

ensure that the resulting vector forms a valid probability distribution (i.e., elements sum to one). The predicted class corresponds to the index of the maximum value in this final output vector. The weights and biases of the hidden layer were trained with the backpropagation algorithm with the cross-entropy error function.

Beside the widely popular neural network, we also used Support Vector Machine (SVM) as a classifier. Support Vector Machines are supervised learning algorithms that aim to find an optimal separating hyperplane between classes. They are particularly well-suited for high-dimensional feature spaces and can achieve high accuracy even with relatively small training datasets. These hyperplanes are defined by support vectors – training samples closest to the decision boundaries. The goal is to maximize the margin between classes [19].

2.2 Automatic feature extraction methods

In addition to manual feature extraction combined with traditional classifiers, we also employed deep learning techniques for terrain classification (Fig. 2).

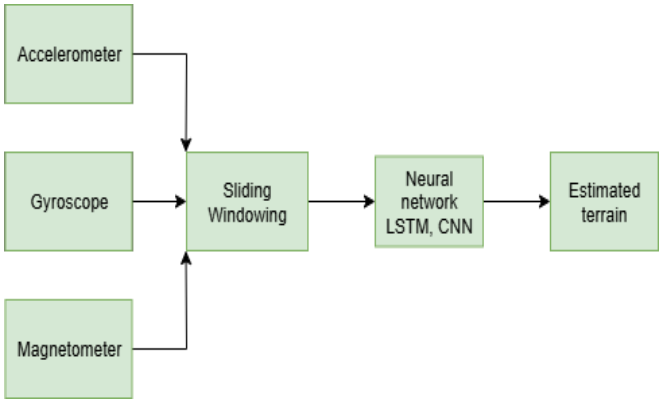

Fig. 2. Parts of the classification algorithm with automatic feature extraction

2.2.1 Long short term memory

Long Short-Term Memory (LSTM) networks are a specialized subset of the recurrent neural network (RNN) family. Unlike feedforward neural networks, RNNs incorporate recurrent connections, allowing neurons to retain information from previous time steps. This architectural feature makes them well-suited for processing sequential data, as they can capture temporal dependencies and patterns over time [11].

The output of the LSTM network is a sequence of hidden states that preserve temporal structure. To convert this sequential output into discrete terrain classes, one or more fully connected layers are appended to the network. These layers map the temporal features extracted by the LSTM to a fixed-size output vector, enabling classification of terrain types based on the input sensor measurements [16].

During the training process, we implemented an LSTM network for terrain classification. The architecture consisted of two LSTM layers, each with 75 hidden units, followed by a fully connected perceptron layer containing 200 neurons. The hyperparameters – such as the number of hidden units and neurons – were selected using a grid search algorithm to maximize validation accuracy.

2.2.2 Convolutional neural network

In this approach, feature extraction was performed automatically by convolutional neural networks (CNNs). These models are capable of handling complex tasks and have demonstrated high performance across various domains. The most widely used variant is the two-dimensional CNN, commonly applied in image processing. However, for sensor data, one-dimensional CNNs are more appropriate and were utilized in this study [17]. Although such CNN architectures are commonly employed in prediction tasks, they can also be effectively applied to classification problems, as demonstrated in this study. The parameters of the CNN layers were optimized with the back-propagation algorithm.

During training, we utilized a CNN comprising three one-dimensional convolutional layers. In CNN ReLU activation function was used. Batch normalization and max pooling were applied between the layers to improve training stability and reduce spatial dimensions, respectively.

3 Data acquisition

To develop a reliable classification model, it was essential to use a dataset containing inertial sensor measurements recorded during the operation of a mobile robot.

3.1 Open source dataset

In order to initially verify the feasibility of vibration-based terrain classification, we utilized an open source dataset prior to collecting measurements from our custom mobile robot platform. This preliminary step allowed us to evaluate different classification methods and feature extraction approaches in a controlled environment before applying them to data collected in real-world conditions [20].

The dataset included measurements from an accelerometer, gyroscope, and magnetometer collected over

six different terrain types: concrete, grass, sand, paving stone, pebbles, and synthetic running track. During data acquisition, the mobile robot operated under open-loop control with a fixed pulse-width modulation (PWM) duty cycle. For each terrain, seven measurements were recorded, each lasting approximately 4.5 seconds. To ensure consistency in the training process, only data collected at 100% PWM duty cycle were used for model development.

The open source dataset was divided into 3 parts. We used 4 sessions from each terrain type for training, the other 2 were used for validation purposes and the last session was used for testing.

3.2 Wheeled mobile robot

To verify selected machine learning models for terrain classification, it was necessary to measure data from the inertial sensors of a mobile robot. For this purpose, we constructed a prototype of a wheeled mobile robot with the necessary IMU sensors.

For data acquisition we assembled a small mobile robot kit with 4-wheel differential drive (Fig. 3). The four wheels were driven by 1.5 V DC motors. To control the direction and speed of the motors we used a two channel L298 DC motor driver [22]. Dimensions of the robot were: 153 × 255 × 72 mm. The robotic system was controlled by an ESP32 microcontroller [21]. This robotic system uses DC power supply with 7.4 V.

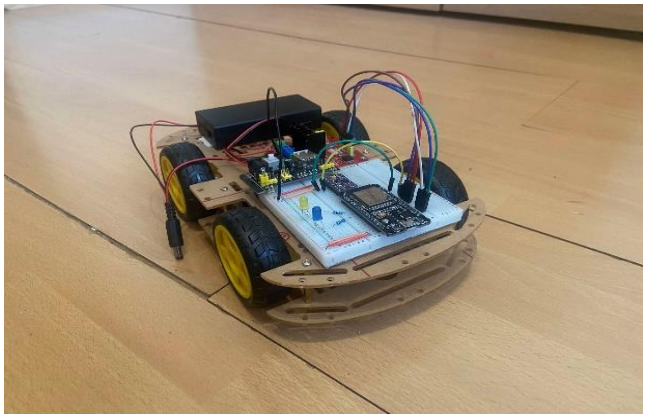

Fig. 3. Wheeled mobile robot kit for the data acquisition

To measure the linear acceleration, the angular velocity and the induction of the magnetic field of the mobile robot we used an *MPU9250* sensor board [22], with a 3-axial accelerometer, gyroscope and magnetometer. The sensor board communicated with the microcontroller using the I^2C bus. Before every measurement we calibrated the sensors.

Table 1. Measuring boundaries of the inertial sensors

Sensor setting	Value
Accelerometer range	±4g
Gyroscope range	±500 °/s
Magnetometer range	± 4800 μT
Low pass filter cutoff frequency	92 Hz

The measured data was transmitted to a distant computer using the *UDP* protocol [23]. We used a constant sampling frequency of 200 Hz to store the recorded data in a *CSV* file. In this study we used 90% and 100% PWM duty cycle for the mobile robot.

3.3 Custom dataset

During the data acquisition we collected measurements from inertial sensors for 5 types of terrain – *concrete, parquet, asphalt, paving stone, synthetic running track*. Every *CSV* file had stored the timestamps and the 9 axes of the inertial sensors. We have paid attention to have a balanced dataset of all types of terrain and running speeds. Later the recorded time series were divided into smaller segments with the length of 1 second (200 samples per segment). Figure 4 shows an example of a comparison of z-axis sensor signals for two selected terrains.

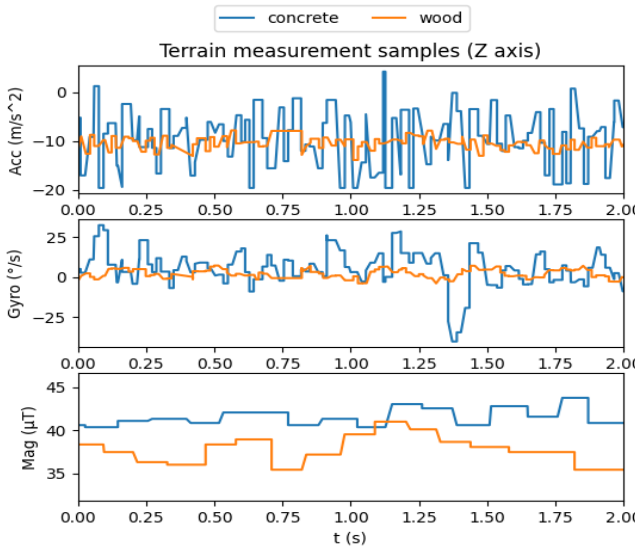

Fig. 4. Comparison of two different terrains on the *z* axis of each of 3 sensors

The dataset was divided into three parts. We used 65% of the data for training, 25% for validation and the rest 10% for testing of the neural networks. To acquire more samples for training we used sliding window approach, where we shifted the sliding window by 50 samples.

4 Experimental results

During training, we evaluated multiple sets of hyperparameters for the models. Model performance was assessed using classification accuracy. For both manual and automatic feature extraction approaches, accuracy was monitored on a validation dataset after each training epoch to mitigate overfitting. Upon completion of training, final model performance was evaluated on a separate test dataset. The training process was terminated upon reaching the maximum number of training epochs.

4.1 Results on the open source dataset

First, we evaluated all the mentioned methods on an open source dataset. Due to the relatively short training time – only a few minutes per model – we were able to train multiple networks with varying hyperparameter configurations. After the experiments, the model achieving the highest validation accuracy was selected and saved for further evaluation.

At the beginning we designed models with a minimal number of neurons. Subsequently, the number of trainable parameters was gradually increased until the accuracy on both the validation and test datasets began to stagnate. The objective was to identify a model that achieved the highest possible accuracy while maintaining a minimal network size. This approach was motivated by the desire to develop a model suitable for deployment on computationally constrained platforms. Tables 2 and 3 show the optimal parameters of the classification methods.

Table 2. Hyperparameters of the neural networks

Method	Optimiser	Learning rate	Batch size	Number of epochs	Number of hidden parameters
FFT + MLP	Adam	1×10^{-3}	32	20	648326
TDF + MLP	Adam	1×10^{-3}	32	60	15686
LSTM	NAdam	5×10^{-3}	20	175	87056
CNN	AdamW	1×10^{-3}	32	60	423894

Table 3. Hyperparameters of the SVM

Kernel	linear
C	0.1
γ	0.1
Number of parameters	70

Table 4. Validation and testing accuracies of the trained models on the open source dataset

Method	Val. Acc. (%)	Val. Acc. Std. (%)	Test. Acc. (%)	Test. Acc. Std. (%)
FFT + MLP	96.03	2.30	83.80	2.36
TDF + SVM	93.45	0.97	95.83	0.97
TDF + MLP	97.02	1.29	92.59	5.59
LSTM	72.22	15.71	74.54	9.78
CNN	83.13	3.52	70.37	2.36

During this process, the dataset was randomly partitioned into training, validation, and test subsets. To evaluate the impact of randomness in both data splitting and neural network weight initialization, the experiment was repeated five times using the same hyperparameter settings. Finally, the average accuracy was computed for both the validation and testing phases, along with the corresponding standard deviation to quantify variability across multiple runs. Based on the results in Tab. 4, all models successfully classified terrain types from the sensor measurements. Apart from one case, the standard deviations of the accuracy values were relatively low, indicating high consistency across multiple runs. Therefore, the influence of random factors – such as dataset splitting and weight initialization – on model performance can be considered negligible.

In the next phase, we selected the classification methods that achieved the highest accuracy for both the manual and automatic feature extraction approaches. These selected models were then validated using a custom dataset collected by our test mobile robot.

4.2 Results on the custom dataset

For the manual feature extraction approach, we selected the Support Vector Machine combined with time-domain features. In addition to delivering promising results on the open source dataset, this method offers the advantages of computational efficiency and low resource requirements. For the automatic feature extraction approach, convolutional neural networks with one-dimensional convolution were selected. Although this architecture demonstrated limited effectiveness on the open source dataset – primarily due to the insufficient amount of training data. The custom dataset contained significantly longer sequences from the inertial sensors, which provided the opportunity to improve classification performance.

In this phase we also calculated the validation and testing dataset for the selected two models. Table 5 shows high validation and testing accuracies for both selected terrain classification methods. As the amount of data in the dataset increased, CNN achieved better results. For some terrains, we even achieved higher classification accuracy than when using the open source dataset.

Table 5. Validation and testing accuracies of the models trained on the custom dataset

Method	Val. Acc. (%)	Test. Acc. (%)
TDF + SVM	91.94	85.25
CNN	96.30	91.67

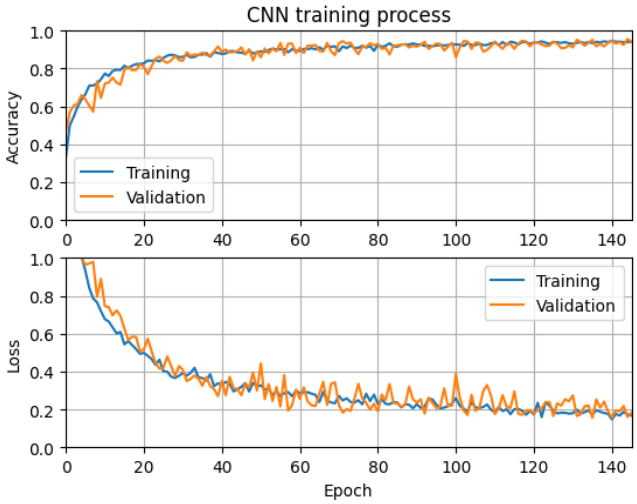

Fig. 5. Training process of the CNN model

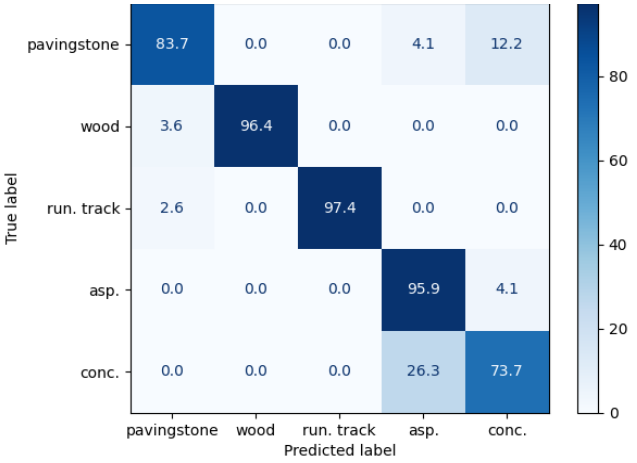

Fig. 6. Confusion matrix of the SVM classifier

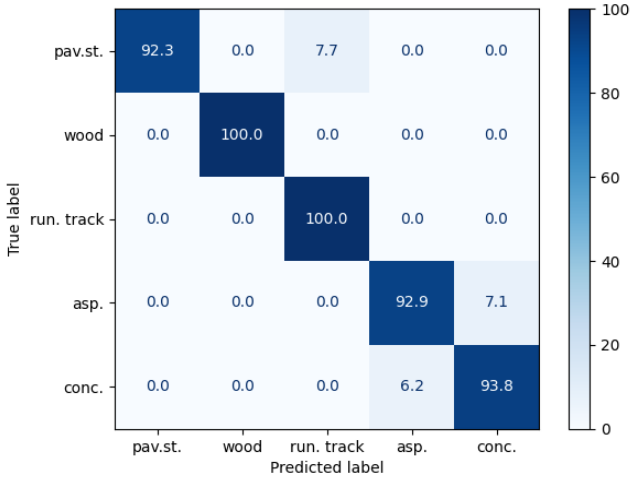

Fig. 7. Confusion matrix of the CNN

The CNN training process is shown in Fig. 5. To further analyze the classification performance, we plotted the confusion matrices for the testing dataset. This allowed us to identify which terrain classes were

most frequently misclassified by the models. Confusion matrices were generated for both the manual feature extraction approach and the automatic feature extraction approach, providing a comparative view of the strengths and weaknesses of each method across different terrain types. Compared to the SVM classifier, CNN achieved a classification accuracy higher than 90% for all classes.

5 Discussion

In the previous section, we compared several classification methods using the open source dataset and subsequently selected the best-performing models for both manual and automatic feature extraction approaches. Later these methods were validated on our custom dataset. Our findings indicate that all evaluated models were capable of learning to classify different terrain types effectively, demonstrating the general viability of both feature extraction strategies for terrain classification tasks.

Our first method, based on Fast Fourier Transform features combined with a Multilayer Perceptron (FFT + MLP), yielded promising results in terms of classification accuracy. However, this approach was deemed less suitable for real-time applications due to its higher computational complexity. Specifically, the model required many input features and contained approximately 40 times more trainable parameters than the TDF + MLP method. Next, TDFs were extracted to capture information about the terrain. These features were used to train MLP and SVM classification models. A major advantage of this approach is the significant reduction in input dimensionality – resulting in an input vector nearly 50 times smaller compared to the FFT-based method. Additionally, the selected features can be computed recursively, allowing for real-time feature extraction with each new inertial sensor sample. Between the two classifiers, the SVM was prioritized due to its suitability for embedded applications, particularly when using a linear kernel. The SVM model was validated on the custom dataset, where it achieved reasonably good performance. The accuracy was slightly lower compared to the results obtained on the open source dataset. This decrease in accuracy may be due to the greater similarity among terrain types in the custom dataset. The terrains in the open source dataset exhibited more distinct material characteristics – such as differences in adhesion, friction, and surface texture – which likely made classification easier.

Next, we conducted experiments with deep neural networks that directly processed raw time-series data. Initially, LSTM networks were trained for terrain classification. Although the LSTM models occasionally produced acceptable accuracy, the overall performance was lower than other methods. More notably, the models exhibited high standard deviation in their accuracy scores across multiple runs, indicating strong sensitivity to initial training conditions, such as weight initialization and data shuffling. Due to this instability and inconsistency, LSTM networks were not considered suitable for the terrain classification task in our application. The final method evaluated was a CNN with one-dimensional convolution. While this approach yielded relatively lower performance metrics on the open source dataset, we recognized its potential for improvement with a larger volume of training data. Consequently, the CNN model was retrained using our custom dataset, which contained longer and more numerous inertial sensor measurements. On this dataset, the model achieved significantly higher accuracy on both the validation and testing sets. These results demonstrate that convolutional neural networks are capable of effectively classifying terrain types, provided that a sufficiently large and representative training dataset is available.

As a final step, we evaluated the confusion matrices to identify which terrain classes were most challenging for the models to distinguish. For the TDF + SVM approach, the confusion matrix revealed that the most frequent misclassifications occurred between concrete and asphalt surfaces. This outcome is reasonable, given that these two terrain types share similar material properties, such as hardness, friction, and texture, which can make them difficult to differentiate based solely on inertial sensor data. On the other hand, the classification accuracy for the remaining terrain classes was nearly perfect. This is a promising result, as it confirms that the classification algorithm is capable of effectively learning and utilizing the vibration patterns associated with different terrain types.

Finally, we evaluated the confusion matrix for the CNN model. In this case, the classification accuracy was nearly perfect across all terrain classes. Although some confusion between asphalt and concrete was still observed – consistent with earlier results – the convolutional neural network demonstrated its ability to distinguish between terrain types with very similar physical characteristics when provided with a sufficiently large and representative training dataset. These results highlight the effectiveness of deep learning methods in capturing subtle distinctions in vibration patterns for terrain classification tasks.

6 Conclusion

In this study, we investigated vibration-based terrain classification methods using inertial sensor data in combination with machine learning models. The primary objective was to compare various classification techniques across both manual and automatic feature extraction approaches. Through a series of experiments, we evaluated the performance of the models with the

accuracy of the validation and testing process. An additional objective of this work was to identify classification methods with minimal computational cost, ensuring that the trained models would be suitable for deployment on hardware-constrained platforms. This consideration was particularly important for real-time terrain classification in embedded robotic systems, where processing power and memory resources are limited. Initially, the selected models were trained using an open source dataset containing terrain-related sensor measurements. Three methods were evaluated for the manual feature extraction approach, and two methods were tested for automatic feature extraction. Based on validation and testing accuracy, one method from each category was selected for further evaluation. These selected models were subsequently tested on a custom dataset collected using the experimental mobile robot to assess their performance under real-world conditions.

For the manual feature extraction approach, the Support Vector Machine combined with time-domain features was selected. This method demonstrated excellent performance on the open source dataset and also yielded promising results when applied to the custom dataset. Its strong generalization capability, along with low computational complexity, makes it a suitable candidate for real-time terrain classification in embedded systems. For the automatic feature extraction approach, a CNN with one-dimensional convolution was selected for validation on the custom dataset. While the testing accuracy of this model on the open source dataset was slightly lower compared to the TDF + SVM method, it achieved superior performance on the custom dataset. This is particularly noteworthy given the higher similarity in the physical properties of the terrains in the custom dataset, indicating that the CNN was able to effectively learn subtle distinctions in the vibration signals associated with each terrain type.

In summary, we successfully trained machine learning models for vibration-based terrain classification using both manual and automatic feature extraction strategies. The achieved terrain classification accuracies of over 90% are comparable to results reported in related studies [7-12]. Our experiments indicate that, for datasets of limited size and for terrain types that are easily distinguishable, manual feature extraction coupled with lightweight classifiers (e.g., TDF + SVM) offers the best trade-off between accuracy and computational cost. Conversely, when terrain classes exhibit highly similar vibration characteristics, deep neural networks with automatic feature extraction (e.g., CNNs) perform well, provided that a sufficiently large and representative training set is available.

Acknowledgement

This work was supported by grant APVV-22-0169 from the Slovak Scientific Grant Agency and project No. 1/0202/23 of the Slovak Grant Agency VEGA. This work was funded by the EU NextGenerationEU through the Recovery and Resilience Plan for Slovakia under the project No. 09I04-03-V02-00034.

References

[1] Q. Zhu *et al.*, "Learning-based Traversability Costmap for Autonomous Off-road Navigation," June 2024, doi: 10.48550/arXiv.2406.08187.

[2] A. Sharma, "Self-Supervised Visual Terrain Classification." 2020. [Online]. Available: https://medium.com/swlh/self-supervised-visual-terrain-classification-6e4f09418328

[3] H. Wu, W. Zhang, B. Li, Y. Sun, D. Duan, and P. Chen, "Visual Terrain Classification Methods for Mobile Robots Using Hybrid Coding Architecture," in *2019 IEEE 4th International Conference on Image, Vision and Computing (ICIVC)*, 2019, pp. 17–22.

[4] D. Qin, G. Zhang, Z. Zhu, X. Zeng, and J. Cao, "An online terrain classification framework for legged robots based on acoustic signals," *Biomimetic Intelligence and Robotics*, vol. 3, no. 2, p. 100091, 2023.

[5] T. J. Pingel, K. C. Clarke, and W. A. McBride, "An improved simple morphological filter for the terrain classification of airborne LIDAR data," *ISPRS journal of photogrammetry and remote sensing*, vol. 77, pp. 21–30, 2013.

[6] K. Maenaka, "MEMS inertial sensors and their applications," in *2008 5th International Conference on Networked Sensing Systems*, IEEE, 2008, pp. 71–73.

[7] E. M. Dupont, C. A. Moore, E. G. Collins Jr, and E. Coyle, "Frequency response method for terrain classification in autonomous ground vehicles," *Autonomous Robots*, vol. 24, no. 4, pp. 337–347, 2008.

[8] P. Sarcevic *et al.*, "Online Outdoor Terrain Classification Algorithm for Wheeled Mobile Robots Equipped with Inertial and Magnetic Sensors," *Electronics*, vol. 12, p. 3238, July 2023.

[9] D. Csík, Á. Odry, J. Sárosi, and P. Sarcevic, "Inertial sensor-based outdoor terrain classification for wheeled mobile robots," in *2021 IEEE 19th International Symposium on Intelligent Systems and Informatics (SISY)*, 2021, pp. 159–164.

[10] M. Wang, L. Ye, and X. Sun, "Adaptive online terrain classification method for mobile robot based on vibration signals," *International Journal of Advanced Robotic Systems*, vol. 18, no. 6, p. 17298814211062035, 2021.

[11] S. Otte, C. Weiss, T. Scherer, and A. Zell, "Recurrent Neural Networks for fast and robust vibration-based ground classification on mobile robots," May 2016, pp. 5603–5608.

[12] F. Vulpi, A. Milella, R. Marani, and G. Reina, "Recurrent and convolutional neural networks for deep terrain classification by autonomous robots," *Journal of Terramechanics*, vol. 96, pp. 119–131, 2021.

[13] A. V. Oppenheim, R. W. Schafer, and J. R. Buck, *Discrete-Time Signal Processing*, Second. Prentice-hall Englewood Cliffs, 1999.

[14] H. J. Nussbaumer, "The fast Fourier transform," in *Fast Fourier transform and convolution algorithms*, Springer, 1981, pp. 80–111.

[15] D. Sadhukhan, C. Moore, and E. Collins, "Terrain Estimation Using Internal Sensors," in Proc. 10th IASTED Int. Conf. Robotics and Applications, Honolulu, HI, USA, Aug. 2004, paper 447-800

[16] Z. Sun, L. Di, and H. Fang, "Using long short-term memory recurrent neural network in land cover classification on Landsat and Cropland data layer time series," *International Journal of Remote Sensing*, vol. 40, no. 2, pp. 593–614, 2019.

[17] M. Olkhovskiy, E. Müllerová and P. Martínek, "Impulse signals classification using one dimensional convolutional neural network," *Journal of Electrical Engineering*, vol. 71, pp. 397-405, 2020

[18] M.-C. Popescu, V. E. Balas, L. Perescu-Popescu, and N. Mastorakis, "Multilayer perceptron and neural networks," *WSEAS Transactions on Circuits and Systems*, vol. 8, no. 7, pp. 579–588, 2009.

[19] S. Suthaharan, "Support vector machine," in *Machine learning models and algorithms for big data classification: thinking with examples for effective learning*, Springer, 2016, pp. 207–235.

[20] P. Sarcevic, "Outdoor Terrain Classification Database." https://github.com/petersarcevic/outdoor_terrain_classification_database, 2023.

[21] Espressif Systems, "ESP32 Hardware Reference Manual." [Online]. Available: https://docs.espressif.com/projects/esp-dev-kits/en/latest/esp32/esp-dev-kits-en-master-esp32.pdf

[22] TDK InvenSense, "MPU-9250 Product Specification." 2015. [Online]. Available: https://invensense.tdk.com/wp-content/uploads/2015/02/PS-MPU-9250A-01-v1.1.pdf

[23] J. Postel, "User Datagram Protocol." 1980. [Online]. Available: https://datatracker.ietf.org/doc/html/rfc768

Received 25 July 2025