This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.



# Ensemble Learning With Weak Classifiers for Fast and Reliable Unknown Terrain Classification Using Mobile Robots

Ayan Dutta and Prithviraj Dasgupta

***Abstract*—We propose a lightweight and fast learning algorithm for classifying the features of an unknown terrain that a robot is navigating in. Most of the existing research on unknown terrain classification by mobile robots relies on a single powerful classifier to correctly identify the terrain using sensor data from a single sensor like laser or camera. In contrast, our proposed approach uses multiple modalities of sensed data and multiple, weak but less-complex classifiers for classifying the terrain types. The classifiers are combined using an ensemble learning algorithm to improve the algorithm's training rate as compared to an individual classifier. Our algorithm was tested with data collected by navigating a four-wheeled, autonomous robot, called Explorer, over different terrains including brick, grass, rock, sand, and concrete. Our results show that our proposed approach performs better with up to 63% better prediction accuracy for some terrains as compared to a support vector machine (SVM)-based learning technique that uses sensor data from a single sensor. Despite using multiple classifiers, our algorithm takes only a fraction (1/65) of the time on average, as compared to the SVM technique.**

***Index Terms*—Ensemble learning, mobile robot, terrain classification.**

## I. Introduction

As humans, we can adjust our walking style or gait to suit the terrain that we are walking on. Similarly, while driving a car, we adjust our car's speed and turning angles according to the terrain. For example, while driving a car on an icy road, we usually reduce the speed of the car and turn slowly than normal, non-icy roads. We are in an era, when autonomous ground vehicles and robots are increasingly used for tasks like surveillance, information collection, extra-terrestrial exploration, and many more. These autonomous vehicles also need to adapt their navigation style (such as gait, speed, etc.) depending on the terrain they are navigating in. Therefore, correctly identifying terrain is a very important task for the robots to navigate efficiently. Terrain classification refers to the task of correctly identifying the terrain, such as grass, sand, bricks, etc., using the robot's on-board sensors. Research on unknown terrain classification has gained popularity since the DARPA grand challenge in 2006 [1] and NASA's Mars Exploration Rovers' autonomous navigation on Mars [2]. Most of the previous researchers have used a single sensor's data and a single powerful classification algorithm for unknown terrain classification. Unlike previous works, in this paper, we have used different types of sensed data and multiple classifiers together for the prediction in a layered fashion.

Over the last decade, many researchers have proposed solutions for unknown terrain classification using different sensors on robots, such as, camera, laser, inertial measurement unit (IMU), vibration sensor, etc. [3], [4]. Different machine learning techniques including support vector machine (SVM) [4], neural networks [3], and $K$-nearest neighbors (KNNs) [5] have been employed to learn patterns from this data and classify the type of terrain. Woods *et al.* [5] have proposed an approach which uses multiple weak classifiers like KNN and decision trees to correctly classify the current terrain type of the robot from the robot's camera data. This paper also uses a similar approach, where we use an ensemble learning technique, which consists of weak classifiers, such as KNN and Naive Bayes (NB). Angelova *et al.* [6] have proposed a learning technique, which learns the model using multiple features of any particular terrain, such as average color, color histogram, and texture. Häselich [7] have proposed a Markov random field-based probabilistic terrain classification approach, which uses the combined 3-D laser data and the images of the terrain. Similarly many researchers use visual data from camera for classifying the terrain [6], [8], [9]. But, in these techniques, successful classification using camera data is susceptible to ambient light conditions. Recently, Filitchkin and Byl [10] have proposed a visual sensor based classification approach which is more robust to different illumination conditions.

Our ensemble learning technique is applied to different types of collected data, such as acceleration, angular rates, and roll-pitch-yaw (RPY). Ojeda *et al.* [3] have used similar sensors, such as gyroscope, accelerometer, encoder, as well as motor current and voltage sensors. Their neural network based learning technique learns one sensor data model at a time and predicts the terrain type from that sensory input data. Brooks and Iagnemma [4] have used the vibration sensor of the robot for prediction of terrain type. The pairwise classifier,

Manuscript received September 7, 2015; revised December 4, 2015; accepted January 21, 2016. This paper was recommended by Associate Editor S. Nahavandi.

The authors are with the Department of Computer Science, University of Nebraska at Omaha, Omaha, NE 68182 USA (e-mail: adutta@unomaha.edu; pdasgupta@unomaha.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSMC.2016.2531700

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.



proposed in [4], uses a voting schema to correctly identify the terrain.

Other sensors, such as LiDAR, have also been used for terrain classification [11]. McDaniel *et al.* [12] have proposed terrain classification using a tree stem identification algorithm, while using the robot's ground-based LiDAR data along with an SVM classifier. The magnitude of the spatial frequency response received by the robot on different terrains, are essentially different which has been exploited in [13] for classification.

Papadakis [14] surveyed the terrain traversability analysis methods proposed by a number of researchers. Under traversability, researchers have looked at terrainability as well. Terrainability refers to the perception and characterization of the current terrain. This survey shows that most of the works on terrainability can be divided into two main categories: 1) geometry-based and 2) appearance-based. In the geometry-based approach, terrain features are realized by the robot using signals received by its on-board sensors, such as IMU and encoder; while interacting with the particular terrain [11], [15]. On the other hand, for appearance-based terrainability prediction mostly depends on the visual sensor data and vision features of the terrain [6], [16]. Bellutta *et al.* [17] and Manduchi *et al.* [18] have proposed terrainability assessment techniques, which use hybrid methodologies, based on fusion of the geometry-based and the appearance-based techniques.

SVM is a very powerful classifier that has been widely used for terrain classification [4], [12]. Although the SVM classifiers are highly accurate due to their ability to model complex, nonlinear decision boundaries, the training time of even the fastest SVMs can be considerably large [19]. For this reason, implementing an SVM classifier onboard of the robot to classify the terrain online, might not be feasible for many robots. On the other hand, if weak classifiers are used, although the classification accuracy might not be as good as SVM, still a reasonable amount of prediction accuracy can be achieved with very low computational complexity. A weak classifier's individual prediction accuracy is inferior to SVM's accuracy, but when multiple of them are used together, the collection of weak classifiers can significantly improve the prediction accuracy [19]. Decoste and Schölkopf [20] have shown that for U.S. Postal Service data set SVM and KNN's performance were at par. It has also been shown that using a set of classifiers together using the ensemble learning methods (e.g., bagging and boosting [21]), performs better than using a single classifier [19], and, the bagging technique performs significantly better than its constituting classifiers even with noisy data.

To the best of our knowledge, the proposed multilayer learning technique, where multiple sensor data and multiple weak classifiers are used simultaneously on a test data set, to correctly identify terrain has not been previously used. Unlike most of the previous works in this domain, our hypothesis is that every type of sensor is not useful/appropriate for classifying all types of terrains. One sensor data might be very sensitive to terrain changes, whereas other sensors might not change significantly. Therefore, to capture the changes in the terrain types, one particular sensor might not be enough. Similar research ideas of using multiple sensor data for achieving a single task, can be observed in autonomous cars, where multiple sensors, such as laser, camera images, are used to detect obstacles [22].

### A. Our Contribution

The main hypothesis, which led us to work on this topic is that not all sensors (nonvisual) are useful for correctly identifying all types of terrains, and, moreover not all classifiers are good in classifying all sorts of sensor data. Based on this hypothesis, in this paper, we specifically ask the following research questions.

1) Can we use multiple low cost (in terms of run time required) classifiers to reduce the time and the use of computational resources, instead of using a single but computationally expensive SVM classifier?
2) Can a set of low cost classifiers, working together in a bagging fashion [23], achieve comparable level of accuracy as an SVM classifier while requiring lower computational resources?
3) Can we use set of sensor data for classification, which do not necessarily depend largely on external factors, like vision techniques do on illumination condition of the environment?
4) What are the performance benefits of combining results from multiple classifiers, as shown in [24]?

To answer these questions, we propose a new algorithm, which combines multiple, low time-consuming classification algorithms, such as KNN and NB, and also merge the classification results from different sensors using these classification algorithms, to reach a decision. We extensively analyze the performances of each and every classifier and sensor, when used individually, and, also when used with other classifiers and sensors. Our results show that our hypothesis is correct, i.e., some sensors are good in identifying certain particular terrains, while other sensors might be better in identifying a different type of terrain. Also, we have proved empirically that a similar level of performance accuracy as SVM can be achieved, if we use multiple, low resource-consuming classifiers while simultaneously reducing the classification time.

This paper is structured as follows. In the next section, we describe the model that we have used and then we describe the proposed algorithm. In Section III, we describe the experimental platform we have used, including the description of the robot and the sensors used, and then we discuss the results. Finally in Section IV, we discuss our findings and conclude this paper.

## II. Problem Setup and Algorithm

Suppose a robot has a set of $k$ on-board sensors $\mathcal{S} = \{s_1, s_2, \ldots, s_k\}$. Example sensors include wheel encoders, accelerometer, gyroscope, laser range finders, camera, etc. We select a set of sensors $S \subset \mathcal{S}$, whose collected data will be used for classification. We also have a set of $l$ candidate terrains $T = \{t_1, t_2, \ldots, t_l\}$. For training the classification algorithm, the robot is run through all the terrains, and all sensors' data are collected.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.



### A. K-Nearest Neighbor

In KNN classification, we look at $K$ nearest data points of a novel test data point, $\mathcal{X}$, in the training data set. An output class label is assigned to the class which has maximum number of occurrences in that $K$ data points [25]. Voronoi tessalation or KD-Tree can be used for partitioning the training data set points and for faster classification [26]. In this paper, we have used two different values of $K$, 5 (KNN5) and 10 (KNN10).

For the KNN classifier, we have used Mahalanobis distance (MD) as the distance metric [27] as it is more suited for data sets with unknown distribution, especially, when data along different dimensions could be distributed differently [28]. MD between two data points $\mathcal{X}$ and $\mathcal{X}'$ is defined as

$$\mathrm{MD}(\mathcal{X}, \mathcal{X}') = (\mathcal{X} - \mathcal{X}')^T \Sigma^{-1} (\mathcal{X} - \mathcal{X}') \tag{1}$$

where $\Sigma$ is the covariance matrix of the inputs (across all classes).

### B. Naive Bayes

In NB, we form a belief network using the training data $X$ and its class label $l$. This belief network can be mathematically written as (derived from [26])

$$P(X, l) = P(l)\Pi_{i=1}^{D} P(x_i|l). \tag{2}$$

Next, Bayes' theorem is applied to predict a class label for test data $\mathcal{X}$, as follows:

$$P(l, \mathcal{X}) = \frac{P(\mathcal{X}|l)P(l)}{P(\mathcal{X})} = \frac{P(\mathcal{X}|l)P(l)}{\sum_l P(\mathcal{X}|l)P(l)}. \tag{3}$$

### C. Bagging Predictors

Our proposed strategy is based on the ensemble learning method, called bagging [21], [23]. In bagging, multiple inaccurate or weak classifiers, $C$, [29] are used as a combined classifier. Each combined classifier gives an unweighted vote to a predicted class label $l$. The class level with the maximum number of votes wins at the end. Our proposed approach uses a modified bagging technique, where not only we have a bag of classifiers, but we also have a set of sensors, both of which are used to reach a decision on the terrain type.

### D. Our Approach

We use a modified version of the bagging algorithm, where we use multiple sensors and multiple classifiers to reach a decision about the terrain type. The process is shown in Algorithm 1. Each test data set, $D_S = \{D_{s_1}, D_{s_2}, \ldots, D_{s_j}\}$, consists of one data member from each sensor $s_i \in S$. Each member of every test data, $D_{s_i}$ is passed separately through a set of classifiers, $C = \{C_1, C_2, \ldots, C_m\}$. Each classifier classifies each $D_{s_i}$ and the classified (or predicted) terrain type wins 1 point. This process is repeated for all data members in test data $D_S$. Finally, the terrain which wins the most number of points, because it got selected as predicted class by most classifiers and for most of the sensor data, is declared as the winner. If multiple terrains win the same number of points, then the winner is undecided and another test data will be needed to identify the winning terrain correctly.

**Algorithm 1:** Terrain Classification Algorithm

*classifyTerrain()*
**Input**: $D_S$: A set of data consisting data from multiple sources.
**Output**: $t_{winner}$: Classified terrain of $data_i$.
$T$: Set of candidate terrains.
$C$: Set of classifiers used;
Points terrain $t_i$ collects, $point_{t_i} \leftarrow 0, \forall t_i \in T$.
$t_{winner} \leftarrow undecided$.
**for** *each* $D_{s_i} \in D_S$ **do**
  **for** *each* $c_k \in C$, *applied on* $D_{s_i}$ **do**
    $t_{win} \in T$: Winner terrain.
    $point_{t_{win}} \leftarrow point_{t_{win}} + 1$; // $t_{win} \in T$.
**if** $point_{t_x} > point_{t_y}, \forall t_y \in T \setminus \{t_x\}$ **then**
  $t_{winner} \leftarrow t_x$;
**else**
  $t_{winner}$ is undecided;
return $t_{winner}$;

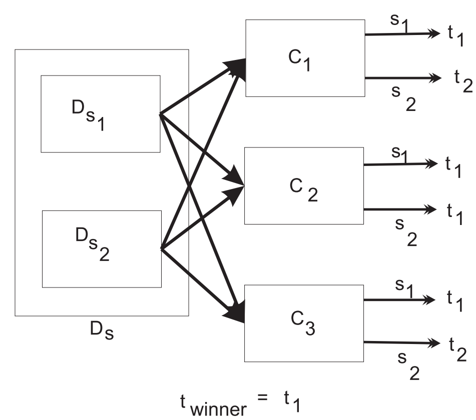



Fig. 1. Illustrative example of the proposed bagging approach.

An example of our proposed bagging algorithm is shown in Fig. 1, where a test data set $D_s$ consisting of two different sensors' data, $D_{s_1}$ and $D_{s_2}$, are passed through three classifiers, $C_1$–$C_3$. Terrain type $t_1$ is the winner four times out of six times while terrain type $t_2$ is the winner only twice. Therefore, for data set $D_s$, $t_1$ is selected as the winner.

In this paper, we have used three classifiers for implementing our algorithm—KNN5, KNN10, and NB. We have also used four types of sensed data—IMU, angular rates, acceleration, and RPY. The algorithm works as following: a test data is made consisting of one data point from each of the four data sources. Now each one of these four data points for that particular test data are passed through KNN5, KNN10, and NB classifiers. Each of the classifiers classifies the four data points, giving a total of $3 \times 4 = 12$ predictions. After all the predictions are done, the highest point winning terrain is declared as the winner.

## III. Experimental Evaluation

### A. Setup

An Explorer robot, made by Coroware IT Solutions, has been used for data collection and testing purposes. The robot

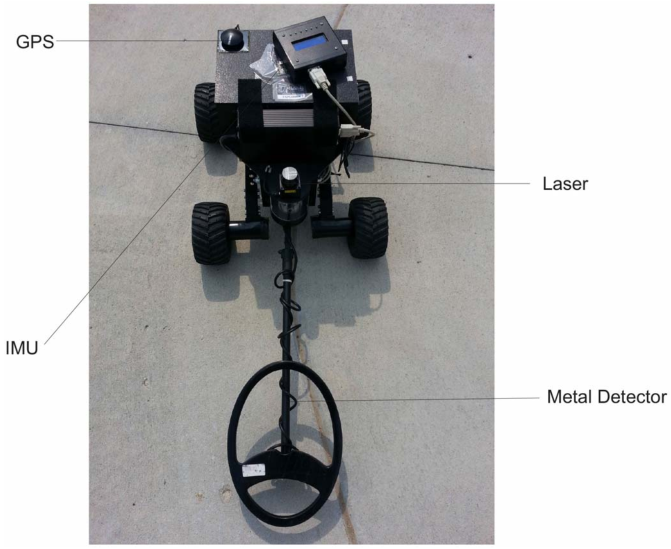



Fig. 2. Different sensors of the Explorer robot.

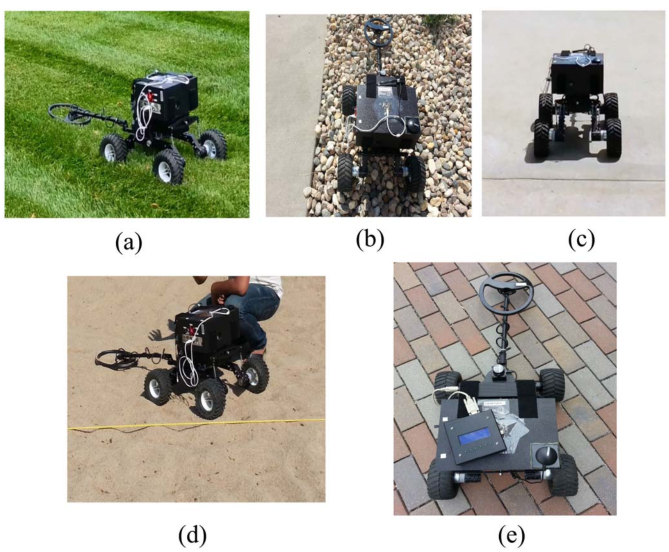



Fig. 3. Sample terrain patches. (a) Grass. (b) Rock. (c) Concrete. (d) Sand. (e) Brick.

is 61 cm long and has a width of 53 cm (Fig. 2). The robot has four skid steer wheels—it can maneuver on rugged terrains. The robot is equipped with a Garmin GPS and Hokuyo laser range finder (range = 5 m). The Explorer robot is also equipped with an IMU sensor (Phidget Spatial 1056_0) which provides acceleration, angular rate and magnetic field strength measurements along three axes. The robot is run on five different terrains (brick, concrete, grass, sand, and rock) and data from all the above mentioned sensors is collected. Fig. 3(a)–(e) show the sample patch of terrains that we have used for testing our algorithm. Two-thirds of the collected data are used for training and rest of the data are used for testing using the hold-out method [30]. Hundred random training data points of acceleration, RPY and angular rates are shown as a 3-D plot in Fig. 4. Fig. 4 shows that both acceleration and RPY data points are visibly different for different terrains, whereas angular rates data for different terrains are more similar in nature. The feature vector size of IMU sensor data is 9, whereas the feature vector sizes of acceleration, angular rate and RPY data is 3 each. First the classifiers are trained with the training data sets. Next, for the measurement of the classification accuracy, each test data point in the test data set is passed through the classifier. Each test data point in the test data set gets an associated classification label (i.e., terrain type) after it is passed through the classifier. Once all the test data points in a particular test data set are passed though the classifier and the classified terrain types are determined, then the percentage of classification accuracy is measured for that particular test data set.

Similar to the approach described in [10], we did not use Explorer robot's on-board computer for calculations. Rather we have set up a communication system using WiFi between the robot and a desktop computer. The desktop computer has an Intel i7 CPU with 12 GB of RAM. Classifiers were implemented in MATLAB. This setup can also be easily transitioned to the robot's on-board processor [10].

### B. Experimental Results

In this section, we first discuss and analyze the performance of each classifier used, combined with each data type. Next, we provide the results for using multiple classifiers on different types of data. After that, we analyze the effect of combining multiple types of data along with individual and multiple classifiers. Finally, we compare the performance of our technique against a classical SVM approach.

*1) Analysis of Performance of Individual Classifiers Along With Single Sensor:* Fig. 5 shows the confusion matrices for IMU, acceleration, RPY, and angular rate data sets, while using KNN5 classifier. Fig. 5(a) shows the confusion matrix for IMU sensor. Rows and columns of confusion matrix denote ground truth terrain types and predicted terrain types, respectively. As can be seen in this figure, for brick, grass, and rock, the prediction accuracy is more than 95%. But this accuracy falls in case of sand terrain to about 76%. Sand terrain is detected 16.83% times as brick, whereas for brick terrain test cases, it is never detected as sand terrain. Concrete terrain has been correctly detected in 82.25% of the test cases, whereas it is detected as brick for 13.75% of the test cases. The prediction accuracy of concrete terrain significantly increases if we use only acceleration or RPY data. It can be noticed in Fig. 5(b) and (c) that the prediction accuracy of concrete terrain increases to 88.5% and 98.5%, i.e., a significant jump of 6.25% and 16.25% for these two sensors, respectively. But on the other hand, using acceleration as the only sensed data leads to around 12% and 5% fall in prediction accuracy for brick and grass terrains, respectively. Significant change in prediction accuracy happens in case of RPY data, while detecting grass test samples. From the prediction accuracy of more than 90%, a significant fall to 31% can be observed. Prediction accuracy of both rock and sand samples remain almost similar for acceleration and RPY data. Accuracy measurements drop significantly while using angular rate data. Using this data, rock can be detected correctly for most number of times, while the sand is correctly detected for a mere 4.95% times.

Next, we test the performance of the KNN10 classifier along with individual sensors, on different terrains (Fig. 6). Accuracy

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.



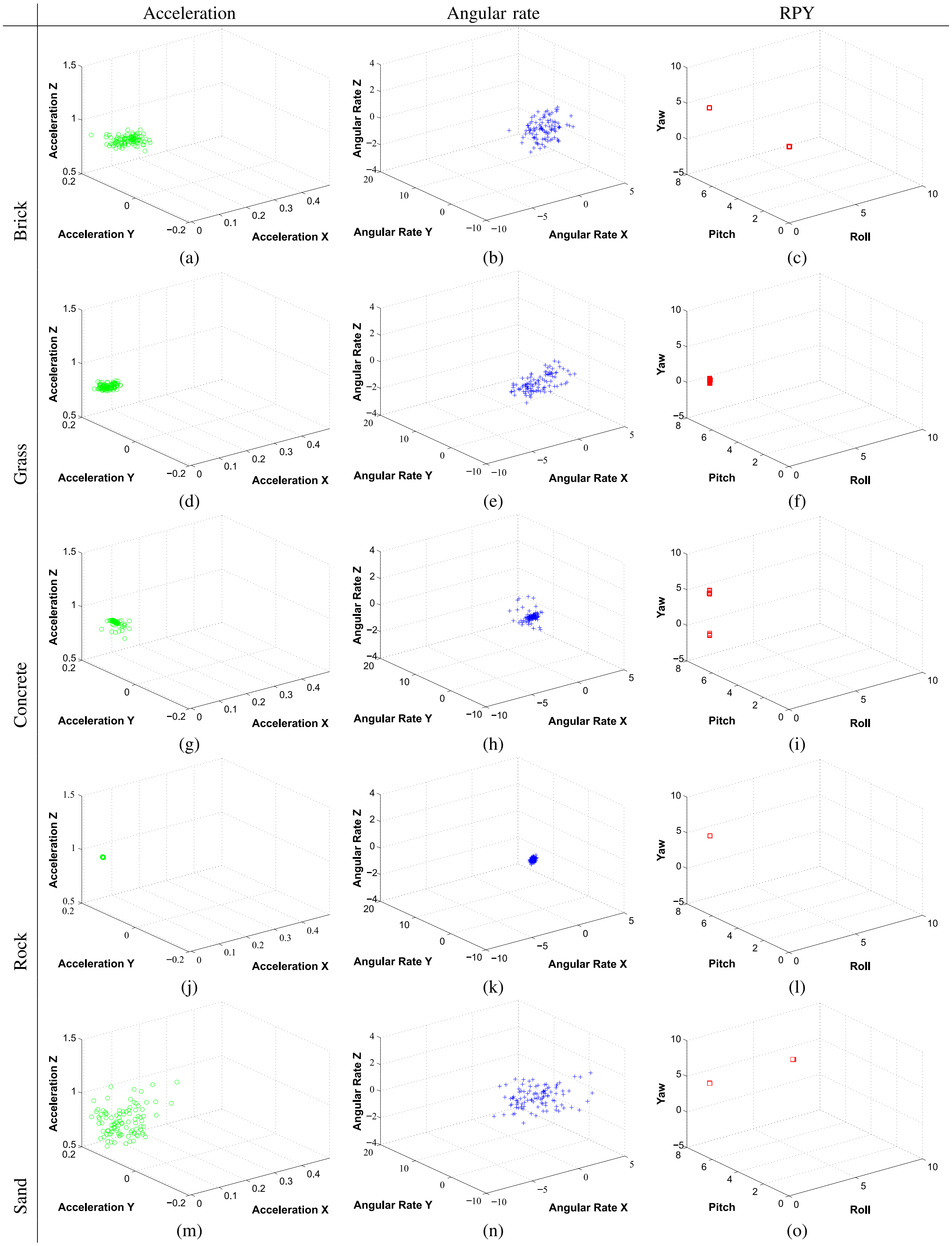



Fig. 4. Sample data of acceleration, angular rates, and RPY collected on different terrains. (a)–(c) Brick. (d)–(f) Grass. (g)–(i) Concrete. (j)–(l) Rock. (m)–(o) Sand.

measurements are similar to KNN5 classifier across different terrains and sensors, though there are few noticeable changes. Using IMU data, KNN10's classification accuracy for sand terrain drops by nearly 5%. On the other hand, for acceleration data, KNN10's performance is better than KNN5 classifier for brick (+3%), grass (+0.75%), and concrete (+0.50%)

| | brick | grass | rock | sand | concrete | undecided |
|---|---|---|---|---|---|---|
| brick | 95.02 | 0 | 0 | 0 | 4.97 | 0 |
| grass | 0 | 97 | 0 | 0 | 3 | 0 |
| rock | 0 | 0 | 100 | 0 | 0 | 0 |
| sand | 16.83 | 1.98 | 0 | 76.23 | 4.95 | 0 |
| concrete | 13.75 | 2.75 | 1 | 0.25 | 82.25 | 0 |

(a)

| | brick | grass | rock | sand | concrete | undecided |
|---|---|---|---|---|---|---|
| brick | 73.13 | 0 | 0 | 0.49 | 26.36 | 0 |
| grass | 2 | 92 | 0 | 0.50 | 5.25 | 0 |
| rock | 0 | 0 | 100 | 0 | 0 | 0 |
| sand | 6.93 | 17.82 | 0 | 68.31 | 6.93 | 0 |
| concrete | 7 | 3.50 | 0.75 | 0.25 | 88.50 | 0 |

(b)

| | brick | grass | rock | sand | concrete | undecided |
|---|---|---|---|---|---|---|
| brick | 100 | 0 | 0 | 0 | 0 | 0 |
| grass | 0 | 31 | 0 | 0 | 69 | 0 |
| rock | 0 | 0 | 100 | 0 | 0 | 0 |
| sand | 28.71 | 0 | 0 | 71.28 | 0 | 0 |
| concrete | 1 | 0.50 | 0 | 0 | 98.50 | 0 |

(c)

| | brick | grass | rock | sand | concrete | undecided |
|---|---|---|---|---|---|---|
| brick | 49.75 | 18.90 | 1.49 | 0 | 29.85 | 0 |
| grass | 12.50 | 50.50 | 9.75 | 0 | 27.25 | 0 |
| rock | 0 | 16.83 | 62.37 | 0 | 20.79 | 0 |
| sand | 32.67 | 17.82 | 0 | 4.95 | 44.55 | 0 |
| concrete | 15 | 36.50 | 21.75 | 1.25 | 25.50 | 0 |

(d)

Fig. 5. Confusion matrix (in %) using KNN5 classifier for (a) IMU, (b) acceleration, (c) RPY, and (d) angular rate data.

| | brick | grass | rock | sand | concrete | undecided |
|---|---|---|---|---|---|---|
| brick | 95.02 | 0 | 0 | 0 | 4.97 | 0 |
| grass | 0 | 97 | 0 | 0 | 3 | 0 |
| rock | 0 | 0 | 100 | 0 | 0 | 0 |
| sand | 18.81 | 4.95 | 0 | 71.28 | 4.95 | 0 |
| concrete | 13.25 | 2.75 | 4 | 0 | 80 | 0 |

(a)

| | brick | grass | rock | sand | concrete | undecided |
|---|---|---|---|---|---|---|
| brick | 76.11 | 0 | 0 | 0.49 | 23.38 | 0 |
| grass | 2.50 | 92.75 | 0.25 | 0.50 | 4 | 0 |
| rock | 0 | 0 | 100 | 0 | 0 | 0 |
| sand | 6.93 | 20.79 | 0 | 65.34 | 6.93 | 0 |
| concrete | 6 | 4.50 | 0 | 0 | 88.75 | 0 |

(b)

| | brick | grass | rock | sand | concrete | undecided |
|---|---|---|---|---|---|---|
| brick | 100 | 0 | 0 | 0 | 0 | 0 |
| grass | 0 | 30.50 | 0 | 0 | 69.50 | 0 |
| rock | 0 | 0 | 100 | 0 | 0 | 0 |
| sand | 28.71 | 0 | 0 | 71.28 | 0 | 0 |
| concrete | 1.75 | 0.50 | 0 | 0 | 97.75 | 0 |

(c)

| | brick | grass | rock | sand | concrete | undecided |
|---|---|---|---|---|---|---|
| brick | 45.77 | 18.90 | 2.48 | 0 | 32.83 | 0 |
| grass | 12.25 | 53.75 | 8.50 | 0 | 25.50 | 0 |
| rock | 0 | 11.88 | 73.26 | 0 | 14.85 | 0 |
| sand | 33.66 | 13.86 | 0 | 2.97 | 49.50 | 0 |
| concrete | 13.75 | 36 | 24 | 1.75 | 25.25 | 0 |

(d)

Fig. 6. Confusion matrix (in %) using KNN10 classifier for (a) IMU, (b) acceleration, (c) RPY, and (d) angular rate data.

terrains. RPY data's effect for correctly classifying the terrains using KNN10, does not vary much from KNN5's performance. Angular rate data's performance also remains poor. It is worthy to note that, using angular rate data along with KNN10 classifier increased the prediction accuracy of sand's test samples by almost 11%.

While applying NB classifier on different sensor data, the results varied significantly (Fig. 7). For example, with IMU data, brick terrain's prediction accuracy goes up to 100%—a 5% increment in accuracy result from the KNN5 and KNN10 classifiers, whereas the grass terrain samples' prediction accuracy goes down to 82.25% from 97%. More significantly, prediction accuracy for grass terrain, using NB classifier drops to almost 72% from KNN5 and KNN10's prediction accuracy measurements. Classification using NB classifier along with RPY data did not have a significant effect, except from the fact that prediction accuracy for grass terrain increases by almost 58% to reach 78%. This finding is significant, as this shows that even though using NB classifier along with acceleration data might not appear to be a very attractive option, yet we can achieve much higher prediction accuracy if we use RPY data instead of acceleration data along with the NB classifier, for detecting grass terrain. Angular rate data's performance did not improve from the previous classifiers' results, except that the rock terrain samples were correctly detected almost every time.

*2) Analysis of Performance of Multiple Classifiers Along With Multiple Sensors:* Fig. 8 shows the confusion matrices for using the three classifiers together on different sensor data. The result we have got in this section corroborates with the findings in [31], where the authors have shown that even though the bagging algorithm's performance is better than its constituting learning algorithms, if we use KNN and NB algorithms, then the performance change is very little than when we use other weaker algorithms, such as decision trees [32]. As SVM is more powerful classifier than both KNN and NB, therefore bagging SVM with other more powerful algorithms, such as multilayer perception (MLP) [33] can gain higher precision than bagging KNN and NB. But, at the same time we should remember that using more powerful algorithms such as MLP and SVM would require considerably more computational resources. Liang *et al.* [31] have also showed empirically that even though bagging KNN and NB does not change the result significantly, but it can still achieve better results than using a single SVM classifier.

In Fig. 9, we summarize the performances of each individual terrain and sensor data. Fig. 9(a) shows the best, worst, and the average performances of each individual terrain type, using only one classifier at a time. This figure shows that even if for every terrain type, the best performance reaches almost 100%, except from the sand terrain (near 80%), the average performances vary greatly. For example, for brick

| | brick | grass | rock | sand | concrete | undecided |
|---|---|---|---|---|---|---|
| brick | 100 | 0 | 0 | 0 | 0 | 0 |
| grass | 0 | 82.25 | 0 | 0 | 17.75 | 0 |
| rock | 0 | 0 | 100 | 0 | 0 | 0 |
| sand | 28.71 | 0 | 0 | 71.28 | 0 | 0 |
| concrete | 12.5 | 0.25 | 0 | 0 | 87.25 | 0 |

(a)

| | brick | grass | rock | sand | concrete | undecided |
|---|---|---|---|---|---|---|
| brick | 69.15 | 0 | 0 | 0 | 30.84 | 0 |
| grass | 0 | 16.50 | 0 | 0.75 | 82.75 | 0 |
| rock | 0 | 0 | 97.02 | 0 | 2.97 | 0 |
| sand | 4.95 | 11.88 | 0 | 74.25 | 8.91 | 0 |
| concrete | 4.5 | 2.75 | 0 | 0.25 | 92.50 | 0 |

(b)

| | brick | grass | rock | sand | concrete | undecided |
|---|---|---|---|---|---|---|
| brick | 100 | 0 | 0 | 0 | 0 | 0 |
| grass | 0 | 78 | 0 | 0 | 22 | 0 |
| rock | 0 | 0 | 100 | 0 | 0 | 0 |
| sand | 28.71 | 0 | 0 | 71.28 | 0 | 0 |
| concrete | 9.50 | 0 | 0 | 0 | 90.50 | 0 |

(c)

| | brick | grass | rock | sand | concrete | undecided |
|---|---|---|---|---|---|---|
| brick | 25.87 | 18.90 | 2.48 | 0 | 52.73 | 0 |
| grass | 8.25 | 29 | 57 | 0.25 | 5.50 | 0 |
| rock | 0 | 0 | 99.00 | 0 | 0.99 | 0 |
| sand | 20.79 | 14.85 | 0 | 5.94 | 58.41 | 0 |
| concrete | 7.25 | 26 | 47.25 | 4.5 | 15 | 0 |

(d)

Fig. 7. Confusion matrix (in %) using NB classifier for (a) IMU, (b) acceleration, (c) RPY, and (d) angular rate data.

| | brick | grass | rock | sand | concrete | undecided |
|---|---|---|---|---|---|---|
| brick | 95.52 | 0 | 0 | 0 | 4.47 | 0 |
| grass | 0 | 97 | 0 | 0 | 3 | 0 |
| rock | 0 | 0 | 100 | 0 | 0 | 0 |
| sand | 19.80 | 1.98 | 0 | 74.25 | 3.96 | 0 |
| concrete | 12.50 | 2.50 | 1 | 0 | 83.75 | 0 |

(a)

| | brick | grass | rock | sand | concrete | undecided |
|---|---|---|---|---|---|---|
| brick | 73.63 | 0 | 0 | 0.49 | 25.87 | 0 |
| grass | 1 | 91.75 | 0.25 | 0.5 | 5.75 | 0.75 |
| rock | 0 | 0 | 100 | 0 | 0 | 0 |
| sand | 6.93 | 17.82 | 0.75 | 68.31 | 6.93 | 0 |
| concrete | 4.50 | 4 | 0.75 | 0 | 90.50 | 0.25 |

(b)

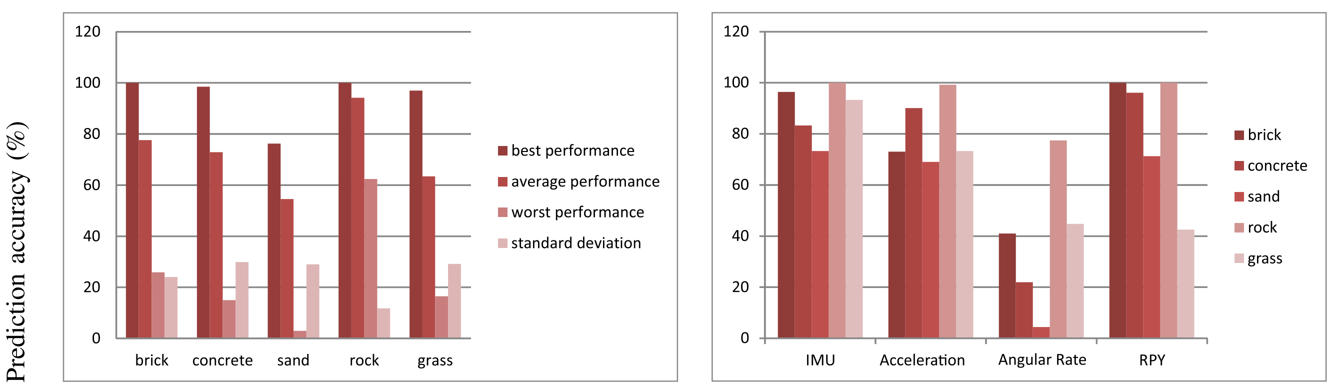
| | brick | grass | rock | sand | concrete | undecided |
|---|---|---|---|---|---|---|
| brick | 100 | 0 | 0 | 0 | 0 | 0 |
| grass | 0 | 30.50 | 0 | 0 | 69.50 | 0 |
| rock | 0 | 0 | 100 | 0 | 0 | 0 |
| sand | 28.71 | 0 | 0 | 71.28 | 0 | 0 |
| concrete | 1.75 | 0.50 | 0 | 0 | 97.75 | 0 |

(c)

| | brick | grass | rock | sand | concrete | undecided |
|---|---|---|---|---|---|---|
| brick | 42.78 | 17.41 | 1.99 | 0 | 34.32 | 3.48 |
| grass | 11.75 | 45.75 | 11 | 0 | 19.50 | 12 |
| rock | 0 | 9.90 | 75.24 | 0 | 10.89 | 3.96 |
| sand | 28.71 | 13.86 | 0 | 3.96 | 52.47 | 0.99 |
| concrete | 12 | 31 | 26 | 1 | 22 | 8 |

(d)

Fig. 8. Confusion matrix (in %) using all three classifiers for (a) IMU, (b) acceleration, (c) RPY, and (d) angular rate data.

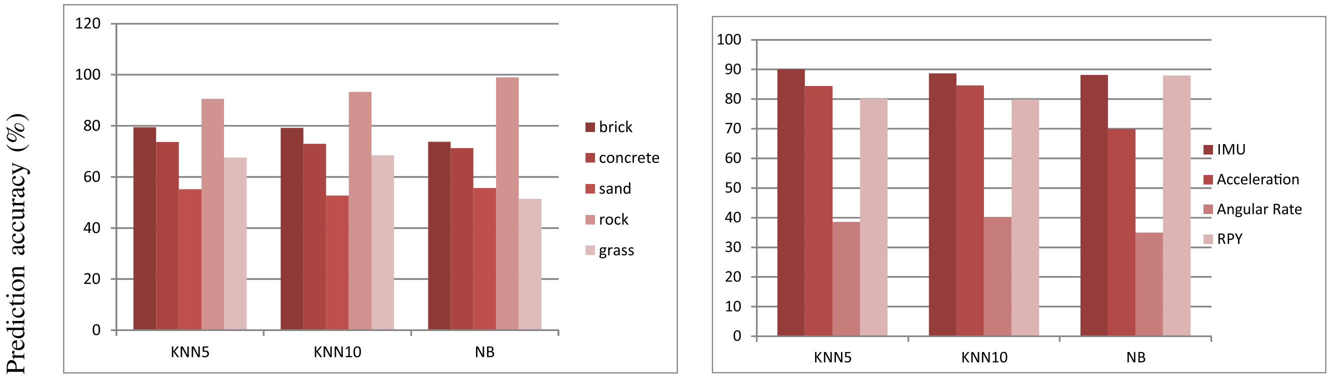



(a) (b)

Fig. 9. Summarized performances of prediction accuracy for (a) terrains and (b) sensor data.

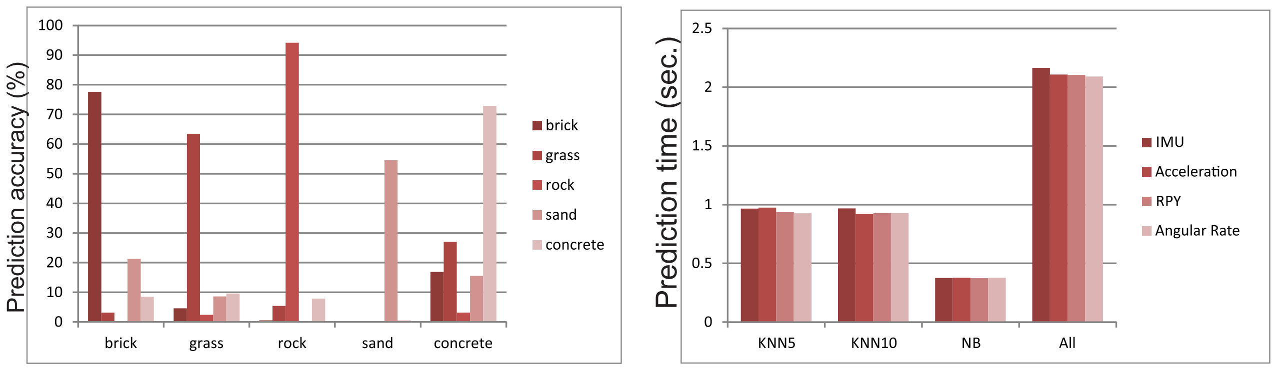
and concrete terrains, the average prediction accuracy is near 80%, whereas for grass it is below 20% and for sand it is near 60%. Prediction accuracy of rock terrain is most consistent, with the best performance being 100% and the average being over 95%. We believe that the main reason behind rock terrain's better performance is that the sensors recorded a completely different data set while on rock terrain, because of its unique geometric characteristics. Previous researchers also found rock (or gravel) [3] to be one of the best performing terrains. Other terrains that we have used in our tests are mostly flat in nature, whereas the rock terrain has unique nonflat characteristics, for which sensor signatures are also significantly different from other terrains. Therefore, across all classifiers, the prediction accuracy of rock terrain remains high.

Fig. 9(b) shows the average performance of prediction accuracy of different sensor data on different terrains. One can easily note that the angular rate data is the worst performer among all types of data. For all types of terrains, angular rate data performed very poorly. The best performance of the




(a) (b)

Fig. 10. Summarized performances of classifiers against different (a) terrains and (b) sensor data.




(a) (b)

Fig. 11. (a) Misclassification rates of different terrains. (b) Run times of different classification algorithms.

angular rate data was registered for the rock terrain, for which it showed almost 80% prediction accuracy, while for the sand terrain, the average prediction accuracy drops to near 5%. This figure is significant in nature, because it captures the fact that not all types of sensor data are useful for detecting all types of terrains. For example, one can notice that RPY data are very effective in detecting brick and concrete type of terrains, but for grass terrain, the accuracy drops significantly. On the other hand, IMU and acceleration data are useful in detecting the grass terrain. For sand terrain, all of the sensor data that we used did not perform very well.

Next, we summarize the performances of each classifier for different terrains as well as for different sensors. Fig. 10(a) shows the summarized version of all the classifiers' performances for different terrains. This figure clearly demonstrates that every classifier's performance level for different terrains are different. Some classifiers are better at detecting a certain terrain, while other classifiers might not be appropriate for detecting that terrain. For example, using the NB classifier, rock terrain is detected correctly almost every time, and, thus, the average prediction accuracy reaches near 100%. On the other hand, KNN5 and KNN10 classifiers do not perform as good as NB for rock terrain, but their performances for brick terrain are better than NB's performance for brick terrain.

We have noticed similar behavior when we have compared every classifier's performance for individual sensor data. This result is shown in Fig. 10(b). This figure demonstrates that even if the performances of all three classifiers are mostly similar for IMU and angular rate data, for RPY and acceleration data, the performances vary significantly. For example, the NB classifier performs best for RPY data, giving on average almost 7% higher prediction accuracy than either KNN5 or KNN10 classifier. On the other hand, both KNN5 and KNN10 perform significantly better than NB classifier for acceleration data with almost 15% higher prediction accuracy than the NB classifier.

In Fig. 11(a), we have summarized how many times each terrain type has been misclassified as other terrains. Brick has been mostly misclassified as concrete (16.92%). We believe that the main reason behind this is the similar, hard texture of these two terrains. It has also been misclassified as grass (4.63%), rock, and sand (both less than 1%) terrains. Grass terrain has been misclassified mostly as concrete terrain (27.01%). It has also been misclassified as rock (5.42%) and brick (3.14%) as well. Grass has been misclassified as sand very few times (0.15%). Rock has never been misclassified as either brick or sand. It has only been misclassified as concrete (3.15%) and grass (2.41%). Sand has been misclassified as brick and concrete types for whopping 21.29% and 15.53% times. It has also been misclassified as grass (8.60%) and rock (0.05%). Concrete terrain's misclassification rates are similar to brick terrain. Concrete has been misclassified as brick 8.5%

TABLE I
COMPARISON OF PREDICTION ACCURACY ON IMU DATA FOR DIFFERENT VALUES OF $K$ IN KNN CLASSIFIER

| Terrains | KNN5 | KNN10 | KNN$\sqrt{N}$ |
|---|---|---|---|
| Brick | 95.02% | 95% | 94.02% |
| Grass | 97% | 97% | 96% |
| Rock | 100% | 100% | 100% |
| Sand | 76.23% | 71.28% | 57.42% |
| Concrete | 82.25% | 80% | 62% |

and as grass 9.62% times, whereas it has been misclassified as rock (7.90%) and sand (0.57%).

In Fig. 11(b), we provide the run-times for our algorithm using different classifiers. It is evident from this figure that KNN5 and KNN10 take almost similar amount of run-time. But the NB classifier takes almost half the run-time of KNN5 and KNN10 to be executed for all types of sensors. On the other hand, if we use three classifiers together, then the amount of time required is almost double that of KNN5 and KNN10's individual running times. But, overall, the run-times of all types of classifiers are reasonable. The maximum amount of time taken is 2.1 s when all three classifiers run together on individual sensor data.

*3) Comparisons:* Next we wanted to see the effect of changing the value of $K$ in the KNN classifier. We have varied the value of $K$ between $5, 10$, and $\sqrt{N}$, where $N$ is the number of data points in the training set. Though there are different techniques available to find the optimal value of $K$, but the rule of thumb is to set $K$ as the square root of the training data points [34]. We have compared the performances of KNN classifier, having different values of $K$ on IMU test data. The result is shown in Table I. This result shows that when $K$'s value is increased to $\sqrt{N}$, then the performance of the KNN classifier drops. Significant changes in results can be noticed for sand and concrete terrains, where the prediction accuracy of the KNN classifier drops to almost 20% for both sand and concrete terrains when $K$'s value is changed from 5 to $\sqrt{N}$. We have also noticed similar behavior in performances for acceleration, angular rate and RPY data. We acknowledge that if more careful empirical tests and/or sophisticated Bayesian techniques, such as described in [34], are used, then KNN's prediction accuracy might increase slightly. But at the same time, it will take more computation time and computational resources to reach an optimum value of $K$. We have also compared the MD metric with the Euclidean distance metric for the KNN classifier. We found that the use of Euclidean distance reduces the prediction accuracy of KNN by 10% on average. This shows that our selections of the value $K$ and the distance metric for KNN classifier were justified.

Next, we compare the performance of our approach of using multiple low-resource consuming classifiers against an SVM classifier. For this comparison, we used the classical SVM model where there are only two classes (terrains) to be classified [35], [36]. Four sets of classes were tested using both SVM classifier and our three classifiers on four types of data—IMU, acceleration, RPY, and angular rate. The four sets of class couples are {brick, grass}, {concrete, grass}, {brick, sand}, and {sand, rock}. The results are shown in Table II. The number in this table indicate the prediction accuracy for different classes. For {brick, grass} set, performances of both the approaches are mostly similar. Only for acceleration and angular rate data, our approach performs worse in detecting grass terrain, than the SVM classifier by only 4.75% and 4.25%, respectively. In comparison our approach performs significantly better (21.89% higher prediction accuracy) in detecting brick terrain using angular rate data. For {concrete, grass} set, our approach performs much better than the SVM classifier in detecting concrete terrain for most of the sensors. On the other hand, the SVM classifier performs better than our approach in detecting grass terrain with most of the sensors. For {brick, sand} terrain set, our approach outperforms SVM classifier in detecting sand terrain, whereas SVM performs better in detecting brick terrain by 0.5% for acceleration and angular rate data. For {sand, rock} terrain set, both SVM and our approach were able to detect rock, using all four types of sensors for 100% of the test data set. But, in detecting sand terrain, our approach's performance is significantly better than SVM classifier. For example, for IMU, acceleration, RPY, and angular rate data, our approach's prediction accuracy is better than SVM by +38.71%, +10.88%, +28.71%, and +63.37%, respectively. We observed that the SVM's performance improved when all the sensors' data were combined together into a single vector, in comparison to when an individual sensors data was used. For example, for {brick, sand} set, with only angular rate data, sand was never correctly detected, but when all sensor data together as a single vector is used, then sand got detected correctly 71.28% times. KNN along with MD can not handle high-dimensional data ($>$10) very well, unless the data is preprocessed and its dimensions are reduced [37]. Therefore the results for the case where all sensors' features are used in a single data are not given. We also report that on an average, SVM (130 s) took 65 times more run time (combined training and testing time) than our proposed ensemble learning (2 s) method. This shows that we can achieve similar or sometimes better prediction accuracy as SVM and also reduce the run time, if we choose the classifiers wisely. If only testing time is considered, then the proposed ensemble learning method (1.2 s) and SVM (0.93 s) took almost similar run times.

*4) Using Multiple Sensors and Multiple Classifiers Together:* Finally, we use multiple classifiers and multiple types of data from a particular terrain to decide the terrain type. As we have seen earlier, the angular rate data's performance across all terrains and all types of classifiers was poor, therefore we leave it out while combining different data. We have only used IMU, acceleration, and RPY data and KNN5, KNN10, and NB classifiers. In all the experiments we have performed using multiple sensors together, we have used three classifiers at the same time. But we have used different sets of data combinations—{IMU, acceleration, RPY}, {IMU, acceleration}, {IMU, RPY}, and {acceleration, RPY}. Results have been shown in forms of confusion matrices in Fig. 12.

While using all three sensors, IMU, acceleration, and RPY, together, we were able to increase the prediction accuracy for most of the terrain types. For example, using three classifiers together, IMU, acceleration, and RPY sensors' average

TABLE II
COMPARISON OF PREDICTION ACCURACY ON DIFFERENT SENSOR DATA BETWEEN SVM AND OUR PROPOSED ENSEMBLE LEARNING METHOD

| IMU | Brick (B) and Grass (G) | Concrete (C) and Grass (G) | Brick (B) and Sand (S) | Sand (S) and Rock (R) |
|---|---|---|---|---|
| SVM | B: 100%, G: 100% | C: 59%, G: 98.5% | B: 100%, S: 71.29% | S: 71.29%, R: 100% |
| Our Approach | B: 100%, G: 100% | C: 97%, G: 98% | B: 100%, S: 75.25% | S: 100%, R: 100% |
| Difference | B: 0%, G: 0% | C: +38%, G: −0.5% | B: 0%, S: +3.96% | S: +38.71%, R: 0% |
| Acceleration | | | | |
| SVM | B: 100%, G: 96.5% | C: 96.5%, G: 92.5% | B: 100%, S: 45.54% | S: 87.13%, R: 100% |
| Our Approach | B: 100%, G: 91.75% | C: 95%, G: 95% | B: 99.50%, S: 85.15% | S: 98.01%, R: 100% |
| Difference | B: 0%, G: −4.75% | C: −1.5%, G: +2.5% | B: −0.5%, S: +39.61% | S: +10.88%, R: 0% |
| RPY | | | | |
| SVM | B: 100%, G: 100% | C: 59%, G: 65% | B: 100%, S: 71.29% | S: 71.29%, R: 100% |
| Our Approach | B: 100%, G: 100% | C: 99.5%, G: 30.5% | B: 100%, S: 71.29% | S: 100%, R: 100% |
| Difference | B: 0%, G: 0% | C: +40.5%, G: −34.5% | B: 0%, S: 0% | S: +28.71%, R: 0% |
| Angular Rate | | | | |
| SVM | B: 45.77%, G: 90.75% | C: 36.25%, G: 79.25% | B: 100%, S: 0% | S: 15.84%, R: 100% |
| Our Approach | B: 67.66%, G: 86.5% | C: 48.75%, G: 64% | B: 99.5%, S: 2.97% | S: 79.21%, R: 100% |
| Difference | B: +21.89%, G: −4.25% | C: +12.5%, G: −15.25% | B: −0.5%, S: +2.97% | S: +63.37%, R: 0% |
| All Data Combined | | | | |
| SVM | B: 100%, G: 100% | C: 59%, G: 98% | B: 100%, S: 71.28% | S: 71.28%, R: 100% |

| | brick | grass | rock | sand | concrete | undecidec |
|---|---|---|---|---|---|---|
| brick | 97 | 0 | 0 | 0 | 3 | 0 |
| grass | 0 | 82 | 0 | 0 | 15 | 3 |
| rock | 0 | 0 | 100 | 0 | 0 | 0 |
| sand | 21 | 0 | 0 | 77 | 2 | 0 |
| concrete | 13 | 0 | 0 | 0 | 87 | 0 |

(a)

| | brick | grass | rock | sand | concrete | undecidec |
|---|---|---|---|---|---|---|
| brick | 76 | 0 | 0 | 0 | 5 | 19 |
| grass | 0 | 75 | 0 | 0 | 20 | 5 |
| rock | 0 | 0 | 100 | 0 | 0 | 0 |
| sand | 5 | 3 | 0 | 76 | 3 | 13 |
| concrete | 11 | 0 | 0 | 0 | 88 | 1 |

(b)

| | brick | grass | rock | sand | concrete | undecidec |
|---|---|---|---|---|---|---|
| brick | 100 | 0 | 0 | 0 | 0 | 0 |
| grass | 0 | 91 | 0 | 0 | 9 | 0 |
| rock | 0 | 0 | 100 | 0 | 0 | 0 |
| sand | 28 | 0 | 0 | 72 | 0 | 0 |
| concrete | 10 | 0 | 0 | 0 | 85 | 5 |

(c)

| | brick | grass | rock | sand | concrete | undecidec |
|---|---|---|---|---|---|---|
| brick | 77 | 0 | 0 | 0 | 0 | 23 |
| grass | 0 | 79 | 0 | 0 | 14 | 7 |
| rock | 0 | 0 | 100 | 0 | 0 | 0 |
| sand | 6 | 0 | 0 | 64 | 0 | 30 |
| concrete | 9 | 0 | 0 | 0 | 86 | 5 |

(d)

Fig. 12. Confusion matrix (in %) using all three classifiers for (a) {IMU, acceleration, RPY}, (b) {IMU, acceleration}, (c) {IMU, RPY}, and (d) {acceleration, RPY} data sets.

prediction accuracy for brick terrain was 89.33%, whereas when used these three sensors together, the prediction accuracy for brick terrain went up to 97%. Similar behavior can be noticed for sand terrain, where three sensors' average prediction accuracy for this type was 72%; but using three sensors' data together took the prediction accuracy level to 82%. For concrete terrain, prediction accuracy is improved than IMU's individual accuracy level, but it performed worse than acceleration and RPY sensors' individual predictions. For rock terrain, we did not notice any change in accuracy. When we used only two sensors together, then {IMU, RPY} sensor set performed the best. The reason is that these two sensors' individual performances were also good. The run time for using three classifiers along with three sensors was 3 s, whereas if we use two sensors instead of three, then the average run time is 2.8 s.

As can be observed in Fig. 12(b) and (d), the number of undecided outputs increases. The reason behind this phenomena is that if the product of number of sensors ($|S|$) and number of classifiers ($|C|$) used produces an even number, then there is a high chance of multiple terrain types to win the same number of points in the end, and, therefore, the number of undecided outputs increases. On the other hand, if this product produces an odd number, then the probability of one classifier winning more points than others increases, and, thus, the number of undecided outputs decreases. As we do not know *a priori* which sensor or classifier is going to work better for a certain type of terrain, therefore, from this result we can say that using multiple sensors and multiple classifiers together will be better for most of the terrain types to reach the final decision.

## IV. CONCLUSION

In this paper, we proposed a novel technique for unknown terrain classification. Most of the previous approaches use a powerful SVM classifier along with visual data, such as camera images for classification. In contrast, we have proposed a novel approach, where we use multiple weak classifiers together. Each weak classifier might not be as good as SVM individually, but they can perform comparably with SVM when combined together in an ensemble manner. The ensemble classifier that we have used in this paper is 65 times less time consuming than SVM. Although SVM has a very high accuracy in prediction, but implementing SVM on a real robot, which does not have much computation resources, might be challenging because of SVM's high computational resource requirement for training phase. On the other hand, we have empirically shown that using multiple weak classification algorithms and multiple types of sensed data together, we can achieve a similar level of accuracy as SVM, with significantly less computation. We have also shown that even without using visual sensor data, such as camera, which can yield high prediction accuracy, we can still achieve reasonable performance from the algorithm. The sensors that we have used in this paper do not depend on external factors like illumination condition of the environment and therefore our approach is more robust in nature. At the same time, our framework is not limited to the classifiers and sensor data we have selected for testing purposes in this paper. Any other sets of sensor data and/or classifiers can be used in conjunction with our framework. We have also showed that using multiple types of sensed data and multiple classifiers together can yield better performances for different terrains, than using a single classifier and a single sensor. As we do not know the best combination of classifier and sensor data which will perform well across all terrains, therefore, it is wise to use multiple classifiers and sensors at the same time. Even when multiple classifiers and sensor data used, our algorithm's run time is always within a reasonable range. We have also seen from the results that the performance of a particular sensor's data along with a particular classifier can not be generalized for all terrain types. Some classifiers always performs better along with some set of sensor data for a particular terrain type than on other terrains. Our findings in this paper will help future researchers in this topic to think beyond the SVM classifier and vision-based terrain classification.

In future, it would be interesting to see the effect of using other complex ensemble learning methods, such as boosting, instead of bagging-based approach. Also, we would like to see the effect of using weaker learning algorithms, such as decision trees, and would like to compare the performances (both run time and classification quality) of those algorithms and our proposed approach. We are also planning to work on prediction of terrain conditions (such as wet or dry) using a mobile robot. Finally, we are planning to develop an algorithm for adaptive robot navigation, which will use our proposed unknown terrain classification algorithm for generating the most-suited navigation pattern (e.g., speed, acceleration, and gait) for any particular terrain for ModRED [38] modular robot.

## ACKNOWLEDGMENT

The authors would like to thank the support of Dr. J. Baca, who helped us in collecting the sensor data using the Explorer robot.

## REFERENCES

[1] S. Thrun *et al.*, "Stanley: The robot that won the darpa grand challenge," *J. Field Robot.*, vol. 23, no. 9, pp. 661–692, 2006.

[2] Y. Cheng, M. W. Maimone, and L. Matthies, "Visual odometry on the Mars exploration rovers—A tool to ensure accurate driving and science imaging," *IEEE Robot. Autom. Mag.*, vol. 13, no. 2, pp. 54–62, Jun. 2006.

[3] L. Ojeda, J. Borenstein, G. Witus, and R. Karlsen, "Terrain characterization and classification with a mobile robot," *J. Field Robot.*, vol. 23, no. 2, pp. 103–122, 2006.

[4] C. A. Brooks and K. Iagnemma, "Vibration-based terrain classification for planetary exploration rovers," *IEEE Trans. Robot.*, vol. 21, no. 6, pp. 1185–1191, Dec. 2005.

[5] M. Woods, J. Guivant, and J. Katupitiya, "Terrain classification using depth texture features," in *Proc. Aust. Conf. Robot. Autom.*, Sydney, NSW, Australia, 2013, pp. 1–8.

[6] A. Angelova, L. Matthies, D. Helmick, and P. Perona, "Fast terrain classification using variable-length representation for autonomous navigation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Minneapolis, MN, USA, 2007, pp. 1–8.

[7] M. Häselich, M. Arends, N. Wojke, F. Neuhaus, and D. Paulus, "Probabilistic terrain classification in unstructured environments," *Robot. Auton. Syst.*, vol. 61, no. 10, pp. 1051–1059, 2013.

[8] S. Zenker, E. E. Aksoy, D. Goldschmidt, F. Wörgötter, and P. Manoonpong, "Visual terrain classification for selecting energy efficient gaits of a hexapod robot," in *Proc. IEEE/ASME Int. Conf. Adv. Intell. Mechatronics (AIM)*, Wollongong, NSW, Australia, 2013, pp. 577–584.

[9] A. Sinha, N. Tan, and R. E. Mohan, "Terrain perception for a reconfigurable biomimetic robot using monocular vision," *Robot. Biomim.*, vol. 1, no. 1, pp. 1–11, 2014.

[10] P. Filitchkin and K. Byl, "Feature-based terrain classification for littledog," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Vilamoura, Portugal, 2012, pp. 1387–1392.

[11] J.-F. Lalonde, N. Vandapel, D. F. Huber, and M. Hebert, "Natural terrain classification using three-dimensional ladar data for ground robot mobility," *J. Field Robot.*, vol. 23, no. 10, pp. 839–861, 2006.

[12] M. W. McDaniel, T. Nishihata, C. A. Brooks, P. Salesses, and K. Iagnemma, "Terrain classification and identification of tree stems using ground-based LiDAR," *J. Field Robot.*, vol. 29, no. 6, pp. 891–910, 2012.

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

[14] P. Papadakis, "Terrain traversability analysis methods for unmanned ground vehicles: A survey," *Eng. Appl. Artif. Intell.*, vol. 26, no. 4, pp. 1373–1385, 2013.

[15] S. Singh *et al.*, "Recent progress in local and global traversability for planetary rovers," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, vol. 2. San Francisco, CA, USA, 2000, pp. 1194–1200.

[16] Y. Guo, A. Song, J. Bao, and H. Zhang, "Optimal path planning in field based on traversability prediction for mobile robot," in *Proc. Int. Conf. Elect. Inf. Control Eng. (ICEICE)*, Wuhan, China, 2011, pp. 563–566.

[17] P. Bellutta, R. Manduchi, L. Matthies, K. Owens, and A. Rankin, "Terrain perception for DEMO III," in *Proc. IEEE Intell. Veh. Symp. IV*, Dearborn, MI, USA, 2000, pp. 326–331.

[18] R. Manduchi, A. Castano, A. Talukder, and L. Matthies, "Obstacle detection and terrain classification for autonomous off-road navigation," *Auton. Robot.*, vol. 18, no. 1, pp. 81–102, 2005.

[19] J. Han, M. Kamber, and J. Pei, *Data Mining, Southeast Asia Edition: Concepts and Techniques*. San Mateo, CA, USA: Morgan Kaufmann, 2006.

[20] D. Decoste and B. Schölkopf, "Training invariant support vector machines," *Mach. Learn.*, vol. 46, nos. 1–3, pp. 161–190, 2002.

[21] N. Abe and H. Mamitsuka, "Query learning strategies using boosting and bagging," in *Proc. 15th Int. Conf. Mach. Learn. (ICML)*, Madison, WI, USA, 1998, pp. 1–9.

[22] R. Paul, R. Triebel, D. Rus, and P. Newman, "Semantic categorization of outdoor scenes with uncertainty estimates using multi-class Gaussian process classification," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Vilamoura, Portugal, 2012, pp. 2404–2410.

[23] L. Breiman, "Bagging predictors," *Mach. Learn.*, vol. 24, no. 2, pp. 123–140, 1996.

[24] K. Walas, "Terrain classification and negotiation with a walking robot," *J. Intell. Robot. Syst.*, vol. 78, no. 3, pp. 401–423, 2014.

[25] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE Trans. Inf. Theory*, vol. 13, no. 1, pp. 21–27, Jan. 1967.

[26] D. Barber, *Bayesian Reasoning and Machine Learning*. New York, NY, USA: Cambridge Univ. Press, 2012.

[27] P. C. Mahalanobis, "On the generalized distance in statistics," in *Proc. Nat. Inst. Sci.*, vol. 2. Kolkata, India, 1936, pp. 49–55.

[28] J. C. Gower, "Properties of Euclidean and non-Euclidean distance matrices," *Linear Algebra Appl.*, vol. 67, pp. 81–97, Jun. 1985.

[29] E.-J. Ong and R. Bowden, "A boosted classifier tree for hand shape detection," in *Proc. 6th IEEE Int. Conf. Autom. Face Gesture Recognit.*, Seoul, South Korea, 2004, pp. 889–894.

[30] P. A. Lachenbruch and M. R. Mickey, "Estimation of error rates in discriminant analysis," *Technometrics*, vol. 10, no. 1, pp. 1–11, 1968.

[31] G. Liang, X. Zhu, and C. Zhang, "An empirical study of bagging predictors for different learning algorithms," in *Proc. 25th Conf. Artif. Intell. (AAAI)*, San Francisco, CA, USA, Aug. 2011, pp. 1802–1803.

[32] D. M. Magerman, "Statistical decision-tree models for parsing," in *Proc. 33rd Annu. Meeting Assoc. Comput. Linguist.*, Boston, MA, USA, 1995, pp. 276–283.

[33] S. Mitra and S. K. Pal, "Fuzzy multi-layer perceptron, inferencing and rule generation," *IEEE Trans. Neural Netw.*, vol. 6, no. 1, pp. 51–63, Jan. 1995.

[34] A. K. Ghosh, "On optimum choice of $k$ in nearest neighbor classification," *Comput. Stat. Data Anal.*, vol. 50, no. 11, pp. 3113–3123, 2006.

[35] J. A. K. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," *Neural Process. Lett.*, vol. 9, no. 3, pp. 293–300, 1999.

[36] N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods*. New York, NY, USA: Cambridge Univ. Press, 2000.

[37] K. Q. Weinberger and L. K. Saul, "Distance metric learning for large margin nearest neighbor classification," *J. Mach. Learn. Res.*, vol. 10, pp. 207–244, 2009.

[38] J. Baca, S. G. M. Hossain, P. Dasgupta, C. A. Nelson, and A. Dutta, "ModRED: Hardware design and reconfiguration planning for a high dexterity modular self-reconfigurable robot for extra-terrestrial exploration," *Robot. Auton. Syst.*, vol. 62, no. 7, pp. 1002–1015, 2014.


**Ayan Dutta** is currently pursuing the Ph.D. degree with the Computer Science Department, University of Nebraska at Omaha, Omaha, NE, USA, under the supervision of Dr. P. Dasgupta.

He is a Research Assistant with the C-MANTIC Laboratory. His research interests include self-assembly and locomotion planning for self-reconfigurable modular robots and multirobot path planning.

**Prithviraj (Raj) Dasgupta** received the Ph.D. degree from the University of California at Santa Barbara, Santa Barbara, CA, USA, in 2001.

He is a Full Professor with the Computer Science Department, University of Nebraska at Omaha, Omaha, NE, USA, and the Founder-Director of the C-MANTIC Robotics Laboratory. He has led multiple, large, federally funded projects in the area of multirobot/multiagent systems and published over 100 research papers in leading conferences and journals in the areas of modular robot systems, multirobot, and multiagent systems.