IMPLEMENTATION OF AN ON-BOARD TERRAIN CLASSIFIER BASED ON PROPRIOCEPTIVE SENSOR DATA FOR A PLANETARY ROVER

Raul Dominguez[1], Lennart Kuhr[2], Jonathan Babel[1], Florian Cordes[1], Giulio Reina[3], and Frank Kirchner[1,4]

[1]*DFKI Robotics Innovation Center Bremen Robert-Hooke-Str. 1, 28359 Bremen, Germany,*
E-mail: name.surname@dfki.de
[2]*Institute of Space Systems, TU Braunschweig, Herman-Blenck-Straße 23, 38108 Braunschweig, Germany,*
E-mail: l.kuhr@tu-braunschweig.de
[3]*Department of Mechanics, Mathematics and Management, Polytechnic of Bari, Via Orabona 4, 70125, Bari, Italy,*
E-mail: giulio.reina@poliba.it
[4]*Robotics Research Group, University of Bremen, Germany*

ABSTRACT

The implementation of a Support Vector Machine based terrain classifier for the hybrid locomotion rover SherpaTT is presented. In the first phase of classification the physical characteristics of the traversed terrain are statistically derived from proprioceptive data (i.e. engineered features). The features are then used by the classifier to distinguish between three different surface types: *sand*, *compact sand* and *concrete*. Based on previous offline studies [7] the terrain classifier has been integrated into the control architecture of the rover, as well as deployed and tested in analog environment. The software component runs completely on the on-board computer (OBC) of SherpaTT, embedded within the Robotics Construction Toolkit (Rock)[1] framework. Insights on the implementation and the software architecture surrounding the classifier are provided. Performance metrics demonstrate that the terrain classifier can run in parallel with the rest of the control software on the OBC, achieving an overall high accuracy in terrain classification.

1. INTRODUCTION

Planetary exploration missions are so far dominated by wheeled rover designs like Curiosity or Perseverance [11, 15]. Although wheeled locomotion is most energy-efficient over flat terrain, it comprises drawbacks when exposed to more demanding unstructured terrain. Especially in environments with steep, sandy slopes and boulders patches, wheeled systems reveal their limitations [9]. In the past, several high slip and excessive sinkage events have been encountered with exploration rovers, which have severely disrupted mission timelines [8]. It took five weeks to free the Opportunity rover from sand in 2006 [16] and rover trajectories need to be frequently adjusted to avoid challenging terrain [1]. The potentially worst situation occurred in 2009, when the Spirit rover got stuck in sand and was unable to recover, ultimately ending the mission [14]. Terrain awareness, the correct modelling of transited surfaces and its classification is a key factor for reliable autonomous navigation. Surface modelling can be used for navigation in order to avoid operational problems like those previously described. Moreover, terrain awareness may enhance navigation capabilities, if the drive settings are adapted in accordance to the terrain properties.

In this publication the authors introduce a software component capable of classifying three different terrain types based on the proprioceptive sensor data of the SherpaTT rover. The component uses a Support Vector Machine (SVM) algorithm [2, 6] at its core. It is integrated into the software control architecture of the mobile exploration rover SherpaTT and embedded into the Robotics Construction Toolkit (Rock) framework, such that it can be executed sufficiently fast during navigation. The terrain classifier uses force torque sensors, joint data and body acceleration estimates to classify the surface into one of three terrain types: *sand, compact sand* and *concrete*. The three types represent distinct classes of surfaces characterized by its deformability and friction properties. In order to achieve a better classification performance as well as a more in-depth characterization of the surface patches, a feature calculation process is executed prior to the classification.

After introducing the main background concepts concerning this work in Section 2, Section 3 explains the challenges faced throughout the implementation of the terrain classifier and the applied solutions. In Section 4 the evaluation of offline and on-board performance of the terrain classifier is described. Finally, Section 5 concludes the paper with a summary and gives an outlook for future work.

[1]The Robot Construction Kit (`http://rock-robotics.org`)

2. BACKGROUND

2.1. SherpaTT

SherpaTT is a hybrid wheeled-leg rover with an actively articulated suspension system. Its locomotion control system provides the basis for advanced locomotive capabilities with the ability to adapt to different terrain types [4]. The rover is designed to operate in unstructured environment. It features terrain adaption based on force torque sensors inputs, trajectory control and path planning as well as environment reconstruction through data fusion of extereoceptive and proprioceptive data. The placement of the sensors that provide proprioceptive input data to the terrain classifier on SherpaTT is shown in Figure 1a. The purpose of the high mobility platform is to access scientifically interesting areas on planetary bodies in a safe and autonomous way. Besides the enhanced locomotion system, SherpaTT benefits of its advanced motion control system (MCS). The MCS guarantees controlled articulation of the complex kinematic suspension system at any time. It is implemented as a middle-layer between high-level processes and the low-level hardware layer, as shown in Figure 1b. The layered architecture of SherpaTT can be described as [4]:

- **High-Level:** the autonomy level of the robot's control system. Resembles i.a. navigation, mapping and path planning software. Related behaviors are independent of the physical states of the robot.
- **Middle-Layer:** resembles an abstraction of the physical robot. Handles command inputs from the operator or high-level processes and calculates according joint movements. It reacts to sensor inputs and thereby assures system stability for example by monitoring the position of the Center of Gravity (CoG) to avoid tip-over.
- **Low-Level:** close to the physical hardware. This comprises sensors and local joint controllers, for example PID controllers for position or velocity control of individual joints. Besides providing commands to the wheel drive actuators, the MCS design aims to mimic a passive suspension with controllable properties.

2.2. Development Tools

The control and perception software on SherpaTT exchanges data and information through the Robotics Construction Toolkit (Rock) framework. The framework not only provides wrapping mechanisms for the libraries and its communications, but also offers the possibility to enforce realtime execution of the control loops, when using a realtime supporting Linux kernel. Rock incorporates a set of useful tools for development, testing and evaluation of software libraries (e.g. runtime data exchange statistics and visualization). In particular, tools for data logging, replaying and memory-signature based data selection from data streams have been employed in this implementation.

Data selection at runtime is one of the key features addressed by the middleware. The problem consists in generating matrices of synchronous sample values for classification online from data streams with multiple frequencies. Each data stream relevant for the classification contains in addition to the relevant data, data which has to be filtered out (e.g. timestamps) since it is useless for the classification. The library *type to vector*[2] is used to select the relevant fields of data types in the samples of the data stream. It uses memory type descriptions of the samples which contain relevant data to select from the incoming streams only the relevant attributes and stack them into the input matrix.

The synchronicity of the input data to the classifier is achieved by labeling sensor data with timestamps at acquisition time. Data streams can have different update rates, but the input matrices need data from multiple sources at the same rate, therefore a strategy is needed to discard or interpolate data.

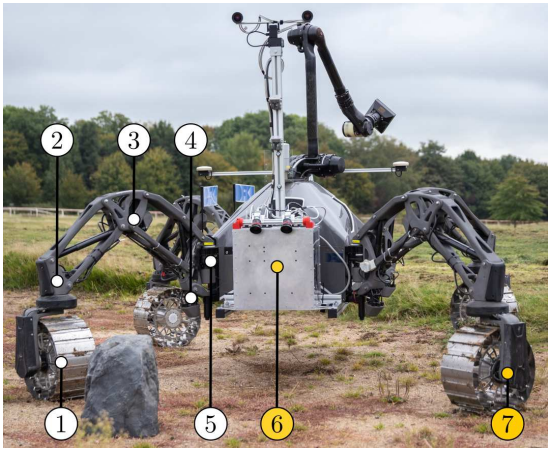
The Rock logging mechanism is used to store the sensor and fused data streams on a hard drive for later offline use (i.e. data collection) while traversing. Two further tools are employed to process the logged data offline: *rock-replay* for testing the runtime functionality on the development environment and *pocolog to msgpack*[3] to select and prepare the data, so that it can be employed for training and testing the classifier. *pocolog to msgpack* converts data from its Rock binary representation to the almost universal *msgpack* format. Moreover, *pocolog to msgpack* allows for the conversion of logged data into Python data frames, the widely used data science and machine learning format. Finally, the runtime inspection tools of Rock are utilized to monitor the processing time of the classifier, the connections between separate software components and to visualize the coherence of all input data streams.

2.3. Terrain Classifier

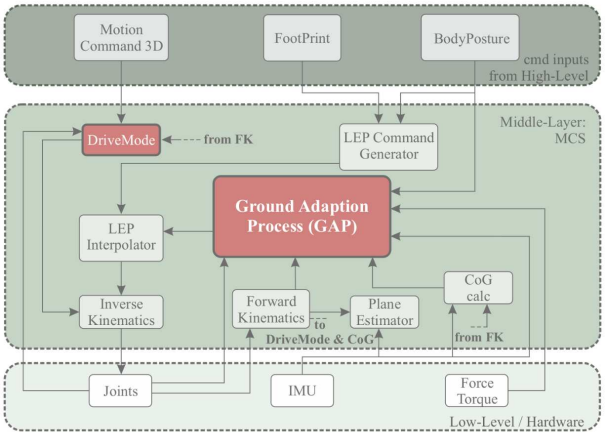
In order to classify three terrain types using proprioceptive sensor data, the supervised machine learning algorithm Support Vector Machine (SVM) is utilized [5]. The application requires the training of a SVM classification model that is later used to classify n-dimensional data points. The model's dimensions are called *features* and they correspond to the inputs that are used to estimate the class. During training the SVM adapts the boundaries among the different classes. The greater the separation between the classes, the clearer the classification of

[2]Type to vector: `https://github.com/rock-data-processing/data_processing-orogen-type_to_vector`

[3]Pocolog to Msgpack: `https://github.com/rock-core/tools-pocolog2msgpack`



(a) Sensor and joint inputs to the Terrain Classifier



(b) Motion Control System of SherpaTT

Figure 1: (a) Input features for the terrain classifier originate from the following actuators (1)-(5) and sensors (6)-(7) on SherpaTT: (1) wheel drive, (2) wheel steering, (3) linear knee, (4) linear hip, (5) pan hip, (6) IMU, (7) Force-Torque. (b) Simplified layout of the Motion Control System of SherpaTT [3].

data points will be. In order to achieve a maximum separation between data classes, the optimal features, computed from the sensor data, need to be identified. In other words, a feature selection process allows to identify the most critical set of input data to the classifier.

Model Training Generally, a SVM aims to separate data into multiple classes. In geometrical terms it divides the data with a n-dimensional hyperplane, the *decision boundary*. The hyperplane parameters, or *weights*, are iteratively optimized to allow for the largest distance between the hyperplane and the nearest data points which eventually are the support vectors. By changing the dot-product calculation, referred to as the kernel trick, within the underlying geometrical condition various correlations of the data's dimensions can be recognized. Correlation can be obtained through a polynomial or a Gaussian radial based function. However, the most simple option is linear correlation. The generated set of weights is referred to as *classification model* and the generation process as such is called *training*.

Model Testing Before deploying the model, its classification performance can be measured using labeled data. This testing process obtains performance measures that are typically visualized with *confusion matrices*. A description of the three class confusion matrix used to present the results in this work is depicted in Figure 2, where T stands for True (correct) and F for False (wrong). The three evaluated aspects or performance measures are *precision*, *recall* and *accuracy*. The step of processing a sample and providing a classification label is also referred to as *prediction*.

- **Accuracy** measures the overall percentage of correctly classified samples:

$$\frac{T_{\text{overall}}}{(T+F)_{\text{overall}}} \tag{1}$$

- **Recall** measures for each set of samples of each class the correctly classified percentage:

$$\frac{T_{\text{class}}}{(T+F)_{\text{actual class}}} \tag{2}$$

- **Precision** measures for each set of predictions per class the percentage of correct predictions:

$$\frac{T_{\text{class}}}{(T+F)_{\text{predicted class}}} \tag{3}$$

Feature Selection For computational reasons and simplicity, it is desired to reduce the dimensionality by selecting the most critical features for the classification task. A large set of statistical moments of the features is considered within the selection process. The features can represent direct sensor parameter as well as physical parameters like mechanical and electrical power, friction coefficients and the wheel speed deviation of each wheel in relation to the others. The proprioceptive sensor data used to calculate the features is listed in Table 1 which is based on [7]. The input data is received from the Motion Control System (MCS), Joint Deployment (JD) and Sensors Deployment (SD). All forces and torques are represented within the Body Coordinate System (BCS).

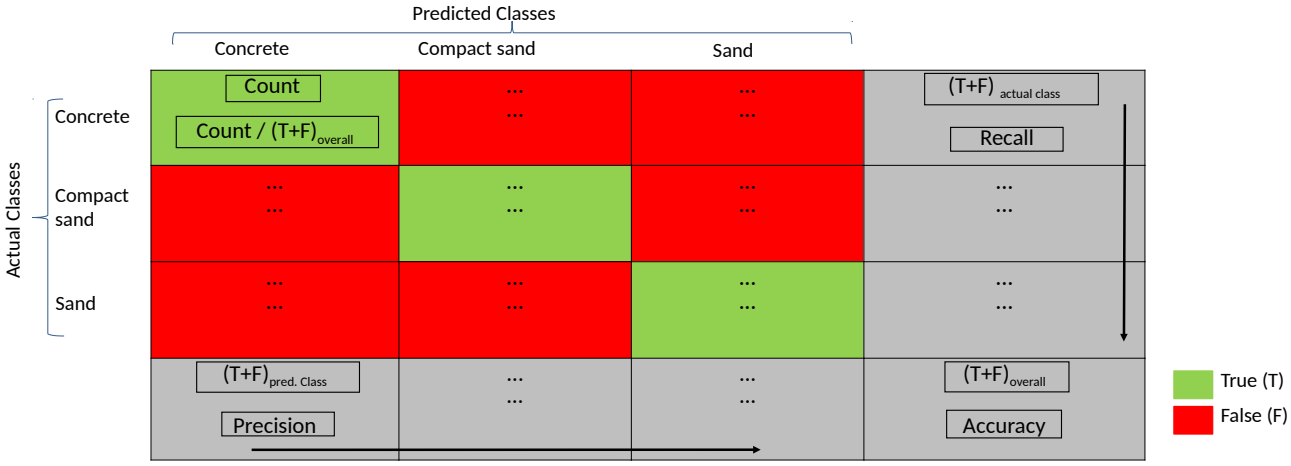


Figure 2: Confusion matrix used to visualize the performance measures, *Precision*, *Recall* and *Accuracy*, of the classifier [10].

Datastream	Feature	Symbol
MCS	Longitudinal Force	F_x
MCS	Vertical Force	F_z
MCS	Drive Torque y-axis	T_y
JD	Motor Current	I
JD	Voltage	V
JD	Dutycycle	d_{pwm}
JD	Angular Wheel Velocity	w
SD	Acceleration X	a_x
SD	Acceleration Z	a_z

Table 1: Proprioceptive inputs and supplying datastream.

Statistical	Feature	Symbol
M,SD	Longitudinal Force	F_x
M,SD	Drive Torque around y-axis	T_y
M,SD	Drive Current	I
M,SD	Acceleration X	a_x
M,SD	Acceleration Z	a_z
M,SD	Mechanical Power	P_m
M,SD	Electrical Power	P_e
M,SD	Friction Coefficient 1	μ_1
M,SD	Friction Coefficient 2	μ_2
M,SD	Friction Coefficient 3	μ_3
M	Angular Wheel Velocity	w
SD	Speed Deviation	Δw

Table 2: Optimal feature set for the classification of terrain types by SherpaTT. The set includes the statistical calculation of mean (M) and standard deviation (SD) per terrain patch.

Using the inputs in Table 1, the physical features listed in Table 2 are statistically determined for each patch of terrain to be classified. The statistical moments, mean, variance, skewness and kurtosis are initially evaluated. A selection of the most useful features for the classification is done using the WB index and the Pearson coefficient to reduce the dimensionality of the data. The equations for physical features calculation and for most critical features selection are both detailed in [7].

3. IMPLEMENTATION

The classifier control loop has a target frequency of 100 Hz, matching the highest input frequency (joint status). A zero order hold is used for data streams with a lower update frequency. The classification is performed on matrices composed with information generated from 100 samples. Thus, a classification is generated every second. In other words, every 100 samples a matrix input for the classifier is completed and a new classification is performed, which happens every 1 s. Since the speed of the rover in the analyzed data sets is 0.1 m/s, the resolution of the categorized patches of terrain is 10 cm long.

The classification results are required frequently to allow other on-board components take advantage of the results (e.g. to improve navigation) and to ensure that the classification result corresponds to the currently traversed surface. Likewise, the loss of data samples due to full queues on the input of the processing components needs to be avoided.

The diagram in Figure 3 presents the implementation approach of the terrain classifier in Rock. The first step, the extraction of a data sample from the relevant fields of data, is done using the *type to vector* Rock tool and needs to be performed efficiently At each execution of the *updateHook* (every 0.01 s) new data is stacked in the input matrix. Every hundred *updateHook* executions, an input matrix is finished and the component triggers the computation of the physical and statistical features as well as the classification itself, the *prediction*. The output of the classification and the computed features are delivered to the calling component which retrives the values through

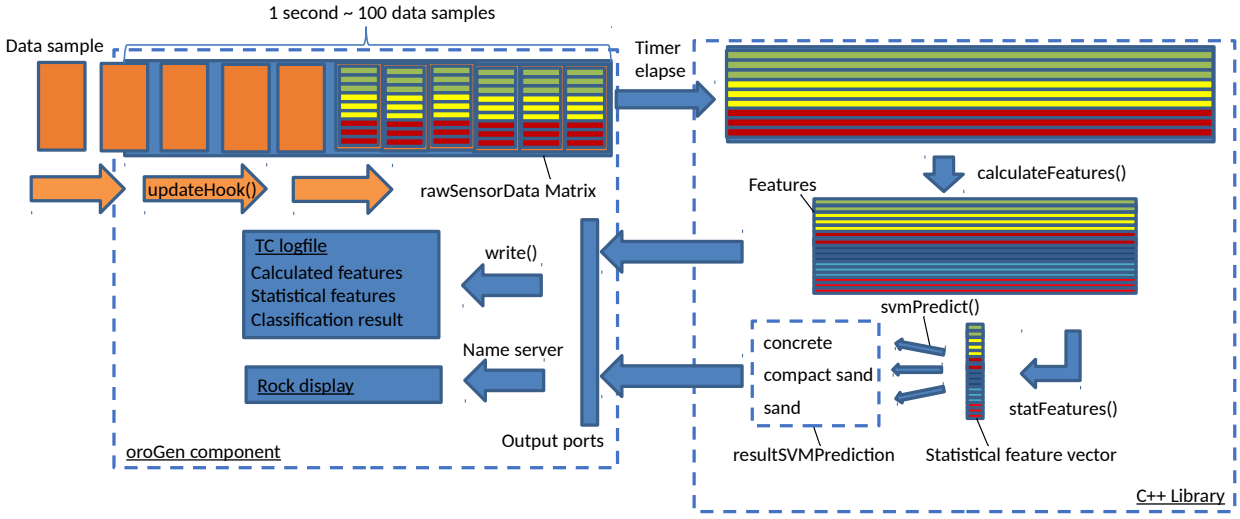


Figure 3: Overview of the terrain classifier library and its integration into Rock.

(a) loose sand (b) compact sand (c) concrete

Figure 4: The test locations where the data sets were acquired.

its output ports.

3.1. Training Data

An essential requirement for a good classification performance is the availability of a consistent and large data collection for training and testing. The data collection available for this implementation consists of a dataset composed of traverses that are acquired by remotely commanding the rover over the three examined terrain types: *loose sand*, *compact sand* and *concrete*. One traverse in the dataset corresponds to driving 10 m forwards and backwards. The following conditions are kept consistent during all traverses: (1) fixed wheel configuration, (2) surfaces without inclination, (3) traverse speed of 0.1 m/s, (4) straight traverses and (5) the electric power generator is running, which causes vibrations. Figure 4 shows images of the rover traversing the different terrains during the data collection.

In terms of quantity, the data collection consists of 2200 training samples for each of the 76 features. Regarding data balance, the data gathered from the terrain type loose sand represents 28%, whereas concrete and compact sand each represent 36% of the total amount of data. When generating a SVM model, the collected data is divided into training and testing sets. For the presented classifier, one of the traverses of each terrain is used for testing and all other traverses are used for training. This yields a percentage of about 75% training data and 25% testing data.

4. EVALUATION

4.1. Offline Classification Performance

The components of a Linear Discriminant Analysis (*LDA*) are plotted to visualize the achieved data separation. In the LDA two Linear Discriminant components are represented on the axes. These components are formed from a reduction of the original feature set. Figure 5 shows a LDA plot of the dataset and highlights the decision boundary of the trained linear SVM kernel. The

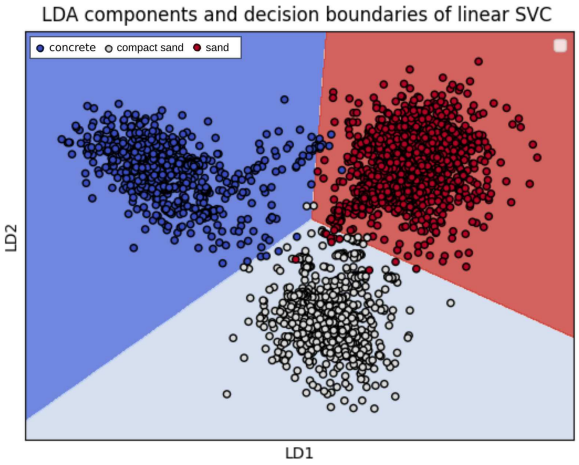


Figure 5: Linear Discriminant Analysis of the dataset. The decision boundaries of linear SVC for three examined terrain types are highlighted with colors: concrete (blue), compact sand (grey) and loose sand (red).

areas of different terrain types are highlighted by the corresponding colors. It is shown that the data of the terrain types can be separated for most of the samples.

The offline evaluation of the SVM classifier reaches an accuracy of 93.97%, depicted in the confusion matrix in Figure 6. A small portion of the collected data where SherpaTT was not moving was manually removed. Based on the sliced data the classification accuracy is increased by 2%. Except the recall for *compact sand* and the precision for *sand* all performances are above 90%, with an overall accuracy of 93.97%.

4.2. On-Board tests

Two main tests were performed to validate the on-board feature calculation and classification performance. The software components were running in parallel with the rest of the control and perception software while the rover was traversing the test area.

The first test was excecuteed indoors at the DFKI Robotics Innovation Center premises, depicted in Figure 7. During the test, the classifier ran at the pursued frequency of 100 Hz. Nevertheless, the classification performance of these tests is not considered representative, since the power generator was not active and the hardest surface was not as firm as concrete.

The second test took place during the final field trials of the ADE project [12] in a sand mine in Wulsbüttel, Northern Germany. These test runs provided comparable conditions with the ones within the training data set, because the power generator was activated. Several traverses of the SherpaTT rover were logged and checked for consistency to validate the feature calculation. The classification accuracy reached 87%. Well balanced recall and

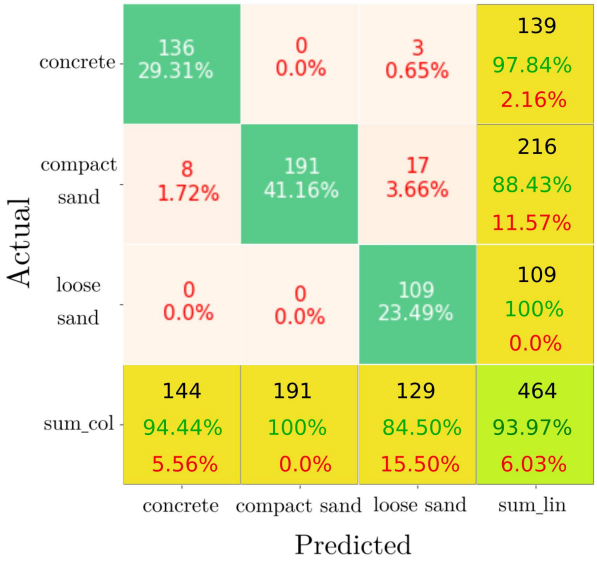
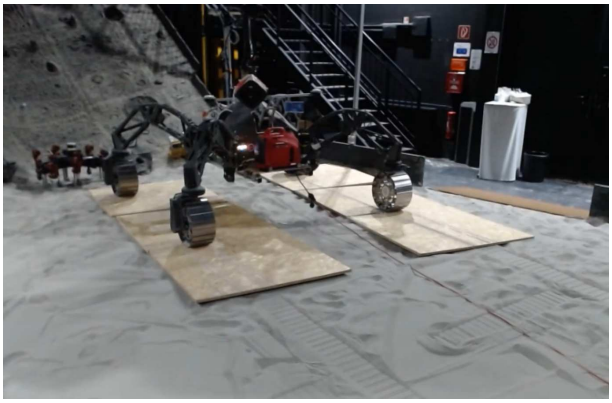


Figure 6: The confusion matrix of the terrain classifier shows the performance of the SVM model with respect to the different terrain types. The overall accuracy yields 93.97%

(a) Loose Sand Tests



(b) Concrete Tests

Figure 7: Validation of execution frequencies for the on-board terrain classifier.

precision values of the classes were also achieved. The underperformance in the field tests can be explained by the encountered surface conditions, which did not closely match any of the previously examined surface types. Due to rainfalls, the surface turned into wet compact soil and sticked to the wheels as shown in Figure 8, causing unaccounted dynamics. Nevertheless, the classification resulted in 87.69% *concrete* and 12.31% *compact sand*, complying with the closest types of terrain the classifier was trained with. The field trials demonstrated that the terrain classifier can be executed on-board of SherpaTT, that it is able to compute correct features and to classify different terrain types successfully while the rover is traversing a surface.

Figure 8: The analog site where the classifier was tested.

4.3. Computational Performance

The execution time of the code has been repeatedly measured. Since the computation is executed on a single thread, the execution time can be identified measuring the averaged wall time of the code execution. The resulting execution time depends on the threading of the operating system which in this case is the Linux distribution Ubuntu 18.04 LTS. The time measures were taken on an i7 processor with a CPU clock speed of 4.6 GHz. Table 3 shows the results of these measurements.

	Wall Time $[ms]$		
Method:	min.	max.	avg.
calculateFeatures():	9	17.1	13.2
calculateStat():	6.3	13	7.1
svmPredict():	0.0	0.001	0.0007
overall:	15.3	30.1	20.3

Table 3: Wall time measurements of the methods of the C++ classification library.

As the averaged execution time of the C++ classification library is 20.3 ms this can lead to a delay of the next data collection step which is repeated every 10 ms and hence can cause the drop of one data sample per second. The drop of one data sample out of the one hundred data samples that are strapped every second is assessed to be acceptable.

5. CONCLUSIONS

The presented work explains the implementation of a terrain classifier, which has been deployed and tested on-board of the hybrid locomotion robotic platform SherpaTT. It has been shown that the SVM classifier provides useful results and can be run on-board along with the rest of the software components.

The combined classification results on collected test and training data yield 93% of overall accuracy. The offline results achieved with SVM have been recently improved with Deep Learning techniques [13], but this approach has not been deployed on-board of the system yet. Thus, the authors aim to also integrate and test the terrain classifier with Deep Learning in the near future on-board of the rover. During field tests a new type of surface was encountered that did not corresponded to any known class by the classifier. Nevertheless, the two closest surface types were selected, which the authors interpret as a robust response. The approach could be further improved by combining it with an unsupervised learning technique to automatically identify anomalies and potentially generate new types of surfaces.

Applications of the terrain classifier include contributions to environment modelling while the rover is traversing a surface and the use of the identified terrain class to adapt various navigation settings. Besides the class of terrain, the module computes physical properties of the surface and yields valuable environmental information. These features can be exploited in future missions to predict errors in localization, e.g. due to different friction coefficients or to generate more realistic contact simulations in order to further improve the control of the system. The terrain type has to be taken into account, when setting costs for multiple potential paths traversing different regions. For instance, paths over a slope of certain inclination may be traversed if the surface is composed of a material with high friction, but the same task could become very challenging if the friction coefficient on that surface is low.

ACKNOWLEDGEMENTS

The financial support of the projects RoBivaL (Robot Soil Interaction Evaluation in Agriculture), funded by the Federal Ministry for Economic Affairs and Climate Action (Grant no. 50RP2150), and OG10-ADE (Autonomous Decision making in Very Long Traverses), funded by the European Union as part of Horizon 2020 (Grant no. 821988), in which this research was conducted, is gratefully acknowledged.

REFERENCES

[1] Arvidson, R. E., Iagnemma, K. D., Maimone, M., et al. 2017, Journal of Field Robotics, 34, 495

[2] Boser, B. E., Guyon, I. M., & Vapnik, V. N. 1992, in Proceedings of the Fifth Annual Workshop on Computational Learning Theory, COLT '92 (New York, NY, USA: Association for Computing Machinery), 144–152

[3] Cordes, F. 2018, publisher: Unpublished

[4] Cordes, F., Kirchner, F., & Babu, A. 2018, Journal of Field Robotics, 35, 1149

[5] Cortes, C. & Vapnik, V. 1995, Machine Learning, 20, 273

[6] Cristianini, N. & Shawe-Taylor, J. 2000, An Introduction to Support Vector Machines and Other Kernel-based Learning Methods (Cambridge University Press)

[7] Dimastrogiovanni, M., Cordes, F., & Reina, G. 2020, Applied Sciences, 10

[8] Gonzalez, R. & Iagnemma, K. 2018, Journal of Field Robotics, 35, 564

[9] Kolvenbach, H. 2021, PhD thesis

[10] Kuhr, L. 2021, Master's thesis, Institute of Space Systems, Technical University of Braunschweig

[11] Moeller, R. C., Jandura, L., Rosette, K., et al. 2021, Space Science Reviews, 217, 5

[12] Ocón, J., Dragomir, I., Cordes, F., et al. 2021, in In Proceedings of the 72nd International Astronautical Congress, (IAC-2021) (Dubai, United Arab Emirates: International Astonautical Federation (IAF))

[13] Ugenti, A., Vulpi, F., Domínguez, R., et al. 2021, Journal of Field Robotics, n/a

[14] Webster, G. & McGregor, V. 2009, NASA's Mars Rover has Uncertain Future as Sixth Anniversary Nears

[15] Welch, R., Limonadi, D., & Manning, R. 2013, in 2013 8th International Conference on System of Systems Engineering (Maui, HI, USA: IEEE), 70–75

[16] Young, K. 2006, Mars rover escapes from the bay of lamentation