Highly Accurate Visual Method of Terrain Classification Based on Novel Image Features

Machine Learning Team

February 13, 2025

Abstract

It is critical for autonomous systems to traverse hazardous terrain safely. In this paper, we propose a novel vision-based terrain classification method that extracts specialized image features including multiscale gray gradient-grade features, multiscale edge strength-grade features, and frequency spectrum-based features. Detailed mathematical formulations are provided along with illustrative graphics. The classifiers—KNN, SVM, and Random Forests—are described and compared. Experimental results demonstrate that the Random Forest classifier can achieve an accuracy of up to 94.66%.

1 Introduction

Robust terrain classification based on image features is essential for safe navigation in autonomous systems. This document provides an in-depth mathematical treatment of the feature extraction methods and classification techniques used in our approach. In what follows, we present detailed derivations of the image features and classifiers and include enhanced graphics to help visualize the key concepts.

2 Feature Extraction

The classification method relies on a feature vector

$$P = \{P_g, P_e, P_A, P_s, P_m\},$$

which is assembled from several image feature classes. In the subsections below, we detail the mathematics behind each class of features.

2.1 Multiscale Gray Gradient-Grade Features

For any pixel (u, v) in an image, the horizontal and vertical gradients are computed by

$$g_u(u, v) = f(u + 1, v) - f(u, v), \quad g_v(u, v) = f(u, v + 1) - f(u, v), \tag{1}$$

and the gradient magnitude is given by

$$g(u,v) = \sqrt{g_u^2(u,v) + g_v^2(u,v)}. \tag{2}$$

A window of size $n_i \times n_i$ is then applied to compute the gradient image. We set the thresholds as

$$th_{gj} = j \times d_g, \quad j = 1, 2, \ldots, 10, \tag{3}$$

where d_g is the spacing between adjacent gradient levels. The proportion of pixels whose gradient exceeds th_{gj} is

$$p_{gj} = \frac{N_{gj}}{n_i^2}, \tag{4}$$

with N_{gj} representing the number of pixels satisfying $g(u,v) > th_{gj}$. For three different scales, the multiscale gray gradient feature vector is constructed as

$$P_g = [P_g^{(1)}, P_g^{(2)}, P_g^{(3)}],$$

with each $P_g^{(i)} = [p_{g1}, p_{g2}, \ldots, p_{g10}]$.

2.1.1 Illustration of Discrete Gradient Computation

Figure 1 visualizes the computation of the horizontal and vertical gradients at a central pixel within a 3×3 image patch.

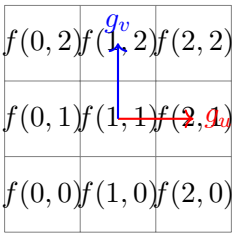

Figure 1: Discrete gradient computation at the central pixel $f(1,1)$ in a 3×3 patch. The red arrow represents the horizontal gradient g_u (with its label placed to the right) and the blue arrow represents the vertical gradient g_v (with its label placed above).

2.2 Multiscale Edge Strength-Grade Features

Using the Canny edge detection algorithm, edges are extracted at different strength levels. The gradient threshold for edge detection is defined by

$$th_{ej} = j \times d_e, \quad j = 1, 2, \ldots, 9, \tag{5}$$

where d_e is the spacing for edge strength levels. The proportion of edge pixels for a given threshold is calculated as

$$p_{ej} = \frac{N_{ej}}{n_i^2}, \tag{6}$$

with N_{ej} being the number of edge pixels with strength above th_{ej}. For multiple scales, the overall edge feature vector is

$$P_e = [P_e^{(1)}, P_e^{(2)}, P_e^{(3)}],$$

with each $P_e^{(i)} = [p_{e1}, p_{e2}, \ldots, p_{e9}]$.

2.3 Frequency Spectrum-Based Features

Three groups of features are extracted from the frequency domain to capture texture information.

2.3.1 Multiscale Frequency-Domain Mean Amplitude Features

For a window of size $n_i \times n_i$, the mean amplitude in the frequency domain is computed as

$$p_i^A = \frac{1}{n_i^2} \sum_{j=1}^{n_i} \sum_{k=1}^{n_i} A(u, v), \tag{7}$$

where $A(u, v)$ is the amplitude at the frequency coordinate (u, v). The multiscale feature vector is then

$$P_A = [p_1^A, p_2^A, p_3^A].$$

2.3.2 Multiscale Spectrum Symmetry Features

The frequency spectrum is divided into four quadrants, and symmetry is assessed by comparing statistical measures between opposing quadrants. Specifically, the mean values in different quadrants are given by

$$m_{i1}^F = \frac{1}{n_i^2} \sum_{x=0}^{n} \sum_{y=0}^{n} |F(u, v)|, \tag{8}$$

$$m_{i2}^F = \frac{1}{n_i^2} \sum_{x=-n}^{0} \sum_{y=0}^{n} |F(u, v)|, \tag{9}$$

$$m_{i4}^F = \frac{1}{n_i^2} \sum_{x=0}^{n} \sum_{y=-n}^{0} |F(u, v)|, \tag{10}$$

and the symmetry features are defined as

$$p_{Fx} = \left| m_{i1}^F - m_{i2}^F \right|, \quad p_{Fy} = \left| m_{i1}^F - m_{i4}^F \right|. \tag{11}$$

Analogously, the standard deviations in the quadrants are computed as

$$\sigma_{i1}^F = \sqrt{\frac{1}{n_i^2} \sum_{x=0}^{n} \sum_{y=0}^{n} \left(|F(u, v)| - m_{i1}^F \right)^2}, \tag{12}$$

$$\sigma_{i2}^F = \sqrt{\frac{1}{n_i^2} \sum_{x=-n}^{0} \sum_{y=0}^{n} \left(|F(u, v)| - m_{i2}^F \right)^2}, \tag{13}$$

$$\sigma_{i4}^F = \sqrt{\frac{1}{n_i^2} \sum_{x=0}^{n} \sum_{y=-n}^{0} \left(|F(u, v)| - m_{i4}^F \right)^2}. \tag{14}$$

The corresponding standard deviation symmetry features are

$$p_{\sigma x} = \left|\sigma_{i1}^F - \sigma_{i2}^F\right|, \quad p_{\sigma y} = \left|\sigma_{i1}^F - \sigma_{i4}^F\right|. \tag{15}$$

The overall multiscale spectrum symmetry feature vector is given by

$$P_s = \left[p_{Fx}^{(1)}, p_{Fx}^{(2)}, p_{Fx}^{(3)}, p_{Fy}^{(1)}, p_{Fy}^{(2)}, p_{Fy}^{(3)}, p_{\sigma x}^{(1)}, p_{\sigma x}^{(2)}, p_{\sigma x}^{(3)}, p_{\sigma y}^{(1)}, p_{\sigma y}^{(2)}, p_{\sigma y}^{(3)}\right].$$

2.3.3 Multiscale Spectrum Amplitude-Moment Features

The amplitude-moment feature measures the weighted contribution of the amplitude with respect to the distance from the center of the frequency spectrum. It is defined as

$$p_i^m = \frac{1}{n_i^2} \sum_{j=1}^{n_i} \sum_{k=1}^{n_i} A(u,v) \cdot d(u,v), \tag{16}$$

where $d(u,v)$ is the Euclidean distance from the point (u,v) to the spectrum center. The multiscale vector is then

$$P_m = [p_1^m, p_2^m, p_3^m].$$

2.3.4 Enhanced Visualization of the Frequency Spectrum

Figure 2 provides a more detailed schematic illustration of a typical frequency spectrum. The central red area represents dominant low-frequency components while peripheral areas illustrate high-frequency components.

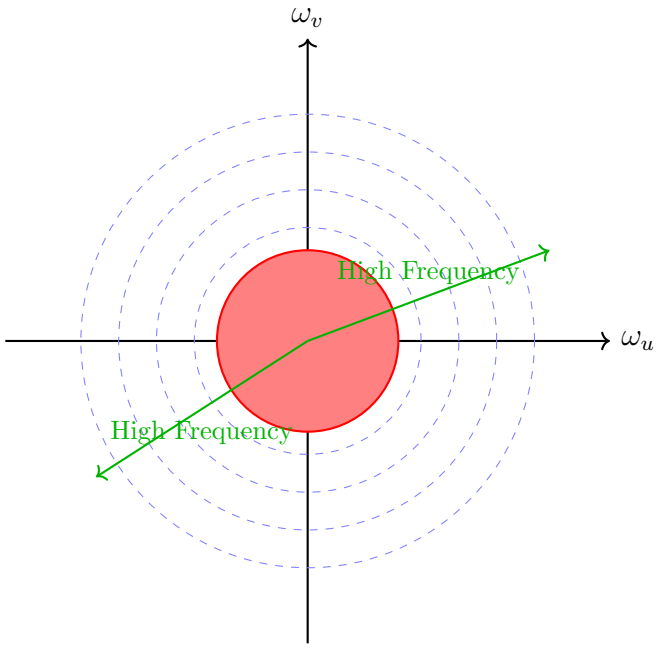

Figure 2: Enhanced frequency spectrum visualization. The red center indicates dominant low-frequency content.

3 Terrain Classification Methods

After assembling the feature vector P, classification is performed using three methods: KNN, SVM, and Random Forests.

3.1 K-Nearest Neighbor (KNN)

For feature vectors P_i and P_j, the Euclidean distance is calculated as

$$d(P_i, P_j) = \sqrt{\sum_{x=1}^{l} (p_{ix} - p_{jx})^2}. \tag{17}$$

For a sample P_a to be classified, the K nearest neighbors are determined and the class label is assigned by

$$c_{\text{knn}}(P_a) = \arg\max_{c_i \in \mathcal{C}} \sum_{j=1}^{K} \delta\Big(c_i, c(P_{\min j})\Big), \tag{18}$$

with

$$\delta(c_i, c(P_{\min j})) = \begin{cases} 1, & \text{if } c_i = c(P_{\min j}), \\ 0, & \text{otherwise.} \end{cases} \tag{19}$$

3.2 Support Vector Machine (SVM)

Given a training set $\{P_i, c_i\}$ with $P_i \in \mathbb{R}^n$ and $c_i \in \{-1, +1\}$, the optimal hyperplane is found by solving the convex optimization problem

$$\min_{w_h, t_h, \xi_i} \left(\frac{1}{2} \|w_h\|^2 + C_s \sum_{i=1}^{n_s} \xi_i \right) \tag{20}$$

subject to

$$c_i(w_h^T P_i + t_h) \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad i = 1, 2, \ldots, n_s. \tag{21}$$

Using Lagrange multipliers, the dual problem becomes

$$\max_{\alpha} \; W_S(\alpha) = \sum_{i=1}^{n_s} \alpha_i - \frac{1}{2} \sum_{i=1}^{n_s} \sum_{j=1}^{n_s} \alpha_i \alpha_j c_i c_j \, (P_i \cdot P_j) \tag{22}$$

subject to

$$\sum_{i=1}^{n_s} \alpha_i c_i = 0, \quad 0 \leq \alpha_i \leq C_s, \quad i = 1, \ldots, n_s. \tag{23}$$

Once the multipliers α_i are obtained, the weight vector is computed by

$$w_h = \sum_{i=1}^{n_s} \alpha_i c_i P_i, \tag{24}$$

and the bias t_h is derived (using any support vector P_j) as

$$t_h = c_j - \sum_{i=1}^{n_s} \alpha_i c_i \, (P_j \cdot P_i). \tag{25}$$

Thus, the SVM classifier for a sample P_a is given by

$$c_{\text{SVM}}(P_a) = \text{sgn}\left(\sum_{i=1}^{n_s} \alpha_i c_i \, (P_a \cdot P_i) + t_h \right). \tag{26}$$

For nonlinearly separable data, a kernel function $K_{\text{svm}}(P_a, P_i)$ is used:

$$c_{\text{SVM}}(P_a) = \text{sgn}\left(\sum_{i=1}^{n_s} \alpha_i c_i \, K_{\text{svm}}(P_a, P_i) + t_h \right). \tag{27}$$

3.2.1 Enhanced SVM Hyperplane Visualization

Figure 3 provides a more detailed schematic of the SVM hyperplane. Support vectors are highlighted (encircled in bold), and the margins are clearly marked.

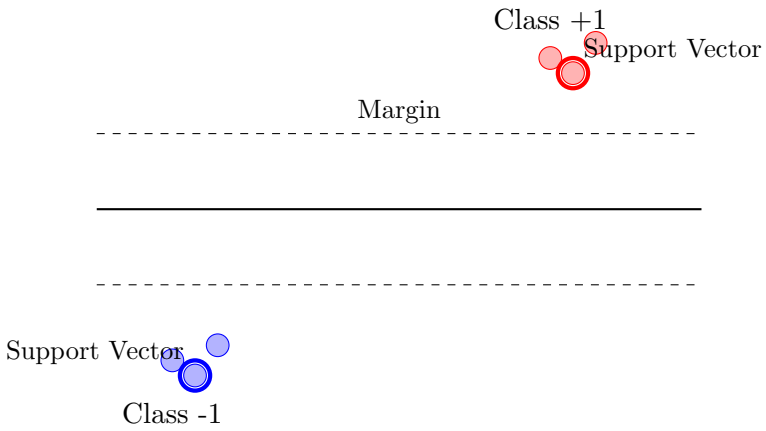

Figure 3: Enhanced SVM hyperplane with marked support vectors and margins.

4 Random Forests (RF)

Random Forests build an ensemble of m decision trees using bootstrapped subsamples and random feature selections. For a given sample P_a, each tree outputs a predicted class $c_j(P_a)$. The final classification is obtained by majority voting:

$$c_{\text{rf}}(P_a) = \arg\max_{c_i \in \mathcal{C}} \sum_{j=1}^{m} \delta\Big(c_i, c_j(P_a)\Big), \tag{28}$$

where the indicator function $\delta(\cdot)$ is defined as:

$$\delta\Big(c_i, c_j(P_a)\Big) = \begin{cases} 1, & \text{if } c_i = c_j(P_a), \\ 0, & \text{otherwise.} \end{cases}$$

4.1 Construction of the Decision Trees

Random Forests combine two key ideas:

1. **Bagging:** A bootstrapped subsample of size w is randomly selected from the training set to build each tree. This reduces the variance in the overall model.

2. **Random Feature Selection:** Instead of considering all features at each node, a random subset of t features is chosen. The best feature from this subset is used to split the node. This additional randomness decorrelates the trees, leading to a more robust ensemble.

Each base learner is typically a CART (Classification and Regression Tree) that is grown recursively until a stopping criterion (such as node purity or maximum depth) is met.

4.1.1 Sample Decision Tree Diagram

Figure 4 shows a simplified decision tree structure representative of one of the trees in a Random Forest:

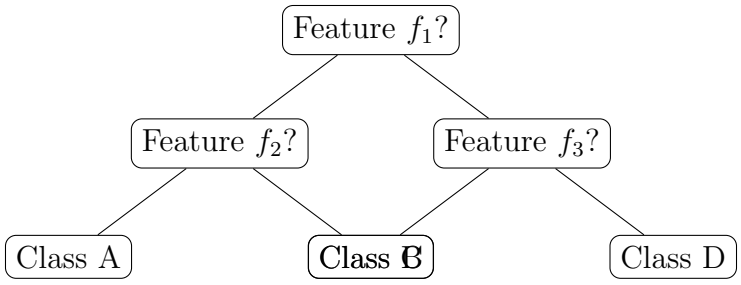

Figure 4: Simplified decision tree diagram representing a single tree in a Random Forest.

4.2 Prediction with Random Forests

For a new sample P_a, each of the m trees produces a class prediction $c_j(P_a)$. The final class is determined by majority voting:

$$c_{\text{rf}}(P_a) = \arg\max_{c_i \in \mathcal{C}} \sum_{j=1}^{m} \delta\Big(c_i, c_j(P_a)\Big).$$

That is, for every possible class $c_i \in \mathcal{C}$, the function $\delta(c_i, c_j(P_a))$ returns 1 if tree j votes for c_i and 0 otherwise. The class with the highest vote sum is selected as the output.

4.3 Advantages and Parameter Considerations

- **Reduced Overfitting:** By averaging the results of several trees, Random Forests generally achieve lower variance than individual decision trees.

- **Robustness:** The use of random subsamples and feature selection minimizes the impact of noise.

- **Parameter Tuning:** The key hyperparameters include:
  - m: Number of trees in the forest.
  - w: Size of the bootstrapped sample.
  - t: Number of randomly selected features at each node.

  A smaller w may reduce variance but can increase bias; therefore, these parameters are typically tuned via cross validation.

4.4 Application in Mars Terrain Classification

In the context of Mars terrain classification, the Random Forest (RF) classifier is used to distinguish among terrain types such as sandy terrain (ST), hard terrain (HT), and gravel terrain (GT). This is accomplished in two main stages: feature extraction and classification.

1. Feature Extraction

Each Mars terrain image is processed to extract a comprehensive feature vector P composed of multiple image characteristics. These features capture both spatial and frequency-domain information and are extracted over multiple scales (e.g., windows of size 5×5, 10×10, and 30×30).

a) Multiscale Gray Gradient-Grade Features (MSGGGFs): For a given grayscale image $f(u, v)$, the gradient at a pixel (u, v) is computed by

$$g_u(u, v) = f(u + 1, v) - f(u, v), \quad g_v(u, v) = f(u, v + 1) - f(u, v). \tag{29}$$

The gradient magnitude is then obtained as:

$$g(u, v) = \sqrt{g_u(u, v)^2 + g_v(u, v)^2}. \tag{30}$$

A threshold for the j-th gradient level is defined by

$$th_{gj} = j \times d_g, \tag{31}$$

where d_g is the gradient spacing parameter. In a window of size $n_i \times n_i$, the proportion of pixels with gradient magnitude exceeding th_{gj} is given by:

$$p_j^g = \frac{N_{gj}}{n_i^2}, \tag{32}$$

with N_{gj} being the number of pixels that satisfy $g(u, v) \geq th_{gj}$. For a fixed window scale, the feature vector is:

$$P_g = \left[p_1^g, p_2^g, \ldots, p_{10}^g\right].$$

Repeating this process over three different scales yields the multiscale gradient feature:

$$\mathbf{P}_g = \left[P_{g1}, P_{g2}, P_{g3}\right].$$

b) Multiscale Edge Strength-Grade Features (MSESGFs): Using an edge detector such as the Canny algorithm, edges are extracted from the image. For a threshold level defined as

$$th_{ej} = j \times d_e, \tag{33}$$

the proportion of edge pixels in a window is:

$$p_j^e = \frac{N_{ej}}{n_i^2}, \tag{34}$$

where N_{ej} is the count of edge pixels detected at threshold th_{ej}. Thus, for each window:

$$P_e = \left[p_1^e, p_2^e, \ldots, p_9^e\right],$$

and over multiple scales, the combined feature is:

$$\mathbf{P}_e = \left[P_{e1}, P_{e2}, P_{e3}\right].$$

c) Frequency-Domain Features: The image window is transformed into the frequency domain via a Fourier transform. Several frequency-based features are then computed:

i) Mean Amplitude Feature (MSFDMAF): The mean amplitude is calculated as:

$$p^A = \frac{1}{n_i^2} \sum_{j=1}^{n_i} \sum_{k=1}^{n_i} A(u, v), \tag{35}$$

where $A(u, v)$ is the amplitude at frequency coordinate (u, v). The multiscale vector is:

$$\mathbf{P}_A = \left[p_1^A, p_2^A, p_3^A\right].$$

ii) Spectrum Symmetry Features (MSSSFs): The Fourier spectrum is divided into quadrants, and symmetry is evaluated by comparing the mean amplitudes and standard deviations between corresponding quadrants. For example,

$$p^{F_x} = \left|m_{i1}^F - m_{i2}^F\right|, \quad p^{F_y} = \left|m_{i1}^F - m_{i4}^F\right|, \tag{36}$$

and similarly for the standard deviations,

$$p^{\sigma_x} = \left|\sigma_{i1}^F - \sigma_{i2}^F\right|, \quad p^{\sigma_y} = \left|\sigma_{i1}^F - \sigma_{i4}^F\right|. \tag{37}$$

These quantities form a vector:

$$P_s = \left[p_1^{F_x}, p_2^{F_x}, p_3^{F_x}, p_1^{F_y}, p_2^{F_y}, p_3^{F_y}, p_1^{\sigma_x}, p_2^{\sigma_x}, p_3^{\sigma_x}, p_1^{\sigma_y}, p_2^{\sigma_y}, p_3^{\sigma_y}\right].$$

iii) Spectrum Amplitude-Moment Features (MSSAMFs): The amplitude-moment, which weights the amplitude by the distance from the center of the spectrum, is computed as:

$$p^m = \frac{1}{n_i^2} \sum_{j=1}^{n_i} \sum_{k=1}^{n_i} A(u, v) \cdot d(u, v), \tag{38}$$

where $d(u, v)$ is the distance from the pixel (u, v) to the center of the Fourier spectrum. For multiple scales:

$$\mathbf{P}_m = \left[p_1^m, p_2^m, p_3^m\right].$$

d) Overall Feature Vector: All of the above features are concatenated to form the complete feature vector:

$$P = \{\mathbf{P}_g, \mathbf{P}_e, \mathbf{P}_A, \mathbf{P}_s, \mathbf{P}_m\}. \tag{39}$$

2. Classification with Random Forests

Once the feature vector P is extracted from each terrain image, it is fed into the Random Forest classifier. The RF classifier consists of m decision trees, each trained on a bootstrapped subset of the training data and using a random subset of features at each split.

For a new sample P_a, each tree j outputs a predicted class $c_j(P_a)$ (where the possible classes are ST, HT, and GT). The final decision is made by majority voting:

$$c_{\text{rf}}(P_a) = \arg\max_{c_i \in \mathcal{C}} \sum_{j=1}^{m} \delta\Big(c_i, c_j(P_a)\Big), \tag{40}$$

with the indicator function defined as:

$$\delta\Big(c_i, c_j(P_a)\Big) = \begin{cases} 1, & \text{if } c_i = c_j(P_a), \\ 0, & \text{otherwise.} \end{cases}$$

The above formulation ensures that the class which receives the highest number of votes across the m trees is selected as the final output.

3. Impact on Mars Rover Navigation

The high classification accuracy—reported as high as 94.66% :contentReference[oaicite:1]index=1—is crucial because it allows Mars rovers to:

- Reliably distinguish between safe terrain (HT) and hazardous terrain (ST and GT).
- Plan safe, traversable paths in real time, thereby avoiding risks such as wheel sinkage or damage.

In summary, the combination of detailed multiscale feature extraction (Equations (29)–(39)) with the ensemble voting strategy of the RF classifier (Equation (40)) provides a robust solution for the challenging problem of Mars terrain classification.

5 Experimental Verification and Discussion

Experiments were conducted on a dataset using feature extraction at three different scales (e.g., 5×5, 10×10, 30×30 windows). The performance of the three classifiers was compared. In our tests, the Random Forest classifier achieved a mean accuracy of 94.66% with a low misclassification rate for hazardous terrain.

Variable Definitions

Below is a complete list of the variables and symbols used throughout the paper along with their definitions:

- $f(u, v)$: Image intensity (or gray level) at pixel coordinates (u, v).
- u, v: Spatial coordinates in the image domain.
- $g_u(u, v)$: Horizontal gradient at pixel (u, v), computed as $f(u + 1, v) - f(u, v)$.
- $g_v(u, v)$: Vertical gradient at pixel (u, v), computed as $f(u, v + 1) - f(u, v)$.
- $g(u, v)$: Gradient magnitude at pixel (u, v), given by $\sqrt{g_u^2(u, v) + g_v^2(u, v)}$.
- n_i: Size of the $n_i \times n_i$ window used for feature extraction at scale i.
- th_{gj}: Gradient threshold for the jth level in the gray gradient feature, defined as $j \times d_g$ for $j = 1, 2, \ldots, 10$.
- d_g: Spacing between adjacent gradient levels.
- N_{gj}: Number of pixels in the window for which $g(u, v) > th_{gj}$.
- p_{gj}: Proportion of pixels with gradient magnitude greater than th_{gj}, computed as $\frac{N_{gj}}{n_i^2}$.
- $P_g^{(i)}$: Gray gradient feature vector at scale i, defined as $[p_{g1}, p_{g2}, \ldots, p_{g10}]$.
- P_g: Multiscale gray gradient feature vector, given by $[P_g^{(1)}, P_g^{(2)}, P_g^{(3)}]$.
- th_{ej}: Edge strength threshold for the jth level in the edge features, defined as $j \times d_e$ for $j = 1, 2, \ldots, 9$.
- d_e: Spacing between adjacent edge strength levels.
- N_{ej}: Number of edge pixels in the window with edge strength exceeding th_{ej}.
- p_{ej}: Proportion of pixels with edge strength above th_{ej}, computed as $\frac{N_{ej}}{n_i^2}$.
- $P_e^{(i)}$: Edge strength feature vector at scale i, defined as $[p_{e1}, p_{e2}, \ldots, p_{e9}]$.
- P_e: Multiscale edge strength feature vector, given by $[P_e^{(1)}, P_e^{(2)}, P_e^{(3)}]$.
- $A(u, v)$: Amplitude at the frequency coordinate (u, v) obtained from the Fourier transform.
- p_i^A: Mean amplitude feature for scale i, computed as $\frac{1}{n_i^2} \sum_{j=1}^{n_i} \sum_{k=1}^{n_i} A(u, v)$.
- P_A: Multiscale frequency-domain mean amplitude feature vector, defined as $[p_1^A, p_2^A, p_3^A]$.

- $F(u, v)$: Frequency spectrum (e.g., Fourier transform) of the image at coordinate (u, v).
- m_{i1}^F: Mean value of $|F(u, v)|$ in quadrant 1 for scale i, computed over the range $x, y = 0$ to n.
- m_{i2}^F: Mean value of $|F(u, v)|$ in quadrant 2 (typically corresponding to negative x-values) for scale i, computed over $x = -n$ to 0 and $y = 0$ to n.
- m_{i4}^F: Mean value of $|F(u, v)|$ in quadrant 4 (typically corresponding to negative y-values) for scale i, computed over $x = 0$ to n and $y = -n$ to 0.
- p_{Fx}: Spectrum symmetry feature along the x-direction, defined as $\left|m_{i1}^F - m_{i2}^F\right|$.
- p_{Fy}: Spectrum symmetry feature along the y-direction, defined as $\left|m_{i1}^F - m_{i4}^F\right|$.
- σ_{i1}^F: Standard deviation of $|F(u, v)|$ in quadrant 1 for scale i.
- σ_{i2}^F: Standard deviation of $|F(u, v)|$ in quadrant 2 for scale i.
- σ_{i4}^F: Standard deviation of $|F(u, v)|$ in quadrant 4 for scale i.
- $p_{\sigma x}$: Symmetry feature for standard deviation along the x-direction, computed as $\left|\sigma_{i1}^F - \sigma_{i2}^F\right|$.
- $p_{\sigma y}$: Symmetry feature for standard deviation along the y-direction, computed as $\left|\sigma_{i1}^F - \sigma_{i4}^F\right|$.
- P_s: Multiscale spectrum symmetry feature vector, given by

$$\left[p_{Fx}^{(1)}, p_{Fx}^{(2)}, p_{Fx}^{(3)}, p_{Fy}^{(1)}, p_{Fy}^{(2)}, p_{Fy}^{(3)}, p_{\sigma x}^{(1)}, p_{\sigma x}^{(2)}, p_{\sigma x}^{(3)}, p_{\sigma y}^{(1)}, p_{\sigma y}^{(2)}, p_{\sigma y}^{(3)}\right].$$

- $d(u, v)$: Euclidean distance from the frequency coordinate (u, v) to the center of the frequency spectrum.
- p_i^m: Amplitude-moment feature for scale i, computed as $\frac{1}{n_i^2} \sum_{j=1}^{n_i} \sum_{k=1}^{n_i} A(u, v) \cdot d(u, v)$.
- P_m: Multiscale spectrum amplitude-moment feature vector, defined as $[p_1^m, p_2^m, p_3^m]$.
- P_i, P_j: Feature vectors corresponding to samples i and j, respectively.
- l: The number of components (or dimensions) in a feature vector.
- $d(P_i, P_j)$: Euclidean distance between feature vectors P_i and P_j.
- P_a: The feature vector of a sample to be classified.
- K: Number of nearest neighbors considered in the KNN classifier.
- $c_{\mathbf{knn}}(P_a)$: Class label assigned to sample P_a by the KNN method.
- $\mathcal{C}$: The set of all possible class labels.

- $\delta(c_i, c(P_{\min j}))$: Indicator function that equals 1 if $c_i = c(P_{\min j})$ (i.e., if the class of the jth nearest neighbor is c_i) and 0 otherwise.
- $c(P_{\min j})$: The class label of the jth nearest neighbor to P_a.
- c_i: Class label associated with sample P_i, with $c_i \in \{-1, +1\}$ in the SVM context.
- n_s: The number of samples in the training set.
- w_h: Weight vector defining the SVM hyperplane.
- t_h: Bias term (offset) of the SVM hyperplane.
- ξ_i: Slack variable for sample i in the SVM formulation.
- C_s: Regularization (penalty) parameter in the SVM optimization problem.
- α_i: Lagrange multiplier corresponding to sample i in the SVM dual problem.
- $W_S(\alpha)$: The dual objective function in the SVM optimization.
- $K_{\mathbf{svm}}(P_a, P_i)$: Kernel function used in SVM for mapping input data to a higher-dimensional space when the data is not linearly separable.
- $c_{\mathbf{SVM}}(P_a)$: The class label assigned to sample P_a by the SVM classifier.
- $\text{sgn}(\cdot)$: The sign function, which returns +1 if its argument is positive and -1 if negative.
- m: The number of decision trees in the Random Forest ensemble.
- $c_j(P_a)$: The class prediction output by the jth decision tree for sample P_a.
- $c_{\mathbf{rf}}(P_a)$: The final class label for sample P_a as determined by majority voting in the Random Forest classifier.
- x, y: Indices used for summation over spatial or frequency domain coordinates in various equations.
- n: An index (or range limit) used in the summation for the frequency domain symmetry features (related to the window or quadrant size); note that in some equations n relates to the current window size and may be associated with n_i.

References

[1] Changela, H.; Chatzitheodoridis, E.; Antunes, A.; Beaty, D.; Bouw, K.; Bridges, J.; Hallsworth, J. *Int. J. Astrobiol.* **2022**, 21, 46.

[2] Ono, M.; Fuchs, T.J.; Steffy, A.; Maimone, M.; Yen, J. Risk-aware planetary rover operation: Autonomous terrain classification and path planning. In *Proc. IEEE Aerospace Conference*; 2015.