Posterior Probability Estimation Techniques Embedded in a Bayes Filter for Vibration-Based Terrain Classification

Philippe Komma and Andreas Zell

Abstract. Vibration signals acquired during robot traversal provide enough information to yield a reliable prediction of the current terrain type. In a recent approach, we combined a history of terrain class estimates into a final prediction. We therefore adopted a Bayes filter taking the posterior probability of each prediction into account. Posterior probability estimates, however, were derived from support vector machines only, disregarding the capability of other classification techniques to provide these estimates. This paper considers other classifiers to be embedded into our Bayes filter terrain prediction scheme, each featuring different characteristics. We show that the best classification results are obtained using a combined k-nearest-neighbor and support vector machine approach which has not been considered for terrain classification so far. Furthermore, we demonstrate that other classification techniques also benefit from the temporal filtering of terrain class predictions.

1 Introduction

In outdoor applications such as rescue missions or agricultural assignments the mobile robot navigates over varying ground surfaces, each possessing different characteristics. To ensure a safe traversal in outdoor environments the robot should adapt its driving style according to the presence of ground surface hazards like slippery or bumpy surfaces. These hazards are denoted as non-geometric hazards [18]. Therefore, most approaches employ a model-based prediction scheme which estimates the current terrain type from sensor readings. In a model generation phase, the model learns the correct assignment of a labeled terrain class given the respective observation. In the recall phase, that is, during terrain traversal over unknown terrain, the robot then uses this model to predict the current ground surface. For

Philippe Komma and Andreas Zell
Chair of Computer Architecture, Computer Science Department,
University of Tübingen, Sand 1, D-72076 Tübingen, Germany
e-mail: {philippe.komma,andreas.zell}@uni-tuebingen.de

A. Howard et al. (Eds.): Field and Service Robotics 7, STAR 62, pp. 79–89.
springerlink.com

acquiring input data, a variety of sensors such as vision [11, 1] or ladar sensors [15, 10] can be employed. Recently, several researchers considered vehicle vibrations for terrain classification as originally proposed in [8]. In this context, vibration data acquired from accelerometers have been successfully applied to planetary rovers [3], autonomous ground vehicles [7], and experimental unmanned vehicles [6]. In [16] a comparison was drawn between different base classifiers providing the model for vibration-based terrain classification. These techniques, however, estimate the terrain type using single sensor measurements only, disregarding the temporal coherence between consecutive measurements. We addressed this problem in [9]. There, we applied a Bayes filter to combine the posterior probabilities of several recent terrain class predictions into a final prediction. In our approach, posterior probability estimation was performed using a support vector machine (SVM) since this classifier was reported to yield the best classification results in a single observation-based prediction scheme [16].

To motivate our current research we first note that the performance of a classifier in the context of Bayes-filtered terrain classification does not depend on the classification quality only but also on the quality of the prediction certainty: Since the final classification is based on the posterior probability of single predictions, it benefits from a model which performs confident correct predictions and uncertain erroneous predictions. Classifiers which provide these characteristics result in a better prediction performance when embedded into our Bayes filter approach. This is because erroneous predictions obtain a lower weight in the filtering process and thus influence the final prediction less significantly. The quality of various classifiers relating to the prediction certainty is unclear and is hence investigated in this paper. Second, the SVM classifier is not an appropriate choice in all domains, especially for online learning [17] where an enduring model generation phase is not applicable. Thus, this paper focuses on the selection of an adequate classifier with regard to its limiting factors such as training and model selection time, storage requirements, and the run-time complexity of the recall phase. We further applied the SVM-KNN classifier introduced in [21] which in our terrain classification task was significantly superior to all other classifiers considered so far.

The remainder of this paper is organized as follows: Section 2 briefly describes our terrain classification model, taking both one and several recent observations into account. The posterior probability estimation techniques of the classifiers to be embedded in our temporally-filtered classification approach are introduced in Sect. 3. After summarizing our experimental setup in Sect. 4 we present and discuss experimental results in Sect. 5. Finally, a conclusion is given in the last section.

2 Terrain Classification Model

This section summarizes our terrain classification technique based on both single observations and temporal filtering of several recent terrain predictions. A detailed description is presented in [9].

The objective of our approach is to estimate the terrain type the robot is currently traversing. Predictions are model-based, assigning a certain terrain class from a set of classes to recorded observations. We represent the observations by acceleration data sampled at a frequency of 100 Hz over a period of 1.28 s. The acceleration data can be regarded as the vibration which the terrain induces to the body of the robot. For feature extraction, we applied the Fast Fourier Transform (FFT) to the raw input signal to determine its FFT amplitude spectrum in a second step. We then normalized the data by scaling each component of the preprocessed vibration signal to have a mean of 0 and a standard deviation of 1. The scaled amplitude spectrum entries constitute the inputs for the terrain classification model.

In the recall phase, the robot predicts the current terrain type, using the terrain classification model generated during training. Therefore, the same preprocessing has to be applied to the acquired vibration data. Using the posterior probability estimation techniques presented in the next section, the application of the final feature vector to the classifier does not only provide a class prediction but also an approximation of the posterior $p(x = i|u)$. This probability distribution denotes the probability that a preprocessed vibration segment u belongs to terrain class i. Next, we describe how $p(x|u)$ can be embedded into a Bayes filter framework.

Using a Bayes filter [14], the state of a dynamic system at a time t is represented by a random variable x_t. In our context, $x_t \in [1;k]$ models the uncertainty with which the robot navigates over one of the k terrain types. Given $t+1$ preprocessed vibration segments $u_{0:t} = \{u_0, u_1, \ldots, u_t\}$ recorded by accelerometer sensors, the estimated target distribution is determined by $p(x_t|u_{0:t})$. In [9] we showed that $p(x_t|u_{0:t})$ can be formally defined as:

$$p(x_i = i|u_{0:t}) = \alpha_t\, p(x_t = i|u_t) \sum_j p(x_t = i|x_{t-1} = j) p(x_{t-1} = j|u_{0:t-1}).$$

Here, $p(x_t|u_t)$ substitutes the measurement probability $p(u_t|x_t)$ and represents the probability that the vibration measurement u_t can be observed when navigating over a certain terrain type x_t. $p(x_t|u_t)$ is derived from the Bayes inversion $p(u_t|x_t) = p(x_t|u_t)\frac{p(u_t)}{p(x_t)}$ assuming that $p(x_t)$ is distributed uniformly. Further note that $p(u_t)$ is constant for all i and can thus be included in the normalizing constant α_t.

The transition probability $p(x_t|x_{t-1})$ denotes the probability that the robot moves from terrain type $x_{t-1} = j$ to $x_t = i$. Bayes filters model the dynamic system by a first-order Markov process assuming that the information provided by the state x_t suffices to predict future states without considering earlier observations. Our approach is based on the heuristic that the terrain class most likely does not change from one measurement to the next. Thus, we assign a relatively large value v to $p(x_t = i|x_{t-1} = i)$. $p(x_t = i|x_{t-1} = j)$, with $i \neq j$, is derived from the following two heuristics: First, the probability $p(x_t = i|x_{t-1} = j)$ should increase with the probability to confuse class i with class j. Second, a transition from state $x_{t-1} = j$ to state $x_t = i$ should be based on the probability to predict the terrain class at time t correctly. Both probabilities can directly be estimated from the confusion matrix. For further details, we refer to [9]. By dynamically changing v, the probability that the

system remains in its current state, we obtain an approach being both reactive and stable enough to detect fast terrain transitions and selective misclassifications. In our implementation, v is either increased or decreased by a constant factor depending on whether the current prediction equals the system state at time $t-1$. Upper and lower bounds for v ensure that the probability of a state transition neither becomes too large nor too small.

For the definition of the initial probability distribution $p(x_0)$, we make no assumptions that the robot is placed on a specific terrain type at time $t = 0$. Hence, $p(x_0)$ is assumed to be uniformly distributed.

3 Posterior Probability Estimation

In this section, we briefly describe all classifiers that have been embedded into our Bayes filter classification approach. Therefore, we explain how posterior probabilities $p(x = i|u)$ can be predicted for each class i under consideration. Since each classifier features different characteristics we conclude this section by indicating in which situations the choice of a certain classifier is appropriate.

k-nearest neighbor classification (KNN) [5] determines the set of k-nearest-neighbors contained in a training set to a testing instance u. Then, we calculate the frequency of occurrence of each class in the neighbor set. The class with the largest frequency becomes the predicted class for the testing instance u. The posterior probability $p(x = i|u)$ is defined as the ratio between the number of occurrences of class i in the neighbor set n_i and the number of considered neighbors k, $p(x = i|u) = \frac{n_i}{k}$.

The multilayer perceptron (MLP) [2] is an instance of an artificial neural network. It consists of artificial neurons which are interconnected in a well-defined manner. These neurons are arranged in three different layers: in an input layer, a hidden layer, and an output layer. When applying an input u to the network input, the neurons of the hidden layer perform a weighted sum of the input components: $net_l = \langle w_l, u \rangle$. Here, net_l denotes the net activation of neuron h_l and w_l is the weight vector determining the specific contribution of each input component to the final sum. We then apply an activation function f_{act}, typically chosen as $f_{act} = tanh(net_l)$, to each net activation to obtain the final output for the neurons of the hidden layer. The determination of the net activation of the output neurons is equivalent to the ones of the hidden layer except that we do not add weighted input coefficients but weighted activations of the hidden neurons. For classification problems, the activation function of the output neurons is replaced by the softmax function which takes the form $f_{act} = \exp(net_m)/\sum_{m'} \exp(net_{m'})$, where net_m is the net activation for output neuron m. Each output neuron represents a certain class to discriminate. The predicted class is the one which is represented by the neuron with the maximum activation. It can be shown [2] that the activations can directly be interpreted as posterior probabilities.

Probabilistic neural networks (PNN) [13] are another instance of artificial neural networks. In the training phase, scaled training patterns are inserted into a matrix W_c, $c \in [1;k]$, according to the class c they belong to. Each row of W_c represents a single pattern. The scaling is performed such that the L_2 norm of each training instance

equals to one. In the recall phase, the same scaling is applied to the test vector u. For each class c, the inner product between each pattern w_i of the weight matrix W_c and the query u is determined yielding the net activation $net_{c,i}$. The net activations are non-linearly transformed using the activation function $f_{act}(net_{c,l}) = \exp((net_l - 1)/\sigma^2)$, where σ is a model parameter defining the size of the Gaussian window. For each class, the sum over all transformed net activations is determined, $s_c = \sum_l f_{act}(net_{c,l})$, and the predicted class becomes the one which maximizes s_c. Given that the probability of each class is distributed uniformly, posterior probabilities $p(x = i|u)$ can then be defined as $p(x = i|u) = (n_i^{-1} s_i)/(\sum_j n_j^{-1} s_j)$, where n_c is the number of training instances for class c.

Given two classes c_1 and c_2 to discriminate, a support vector machine (SVM) [4] establishes a separating hyperplane such that each instance of the first class resides in one subspace and each instance of the other class resides in the other subspace. To increase generalization we maximize the margin which is the distance from the hyperplane to the instances closest to it. In the non-separable case, that is, if no hyperplane exists which separates the two classes, instances of class c_1 are allowed to reside in the subspace representing class c_2 and vice versa. However, a penalty term is added for each non-separable training point. Problems exist, which are not linearly separable in the original space spanned by the training data but which become linearly separable when mapping the inputs u_i into a higher dimensional feature space, $z = \phi(u)$. Using the "kernel trick" the actual mapping does not have to be performed. Instead, we exploit the fact that the inner product of basis functions $\phi(x)^T \phi(y)$ is replaced by a kernel function $K(x,y)$. In our experiments, we used the radial basis function kernel defined as $K(x,y) = \exp\left(-\|x-y\|^2/\sigma^2\right)$. Multi-class classification using n classes is achieved by establishing $n(n-1)/2$ binary classifiers in a one-versus-one classification scheme. Adopting the technique of [12], a parameterized sigmoid function is applied to the decision value of each binary classification which results in posterior probabilities of both classes. Finally, we obtain the posterior for each class i, $p(x = i|u)$, using the pairwise coupling method of [19].

The SVM-KNN approach [21] combines the characteristics of both the KNN and the SVM classifiers. It does not require a training phase. Instead, predictions are performed by first pruning the training set. Therefore, the k-nearest-neighbors to a given query u are identified. Then, a multi-class SVM is trained online using the pairwise distances between all entries of the union of the query and the neighbor set. Prior to the SVM model training, these distances have to be transformed into a kernel matrix. In our approach, this is realized by applying the function $f(d_{ij}) = \exp(-d_{ij}^2/\sigma^2)$ to the pairwise distances d_{ij}. As distance function we chose the L_2 norm. The posterior probability $p(x|u)$ is then obtained by applying the query u to the trained SVM.

Classifier selection should be handled with care since each approach has different characteristics. KNNs and PNNs belong to the class of lazy learning techniques. That is, all computations are delayed until a prediction query is requested. On the one hand, this renders a time-demanding training phase unnecessary which is advantageous if the underlying phenomenon changes frequently. On the other hand, all patterns have to be available at run-time which might pose a problem if storage

Table 1 The respective model parameter(s), the number of considered candidates during model selection, model selection and training times, prediction complexity, and storage requirements of the proposed classifiers.

	KNN	MLP	PNN	SVM	SVM-KNN
model param.	k = 6	hid. = 96	$\sigma = 0.07$	C = 9.05,$\sigma = 0.02$	k = 640, C = 9.05, $\sigma = 0.02$
model sel. cand.	31	8	64	196 (14×14)	k: 30; C,σ: 196
model sel. time (h)	0:52:55	32:34:00	0:07:54	24:55:47	50:09:40
training time (h)	-	1:08:43	0:00:01	0:00:54	-
testing time (ms)	13.21	0.02	0.54	1.07	464.9
storage req. (kB)	763.5	52.5	763.5	22.5	763.5

is limited. Given that the acquired training set consists of n samples, storage requirements are $O(n \cdot d)$, where d is dimensionality of a training instance. Furthermore, if the calculating capacity is constrained in the recall phase, the desired prediction frequency might not be accomplished due to a large set of training patterns. For example, when using the KNN classifier, a naïve approach involves $O(n)$ distance calculations to determine the k-nearest-neighbors. Although accelerating data structures like M-trees [20] exist, high-dimensional nearest-neighbor search is known to be a non-trivial task suffering from the curse of dimensionality.

MLP and SVM classifiers typically provide compact models, resulting in a fast prediction performance. Model training, however, is computationally much more demanding since both methods iteratively try to minimize a given error function. The time spent on choosing a classifier with a good generalization behavior is significantly increased by the model selection process which has to consider a sufficiently large set of candidate model parameter settings.

The SVM-KNN approach is characterized by an involved model selection and testing phase. Since a class prediction also requires the determination of the k-nearest-neighbor set to a given query, the training set has to be present at run-time. We note, however, that this approach still guarantees predictions performed in real-time. Hence, we included SVM-KNN classification in our investigations.

Table 1 summarizes the key characteristics for the proposed classifiers: the respective model parameter(s) which yielded the best generalization and model selection times along with the number of tested model candidates. For the respective best classification model we further present the training time using data contained in one fold of a 5-fold cross validation scheme, the average testing time for a single query, and storage requirements (measured in kB). We performed all run-time analyses on a Pentium D 3.0 GHz desktop PC. For the storage considerations, we represented each floating point number as *double*, each 8 bytes in size.

4 Experimental Setup

In our experiments, an Xsens MTi altitude and heading reference system was mounted on an aluminum plate on top of our RWI ATRV-Jr outdoor robot to measure

Fig. 1 The employed terrain types: 1: indoor floor, 2: asphalt, 3: gravel, 4: grass, 5: boule court.

vibration signals in left-right direction at 100 Hz. During data acquisition, the robot navigated over five different terrains (Fig. 1): indoor PVC floor, asphalt, gravel, grass, and clay (the surface of a boule court). To not constrain the model to work at a certain driving speed, we varied the speed between 0.2, 0.4, and 0.6 m/s. In total, the dataset consists of 7635 patterns, corresponding to approximately 1.5 hours of robot navigation.

We performed individual terrain classifications using vibration data acquired during 1.28 s of robot travel. For two consecutive segments we permit an overlap of 28 samples to achieve a prediction frequency of 1 Hz. The combination of terrain class predictions was realized by our adaptive Bayes filter approach introduced in Sect. 2. To quantify the performance of the latter, we applied the following evaluation procedure: We assembled consecutive vibration segments representing the same terrain type to give a travel distance of constant length. Then, assembled segments of varying terrain types were grouped together yielding the final test set. In different experiments, we varied the distance covered by a robot before it reaches a new terrain class. This distance is denoted as the *travel distance d* (measured in meters) in the following. In each experiment, d was chosen from the set $d \in \{0;4;8;12;16\}$. The 0 m experiment describes the worst case scenario for approaches based on temporal filtering. Here, single segments of varying terrain classes are concatenated, each representing data acquired during 1 s of robot travel. Since the robot speed varies between 0.2, 0.4, and 0.6 m/s, this experiment includes travel distances of 0.2, 0.4, and 0.6 m. Note that according to the confusion matrix, certain terrain transitions are easier to detect than other ones. Hence, the results depend on the order in which assembled terrain segments of varying terrain type are presented. We minimized this effect by randomly permuting this order and averaging the classification results determined after 20 reruns of a particular experiment.

As quality measure we used the true positive rate (TPR). It is the ratio (measured in per cent) of the number of correct predictions for which the predicted class x_t equals the actual class $\hat{x}_t$ and the number of instances contained in the test set. We derived the prediction performance using 5-fold cross validation and averaging the true positive rate over all five folds.

5 Experimental Results

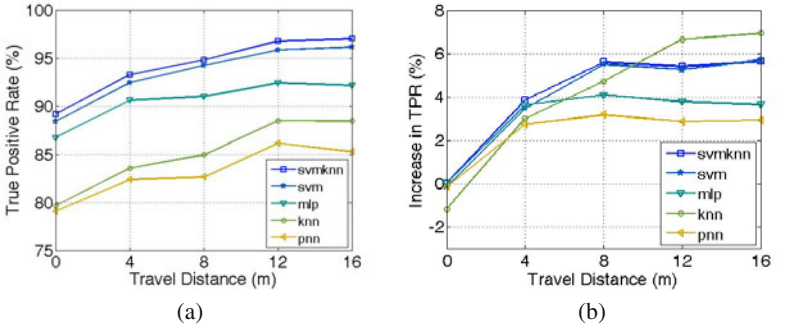
Table 2 shows the results for the proposed classifiers when using single observations (SO) and Bayes-filtered posterior probabilities of recent predictions (AB). Note that

the true positive rate for the single observation-based approach differs between varying experiments. This is due to the model evaluation procedure introduced in the previous section which selects a varying test set for each travel distance.

Related to both the SO and the AB approach, the SVM-KNN technique yields the best prediction performance, followed by SVM, MLP, KNN, and PNN classification (Fig. 2(a)). The differences in the true positive rates of the applied classifiers proved to be statistically significant, using a two-tailed t-test at a significance level of 1%. The combined support vector machine and k-nearest-neighbor approach benefits from the reduced training set resulting in another configuration of the separating hyperplane. This hyperplane results in a higher generalization as compared to the one established by the SVM approach which uses all training patters at once. The classification performance of each classifier is also reflected in the increase of the true positive rate obtained when using the adaptive Bayes technique in comparison with the single observation approach (Fig. 2(b)): the larger the true positive rate for a given classifier, the larger the benefit of using temporally-filtered predictions. This statement holds for all classifiers but the KNN approach: Here, the adaptive Bayes technique results in the largest increase of the true positive rate. Investigations

Table 2 Prediction performance (in %) for varying classifiers and travel distances (dist.) using single observation-based (SO) and adaptive Bayes filter-based (AB) terrain classification.

dist. (m)	0		4		8		12		16	
approach	SO	AB	SO	AB	SO	AB	SO	AB	SO	AB
SVM-KNN	89.1	89.1	89.8	93.3	89.8	94.8	91.8	96.7	91.8	97.0
SVM	88.5	88.4	89.3	92.4	89.3	94.2	91.0	95.8	90.9	96.1
MLP	86.7	86.7	87.4	90.6	87.4	91.0	89.0	92.4	88.9	92.1
KNN	80.6	79.7	81.2	83.6	81.1	85.0	82.9	88.5	82.7	88.4
PNN	79.2	79.1	80.2	82.4	80.1	82.7	83.7	86.1	82.8	85.3



(a) (b)

Fig. 2 (a) True positive rates for the adaptive Bayes approach and (b) the relative increase of classification performance for the adaptive Bayes approach related to single observation-based classification when varying the classifier and the travel distance.

revealed that in the case of a misclassification the posterior probability of the erroneously predicted class p_e was rather small on average, $p_e = 0.64 \pm 0.19$, in comparison with the ones obtained for a correctly predicted class, $p_c = 0.87 \pm 0.18$. Hence, the transition into another, that is, erroneous system state becomes less likely if the system previously resided in the correct state, no terrain transition occurred, but the classifier erroneously estimated the wrong terrain class. We obtained similar results using SVM and SVM-KNN classifiers: Here, the posterior probability of erroneous predictions was $p_e = 0.69 \pm 0.18$ and $p_e = 0.66 \pm 0.18$, respectively. PNN and MLP classifiers perform wrong predictions with higher confidences: $p_e = 0.87 \pm 0.17$ and $p_e = 0.84 \pm 0.17$, respectively. Referring to Fig. 2(b), this is another explanation for the smaller increase in TPR compared to other classifiers which provide more uncertain erroneous predictions. In addition, we observed that the posterior probabilities of correct predictions were larger than 0.87 on average for all classifiers.

Fig. 2(b) shows the true positive rates when applying the Bayes filter technique to the proposed classifiers. It reveals that all classifiers are not influenced by high-frequent terrain changes in a significant manner when embedded into our Bayes filter prediction scheme.

6 Conclusion

In this paper, we systematically investigated the applicability of several posterior probability estimation techniques in the context of terrain classification based on temporal coherence. We exploited temporal coherence using a Bayes filter approach which takes several recent terrain class predictions into account. Depending on the choice of the classifier and the distance, a robot has to navigate over a certain terrain type before a terrain transition occurs, the classification performance increased by up to 6.9%. This number denotes the increase of classification performance related to a classification approach based on individual observations only. We showed that the Bayes filtering approach was nearly always superior to the single-observation approach with the only exception of the KNN classifier at a travel distance of less than or equal to 0.6 m. The significantly best experimental results were obtained using a combined support vector machine and k-nearest neighbor approach which has not been employed in the domain of terrain classification so far. Further investigation revealed that the various classifiers did not only differ in classification performance but also in the confidence of erroneous predictions. In the context of Bayesian filtering this is an important issue since a decrease in this confidence results in a decreased influence of wrong predictions on the final classification.

As a further contribution we examined the proposed classifiers with respect to their limiting factors such as storage requirements, prediction times, model generation times, and model selection times. The results provide criteria for choosing an appropriate classifier for a variety of hardware configurations.

References

1. Angelova, A., Matthies, L., Helmick, D.M., Perona, P.: Fast terrain classification using variable-length representation for autonomous navigation. In: Proceedings of the Conference on Computer Vision and Pattern Recognition, Minneapolis, MN, USA, pp. 1–8 (2007)
2. Bishop, C.: Neural Networks for Pattern Recognition. Oxford University Press, Oxford (1995)
3. Brooks, C.A., Iagnemma, K.: Vibration-based terrain classification for planetary exploration rovers. IEEE Transactions on Robotics 21(6), 1185–1191 (2005)
4. Burges, C.J.C.: A tutorial on support vector machines for pattern recognition. Data Mining and Knowledge Discovery 2, 121–167 (1998)
5. Cunningham, P., Delany, S.: k-nearest neighbour classifiers. Technical report, UCD School of Computer Science and Informatics (2007)
6. DuPont, E.M., Moore, C., Roberts, R.G.: Terrain classification for mobile robots traveling at various speeds: An eigenspace manifold approach. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2008), pp. 3284–3289 (2008)
7. DuPont, E.M., Moore, C.A., Collins Jr., E.G., Coyle, E.: Frequency response method for terrain classification in autonomous ground vehicles. Autonomous Robots 24(4), 337–347 (2008)
8. Iagnemma, K., Dubowsky, S.: Terrain estimation for high-speed rough-terrain autonomous vehicle navigation. In: Proceedings of the SPIE Conference on Unmanned Ground Vehicle Technology IV, Orlando, FL, USA (2002)
9. Komma, P., Weiss, C., Zell, A.: Adaptive bayesian filtering for vibration-based terrain classification. In: IEEE International Conference on Robotics and Automation (ICRA 2009), Kobe, Japan, May 2009, pp. 3307–3313 (2009)
10. Lalonde, J.-F., Vandapel, N., Huber, D.F., Hebert, M.: Natural terrain classification using three-dimensional ladar data for ground robot mobility. Journal of Field Robotics 23(10), 839–861 (2006)
11. Manduchi, R., Castano, A., Talukder, A., Matthies, L.: Obstacle detection and terrain classification for autonomous off-road navigation. Autonomous Robots 18, 81–102 (2005)
12. Platt, J.: Advances in Large-Margin Classifiers. chapter: Probabilities for SV Machines, pp. 61–74. MIT Press, Cambridge (2000)
13. Specht, D.: Probabilistic neural networks. Neural Networks 3(1), 109–118 (1990)
14. Thrun, S., Burgard, W., Fox, D.: Probabilistic Robotics. In: Intelligent Robotics and Autonomous Agents, MIT Press, Cambridge (2005)
15. Vandapel, N., Huber, D., Kapuria, A., Hebert, M.: Natural terrain classification using 3-d ladar data. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2004), New Orleans, LA, April 2004, pp. 5117–5122 (2004)
16. Weiss, C., Fechner, N., Stark, M., Zell, A.: Comparison of different approaches to vibration-based terrain classification. In: Proceedings of the 3rd European Conference on Mobile Robots (ECMR 2007), Freiburg, Germany, pp. 7–12 (2007)
17. Weiss, C., Zell, A.: Novelty detection and online learning for vibration-based terrain classification. In: Proceedings of the 10th International Conference on Intelligent Autonomous Systems (IAS 2008), Baden-Baden, Germany, pp. 16–25 (2008)

18. Wilcox, B.H.: Non-geometric hazard detection for a Mars microrover. In: Proceedings of the AIAA/NASA Conference on Intelligent Robots in Field, Factory, Service and Space, Houston, TX, USA, vol. 2, pp. 675–684 (1994)
19. Wu, T.-F., Lin, C.-J., Weng, R.C.: Probability estimates for multi-class classification by pairwise coupling. Journal of Machine Learning Research 5, 975–1005 (2004)
20. Yianilos, P.N.: Data structures and algorithms for nearest neighbor search in general metric spaces. In: Proceedings of the fourth annual ACM-SIAM Symposium on Discrete algorithms (SODA 1993), Philadelphia, PA, USA, pp. 311–321 (1993)
21. Zhang, H., Berg, A., Maire, M., Malik, J.: SVM-KNN: Discriminative nearest neighbor classification for visual category recognition. In: Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2006), Washington, DC, USA, pp. 2126–2136 (2006)