Terrain Classification for a Quadruped Robot

Jonas Degrave*, Robin Van Cauwenbergh*, Francis wyffels*, Tim Waegeman*, and Benjamin Schrauwen*
*Electronics and Information Systems (ELIS), Ghent University
Sint-Pietersnieuwstraat 41, B-9000 Ghent, Belgium

***Abstract*—Using data retrieved from the Puppy II robot at the University of Zurich (UZH), we show that machine learning techniques with non-linearities and fading memory are effective for terrain classification, both supervised and unsupervised, even with a limited selection of input sensors. The results indicate that most information for terrain classification is found in the combination of tactile sensors and proprioceptive joint angle sensors. The classification error is small enough to have a robot adapt the gait to the terrain and hence move more robustly.**

***Keywords*—terrain, classification, quadruped robot, reservoir computing, proprioception**

I. Introduction

Terrain classification plays an important role in the control of legged robots, as it allows the robots to adapt to the terrain. On different terrains, different gaits will be more suitable and therefore a robot capable of switching from one gait to another in reaction to a terrain change, will be able to locomote more robustly.

There has been done some research on terrain classification for robots through sensor data from advanced sensors, such as camera imagery [1] or laser scanners [2]. This way, visual features in the terrain are used to discover the terrain type and subsequently the terrain properties.

Previous research also demonstrated the importance of proprioceptive sensors for amphibian robots, such as inertia-sensors, angle-encoders and current measurements on the motors [3]. Furthermore has it been shown for quadruped robots that force-sensing in the legs and current-use in the motors deliver reasonable results upon processing with an Adaboost algorithm [4]. Even only using proprioceptive and contact sensors proved effective in ground discrimination [5].

Firstly, perceiving and understanding the environment in which the robot operates has a high impact on the performance of the robot's locomotion, making it important to add sensors to the robot that provide information on the terrain. However, it is unfavorable to add unnecessary or complex sensors to the robot if they yield no further information. Those superfluous sensors would only increase the complexity of the robot design, while offering little possibilities for better control. Hence, it is important to know which sensors actually provide the most valuable data for terrain classification.

Secondly, instead of developing a single system which takes in raw data to directly determine the robot's actions, it can be better to divide the problem and conquer the easier sub-problems. One of the sub-problems is for the robot to recognize in real time the type of terrain while walking on it, using the data it receives from as few sensors as possible. Previous research in this area includes for instance the application of clustering techniques to detect transitions from one terrain to the next in order to achieve unsupervised classification, applied on the RHex-robot [6], [7].

Therefore the research in this paper is twofold. Firstly, we want to identify which sensors provide most information on the terrain. In order to achieve this, we try different combinations of sensors often found in robots and evaluate the capability of supervised and unsupervised machine learning techniques to derive information from recorded data of those sensors. Secondly, we will also evaluate which machine learning techniques work best to classify the terrain based on these sensors, and which features are necessary for the techniques to function.

The rest of the paper is structured as follows. In Section II we will review the machine learning techniques used in this paper and go through Linear Regression (LR), Extreme Learning Machines (ELM), Reservoir Computing (RC), Slow Feature Analysis (SFA) and Independent Component Analysis (ICA). In Section III we will present the hardware used to retrieve the data and the methods used to process this data. In section IV we will describe our experiments and results. Finally, conclusions will be drawn in section V.

II. Machine Learning Techniques for Terrain Classification

A. Linear Regression

Linear regression (LR) is a supervised approach to modeling the relationship between K input variables and L scalar output variables. The relation between the two is described as:

$$\mathbf{W_{out}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}.$$

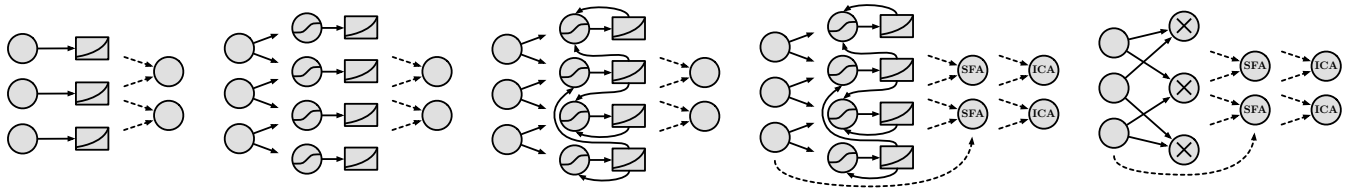
$\mathbf{X}$ is a matrix where every row is a time-step with in the columns the different input variables and $\mathbf{y}$ is a matrix and has in every row the corresponding desired outputs. With the transformation matrix $\mathbf{W_{out}}$ we can now process new signals $\mathbf{X}'$:

$$\hat{\mathbf{y}} = \mathbf{X}'\mathbf{W_{out}}. \tag{1}$$

The sensor signals are very noisy however, therefore we first low-pass filter these inputs with an exponential moving average (see Figure 1a), as shown in the equation below:

$$\mathbf{x}(n) = (1-\alpha)\,\mathbf{x}(n-1) + \alpha\,\mathbf{u}(n). \tag{2}$$

Here, $\mathbf{x}(n)$ and $\mathbf{u}(n)$ are respectively the input of the linear regression and the sensor signal at time-step n. α is the leak rate. As the resulting signals still contains a lot of noise, no further regularization is needed.



(a) Linear Regression (b) Extreme Learning Machine (c) Reservoir Computing (d) Reservoir Computing Slow Feature Analysis (e) Quadratically Expanded Slow Feature Analysis

Fig. 1. Visual representation of the machine learning techniques used. The circles depict nodes, the rectangles filtering. Circles with a hyperbolic tangent curve have a non-linear activation; circles with a cross multiply inputs; rectangles with an exponential curve are low-pass filters. The solid arrows represent fixed weights, the dashed arrows represent trained weights.

B. Extreme Learning Machine

With the goal of having LR model the non-linearities more accurately, we first expand the sensor signals into a larger space by adding a hidden layer of non-linear nodes. The weights of these nodes are fixed, and are randomly selected from the set $\{-1, 0, 1\}$. This technique is similar to Extreme Learning Machine (ELM), a technique to train single-hidden layer feedforward neural networks [8]. The only difference from the standard implementation is that we use leaky nodes in the hidden layer to filter the noise in their state, as we did earlier with the linear regression. The update equations for these leaky ELMs are therefore given by:

$$\mathbf{x}(n) = (1-\alpha)\,\mathbf{x}(n-1) + \alpha\,\tanh\left(\mathbf{W_{in}}\mathbf{u}(n)\right)$$

$$\mathbf{y}(n) = \mathbf{W_{out}}\mathbf{x}(n).$$

Here, $\mathbf{W_{in}}$ is an $N \times K$ matrix containing the fixed weights of the N nodes in the hidden layer. We model the non-linearity by using hyperbolic tangent nodes, as shown in Figure 1b. The matrix $\mathbf{W_{out}}$ is obtained by using LR, as discussed in section II-A, but this time with the expanded set of signals as inputs.

C. Reservoir Computing

The ELM systems can be expanded further temporally and non-linearly by adding fixed weight connections between the nodes in the hidden layer, as shown in Figure 1c. The resulting system is a Reservoir Computing system (RC). The term Reservoir Computing has been introduced in [9] to cover multiple previous computing techniques developed independently: Liquid State Machines (LSM) [10], Echo State Networks (ESN) [11] and BackPropagation DeCorrelation (BPDC) [12].

The weights of recurrent connections in the hidden layer nodes are fixed and randomly selected from a standard normal distribution. Similarly to the ESN in [13], the update equations of our reservoir computing systems are as follows:

$$\mathbf{x}(n) = (1-\alpha)\,\mathbf{x}(n-1) + \alpha\,\tanh\left(\mathbf{W_{res}}\mathbf{x}(n-1) + \mathbf{W_{in}}\mathbf{u}(n)\right)$$

$$\mathbf{y}(n) = \mathbf{W_{out}}\mathbf{x}(n).$$

$\mathbf{W_{res}}$ is an $N \times N$ matrix containing the fixed weights between the nodes in the hidden layer. After sampling the weights, the matrix is rescaled to have a spectral radius σ. Typically, σ is a good indicator of the echo state property [14] and is often chosen proximate to 1, close to the edge of chaos, where reservoir computing systems possess high computational power [15].

In Reservoir Computing, the hidden layer of recurrent nodes is often referred to as the reservoir. Because of the recurrent connections between the nodes, a fading memory is introduced into the system. We already inserted some memory with leaky nodes, but the hidden layer nodes in an ELM cannot act dynamically at a certain time-step based on the result of the previous time-step. Therefore, the type of memory used in LR and ELM does not add dynamics, but merely serves as a noise filter, opposed to the dynamic properties in a reservoir. Consequently, RC has memory capacity [16] as RC systems can learn relations with the past, while ELM has no memory capacity due to the lack of recurrent connections between the hidden-layer nodes.

D. Unsupervised learning

It would be interesting for an autonomous learning robot to have it learn terrains autonomously as well, without being shown a distinction between terrains in advance. To achieve this, we continue from the non-linear, temporal expansion created by the reservoir, as the dynamics generated in a reservoir are suited for terrain classification, albeit supervised, which we will establish in section IV-A2.

However, to make the system unsupervised, we process the output of the reservoir further with unsupervised techniques instead of using LR. First, we apply slow feature analysis (SFA) to derive the slow changing features in the output of the reservoir. Secondly, we apply independent component analysis (ICA) to these features in order to find the maximally statistically independent features. The complete setup is depicted in Figure 1d. This method has already been used in the context of robotics for robot localization [17]. There, they referred to the output of the ICA-layer as place cells, because they behave similarly to nodes found in the hippocampus of rodents [18]. Contrary to the place cells in rodents, which fire at a certain location, we have created *terrain* cells, which fire on a certain terrain.

As a baseline comparison for the terrain cells obtained by using reservoir computing and slow feature analysis (RC-SFA), we also expand the input signals quadratically instead of using a reservoir. This means that we use the product of each combination of 2 different input signals alongside the original signals as input for the slow feature analysis, shown in Figure 1e. We will refer to this system as SFA2.

1) Slow Feature Analysis: Slow Feature Analysis is an unsupervised machine learning algorithm which extracts slow varying signals from faster varying signals [19]. Suppose $\mathbf{x}(t)$ is a multi-dimensional input signal. SFA generates a slow varying output signal $y_i(t) = g_i(\mathbf{x}(t))$ by searching for the best functions g_i in a certain space such that

$$\Delta(y_i) = \langle \dot{y}_i^2 \rangle_t \tag{3}$$

is minimized under the following conditions:

$$\langle y_i \rangle_t = 0 \tag{4}$$

$$\langle y_i^2 \rangle_t = 1 \tag{5}$$

$$\forall j < i, \langle y_i y_j \rangle_t = 0. \tag{6}$$

This can be solved efficiently by the algorithm proposed in [19].

2) Independent Component Analysis: Independent Component Analysis [20] is an algorithm that separates multivariate signals in order to achieve maximally uncorrelated signals. It assumes there is a linear connection between the input signals $\mathbf{x}(t)$ and the underlying uncorrelated signals $\mathbf{s}(t)$. Therefore, the maximally uncorrelated signals can be regenerated using a matrix $\mathbf{W_{ICA}}$:

$$\mathbf{s}(t) = \mathbf{W_{ICA}}\mathbf{x}(t). \tag{7}$$

To find this matrix $\mathbf{W_{ICA}}$, we use the FastICA-algorithm [21].

E. Covariance Matrix Adaptation Evolution Strategy

To optimize the parameters of the different techniques described above, we use *Covariance Matrix Adaptation Evolution Strategy* (CMA-ES). CMA-ES is an evolutionary optimization algorithm which makes very few assumptions on the nature of the system. Only the ranking between the candidates is used in the learning process, no gradients or quantification of the input is needed. CMA-ES proves to be useful for non-separable, ill-conditioned or noisy objective functions [22]. Also, the different evaluations within one generation are independent, so the algorithm can easily be parallelized. In this paper, we use the implementation of CMA-ES written by Nikolaus Hansen [22].

CMA-ES has only one free parameter left, the population size λ. For this parameter, we use $\lambda = \lfloor 4 + 3\ln(M) \rfloor$ with M the number of parameters which need to be optimized. This choice has been made as a balance between quick convergence and the amount of parallelization possible, but is still arbitrary. The λ-parameter functions on a meta-level and therefore has little effect on the conclusions made in this paper.

III. Methodology

A. The Dataset

The dataset used in this work has been harvested at the University of Zurich, using their quadruped robot, Puppy II (Fig. 2) [23]. This robot has four identical legs, each controlled by a single servomotor at the hip joint. The knee joints are not directly controllable, but are passively moveable due to the spring attached between the upper and lower limb. Underneath the feet there is an adhesive skin attached with asymmetrical friction. This aids the robot in moving its legs forward and enforcing its grip while moving the legs backward. Puppy II

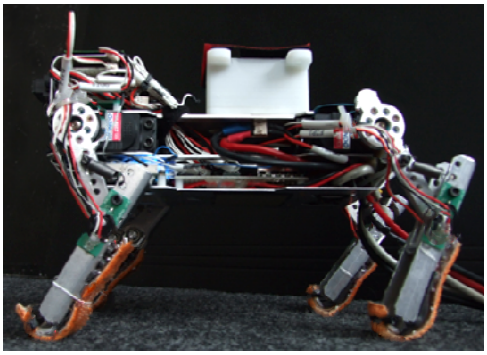


Fig. 2. Puppy II, from the Artificial Intelligence Laboratory, Department of Informatics, University of Zurich. This is a side-view with the front of the robot on the left.

TABLE I. Number of available trials in function of terrain type, gait frequency and gait. (F: blue foil, S: Styrofoam, L: linoleum, C: Cardboard, R: rubber)

	bound right					turn left					random				
	F	S	L	C	R	F	S	L	C	R	F	S	L	C	R
0,25Hz	3	1	-	-	-	2	1	-	-	-	-	-	-	-	-
0,50Hz	5	2	-	-	-	4	2	-	-	-	-	-	-	-	-
0,75Hz	4	2	-	-	-	4	2	-	-	-	-	-	-	-	-
1,00Hz	2	13	9	3	5	4	4	3	-	-	4	3	4	-	-
1,25Hz	6	1	-	-	-	-	3	-	-	-	-	-	-	-	-
1,52Hz	3	2	-	-	-	-	6	-	-	-	-	-	-	-	-
1,72Hz	8	3	-	-	-	-	-	-	-	-	-	-	-	-	-

is fitted with a number of sensors. There is a potentiometer attached to each joint, for a total of eight potentiometers. Each leg has a tactile sensor which measures the force exerted on the paw when touching the ground. The robot has an IMU and there is also a single external sensor used, namely an overhead camera. This camera is not used directly, but the position and velocity derived from the camera footage is.

The trials have a varying length, many between 3000 and 6500 samples with a sampling rate of 50Hz. The measurements have been made on five different terrains, at seven different frequencies, with three different gaits. The *bound right* and the *turn left* gait are generated by simple sine waves on the joints, with a different offset, amplitude and phase for each motor. In the *bound right* gait, the robot moves in a clockwise circle due to a slightly higher amplitude on the left legs. In the *turn left* gait, the robot moves counter-clockwise due to a much larger amplitude on the right hind leg. The *random* gait consists of random motor commands, sufficiently smooth not to exceed the motor bandwidth. For an overview of the trials available for each combination of parameters, we refer to Table I.

After testing a few combinations of sensor signals, we settle on a selection that delivers an optimal or near-optimal performance for all techniques that we examine. In this paper, we will mainly evaluate and compare the use of these eighteen signals: the angles of the four hips, the angles of the four knees, the tactile sensors on the four feet, the three-dimensional accelerometer and the three-dimensional gyroscope. For completeness' sake, we also evaluate the use of all information available in the dataset. We normalize each sensor signal to a zero mean and a unit variance. Then we select a total of 3000 samples for each terrain type and do this five times. Three of these are used for threefold cross-validation while

TABLE II. THE DIFFERENT GROUPS OF SENSORS AND THE SENSOR INFORMATION THEY CONTAIN

	useful	reduced	minimal	leg	agt	imu	joints	touch
Number of dimensions	55	21	18	12	10	9	8	4
Tactile sensors	×	×	×	×	×			×
Joint angle sensors	×	×	×	×			×	
Accelerometer and gyroscope	×	×	×		×	×		
Velocity	×	×						
Compass	×	×						
Distance to wall	×	×						
Magnetometer	×					×		
Other/derivative sensors	×							

optimizing the parameters, the other two are used for twofold cross-validation during the verification of the parameters. Any remaining relevant trials are added to the test set during verification. We split all trials into parts as a final step, with lengths varying from 281 to 562 samples. Most of these are about 375 samples long (7.5s).

These sensors were combined into a couple of groups which we evaluate on how much information they contain on the terrain type. For a detailed description of what sensor group contains which sensors, we refer to Table II.

B. System Classification Score

To evaluate the accuracy of the methods, the terrain is classified at each time-step. This means that we measure the effectiveness of the techniques as a realtime sensor. For the supervised techniques, we have 5 output nodes, the same as the number of terrains in the dataset. We train the output nodes to be 1 when the robot is walking on their respective terrain and to be -1 when the robot is not walking on its terrain. At each time-step, we classify by choosing the terrain corresponding to the output node with the highest value. The score of the system is the percentage of the time-steps correctly classified.

This approach however does not work for unsupervised techniques, as they do not have corresponding output nodes, as the system did not have any example data. To be able to give meaning to the output of the ICA-layer, we reconstruct the probability that the robot is on a certain terrain, given the terrain cells, with Bayes' theorem:

$$P(\mathbf{x_r}|\mathbf{y_{ICA}}) = \frac{P(\mathbf{y_{ICA}}|\mathbf{x_r})P(\mathbf{x_r})}{P(\mathbf{y_{ICA}})} \tag{8}$$

where $P(\mathbf{x_r})$ is the prior on the terrain vector and is related to our dataset and $P(\mathbf{y_{ICA}})$ is a normalization factor which does not need to be calculated explicitly. The classifier picks the terrain with the highest probability of $P(\mathbf{x_r}|\mathbf{y_{ICA}})$ as the correct terrain. The score of the unsupervised system, is the percentage of time-steps correctly classified this way.

$P(\mathbf{y_{ICA}}|\mathbf{x_r})$ can be obtained as follows, since the outputs of the ICA layer are statistally independent:

$$P(\mathbf{y_{ICA}}|\mathbf{x_r}) = \prod_{i=1}^{N_{ICA}} P(y^i_{ICA}|\mathbf{x_r}) \tag{9}$$

with y^i_{ICA} the output of terrain cell i. Finally, we can estimate $P(y^i_{ICA}|\mathbf{x_r})$ based on our train set, by creating a histogram of the terrain cells given a certain terrain.

TABLE III. TRAIN AND TEST RESULTS FOR EIGHT SIGNAL SELECTIONS (IN PERCENTAGE, MORE IS BETTER). THE ENTIRE TABLE HAS A SINGLE COLOR GRADIENT: MAXIMAL VALUE IS GREEN, MINIMAL IS RED, AVERAGE OF THE TWO IS YELLOW.

	TRAIN		TEST	
	LR	RC	LR	RC
useful	89.06	95.16	71.53	76.30
reduced	79.66	96.27	64.08	79.36
minimal	82.28	95.96	67.36	81.62
leg	75.70	95.42	63.63	81.53
agt	73.36	95.68	66.69	82.84
imu	53.35	91.47	43.98	72.25
joints	59.85	91.99	47.52	74.12
touch	64.45	93.84	57.89	84.69

IV. EXPERIMENTS

A. Supervised

1) Sensor Selection: The first experiment determines which sensor combinations provide the best information for supervised terrain classification. We test the eight different sensor combinations ranging from 4 to 55 signals out of 64 available in the dataset.

We process the sensor data of these combinations with PCA to compress the input signals, with the dimension reduction rate as a parameter. The compressed signals are then classified with LR and RC. All parameters of the different techniques are optimized using CMA-ES for each combination of sensors. In Table III the percentage of correctly classified time-steps is shown.

Note that the dataset has few trials for each combination of terrain, gait and gait frequency. Only some of these combinations are included in the training set, which increases the difficulty of correctly classifying the entire test set as the algorithms need to generalize over gaits and frequencies. Consequently there is a gap between the performance on the train set and the performance on the test set. This demonstrates the correlation of terrain classification performance and the actions of the robot, confirming the findings in [5].

As can be expected, LR performs better when it has a higher input dimensionality, as additional, less useful signals do not interfere with useful signals. On the other hand, RC combines its input signals and needs a good balance between the input dimensionality and the actual useful information these signals contain.

If we take a look at `imu` and `joints` in Table III, we notice a poor performance compared to the other results. These are the only signal selections without the tactile sensors. `leg` and `agt` are exactly the same as `joints` and `imu` with the exclusion of the tactile sensors (and the inclusion of the magnetometer in `imu`). This clearly indicates the importance of the tactile sensors for supervised terrain classification. RC even achieves the best result solely using the tactile sensors.

2) Selection of the Processing System: In order to compare the different supervised systems discussed before (LR, ELM and RC), we compare their performance for two sensor combinations: `touch` and `minimal`. We picked `minimal` because it seemed to strike a good balance between dimensionality and performance across both techniques. Since RC achieved its best score on the sensor combination `touch`, it was added

TABLE IV. TRAIN AND TEST RESULTS FOR COMPARISON OF RC AND ELM WITH VARYING NUMBER OF NODES IN THE HIDDEN LAYER (IN PERCENTAGE, MORE IS BETTER). SIGNAL SELECTIONS TOUCH AND MINIMAL ARE TESTED, USING THE ENTIRE DATASET.

		TOUCH			MINIMAL		
		50N	100N	200N	50N	100N	200N
TRAIN	LR	64.45			82.28		
	ELM	83.04	84.19	85.75	87.32	91.84	94.38
	RC	88.17	90.40	92.99	88.27	91.73	95.01
TEST	LR	57.89			67.36		
	ELM	72.83	72.47	75.52	74.58	79.82	80.80
	RC	80.30	83.04	85.19	74.09	77.28	81.43

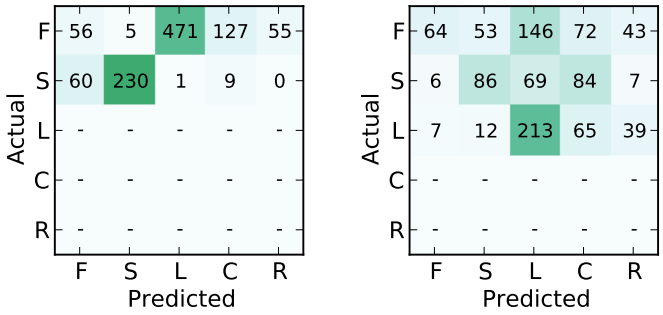


Fig. 3. Reservoir computing confusion matrix for testing gait frequency (left) and gait (right) generalization.

TABLE V. TRAIN AND TEST RESULTS FOR EIGHT SIGNAL SELECTIONS (IN PERCENTAGE, MORE IS BETTER). THE ENTIRE TABLE HAS A SINGLE COLOR GRADIENT: MAXIMAL VALUE IS GREEN, MINIMAL IS RED, AVERAGE OF THE TWO IS YELLOW.

	TRAIN		TEST	
	SFA2	RC-SFA	SFA2	RC-SFA
useful	58.11	79.75	38.94	51.69
reduced	60.98	81.03	36.27	58.28
minimal	61.72	79.26	37.90	57.29
leg	53.71	72.25	33.82	58.33
agt	51.49	75.21	26.99	52.75
imu	44.45	70.83	29.16	55.29
joints	38.22	71.15	27.50	54.58
touch	48.09	75.53	35.97	50.29

TABLE VI. TRAIN AND TEST RESULTS FOR COMPARISON OF RC-SFA AND ELM-SFA WITH VARYING NUMBER OF NODES IN THE HIDDEN LAYER (IN PERCENTAGE, MORE IS BETTER). THE BOUND RIGHT GAIT AT ALL FREQUENCIES WERE USED.

		MINIMAL		
		50N	100N	200N
TRAIN	SFA2	61.72		
	ELM-SFA	81.92	82.93	86.24
	RC-SFA	83.97	82.71	86.60
TEST	SFA2	37.90		
	ELM-SFA	63.16	70.31	72.64
	RC-SFA	65.64	68.62	71.92

as well. From the results in Table IV it is clear that the non-linearities introduced by expanding the input sensors are necessary for a good result, as linear regression has over 10% more misclassifications then any other technique with 100 hidden nodes. The results also show that given enough input signals (`minimal`: 18 dimensions), RC and ELM achieve virtually the same performance. Using only four tactile sensors on the other hand, requires a system with recurrent nodes, as RC outperforms ELM roughly 10%, and at the same time achieves the best result overall, even while having less input information.

3) Generalizability: Lastly we investigate the generalizability of the methods to other gaits or gait frequencies. We train the system on the single gait 'bound right' at the single frequency 1Hz. Subsequently we test it on other frequencies of the same gait on the one hand and other gaits at the same frequency on the other hand. Figure 3 depicts both these approaches for RC. The extent of this experiment is limited by the available trials in the dataset, as can be seen in Table I. The dataset contains only three terrains for the gait generalization and two terrains for the gait frequency.

Looking at the gait frequency we notice a high performance for styrofoam (S) but a complete misclassification for foil (F), with a similar result for LR. Looking back at table I we notice that only 2 trials on foil are available for training, while there are 13 trials on styrofoam (only 5 used during training). This explains the poor performance for foil, meaning the performance for styrofoam might indicate a possible generalization to different frequencies.

Gait generalization on the other hand seems less feasible. The result for foil can again be attributed to its limited train set, but styrofoam has an equally poor performance. Only linoleum (L) achieves a fairly decent classification. If we would have a larger dataset and use multiple gaits during training, gait generalization might be possible. Only a single gait and gait frequency on the other hand does not carry enough information to correctly classify new gaits.

B. Unsupervised

1) Sensor selection: Table V depicts the results for the experiment with the unsupervised techniques, similarly as the experiment with supervised techniques. The entire dataset is used and PCA is applied for compacting the input signals.

Even though SFA2 and RC-SFA are mostly the same system, the different expansion of the input signals clearly has a large impact on the performance. This indicates that the way reservoirs expand the dynamics of the system is beneficial for classifying terrains. SFA2 performs relatively close to a random terrain sensor (25%), while RC-SFA performs about 10 to 20% better. The signal selections `imu` and `joints` seem to perform rather well, unlike with the supervised techniques. Solely using the tactile sensors still achieves a decent performance, but SFA seems to benefit from more input sensors.

2) Selection of the Processing System: Similarly as in section IV-A2, we investigate the importance of the recurrent connections in the reservoir. To do this, we compare the results of reservoir computing with leaky ELM's. In Table VI, RC-SFA is compared with ELM-SFA using the sensor group `minimal` and the bound right gait at all frequencies.

So with unsupervised learning techniques, the same conclusion of the supervised techniques holds as well. Here, the recurrent connections do not seem to make a significant difference either on the `minimal` sensor group. Note on the other hand the decent results for unsupervised classification

when only using a single gait, as opposed to the results of the complete dataset from Table V.

V. Conclusion

In this paper, we showed that a limited but appropriate selection of input sensors is sufficient to perform terrain classification on our legged robot. We have demonstrated that good results are achievable with nearly all methods tested, using the combination of a 6 DoF IMU, joint angle sensors and tactile sensors on the feet. The fact that linear regression performed badly on the data, indicates the importance of adding non-linearities. For these non-linearities, we found that reservoirs perform better than quadratically expanding, as is demonstrated by the unsupervised classification method. For supervised learning, reservoir computing led to drastic better performance when using few sensors, which indicates that the richness of the non-linear temporal expansion is beneficial for classification with less information. The memory capacity seems to be an important element, as the filtering used in the ELMs proved to be insufficient.

How generally applicable this conclusion is, cannot be reliably determined from this study, since only one robot was used to obtain the data. We note that this observation was reached on all tested terrains and gaits. Given sufficient data in the train set, it was possible to classify terrains even at unseen frequencies. However, when trained on a single gait, the methods studied here were not very effective at generalizing to unseen gaits.

We want to conclude that the challenge is to find an appropriate set of input sensors to classify terrains in quadruped robots. On the Puppy II robot, we found that a limited set of sensor inputs were enough to perform good quality terrain classification, when using methods which take into account that there are underlying dynamics and non-linearities in the system. We found that reservoir computing is a good way to take these dynamics and non-linearities into account, since the fading memory introduced by the recurrent connections between the nodes improved performance when few sensors are available, compared to other similar techniques.

VI. Acknowledgments

The authors would like to thank Matej Hoffmann of the University of Zurich again for lending us the dataset obtained from the Puppy II robot. The research leading to these results has received funding from the European Community's Seventh Framework Programme FP7/2007-2013 – Challenge 2 – Cognitive Systems, Interaction, Robotics – under grant agreement No 248311 - AMARSi.

References

[1] J. Buchli, M. Kalakrishnan, M. Mistry, P. Pastor, and S. Schaal, "Compliant quadruped locomotion over rough terrain," in *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, oct. 2009, pp. 814 –820.

[2] C. Plagemann, S. Mischke, S. Prentice, K. Kersting, N. Roy, and W. Burgard, "Learning predictive terrain models for legged robot locomotion," in *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, sept. 2008, pp. 3545 –3552.

[3] P. Giguere, G. Dudek, C. Prahacs, and S. Saunderson, "Environment identification for a running robot using inertial and actuator cues," *Proc. of Robotics Science and System*, August 2006.

[4] M. Hoepflinger, C. Remy, M. Hutter, L. Spinello, and R. Siegwart, "Haptic terrain classification for legged robots," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, may 2010, pp. 2828 –2833.

[5] M. Hoffmann, N. M. Schmidt, R. Pfeifer, A. K. Engel, and A. Maye, "Using sensorimotor contingencies for terrain discrimination and adaptive walking behavior in the quadruped robot puppy," in *From Animals to Animats 12*. Springer, 2012, pp. 54–64.

[6] P. Giguere and G. Dudek, "Clustering sensor data for autonomous terrain identification using time-dependency," *Auton. Robots*, vol. 26, no. 2-3, pp. 171–186, Apr. 2009. [Online]. Available: http://dx.doi.org/10.1007/s10514-009-9114-2

[7] P. Giguere, "Unsupervised learning for mobile robot terrain classification," 2009.

[8] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: Theory and applications," *Neurocomputing*, vol. 70, no. 1-3, pp. 489 – 501, 2006. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0925231206000385

[9] D. Verstraeten, B. Schrauwen, M. d'Haene, and D. Stroobandt, "An experimental unification of reservoir computing methods," *Neural Networks*, vol. 20, no. 3, pp. 391–403, 2007.

[10] W. Maass, T. Natschläger, and H. Markram, "Real-time computing without stable states: A new framework for neural computation based on perturbations," *Neural computation*, vol. 14, no. 11, pp. 2531–2560, 2002.

[11] H. Jaeger and H. Haas, "Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication," *Science*, vol. 304, no. 5667, pp. 78–80, 2004.

[12] J. J. Steil, "Backpropagation-decorrelation: Online recurrent learning with o (n) complexity," in *Neural Networks, 2004. Proceedings. 2004 IEEE International Joint Conference on*, vol. 2. IEEE, 2004, pp. 843–848.

[13] H. Jaeger, M. Lukoševičius, D. Popovici, and U. Siewert, "Optimization and applications of echo state networks with leaky-integrator neurons," *Neural Networks*, vol. 20, no. 3, pp. 335–352, 2007.

[14] K. Caluwaerts, F. wyffels, S. Dieleman, B. Schrauwen *et al.*, "The spectral radius remains a valid indicator of the echo state property for large reservoirs," in *International Joint Conference on Neural Networks (IJCNN-2013)*, 2013.

[15] M. Lukoševičius and H. Jaeger, "Survey: Reservoir computing approaches to recurrent neural network training," *Computer Science Review*, vol. 3, no. 3, pp. 127–149, 2009.

[16] M. Hermans and B. Schrauwen, "Memory in linear recurrent neural networks in continuous time," *Neural Networks*, vol. 23, no. 3, pp. 341–355, 2010.

[17] E. Antonelo and B. Schrauwen, "Learning slow features with reservoir computing for biologically-inspired robot localization," *NEURAL NETWORKS*, vol. 25, pp. 178–190, 2012. [Online]. Available: http://dx.doi.org/10.1016/j.neunet.2011.08.004

[18] E. I. Moser, E. Kropff, and M.-B. Moser, "Place cells, grid cells, and the brain's spatial representation system," *Annu. Rev. Neurosci.*, vol. 31, pp. 69–89, 2008.

[19] L. Wiskott and T. J. Sejnowski, "Slow feature analysis: Unsupervised learning of invariances," *Neural computation*, vol. 14, no. 4, pp. 715–770, 2002.

[20] P. Comon, "Independent component analysis, a new concept?" *Signal Processing*, vol. 36, no. 3, pp. 287 – 314, 1994. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0165168494900299

[21] A. Hyvärinen and E. Oja, "Independent component analysis: algorithms and applications," *Neural Networks*, vol. 13, no. 4-5, pp. 411 – 430, 2000. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0893608000000265

[22] N. Hansen, "The CMA evolution strategy: a comparing review," in *Towards a new evolutionary computation. Advances on estimation of distribution algorithms*, J. Lozano, P. Larranaga, I. Inza, and E. Bengoetxea, Eds. Springer, 2006, pp. 75–102.

[23] N. M. Schmidt, M. Hoffmann, K. Nakajima, and R. Pfeifer, "Bootstrapping perception using information theory: Case studies in a quadruped robot running on different grounds," *Advances in Complex Systems*, p. 1250078. [Online]. Available: http://www.worldscientific.com/doi/abs/10.1142/S0219525912500786