Terrain Classification Using a Radial Basis Function Network

[1]Tiny du Toit and [2]Hennie Kruger
Computer Science and Information Systems Department
North-West University
Potchefstroom, South Africa
e-mail: [1]Tiny.DuToit@nwu.ac.za, [2]Hennie.Kruger@nwu.ac.za

Abstract*—In this paper, inertial contact sensor based terrain classification is performed with a Radial basis function network (RBFN). Compared to the more popular Multilayer perceptrons, RBFNs are also intelligent techniques and universal approximators, but with a much simpler structure and shorter training time. It has been shown that RBFNs are efficient classifiers and consequently may be used for terrain classification. For the experiments, a mobile robot platform recorded vibration training data with an inertial measurement unit (IMU) while traversing five different terrains: asphalt, carpet, dirt, paving, and tiles. The composition of these terrains induces specific vibrations in the mobile platform which are measured by the IMU. The vibration signatures are comprised of the mobile robot's linear acceleration, orientation, and the earth's magnetic field. In contrast to most terrain classification techniques found in literature, no pre-processing of the data is performed. This reduces the computational overhead needed for real-time classification. A RBFN is then trained using a hybrid conjugate gradient descent method and *k*-fold cross-validation. Identification of the terrain is performed in real-time. The results are compared to those obtained by a Naïve Bayes method and a Support Vector Machine, which have also been successfully applied to terrain classification in literature. It was found that the RBFN outperformed these other techniques by a relatively large margin. Consequently, the RBFN with no pre-processing of the input data may be used as a contact sensor based terrain classification method.

Keywords–classification; IMU; inertial measurement unit; Radial basis function network; sensor; terrain classification.

I. INTRODUCTION

Mobile robots are employed in many different operational fields like supply and logistics, surveillance, search and rescue missions, agricultural applications, transportation, cleaning, inspection and entertainment [1][2]. For these operations, it may be necessary to traverse some indoor or off-road terrain, which might influence the vehicle's performance. The efficiency of these vehicles can be improved by detecting their environment. This act of identifying the type of terrain being traversed from a list of candidate terrains such as dirt, sand, or gravel, is called terrain classification [3].

It may be beneficial to identify the current terrain type as the terrain conditions may have an influence on both the motion control and planning stages of the vehicle's trip. Once the mobile robot's control system has knowledge of the surface it is travelling on, it will be easier to maneuver over uneven terrain or around obstacles. In addition, knowledge of the terrain will allow the vehicle to drive at higher speeds. By classifying the terrain, an automated driving process can be obtained which is terrain-dependent.

Research on the identification of terrain types can be divided into two groups: methods relying on noncontact sensors [3] - [6] and methods utilizing contact sensors [7] – [10]. Examples of noncontact sensors are vision sensors and laser scanners. A vision sensor like a charge-coupled device (CCD) camera uses techniques that extract textures and colors from the sensor data to classify it into variable terrains like forests and the sky. Unfortunately, the performance of these techniques is highly dependent on environmental factors like lighting conditions and climate effects and consequently the sensor information can be distorted. Laser scanner sensor data that are obtained from a terrain are converted into frequency information. Learning algorithms then use this information to classify the terrain. A disadvantage of such a method is that it needs numerous data points, which may hinder real-time classification.

Factors like friction, cohesion, damping, stiffness and surface irregularity comprise the terrain interface that is presented to the mobile robot [11]. As the mobile robot traverses the specific terrain, these terrain properties combined with the robot dynamics produce vibrational signatures in body motion. Methods based on contact sensors classify a terrain using sensor information like the vibration frequency or the slope ratio of the mobile robot's body into the terrain type. This enables the mobile robot to choose an appropriate driving mode, which allows the vehicle to traverse the terrain most effectively, prevents physical damage and keep wheels from sinking into the ground.

The goal of this paper is to perform terrain classification using a Radial basis function network (RBFN) as opposed to the well-known Multilayer perceptron (MLP) neural network, which has also been applied to this problem [12]. The MLP that is trained by the backpropagation rule is one of the most used and important neural network models [13]. Owing to its powerful universal approximation capability, the MLP is extensively used in classification, regression, prediction, system identification, control, feature extraction, and associative memory. Broomhead and Lowe [14] proposed the RBFN in 1988. This neural network has become a good alternative to the MLP, since it has equivalent capabilities as the MLP model, but can be trained much faster.

Previous studies have shown that RBFNs in general are efficient classifiers [1][15]. More specifically, in one study [1] a RBF network has been used for terrain classification where a Discrete Fourier transform was implemented to perform feature extraction. Unfortunately, such pre-processing of the data is a time-consuming task, which may prevent the real-time identification of the terrain. Although the aim of this paper is to investigate the feasibility of a RBFN to perform terrain classification, the results that are obtained will be compared to those achieved by the Naïve Bayes method and the Support Vector Machine (SVM) technique. These two models are also used for terrain classification in literature and the comparison will place the findings in the context of other popular techniques.

Terrain classification will be performed based on real-time vibration data obtained from an inertial measurement unit (IMU) contact sensor. No pre-processing, as reported in previous studies, of the data is performed. The assumption is that the output of the IMU sensor is influenced by the vibrations induced in the platform while traversing different terrains. The test vehicle, a Lego Mindstorms EV 3 mobile robot, is augmented by an IMU mounted on a Raspberry Pi 2 computer. Data that is collected from the IMU on the moving test vehicle is used as the terrain signature. This signature will then be classified by a trained RBFN as one of five predetermined terrains - asphalt, carpet, dirt, paving, or tiles.

The remainder of the paper is organized as follows. In Section II, the relatively simple structure and training of the RBFN will be discussed. A variant of the gradient descent method is used for training. Experiments performed to determine the accuracy of terrain classification using a RBFN will be considered in Section III. The results that were obtained will be examined in Section IV. Finally, some concluding remarks will be presented in Section V.

II. Radial Basis Function Networks

In this section, the RBFN architecture and training of the model will be considered.

A. Architecture

A RBFN is a feedforward neural network with three layers $(J_1 - J_2 - J_3)$ [15] – [17] as shown in Figure 1. In the input, hidden and output layers there are J_1, J_2 and J_3 neurons respectively. The bias in the output layer is denoted by $\phi_0(\vec{x}) = 1$ while the nonlinearity at the hidden nodes is denoted by the $\phi_k(\vec{x})$'s. Each hidden layer node uses a Radial basis function (RBF), denoted by $\phi(r)$ for its nonlinear activation function. The hidden layer performs a nonlinear transformation of the input. This nonlinearity is then mapped into a new space by the output layer, which acts as a linear combiner. Normally, all hidden nodes utilize the same RBF; the RBF nodes have the nonlinearity $\phi_k(\vec{x}) = \phi(\vec{x} - \vec{c}_k), k = 1, \dots, J_2$, where $\vec{c}_k$ denotes the center or prototype of the kth node and $\phi(\vec{x})$ is an RBF. An extra neuron in the hidden layer can model the biases of the output layer neurons. This neuron has a constant activation function $\phi_0(r) = 1$. The RBFN determines a global optimal solution for the adjustable weights in the minimum mean square error (MSE) sense by using the method of linear optimization. The output of the RBF network, provided by input $\vec{x}$, is given by

$$y_i(\vec{x}) = \sum_{k=1}^{J_2} w_{ki}\phi(\|\vec{x} - \vec{c}_k\|), i = 1, \dots, J_3, \quad (1)$$

where $y_i(\vec{x})$ is the ith output, w_{ki} denotes the connection weight from the kth hidden neuron to the ith output unit, and $\|\cdot\|$ is the Euclidian norm. The RBF usually utilizes the Gaussian function $\phi(\cdot)$ and such a model is normally called the Gaussian RBF network.

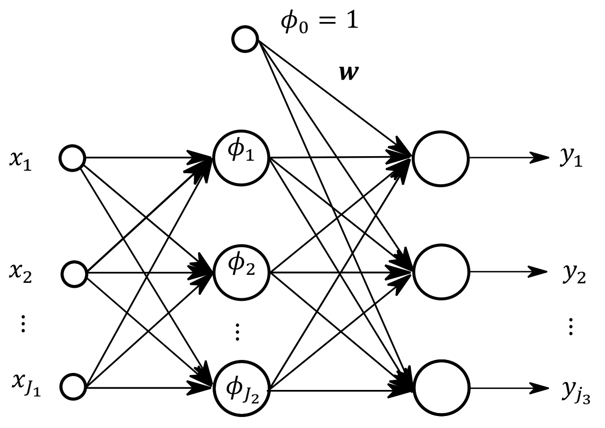

Figure 1. RBF network architecture [16].

Given a set of N pattern pairs $\{(\vec{x}_p, \vec{y}_p) | p = 1, \dots, N\}$, (1) can be expressed in matrix form as

$$\boldsymbol{Y} = W^T \Phi \quad (2)$$

where $\boldsymbol{W} = [w_1, \dots, w_{J_3}]$ is a $J_2 \times J_3$ matrix, $\vec{w}_i = (w_{1i}, \dots, w_{J_2 i})^T$, $\Phi = [\vec{\phi}_1, \dots, \vec{\phi}_N]$ is a $J_2 \times N$ matrix, $\vec{\phi}_p = (\phi_{p,1}, \dots, \phi_{p,J_2})^T$ is the hidden layer output for the pth sample, specifically, $\phi_{p,k} = \phi(\|\vec{x}_p - \vec{c}_k\|)$, $\boldsymbol{Y} = [y_1\ y_2\ \dots y_N]$ is a $J_3 \times N$ matrix, and $\vec{y}_p = (y_{p,1}, \dots, y_{p,J_3})^T$.

The RBFN is a universal approximator [16]. If the RBF is appropriately chosen, the RBF network can theoretically approximate any continuous function arbitrarily well. The Gaussian RBF is expressed as $\phi(r) = \exp(-r^2/2\sigma^2)$ where $r > 0$ represents the distance from a data point $\vec{x}$ to a center $\vec{c}$ and σ is utilized to control the smoothness of the interpolating function. The Gaussian RBF is a localized RBF with the property that $\phi(r) \to 0$ as $r \to \infty$.

Training of an RBFN is usually performed by a two-phase strategy. During the first phase, suitable centers $\vec{c}_k$ and their corresponding standard deviations, σ_k, also known as widths or radii are determined. The network weights $\boldsymbol{W}$ are adjusted in the second phase. The training approach that is followed in this research is the supervised learning of all the parameters by the relatively simple gradient descent method.

B. *Training*

There is one output unit for each of the five terrain class values (asphalt, carpet, dirt, paving, and tiles). The model trained for the *i*th output unit (class value) is given by:

$$y_i(x_1, x_2, \dots, x_m) = g\left(w_{i,0} + \sum_{k=1}^{b} w_{i,k} \exp\left(-\sum_{j=1}^{m} \frac{\left(x_j - c_k\right)^2}{2\sigma_{global}^2}\right)\right), \tag{3}$$

where the activation function $g(\cdot)$ is a logistic function [18]. A Gaussian RBF network with the same global variance parameter σ_{global} for all RBF centers still has universal approximation capability [16]. The appropriate parameter values for $w_{i,k}$ and σ_{global} are found by identifying a local minimum of the penalized squared error on the training data. Given p classes, the error function can be expressed as

$$L_{SSE} = \left(\frac{1}{2}\sum_{k=1}^{n}\sum_{i=1}^{p}\left(y_{k,i} - f_i(\vec{x}_k)\right)^2\right) + \left(\lambda \sum_{i=1}^{p}\sum_{k=1}^{b} w_{i,k}^2\right), \tag{4}$$

where $y_{k,i} = 0.99$ if data point $\vec{x}_i$ has the *i*th class value, and $y_{k,i} = 0.01$ otherwise. Instead of using 1.0 and 0.0, the values 0.99 and 0.01 are used to aid the optimization process. Additionally, in (4), L_{SSE}, is divided by n, the number of training data points, as this was determined through empirical observation to improve convergence with the optimization methods used [19]. Standard calculus is utilized to find the corresponding partial derivatives, which is comprised of the gradients of the error function with respect to the network parameters. Backpropagation is employed to calculate the partial derivatives in the same manner as in Multilayer perceptrons. The hybrid conjugate gradient descent method specified by [20] is used for training.

Before training starts, all numeric inputs in the data are normalized to the [0, 1] interval. This data are transformed back into the original space when predictions are produced. The mode (for nominal attributes) or the mean (for numeric ones) is used to impute missing values. Additionally, nominal attributes are binarized and constant attributes are removed. These same transformations are performed for new inputs when the predictions are made.

Initialization of the network parameters is another important aspect of the training procedure. The initial weights of the output layer are sampled from $\mathcal{N}(0, 0.1)$. This strategy was empirically determined based on the familiar heuristic of choosing small, randomly distributed initial weights [19].

As the *k*-means algorithm is often used to train the hidden layer of the RBFN in an unsupervised process, it is utilized to determine the initial hidden unit centers c_k. Furthermore, the initial value of the variance parameter σ_{global} is set to the maximum squared Euclidian distance between any pair of cluster centers. This ensures that the initial value of the variance parameter is not too small.

The learning process is accelerated on a multi-core computer by parallelizing the calculation of the error function and its gradient on a user-specified number of threads.

In the next section, the experiments that are performed to determine the RBFN terrain classification accuracy will be discussed.

III. EXPERIMENTAL DESIGN

The goal of the experiments is to identify the type of terrain being traveled on by a mobile robot, from a list of candidate terrains. Figure 2 shows the Lego Mindstorms EV3 experimental platform used in the investigation. The mobile robot has a Raspberry Pi 2 computer attached to the front with a Sense HAT inertial measurement unit (IMU) in turn connected to the Raspberry Pi. The Sense HAT is readily available and includes the following sensors: A gyroscope, an accelerometer, and a magnetometer. The mobile robot platform is battery powered and moves on rubber treads. An additional battery pack (not shown) is mounted on top and powers the Raspberry Pi computer. The five terrain types used in the study are displayed in Figures 3 to 7.

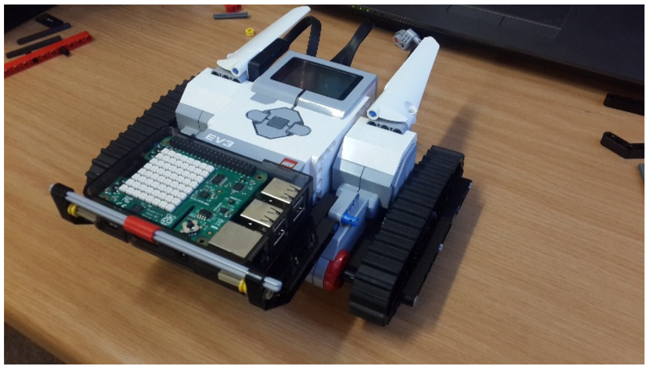

Figure 2. Lego Mindstorms EV 3 mobile robot.

The terrain (asphalt, carpet, dirt, paving, or tiles) on which the mobile robot is currently travelling is identified in real-time. The assumption is that the vibrations induced in the test vehicle and measured by the output of the IMU sensor represent a signature, which can be used to accurately classify the terrains. The data for each terrain is sampled at an irregular rate of $\approx 16\frac{2}{3}$ Hz for a 600 second duration. The RBFN is then trained offline using the RBFN training scheme discussed in Section II (B). Three outdoor terrains (asphalt, dirt, and paving) and two indoor terrains (carpet and tiles) were analyzed.

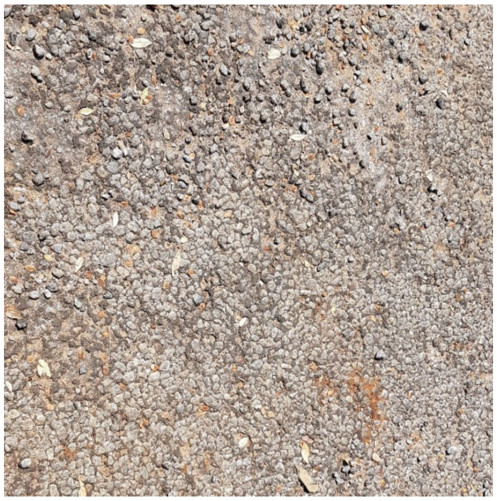
Figure 3. Asphalt.

Figure 4. Carpet.

The RBFN architecture for this specific problem has five outputs that serve to identify the terrain type. Each of the output values $y_i \in [0,1]$ denotes the likelihood that a given signal presented as an input to the RBFN matches one of the five candidate terrains. In addition, the RBFN architecture has twelve inputs, which correspond to the dimension of the input signal data point. Each of these input signal data points received from the Sense HAT IMU can be denoted as:

$$[p\ r\ y\ a_x\ a_y\ a_z\ g_x\ g_y\ g_z\ m_x\ m_y\ m_z],$$

where $p, r,$ and y denote the pitch, roll and yaw (measured in degrees), a is the linear acceleration (m/s^2) measured in three dimensions (a_x, a_y and a_z), g is the rate of turn (degrees/second), also measured in three dimensions (g_x, g_y and g_z) and m denotes the earth's magnetic field (gauss), measured in three dimensions (m_x, m_y and m_z) of the mobile robot respectively.

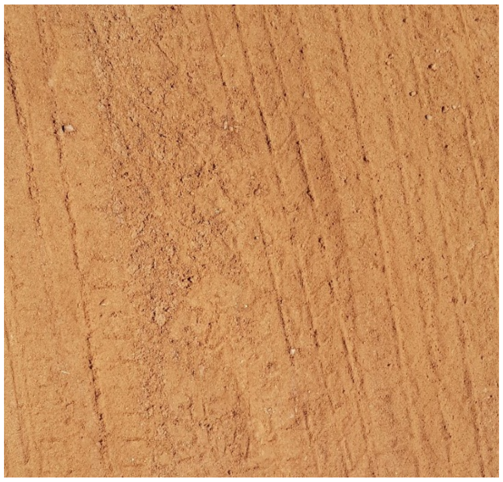
Figure 5. Dirt.

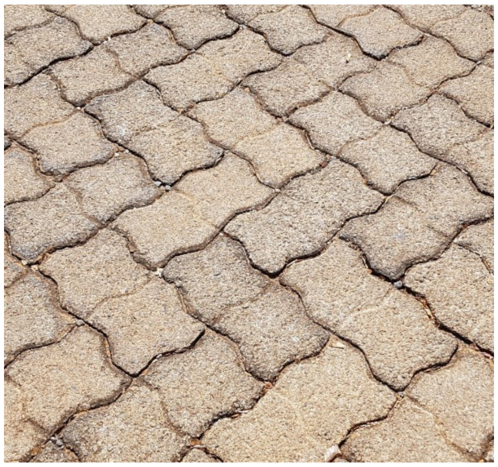
Figure 6. Paving.

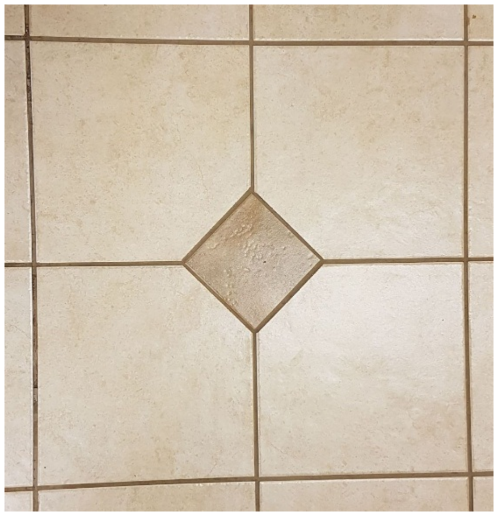
Figure 7. Tiles.

The Weka system [19] was used for data processing, presentation, classifier training and testing. The terrain classification training dataset contained twelve inputs, five outputs and a total of 49993 samples. For the experiments, 10-fold cross-validation was performed. Results obtained by the RBFN were compared to those found by a SVM model and a Naïve Bayes technique, which are two popular methods found in the literature used for supervised terrain classification [9][10]. In the following section, the results will be discussed.

IV. DISCUSSION

The classification accuracy results obtained by the experiments are shown in Figure 8.

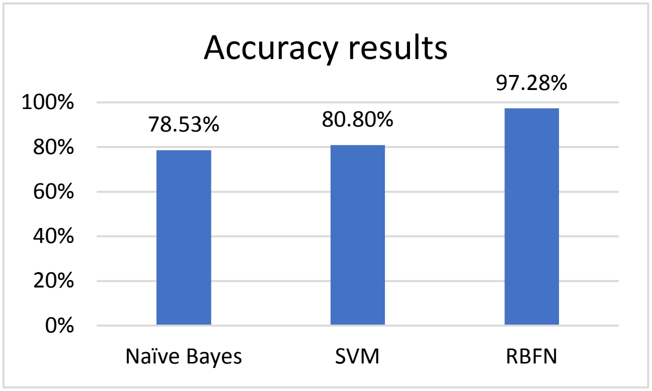

Figure 8. Terrain classification results.

From Figure 8 it can be observed that the machine learning algorithms, ordered from best to worst, are the RBFN, SVM and Naïve Bayes. The latter two techniques produced nearly the same classification accuracy. These results show that the RBFN is a feasible terrain classification technique compared to the other two models and may outperform these techniques by a relatively large margin. This is a promising result as no pre-processing has been performed on the training data.

To summarize, the RBFN applied to terrain classification has the following advantages:

- Compared to the MLP, the RBFN has less model complexity, exhibit better comprehensibility and is easier to construct due to its simpler structure.
- No pre-processing of the input data is performed like in previous studies.
- Classification of the terrain can be performed in real-time because of the onboard IMU contact sensor.
- In terms of predictive accuracy, the RBFN outperformed the Naïve Bayes technique and the SVM model.

Based on these findings, the RBFN is without doubt a technique to consider for terrain classification.

V. CONCLUSION

In this paper, real-time classification of five given terrains was performed with a RBFN. In contrast to other techniques found in the literature, no pre-processing of the mobile robot platform's IMU vibration sensor data was performed. Eliminating feature extraction reduces the computational overhead needed to identify the terrain in real-time. The results have shown that even without feature extraction, the RBFN is still a feasible model for contact sensor based terrain classification compared to other popular models used for this task. It can be used as an alternative to the MLP model due to its simpler structure and shorter training times. The RBFN has the capability to accurately recognize complex vibration signature patterns and can easily adapt to new terrain signatures by providing the model with new training examples. Unfortunately, compared to the other techniques, offline training of the model can be time consuming.

Future work includes a comparison between the RBFN and MLP models to determine if the RBFN model outperforms the MLP model in terms of terrain classification accuracy. Also, a more detailed comparison with the existing methods must be performed. Metrics like latency (velocity) can be included in the results. Finally, it can be determined if the technique can be applied to other types of robots and how they must be adapted for this task.

ACKNOWLEDGMENT

The authors would like to thank Mr. Ryno Marx for assembling the mobile robot platform and for acquiring the vibration sensor data for the five terrains.

REFERENCES

[1] T. Kurban and E. Besdok, "A Comparison of RBF neural network training algorithms for inertial sensor based terrain classification," Sensors, vol. 9, 2009, pp. 6312—6329.
[2] D. Sadhukhan, "Autonomous ground vehicle terrain classification using internal sensors," Florida State University, Master's thesis, 2004.
[3] L. Ojeda, J. Borenstein, G. Witus, and R. Karlsen, "Terrain characterization and classification with a mobile robot," Journal of Field Robotics, vol. 23(2), 2006, pp. 103—122.
[4] A. Angelova, L. Matthies, D. Helmick, and P. Perona, "Fast terrain classification using variable-length representation for autonomous navigation," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Minneapolis, MN, USA, 2007, pp. 1-8.
[5] A. Talukder et al., "Autonomous terrain characterization and modelling for dynamic control of unmanned vehicles," in Proceedings of the IEEE Conference on Intelligent Robots and Systems (IROS), Lausanne, Switzerland, 2002.
[6] R. Manduchi, A. Castano, A. Talukder, and L. Matthies, "Obstacle detection and terrain classification for autonomous off-road navigation," Autonomous Robots, vol. 18, 2005, pp. 81–102.

[7] B. Park, J. Kim, and J. Lee, “Terrain feature extraction and classification for mobile robots utilizing contact sensors on rough terrain,” Procedia Engineering, vol. 41, 2012, pp. 846-853.

[8] R. Jitpakdee and T. Maneewarn, “Neural networks terrain classification using inertial measurement unit for an autonomous vehicle,” SICE Annual Conference, The University Electro-Communications, Japan, 2008.

[9] C. C. Ward and K. Iagnemma, “Speed-independent vibration-based terrain classification for passenger vehicles,” Vehicle System Dynamics, vol. 47, no. 9, 2009, pp. 1095–1113.

[10] M. Happold, M. Ollis, and N. Johnson, “Enhancing supervised terrain classification with predictive unsupervised learning,” Robotics: Science and Systems II, University of Pennsylvania Philadelphia, 2006.

[11] F. L. Garcia Bermudez, R. C. Julian, D. W. Haldane, P. Abbeel, and R. S. Fearing, “Performance analysis and terrain classification for a legged robot over rough terrain,” “IEEE/RSJ International Conference on Intelligent Robots and Systems”, Vilamoura, Algarve, Portugal, October 7-12, 2012.

[12] T. Y. Kim, G. Y. Sung, and J. Lyou, “Robust terrain classification by introducing environmental sensors,” IEEE International Workshop on Safety Security and Rescue Robotics (SSRR), 2010.

[13] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, “Learning internal representations by error propagation,” in Parallel Distributed Processing: Explorations in the Microstructure of Cognition, D. E. Rumelhart and J. L. McClelland, Eds., vol. 1, pp. 318–362, MIT Press, Cambridge, Mass, USA, 1986.

[14] D. S. Broomhead and D. Lowe, “Multivariable functional interpolation and adaptive networks,” Complex Systems, vol. 2, no. 3, 1988, pp. 321–355.

[15] C. S. K. Dash, A. K. Behera, S. Dehuri, and S.-B. Cho, “Radial basis function neural networks: a topical state-of-the-art survey,” Open Computer Science, 6(1), 2016, pp. 33–63.

[16] Y. Wu, H. Wang, B. Zhang, and K.-L. Du. “Using radial basis function networks for function approximation and classification,” International Scholarly Research Network, Applied Mathematics, Volume 2012, doi:10.5402/2012/324194.

[17] H. B. Demuth, M. H. Beale, O. De Jess, and M. T. Hagan, “Neural network design,” 2nd edition, Martin Hagan, USA, 2014.

[18] E. Frank, “Fully supervised training of gaussian Radial basis function networks in WEKA,” [Online]. http://www.cs.waikato.ac.nz/~ml/publications/2014/rbf_networks_in_weka_description.pdf 2017.03.09.

[19] E. Frank, M. A. Hall, and I. H. Witten, “The WEKA workbench. Online appendix for ‘Data mining: Practical machine learning tools and techniques’,” Morgan Kaufmann, Fourth Edition, 2016.

[20] Y. H. Dai and Y. Yuan, “An efficient hybrid conjugate gradient method for unconstrained optimization,” Annals of Operations Research, 103, 2001, pp. 33-47.