Pose Consistency KKT-Loss for Weakly Supervised Learning of Robot-Terrain Interaction Model

Vojtěch Šalanský, Karel Zimmermann, Tomáš Petříček, and Tomáš Svoboda

***Abstract*—We address the problem of self-supervised learning for predicting the shape of supporting terrain (i.e. the terrain which will provide rigid support for the robot during its traversal) from sparse input measurements. The learning method exploits two types of ground-truth labels: dense 2.5D maps and robot poses, both estimated by a usual SLAM procedure from offline recorded measurements. We show that robot poses are required because straightforward supervised learning from the 3D maps only suffers from: (i) exaggerated height of the supporting terrain caused by terrain flexibility (vegetation, shallow water, snow or sand) and (ii) missing or noisy measurements caused by high spectral absorbance or non-Lambertian reflectance of the measured surface. We address the learning from robot poses by introducing a novel KKT-loss, which emerges as the distance from necessary Karush-Kuhn-Tucker conditions for constrained local optima of a simplified first-principle model of the robot-terrain interaction. We experimentally verify that the proposed weakly supervised learning from ground-truth robot poses boosts the accuracy of predicted support heightmaps and increases the accuracy of estimated robot poses. All experiments are conducted on a dataset captured by a real platform. Both the dataset and codes which replicates experiments in the paper are made publicly available as a part of the submission.**

***Index Terms*—Deep learning for visual perception, representation learning.**

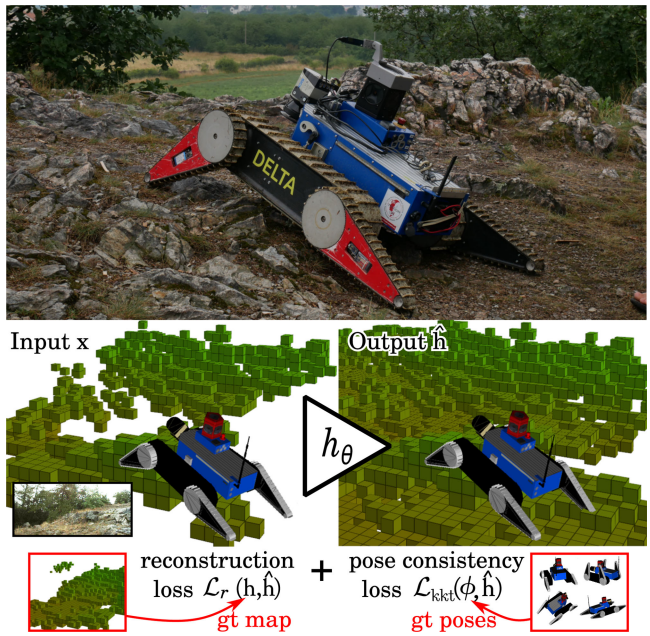

Fig. 1. **Top:** The real platform is equipped with four independently actuated flippers, which allows traversing complex terrains. **Bottom:** The input $\mathbf{x}$ of the prediction network h_θ is the sparse heightmap obtained by projecting the ICP-aligned lidar scans on the discretized horizontal plane, enriched by visual features. The proposed approach learns to predict the support heightmaps $\hat{\mathbf{h}}$ by enhancing the reconstruction loss by the pose-consistency loss. The proposed pose-consistency loss explicitly models the robot-terrain interaction and enforces the predicted support terrain to be physically consistent with the robot poses. Ground-truth map and ground-truth poses used for learning are obtained from all the measurements along the training trajectories.

I. Introduction

ACCURATE real-time prediction of robot-terrain interaction from raw sensory measurements is crucial for many mobile robotic tasks ranging from computing traversability/costmap for high-level path planning [1] to state representation for low-level motion control [2]. Even though a usual low-level map such as ICP-aligned lidar scans (optionally discretized to voxelmap or heightmap) is often sparse and assumes terrain to be rigid, it is often used as an input to these tasks [1], [3]–[6]. In contrast to others, we propose to predict an intermediate representation – *the shape of supporting terrain*. We show that such architecture outperforms existing state-of-the-art methods in terms of accuracy of predicted supporting terrain and consequently estimated robot poses. Since there is no straightforward way to obtain the ground-truth shape of supporting terrains without manual annotations, we propose to learn it in a self-supervised way from "future" maps and robot poses optimized in simultaneous localization and mapping pipeline (SLAM).

We define the supporting terrain as the layer of terrain, which can provide rigid support for the robot during its traversal. For example, for flexible terrain such as grass, the supporting terrain corresponds to the shape of the lidar-immeasurable rigid layer of the terrain (ground). The shape of supporting terrain is modelled by a dense 2.5D heightmap.

The most straightforward way of estimating the supporting heightmap is linear interpolation from ICP-aligned lidar measurement. We show that such an approach typically suffers from

Manuscript received December 22, 2020; accepted April 8, 2021. Date of publication April 30, 2021; date of current version May 18, 2021. This letter was recommended for publication by Associate Editor M. Ghaffari and Editor S. Cadena Lerma upon evaluation of the reviewers' comments. This work was supported in part by OP VVV MEYS funded project CZ.02.1.01/0.0/0.0/16_019/0000765 "Research Center for Informatics", by Czech Science Foundation under Project 20-29531S; Grant Agency of the CTU Prague under Project SGS20/128/OHK3/2T/13 and in part by Defense Advanced Research Projects Agency (DARPA). *(Corresponding author: Vojtech Salansky.)*

The authors are with the Department of Cybernetics, Faculty of Electrical Engineering, Czech Technical University in Prague, 166 36 Prague, Czech Republic (e-mail: salanvoj@fel.cvut.cz; zimmerk@fel.cvut.cz; petrito1@fel.cvut.cz; svobodat@fel.cvut.cz).

This letter has supplementary downloadable material available at https://doi.org/10.1109/LRA.2021.3076957, provided by the authors.

Digital Object Identifier 10.1109/LRA.2021.3076957

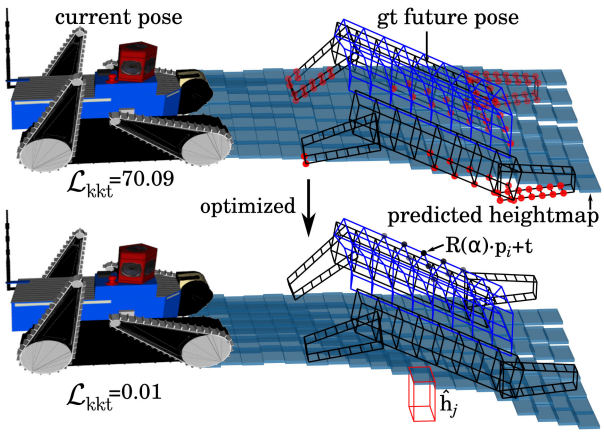

Fig. 2. Robot is represented as a set of mass points $\mathbf{p}_i, m_i$ Terrain is represented as heightmap $\hat{h}_j$. **Top:** Network predicts the shape of the supporting terrain (blue heightmap) from measurements available in the current pose. The ground-truth pose is inconsistent with the terrain since the robot model is in collision (red points). **Bottom:** Optimization of KKT-loss for a given ground-truth pose yields physically plausible reconstruction, which does not force the terrain to copy the shape of the robot.

(i) missing measurements caused by terrain self-occlusions, (ii) the low spatial resolution of distant terrains, (iii) missing or noisy measurements caused by high spectral absorbance or non-Lambertian reflectance of the measured surface, (iv) exaggerated height of supporting terrain caused by terrain deformations (flexible vegetation, shallow water, snow or sand).

Issues (i) and (ii) could be partially overcome by learning to interpolate the terrain in a self-supervised way, where ground truth is estimated from offline-optimized 3D maps of the environment [7]. This approach partially suppresses (i) the terrain occlusions and (ii) low-spatial resolution by introducing the measurements from additional viewpoints into the learning procedure, however (iii) the lidar unfriendly surfaces and (iv) the terrain flexibility remains unresolved.

In order to address issues (iii) and (iv), we enhance the fully-supervised reconstruction loss of [7] by a new pose-consistency loss; see Fig. 1 for the outline. The proposed architecture simultaneously optimizes two losses: Reconstruction loss $\mathcal{L}_r(\mathbf{h}, \hat{\mathbf{h}})$, where ground-truth heightmap $\mathbf{h}$ (obtained from offline-optimized maps) enforces the predicted support terrain $\hat{\mathbf{h}}$ to be close to its rigid reconstruction, and pose consistency loss $\mathcal{L}_{\text{kkt}}(\boldsymbol{\phi}, \hat{\mathbf{h}})$, where ground-truth robot pose $\boldsymbol{\phi}$ implicitly enforces the predicted supporting terrain to provide a necessary (yet collision-free) support of the robot. The pose consistency loss provides additional supervision on places, where ground-truth heightmap $\mathbf{h}$ is not available. Fig. 2 shows supporting terrain after optimization of the pose consistency loss.

To construct the pose-consistency loss, we exploit a simple yet non-convex first-principle model, which assumes that the robot pose on an uneven terrain corresponds to a minimum of its potential energy with respect to robot-terrain collision constraints. The solution to this problem provides the physically plausible robot pose on a given heightmap. We construct the pose-consistency loss to answer the opposite question: *does the predicted heightmap make the ground-truth pose to be a solution to this problem?* To do so, we simplify the problem and search for heightmaps, for which the ground-truth pose satisfies the necessary Karush-Kuhn-Tucker (KKT) conditions [8] for the constrained local optima of this first-principle model. The proposed KKT-loss is constructed to measure the distance from these KKT conditions. Consequently, its optimization directly leads towards physically consistent heightmaps in the sense that any heightmap, which zeros the KKT-loss, is considered to be physically consistent with the ground-truth pose.

Main contribution lies in:

- introducing a novel KKT-loss, which allows for weakly-supervised learning of the supporting terrain from ground-truth poses and suggesting an algorithm for its efficient optimization
- evaluating the proposed method on a real dataset and publishing the codes and the dataset, which replicates the results reported in the experiments.[1]

II. Related Work

This section consists of two parts. The first part summarizes approaches that predict the robot-terrain interaction and discuss their relation to the proposed supporting terrain. The second part discusses learning approaches that resemble proposed KKT-loss in the sense that they learn to predict values connected with ground-truth labels through an optimization problem.

A. Robot-Terrain Interaction Models

Typical quantities, which are predicted in order to model the robot-terrain interaction, are arbitrary values that represent the expected behaviour of the robot on uneven terrain. We briefly discuss several different quantities such as the expected pose of the robot on the terrain [9], robot-terrain reaction score [6], friction/slippage coefficient [10]. Most of these methods is not a direct competitor; however, their accuracy heavily depends on the accuracy of reconstructed terrain, which is provided as an input. Consequently, we discuss different models of terrain shapes such as ICP-aligned pointclouds [11] and neural-network-refined voxelmaps [7].

Geometrical analysis: The most straightforward way of predicting the robot-terrain interaction is the direct geometrical analysis of the terrain shape, such as point cloud or heightmap. Geometrical analysis typically exploits heuristics based on manually chosen features, such as terrain normals, height difference, slope, roughness, and robot shape [12], [13]. Some approaches [9], [14] iteratively optimizes the robot-terrain transformation to obtain the contact points and static robot pose.

Self-supervised learning: Other methods learn to predict the robot-terrain interaction directly. Methods such as Suryamurthy *et al.* [5] learn to estimate terrain roughness from RGB images for wheeled Centauro robot, where terrain roughness labels are automatically computed from SfM optimized heightmaps. In contrast to us, such an approach inherently suffers from the inability to assess the terrain's flexibility and the inability to reconstruct visually homogeneous terrain. Many others directly learn from the recorded behaviour of robot on the terrain. For example, Wellhausen *et al.* [6] estimate the ground reaction score for the legged ANYmal robot, where force-torque sensors automatically estimate labels. Similarly, Angelova *et al.* [10] estimate the slippage coefficient for a planetary rover. The

[1] github.com/ctu-vras/pose-consistency-kkt-loss

approach proposed by Chavez *et al.* [15] learn to predict the terrain traversability estimated from the simulator and Nouza *et al.* [16] predicts robot poses.

Most of the previously discussed methods heavily depend on the accuracy of provided terrain shape. Since ICP-aligned point clouds [11] or heightmaps are inaccurate, it is possible to improve their accuracy by a refinement step, which provides a more accurate estimate of a terrain shape. Methods such as [7], [17], [18] complete the sparse laser measurements using the neural networks. In contrast to our approach, these methods do not take the robot-terrain interaction (such as robot pose) into account, and therefore they can only reconstruct the rigid terrain with lidar-friendly surfaces.

In contrast to the straightforward prediction of previously mentioned quantities (terrain roughness score, slippage, ground reaction score, or the traversability), we suggest predicting the supporting terrain as a preliminary representation, which is more suitable input to these methods than a noisy 3D point clouds. This claim is experimentally verified on the pose prediction and heightmap reconstruction problem.

B. Learning With Implicit Optimization

The problem of learning the shape of support terrain from ground-truth poses belongs to the class of learning approaches, where it is not easy to obtain the ground truth for fully-supervised learning. However, it is possible to use predicted values as an input to an optimization problem, the solution of which can be compared to an alternative ground truth that is easy to obtain. It has been recently shown that a convex optimization problem could be used as a differentiable layer in neural network [19]. While various applications ranging from optimal control to signal denoising has been shown, it cannot be directly used since we need to differentiate through the non-convex first principle model of the robot-terrain interaction.

Inverse reinforcement learning (IRL) is a class of methods, which use an optimization layer to relate predicted values (costmap) to ground truth (expert trajectories). Such approaches exploit an expert driver to collect trajectories, which serves as easy-to-obtain ground truth for learning the costmap model, for which the straightforward ground truth is typically not available. The IRL assumes that the expert driver has a latent costmap, for which the executed trajectory is optimal (i.e. has the lowest sum of costs). Given this assumption, it is possible to learn to predict this costmap by searching the optimal trajectories on predicted costmaps and then comparing them with those executed by the driver. The learning requires backpropagating through the layer that performs the search of optimal trajectories for a given costmap. Authors typically either train differentiable policies that imitate drivers' behaviour or use Dijkstra in each iteration and than backpropagate gradients along the fixed optimal trajectory only. IRL has been used on several mobile platforms. For example, Silver *et al.* [1] apply it on the six-wheeled Crusher military platform, while Wulfmeier *et al.* [4] or Zeng *et al.* [20] learned spatial traversability for driving an autonomous car in complex urban environments. In contrast to previous self-supervised approaches, the IRL allows to learn also the non-traversable terrain directly on the real platform from the expert behaviour. However, the connection between the expert trajectories and the traversability is often weak; therefore, the problem is typically ill-posed [21]. In addition to that, expert trajectories are often sub-optimal, which significantly harm the resulting costmap. Since IRL methods use different ground truth (the expert trajectories) and the optimization layer (search for cost-minimizing path), it cannot be easily applied for learning from ground-truth poses.

Multiple Instance Learning (MIL): One way to avoid direct backpropagation through an optimization layer is to explicitly generate all possible predictions, which make a given ground truth to be the solution of the underlying optimization problem, and then train on these predictions via MIL. For example, the proposed KKT-loss enforces the predicted support heightmap to provide a necessary (yet collision-free) support of the robot. One could achieve a similar effect by employing the MIL [22]. Such an approach would generate huge positive bags consisting of all supporting terrains, which are physically consistent with a given ground-truth pose, and then minimize the reconstruction loss from the closest sample in each positive bag. However, the number of heightmaps consistent with a single pose is immense since it grows exponentially with the predicted heightmap size. In contrast to the multiple instance learning, the proposed KKT-loss does not require to generate all pose-consistent heightmaps explicitly; it just measures the distance from KKT conditions.

Multi-task learning: Another alternative, which allows to easily backpropagate through the optimization problem, is to train a network, which directly estimates solutions of the optimization layer and use this network as a simply differentiable replacement for the optimization layer. A similar idea appears either in multitask learning [23] or weakly-supervised learning [24]. The main disadvantage is that the replacement network strongly biases the learning process by its own inaccuracy. This approach is detailed in the paper, and the proposed KKT-loss is quantitatively compared with it.

III. THEORY

We train a convolutional network $h_\theta : \mathbb{X} \to \mathbb{H}$, which maps *sparse* heightmaps $\mathbf{x} \in \mathbb{X}$, estimated by projecting the ICP-aligned [11] lidar scans on the discretized horizontal plane (optionally enriched by visual features), on *dense support* heightmaps $\hat{\mathbf{h}} \in \mathbb{H}$, where θ is the vector of network parameters, see Fig. 3 for an overview. In our experiments, the input $\mathbf{x}$ is either a two-channel 2D array with heights in the first channel and NaNs binary encoded in the second channel or a multichannel 2D array with heights, NaNs, and visual features. We exploit two types of self-supervised labels (i) ground-truth heightmaps $\mathbf{h} \in \mathbb{H}$, (ii) ground-truth robot poses $\boldsymbol{\phi} \in \Phi$, both estimated offline from recorded measurements by a SLAM pipeline [11]. Therefore, the ground truth for a prediction in a certain time-stamp also includes future measurements. Both types of labels are imperfect. Heightmaps are noisy, incomplete, and overestimate support heights on flexible terrains. Robot poses constrain the shape of the underlying terrain only by its physical consistency with the predicted terrain. Learning is defined as the minimization of composite loss on both types of labels.

Learning from ground-truth heightmaps: Given a ground-truth heightmap $\mathbf{h}$ and predicted supporting heightmap $\hat{\mathbf{h}}$, we use (optionally asymmetric) L2-loss $\mathcal{L}_r(\mathbf{h}, \hat{\mathbf{h}}) = \| \max\{a(\mathbf{h} - \hat{\mathbf{h}}), \hat{\mathbf{h}} - \mathbf{h}\}\|_2^2$, which optionally provides decreased loss for underestimated heights. Especially, if $a = 0$ the reconstruction loss is quadratic upper bound, if $a = 1$ the reconstruction loss becomes a usual L2 loss.

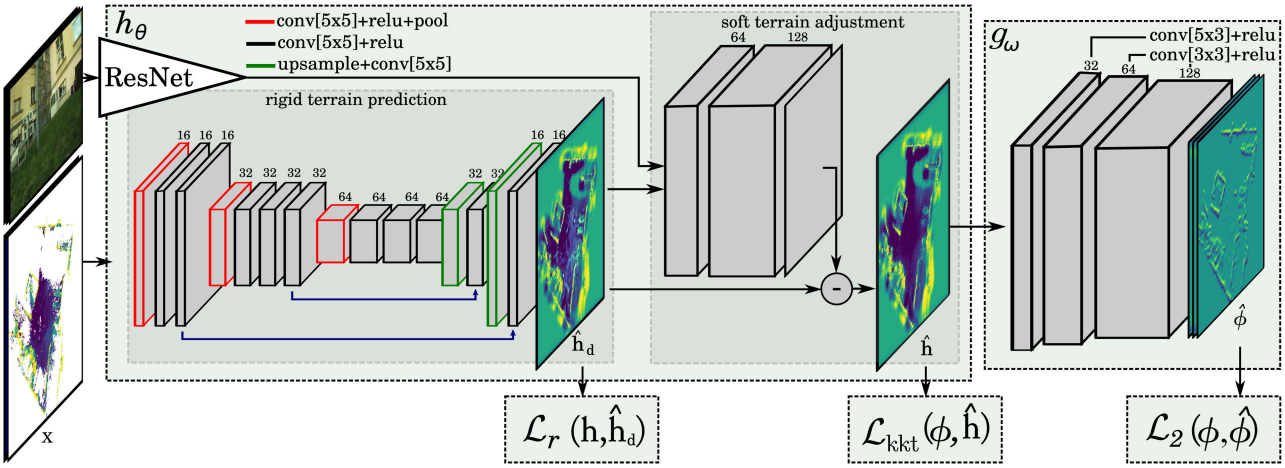

Fig. 3. Visualization of network architectures. Network h_θ predicts the dense supporting heightmap from sparse input. Training minimizes reconstruction loss and pose consistency loss. We propose two types of pose consistency losses: (i) KKT-loss and (ii) pose predicting loss (concatenation of pretrained pose regression network g_ω predicting roll, pitch, z on the dense supporting heightmap with L2 loss on ground-truth poses).

Learning from ground-truth poses: We introduce an additional penalty for predicting heightmaps, which are physically inconsistent with the ground-truth robot pose, such as terrains that either does not provide sufficient stability of all degrees of freedom of the robot or colliding with the robot body. To enforce the penalty, we propose two different pose-consistency losses: (i) the KKT-loss $\mathcal{L}_{kkt} : \Phi \times \mathbb{H} \to \mathbb{R}$, which is purely based on the first-principle model (Section III-B) and (ii) pose-predicting loss $\mathcal{L}_p : \Phi \times \mathbb{H} \to \mathbb{R}$ (Section III-C), which contains pose-predicting regressor g_ω followed by L2 loss on predicted $\hat{\phi}$ and ground-truth pose ϕ: $\mathcal{L}_p(\phi, \hat{\mathbf{h}}) = \mathcal{L}_2(\phi, g_\omega(\hat{\mathbf{h}}))$. The pose-predicting regressor g_ω has to be trained in advance.

A. Architecture

We study two different cases: (i) strictly rigid terrain, on which the supporting terrain is equivalent to its dense reconstruction, and (ii) partially flexible terrain, on which the predicted support terrain is lower or equal to its dense reconstruction.

1) Strictly Rigid Terrain: In this case, the input $\mathbf{x}$ is only the sparse heightmap (2D array with NaNs). The learning minimizes both losses simultaneously $\mathcal{L}_r(\mathbf{h}, \hat{\mathbf{h}}) + \mathcal{L}_{\text{kkt}}(\phi, \hat{\mathbf{h}})$. For the reconstruction loss, we use $a = 1$, since the output should directly correspond to ground truth $\mathbf{h}$.

2) Partially Flexible Terrain: In this case, the input $\mathbf{x}$ is the sparse heightmap enriched by visual features obtained by projecting outputs of RGB network on the heightmap. We observed that the simultaneous minimization of both losses on a flexible terrain suffers from undesirable interference. The reconstruction loss pulls predicted heights towards lidar-measured heights on flexible terrain, while the pose consistency loss enforces lower heights that do not collide with the robot body. To avoid such undesirable behaviour, we propose to divide h_θ into two sub-networks. The first sub-network (denoted as "rigid terrain prediction" in Fig. 3) is responsible for reconstructing the terrain $\hat{\mathbf{h}}_d$ as it has been rigid, the second sub-network (denoted as "soft terrain adjustment" in Fig. 3) predicts the terrain flexibility. The reconstruction loss is connected between these sub-networks, while the pose consistency loss is connected in the end. The learning minimizes compound loss: $\mathcal{L}_r(\mathbf{h}, \hat{\mathbf{h}}_d) + \mathcal{L}_{\text{kkt}}(\phi, \hat{\mathbf{h}}) + \mathcal{L}_2(\hat{\mathbf{h}}_d, \hat{\mathbf{h}})$, where the last term slightly encourages the predicted supporting terrain to be similar to the rigid reconstruction $\hat{\mathbf{h}}_d$, see Fig. 3. The reconstruction loss naturally enforces that the rigid reconstruction $\hat{\mathbf{h}}_d$ is a necessary part of the predicting process without influencing the second sub-network. In this scenario, we experimented with asymmetric reconstruction loss (i.e. with $a < 1$). Eventually, it turned out that the best is to enforce the upper bound directly into the h_θ architecture by making the "soft terrain adjustment" sub-network only able to decrease $\hat{\mathbf{h}}_d$ by subtracting the non-negative outputs from $\hat{\mathbf{h}}_d$.

B. KKT Loss

This section is structured as follows: Section III-B1 introduces a simple first-principle model, which allows to estimate robot pose ϕ, given a predicted heightmap $h_\theta(\mathbf{x})$. Section III-B2 then employs this model to construct the KKT loss. Section III-B3 describes efficient estimation of KKT loss by the inner-loop minimization of its KKT multipliers.

1) First-Principle Model: Let us assume that a set of mass points represents the robot $(\mathbf{p}_i, m_i)\ i = 1 \dots N$, where $\mathbf{p}_i \in \mathbb{R}^3$ are 3D coordinates and $m_i \in \mathbb{R}$ are weights, see Fig. 2. We omit modeling the passive compliance[2] of flipper-motors and assume, that the positions of four independently articulated flippers in a given time are fixed in a measured position, therefore flipper points become a rigid part of the model. Robot pose $\phi = [\boldsymbol{\alpha}, \mathbf{t}]$ is uniquely determined by its roll, pitch, yaw angles denoted by $\boldsymbol{\alpha}$ and translation vector $\mathbf{t} \in \mathbb{R}^3$. Matrix $\mathbf{R}(\boldsymbol{\alpha})$ denotes rotation matrix corresponding to rotation angles $\boldsymbol{\alpha}$. Given a predicted heightmap $\hat{\mathbf{h}} = h_\theta(\mathbf{x}) \in \mathbb{H}$, we define robot pose estimation problem as the minimization of its potential energy

$$\sum_i m_i \cdot g \cdot [\mathbf{R}(\boldsymbol{\alpha}) \cdot \mathbf{p}_i + \mathbf{t}]_z \tag{1}$$

[2]By passive compliance, we refer to flipper motors inability to lift the body without any additional support of the main tracks, which results in a passively smooth motion over the terrain.

with respect to collision constraints

$$\hat{h}_i - [\mathbf{R}(\boldsymbol{\alpha}) \cdot \mathbf{p}_i + \mathbf{t}]_z \leq 0 \quad \forall_i,$$

where $[.]_z$ denotes the z-coordinate of the vector inside the brackets and $\hat{h}_i$ is the height corresponding to point $\mathbf{R}(\boldsymbol{\alpha}) \cdot \mathbf{p}_i + \mathbf{t}$. Note, that if the pose needs to optimized (which is not the case of this section) then the point-height correspondences have to be re-established in each iteration.

2) Consistency of Predicted Heightmap With Ground-Truth Pose: We express the consistency of the predicted heightmap $\hat{\mathbf{h}}$ with a ground-truth pose $\boldsymbol{\phi}$ as the L2-distance from the necessary conditions for the constrained local optimality of $\boldsymbol{\phi}$ in this model. In particular, given the Lagrangian of the pose estimation model

$$L(\boldsymbol{\alpha}, \mathbf{t}, \mathbf{h}, \boldsymbol{\lambda}) = \sum_i m_i \cdot g \cdot [\mathbf{R}(\boldsymbol{\alpha}) \cdot \mathbf{p}_i + \mathbf{t}]_z + \\ + \sum_i \lambda_i (\hat{h}_i - [\mathbf{R}(\boldsymbol{\alpha}) \cdot \mathbf{p}_i + \mathbf{t}]_z), \tag{2}$$

where $\boldsymbol{\lambda}$ denotes the KKT multipliers, we express necessary optimality conditions:

Stationarity conditions

$$\frac{\partial L(\boldsymbol{\alpha}, \mathbf{t}, \mathbf{h}, \boldsymbol{\lambda})}{\partial \boldsymbol{\alpha}, \mathbf{t}} = \sum_i (m_i g - \lambda_i) \frac{\partial [\mathbf{R}(\boldsymbol{\alpha}) \cdot \mathbf{p}_i + \mathbf{t}]_z}{\partial \boldsymbol{\alpha}, \mathbf{t}} = 0\,,$$

complementary slackness conditions

$$\lambda_i \cdot (\hat{h}_i - [\mathbf{R}(\boldsymbol{\alpha}) \cdot \mathbf{p}_i + \mathbf{t}]_z) = 0\,,$$

primal feasibility conditions

$$\hat{h}_i - [\mathbf{R}(\boldsymbol{\alpha}) \cdot \mathbf{p}_i + \mathbf{t}]_z \leq 0 \quad \forall_i\,,$$

dual feasibility conditions

$$\boldsymbol{\lambda} \geq \mathbf{0}\,.$$

While stationarity together with complementary slackness assure physical stability of the robot on the predicted heightmap $\hat{\mathbf{h}}$ in all considered degrees of freedom $\boldsymbol{\phi}$, the primal feasibility assures that the predicted heightmap does not collide with the robot model.

The KKT-loss of the predicted heightmap $\hat{\mathbf{h}}$ for a ground-truth pose $\boldsymbol{\phi}$ is then defined as follows

$$\mathcal{L}_{\text{kkt}}(\boldsymbol{\phi}, \hat{\mathbf{h}}) = \min_{\boldsymbol{\lambda}} \left\{ \left\| \sum_i (m_i g - \lambda_i) \frac{\partial [\mathbf{R}(\boldsymbol{\alpha}) \cdot \mathbf{p}_i + \mathbf{t}]_z}{\partial \boldsymbol{\alpha}, \mathbf{t}} \right\|_2^2 + \right. \\ + \sum_i \left(\lambda_i \cdot (\hat{h}_i - [\mathbf{R}(\boldsymbol{\alpha}) \cdot \mathbf{p}_i + \mathbf{t}]_z) \right)^2 + \\ \left. + C \cdot \left(\max\{0, \hat{h}_i - [\mathbf{R}(\boldsymbol{\alpha}) \cdot \mathbf{p}_i + \mathbf{t}]_z\} \right)^2 \;\middle|\; \boldsymbol{\lambda} \geq 0 \right\}, \tag{3}$$

where $C > 0$ is a learning hyper-parameter. In our experiments, we use $C = 100$, since it was a reasonable compromise between resulting primal feasibility and obtaining an optimization-friendly landscape of the KKT-loss. However we have not noticed any significant impact on the results for $C = 10$ or $C = 1000$ (if the learning rate was correspondingly adjusted).

Algorithm 1: Estimating the gradient of KKT-loss.

Input: Training batch $(\mathbf{x}_j, \boldsymbol{\phi}_j),\ j = 1 \dots M$
Robot model $p_i, m_i,\ i = 1 \dots N$
for $j = 1 \dots M$ **do**
  Predict heights (feedforward pass): $\hat{\mathbf{h}} = h_\theta(\mathbf{x}_j)$
  Estimate $\boldsymbol{\lambda}^*$ by solving the non-negative least squares Problem (4).
  Construct KKT-loss $\mathcal{L}_{\text{kkt}}(\boldsymbol{\phi}_j, h_{\boldsymbol{\theta}}(\mathbf{x}_j))$ by substituting inner loop minimizer $\boldsymbol{\lambda}^*$ into eq. (3).
  Cumulate gradient $\frac{\partial \mathcal{L}_{\text{kkt}}(\boldsymbol{\phi}_j, h_{\boldsymbol{\theta}}(\mathbf{x}_j))}{\partial \boldsymbol{\theta}}$ with respect to parameters $\boldsymbol{\theta}$
end

3) Estimating the KKT-Loss by Efficient Inner-Loop Optimization: Since the Lagrangian $L(\boldsymbol{\alpha}, \mathbf{t}, \mathbf{h}, \boldsymbol{\lambda})$ is linear in KKT multipliers, the minimization over $\boldsymbol{\lambda} > 0$ reduces to the following non-negative least squares problem:

$$\boldsymbol{\lambda}^* = \arg\min_{\boldsymbol{\lambda}} \left\{ \left\| \sum_i (m_i g - \lambda_i) \frac{\partial [\mathbf{R}(\boldsymbol{\alpha}) \cdot \mathbf{p}_i + \mathbf{t}]_z}{\partial \boldsymbol{\alpha}, \mathbf{t}} \right\|_2^2 + \right. \\ \left. + \sum_i \left(\lambda_i \cdot (\hat{h}_i - [\mathbf{R}(\boldsymbol{\alpha}) \cdot \mathbf{p}_i + \mathbf{t}]_z) \right)^2 \;\middle|\; \boldsymbol{\lambda} \geq 0 \right\}, \tag{4}$$

which is known to have an efficient solution [25]. For the sake of simplicity we skip the terms which does not depend on λ_i. Learning of parameters $\boldsymbol{\theta}$ with the KKT-loss is summarized in the following algorithm:

We provide fully differentiable implementation of the KKT-loss, which allows for learning through standard PyTorch interfaces such as `kkt_loss(net).backward()`.

C. Pose-Predicting Loss

We suggest to follow the idea of multi-task learning for semantic segmentation, where a classifier of another task is trained in advance and then used to estimate labels for the original task. Similarly, we start by learning the pose regressor $g_{\boldsymbol{\omega}} : \mathbb{H} \to \Phi$, which predicts the robot pose $\boldsymbol{\phi}$ given the heightmap $\mathbf{h}$; see Fig. 3 for network architecture details.

Since the robot observes a significantly larger area than it physically visits, there are many heightmap patches for which the pose is unknown. We denote J to be the set of indices of heightmap patches for which the real ground-truth pose is known and K to be the set of all indices. Since J is relatively small for training a reliable pose regressor, we decided to exploit also the real heightmap patches $K \setminus J$ for which the real pose is unknown by estimating synthetic robot poses on them.

In particular, given a pretrained reconstruction network $h_{\boldsymbol{\theta}}(\mathbf{x})$, we first reconstruct each dense heightmap $\hat{\mathbf{h}} = h_{\boldsymbol{\theta}}(\mathbf{x})$ and then find corresponding ground-truth pose as the solution of the first principle model:

$$\boldsymbol{\phi} = \arg\min_{\boldsymbol{\alpha}, \mathbf{t}} \sum_i m_i \cdot g \cdot [\mathbf{R}(\boldsymbol{\alpha}) \cdot \mathbf{p}_i + \mathbf{t}]_z \\ \text{subject to: } \hat{\mathbf{h}}_i - [\mathbf{R}(\boldsymbol{\alpha}) \cdot \mathbf{p}_i + \mathbf{t}]_z \leq 0 \quad \forall_i, \tag{5}$$

The solution is searched by the steepest gradient method with a quadratic exterior penalty function.

Given these synthetically estimated poses we learn the pose predicting regressor $g_{\boldsymbol{\omega}}$ on mixed training data K, consisting of both real and synthetically estimated poses

$$\arg\min_{\omega} \sum_{k \in K} \mathcal{L}_2(\boldsymbol{\phi}_k, g_{\boldsymbol{\omega}}(\mathbf{h}_k)). \tag{6}$$

Finally, we construct the pose predicting loss $\mathcal{L}_p$ as the L2 error between ground-truth pose $\boldsymbol{\phi}$ and predicted pose $\hat{\boldsymbol{\phi}} = g_{\boldsymbol{\omega}}(\hat{\mathbf{h}})$

$$\mathcal{L}_p(\boldsymbol{\phi}, \hat{\mathbf{h}}) = \mathcal{L}_2(\boldsymbol{\phi}, g_{\boldsymbol{\omega}}(\hat{\mathbf{h}})) = \mathcal{L}_2(\boldsymbol{\phi}, \hat{\boldsymbol{\phi}}) \tag{7}$$

and train the heightmap predicting network h_θ by minimizing pose predicting loss on real ground-truth poses $\boldsymbol{\phi}_j, j \in J$.

$$\sum_{j \in J} \mathcal{L}_p(\boldsymbol{\phi}_j, h_{\boldsymbol{\theta}}(\mathbf{x}_j)) \tag{8}$$

Since the change in parameters θ changes the distribution of reconstructed heightmaps $\hat{\mathbf{h}}_k$, the pose regressor should be re-trained. Theoretically, we should iterate this process until a fixed point is reached; however, we observed in the practice that a good initialization of h_θ is sufficient.

The pose predicting loss can be understood as a counterpart of the proposed KKT-loss. In contrast to KKT-loss, the pose predicting loss suffers from the regressor bias. The bias is strong since flexible terrains and lidar-unfriendly surfaces cannot be used for the regressor training. In the other hand, the pose-predicting loss does not require any inner-loop optimization, therefore its backpropagation is faster.[3]

IV. Experiments

We evaluate the performance of the proposed methods on a real-world dataset. The achieved results are compared with two state-of-the-art methods [7], [16]. For the sake of a fair comparison, all methods are implemented and trained from scratch on the same network architectures, training/testing subsets and hyper-parameters[4].

A. Dataset

Dataset consists of trajectories collected by the tracked robot during the traversal of various rigid and flexible obstacles. The robot was equipped with a spherical camera PointGrey Ladybug3, LiDAR SICK LMS-151, and inertial measurement unit (IMU) Xsens MTi-G. Internal camera parameters were factory calibrated, camera-lidar calibration was based on [26]. The data consist of the sparse input heightmaps $\mathbf{x}$ enriched by projected RGB-ResNet [27] features, ground-truth heightmaps $\mathbf{h}$ and ground-truth robot poses $\boldsymbol{\phi}$. All heightmaps are obtained as follows: Given a set of lidar scans, odometry measurements, and IMU data, we estimate a 3D point cloud map by a SLAM approach [11]. We filter out the ceiling automatically and discretize the resulting 3D point cloud into a heightmap with 10-cm bins. This heightmap is transformed into a robot-centric frame with gravity-aligned z-axis (i.e., the frame with roll and pitch equal to zero). The whole dataset consists of 871 training, 335 validation, and 745 testing heightmaps each of size $25.6\,\mathrm{m} \times 25.6$ m. It contains over $4 \cdot 10^6$ predictable heights (usually only a part of the heightmap is known) and more than $2 \cdot 10^4$ robot poses (there are multiple robot poses for each heightmap). The $2/3$ of the dataset (used for the "Strictly rigid terrains" experiment) consist only of rigid terrain, collected indoor with robot traversing over various obstacles and driving on uneven terrain in experimental mines (see Fig. 4). The remaining $1/3$ of the dataset (used in the "Partially flexible terrains" experiment) contains both rigid and flexible terrains. This part of the dataset was collected outdoor on various terrains such as cobblestones, rocks, paths, grass, or clay (see Fig. 5). While the proposed methods could be adapted to different robot models, the dataset we collected is specific to the robot and sensor model.

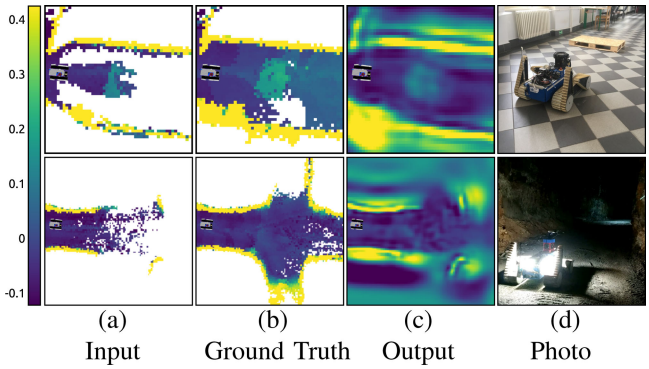

Fig. 4. Example crop from testing data. First row: Robot approaching the obstacle in the hallway. The scale is in meters. Yellow pixels are walls, dark blue denotes a floor an light blue blob in the middle is the obstacle. Second row: Robot in the experimental mine has noisy and missing data on the floor, our network fill the gaps.

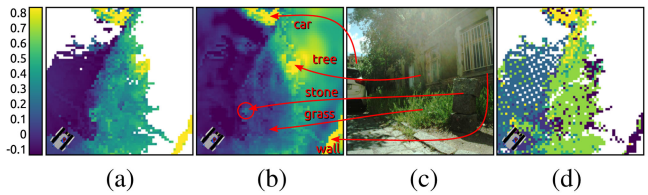

Fig. 5. Example crop from testing data. Robot measured an input heightmap (a) and predicts the supporting terrain (b) using the ResNet features (d). The scene is shown on image (c). Only maximum channel of ResNet features is shown in the image (d). The heightmaps (a) and (b) are in meters.

Input $\mathbf{x}$: The sparse input heightmap at time t is the heightmap estimated from lidar, IMU, and odometry measurements captured until this time by the procedure described above. We extend the heightmap by RGB features obtained from the camera image captured in time t. The RGB features are estimated from the camera image by a pre-trained ResNet network. The resulting 150 features are upsampled to original image resolution and projected onto the heightmap. The input for terrain reconstruction $\mathbf{x}$ has dimension $256 \times 256 \times 152$. The sparse array with terrain heights is in the first channel (see Fig. 4(a)), the binary mask of unknown measurements is in the second channel and the camera features in the subsequent 150 channels. In the experiment showing rigid terrain reconstruction, we use only the first two channels consisting of a sparse measurement and a corresponding mask.

[3]The backpropagation over the KKT-loss for a single robot pose takes about 35 ms on a quad-core 2.3 GHz Intel i7 processor with 16 GB RAM.

[4]All the approaches were trained using Adam optimizer with learning rate 10^{-4}, and the final weights were picked using the validation set.

Ground truth: The ground-truth label $\mathbf{h}$ at time t is the heightmap estimated from lidar, IMU, and odometry measurements captured along the whole recorded trajectory by the procedure described above. In contrast to sparse inputs, the ground truth also contains future measurements, therefore it is significantly denser and more accurate (compare Figures 4(a) and 4(b)). Nevertheless, the ground-truth heightmaps are still often inaccurate and incomplete, and they suffer from discretization along all axes (see Fig. 4(b)). Each ground-truth pose $\boldsymbol{\phi}$ contains roll and pitch angles of the robot and z coordinate of the position, resulting from the same SLAM procedure.

B. Methods

We evaluate the performance of proposed methods (r+kkt and r+p), which minimize compound loss described in Section III-A, with two state-of-the-art methods (li [16] and r [7]) in terms of the supporting terrain reconstruction accuracy and pose predicting accuracy. All compared methods use the same SLAM procedure [11] to align the input measurements and the pose is always predicted from estimated supporting terrain $\hat{\mathbf{h}}$ by pose predicting network g_ω. g_ω is trained[5] to minimize $\mathcal{L}_2(\boldsymbol{\phi}, g_\omega(\hat{\mathbf{h}}))$. A detailed description of implemented methods is in the following paragraphs.

li: This method is based on work [16], which predicts the robot poses from heightmaps by linear and Gaussian process regression on linearly interpolated heightmaps. We replicate this work, by filling the missing measurement by **l**inear **i**nterpolation and then predicting the pose by g_ω.

r: This method replaces the linear interpolation by deep convolution network h_θ trained by minimizing **r**econstruction loss $\mathcal{L}_r(\mathbf{h}, h_\theta(\mathbf{x}))$ similarly to [7]. The pose estimation remains the same as in work [16].

r+p: Method corresponds to learning with pose-predicting loss (Section III-C). This method first pretrains the h_θ and g_ω network in the same way as in previous methods. Then we connect the computational graphs of both networks h_θ and g_ω and train only θ parameters by backpropagating the **p**ose-predicting errors of g_ω and **r**econstruction errors of h_θ. In particular we minimize the weighted sum $\mathcal{L}_r(\mathbf{h}, \hat{\mathbf{h}}_d) + \mathcal{L}_p(\boldsymbol{\phi}, \hat{\mathbf{h}}) + \mathcal{L}_2(\hat{\mathbf{h}}, \hat{\mathbf{h}}_d)$.

r+kkt: Method corresponds to learning with KKT-loss (Section III-B). The pre-training procedure is the same, the final training minimizes weighted sum: $\mathcal{L}_r(\mathbf{h}, \hat{\mathbf{h}}_d) + \mathcal{L}_{\text{kkt}}(\boldsymbol{\phi}, \hat{\mathbf{h}}) + \mathcal{L}_2(\hat{\mathbf{h}}, \hat{\mathbf{h}}_d)$.

C. Evaluation and Discussion

1) Strictly Rigid Terrains: This experiment demonstrates the ability of proposed methods to reconstruct the heightmaps and predict poses from sparse inputs on rigid terrains; see Fig. 4 for testing examples and for the qualitative results. Since terrains are assumed to be rigid, we have omitted the "soft terrain adjustment" part of h_θ for this experiment (see Fig. 3). The quantitative results on the testing data are depicted in Table I. The first three rows show Root-Mean-Square error (RMS) on *fully-measured* heightmaps (i.e. those, where input heightmap $\mathbf{x}$ does not contain any NaNs). The second part of the table shows how methods perform on *all* heightmaps, including also incomplete input heightmaps $\mathbf{x}$.

[5]To train the g_ω network we used $5 \cdot 10^5$ terrain-poses, and the testing error evaluated on ground-truth heightmaps was 0.070.

TABLE I
MEAN AVERAGE TERRAIN RECONSTRUCTION ERRORS AND POSE ESTIMATION ERRORS FOR RIGID TERRAIN RECONSTRUCTION

	errors	li [11] [16]	r [11] [7] [16]	r+p (Sec.III-C)	r+kkt (Sec.III-B)
measured	roll [rad]	0.113	0.077	0.075	**0.073**
	pitch [rad]	0.075	0.069	0.065	**0.064**
	z [m]	0.067	**0.049**	0.072	0.057
all	roll [rad]	0.140	0.082	0.078	**0.075**
	pitch [rad]	0.191	0.089	0.072	**0.071**
	z [m]	0.140	0.080	0.082	**0.068**
	heightmap [m]	0.202	0.118	0.096	**0.089**

TABLE II
MEAN AVERAGE TERRAIN RECONSTRUCTION ERRORS AND POSE ESTIMATION ERRORS ON PARTIALLY FLEXIBLE TERRAIN

	errors	li [11] [16]	r [11] [7] [16]	r+p (Sec.III-C)	r+kkt (Sec.III-B)
measured	roll [rad]	0.089	0.084	0.056	**0.048**
	pitch [rad]	0.075	0.064	0.048	**0.042**
	z [m]	0.145	0.142	0.056	**0.045**
	heightmap*[m]	0.124	0.131	**0.057**	0.059
all	roll [rad]	0.093	0.098	0.063	**0.057**
	pitch [rad]	0.071	0.084	0.051	**0.044**
	z [m]	0.131	0.134	0.067	**0.055**
	heightmap*[m]	0.114	0.129	**0.075**	**0.075**

The results demonstrate that the supporting terrain reconstruction is an important intermediate step for pose prediction since the linear interpolation (li) suffers from sparsity of input heightmaps. We can also see that the backpropagation from ground-truth poses via proposed pose consistency losses ($\mathcal{L}_{kkt}$ and $\mathcal{L}_p$), improves the accuracy of the predicted terrains and poses. The results show that the proposed methods decrease the reconstruction error and pose predicting error by approximately 20%. Since the ground-truth height of extremely high and obviously untraversable obstacles such as walls, trees, or buildings is not well defined (e.g. due to unreliability of ceiling removal procedure), the heightmap reconstruction error is evaluated only on bins, where ground-truth height is lower than 0.3 m.

2) Partially Flexible Terrains: This experiment evaluates the ability to predict the shape of supporting terrain on both types of terrain (flexible and rigid); see Fig. 5 for an example of the testing data. In this particular example, the sparse input heightmap contains the heights measured by lidar on the high grass. Since these heights are misleading for predicting the robot pose on this type of terrain, the height of the grass is suppressed by the "soft terrain adjustment" part of the network (see Fig. 3). The rigid objects such as cars, stones, trees, roads, or buildings remain at the original level. The results are summarized in Table II. Since the ground-truth supporting terrain is generally unknown, we semi-manually annotated a selected subset of terrains, by fitting the local ground plane between robot-terrain contact points. The heightmap reconstruction error is denoted with * to point out that it is computed on the manually annotated subset of the dataset.

The benefit of backpropagating the pose consistency loss to the heightmap reconstructing network is even more obvious since it allows to predict the correct height on flexible terrains. Consequently, the pose predicting error and reconstruction error of proposed methods (r+p and r+kkt) is reduced by 50% with respect to the state-of-the-art methods (r and li). The accuracy

of r+kkt slightly outperforms r+p, since it is not biased by the g_ω regressor.

V. Conclusion

We have addressed the problem of learning to predict supporting terrain from SLAM-optimized 3D maps and robot poses. We have demonstrated that using the pose-consistency loss, which expresses the physical consistency of predicted terrain with the robot pose, improves the accuracy of predicted heightmaps and poses. In particular, two different pose consistency losses have been proposed: (i) KKT-loss and (ii) pose-predicting loss. The KKT-loss has been defined as the distance from the necessary conditions of the first principle model for a given ground-truth pose. The pose-predicting loss, first learns the pose predicting regressor and then minimizes the distance between predicted pose and ground-truth pose. In contrast to KKT-loss, the pose predicting loss requires prior learning and suffers from the regressor bias. The bias is strong since flexible terrains and lidar-unfriendly surfaces cannot be used for the training of the regressor. On the other hand, the pose-predicting loss does not require any inner-loop optimization, therefore its backpropagation is faster.

The experimental comparison shows that the proposed end-to-end differentiable architecture with proposed pose consistency losses reduces the error of predicted terrains and robot poses by 50% on the partially flexible dataset and 20% on the strictly rigid dataset when compared to state-of-the-art methods. Generalization of the trained model depends on a particular type of the robot model and a variety of training terrains. We believe that KKT-loss is directly usable for a broad range of mobile platforms such as wheeled or skid-steer robots, however its generalization for legged robots is unclear.

The dataset and codes replicating the reported experiments are made publicly available. We also provide the fully differentiable implementation of the KKT-loss, which can be simply included in any future learning methods through the standard PyTorch interfaces such as `kkt_loss(net, robot).backward()`.

References

[1] D. Silver, J. A. Bagnell, and A. Stentz, "Learning from demonstration for autonomous navigation in complex unstructured terrain," *Int. J. Robot. Res.*, vol. 29, no. 12, pp. 1565–1592, 2010.

[2] M. Pecka, K. Zimmermann, M. Reinstein, and T. Svoboda, "Controlling robot morphology from incomplete measurements," *IEEE Trans. Ind. Electron.*, vol. 64, no. 2, pp. 1773–1782, Feb. 2017.

[3] P. Krüsi, P. Furgale, M. Bosse, and R. Siegwart, "Driving on point clouds: Motion planning, trajectory optimization, and terrain assessment in generic nonplanar environments," *J. Field Robot.*, vol. 34, no. 5, pp. 940–984, 2017.

[4] M. Wulfmeier, D. Rao, D. Z. Wang, P. Ondruska, and I. Posner, "Large-scale cost function learning for path planning using deep inverse reinforcement learning," *Int. J. Robot. Res.*, vol. 36, no. 10, pp. 1073–1087, 2017.

[5] V. Suryamurthy, V. S. Raghavan, A. Laurenzi, N. G. Tsagarakis, and D. Kanoulas, "Terrain segmentation and roughness estimation using RGB data: Path planning application on the centauro robot," in *Proc. IEEE-RAS 19th Int. Conf. Humanoid Robots*, 2019, pp. 1–8.

[6] L. Wellhausen, A. Dosovitskiy, R. Ranftl, K. Walas, C. Cadena, and M. Hutter, "Where should I walk? Predicting terrain properties from images via self-supervised learning," *IEEE Robot. Automat. Lett.*, vol. 4, no. 2, pp. 1509–1516, Apr. 2019.

[7] K. Zimmermann, T. Petříček, V. Šalanský, and T. Svoboda, "Learning for active 3D mapping," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 1548–1556.

[8] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge Univ. Press: Cambridge, U.K., 2004.

[9] S. Fabian, S. Kohlbrecher, and O. Von Stryk, "Pose prediction for mobile ground robots in uneven terrain based on difference of heightmaps," in *Proc. IEEE Int. Symp. Safety, Secur., Rescue Robot. (SSRR)*, 2020, pp. 49–56.

[10] A. Angelova, L. Matthies, D. Helmick, G. Sibley, and P. Perona, "Learning to predict slip for ground robots," in *Proc. 2006 IEEE Int. Conf. Robot. Automat.*, 2006, pp. 3324–3331.

[11] F. Pomerleau, P. Krüsi, F. Colas, P. Furgale, and R. Siegwart, "Long-term 3D map maintenance in dynamic environments," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2014, pp. 3712–3719.

[12] I. Bogoslavskyi, O. Vysotska, J. Serafin, G. Grisetti, and C. Stachniss, "Efficient traversability analysis for mobile robots using the kinect sensor," in *Proc. Eur. Conf. Mobile Robots*, 2013, pp. 158–163.

[13] J. Gu, Q. Cao, and Y. Huang, "Rapid traversability assessment in 2.5D grid-based map on rough terrain," *Int. J. Adv. Robotic Syst.*, vol. 5, no. 4, pp. 389–394, 2008.

[14] M. Brunner, T. Fiolka, D. Schulz, and C. M. Schlick, "Design and comparative evaluation of an iterative contact point estimation method for static stability estimation of mobile actively reconfigurable robots," *Robot. Auton. Syst.*, vol. 63, pp. 89–107, 2015.

[15] R. O. Chavez-Garcia, J. Guzzi, L. M. Gambardella, and A. Giusti, "Learning ground traversability from simulations," *IEEE Robot. Automat. Lett.*, vol. 3, no. 3, pp. 1695–1702, Jul. 2018.

[16] T. Nouza, "Safe adaptive traversability learning for mobile robots," Master's thesis, Czech Tech. Univ. in Prague, Prague, 2014. [Online]. Available: http://hdl.handle.net/10467/24431

[17] W. Van Gansbeke, D. Neven, B. De Brabandere, and L. Van Gool, "Sparse and noisy lidar completion with RGB guidance and uncertainty," in *Proc. 16th Int. Conf. Mach. Vis. Appl.*, 2019, pp. 1–6.

[18] F. Ma, G. V. Cavalheiro, and S. Karaman, "Self-supervised sparse-to-dense: Self-supervised depth completion from lidar and monocular camera," in *Proc. Int. Conf. Robot. Autom.*, 2019, pp. 3288–3295.

[19] A. Agrawal, B. Amos, S. Barratt, S. Boyd, S. Diamond, and J. Z. Kolter, "Differentiable Convex Optimization Layers," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, pp. 1–13, 2019.

[20] W. Zeng *et al.*, "End-to-end interpretable neural motion planner," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 8652–8661.

[21] Q. P. Nguyen, B. K. H. Low, and P. Jaillet, "Inverse reinforcement learning with locally consistent reward functions," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 28, pp. 1–9, 2015.

[22] M.-A. Carbonneau, V. Cheplygina, E. Granger, and G. Gagnon, "Multiple instance learning: A survey of problem characteristics and applications," *Pattern Recognit.*, vol. 77, pp. 329–353, 2018.

[23] S. Ruder, "An Overview of Multi-Task Learning in Deep Neural Networks," 2017, *arXiv:1706.05098*. [Online]. Available: https://arxiv.org/abs/1706.05098

[24] Z. Huang, X. Wang, J. Wang, W. Liu, and J. Wang, "Weakly-supervised semantic segmentation network with deep seeded region growing," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 7014–7023.

[25] C. L. Lawson and R. J. Hanson, "Solving least squares problems," *Soc. Ind. Appl. Math.*, 1995, doi: 10.1137/1.9781611971217.

[26] J. Brabec, "Automated camera calibration from laser scanning data in natural environments," bachelor's thesis, Czech Tech. Univ. in Prague, Prague, 2014. [Online]. Available: http://hdl.handle.net/10467/24152

[27] B. Zhou *et al.*, "Semantic understanding of scenes through the ADE20 K dataset," *Int. J. Comput. Vis.*, vol. 127, no. 3, pp. 302–321, 2019.