Controlling Robot Morphology from Incomplete Measurements

Martin Pecka, Karel Zimmermann, *Member, IEEE,* Michal Reinstein, *Member, IEEE,*
Tomas Svoboda, *Member, IEEE,*

***Abstract*—Mobile robots with complex morphology are essential for traversing rough terrains in Urban Search & Rescue missions (USAR). Since teleoperation of the complex morphology causes high cognitive load of the operator, the morphology is controlled autonomously. The autonomous control measures the robot state and surrounding terrain which is usually only partially observable, and thus the data are often incomplete. We marginalize the control over the missing measurements and evaluate an explicit safety condition. If the safety condition is violated, tactile terrain exploration by the body-mounted robotic arm gathers the missing data.**

I. Introduction

Since exploration of unknown disaster areas during *Urban Search & Rescue* missions (USAR) is often dangerous, teleoperated robotic platforms are usually used as a suitable replacement for human rescuers. Motivation to our research comes from field experiments with a tracked mobile robot with four articulated subtracks (flippers, see Fig. 1). The robot morphology allows to traverse complex terrain. A high number of articulated parts brings, however, more degrees of freedom to be controlled. Manual control of all available degrees of freedom leads to undesired cognitive load of the operator, whose attention should be rather focused on reaching the higher-level USAR goals. To reduce the cognitive load of the operator, the autonomy of the platform has to be increased; however, it still has to fall within the bounds accepted by the operators—a compromise known as *accepted autonomy* has to be reached [1].

In [2], a Reinforcement-Learning–based *autonomous control* (AC) of robot morphology (configuration of flippers) is proposed. Its goal is to allow smooth and safe traversal of complex and previously unknown terrain while letting the operator specify the desired speed vector. The traversing task is called *Adaptive Traversal* (AT). Natural and disaster environments (such as forests or collapsed buildings) yield many challenges that include incomplete or incorrect data due to reflective surfaces such as water, occluded view, presence of smoke, and deformable terrain such as deep snow or piles of rubble. Since simple interpolation of the missing terrain profile has proved to be insufficient, we presented an improved AC algorithm that better handles incomplete sensory data (using marginalization) [3].

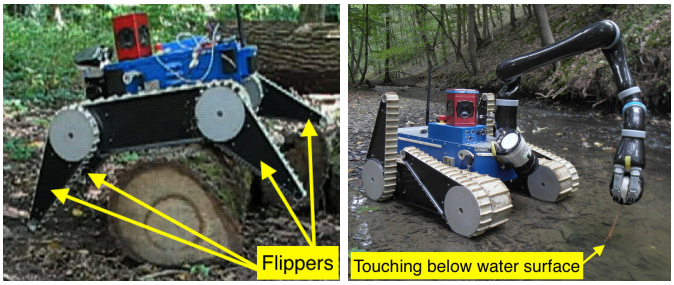


Fig. 1. **Left:** Controlling robot morphology (flippers) allows for traversing obstacles. **Right:** Robotic arm inspects terrain below water surface compensating thus incomplete lidar measurement.

In this work, we extend and improve the AC pipeline introduced in our previously published work [2], [3] (see Fig. 2 for an overview). **The novel contributions include: (i)** introducing a safety measure which allows to invoke tactile exploration of non-visible terrain if needed; **(ii)** several strategies for the tactile exploration with a body-mounted robotic arm; **(iii)** two Q-function representations which allow easier marginalization and achieve comparable (or better) results; **(iv)** and finally, an extensive experimental evaluation of the Autonomous Control. The real-world experiments cover more than 115 minutes of robot time during which the robot traveled 775 meters over rough terrain obstacles.

II. Related work

Many approaches focus on optimal robot motion control in environments with a known map, leading rather to the research field of trajectory planning [4], [5], [6]. Contrary to planning, AC is useful in previously unknown environments and hence can provide crucial support to the actual procedure of map creation. We rather perceive AC as an independent complement to trajectory planning and not as its substitution.

Many authors [7], [4], [8] estimate terrain traversability only from exteroceptive measurements (e.g. laser scans) and plan the (flipper) motion in advance. In our experience, when the robot is teleoperated, it is often impossible to plan the

Manuscript received Nov 30, 2015; revised April 7, 2016 and May 10, 2016; accepted May 11, 2016. The research leading to these results has received funding from the European Union under grant agreement FP7-ICT-609763 TRADR; from the Czech Science Foundation under Project GA14-13876S, and by the Grant Agency of the CTU Prague under Project SGS15/081/OHK3/1T/13.

All authors are with the Dept. of Cybernetics, Faculty of Electrical Engineering, Czech Technical University in Prague, Czech republic. K. Zimmermann is the corresponding author (phone: +420-22435-5733, email: zimmerk@fel.cvut.cz).

M. Pecka and T. Svoboda are partly with the Czech Institute of Cybernetics Robotics and Informatics, Czech Technical University in Prague, Czech republic.

flipper trajectory in advance from the exteroceptive measurements only. The reasons are three-fold: (i) it is not known in advance, which way is the operator going to lead the robot, (ii) the environment is usually only partially observable, (iii) analytic modeling of Robot–Terrain Interaction (RTI) in a real environment is very challenging because the robot can slip or the terrain may deform. Ho et al. [9] directly predict the terrain deformation only from exteroceptive measurements to estimate traversability. They do not provide any alternative solution when exteroceptive measurements are missing. Abbeel et al. [10] use a different approach—they use only proprioceptive measurements for helicopter control, which often works well for aerial vehicles (unless obstacle avoidance is required). We propose that reactive control based on all available measurements is needed for ground vehicles (where obstacle avoidance or robot–ground interaction is essential).

An ample amount of work [11], [12], [13] has been devoted to the recognition of traversal-related manually defined classes (e.g. surface type, expected power consumption or slippage coefficient). However, such classes are often weakly connected to the way the robot can actually interact with the terrain. Few papers describe the estimation of RTI directly. For example, Kim et al. [14] estimate whether the terrain is traversable or not, and Ojeda et al. [15] estimate power consumption on different terrain types. In literature, the RTI properties are usually specified explicitly [15], [16], [14] or implicitly (e.g. state estimation correction coefficient [17], [18]).

Since RTI properties do not directly determine the optimal reactive control, their estimation can be completely avoided. Zhong et al. [19] present a trajectory tracking approach, in which they control a hexapodal robot and utilize force sensors in the legs to detect unexpected obstacles and walk over them. The algorithm tries to minimize the trajectory error caused by obstacles, so that the underlying controller does not need to take them into account. We proposed a different algorithm [2] that explicitly takes the terrain into account (which should yield better results than trying to hide the terrain from the controller). The algorithm is based on Reinforcement Learning, which has been successfully used e.g. in learning propeller control for acrobatic tricks with an RC helicopter [10], [20]. Since it is possible to model the helicopter-air interactions quite plausibly, an RTI model can be used to speed up the learning. In case of ground vehicles, analytical modeling of RTI is very difficult. Therefore, we rather focus on a model-free RL technique called Q-learning (used e.g. to find optimal control in [21]). In Q-learning, state is mapped to optimal actions by taking "argmax" of the so-called Q function (the sum of discounted rewards). In our case, the state space has high dimension (some dimensions with continuous domain), and therefore the Q function cannot be trained for all state–action pairs. Thus, it is modeled either by Regression Forests (RF) or by Gaussian Processes (GP). Regression Forests are known to provide good performance when a huge training set is available [22], with learning complexity linear in the number of training samples. Gaussian Processes present an efficient solution in the context of Reinforcement Learning for control [23].

To deal with incomplete data, the Q function values have to be marginalized over missing features. Such marginalization is often tackled by sampling [24], [25] or EM algorithm [26]. Especially for GPs with Squared Exponential kernel, the Moment Matching marginalization method was proposed by Deisenroth et al. [23]. Marginalization by Gibbs sampling was evaluated for GPs and piecewise constant functions in [3].

We are not aware of any real mobile platform which would use a robot arm as an active sensor for inspecting unknown terrain. Most of the efforts in active inference are directed towards active classification [27], [28], [29] or active 3D reconstruction. Doumanoglou et al. [27] use two robotic arms for folding an unknown piece of cloth whose type is recognized from RGBD data (Kinect). One view is usually insufficient, therefore the cloth needs to be turned around to generate an alternative view. The turning action is implicitly learned with Decision Forests. Bjorkman et al. [28] also recognize objects from RGB-D data. In contrast to [27], Bjorkman et al. use the robotic arm as an active sensor, to touch the self-occluded part of the object in order to reconstruct the invisible 3D shape. While all these classification approaches actively evaluate features in order to discriminate the true (*single*) object class from other possible classes as fast as possible, the Q-learning–based inference presented here evaluates the features in order to find some of the (*multiple*) suitable flipper configurations that allow for a safe and efficient traversal.

III. Overview

Q-learning: The proposed AT solution is adapted from the RL technique called Q-learning (described first to emphasize the differences). The first step in the learning process is driving manually the robot over obstacles to collect a dataset. The state $\mathbf{x}$ (e.g. body pitch angle or terrain shape; see Section IV) is sampled at regular time intervals $t = 0, 1, \ldots, T$. At each time instant t, the operator chooses an action c^t (e.g. the desired flipper positions) that allows to go over the obstacle. After the dataset is collected, each state-action pair $(c^t, \mathbf{x}^t)$ is assigned a reward r^t reflecting suitability of choosing the action in the given state.

Then the iterative Q-learning process starts, which estimates the q^t-values that represent the *sum of discounted rewards* the robot can gather by starting in state $\mathbf{x^t}$, executing action c^t, and always taking the action leading to maximum q from the following state onwards [30]. The q^t and Q values are computed using the recurrent Q-learning formulas [31]:

$$q_i^t := q_{i-1}^t + \alpha\Big[r^t + \gamma \max_{c'} Q_{i-1}(c', \mathbf{x^{t+1}}) - Q_{i-1}(c^t, \mathbf{x^t})\Big] \quad (1)$$

$$Q_i(c, \mathbf{x}) := \mathrm{mean}(q_i^t \mid c^t = c \wedge \mathbf{x^t} = \mathbf{x}) := \mathrm{mean}(q_i(c, \mathbf{x})) \quad (2)$$

where $q_1^t := r^t$, $\alpha \in [0, 1]$ is the learning rate and $\gamma \in [0, 1]$ is the discount factor. From the computation above, it follows that $Q_i(c, \mathbf{x})$ is an unbiased estimator of $E[q_i(c, \mathbf{x})]$.

When the Q-learning is done, we denote $Q = Q_i$ and $q^t = q_i^t$, and the optimal action can be computed as:

$$c^*(\mathbf{x}) \quad = \quad \operatorname*{argmax}_c Q(c, \mathbf{x}) \quad (3)$$

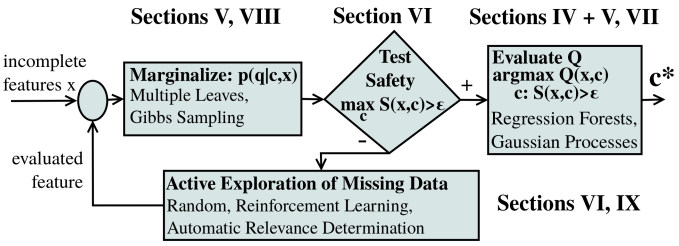


Fig. 2. **Principle overview:** individual blocks in this scheme correspond to Sections IV-VI.

QPDF: In this paper, we generalize the standard Q-learning to an algorithm that learns a distribution called QPDF instead of the Q function. For the QPDF (denoted as $p(q|c,\mathbf{x})$) it holds that

$$Q(c,\mathbf{x}) = E[q(c,\mathbf{x})] = \int q \cdot p(q|c,\mathbf{x})\, dq$$

There are two reasons for modeling the full QPDF: (i) measuring the safety of flipper configurations and (ii) marginalization when only incomplete measurements of $\mathbf{x}$ are available. In Section V, two QPDF models are presented: (i) Regression Forests and (ii) Uncertain Gaussian Processes.

Given the QPDF and full feature vector $\mathbf{x}$, the optimal action $c^*(\mathbf{x})$ is:

$$\begin{aligned} c^*(\mathbf{x}) &= \operatorname*{argmax}_c Q(c,\mathbf{x}) = \operatorname*{argmax}_c E[q(c,\mathbf{x})] = \\ &= \operatorname*{argmax}_c \int q \cdot p(q|c,\mathbf{x})\, dq \end{aligned} \tag{4}$$

Missing Data: While proprioceptive data are usually fully available, the exteroceptive data are often incomplete. This occurs in case of reflective surfaces such as water or in presence of smoke. We denote the missing parts of measurements as $\overline{\mathbf{x}}$, and the available measurements as $\widetilde{\mathbf{x}}$, i.e. $\mathbf{x} = [\overline{\mathbf{x}}, \widetilde{\mathbf{x}}]$. In the case that $\overline{\mathbf{x}}$ is not empty, $p(q|c,\mathbf{x})$ is marginalized over the missing data $\overline{\mathbf{x}}$ to estimate $p(q|c,\widetilde{\mathbf{x}})$. The marginalization processes for different QPDF models are described in Section V. Given the marginalized distribution $p(q|c,\widetilde{\mathbf{x}})$ and measurement $\widetilde{\mathbf{x}}$, the optimal action c^* is estimated by a small modification of Equation 4:

$$c^*(\widetilde{\mathbf{x}}) = \operatorname*{argmax}_c \int q \cdot p(q|c,\widetilde{\mathbf{x}})\, dq. \tag{5}$$

Any state-action pair yielding a negative q-value is interpreted as unsafe considering our definition of the reward function[1]. Therefore, the probability that the q-value is positive (safe) can be computed, and only sufficiently safe state-action pairs are to be considered further. The general trend is that the more features are missing, the higher is the scatter of q-values. Hence, we define the *safety measure*

$$S(c,\widetilde{\mathbf{x}}) = \int_0^\infty p(q \mid c,\widetilde{\mathbf{x}})\, dq, \tag{6}$$

[1]This assumes the user-denoted penalty for dangerous states to be sufficiently high and discount factor sufficiently different from one; see Section IV for definition of the reward function.

that corresponds to the probability of achieving a safe state ($q \geq 0$) with action c. Search for the optimal action c^* (Equation 5) is restricted only to safe actions:

$$S(c,\widetilde{\mathbf{x}}) > \epsilon. \tag{7}$$

Active Exploration: If none of the available actions satisfies the safety condition (Equation 7), the robotic arm is used to measure some of the missing terrain features; see Fig. 2 for the pipeline overview. In Section VI, we propose several strategies that guide the active exploration of missing features in order to find a safe action as fast as possible. If all terrain features have already been measured and there is still no action satisfying the safety condition, manual flipper control is requested from the operator.

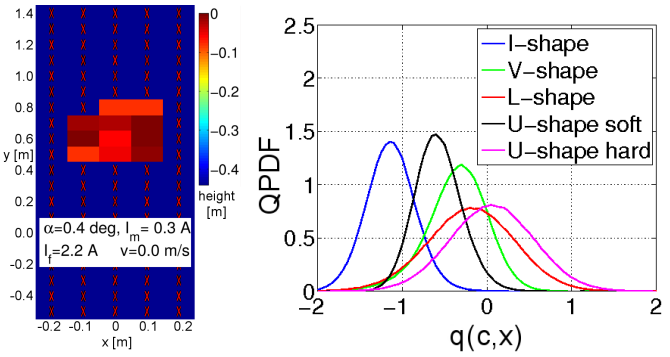


Fig. 3. **Example of insufficient data. An active exploration is necessary.** The left figure shows the input data; the missing heights in the DEM are outlined by red crosses in a blue rectangle, pitch is denoted by α, mean absolute current over both main tracks is denoted I_m, mean absolute current in the engines lifting the front flippers is denoted by I_f. More details on the features are given in Section IV. The right figure contains QPDFs for the five flipper configurations ("*-shape"). The horizontal axis corresponds to the sum of discounted rewards (higher are better), vertical axis contains QPDF. Figure adapted from [3].

Fig. 3 shows an example situation when active exploration is needed. Looking at the right figure, the highest value of the safety measure $S(c,\widetilde{\mathbf{x}})$ is approximately 0.5. If the safety limit ϵ is 0.8, tactile exploration is activated, because no action satisfies the safety limit in the current state.

IV. Adaptive Traversability Task

The AT task is solved for a tracked robot equipped with two main tracks, four independent articulated subtracks (*flippers*) with customizable compliance[2], rotating 2D laser scanner (SICK LMS-151), Kinova Jaco robotic arm, and an IMU (Xsens Mti-G); see Fig. 1. The task is detailed in the following paragraphs, and a short summary is given in Table I.

States: The state of the robot and the local neighboring terrain is modeled as n-dimensional feature vector $\mathbf{x} \in \mathbb{R}^n$ consisting of: **i) exteroceptive features:** Individual scans from one sweep of the rotating laser scanner (3 seconds) are put into an Octomap [32] with cube size of $5\,\mathrm{cm}$. This

[2]Upper limit of current in the flipper motor used to hold the flipper in position.

TABLE I. DESCRIPTION OF THE STATES, ACTIONS AND REWARDS

State	$\mathbf{x} \in \mathbb{R}^n$	DEM, speed, roll, pitch, flipper angles, compliance, currents in flippers, actual flipper configuration
Actions	$c \in \mathbf{C} = \{1 \dots 5\}$	5 pre-set flipper configurations [2]
Reward	$r(c, \mathbf{x}) : \mathbf{C} \times \mathbb{R}^n \to \mathbb{R}$	$\alpha \times$ user reward $s_{c,\mathbf{x}} + \beta \times$ pitch penalty $+$ $\gamma \times$ roughness penalty

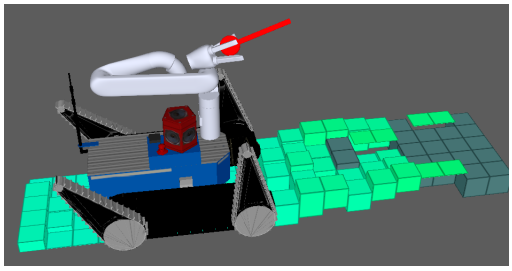


Fig. 4. **Digital Elevation Map (DEM):** Example of the DEM representation with dark green used for missing values and light green representing height estimate included in the feature space.

Octomap is then cropped to close neighborhood of the robot ($50\,\mathrm{cm} \times 200\,\mathrm{cm}$ size). Further, the cubes are aggregated into $10\,\mathrm{cm} \times 10\,\mathrm{cm}$ columns and mean height in each of these columns is computed. This yields a local representation of the terrain with x/y sub-sampled to $10\,\mathrm{cm} \times 10\,\mathrm{cm}$ tiles (bins) and vertical resolution of $5\,\mathrm{cm}$. This is what we call a Digital Elevation Map (DEM); see Fig. 4. Heights in the bins are used as exteroceptive features. **ii) proprioceptive features:** Robot speed (actual and desired), roll, pitch, flipper angles, compliance thresholds, actual current in flippers and actual flipper configuration.

Actions: The robot has many degrees of freedom, but only some of them are relevant to the traversal. The speed and heading of the robot are controlled by the operator. AC is used to control the pose of the four flippers and their compliance, yielding together 8 DOF. Further simplification of the action space is allowed by observations made during experiments—only 4 discrete (laterally symmetric) flipper configurations are enough for most of the terrain types, and 2 different levels of compliance are also sufficient. The arm has to be in a stable default "transport" position when the robot moves, so its DOFs are ignored. Finally, 5 *flipper configurations* denoted by $c \in \mathbf{C} = \{1 \dots 5\}$ are defined. These configurations named *I-shape*, *V-shape*, *L-shape*, *U-shape soft* and *U-shape hard* are described in detail in [2].

Rewards: The reward function $r(c, \mathbf{x}) : (\mathbf{C} \times \mathbb{R}^n) \to \mathbb{R}$ assigns a real-valued reward for using c in state $\mathbf{x}$. It is expressed as a weighted sum of (i) user-denoted bipolar penalty $s_{c,\mathbf{x}}$ specifying whether executing c in state $\mathbf{x}$ is *permitted* (safe), (ii) high pitch angle penalty (preventing robot's flip-over), and (iii) the motion roughness penalty measured by accelerometers.

V. QPDF REPRESENTATION AND LEARNING

In our previous work [2] piecewise constant functions were introduced as a method to represent Q functions. For the case of missing features, Gaussian Processes with Rational Quadratic kernel were used to represent Q functions in our following work [3]. In the latter work, Regression Forests are trained on fetaures completed by Gibbs sampling marginalization of the missing features. In this section, we propose two new approaches to QPDF representation that tackle the case of incomplete data.

A. Regression Forests

The first method is based on Regression Forests with incomplete data on their input, representing the QPDF in their leaves (instead of first estimating the missing features and then computing Q from a full feature vector, as the previous method does). Thus we avoid the unnecessary step of reconstructing the missing features, and can directly use the incomplete input to estimate QPDF.

Learning: The QPDF for each configuration is modeled independently by a Regression Forest. The trees are constructed sequentially, always building one until all leaves are *terminal* (see further), and then starting to build another one. To train each particular tree, a training set consisting of m training samples $[\mathbf{x}_1, \dots, \mathbf{x}_m]$ is given, with corresponding q-values $[q_1, \dots, q_m]$. Each training sample $\mathbf{x}_k$ is an n-dimensional vector of features $\mathbf{x}_k = [x_k^1 \dots x_k^n]^\top$. The tree is built by a greedy recurrent algorithm, that selects the splitting feature $j^* \in J = \{1 \dots n\}$ and split threshold s^*. The splitting feature and threshold are selected to minimize the weighted variance of q-values in the left and right sub-tree in each node as follows [3]:

$$(s^*, j^*) = \underset{(s,j)}{\operatorname{argmin}} \; |R_1(s,j)| \cdot \underset{k \in R_1(s,j)}{\operatorname{var}}(q_k) + |R_2(s,j)| \cdot \underset{k \in R_2(s,j)}{\operatorname{var}}(q_k)$$

where $R_1(s,j) = \{k \mid x_k^j \leq s\}$ is the set of indices descending to the left sub-tree, and $R_2(s,j) = \{k \mid x_k^j > s\}$ is the set of indices descending into the right sub-tree. The tree is constructed recursively. If a stopping criterion is satisfied (either minimum number of samples per node, or tree height), a *terminal leaf* is created, which contains discretized QPDF histogram (estimated from q-values of all training samples that descended to that leaf). Specifically, if the value of the splitting feature is unknown in sample $\mathbf{x}_i$ (e.g. occluded), then it descends into both sub-trees.

Marginalization: To obtain the marginalized distribution $p(q|c, \widetilde{\mathbf{x}})$, sample $\widetilde{\mathbf{x}}$ is put to the input of the forest. If a tested feature is missing in $\widetilde{\mathbf{x}}$, the algorithm descends into both sub-trees similarly to the learning procedure. The final QPDF is then a weighted average of histograms in all reached leaves in all trees (properly normalized to be a distribution). Weights are given by prior probabilities of leaves estimated from training data. We call this *Multiple Leaves marginalization*.

B. Gaussian Processes

Gaussian processes [23] are the extension of multivariate Gaussians to infinite-size collections of real valued variables and can be understood as joint Gaussian distributions over random functions. The essential part of GP learning is given by the choice of a kernel function (parametrized by a set of

hyper-parameters $\boldsymbol{\theta}$). We use the common *Squared Exponential* kernel function (SE), for which the *Uncertain* Gaussian Processes are derived in [33]. This allows processing features with unknown or uncertain values. In case Uncertain GPs are not necessary, i.e. Gibbs sampling is used to handle uncertain values (as in [3]), the *Rational Quadratic* (RQ) kernel that performs slightly better than SE can be used. Both SE and RQ kernels enable *Automatic Relevance Determination* [34], which can be interpreted as embedded feature selection performed automatically when optimizing over the kernel hyper-parameters $\boldsymbol{\theta}$. The ARD values are utilized in Section VI-B.

Learning: A standard regression model is used, assuming the data $\mathcal{D} = \{\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_m]^T, \mathbf{q} = [q_1, \ldots, q_m]^T\}$ were generated according to $q_i = h(\mathbf{x}_i) + \epsilon_i$, where $h : \mathbb{R}^n \to \mathbb{R}$, and $\epsilon_i \sim \mathcal{N}(0, \sigma_\epsilon^2)$ is independent Gaussian noise. Thus, there is a direct connection between $h(\mathbf{x})$ and the QPDF. For each configuration c, the Uncertain GP learning procedure is used to train a GP model that predicts the given q-values. The learning procedure[3] is described in detail in [33].

Marginalization: GPs consider h as a random function in order to infer posterior distribution $p(h|\mathcal{D})$ over h from the GP prior $p(h)$, the data $\mathcal{D}$, and assumption on smoothness of h [33]. The posterior is estimated to make predictions at inputs (the testing data) $\mathbf{x} \in \mathbb{R}^n$ about the function values $h(\mathbf{x})$, which can be used as the QPDF. Since the posterior is no longer a Gaussian, it is approximated by a Gaussian distribution, using e.g. the Moment Matching method described in [23].

VI. Tactile Terrain Exploration

Given the QPDF, safety condition (Equation 7) is evaluated for all possible configurations. If more than one safe configuration exists, AC chooses the one that yields the highest q-value mean. If none of the configurations is safe, the robot is stopped and Tactile Terrain Exploration (TTE) is triggered (example situation is depicted in Fig. 3). This exploration utilizes the robotic arm to measure the height in DEM bins in which measurements are missing[4]. The arm actively explores the missing heights until the safety condition (Equation 7) is satisfied for at least one configuration, or there are no more missing heights (we refer to both these cases as *final states*). If the state in the latter case is still unsafe, the operator is asked to control the flippers manually.

We propose several TTE strategies. The simplest TTE strategy selects the bin to be explored randomly from the set of all missing bins—we refer to this strategy as **Random**. Further, we propose and evaluate also two better TTE strategies: (i) the Reinforcement-Learning–based strategy trained on synthetically generated training exploration roll-outs (further referred to as **RL** strategy), and (ii) a strategy based on *Automatic Relevance Determination* coefficients for QPDFs modeled by the GP (further referred to as the **ARD** strategy).

[3]Due to the page limitation, the detailed equations are not given here.

[4]The exploration using robotic arm is inherently slow. However, when needed, it is still worth the extra time.

A. RL from Synthetically Generated Training Set

The Reinforcement-Learning–based TTE learns a policy that minimizes the number n of tactile measurements needed to satisfy the safety condition. In our implementation, a **state** is the union of the state used in the AT task (i.e. the proprioceptive and exteroceptive measurements), and the binary mask denoting DEM bins with missing heights. **Actions** are discrete decisions to measure the height in particular bins. **Rewards** equal zero until a final state is reached. In the final state, the roll-out ends and a reward equal to $1/n$ is assigned (i.e. the longer it takes, the lower the reward).

Since it is not easy to collect sufficient amount of real examples with naturally missing features, we generate training samples from the real data with synthetically occluded DEMs. The active exploration policy is thus trained by revealing the already known (but synthetically occluded) heights. The Q-learning algorithm learns the strategy in several episodes. The initial training set is generated by simulating thousands of TTE roll-outs with the *Random* strategy. The Q function is modeled by a Regression Forest similar to the one used in Section V (but this Q function is different from the one used for Autonomous Control!). Once the Q function is learned, the corresponding strategy is used to guide training data collection in the following episode by the DAgger algorithm [35]. In each episode of the DAgger algorithm, the learned policy is used to select bins just with 0.5 probability, otherwise the Random strategy is used (which supports exploration in the policy space). After each episode, the policy is updated using the Q-learning recurrent formula (Equation 1).

B. ARD for Gaussian Processes

In Section V-B, it is mentioned that both SE and RQ kernels allow for *Automatic Relevance Determination* (ARD), which acts as feature selection. The ARD values are computed during kernel hyper-parameters optimization (when training the GP), so no extra computing power is needed. When the learning is done, for each dimension (feature) d of the input data, we have a number $ARD(d)$ that describes how much this dimension influences the output of the GP (lower values mean higher importance). The TTE strategy utilizing ARD values is as follows: **i)** estimate QPDFs using all GP models, **ii)** select the action (GP model) with the highest Q-value (QPDF mean), **iii)** in this GP, compare $ARD(d)$ values for all DEM bin features that are missing in the current state, and choose the bin with the minimum $ARD(d)$ value, **iv)** the chosen bin is then explored using the arm. This corresponds to choosing the missing feature whose value, if known, maximally influences the QPDFs.

EXPERIMENTS

Experimental evaluation is divided into three sections. In Section VII, we test the ability of AC to decrease cognitive load of human operators while maintaining roughly the same or better performance. Experiments in Section VIII demonstrate that if the DEM is partially occluded, the proposed method

yields better results than the previous methods. Last, Section IX compares Random, ARD and RL methods for tactile exploration.

In the experiments, different Q function/QPDF representations are denoted by **PWC** for piecewise constant function proposed in [2], **GP-RQ** stands for Gaussian Processes with Rational Quadratic kernel used in [3], **GP-SE** denotes the Uncertain GPs with Squared Exponential kernel, and finally the Regression Forests defined in Section V are referred to as **Forest**. The PWC and GP-RQ models can be used either with Least Squares (**LSq**) interpolation of missing features, or with **Gibbs** sampling used to marginalize the Q function over the missing data. Regression Forests utilize the **Multiple Leaves marginalization**.

A metric called *success rate* is used throughout the experiments to measure the traversal performance both on training data (in the learning phase) and on test data. It requires that the bipolar manually-assigned part of reward $s_{c,\mathbf{x}}$ defined in Section IV is assigned for all actions in all states in the dataset (not just for a single action, as is required for the learning). The success rate denotes the ratio of states, in which the AC algorithm selects one of the desired (safe) configurations. Formally:

$$\text{success rate}(\mathbf{X}) = \frac{|\{\mathbf{x} \in \mathrm{X} : c = c^*(\mathbf{x}) \wedge s_{c,\mathbf{x}} = 1\}|}{|\mathbf{X}|} \qquad (8)$$

where $\mathbf{X}$ is a set of states, and $c^*(\mathbf{x})$ is the optimal configuration from Equation 3 or Equation 5 (depends on the used AC algorithm).

VII. Autonomous Control for Teleoperation

We evaluate performance of the AC algorithm (without tactile exploration) on a large dataset comprising of 8 different obstacles (some of them depicted in Fig. 5) in 3 types of environment (forest, stairs, hallway) with the robot driven by 3 different operators in both MC and AC modes[5]. Each of the traversals is repeated 3-10 times to allow for statistical evaluation. The operators driving the robot are denoted as **E** (Experienced), **IE** (InExperienced) and **IE2** (InExperienced #2). The experiments cover more than 115 minutes of robot time during which the robot traveled over 775 meters.

Experiments in this section only show the results achieved with Regression Forests; other Q function representations were tested in [2], [3], and Uncertain Gaussian Processes were only tested together with the tactile exploration (see Section IX), since without TTE they performed worse than the Regression Forests (and for creating such a large dataset, we had to choose one method).

A. Training Procedure

The algorithm was trained in controlled lab conditions using two artificial obstacles created from EUR pallets[6] and a staircase. The first obstacle is just a single pallet and the second one is a simple simulated staircase composed from three pallets.

[5]See the attached multimedia showing the test drives.

[6]Type EUR 1: 800×1200×140 mm, see en.wikipedia.org/wiki/EUR-pallet

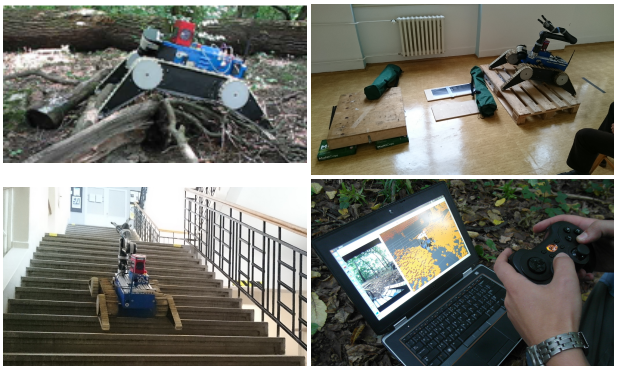


Fig. 5. Top left: *Forest* obstacle. Top right: *Rubble* obstacle. Bottom left: *Stairs* obstacle. Bottom right: Operator controlling the robot using only sensor data.

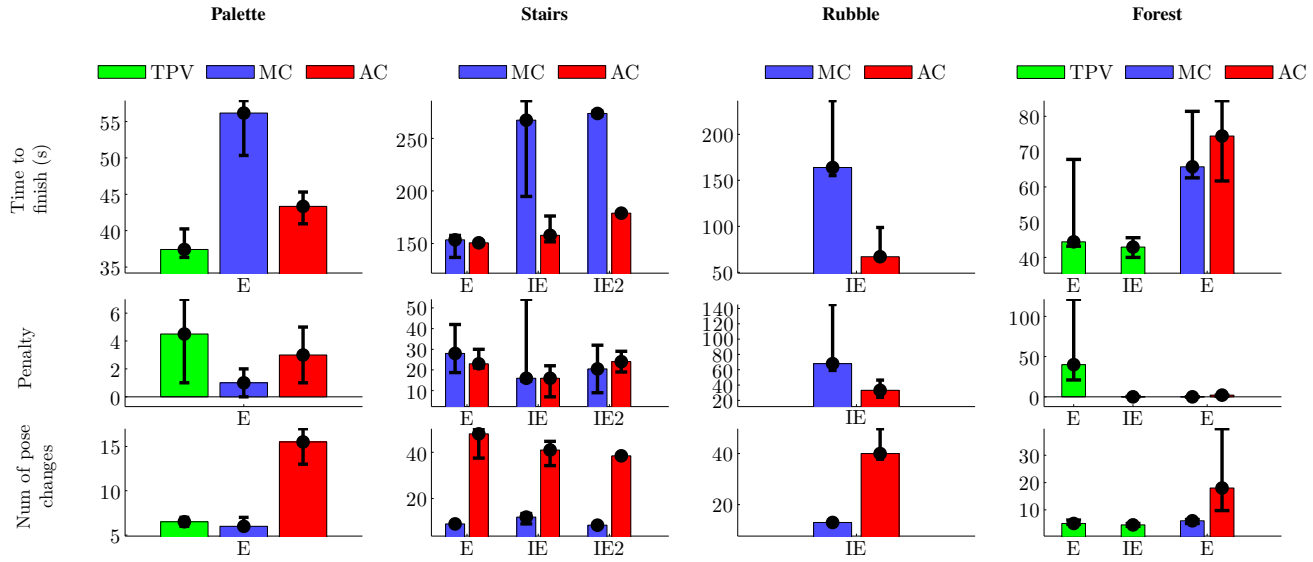
We trained the QPDFs represented by RF (one QPDF per flipper configuration) using the algorithm described in Section V. Except the standard learning validation metrics, we also evaluated the *success rate* (Equation 8). We trained the RF QPDF model, and we accomplished a success rate of 97 % (which is shown in Fig. 6).

B. Testing Procedure

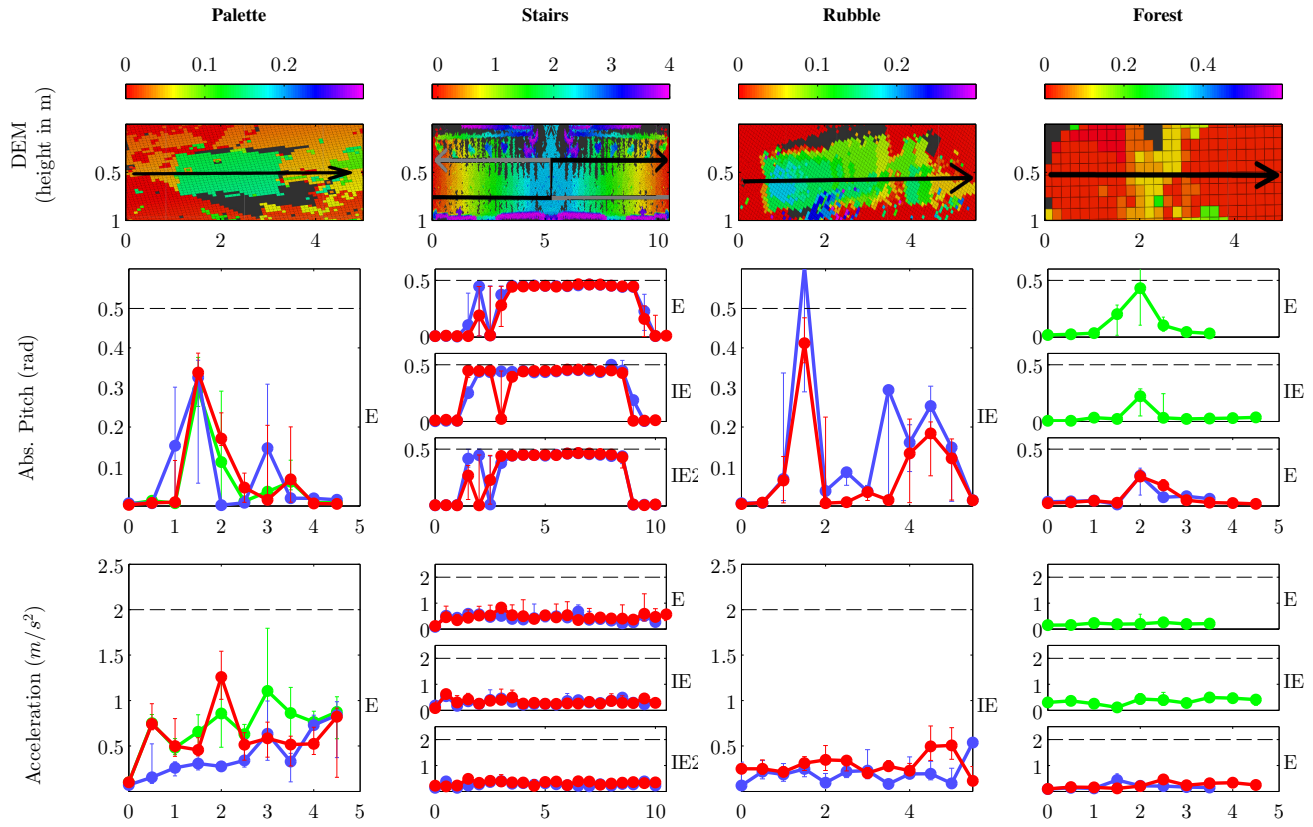
Each obstacle was traversed multiple times with both manual (**MC**) and autonomous flipper control (**AC**) using RFs following Equation 5, and the sensed states contained naturally missing DEM features. We emphasize that the complexity of testing obstacles was selected in order to challenge robot hardware capabilities. See the examples in Fig. 5 and the elevation maps (DEM) of testing obstacles computed online by the robot in Table IIb.

There is an additional mode called **TPV** (*Third Person View*) in which the operator had not only the robot sensory data available, but he directly looked at the robot (thus having much more information than the robot can get). Except for the TPV mode, the operators were only allowed to drive the robot based on data coming from the robot sensors (3D map + robot pose from sensor fusion [36]), which should accomplish a fair comparison of AC and MC. The TPV mode should be treated as a sort of baseline—it is not expected that AC or MC could be better than TPV in all aspects.

To compare AC and MC quality, three different metrics were proposed and evaluated: (i) traversal time (start and end points are defined spatially), (ii) a sum of pitch angle penalty and roughness of motion penalty, and (iii) the number of flipper configuration changes (which increases cognitive load of the operator in MC, and with the current manual controller, it also takes approx. 1 s to change the flipper configuration and the robot has to be stopped). Table IIa and Table III show quantitative evaluation of some of the experiments. Table IIa depicts 4 out of 8 experiments carried out to verify performance of AC using the best method found—Regression Forests with Multiple Leaves marginalization. All errorbars denote quartiles of the measured values and the circles are in the positions of medians.



(a) **Overall statistics of the experiments.** The computation of penalties is described in (b). High penalties for experienced operator with 3rd person view (TPV) are given by the fact that an experienced operator allows himself to drive harsher to finish the task faster.



(b) **Penalty details.** The horizontal axis always denotes distance traveled during the experiment. Dashed lines in *Pitch* and *Acceleration* show the thresholds (0.5 rad or 2.5 $\mathrm{m\,s^{-2}}$) for counting a *penalty point* (which are plotted in Table IIa). *Acceleration* reflects the "roughness of motion" (the higher it is, the worse for the mechanical construction of the robot). It is computed as $\sqrt{a_x^2 + a_z^2}$ and is averaged over 0.2 s intervals (where a_x is the horizontal acceleration perpendicular to robot motion, and a_z is vertical acceleration with gravity subtracted).

TABLE II. EXPERIMENTAL EVALUATION OF ADAPTIVE TRAVERSABILITY

Table III summarizes all MC/AC experiments (excluding TPV mode experiments, since they should not be compared with MC/AC).

TABLE III. EVALUATION OF ALL AT EXPERIMENTS

Obstacle	Operator	Time to finish [s]		Penalty		Pose changes	
		MC	AC	MC	AC	MC	AC
Pallet long	E	56.2	**43.3**	**1**	3	**6**	16
Pallet short	E	41.0	**39.3**	**2**	4	**6**	15
Stairs	E	154.0	**150.6**	28	**23**	**9**	48
	IE	267.3	**157.9**	**16**	**16**	**12**	41
	IE2	273.7	**178.8**	**21**	24	**9**	39
Rubble 1	IE	164.0	**66.9**	68	**33**	**13**	40
Rubble 2	IE2	114.0	**63.2**	7	**3**	**10**	26
Forest 1	E	**65.7**	74.4	**0**	2	**6**	18
Forest 2	E	36.8	**35.7**	N/A	N/A	**2**	3
Forest 3	E	132.1	**75.3**	N/A	N/A	**4**	10

Each pair of columns (MC/AC) shows the medians of the 3 metrics evaluated for the experiments. Of each pair, the value in bold is better. Experiments *Forest 2* and *Forest 3* are those conducted in [2]. Both robot construction and AT algorithm changed in the meantime, so the values should not be compared to the new results.

C. Results

It can be seen in Table III that the *Time to finish* with AC tends to be shorter or comparable to MC (and with TPV, it is even shorter, as expected). Subjectively, the operators report a much lower level of cognitive load when driving with AC, which means they can pay more attention to exploration or other tasks.

Penalties with AC are also mostly better or comparable to MC. The *number of flipper configuration changes* for AC is approximately 2- to 4-times higher than for MC. However, with AC, there is no time penalty for changing flipper configurations, and it also adds no more cognitive load to the operator.

From the experiments conducted it follows that AC yields similar or even better performance than MC. Furthermore, AC allows the operator to concentrate rather on higher-level tasks while having the tedious and low-level flipper control done automatically.

VIII. ROBUSTNESS TO MISSING EXTEROCEPTIVE DATA

In this experiment, we quantitatively evaluate robustness to the number of missing features for the various Q/QPDF representations. The robustness is presented as the relation between success rate and the number of synthetically occluded DEM bins.

The Regression Forests first compute the marginalized QPDFs as described in Section V, and then choose a configuration according to Equation 5. The LSq interpolation/Gibbs sampling methods first interpolate or marginalize the missing data, then compute the Q function on the interpolated data and choose the configuration according to Equation 3.

For this experiment, a dataset consisting of hundreds of captured robot states (interoceptive + full exteroceptive features) is used. The bipolar manual annotations $s_{c,\mathbf{x}}$ are assigned to all state-action combinations.

For $i = 0 \ldots 100$, the set "states$_i$" is generated from the dataset by occluding i DEM bins in each of the captured states $\mathbf{x}$ (the same manual annotation $s_{c,\mathbf{x}}$ is used for all states $\tilde{\mathbf{x}}$ generated from $\mathbf{x}$). To avoid combinatorial explosion, we did not try all combinations of i occluded bins. We chose to successively occlude DEM bins from the front of the robot, until i bins are occluded. Therefore, the dataset the robustness is tested on contains tens of thousands of different states. The success rate in Fig. 6 is computed as success rate(states_i) according to Equation 8.

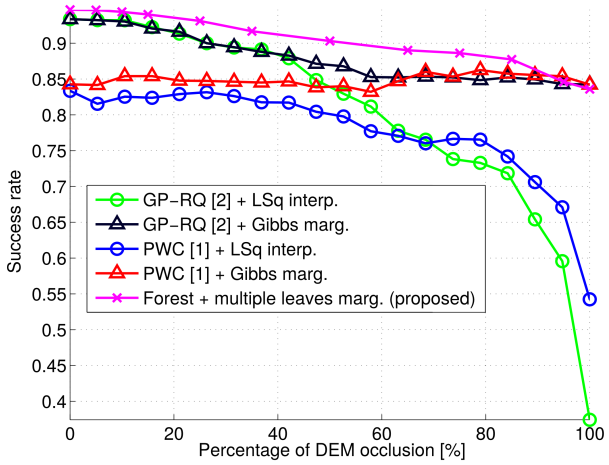


Fig. 6. **Robustness to DEM occlusion:** The chart shows the influence of DEM occlusion (percentage of DEM bins in which measurements are not available) on AC success rate. When 100 % of DEM is occluded, the marginalized policies still depend on proprioceptive measurements, while LSq interpolation reconstructs only flat terrain.

Fig. 6 shows superiority of marginalizing methods over LSq interpolation. Up to a DEM occlusion level of 40 %, all methods behave comparably. The reasons are two-fold: (i) the part of the occluded DEM is far in front of the robot and there is no way to sense it from the proprioceptive measurements, (ii) the obstacle hidden in this part of DEM is usually far enough, therefore the V-shape configuration (the one for flat terrain) is still allowed in most of the testing data. When more than 40 % are hidden, success rate of the LSq interpolation method drops rapidly down towards $0.4 - 0.5$ (i.e. 40 %-50 % of states in which the permitted configuration is selected) for both GP and PWC, while the marginalizing methods preserve high precision. The figure also demonstrates that the proposed Regression Forests provide better success rate than the previous methods [2], [3].

IX. TACTILE TERRAIN EXPLORATION

To compare the strategies for Tactile Terrain Exploration (TTE), we evaluate them on real (test) data with the front 50 % of the DEM synthetically occluded (since it is not easy to provide a sufficient amount of real examples with naturally missing features). Active exploration is simulated by revealing the already known DEM heights.

The performance of TTE strategies can be expressed as the average number of actively measured bin heights until a safe configuration is found. However, for this experiment, we let the exploration continue even if a safe configuration has already been found, to see how much further exploration helps.

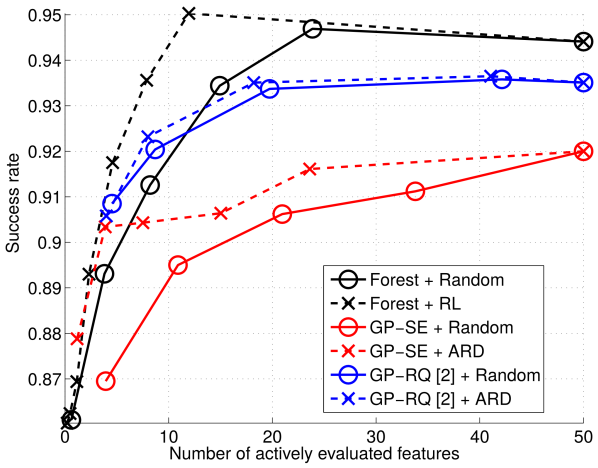


Fig. 7. **Comparison of TTE methods:** Curves on this graph show success rate (with 50 % DEM bins occluded) as a function of the number of measured bin heights. The compared TTE strategies are described in Section VI.

For different QPDF models and TTE strategies, the relation between the number of measured heights and the success rate is depicted in Fig. 7.

An ideal QPDF model and strategy would achieve 100 % success rate with a single evaluated feature, i.e. the upper-left corner in Fig. 7. The closer is the curve to this corner, the better is the method. Results with the lowest success rate were achieved with the GP-SE method (however, the ARD strategy yields a significant improvement). Better results were achieved by the GP-RQ method (for which the ARD strategy yields only small improvement compared to the Random strategy). The reason is that the RQ kernel allows for better generalization than the SE kernel. For less than 15 features actively evaluated (i.e. smaller safety thresholds), the GP-RQ method achieves higher success rate than the Regression Forest method with Random strategy. The best method in this comparison are Regression Forests combined with the RL strategy, which achieve the best success rate.

X. Conclusion

We extended the Autonomous Control algorithm [2], [3] that increases autonomy in mobile robot control and reduces cognitive load of the operator. To deal with only partially observable terrain, missing or incorrect data, we (i) designed and experimentally verified a more occlusion-robust QPDF model, and (ii) we exploit a body-mounted robotic arm as an additional active sensor for Tactile Terrain Exploration. TTE is used in dangerous situations, where all actions have negative expected rewards. The previous methods have to choose one of the actions, even if the best expected reward is negative. By tactile exploration of the unobserved part of the terrain, the reward estimates get better and at least one of them should get positive if the terrain is traversable. Several TTE strategies were proposed and experimentally evaluated. We conclude that the overall highest success rate was achieved by combining Regression Forests with the RL strategy for the arm-based exploration of missing data.

References

[1] G. J. M. Kruijff et al., "Designing, developing, and deploying systems to support human-robot teams in disaster response," *Advanced Robotics*, vol. 28, no. 23, pp. 1547–1570, 2014.

[2] K. Zimmermann, P. Zuzánek, M. Reinstein, and V. Hlaváč, "Adaptive traversability of unknown complex terrain with obstacles for mobile robots," in *Proc. IEEE Int. Conf. Rob. Autom.*, 2014, pp. 5177–5182.

[3] K. Zimmermann, P. Zuzánek, M. Reinstein, T. Petříček, and V. Hlaváč, "Adaptive traversability of partially occluded obstacles," in *Proc. IEEE Int. Conf. Rob. Autom.*, 2015.

[4] F. Colas, S. Mahesh, F. Pomerleau, M. Liu, and R. Siegwart, "3D path planning and execution for search and rescue ground robots," in *IEEE Int. Conf. Intell. Rob. Syst.*, Nov 2013, pp. 722–727.

[5] M. Brunner, B. Bruggemann, and D. Schulz, "Towards autonomously traversing complex obstacles with mobile robots with adjustable chassis," in *13th Int. Carpathian Cont. Conf.*, 2012, pp. 63–68.

[6] C.-C. Tsai, H.-C. Huang, and C.-K. Chan, "Parallel elite genetic algorithm and its application to global path planning for autonomous robot navigation," *IEEE Tran. Ind. Electron.*, vol. 58, pp. 4813–4821, 2011.

[7] M. Gianni, F. Ferri, M. Menna, and F. Pirri, "Adaptive robust three-dimensional trajectory tracking for actively articulated tracked vehicles (AATVs)," *Jour. Field Rob.*, 2015.

[8] S. Martin, L. Murphy, and P. Corke, "Building large scale traversability maps using vehicle experience," in *Int. Symp. Experimental Rob.*, 2013, pp. 891–905.

[9] K. Ho, T. Peynot, and S. Sukkarieh, "A near-to-far non-parametric learning approach for estimating traversability in deformable terrain," in *IEEE Int. Conf. Intell. Rob. Syst.*, Nov 2013, pp. 2827–2833.

[10] P. Abbeel, A. Coates, M. Quigley, and A. Y. Ng, "An application of reinforcement learning to aerobatic helicopter flight," in *Advances in Neural Inform. Process. Syst. 19*. MIT Press, 2007, p. 2007.

[11] C. Weiss, H. Frohlich, and A. Zell, "Vibration-based terrain classification using support vector machines," in *Proc. IEEE Int. Conf. Intell. Rob. Syst.*, 2006, pp. 4429–4434.

[12] K. Kim, K. Ko, W. Kim, S. Yu, and C. Han, "Performance comparison between neural network and SVM for terrain classification of legged robot," in *Proc. Soc. Instrument Cont. Eng. Annual Conf.*, 2010, pp. 1343–1348.

[13] E. M. DuPont, C. A. Moore, and R. G. Roberts, "Terrain classification for mobile robots traveling at various speeds: An eigenspace manifold approach," in *Proc. IEEE Int. Conf. Rob. Autom.*, 2008, pp. 3284–3289.

[14] D. Kim, J. Sun, S. Min, O. James, M. Rehg, and A. F. Bobick, "Traversability classification using unsupervised on-line visual learning for outdoor robot navigation," in *Proc. IEEE Int. Conf. Rob. Autom.*, 2006, pp. 518–525.

[15] L. Ojeda, J. Borenstein, G. Witus, and R. Karlsen, "Terrain characterization and classification with a mobile robot," *Jour. Field Rob.*, vol. 23, pp. 103–122, 2006.

[16] K. Ho, T. Peynot, and S. S. Sukkarich, "Traversability estimation for a planetary rover via experimental kernel learning in a Gaussian Process framework." in *Proc. IEEE Int. Conf. Rob. Autom.*, 2013, pp. 3475–3482.

[17] M. Reinstein, V. Kubelka, and K. Zimmermann, "Terrain adaptive odometry for mobile skid-steer robots," in *IEEE Int. Conf. Rob. Aut.*, 2013, pp. 4706–4711.

[18] M. Reinstein and M. Hoffmann, "Dead reckoning in a dynamic quadruped robot based on multimodal proprioceptive sensory information," *IEEE Tran. Rob.*, vol. 29, no. 2, pp. 563–571, 2013.

[19] G. Zhong, H. Deng, G. Xin, and H. Wang, "Dynamic hybrid control of a hexapod walking robot: Experimental verification," *IEEE Tran. Ind. Electron.*, vol. PP, no. 99, pp. 1–1, 2016.

[20] P. Abbeel and A. Y. Ng, "Exploration and apprenticeship learning in reinforcement learning," in *Proc. 22nd Int. Conf. Machine Learning*, 2005, pp. 1–8.

[21] J. Yu, C. Wang, and G. Xie, "Coordination of multiple robotic fish with applications to underwater robot competition," *IEEE Tran. Ind. Electron.*, vol. PP, no. 99, pp. 1–8, 2015.

[22] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake, "Real-time human pose recognition in parts from single depth images," in *IEEE Conf. Comput. Vision Pattern Recognition*, 2011, pp. 1297–1304.

[23] M. Deisenroth, D. Fox, and C. Rasmussen, "Gaussian Processes for data-efficient learning in robotics and control," *IEEE Tran. Pattern Anal. Machine Intell.*, vol. 37, no. 2, pp. 408–423, 2015.

[24] D. J. Lizotte, L. Gunter, E. Laber, and S. A. Murphy, "Missing data and uncertainty in batch Reinforcement Learning," in *Proc. Neural Inform. Process. Syst.*, 2008.

[25] M. A. Tanner and W. Wong, "The calculation of posterior distributions by data augmentation," *Jour. Amer. Stat. Assoc.*, vol. 82, pp. 528–540, 1987.

[26] Z. Ghahramani and M. I. Jordan, "Supervised learning from incomplete data via an EM approach," in *Neural Inform. Process. Syst.* Morgan Kaufmann, 1994, pp. 120–127.

[27] A. Doumanoglou, T.-K. Kim, X. Zhao, and S. Malassiotis, "Active random forests: An application to autonomous unfolding of clothes," in *European Conf. Comput. Vision*. Springer, 2014, pp. 644–658.

[28] M. Bjorkman, Y. Bekiroglu, V. Hogman, and D. Kragic, "Enhancing visual perception of shape through tactile glances," in *Proc. IEEE Int. Conf. Intell. Rob. Syst.*, 2013, pp. 3180–3186.

[29] Z. Jia, Y.-J. Chang, and T. Chen, "A general boosting-based framework for active object recognition," in *British Machine Vision Conf.*, 2010, pp. 1–11.

[30] C. J. Watkins and P. Dayan, "Q-learning," *Machine Learning*, vol. 8, no. 3-4, pp. 279–292, may 1992.

[31] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement learning: A survey," *Jour. of Artificial Intell. Research*, vol. 4, pp. 237–285, 1996.

[32] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "OctoMap: An efficient probabilistic 3D mapping framework based on octrees," *Auton. Rob.*, 2013.

[33] M. P. Deisenroth, *Efficient Reinforcement Learning using Gaussian Processes*. KIT Scientific Publishing, 2010, vol. 9.

[34] C. E. Rasmussen, "Gaussian Processes in machine learning," in *Advanced Lectures Machine Learning*. Springer, 2004, pp. 63–71.

[35] S. Ross and J. A. Bagnell, "Agnostic system identification for model-based reinforcement learning," in *Proc. 29th Int. Conf. Machine Learning*, 2012, pp. 1703–1710.

[36] V. Kubelka, L. Oswald, F. Pomerleau, F. Colas, T. Svoboda, and M. Reinstein, "Robust data fusion of multimodal sensory information for mobile robots," *Jour. Field Rob.*, vol. 32, no. 4, pp. 447–473, 2015.

Martin Pecka received the Mgr. (M.Sc.) degree in theoretical informatics at the Faculty of Mathematics and Physics, Charles University in Prague, Czech republic, in 2012.

He has currently been a Ph.D. student at the Department of Cybernetics, Czech Technical University in Prague (CTU), and a Research Assistant at the Czech Institute of Informatics, Robotics and Cybernetics, CTU in Prague. His main research interests are in the fields of machine learning, specifically reinforcement learning concerning safety of execution, and robotics.

Karel Zimmermann (M'08) received the Ph.D. degree in cybernetics from the Czech Technical University in Prague, Czech Republic, in 2008.

He worked as Postdoctoral Researcher with the Katholieke Universiteit Leuven (2008-2009). Since 2009, he has been a Postdoctoral Researcher at the Czech Technical University in Prague. His current research interests include learnable methods for tracking, detection and robotics.

Dr. Zimmermann serves as a reviewer for major journals such as TPAMI, IJCV and conferences such as CVPR, ICCV, ICRA, IROS. He received the best reviewer award at CVPR 2011 and the main prize for the best PhD thesis in Czech Republic in 2008 (awarded by the Czech Society for Pattern Recognition).

Michal Reinstein (M'11) received the Ing. (M.Sc.) and Ph.D. degrees in engineering of aircraft information and control systems from the Faculty of Electrical Engineering, Czech Technical University in Prague (CTU), Czech republic, in 2007 and 2011, respectively.

He is currently working as Assistant Professor at the Center for Machine Perception, Dept. of Cybernetics, CTU in Prague. His most recent research interests concern application of machine learning and data fusion to satellite imagery aiming to support deployment of large-scale robotic platforms. In the past years since 2011, the main topics of his research were: sensory-motor interaction in legged robots, multimodal data fusion for robots intended for Urban Search & Rescue, and adaptive traversability of unknown terrain using reinforcement learning.

Tomas Svoboda (M'01) received the Ph.D. degree in artificial intelligence and biocybernetics from the Czech Technical University in Prague, Czech republic, in 2000.

Later, he spent three post-doc years with the Computer Vision Group at the ETH Zurich. Currently, he is Associate Professor and Deputy Head of the Department of Cybernetics at the Czech Technical University in Prague, the Director of EECS study programme, and he is also on board of Open Informatics programme. He has published papers on multicamera systems, omnidirectional cameras, image based retrieval, learnable detection methods, and USAR robotics. His current research interests include multimodal perception for autonomous systems, object detection and related applications in automotive industry.