A Bayesian Regression Approach to Terrain Mapping and an Application to Legged Robot Locomotion

A Bayesian Regression Approach to Terrain Mapping and an Application to Legged Robot Locomotion

Christian Plagemann*
Stanford University
Artificial Intelligence Lab
Stanford, USA

Sebastian Mischke
University of Freiburg
Dept. of Computer Science
Freiburg, Germany

Sam Prentice
Massachusetts Institute
of Technology, CSAIL
Cambridge, USA

Kristian Kersting
Fraunhofer Institute IAIS
Dept. of Knowledge Discovery
Sankt Augustin, Germany

Nicholas Roy
Massachusetts Institute
of Technology, CSAIL
Cambridge, USA

Wolfram Burgard
University of Freiburg
Dept. of Computer Science
Freiburg, Germany

Abstract

We deal with the problem of learning probabilistic models of terrain surfaces from sparse and noisy elevation measurements. The key idea is to formalize this as a regression problem and to derive a solution based on nonstationary Gaussian processes. We describe how to achieve a sparse approximation of the model, which makes the model applicable to real-world data sets. The main benefits of our model are (1) that it does not require a discretization of space, (2) it also provides the uncertainty for its predictions and (3) it adapts its covariance function to the observed data, allowing more accurate inference of terrain elevation at points that have not been observed directly.

As a second contribution, we describe how a legged robot equipped with a laser range finder can utilize the developed terrain model to plan and execute a path over rough terrain. We show how a motion planner can use the learned terrain model to plan a path to a goal location, using a terrain-specific cost model to accept or reject candidate footholds.

1 Introduction

Terrain models are used to represent the distribution of surface elevation in an environment. They are applied in numerous areas, such as weather forecasting, city planning, geological surveys, computer game development and also outdoor robotics. This paper focuses on terrain modeling in robotics applications, in which the task is to statistically model a set of noisy elevation samples for the purpose of planning and executing motion plans. The approach, however, is formulated in a general form, such that it can be applied in other domains as well.

*Contact Information: plagemann@stanford.edu. 353 Serra Mall, Stanford, CA 94305-9010, Phone: +1 (650) 723-9558.

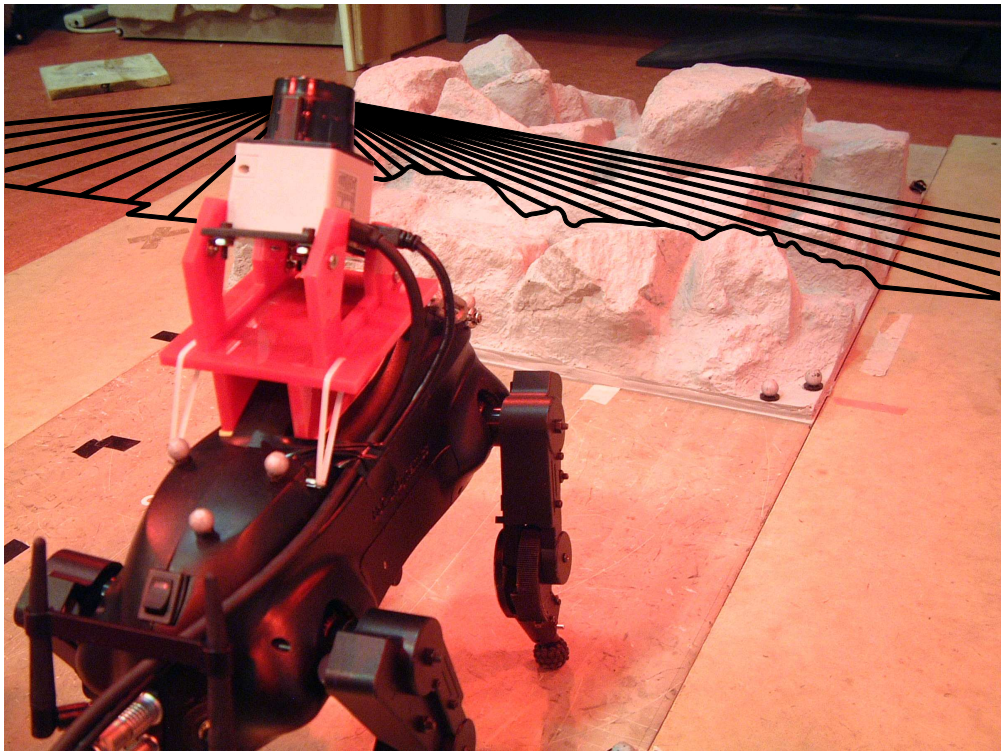


Figure 1: Our quadruped robot is equipped with a laser sensor in order to observe its local surrounding. By bending its leg joints, it is able to acquire dense 3D scans of the environment.

As a motivating scenario, consider the situation depicted in Fig. 1. Our quadruped robot, specifically the Boston Dynamics LittleDog, is equipped with a laser range finder as its primary sensor. It faces the task of planning and executing a path over rough terrain. In this paper, we assume that the true terrain elevation is a well-defined *function* of 2D locations. Overhanging structures, which would violate this assumption, can be dealt with by considering mixtures of terrain models or by other means—as will be outlined later in the paper. Acquiring and representing models of such terrain is a challenging task. First of all, terrains are defined over a continuous space such that the space of all models has in principle infinitely many dimensions. Discretizing this space, for example with a grid representation, can result in models of enormous size and can introduce a strong bias due to an improper choice of grid resolution. Furthermore, we must rely on the robot's noisy sensors to gather information about the world, which requires statistical inference in the continuous and high-dimensional space of the model. The amount of noise in the range measurements is comparatively high due to a number of factors including the noise of the light-weight sensor that was required, accumulated calibration errors and the fact that the position of the robot cannot be known perfectly. Standard discretizations do not allow the inference process to take advantage of correlations between measurements, resulting in a loss of information that leads to noisier terrain estimates, in turn leading to the selection of statically unstable and kinematically infeasible configurations of the robot. Finally, we wish to be able to deal with a varying data density, to balance smooth inference of the terrain against the preservation of discontinuities and to make sensible predictions about unseen parts of the terrain.

To this aim, we developed a novel, probabilistic terrain modeling approach based on a Gaussian

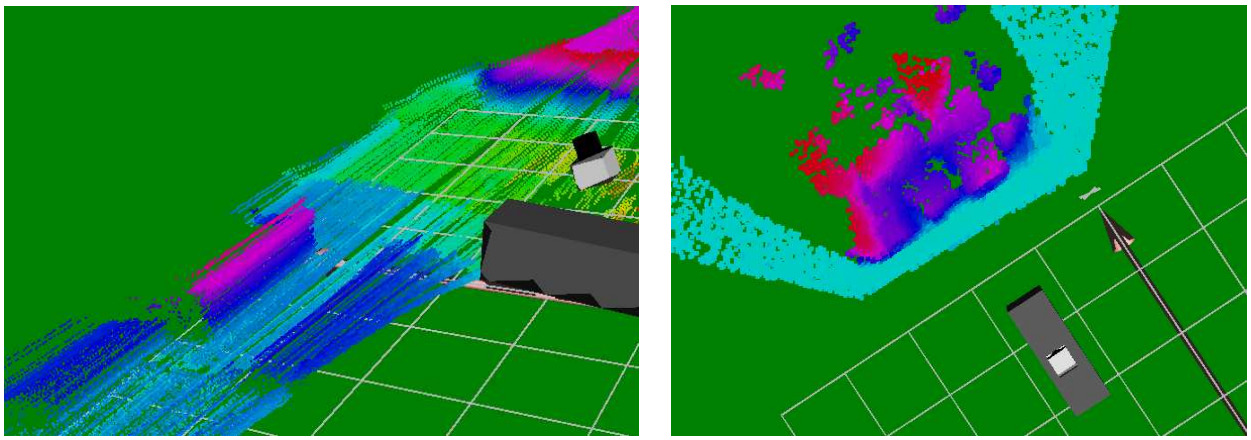


Figure 2: Online visualization of the scanning procedure using the controller application of our implemented system. The robot bends its legs to acquire a 3D scan (left) and learns a probabilistic terrain regression model (right). The terrain elevations are color coded, ranging from green/blue (lowest elevation) to pink/red (highest).

process (GP) formulation that adapts its generalization behavior to local structure in the terrain and that can be learned efficiently using a decomposition of the model into smaller, overlapping sub-models. As a result, our model deals with noisy data in a statistically sound manner, does not require a discretization of space and provides estimates of the predictive uncertainty so that predictions about unseen locations can be treated more cautiously in further processing. An additional distinctive advantage of our model is that it also yields an estimate of the terrain gradient and its uncertainty. The terrain gradient is an important feature for selecting stable foothold locations for the legged robot.

As a second contribution of this paper, we describe how the terrain mapping approach was implemented on the legged robot depicted in Fig. 1 to enable it to autonomously sense and traverse rough terrain. Specifically, we discuss how the system was calibrated, how our locally adaptive GP model allows us to select safe foothold locations and how to plan a path to a goal location. Figure 2 shows screen shots of our controller application in some example situations, i.e., during scan acquisition (left) and terrain adaptation (right). To the best of our knowledge, we describe here the first legged robotics system to autonomously sense, plan and traverse a terrain surface of the given complexity.

In the following, we first relate our approach to other works in the literature. In Sec. 3, we formalize terrain modeling as a nonparametric regression problem, describe how to adapt to local terrain structure and show how to increase the time and space efficiency of the approach by factorizing the model into overlapping submodels. Sec. 4 presents our experimental evaluation on real and simulated data and Sec. 5 documents the integration and evaluation of the terrain mapper on our quadruped robot.

2 Related Work

Terrain modeling and map building are central tasks within robotics and other disciplines. A broad overview of methods used for modeling terrain data was given by Hufentobler (2004). Elevation

maps (or "elevation grids") have been used as an efficient data structure for representing dense terrain data (Bares et al., 1989; Pfaff and Burgard, 2005; Parra et al., 1999; Hygounenc et al., 2004). The central idea is to partition the input space into quadratic cells of equal size and to merge observations falling in the same cell to a single elevation value. Grids allow for constant time access to the elevation data and the time required to update the cells is linear in the number of measurements. Extensions to more flexible 3D representations have been addressed in recent work (Pfaff and Burgard, 2005). Multi-level surface maps (Triebel et al., 2006) allow for modeling of both overhanging structures and vertical objects.

The general problem of grid-based approaches is the choice of the cell size as well as how to adapt the grid resolution locally. Choosing a too large cell size limits the representable complexity of the underlying terrain. A small cell size, on the other hand, may lead to many under-determined cells where uncertainties are high, or even cells without any observations, for which no prediction can be made. Consequently, there has also been much work on filling gaps in grid maps. Different smoothing methods can be applied. Bi-linear and other means of local interpolation (Rees, 2000; Polidori and Chorowicz, 1993) are often used to preprocess dense and accurate data. Within robotics, Früh et al. (2005), who addressed the automated acquisition of three-dimensional city models, applied local interpolation to fill gaps in the acquired data. Triangle-based interpolation, e.g. based on Delaunay tessellation (Delone and Novikov, 1993; Barber et al., 1996), is similar to bi-linear interpolation, but can be applied more readily to points that are not distributed equidistantly.

As sensors for acquiring terrain elevation data, laser range finders are a popular choice, as they measure the geometry of the environment directly and accurately. Kweon and Kanade (1992) used laser sensors to sense terrain elevation and a grid model to build maps. Geometric reasoning is used in their approach to model the uncertainty in the grid. Huber and Hebert (1999) deal with the problem of aligning multiple scans and discuss how to handle resolution limits. Millimeter-wave radar was used by Foessel et al. (2001) to learn occupancy grid-based terrain models.

Miller (2002) fused the measurements from a laser range finder and a calibrated camera mounted on a helicopter to construct terrain models. Another camera-based approach was proposed recently by Kolter et al. (2009), who equipped a quadruped robot with a stereo-camera system and dealt with the problem of how to infer a dense elevation map of the terrain from the sparse stereo-correspondences. Popular approaches from the computer vision literature that do not require a stereo setup are to extract 3D shape from shading (Bors et al., 2003) or from shadows (Daum and Dudek, 1998).

An alternative approach to dealing with non-flat terrain within robotics is to assume a parametric model of terrain types and to learn the parameters of the model to predict the vehicle dynamics (Iagnemma et al., 2004). Brooks and Iagnemma (2009) proposed to learn classifiers for terrain types and Wellington et al. (2005) presented a generative model to infer layers of terrain types including hidden obstacles.

In parallel work to ours, Hadsell et al. (2009) developed a different kernel-based approach to terrain reconstruction for robotics applications, that explicitly takes the constraints induced by the perceived free space into account. By the time of writing, documentation of their work was not available yet.

The terrain regression algorithm presented in this paper builds on the Gaussian process (GP) model (see, e.g., Rasmussen and Williams (2006) for an overview and additional references). Compared to grid-based representations it has the advantage of not assuming a fixed discretization of space and of providing a sound and direct way of estimating predictive uncertainties. The explicit model of uncertainty that a GP provides has led to their successful application in a wide range of other applications areas such as for developing positioning systems using cellular networks (Schwaighofer et al., 2004).

Within robotics, GP models have recently become popular, for instance for measurement modeling (Brooks et al., 2006) or for model-based failure detection (Plagemann et al., 2007), because they naturally deal with noisy measurements, unevenly distributed observations and they fill small gaps in the data with high confidence while assigning higher predictive uncertainty in sparsely sampled areas.

This work follows up on Lang et al. (2007), where we proposed the first GP-based approach to terrain regression in robotics. This work simplifies and extends the original approach in several respects. First, we exchanged the iterative procedure for fitting local lengthscales by an analytic link function and second, we show how to scale to large data sets by dividing the model into a set of overlapping submodels. We also extend Plagemann et al. (2008b) by (1) documenting the first walk of a quadruped robot over rocky terrain using own elevation observations only and (2) by giving additional technical details about all parts of the approach—including how to learn the model from data in Sec. 3.1, a description of our calibration procedure in Sec. 5.1 and a discussion of the individual parts in Sec. 4.5.

In a parallel line of research (Plagemann et al., 2008a), we are investigating how to *jointly* learn all parameters in nonstationary Gaussian process models as opposed to decoupling the optimization of local lengthscales from the hyperparameter search done in this work. While the joint approach has been demonstrated to solve a number of relevant regression problems, including terrain regression tasks, it is not ready yet to be applied to larger data sets and in time-constrained settings.

Our work builds on the nonstationary covariance function introduced by Paciorek and Schervish (2004), see also the references in there, but several other approaches for specifying nonstationary regression models can be found in the literature. For instance, Sampson and Guttorp (1992) map a nonstationary spatial process (not based on GPs) into a latent space, in which the problem becomes approximately stationary. Schmidt and OHagan (2003) followed this idea and used GPs to implement the mapping. Similar in spirit, Pfingsten et al. (2006) proposed to augment the input space by an additional latent input to tear apart regions of the input space that are separated by abrupt changes of the function values.

Additional approaches to nonstationary regression have been developed for the task of learning to control robotic manipulators. Approaches such as locally weighted projection regression, LWPR (Vijayakumar and Schaal, 2000; Vijayakumar et al., 2005), the Bayesian approach by Ting et al. (2006) and others address similar problems as the approach presented in this paper. Furthermore, Kim et al. (2005), see also references in there, used mixtures of stationary Gaussian processes with a random (Bayesian) partition model and, recently, Nguyen-Tuong et al. (2008) proposed a distance-based measure for partitioning the data set and producing weighted predictions.

Sparse approximations to GP models, see e.g. Quinonero-Candela and Rasmussen (2006), provide a way of making the approach substantially more time and space efficient. Recently, they have been combined intimately with the mixture of expert approach Snelson and Ghahramani (2007) using stationary covariance functions. In principle, such sparse approximation techniques could be combined with approaches to nonstationary regression to build an alternative, efficient nonstationary model—e.g., following Schmidt and OHagan (2003), who map the nonstationary process into a latent space where it becomes approximately stationary. Shen et al. (2005) proposed to group similar matrix-vector multiplications during GP learning and inference using *kd*-trees to speed up computations.

Legged robot locomotion has been studied intensively in the past. Overviews and descriptions of approaches can be found in Hauser et al. (2008) or Siegwart and Nourbakhsh (2004), among others. The goal is to build versatile robots that are able to traverse uneven terrain as robustly, quickly and easily as humans or animals do. This would enable the development of, e.g., autonomous disaster assistant robots or versatile mobile transportation systems. The two major problems still to be solved are (a) perception of the surrounding terrain and (b) planning and executing appropriate motion sequences. This paper primarily deals with the perception problem (sensing, representing and reasoning about terrain) and it gives a description of how the developed algorithms have been integrated into a complete quadruped system that can plan and execute paths across a perceived terrain.

3 Terrain Mapping as a Regression Problem

Traversable surfaces can be characterized by a function $f : \mathbb{R}^2 \to \mathbb{R}, f(\mathbf{x}) = z$, where $\mathbf{x} = (x, y)$ indexes a location in the 2D plane and z denotes the corresponding terrain elevation in the direction opposing the gravity vector. The task in terrain mapping is to recover the true elevation function f from a set of noisy elevation measurements $\mathcal{D} = \{\mathbf{x}_i, z_i\}_{i=1}^n$. In other terms, we wish to model the relationship between variables $\mathbf{x}_i$ and dependent variables z_i. This is called *regression analysis* in statistics.

The most common approach for solving regression problems is to assume a parametric form of f, e.g., linear, piece-wise linear, or polynomial and to fit the parameters to the observed data. In this paper, we take a different approach and model the dependency nonparametrically using the Gaussian process (GP) model [see, for example, (Rasmussen and Williams, 2006) or the references in Sec. 2]. Gaussian processes can be seen as a generalization of the Gaussian distribution to the space of functions. Analogously to the Gaussian distribution, which defines the distribution of a finite-dimensional variable in terms of a mean value and a covariance matrix, a Gaussian process defines the distribution of an (infinite-dimensional) function in terms of a mean function and a covariance function. The relationship between the two models becomes apparent from the following definition.

In the Gaussian process model, *any finite set* of samples $\{z_1, \ldots, z_n\}$ is jointly Gaussian distributed,

$$p(z_1, \ldots, z_n \mid \mathbf{x}_1, \ldots, \mathbf{x}_n) \sim \mathcal{N}(\boldsymbol{\mu}, K) \,, \tag{1}$$

with mean $\boldsymbol{\mu} \in \mathbb{R}^n$ and covariance matrix K. $\boldsymbol{\mu}$ is typically assumed $\mathbf{0}$ and K is specified in terms

of a parametric covariance function k and a global noise variance parameter σ_n^2, $K_{ij} = k(\mathbf{x_i}, \mathbf{x_j}) + \sigma_n^2 \delta_{ij}$. Intuitively, the covariance function describes how closely related several target values z_i are given their corresponding input vectors $\mathbf{x}_i$. For the most common choices of covariance functions, similar input vectors $\mathbf{x}_i$ and $\mathbf{x}_j$ lead to a large covariance value $k(\mathbf{x}_i, \mathbf{x}_j)$, which in turn leads to more highly correlated target values z_i and z_j. Thus, the covariance function can be seen as defining the smoothness of z w.r.t. to $\mathbf{x}$.

In Bayesian terminology, the covariance function k represents the prior knowledge about the target distribution as it does not depend on the observed target values z_i in $\mathcal{D}$. A common choice in practice is the squared exponential (SE) covariance function

$$k(\mathbf{x}_i, \mathbf{x}_j) = \sigma_f^2 \exp\left(-\frac{1}{2}\sum_{d=1}^{2}\frac{(\mathbf{x}_{i,d} - \mathbf{x}_{j,d})^2}{\ell_d}\right), \tag{2}$$

where σ_f denotes the amplitude (or signal variance) and ℓ_d is the characteristic lengthscale of dimension d, i.e., the *scale* of the dimension. These parameters, along with the global noise variance σ_n^2, are known as the hyperparameters of the process and denoted as $\mathbf{\Theta} = (\sigma_f, \boldsymbol{\ell}, \sigma_n)$. Intuitively, the hyperparameters define the smoothness of the functions modeled by the process as well as the noise level which leads to the observed function values.

Predictions about unobserved locations can be made analytically in the Gaussian process model. Since any set of samples from the process is jointly Gaussian distributed, the prediction of a new target value z^* at a given location $\mathbf{x}^*$ can be performed by conditioning the $n{+}1$-dimensional joint Gaussian of $\{z_1, \ldots, z_n, z^*\}$ on the known target values of the training set $\{z_1, \ldots, z_n\}$. This yields a predictive Gaussian distribution $z^* \sim \mathcal{N}(\mu^*, v^*)$ defined by

$$\mu^* = E(z^*) = \mathbf{k}^T \left(K + \sigma_n^2 I\right)^{-1} \mathbf{z}\,, \tag{3}$$

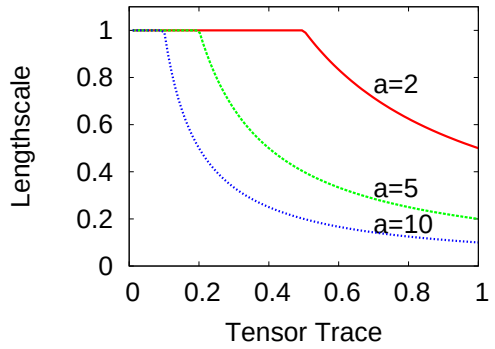
$$v^* = V(z^*) = k^* + \sigma_n^2 - \mathbf{k}^T \left(K + \sigma_n^2 I\right)^{-1} \mathbf{k}\,, \tag{4}$$

with $K \in \mathbb{R}^{n\times n}$, $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$, $\mathbf{k} \in \mathbb{R}^n$, $k_j = k(\mathbf{x}^*, \mathbf{x}_j)$, $k^* = k(\mathbf{x}^*, \mathbf{x}^*) \in \mathbb{R}$ and the training targets $\mathbf{z} \in \mathbb{R}^n$. Note that it is straightforward to predict an entire *vector* of elevation values $\boldsymbol{\mu}^*$ and the corresponding uncertainties $\mathbf{v}^*$ instead of scalar values μ^* and v^*. Details are given, for example, by Plagemann (2008) or Rasmussen and Williams (2006).

3.1 Locally Adaptive Gaussian Processes

A limitation of the standard GP framework is the assumption of constant lengthscales ℓ over the whole input space. Intuitively, the lengthscales describe the area in which observations strongly influence one another. For terrain models, one would like to use locally varying lengthscales to account for the different situations. For example, in flat plains, each terrain elevation is strongly correlated to the elevations in its neighborhood. Conversely, in high-variance, "wiggly" terrain and at discontinuities, the terrain elevations are correlated over very short distances only. To address this problem of *nonstationarity*, an extension of the squared exponential (SE) covariance function was described by Paciorek and Schervish (2004). It takes the form

$$k(\mathbf{x}_i, \mathbf{x}_i) = |\Sigma_i|^{\frac{1}{4}} |\Sigma_j|^{\frac{1}{4}} \left|\frac{\Sigma_i + \Sigma_j}{2}\right|^{-\frac{1}{2}} \cdot \exp\left[-(\mathbf{x}_i - \mathbf{x}_j)^T \left(\frac{\Sigma_i + \Sigma_j}{2}\right)^{-1} (\mathbf{x}_i - \mathbf{x}_j)\right]. \tag{5}$$



Missing data	Mean Squared Error	
	HGP	WA
15%	0.104	0.437
30%	0.168	0.531
50%	0.406	0.735

Figure 3: Left: The parametric function $\ell(\mathbf{x}_i)$ links local terrain characteristics (represented by the "tensor trace", see text) to lengthscales. Right table: Placing a *hyper-GP* (HGP) over the latent lengthscales reduces the prediction error w.r.t. to *weighted averaging* (WA) for different levels of omitted data points.

Here, each input location $\mathbf{x}$ is assigned a *local* covariance matrix Σ and the covariance between two targets z_i and z_j is calculated by averaging between the two local kernels at the input locations $\mathbf{x}_i$ and $\mathbf{x}_j$. In this way, the local characteristics at both locations influence the modeled covariance of the corresponding target values. Note the similarity between Eq. (5) and Eq. (2). For the (stationary) squared exponential covariance function in Eq. (2), we have constant lengthscales ℓ_d, while Eq. (5) allows *variable* lengthscales contained in the matrices Σ_i.

We now face the question of how to adapt the latent variables $\{\Sigma_i\}_{i=1}^n$ to the observed data. The related problem of adapting smoothing kernels to local image structure has been well studied by the computer vision community. Although image processing algorithms are typically designed for dense and uniformly distributed data, we can nevertheless use findings from the field to solve the terrain regression task. Middendorf and Nagel (2002) present a technique for iterative kernel adaptation in the context of optical flow estimation in image sequences. Their approach builds on the concept of the so called *gray-value structure tensor* (GST), which captures the local structure of an image or image sequence by building the locally weighted outer product of gray-value gradients in the neighborhood of the given image location. Translated to the terrain regression task, we consider the *elevation structure tensor* (EST)

$$\mathrm{EST}(\mathbf{x}_i) := \overline{(\nabla z)_i (\nabla z)_i}^T \tag{6}$$

as introduced originally by Lang et al. (2007). Here, $\overline{\cdot}$ denotes the locally weighted averaging operator. This yields a tensor, representable as a 2×2 real-valued matrix, which describes how the terrain elevation changes in the local neighborhood of location $\mathbf{x}_i$. Intuitively, the local kernels Σ_i in Eq. (5) should be related to the *inverse* of $\mathrm{EST}(\mathbf{x}_i)$ to favor strong smoothing in flat areas and little smoothing across strong edges, see also Middendorf and Nagel (2002) for a discussion. Lang et al. (2007) present several iterative algorithms to adapt the Σ_i in this way. We take a more direct approach and define

$$T(\mathbf{x}_i) = \mathrm{trace}\,(\,\mathrm{EST}(\mathbf{x}_i)\,) = \mathrm{trace}\left(\overline{(\nabla z)_i (\nabla z)_i}^T\right) \tag{7}$$

and

$$\ell(\mathbf{x}_i) = \begin{cases} a \cdot T(\mathbf{x}_i)^{-1} & \text{if } a \cdot T(\mathbf{x}_i)^{-1} < \ell_{max} \\ \ell_{max} & \text{else} \end{cases} \tag{8}$$

to get a single scalar representation of the terrain's local characteristics at location $\mathbf{x}_i$. This *local* lengthscale $\ell(\mathbf{x}_i)$ is small in high variance terrain and large in flat parts. $\ell(\mathbf{x}_i)$ is bounded by ℓ_{max} to prevent lengthscales from going to infinity in the extreme case of noise-free, flat regions. The parameter a, which defines the proportional relationship between local lengthscales and the inverse ESTs, is learned in parallel to the search for the GP's hyperparameters – as described below. Finally, we set the local kernels Σ_i to the isotropic kernels $\Sigma_i = \ell^2(\mathbf{x}_i) \cdot \mathrm{I}$ (i.e., a diagonal matrix with eigenvalues $\ell(\mathbf{x}_i)$). The left diagram in Fig. 3 visualizes $\ell(\mathbf{x}_i)$ for different parameter settings.

To be able to make elevation predictions at arbitrary locations, we need to evaluate the covariance function at arbitrary locations and, thus, need to have local kernels at any point in the input space. We are only able to calculate gradients and kernels directly where we have sufficient elevation observations in the local neighborhood. Whereas Lang et al. (2007) use weighted averaging to calculate kernels in regions with few or no observations, we propose to instead put another GP prior on the local kernels' parameters. We call this a *hyper-GP*, since its predictions are used as hyperparameters for the GP that models the elevations. The notation for the hyper-GP as well as for the elevation GP is visualized in Fig. 4.

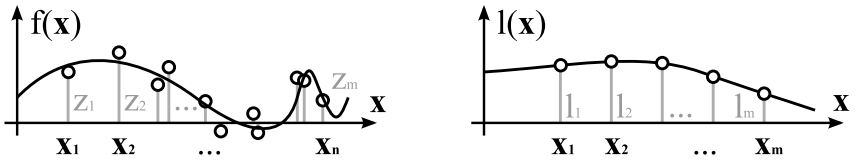


Figure 4: The notation used in this paper. The left diagram depicts a 1D elevation function, which maps locations $\mathbf{x}_i$ to elevations $f(\mathbf{x}_i)$ and the noisy observations z_i. The right diagram visualizes the corresponding (latent) lengthscale function, which links locations $\mathbf{x}_i$ to lengthscales ℓ_i.

The hyper GP represents the function $\mathbf{x} \rightarrow \ell(\mathbf{x})$ for the lengthscales of the elevation process. As the lengthscales have to be positive, we transform its training data set into log space. The hyper GP itself uses an isotropic lengthscale ℓ_h, which can be treated as an additional hyperparameter of the model. At inference time, we calculate the most likely lengthscale given by the mean prediction of the hyper GP and use the resulting kernels for elevation predictions of the elevation GP. The improvement in elevation prediction using the hyper GP (HGP) with respect to the weighted averaging (WA) approach of Lang et al. (2007) is shown in the table in Fig. 3. We give the results for three different fractions of points removed from a representative training set. As can be seen from the measured values, HGP leads to a large reduction in mean squared prediction error (76.2%, 68.4% and 44.8% for the three different data densities respectively).

Learning our model from given training data consists of two parts: (1) learning a small number of system parameters [e.g., the global noise level σ_n or the factor a in the link function $\ell(\mathbf{x})$] and (2) analytically calculating local lengthscales according to Eq. (8) and the actual GP model according to Eq. (3) and (4). In our system, the first part is implemented as an outer loop, in which the system parameters are adapted using random sampling. An alternative way would be to derive the gradient of the models data likelihood w.r.t. the system parameters and to apply gradient descent-based optimization. The general schema of such an approach is outlined in (Plagemann et al., 2008a). For this work, we chose the sampling-based variant, since it is easier to implement and it has to be executed only once, whenever drastic changes to the operational environment occur. Part two of the learning procedure involves just basic linear algebra and the inversion of the GPs covariance matrix. Since the latter can be a computationally expensive operation for large matrices,

we describe in the following how to divide a single, large model into several smaller parts—which leads to a substantial reduction in time and space complexity.

3.2 Model Tiling

Learning a Gaussian process model involves inverting the $n \times n$ covariance matrix K [see Eq. (3)], where n is the number of data points in the training set, no matter which covariance function is used. This implies a cubic asymptotic time complexity, which limits the applicability of the standard model to small and medium-sized data sets. It is therefore not surprising that much research has been dedicated to the question of how to achieve *sparse approximations* of GP models that are (a) substantially more time and space efficient and (b) retain a high prediction accuracy (see Sec. 2 for an overview of approaches).

For terrain regression, where the space is typically explored in an incremental manner, we propose to use an ensemble of overlapping GPs, where every sub-model is assigned to a specific region in the input space. Mixtures of GPs were first introduced by Tresp (2000) as a variant of the general mixture of experts model of Jacobs and Jordan (1991). This allows us to create, update and discard submodels—which we call *model tiles*—on the fly as needed. An efficient 2D indexing function is used to determine for every surface point the set of affected tiles.

The tiling approach is motivated by the insight that our kernel functions are inherently local (the standard squared exponential as well as its nonstationary extension). Thus, if the "support region" of the kernel (i.e., where it is most active) is appropriately smaller than the tiles, there is a minimal loss of model precision. This can be shown nicely using the concept of the *equivalence kernel* for Gaussian process regression (Sollich and Williams, 2004). Under this view, the GP mean predictions, Eq. (3), are expressed as dot products $\mu^* = \mathbf{h}(\mathbf{x}^*)^T\mathbf{z}$ of the vector of target values $\mathbf{z}$ and weight vectors $\mathbf{h}(\mathbf{x}^*)$. Here, the weight function $\mathbf{h}(\mathbf{x}^*)$, aka the equivalent kernel (EK), depends both on the query location $\mathbf{x}^*$ and on the covariance function k. Due to the inversion of the covariance matrix in Eq. (3), the EK is not straightforward to calculate even for the simple case of the stationary squared exponential [see Eq. (2)]. Sollich and Williams (2004) derive an approximation for this case, which shows that the EK is strongly localized and, thus, disregarding target values z that lie far away from the query point $\mathbf{x}^*$ introduces a minimal error for $\mathbf{h}(\mathbf{x}^*)^T\mathbf{z}$. This observation is indeed the basis for many Gaussian process approximations, such as that of Shen et al. (2005). Numerical experiments revealed that our nonstationary covariance function as defined in Eq. (5) is stronger or equally localized compared to the stationary squared exponential components it is composed of. Thus, we can safely apply the tiled approximation outlined above to both the latent lengthscale process as well as to the nonstationary elevation process. For ease of implementation, we use the same tiling and indexing for both processes—but this is not necessary in general.

Concretely, we assume rectangular tiles that overlap with their neighboring tiles by a fraction w of their side length. The exact placement and size of the tiles has little influence on the regression results, as long as reasonable bounds are met. These bounds are explored and evaluated in the experimental section. In principle, the size of the individual tiles could be linked analytically to the lengthscales of the covariance function at the respective locations and this is a topic of ongoing research. In this work, we treat w as a constant parameter. For a prediction at input location $\mathbf{x}$, we determine the GP segment which we consider most likely to have the best approximation for $\mathbf{x}$,

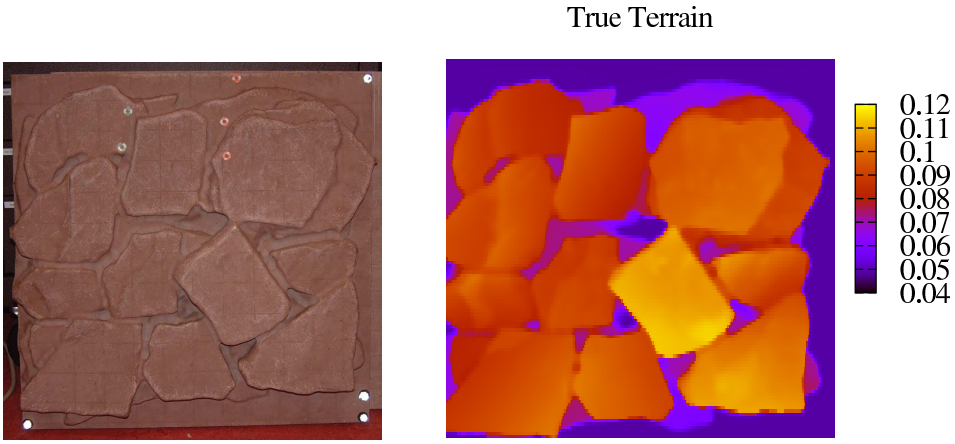


Figure 5: Left: Real terrain board used for evaluation. The side lengths of the board are approximately 60 cm. Right: The ground truth elevation structure of the terrain known from manufacturer specifications (in meters).

i.e., the segment which has a center that is the shortest Euclidean distance to $\mathbf{x}$.

The asymptotic time complexity of the tiled GP approach can be estimated as follows. Let us assume (1) that every segment contains at most a fraction c of all training data n and (2) that segments overlap by a fraction v of their inputs. Every segment then uniquely covers $cn - cnv = cn(1-v)$ of the training data, which makes it necessary to use $\frac{n}{cn(1-v)} = (c(1-v))^{-1}$ segments to cover the whole input space. Since the complexity of building a single segment GP is $O\left((cn)^3\right)$, the complexity of the whole model turns out to be $O((c(1-v))^{-1} \cdot (cn)^3) = O(c^2(1-v) \cdot n^3)$. If the segment size c is now set anti-proportionally to n (which corresponds to a constant segment size) and the segment overlap v is held constant, the overall complexity is linear in n.

In Sec. 4.3 of the following experimental evaluation, we show that the tiles can be resized to trade off time efficiency against prediction accuracy—depending on the requirements in the given application.

4 Experiments

The goal of this experimental evaluation and the following case study on legged robot locomotion in Sec. 5 is to demonstrate the usefulness of the approach for modeling real terrain data. We show that our locally adaptive Gaussian process model is more accurate than conceptually simpler approaches such as stationary GPs or elevation grid maps. We additionally analyze the benefits of our tiling approach introduced in Sec. 4.3 with respect to runtime and accuracy performance.

As the evaluation metric, we consider the mean squared prediction error (MSE), which is defined as MSE $:= m^{-1} \sum_{i=1}^{m} (f(\mathbf{x_i}) - \mu_i)^2$, where m is the number of test points, μ_i denotes the predicted mean elevation at location $\mathbf{x}_i$ and $f(\mathbf{x}_i)$ is the respective true elevation. The following experiments were conducted in a batch setting, that is, full data sets were recorded and then randomly split into training and test sets, respectively. As discussed in Sec. 4.5, our algorithm is not restricted to the batch setting in principle.

In the first experiment, in which we compare to the standard Gaussian process model, we additionally consider the negative log predictive density (NLPD). This measure is defined as NLPD $:= m^{-1} \sum_{i=1}^{m} \log p_{\text{model}}(f(\mathbf{x}_i)|\mathbf{x}_i)$, where $p_{\text{model}}(\cdot|\mathbf{x}_i)$ stands for the predictive distribution at location $\mathbf{x}_i$ and $f(\mathbf{x}_i)$ is the true elevation value. In contrast to the MSE, the NLPD also takes the spread of the predictive distribution into account and, thus, arguably is a better measure for comparing predictive distributions. However, since the MSE is more widely used in the literature, since it is easier to interpret and since the NLPD is not available for simpler benchmarks such as bi-linearily interpolated grids, we use the MSE as the primary measure in the remainder of the section.

All experiments were conducted using a C/C++ implementation on a Linux desktop PC with a single 2 GHz CPU. The linear algebra subroutines were optimized using the UMFPACK package (Davis, 2004).

4.1 Adapting to Local Terrain Structure

In the first experiment, we evaluated the performance of the GP terrain regression model using the standard squared exponential (SE) covariance function against our nonstationary covariance function with local lengthscale adaption. To this aim, we considered the rocky terrain depicted in the left image in Fig. 5 and simulated 2 500 laser observations from a single viewpoint using the known elevation values depicted in the right diagram in the same figure. We uniformly selected 4 350 points from the true terrain for evaluation. We then conducted Monte Carlo search in the parameter space of the covariance functions and on the parameters of the adaption procedure. In a preliminary run over 34 000 configurations, we determined general ranges for the parameters and in a secondary search, we evaluated 10 000 configurations in the predetermined ranges. A scatter plot of the results in terms of MSE and NLPD is given in the top left diagram in Fig. 6.

The (red) boxes in the diagram depict the error values for randomly sampled parameter vectors in the standard GP model and the (cyan) crosses show the results for our nonstationary extension, respectively. The goal is to minimize the MSE (error of the mean predictions) and also the NLPD (which considers the entire predictive distribution). It can be seen from the diagram that the nonstationary covariance function is able to achieve better MSEs of predictions w.r.t. the ground truth and also better NLPDs. The improvements in terms of NLPD is notable in particular—for the normal distribution, an NLPD difference of 0.5 corresponds to an absolute error of up to one standard deviation, which is a substantial mismatch.

To give a visual impression of these results in terms of regression functions, we plot the predictive distributions for three different settings in Fig. 6. The top right panel shows a cut through the regression surface obtained using our locally adaptive GP model. The x-axis points to the right, elevations are plotted upwards and the y-coordinate is fixed. On the secondary axis (scale on the right-hand side of the diagram), we give the adapted local lengthscales and the respective tensor traces. As can be seen from the plot, the GP fits the observed data well and produces a large predictive uncertainty in un-observed areas. This result corresponds to the optimal parameter vector marked by the large black cross in the top left diagram in Fig. 6.

The two lower panels in Fig. 6 show the same cut for the standard Gaussian process model. The

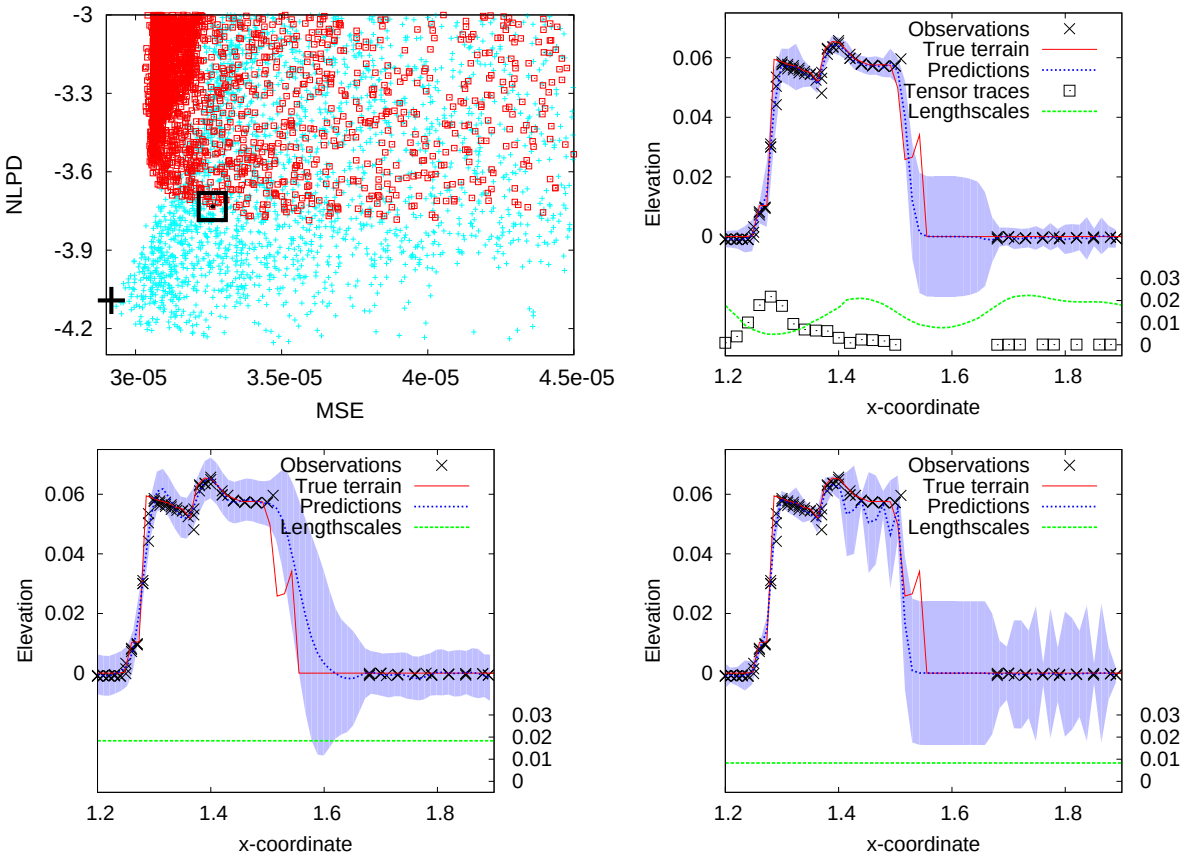
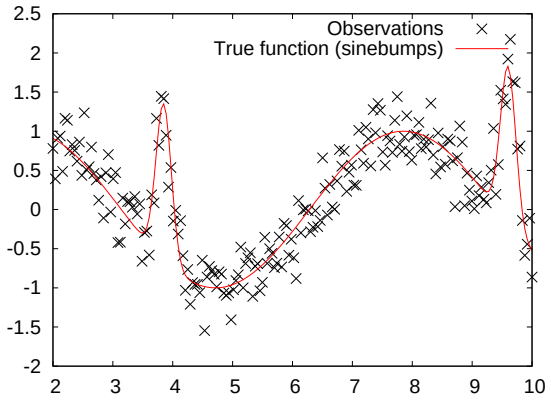


Figure 6: Top left: The Monte Carlo search for the stationary GP's parameters (red boxes) and for the nonstationary GP's (bright crosses). The selected optima, which minimize the MSE as well as the NLPD error, are marked by a large black box and by a large black cross, respectively. Top right: Regression model learned using our locally adaptive GP approach. Here, we visualize a vertical cut through the terrain surface (y is fixed, x varies on the horizontal axis in meters, terrain elevation on the vertical axis, in meters). We also give the adapted lengthscales below the curve using a second vertical axis. Bottom: Regression results obtained using the standard, stationary GP model for two different settings of the constant lengthscale parameter (left: large lengthscale, right: small lengthscale).

left panel shows the results for a comparably large, constant lengthscale parameter (the respective errors are marked in the upper left diagram by a large black box), while a smaller lengthscale parameter is used for the right-hand panel. It can be seen from the two plots that none of these settings yields the tight data fit of the locally adaptive model in the upper right panel. This is due to the fact that a GP with a stationary covariance function cannot fit all parts of a function (e.g., smooth, wiggly and discontinuous ones) well.

4.2 Alternative Adaptive Models on Benchmark Data

In Sec. 3.1, we proposed to adapt the local lengthscales depending on the local terrain gradients. We evaluated in a second experiment, how this compares to other regression approaches which also adapt their scales locally. It should be noted that there is no "correct" lengthscale, which can be compared against as the ground truth. Rather, the lengthscales are latent variables (or parameters, depending on the viewpoint) in our model. Other models that have similar *scale* variables may set these to different values to achieve optimal performance. Thus, we evaluate the *prediction errors*



	Time	MSE
LA-GP A	**2.64 (0.069)**	0.0060 (0.0018)
LWPR	11.54 (0.35)	0.0065 (0.0018)
LA-GP B	7.24 (0.66)	**0.0043 (0.0014)**

Figure 7: Left: Sinebumps benchmark data set to evaluate nonstationary regression approaches (Dimatteo et al., 2001). Right: The regression results for our locally adaptive, tiled GP approach (*LA-GP A*), locally weighted projection regression (*LWPR*) and locally adaptive GPs (*LA-GP B*). We give the average time requirements in seconds and the mean squared prediction errors (standard deviations are given in brackets).

of the different approaches on an independent test set to assess how well the data is modeled.

As the test data set, we use the *sinebumps* scenario depicted in the left diagram in Fig. 7. This is a benchmark data set frequently used to evaluate nonstationary regression approaches (Dimatteo et al., 2001; Paciorek and Schervish, 2004). It consists of a sine function $y = \sin(x)$ and several "bumps" generated by an additive term $2 \cdot \exp(-30 \cdot (x - c)^2))$. For each of 30 independent runs, we generated 600 training points and 200 independent test points per input space unit.

The different models compared were our terrain-gradient based, tiled adaptation scheme (denoted as *LA-GP A*), locally weighted projection regression, *LWPR* (Vijayakumar et al., 2005) and a different, non-tiled, locally adaptive GP model, *LA-GP B* (Plagemann et al., 2008a). The regression results in terms of the average time requirements for learning and prediction (in seconds) and mean squared prediction errors are given in the table in Fig. 7; standard deviations are given in brackets.

For *LWPR*, we optimized the parameters as suggested in Klanke and Vijayakumar (2008). Concretely, we searched for initial valued for the distance metric D by increasing the parameter in 10% steps from 1 000 to 100 000 and found the optimum at approximately 17 500. Afterwards, we searched for a learning rate parameter α in 10% steps between 0.01 and 100 and found the optimum at approximately 0.66. *LA-GP B* uses the same nonstationary covariance function as proposed in this paper. Instead of linking the local lengthscales to the function gradient, it treats them as free parameters, which are optimized w.r.t. the data-likelihood of the training set. As expected, this approach can fit the data better—but it is also more demanding computationally and has not been made scaleable to larger data sets yet (e.g., using sparse approximations or model tiling as proposed in this paper).

4.3 Splitting the Terrain into Overlapping Sub-Models

We evaluated the benefits of segmenting the input space to a set of overlapping tiles. To this aim, we applied different tile sizes to the terrain model analyzed in the first experiment (see Sec. 4.1). We measured the prediction accuracy in the innermost $0.0025m^2$ of a tile while linearly increasing the

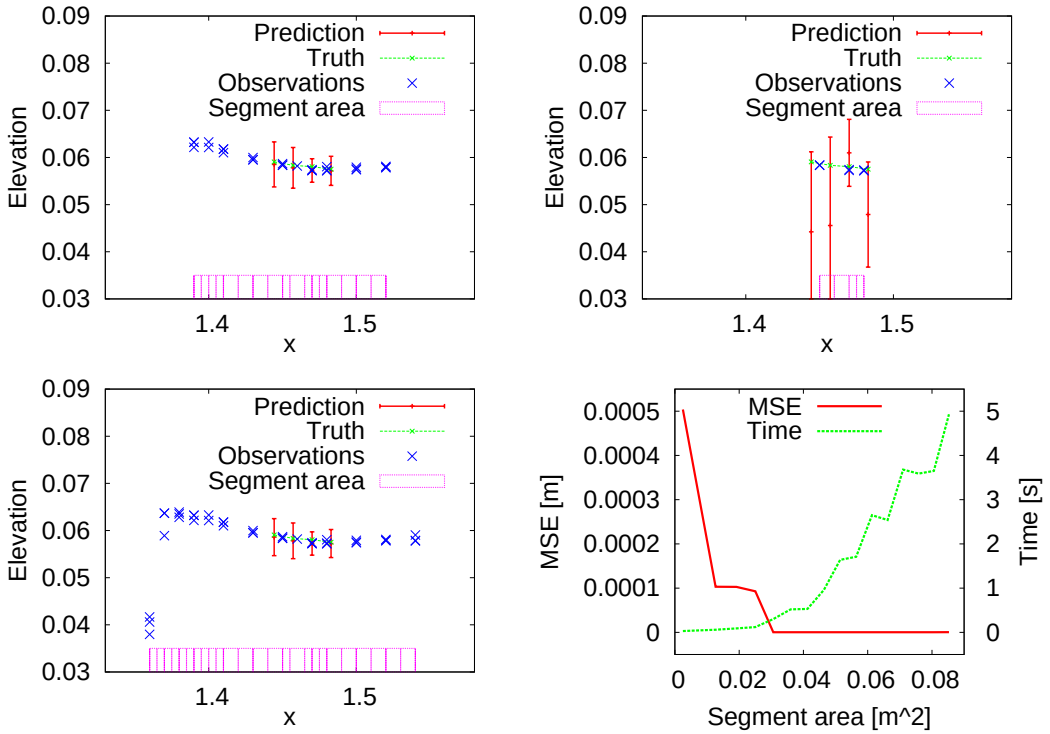


Figure 8: Top left: A cut through the 2D regression function similar to Fig. 6 for the case of a well chosen segment size ($0.03m^2$). Top right: Too small segments (here: $0.0025m^2$) lead to large prediction errors. Bottom left: Overly large segments (here: $0.056m^2$) do not improve the predictions, but make the approach less efficient. Bottom right: the relationship between the model errors (MSE in meters) and the training times (in seconds) for different segment sizes.

tile area and thereby also the amount of training data of the associated GP. The first three diagrams in Fig. 8 give a visual impression of the effects that different tile sizes have on the regression function in its center. Increasing the tile size from $0.03m^2$ (top left) to $0.056m^2$ (bottom left) does not lead to a notable improvement. On the other hand, setting the segment size too small (e.g., $0.0025m^2$ in the top right diagram) leads to an increased prediction error. See also the caption of the figure.

The quantitative relationship of the prediction error and time requirements for training depending on the size of the tiles is given in the lower right diagram in Fig. 8. It can be seen that with an increasing tile size, the prediction error (MSE) decreases quickly and almost converges as the area reaches $0.03m^2$. The runtime, however, continues to grow cubically with the segment size beyond this point. It is relatively straight-forward to construct a joint cost function that includes the prediction error and the time requirements and which weights both quantities depending on the requirements of the application.

The runtime requirements for learning from approximately 100 000 points using our C++ implementation are on the order of 1.5 seconds with overlapping stationary GPs and 3 seconds with overlapping nonstationary GPs. It is not possible to process data sets of this size with regular GPs that do not utilize sparse approximations.

Figure 9: Our quadruped robot scans the surface of a terrain board by bending its legs to sweep the scan line of a back-mounted laser range finder over the terrain surface.

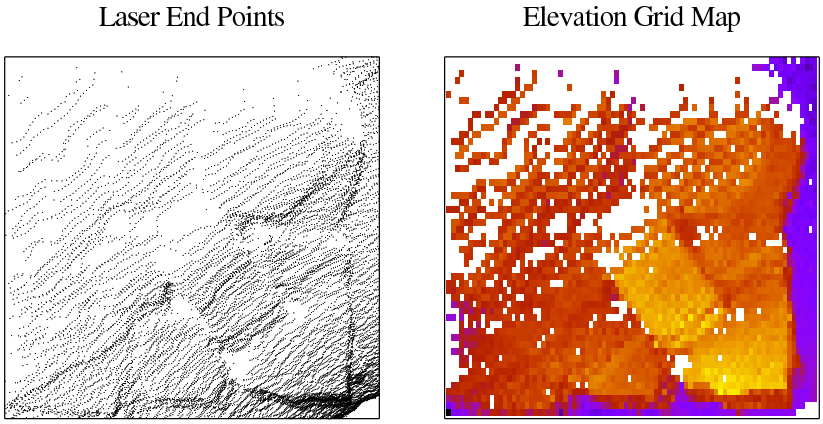


Figure 10: Left: Top view on the set of elevation measurements obtained by the real robot (the true terrain is depicted in Fig. 5). Right: A grid-map approximation of this data set. The elevation values are color-coded ranging from purple/dark (0cm) to yellow/bright (approx. 7cm). .

4.4 Mapping Accuracy on Real Terrain

We evaluated our terrain model with a real quadruped robot in a situation similar to the one depicted in Fig. 9. The robot, called *LittleDog*, was developed by Boston Dynamics. We have equipped the robot with a Hokuyo URG laser scanner. A high-resolution motion capture system, the Vicon MX, yields estimates of the robot pose using measurements from reflective markers attached to the robot's body. The laser sensor is mounted to the back of the robot in a 25° angle facing towards the ground so that 3D range scans can be recorded by executing a tilt motion using the front and rear legs. The evaluation in this section concentrates on the question of how accurately the elevation structure of the terrain board can be recovered from a single such 3D scan.

We evaluated our locally adaptive GP approach using scans of a rocky terrain surface acquired by the quadruped robot against a known ground-truth model of the terrain acquired using a high-accuracy metrology system. Figure 5 depicts the true elevation structure of this terrain (see the caption for details). The left diagram in Fig. 10 shows a top view on the raw set of laser endpoints that were acquired by the robot when it executed a tilt motion. It can be seen from the uneven distribution of points that parts of the terrain are not sampled densely due to occlusions and a larger distance to the sensor location (which was located towards the bottom right w.r.t. the diagram).

The state-of-the-art way of representing such data for further processing, such as path planning, is to build an elevation grid map (see Sec. 2 for a discussion). A probabilistic elevation grid map is

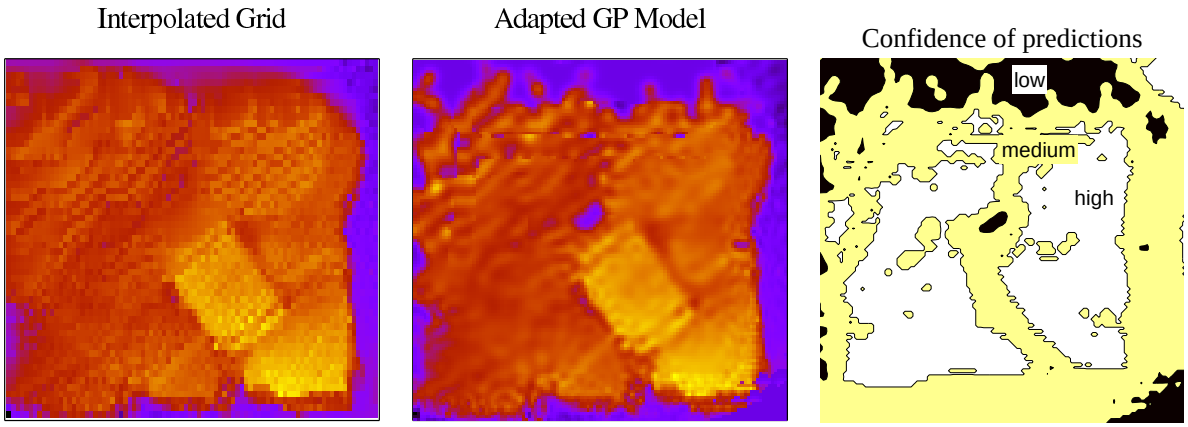


Figure 11: Mapping results using the bi-linearily filled elevation grid model (left) and the mean predictions of our adapted GP model (center). The predictive uncertainties of our model are visualized in the right diagram (discretized to three classes for better readability). These uncertainties correspond to the errorbars in the slice-view visualization in Fig. 6.

built by discretizing the x-y space and by fitting 1D Gaussians to the elevation samples falling into the grid cells respectively. The result of this operation is depicted in the right diagram in Fig. 10. We compare the accuracy of this model as well as its bi-linearly filled completion to our adapted, tiled Gaussian process model. Alternative, non-sparse regression models were not able to deal with the amount of data in this test (approx. 100 000 points) and the standard, stationary Gaussian process model was not competitive due to the problems discussed in Sec. 4.1.

Bi-linear interpolation is the extension of linear interpolation to bi-variate functions. The result of such an operation applied to an incomplete elevation grid map is depicted in the left diagram in Fig. 11. The results obtained by our locally adapted GP approach are depicted in the middle diagram. Here, we plot the mean predictions for terrain elevations. The predictive uncertainties of our model are visualized in the right-most diagram in this figure. For better visibility, we discretized the uncertainties to three classes: high, medium and low predictive uncertainty. These estimates can be utilized in the cost function of a path planner as described in Sec. 5.3 to avoid uncertain areas.

We quantitatively compared the prediction errors of our locally adapted GP model to the baseline elevation grid model models and to a bi-linearily interpolated, dense grid. In Fig. 12, we give the squared error of elevation predictions averaged over 10 000 samples drawn randomly from the terrain. The error-bars give the standard deviations of the individual sample sets. To assess the influence of the grid resolution for the two grid-based models, we tested six different numbers of cells per grid dimension (x-axis). Since the standard elevation grid does not make predictions in occluded or less densely sampled areas, its performance was evaluated on its occupied cells only.

It can be seen from the diagram that our locally adapted GP model predicts the true terrain elevation as accurately as the elevation grid at its optimal resolution. More importantly, the GP model achieves this accuracy averaged over all test points, while the grid model is evaluated on occupied cells only.

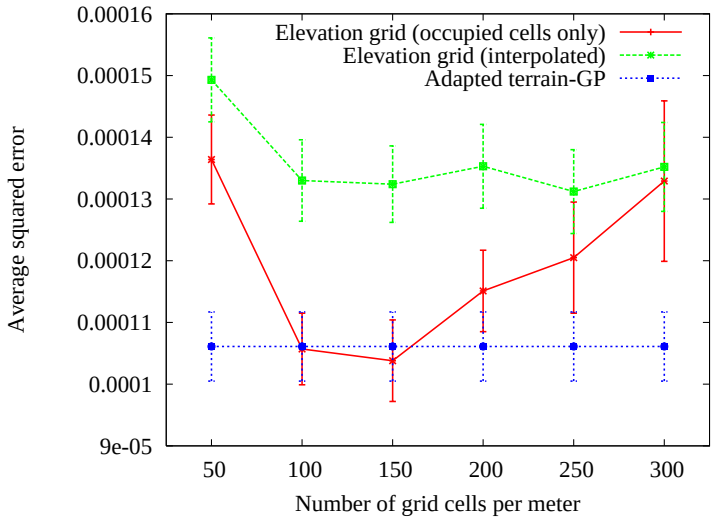


Figure 12: Errors of elevation predictions for our adapted, tiled GP model, the sparse elevation grid and its bi-linear completion. The elevation grid (without interpolation) was evaluated on occupied cells only. At its optimal resolution it predicts the true elevation of occupied cells only as accurately as the locally adapted GP model does averaged over all test points.

4.5 Discussion

In Sec. 5, we will describe our integrated system for a quadruped robot that scans the terrain, plans a path to a goal location and executes the plan autonomously. Before that, we discuss the basic properties of the locally adapted, tiled GP model as they became apparent from this part of the evaluation.

As a first remark, no algorithm can predict the elevation at unseen locations perfectly correct and, thus, a prediction error of zero is not attainable in general. Rather, the goal is to minimize the prediction error as much as possible in different situation: (1) in densely sampled ares, (2) in areas containing some measurements and (3) in unobserved areas. While all reasonable mapping approaches output a high model uncertainty in unobserved areas (including probabilistic elevation grids, Gaussian processes, etc.), the behavior in areas with a high or a medium sampling density is quite different. Stationary Gaussian processes, on the one hand, cannot represent natural terrain with its locally changing smoothness well, even if it is sampled densely (see, e.g., the lower plots in Fig. 6). Elevation grids, on the other hand, cannot deal with medium or low sampling densities well, because they either contain a large number of empty cells at a high grid resolution, or they introduce a strong bias by averaging over too large areas at a low grid resolution.

Our locally adaptive, tiled GP model can be seen as combining the benefits of elevation grids and Gaussian processes. The nonstationary covariance function allows the model to adapt well to densely sampled parts of the elevation function, while the tiled approximation avoids the cubic runtime complexity of the standard Gaussian process model. Compared to grid maps, the GP-based approaches provide a sound way of extrapolating the observed parts of the elevation function to unobserved locations and of estimating the respective predictive uncertainty. To achieve maximal robustness in practice, a system would use the uncertain predictions only to guide the search for potential paths towards promising areas. The critical decisions, however, such as actual foot place-

ments, would be based on the most recent and trustworthy sensor measurements. In our approach, this can be achieved by continuously updating the model as data arrives. There exists a large body of work on the topic of online updates in Gaussian process models [see, e.g., Csato and Opper (2002)]. Such an extension is applicable in principle, but has not been implemented in the current system. As such, our current implementation supports online adaptation only on the tiling level: new tiles are created on-the-fly as needed, but existing, already instantiated and learned tiles are not updated with newly arriving data.

A certain limitation of the current system is that it assumes that the true terrain elevation is a well-defined *function* of 2D locations. This means that overhanging structures, which would violate this assumption, cannot be dealt with directly. The two most promising directions for dealing with such situation are (1) to take a hybrid approach in which the environment is represented by a discrete set of 2D terrain models, each of which is placed at an arbitrary location in 3D space, or (2) to consider mixtures of Gaussian process models along the lines of, e.g., Tresp (2000) or Rasmussen and Ghahramani (2002).

Another possible extension to the current system would be to adapt the size of the tiles to the local situations, e.g., depending on the local data density or on the adapted local lengthscales. In this work, we chose a conservative global partitioning, which can be implemented more easily.

5 Case Study: Legged Robot Locomotion

In this section, we describe how the proposed terrain model was integrated into our quadruped robot to enable it to scan a terrain surface, select footholds and plan a trajectory to a goal location that is collision-free and statically stable.

The advantages of using legged robots over traditional wheeled robots are the ability to move in rough and unstructured terrain and to step over obstacles. Without accurate knowledge of the terrain, these advantages cannot be realized as motion planners require a model of terrain height for computing stability and avoiding collisions. Our overall planning approach is an adaptation of the probabilistic roadmap algorithm (Kavraki et al., 1996). Our simplified model of the quadruped robot is a body with four two-link legs and point feet. The planning algorithm is a search for motions of single legs from static stance to static stance that maintain static stability over uneven terrain. We first randomly sample a set of potential footholds across the terrain, which are used to generate a graph of potential "stances", that is, statically stable and kinematically feasible positions of the robot. Graph search is then used to find a sequence of stances from the start to the goal; the sequence of stances can then be converted to series of planned joint angles.

5.1 Calibration

One of the critical issues in most robotic systems and in our setup particularly, is the calibration of the individual sensors. Already small errors in the transformation between the laser sensor and the robot, for instance, can lead to large inconsistencies in the collected data set. Note that this paper does not address the problem of simultaneous localization and mapping. Rather, we deal with the terrain inference problem here and assume reasonably accurate knowledge of the robot's pose.

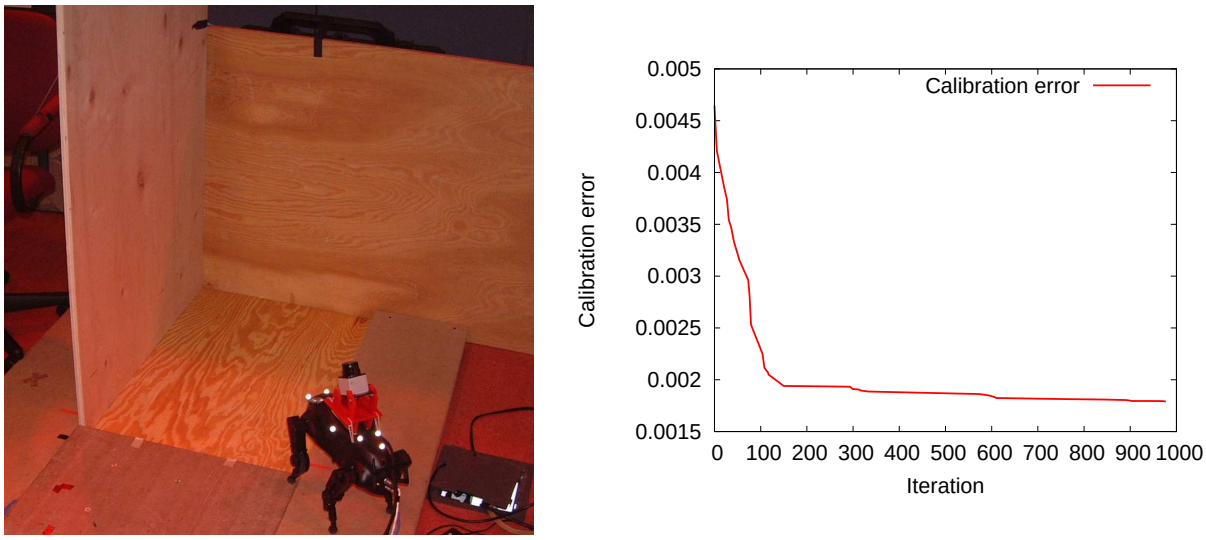


Figure 13: Left: The physical setup used to calibrate the robot. Right: Convergence behavior of the calibration error.

The specifics of the Vicon system used to track the robot's position required us to automatically infer the time offsets between the laser data and pose measurements and to calculate the six degree of freedom transformation between a known position of the robot and the actual laser sensor position at the same time. To compute both these calibration quantities, we recorded a 3D range scan of three orthogonal boards placed in front of the robot (side lengths approximately 1m). The physical setup is depicted in the left image in Fig. 13.

We optimized the seven parameters of the transformation (6D laser configuration w.r.t the robot and a 1D time offset) in a sampling-based fashion similar to simulated annealing (Kirkpatrick et al., 1983). Here, random parameter samples are evaluated using a predefined objective function. In our setting, this is the average squared error of the laser end points relative to the known calibration pattern, that is, the board surfaces. To be more robust against wrong data associations caused by strong miscalibration in the beginning of the optimization procedure, we discard laser end points that fall close to the boundaries of the boards. In each iteration, we sample a new parameter vector from a Gaussian distribution centered at the current optimum. The variance of this Gaussian (which is comparable to the *temperature* variable in simulated annealing) defines from which area new parameter samples should be drawn around the current optimum. This variance is reduced by a fixed delta after each iteration. By decreasing the temperature level gradually over time, accurate calibration parameters are typically obtained within 300-800 iterations. The convergence of the calibration error during a typical run is visualized in the right diagram in Fig. 13.

5.2 Sampling Footholds

Let us assume that the planning problem is to find a motion plan that is essentially a futtock (midline) motion across the uneven terrain from the start position to the goal. This assumption will allow us to simplify the sampling to examining potential footholds around the straight line to the goal, selecting footholds $\phi = (x, y, z)$ according to some regular discretization around the line of intended motion. We do this without loss of generality; we can easily support more complex

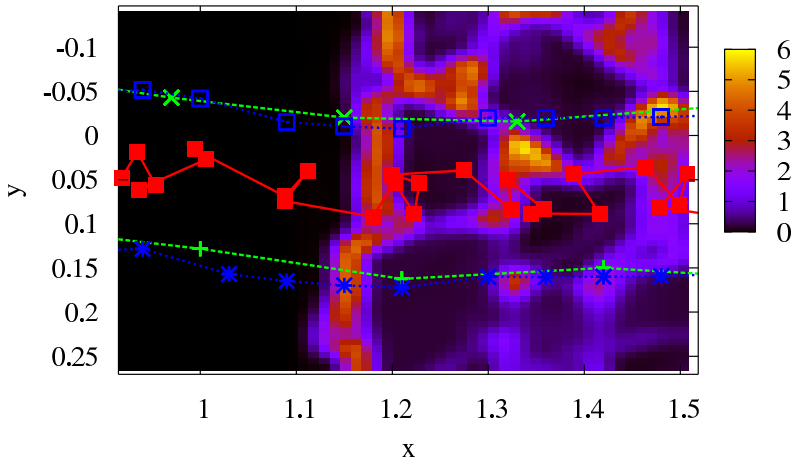


Figure 14: Parts of a plan generated by our algorithm including the underlying cost function, which depends on the terrain gradient and the uncertainty about elevation predictions estimated by the Gaussian process model. The red line (filled boxes) depicts the trajectory of the center of body, the other lines (stars and empty boxes) visualize feet motions. The cost function is color coded ranging from black (low costs) to yellow/light-gray (high costs). Axis dimensions are given in meters.

scenarios by choosing different sampling strategies. The sampling problem is outside the scope of this paper but has been discussed by Hsu et al. (2005) and others.

Each sampled foothold is evaluated with respect to a cost function and rejected if the expected cost is above some threshold. The cost function may consist of many features including terrain elevation and roughness. In this work we considered only the uncertainty in the terrain model (i.e., the GP predictive uncertainty) at the sampled foothold ϕ and the terrain gradient (i.e., slope). The terrain slope can be calculated analytically from the learned GP, which is an additional advantage of the Gaussian process approach. An example of a cost function including a set of selected footholds and a path is given in Fig. 14.

5.3 Planning with Stance Graphs

We next generate feasible stances of the robot from the discrete set of footholds. A stance is an assignment of each foot i to a foothold, $\phi_i = (x_i, y_i, z_i)$, such that it is kinematically feasible for the robot to place its feet at each of the four footholds and remain statically stable. Note that determining whether a stance is feasible or not is not directly computable from a set of foot positions because the feet do not provide a unique description of the pose of the robot. The robot has 18 degrees of freedom total: six degrees of freedom of the center of the body $(x, y, z, roll, pitch, yaw)$ and the three joints $(hip_x, hip_y, knee)$ in each leg. Under the assumption that the positions of the feet are fixed, the feet constitute 12 constraints, leaving six unconstrained degrees of freedom, corresponding to the position of the center of body. A stance s_i is therefore an assignment of feet to footholds $\phi_{1..4}$ and a selection of a center of body position ξ.

Given an assignment of the center of the body position for a set of foot positions, the known kinematics can be used to recover the joint angles of the legs and determine if the pose is consistent with the dimensions of the leg links and the limits on the joint angles. Given knowledge of the joint angles and that the stance is kinematically feasible, the center of mass can then be determined; if

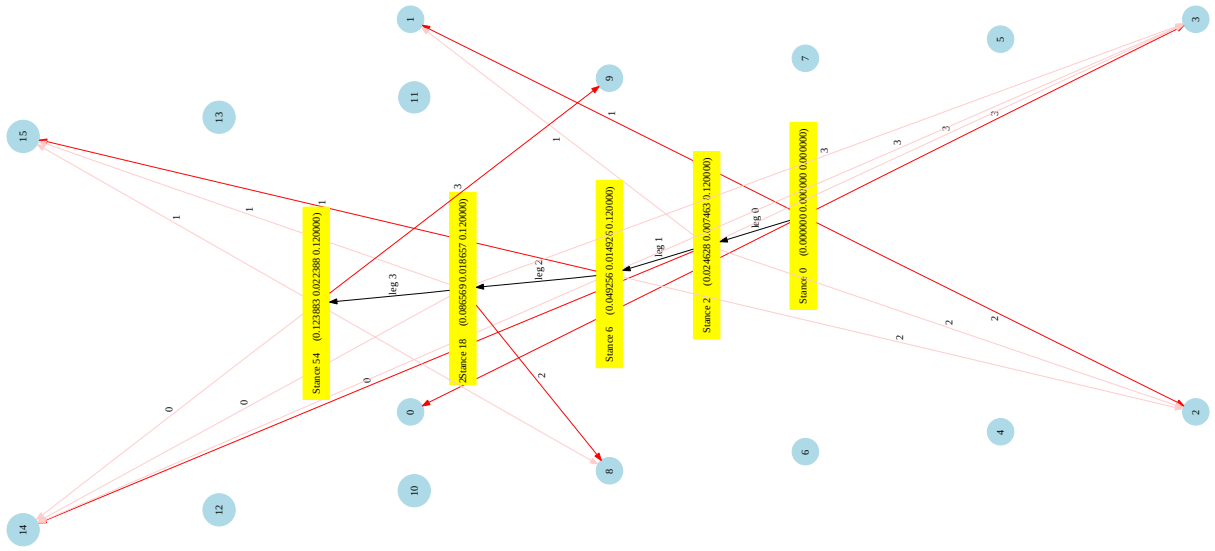


Figure 15: An example stance graph generated from the set of foothold locations. Each stance node (yellow) is connected to four foothold nodes (blue). Each two connected stance nodes have three stance legs in common.

the projection of the center of mass onto the ground plane lies outside the support polygon (the convex hull of the four feet on the ground plane), then the stance is not statically stable and the robot will fall.

In assigning the position of the center of body for a given set of foot positions, we would ideally choose a center of body that provides static stability. Unfortunately, no closed form solution exists for finding a feasible and stable center of body and the problem is in general non-convex. We therefore use a heuristic search strategy around the centroid of the support polygon. If none of the sampled centers of body provide a kinematically feasible and stable solution to the robot position given the foot positions, then the foot positions are rejected as an infeasible stance.

The feasible stances constitute nodes in a stance graph, to which we then add edges between pairs of stances s_i and s_j when a feasible motion exists to transform the robot from the start stance s_i (foot positions and center of body) to the end stance s_j (see Fig. 15). This problem is also underdetermined, in that an arbitrarily complex motion may be required to move from one stance to another. We therefore simplify this problem to consider motions consisting of a stance phase, during which the robot shifts its center of body to remain stable while stepping and a foot swing phase during which a foot is moved from one foothold to another.

Once the stance graph has been built, we use standard breadth-first search to find the shortest feasible sequence of stances from the start stance to a goal stance that gives a center of body position with some ϵ of the desired goal position, in practice, combining the search process with the stance graph generation. Additionally, we add a gait-order constraint, so that the plan must consist of a well-formed gait in which foot i is followed by foot $i + 1 \mod 4$. By augmenting each stance variable with an additional foot-ordering variable ψ, this gait-ordering constraint dramatically improved the planning speed. Finally, we also use a hash table to prune the search, such that if two different routes are found to the same stance node s, then the search along the longer path is terminated. The full planning algorithm is given in Algorithm 1.

Algorithm 1 The Planning Process.

Require: Terrain model, start stance s_0 and goal $\mathbf{x}_g$.
  Sample footholds Φ using terrain model
  Initialize $Q \leftarrow s_0$
  while Q is not empty **do**
    $s \leftarrow$ pop Q
    for all ϕ **do**
      $s' \leftarrow s$
      Update position of foot to move, $\phi(\psi(s')) \leftarrow \phi$ in s'
      Update foot to move, $\psi(s') \leftarrow \psi(s') + 1 \mod 4$
      Search for new center of body position $\xi(s')$
      if $||\xi(s') - \mathbf{x}_g|| < \epsilon$ **then**
        return Parents $[s']$.
      end if
      if $\xi(s')$ exists **then**
        Set parent, $\pi[s'] = s$
        Push $Q \leftarrow s'$
      end if
    end for
  end while
  return nil

5.4 Terrain Mapping and Path Planning

In this section, we describe an evaluation of our probabilistic terrain model in conjunction with the described trajectory planning algorithm. The first experimental setup was to sample 1 000 random starting locations in front of the terrain board depicted in Fig. 5 and to pick corresponding goal locations behind it. The resulting paths were approximately two meters long, starting in a flat area and leading over the terrain board to another flat area behind it. Concretely, the lateral offsets (y-coordinate) of the start and end locations w.r.t. a straight trajectory over the terrain board were sampled from a uniform distribution over [-10 cm, 10cm].

For each of the location pairs and each of three alternative terrain mapping algorithms (our locally adaptive GP approach, the elevation grid and the interpolated elevation grid), the planner generated a set of footholds and searched for the best path towards the goal location. We then evaluated (1) the maximal path length that could be constructed given the kinematic constraints of the robot and (2) the errors of the elevation predictions at the selected foothold locations. An example plan and the cost function computed from the underlying terrain model are depicted in Fig. 14

Fig. 16 summarizes our results. The left bar plot shows the maximal length of generated plans and the right plot gives the mean squared errors (scaled by 10^{-3}) of elevation predictions at the planned footholds. It can be seen from the left diagram that it was always possible to plan the maximal path of 2 meters using the interpolated grid and the locally adapted GP model. Using the sparse grid, however, the plans never exceeded a length of 1.6 meters, which is not surprising given the large number of unknown cells which prohibit foot placements. As can be seen from the the mean squared error values in the right diagram, the locally adapted GP model better predicts the true terrain elevations at the chosen foothold locations than the interpolated grid model, which means that there is a lower risk of failure when executing these plans. Finally, Fig. 17 shows snapshots from a video documenting our real robot traversing the terrain board using the Gaussian process model learned from own elevation measurements. Figure 18 gives the corresponding trajectory taken by the robot during an autonomous walk over the terrain board in this experiment.

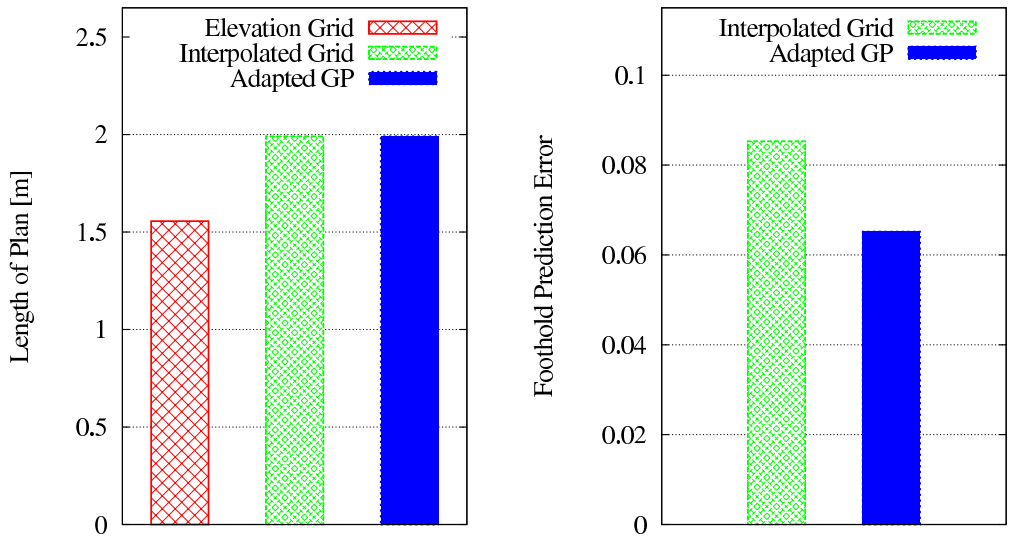


Figure 16: Evaluation of 1 000 plans generated using the different terrain models. The left bar plot shows the maximal length of generated plans and the right plot gives the mean squared errors (scaled by 10^{-3}) of elevation predictions at the planned footholds.

6 Conclusions and Outlook

In this paper, we presented a novel, probabilistic terrain mapping approach based on nonstationary, tiled Gaussian processes. Our system balances smoothing against the preservation of structural features and it is capable of accurately predicting elevations in the presence of noise and it also estimates the uncertainty of its predictions. As an application scenario, we considered the terrain mapping problem for a legged robot equipped with a laser range finder. We document the key parts of an implemented system—including physical setup, calibration, foothold selection and trajectory planning—that was able to traverse a rocky terrain using own range measurements only.

In the future, we plan to apply alternative ways of learning the nonstationary GP model (Plagemann et al., 2008a) and to evaluate the difference in modeling accuracy. We would also like to address the more general SLAM problem, in which the robot does not assume knowledge of its own position. Future work could also consider a reinforcement learning variant of our foot trajectory planning along the lines of Neumann et al. (2007) using specific reward functions to learn obstacle avoidance or stable and energy-efficient movements. Furthermore, we would like to extend our model towards online model updates beyond the existing capabilities of adding and exchanging entire model segments and we intend to evaluate whether variable segment size would lead to additional benefits.

Acknowledgments

We would like to thank Tobias Lang and Russ Tedrake, as well as the DARPA Learning Locomotion project (AFRL contract #FA8650-05-C-7262). This work has been supported partly by the EC under contract numbers FP6-004250-CoSy, by the DFG under contract number SFB/TR-8 and by the German Ministry for Education and Research (BMBF) through the DESIRE project.

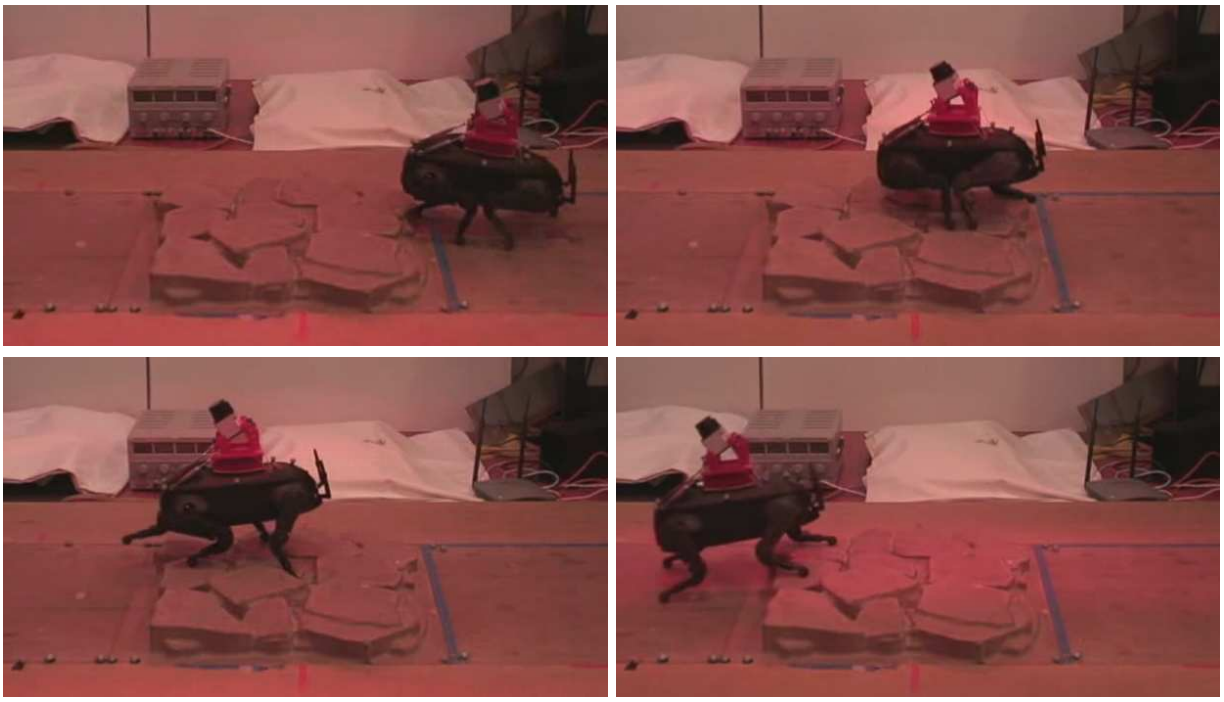


Figure 17: Snapshots of a video documenting the walk over the terrain using the learned terrain model from own elevation observations (from top left to bottom right).

References

Barber, C. B., Dobkin, D. P., and Huhdanpaa, H. (1996). The quickhull algorithm for convex hulls. *ACM Trans. Math. Softw.*, 22(4):469–483.

Bares, J., Hebert, M., Kanade, T., Krotkov, E., Mitchell, T., Simmons, R., and Whittaker, W. (1989). Ambler—an autonomous rover for planetary exploration. *IEEE Computer*, 22(6):18–26.

Bors, A., Hancock, E., and Wilson, R. (2003). Terrain analysis using radar shape-from-shading. *Terrain*, 25(8):974–992.

Brooks, A., Makarenko, A., and Upcroft, B. (2006). Gaussian process models for sensor-centric robot localisation. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*.

Brooks, C. and Iagnemma, K. (2009). Visual detection of novel terrain via two-class classification. In *Proceedings of the 24th Annual ACM Symposium on Applied Computing*, Honolulu, Hawaii.

Csato, L. and Opper, M. (2002). Sparse on-line gaussian processes. *Neural Computation*, 14(3):641–668.

Daum, M. and Dudek, G. (1998). On 3-d surface reconstruction using shape from shadows. In *IEEE Conference on Computer Vision and Pattern Recognition*.

Davis, T. A. (2004). A column pre-ordering strategy for the unsymmetric-pattern multifrontal method. *ACM Trans. Math. Softw.*, 30(2):165–195.

Delone, B. N. and Novikov, S. P. (1993). *Discrete geometry and topology: on the 100th anniversary of the birth of Boris Nikolaevich Delone: collection of papers*. American Mathematical Society.

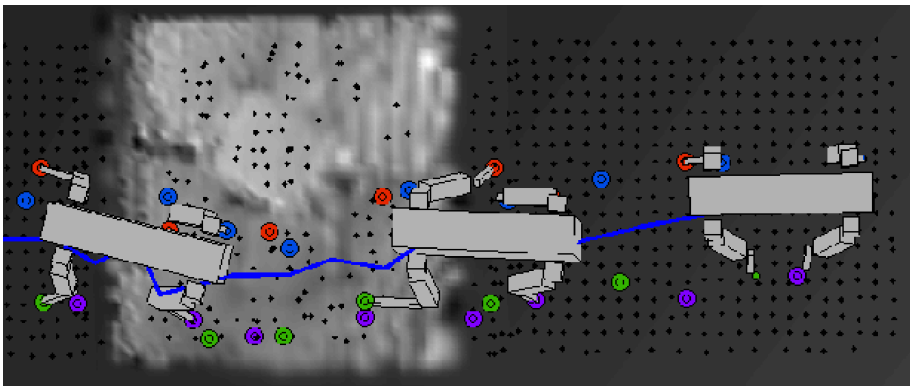


Figure 18: Path taken by our real robot during an autonomous walk over the terrain board (from right to left).

Dimatteo, I., Genovese, C., and Kass, R. (2001). Bayesian curve-fitting with free-knot splines. *Biometrika*, 88(4):1055–1071.

Foessel, A., Bares, J., and W.R.L., W. (2001). Three-dimensional map building with mmw radar. In A. Halme, R. C. and Prassler, E., editors, *Proceedings of the 3rd International Conference on Field and Service Robotics*, Helsinki, Finland.

Früh, C., Jain, S., and Zakhor, A. (2005). Data processing algorithms for generating textured 3d building facade meshes from laser scans and camera images. *International Journal of Computer Vision*, 61(2):159–184.

Hadsell, R., Bagnell, D., and Herbert, M. (2009). Accurate rough terrain estimation with space-carving kernels. In *Proc. of the Robotics: Science and Systems Conference (RSS)*. to appear.

Hauser, K., Bretl, T., Latombe, J., Harada, K., and Wilcox, B. (2008). Motion planning for legged robots on varied terrain. *The International Journal of Robotics Research*.

Hsu, D., Sánchez-Ante, G., and Sun, Z. (2005). Hybrid PRM sampling with a cost-sensitive adaptive strategy. In *Proc. IEEE Int. Conf. on Robotics & Automation*.

Huber, D. and Hebert, M. (1999). A new approach to 3-D terrain mapping. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Hufentobler, M. (2004). *Terrain Modelling with Triangle Based Free-Form Surfaces*. PhD thesis, University of Zurich.

Hygounenc, E., Jung, I., Souères, P., and Lacroix, S. (2004). The autonomous blimp project of laas-cnrs: Achievements in flight control and terrain mapping. *I. J. Robotic Res.*, 23(4-5):473–511.

Iagnemma, K., Kang, S., Shibly, H., and Dubowsky, S. (2004). On-line terrain parameter estimation for planetary rovers. *IEEE Transactions on Robotics and Automation*, 20(5):921–927.

Jacobs, R. A. and Jordan, M. I. (1991). Adaptive mixture of local experts. *Neural Computation*, 3:79–87.

Kavraki, L. E., Svestka, P., Latombe, J.-C., and Overmars, M. (1996). Probabilistic roadmaps for path planning in high dimensional configuration spaces. *IEEE Trans. on Robotics and Automation*, 12(4).

Kim, H.-M., Mallick, B., and Holmes, C. (2005). Analyzing nonstationary spatial data using piecewise gaussian processes. *Journal of the American Statistical Assoc.*, 100.

Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, 220(4598):671–680.

Klanke, S. and Vijayakumar, S. (2008). A library for locally weighted projection regression. http://www.ipab.inf.ed.ac.uk/slmc/software/lwpr/lwpr_doc.pdf.

Kolter, J. Z., Kim, Y., and Ng, A. Y. (2009). Stereo vision and terrain modeling for quadruped robots. In *Proceedings of the International Conference on Robotics and Automation (ICRA)*.

Kweon, I. and Kanade, T. (1992). High-resolution terrain map from multiple sensor data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):278–292.

Lang, T., Plagemann, C., and Burgard, W. (2007). Adaptive non-stationary kernel regression for terrain modelling. In *Proc. of the Robotics: Science and Systems Conference (RSS)*.

Middendorf, M. and Nagel, H. (2002). Empirically convergent adaptive estimation of grayvalue structure tensors. In *Proc. of the DAGM Symposium on Pattern Recognition*, pages 66–74.

Miller, J. R. (2002). *A 3D Color Terrain Modeling System for Small Autonomous Helicopters*. PhD thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA.

Neumann, G., Pfeiffer, M., and W., M. (2007). Efficient continuous-time reinforcement learning with adaptive state graphs. In *Proc. of the European Conference on Machine Learning (ECML)*.

Nguyen-Tuong, D., Seeger, M., and Peters, J. (2008). Local gaussian process regression for real time online model learning and control. In *Adv. in Neural Inform. Processing Systems (NIPS)*.

Paciorek, C. and Schervish, M. (2004). Nonstationary covariance functions for Gaussian process regression. In *Adv. in Neural Information Processing Systems 16*. MIT Press, Cambridge, MA.

Parra, C., Murrieta-Cid, R., Devy, M., and Briot, M. (1999). 3-d modelling and robot localization from visual and range data in natural scenes. In *ICVS '99: Proceedings of the First International Conference on Computer Vision Systems*, pages 450–468, London, UK. Springer-Verlag.

Pfaff, P. and Burgard, W. (2005). An efficient extension of elevation maps for outdoor terrain mapping. In *Proc. of the Int. Conf. on Field and Service Robotics (FSR)*, pages 165–176.

Pfingsten, T., Kuss, M., and Rasmussen, C. (2006). Nonstationary gaussian process regression using a latent extension of the input space. In *Extended Abstract in Proc. of ISBA Eighth World Meeting on Bayesian Statistics*, Valencia, Spain.

Plagemann, C. (2008). *Gaussian Processes for Flexible Robot Learning*. PhD thesis, University of Freiburg, Department of Computer Science.

Plagemann, C., Fox, D., and Burgard, W. (2007). Efficient failure detection on mobile robots using particle filters with gaussian process proposals. In *Proc. of the Twentieth International Joint Conference on Artificial Intelligence (IJCAI)*, Hyderabad, India.

Plagemann, C., Kersting, K., and Burgard, W. (2008a). Non-stationary gaussian process regression using point estimates of local smoothness. In *Proc. of the European Conference on Machine Learning (ECML)*, Antwerp, Belgium.

Plagemann, C., Mischke, S., Prentice, S., Kersting, K., Roy, N., and Burgard, W. (2008b). Learning predictive terrain models for legged robot locomotion. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Nice, France.

Polidori, L. and Chorowicz, J. (1993). Comparison of bilinear and brownian interpolation for digital elevation models. *ISPRS journal of photogrammetry and remote sensing*, 48:18–23.

Quinonero-Candela, J. and Rasmussen, C. (2006). A unifying view of sparse approximate gaussian process regression. *JMLR*, 6(2):1939–1960.

Rasmussen, C. and Ghahramani, Z. (2002). Infinite mixtures of Gaussian process experts. In *Advances in Neural Information Processing Systems 14*. MIT Press, Cambridge, MA.

Rasmussen, C. E. and Williams, C. K. (2006). *Gaussian Processes for Machine Learning*. Adaptive Computation and Machine Learning. The MIT Press, Cambridge, Massachusetts.

Rees, W. G. (2000). The accuracy of digital elevation models interpolated to higher resolutions. *International Journal of Remote Sensing*, 21:7–20.

Sampson, P. and Guttorp, P. (1992). Nonparametric estimation of nonstationary spatial covariance structure. *Journal of the American Stat. Association*, 87:108–119.

Schmidt, A. and OHagan, A. (2003). Bayesian inference for nonstationary spatial covariance structure via spatial deformations. *JRSS, series B*, 65:745–758.

Schwaighofer, A., Grigoras, M., Tresp, V., and Hoffmann, C. (2004). A Gaussian process positioning system for cellular networks. In *Proc. of Neural Information Proc. Systems (NIPS)*.

Shen, Y., Ng, A. Y., and Seeger, M. (2005). Fast gaussian process regression using KD-trees. In *Proc. of the Conf. on Neural Information Processing Systems (NIPS)*.

Siegwart, R. and Nourbakhsh, I. R. (2004). *Introduction to autonomous mobile robots*. MIT press.

Snelson, E. and Ghahramani, Z. (2007). Local and global sparse gaussian process approximations. In *Artificial Intelligence and Statistics*, volume 11.

Sollich, P. and Williams, C. K. I. (2004). Using the equivalent kernel to understand gaussian process regression. In *Proc. of the Conf. on Neural Information Processing Systems (NIPS)*.

Ting, J., Mistry, M., Peters, J., Schaal, S., and Nakanishi, J. (2006). A bayesian approach to nonlinear parameter identification for rigid body dynamics. In *Proceedings of Robotics: Science and Systems*, Philadelphia, USA.

Tresp, V. (2000). Mixtures of gaussian processes. In *Proc. of the Conf. on Neural Information Processing Systems (NIPS)*.

Triebel, R., Pfaff, P., and Burgard, W. (2006). Multi-level surface maps for outdoor terrain mapping and loop closing. In *Proc. of the Intern. Conference on Intelligent Robots and Systems (IROS)*.

Vijayakumar, S., D'Souza, A., and Schaal, S. (2005). Incremental online learning in high dimensions. *Neural Computation*, 17:2602–2634.

Vijayakumar, S. and Schaal, S. (2000). Locally weighted projection regression: Incremental real time learning in high dimensional space. In *Proc. of the International Conference on Machine Learning (ICML)*, pages 1079–1086, San Francisco, CA, USA.

Wellington, C., Courville, A., and Stentz, A. (2005). Interacting Markov random fields for simultaneous terrain modeling and obstacle detection. In *Proc. of Robotics: Science and Systems*.