These Maps Are Made For Walking: Real-Time Terrain Property Estimation for Mobile Robots

Parker Ewen[1], Adam Li[1], Yuxin Chen[1], Steven Hong[1], and Ram Vasudevan[1]

***Abstract*— The equations of motion governing mobile robots are dependent on terrain properties such as the coefficient of friction, and contact model parameters. Estimating these properties is thus essential for robotic navigation. Ideally any map estimating terrain properties should run in real time, mitigate sensor noise, and provide probability distributions of the aforementioned properties, thus enabling risk-mitigating navigation and planning. This paper addresses these needs and proposes a Bayesian inference framework for semantic mapping which recursively estimates both the terrain surface profile and a probability distribution for terrain properties using data from a single RGB-D camera. The proposed framework is evaluated in simulation against other semantic mapping methods and is shown to outperform these state-of-the-art methods in terms of correctly estimating simulated ground-truth terrain properties when evaluated using a precision-recall curve and the Kullback-Leibler divergence test. Additionally, the proposed method is deployed on a physical legged robotic platform in both indoor and outdoor environments, and we show our method correctly predicts terrain properties in both cases. The proposed framework runs in real-time and includes a ROS interface for easy integration.**

***Index Terms*— Semantic Mapping, Legged Robots.**

I. Introduction

Mapping from images or point clouds enables mobile robots to perform object avoidance and terrain traversal [1]. Prior work in mapping for mobile robots has focused on generating maps by reconstructing the surface geometry in the vicinity of the robot as a 2.5-D polygonal mesh [2], often referred to as an elevation map or contact surface. These representations describe the geometry of the robot's surroundings, but have no information regarding the properties of the underlying terrain such as the friction coefficient. The equations of motion governing the behavior of mobile robots are a function of both the internal state of the robot and the properties of the terrain over which the robot is traversing [3]. As a result, maps used for robot navigation should include information about these properties. Work has been done in estimating terrain properties such as friction or the internal shear coefficients of granular surfaces from single RGB images [4], [5], [6]. Building semantic maps with terrain property estimates using these methods has remained a challenge due to the difficulty of incorporating prior information into these maps.

[1]All authors affiliated with the Robotics Institute at the University of Michigan, 2505 Hayward Street, Ann Arbor, Michigan, USA, {pewen, adamli, chyuxin, hongsn, ramv}@umich.edu

This work is supported by the Ford Motor Company via the Ford-UM Alliance under award N022977, by the Office of Naval Research under Award Number N00014-18-1-2575, and in part by the National Science Foundation under Grant 1751093.

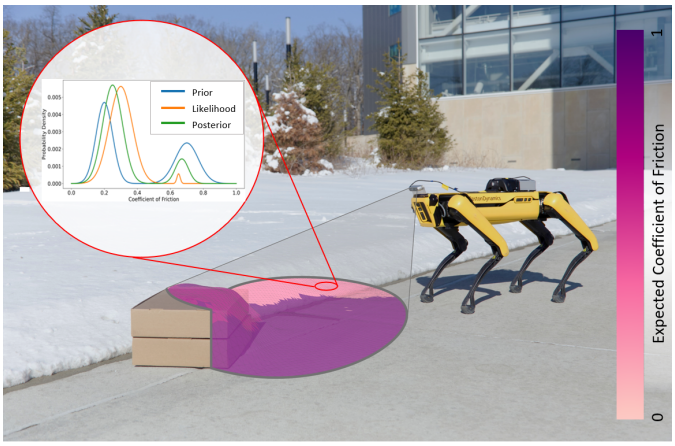

Fig. 1: An illustration of the real-time semantic mapping method proposed in this paper which recursively estimates terrain height and friction properties from RGB-D images. A triangular mesh represents the probabilistic estimate of the contact surface of the robot's surroundings. The proposed algorithm estimates terrain classes for each face of the triangular mesh using Bayesian inference and off-the-shelf semantic segmentation networks. Probability distributions for terrain properties are then computed (shown in the red inlay for one region). This algorithm runs on several robot systems including Boston Dynamics' Spot and Agility Robotics' Digit.

As illustrated in Fig. 1, the contributions of this paper are two-fold. First, we develop a novel dataset that is used to model the relationship between the coefficient of friction and a variety of terrain classes. Second, we propose a robot-centric semantic mapping framework by which geometric and terrain properties are estimated using a closed-form Bayesian inference algorithm.

The maps generated by this algorithm map 5m around the robot with discretizations of 2cm and are computed at speeds between 9Hz $\pm$5Hz. The proposed mapping algorithm is accessible via an open-source ROS package which takes RGB-D images and camera pose mean and covariance estimates as inputs and outputs the semantic map. We use our semantic mapping framework to estimate the coefficient of friction both in simulation and in the real-world and show that it outperforms existing methods from the literature in terms of accuracy in property estimation.

The remainder of this paper is organized as follows: Section II summarizes the mapping literature and Section III introduces preliminary concepts used throughout the paper. Section IV provides an overview of our method and Sections V and VI pertain to the recursive elevation mapping and recursive terrain property estimation portions of our algorithm, respectively. Section VII discusses implementation details and describes our dataset used to model the coefficient of friction for ten terrain classes. Section VIII describes the evaluation of our algorithm

both in simulation and in real-world scenarios. Section IX follows with concluding remarks and a discussion on future work.

II. Related Works

This section describes the existing literature on geometric and semantic mapping methods with an emphasis on algorithms pertaining to legged locomotion. Geometric maps represent the interface between free-space and occupied space, and are used to model the environment around the robot. Semantic maps include information such as terrain class, terrain properties, or additional high-level labels within the representation. Often semantic maps include a geometric representation onto which the semantic information is projected, and thus in these cases, they may contain more information than geometric maps.

A. Geometric Mapping

A robot's surroundings can be geometrically modelled by constructing representations of the underlying terrain surface using range sensor data. These range sensors produce sparse point cloud representations, which must then be converted into continuous or piecewise structures to be of use for planning. We organize prior work in geometric mapping into either volumetric or 2.5-D piecewise-planar categories.

Volumetric representations map the 3D geometry of a scene using discretized volumes, such as voxelized occupancy grids, or via Truncated Signed Distance Fields (TSDF) or Euclidean Signed Distance Fields (ESDF). Voxelized occupancy maps, for instance, are the three dimensional equivalents to occupancy grid maps. Voxels are given a binary label representing either free-space or occupied space, and the probability of each label given new data is updated by applying a Bayes filter. Such volumetric representations are often memory and computational resource intensive, but one may address some of these limitations by applying memory-efficient representations such as octrees [7]. Other representations, like TSDFs and ESDFs, map the distance to the nearest occupied cell rather than storing the entire volumetric representation of the environment [8].

Often for legged robotic locomotion and footstep planning, only a lower-dimensional rather than volumetric representation is required. Point-foot robots, such as Boston Dynamics's Spot, require maps with discretizations on the order of 1cm to perform footstep planning due to their footprint size [2]. State-of-the-art volumetric representations with 1cm resolution are only able to operate at around 1Hz [7].

The most common geometric mapping paradigm in legged robotics is the elevation map, where a piecewise-planar mesh is used to represent the terrain [1], [2]. Such maps also have parallels in the surface reconstruction community [9]. Recently, Bayesian inference has been applied to recursively update an elevation map to minimize the impact of noise from the range sensor measurements and robot pose estimation [2]. Such methods run at 20-100Hz, and have been used for legged locomotion and footstep planning [1]. One shortcoming of geometric mapping techniques is that they fail to account for terrain properties, which are an essential component in the stability of a legged robotic platform [3]. To compliment geometric mapping techniques, it is important to include information on terrain properties via semantic mapping.

B. Semantic Mapping

Semantic mapping is a broad field since high-level labels are task dependent. Methods predicting terrain properties are of interest for robotic navigation, and we limit our focus within semantic mapping to such methods. Semantic labels or attributes can be represented in abstract topological layers [10], but it has become commonplace for semantic information to be estimated from images or point clouds using neural networks and then projected onto geometric representations [11]. Visual information has been previously applied to predict the expected value for the coefficient of friction on roads [12] and other common terrain types [6], [5], as well as slip predictions for wheeled robots [13]. Current semantic mapping methods are non-recursive, meaning they cannot use priors to refine estimates, and as such, these methods are not robust while dealing with noisy sensor data.

One common semantic mapping paradigm is traversability estimation. Traversability mapping has been used to bypass the need to estimate terrain properties by instead estimating which regions in the environment a robot can traverse [11], [14]. Such mapping strategies often fail to consider that traversability is a function of the robot's internal state, such as the acceleration of the robot's center-of-mass or wheel velocity, and as such, traversability estimation methods often over- or under-approximate traversable regions [15]. Recent work has shown how to incorporate traversibility classifications into a voxel occupancy grid map using a Bayesian inference framework to update traversibility estimates based on new observations [11]. While this represents a large step towards a recursive framework for semantic mapping, it fails to address the short-comings associated with the robot-dependent nature of traversability.

III. Preliminaries

This section introduces notation, geometric concepts, coordinate frames, semantic segmentation, and probability theory used within this paper. Vectors, written as columns, are typeset in bold and lowercase, while sets and matrices are typeset in uppercase. The element i of a vector $\boldsymbol{x}$ is denoted as x_i. An n-dimensional open (resp. closed) interval is denoted by $(a, b)^n$ (resp. $[a, b]^n$). The power set of a set $\mathcal{A}$ is denoted as $2^{\mathcal{A}}$. Throughout the paper we let f refer to a probability mass or density function.

A. Geometry

Barycentric coordinate systems specify the location of a point with respect to the vertices of a b-dimensional simplex. Given a point $\boldsymbol{p} \in \mathbb{R}^a$ in Euclidean space, we compute the corresponding b-dimensional Barycentric coordinate representation $\boldsymbol{\lambda} \in \mathbb{R}^b$ using a change of basis function $g : \mathbb{R}^a \to \mathbb{R}^b$

[16], with $a \geq b$. We test whether a point $\boldsymbol{p}$ lies within a b-dimensional simplex using the Barycentric coordinates by the following theorem:

Theorem 1 ([17, (2)]). *Let $\boldsymbol{p} \in \mathbb{R}^a$ be a point in Euclidean space and let $\boldsymbol{\lambda} \in \mathbb{R}^b$ be its corresponding Barycentric coordinates. Point $\boldsymbol{p}$ lies within the simplex if and only if $\lambda_i \in (0, 1)$ for all λ_i in $\boldsymbol{\lambda}$.*

B. Sensors and Coordinate Frames

To simplify exposition, we assume the following:

Assumption 2. *An RGB-D camera that is attached to a robot with known camera intrinsics is used to collect data.*

These RGB-D images, camera intrinsics, and an estimated camera pose are used to compute projected point clouds within the camera frame. This process is discussed here as well as in Section III-C. This subsection introduces the notation used to describe coordinate frames and transformations between coordinate frames. Note that one could extend the algorithms presented in this paper to multiple cameras or even LiDAR; we focus on a single RGB-D camera for simplicity.

We denote a vector to point $\boldsymbol{p}$ in coordinate frame A as $\boldsymbol{p}_A$. The rotation matrix from coordinate frame A to B parameterized by the rotation angles $\boldsymbol{q}$ between the two frames is denoted as $R_A^B(\boldsymbol{q})$. Three coordinate frames are used within the paper: the world frame W, the sensor frame S, and the mapping frame M. The inertial frame is space-fixed and the environment is assumed to be static with respect to this frame. The sensor frame S is located at the center of the camera with the z-axis pointed out of the camera into the scene. The transformation from the robot's center-of-mass to the sensor frame is static, and we assume that this transformation is known. Lastly, the mapping frame M is defined in relation to the location of the robot. Its origin corresponds to the robot's center-of-mass projected onto the terrain, the x-axis (resp. y-axis) is oriented towards the front (resp. left-side) of the robot, and the z-axis is aligned with the z-axis of the inertial frame. Given the vector to any range sensor measurement point in the sensor frame, $\boldsymbol{p}_S$, we transform it into the mapping frame via an affine transform:

$$\boldsymbol{p}_M = R_S^{M\intercal}(\boldsymbol{q}) \cdot \boldsymbol{p}_S - \boldsymbol{t}_S^M \tag{1}$$

where $\boldsymbol{t}_S^M$ represents the translation from the sensor frame to the map frame.

C. Semantic Segmentation

Semantic segmentation assigns class probability scores to each pixel in an image. The classes in semantic segmentation are task-dependent. This paper focuses on the list of terrain classes described in Table I.

It is common to use convolutional neural networks to estimate pixel-wise class probability scores [18]. Let $I \in \mathbb{R}^{w \times h \times 3}$ denote an RGB image, where h, w are the height and width of the image in pixels. A trained semantic segmentation network takes an input image and outputs the pixel-wise terrain class probability scores $T \in \mathbb{R}^{w \times h \times k}$ for k terrain classes in the form of a k-dimensional Categorical Distribution. The accuracy of the semantic segmentation depends on the network used. Note, this is not the emphasis of this paper. We use the aligned depth image $D \in \mathbb{R}^{w \times h}$ to project the pixel-wise terrain class probability scores T into the sensor frame using the camera intrinsics and the camera projection equation [19, (10.38)] to obtain a point cloud representation of the semantically segmented image.

D. Probability

This subsection reviews the Categorical Distribution, the Dirichlet Distribution, and their relation to one another. We denote a random variable as z or the vector of random variables as $\boldsymbol{z}$. The Categorical Distribution is a discrete k-dimensional distribution parameterized by a vector $\boldsymbol{\theta} \in [0, 1]^k$. The probability mass function of the Categorical Distribution represents the probability that sample z belongs to class i, where $i \in \{1, 2, \ldots, k\}$:

$$f(z = i|\boldsymbol{\theta}) = \theta_i \tag{2}$$

The Dirichlet Distribution is a continuous k-variate probability distribution which is parameterized by a vector $\boldsymbol{\alpha} \in \mathbb{R}^k_{\geq 0}$ of positive reals. The probability density function of the Dirichlet Distribution is defined below:

$$f(\boldsymbol{\theta}|\boldsymbol{\alpha}) = \frac{\Gamma(\sum_{j=1}^k \alpha_j)}{\sum_{j=1}^k \Gamma(\alpha_j)} \prod_{j=1}^k \theta_j^{\alpha_j - 1} \tag{3}$$

where

$$\Gamma(\alpha_j) = \int_0^\infty x^{\alpha_j - 1} \exp(-x) dx. \tag{4}$$

Suppose we obtain n measurements $\mathcal{Z} = \{z_1, \ldots, z_n\}$ of a given region, represented as random variables drawn from a Categorical Distribution. Our goal is to apply Bayesian inference to predict the probability that a new measurement of the same region belongs to terrain class i given prior measurements $\mathcal{Z}$. That is we want to compute $f(z = i|\mathcal{Z}, \boldsymbol{\alpha})$. Note, that we have assumed for full generality that $f(z|\mathcal{Z}, \boldsymbol{\alpha})$ is a function of some hyperparameters $\boldsymbol{\alpha}$. To do this, one could compute $f(z|\mathcal{Z}, \boldsymbol{\alpha}) = \int_{\boldsymbol{\theta}} f(z|\boldsymbol{\theta}) f(\boldsymbol{\theta}|\mathcal{Z}, \boldsymbol{\alpha}) \mathrm{d}\theta$, but this would require constructing $f(\boldsymbol{\theta}|\mathcal{Z}, \boldsymbol{\alpha})$. By applying Bayes Theorem, one can write

$$f(z|\mathcal{Z}, \boldsymbol{\alpha}) = \int_{\boldsymbol{\theta}} f(z|\boldsymbol{\theta}) \frac{f(\mathcal{Z}|\boldsymbol{\theta}, \boldsymbol{\alpha}) f(\boldsymbol{\theta}|\boldsymbol{\alpha})}{f(\mathcal{Z}|\boldsymbol{\alpha})} \mathrm{d}\theta. \tag{5}$$

Generally, this integral is hard to compute exactly.

To compute a closed form expression for $f(z|\mathcal{Z}, \boldsymbol{\alpha})$, we use the notion of conjugate prior [20]. In particular, we choose to represent $f(\boldsymbol{\theta}|\boldsymbol{\alpha})$ as a Dirichlet Distribution, which is the conjugate prior to the Categorical Distribution $f(z|\boldsymbol{\theta})$. With this choice, one can prove that $f(\boldsymbol{\theta}|\mathcal{Z}, \boldsymbol{\alpha})$ is also Dirichlet Distribution parameterized by a vector $\tilde{\boldsymbol{\alpha}}$:

$$\tilde{\alpha}_j = \alpha_j + \sum_{z_i \in \mathcal{Z}} 1\{z_i = j\}, \tag{6}$$

where $1\{z_i = j\}$ is equal to 1 when the expected terrain class of measurement z_i is class j and is zero otherwise [20]. By using this property in (5), one can prove [20, (3)] the

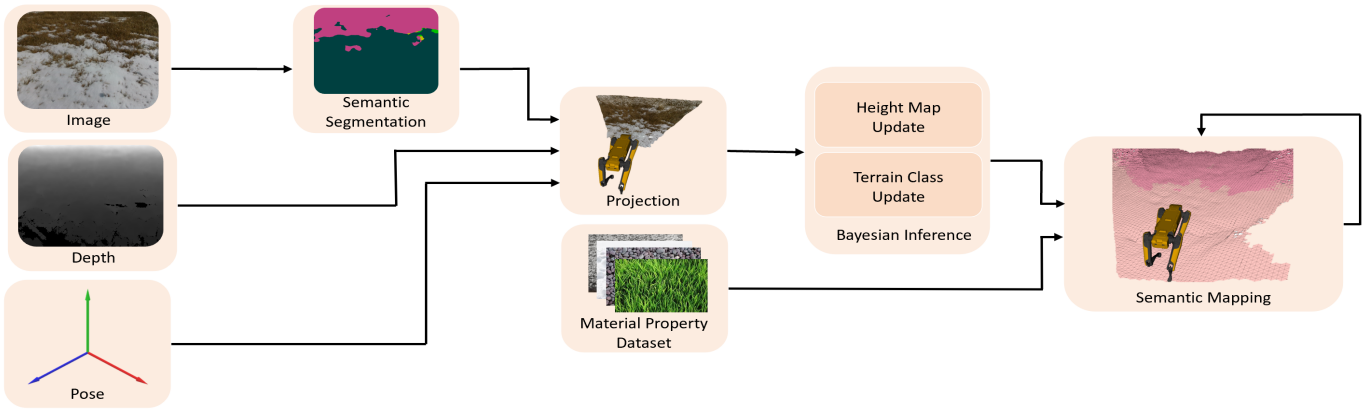

Fig. 2: A flow diagram illustrating the behavior of Algorithm 1. RGB-D images are semantically segmented using an off-the-shelf semantic segmentation network. Using the camera's estimated pose and associated depth image, the pixel-wise probabilistic terrain class estimates are projected into the map. The height map is updated using a 1D Kalman filter and the terrain class estimates, alongside our novel material property dataset, are used to recursively estimate terrain properties for each region of the map.

Algorithm 1: Recursive Semantic Mapping

Algorithm:

1 $\mathcal{G} \leftarrow$ groundPlane() `// Sec. V-A`
2 $\mathcal{V}$ collection of vertices `// Sec. V-A`
3 $\Xi \leftarrow$ triangulation($\mathcal{V}$) `// Sec. V-A`
4 **while** *robot is running* **do**
5 $I, D, \boldsymbol{q} \leftarrow$ getImageAndSensorPose()
6 $T \leftarrow$ semanticallySegmentImage(I)
7 $P_M \leftarrow$ projectImage($T, D, \boldsymbol{q}$) `// Sec. III-C`
8 assign points $\boldsymbol{p}_M \in P_M$ to $\xi \in \Xi$ `// Alg. 2`
9 $\mathcal{V} \leftarrow$ updateElevationMap($\mathcal{V}, \Xi$) `// Alg. 3`
10 $\Xi \leftarrow$ updateTerrainPrediction(P, Ξ) `// Sec. VI`

probability that a new measurement of the same region belongs to terrain class i given prior measurements $\mathcal{Z}$ is:

$$f(z = i|\mathcal{Z}, \boldsymbol{\alpha}) = \frac{\tilde{\alpha}_i}{\sum_{j=1}^{k} \tilde{\alpha}_j}. \tag{7}$$

IV. Semantic Mapping and Bayesian Inference

As illustrated in Fig. 2, this section summarizes our robot-centric semantic mapping algorithm used to estimate the terrain surface profile and properties using a triangular mesh representation given an RGB-D camera with known pose (Algorithm 1). Subsequent sections describe each step of Algorithm 1 in detail. The mesh is described using two collections. The first is the collection $\mathcal{V} \subset (\mathbb{R}^4)^m$ of vertices $\boldsymbol{v} = [v_x, v_y, v_z, v_{\sigma^2}]$, where m is the number of vertices within the mesh. The first three components of a vertex, v_x, v_y, and v_z, correspond to the Euclidean position of the vertex with respect to the mapping frame M, and the last component v_{σ^2} corresponds to the variance of v_z. The second is the collection $\Xi \subset \mathcal{V}^3 \times (\mathbb{R}^{3+k})^l \times \mathbb{R}^k_{\geq 0}$ of mesh elements, or faces, ξ. An element ξ is a collection of three components: the three vertices whose connecting line segments define the perimeter of the face, interior points, and a vector of Dirichlet parameters. The interior points are discussed in Section V-B.

We start by defining a flat ground plane $\mathcal{G}$ with zero height (Line 1). Next, vertices $\boldsymbol{v} \in \mathcal{V}$ and mesh elements $\xi \in \Xi$ are initialized (Lines 2-3, Section V-A). We retrieve the RGB-D image, I and D, and camera pose estimate in the world frame, $\boldsymbol{q}$, from the robot (Line 5). A semantic segmentation network takes the RGB image I and outputs pixel-wise terrain class probability scores, T (Line 6). These pixel-wise scores are projected into the mapping frame (Line 7, Section III-B) and assigned as interior points to a mesh element ξ (Line 8, Section V-B). Interior points are used to compute vertex heights, vertex height covariance, and terrain labels of the corresponding mesh element ξ. The height map is updated using the projected points (Line 9, Section V-C). and terrain properties are recursively updated via the Dirichlet-Categorical conjugacy relationship (Line 10, Section VI).

V. Recursive Elevation Mapping

This section describes how our algorithm recursively estimates the elevation map given range sensor measurements. We begin with the initialization of a piece-wise planar triangular mesh that represents the contact surface. Next we construct a technique to assign range sensor measurements as interior points to their corresponding triangular mesh element ξ. Finally, we describe how to update the elevation map.

A. Mesh Initialization

At startup, we define a flat ground plane $\mathcal{G}$ with zero height (Line 1) and initialize a grid pattern of evenly-spaced vertices $\boldsymbol{v} \in \mathcal{V}$ with zero height and zero variance (Line 2). The set of faces $\xi \in \Xi$ are initialized (Line 3) by triangulating these vertices into a set of equal-sized, isosceles, right-angled triangles. The set of interior points of each face is initialized as an empty set and the Dirichlet parameters are initialized as a vector of zeros.

B. Point Assignment

Interior points represent the set of points $\boldsymbol{p}_M$ whose projection lie within the 2-dimensional simplex defined by the perimeter of ξ. The process by which points are projected and assigned as interior points is described in Alg. 2. The camera on the robot collects RGB-D images that are semantically

Algorithm 2: Assign points $\boldsymbol{p}_M \in P_M$ to $\xi \in \Xi$

Requires : P_M, $\mathcal{V}$, Ξ

1 **for** $\boldsymbol{p}_M \in P_M$ **do**
2 **for** $\xi \in \Xi$ **do**
3 $\tilde{\boldsymbol{p}}_M, \tilde{\boldsymbol{v}} \leftarrow$ groundPlaneProjection($\xi, \boldsymbol{p}_M$)
4 $\boldsymbol{\lambda} \leftarrow$ computeBarycentricCoords($\tilde{\boldsymbol{p}}_M, \tilde{\boldsymbol{v}}$)
5 **if** *for all* $\lambda_i \in \boldsymbol{\lambda}, \lambda_i \in [0, 1]$ **then**
6 $\xi \leftarrow$ add interior point $\boldsymbol{p}_M$

segmented using a neural network (Section III-C) before being projected into the mapping frame as a point cloud $P_M \subset (\mathbb{R}^{3+k})^n$ (Lines 5-7, Alg. 1) made up of n points (Section III-B). The first three components of a point $\boldsymbol{p}_M \in P_M$ correspond to the Euclidean coordinates of the point in the mapping frame, while the last k components correspond to the terrain class probability score output from the semantic segmentation network. Next, we project points $\boldsymbol{p}_M \in P_M$ and the mesh vertex coordinates $\boldsymbol{v} \in \mathcal{V}$ onto the ground plane $\mathcal{G}$ by projecting $\boldsymbol{p}_M$ and $\boldsymbol{v}$ onto their first two coordinates. We obtain the projected point $\tilde{\boldsymbol{p}}_M = [p_x, p_y]$ as well as the three projected vertices $\tilde{\boldsymbol{v}}_i = [v_{xi}, v_{yi}]$ of a mesh element ξ for each $i \in \{1, 2, 3\}$. Given $\tilde{\boldsymbol{p}}$ and $\tilde{\boldsymbol{v}}$, we compute the Barycentric coordinates $\boldsymbol{\lambda} = [\lambda_1, \lambda_2, \lambda_3]$ using the following linear transform (Line 4, Alg. 2):

$$\begin{bmatrix}\lambda_1\\ \lambda_2\\ \lambda_3\end{bmatrix} = \begin{bmatrix}1 & 1 & 1\\ v_{x1} & v_{x2} & v_{x3}\\ v_{y1} & v_{y2} & v_{y3}\end{bmatrix}^{-1} \begin{bmatrix}1\\ p_x\\ p_y\end{bmatrix}. \tag{8}$$

We apply Theorem 1 to determine whether to assign a point as an interior point to mesh element ξ (Line 6, Alg. 2).

C. Elevation Map Computation

Next we describe how Algorithm 3 recursively estimates the elevation map given the range sensor measurements (Line 9, Alg. 1). These interior points from the preceding section are now used to update the elevation map.

For a vertex $\boldsymbol{v} \in \mathcal{V}$, we take the interior points from the surrounding mesh elements (Line 5, Alg. 3) and apply a 1-dimensional Kalman filter update to estimate the mean height v_z and variance v_{σ_z} of the vertex. Given the depth image used to compute the point cloud P_M has sensor noise, there is variance in the Euclidean coordinates of $\boldsymbol{p}_M \in P_M$. Once assigned to a mesh element, the elevation map depends only on the height of the points $\boldsymbol{p}_M \in P_M$, so we only consider the variance of the third Euclidean coordinate.

Recall that the third component of $\boldsymbol{p}_M$, which we denote $p_{M,3}$ describes its height. By the error propagation law [21], the variance of $p_{M,3}$ is computed (Line 7, Alg. 3):

$$\sigma^2 = J_s \Sigma_s J_s^\intercal + J_p \Sigma_p J_p^\intercal, \tag{9}$$

where Σ_s and Σ_p are the range sensor measurement noise and the sensor pose covariance matrix, respectively, and J_s and J_p, are constructed by taking the following partial derivatives:

$$J_s := \frac{\partial p_{M,3}}{\partial \boldsymbol{p}_{\mathcal{S}}} = (R_{\mathcal{S}}^{\mathcal{M}\intercal}(\boldsymbol{q}))_3 \tag{10}$$

Algorithm 3: Update Elevation Map

Requires : $\mathcal{V}$, Ξ

1 $\boldsymbol{q} \leftarrow$ cameraPose()
2 $\Sigma_s \leftarrow$ sensorNoiseModel()
3 $\Sigma_p \leftarrow$ robotPoseCovariance()
4 **for** $\boldsymbol{v} \in V$ **do**
5 $\bar{P}_M \leftarrow$ getSurroundingInteriorPoints($\boldsymbol{v}, \Xi$)
6 **for** $\bar{\boldsymbol{p}}_M \in \bar{P}_M$ **do**
7 $\bar{p}_{M,3}, \sigma^2 \leftarrow$ heightVariance($\bar{\boldsymbol{p}}_M, \boldsymbol{q}, \Sigma_s, \Sigma_p$)
8 $v_z, v_{\sigma^2} \leftarrow$ 1DKalmanFilter($\bar{p}_{M,3}, \sigma^2$)

Return : $\mathcal{V}, \Xi$

$$J_p := \frac{\partial p_{M,3}}{\partial R_{\mathcal{S}}^{\mathcal{M}}(\boldsymbol{q})} = (R_{\mathcal{S}}^{\mathcal{M}\intercal}(\boldsymbol{q}))_3 \times \boldsymbol{p}_{\mathcal{S}}, \tag{11}$$

where $(R_{\mathcal{S}}^{\mathcal{M}\intercal}(\boldsymbol{q}))_3$ denotes the third row of $R_{\mathcal{S}}^{\mathcal{M}\intercal}(\boldsymbol{q})$ and $\times$ denotes the cross product. The mean and variance of the vertex height, v_z and v_{σ^2}, are updated using a 1-dimensional Kalman filter (Line 8, Alg. 3) for all the interior points from the surrounding mesh elements:

$$v_z \leftarrow \frac{v_z \cdot \sigma^2 + z \cdot v_{\sigma^2}}{\sigma^2 + v_{\sigma^2}} \tag{12}$$

$$v_{\sigma^2} \leftarrow \frac{v_{\sigma^2} \cdot \sigma^2}{v_{\sigma^2} + \sigma^2}. \tag{13}$$

VI. Recursive Terrain Property Estimation

The objective of our semantic mapping algorithm is to estimate the distribution of terrain properties of the environment around the robot. Motivated by prior work [4], [13], we use data to construct a conditional probability distribution, $f(\boldsymbol{\psi} \mid z = i)$, of terrain property, $\boldsymbol{\psi}$, conditioned on a terrain class estimate for a region $z = i$. Using this model, given $\mathcal{Z}$ measurements of a region that is interior to ξ, we then apply the Law of Total Probability to compute this region's predicted terrain property:

$$f(\boldsymbol{\psi} \mid \mathcal{Z}, \boldsymbol{\alpha}) = \sum_{i=1}^{k} f(\boldsymbol{\psi} \mid z = i) f(z = i \mid \mathcal{Z}, \boldsymbol{\alpha}) \tag{14}$$

Note, this paper is interested in estimating the friction coefficient; however, the presented theory can be extended to other terrain properties of interest. We next discuss the components of (14) before presenting a closed-form solution for recursively estimating the coefficient of friction within our semantic map (Line 10, Alg. 1).

Following Section V-B, semantically segmented pixels $\mathcal{Z}$ are projected into the mapping frame and assigned as interior points to mesh elements ξ. Recall from Sections III-C, III-D and (2), the pixel-wise terrain class probability generated from a semantic segmentation network represent parameters $\boldsymbol{\theta}$, which are used to update $\boldsymbol{\alpha}$ via (6). For each mesh element, we compute $f(z = i \mid \mathcal{Z}, \boldsymbol{\alpha})$ using (7).

Terrain properties are not constant across a terrain class and thus should not be estimated by a single value. Rather, these properties should be modelled using a conditional probability

distribution $f(\boldsymbol{\psi} \mid z = i)$. This model is fit using data collected from each class. As we show in Section VII, we create a well-fit model by selecting an appropriate mean μ_i and variance σ_i^2 for a unimodal Gaussian distribution $f(\boldsymbol{\psi} \mid z = i) = \mathcal{N}(\mu_i, \sigma_i^2)$.

Substituting (7) and the formula for the unimodal Gaussian into (14) gives a closed-form estimate for the terrain properties within a mesh element:

$$f(\boldsymbol{\psi} \mid \mathcal{Z}, \boldsymbol{\alpha}) = \sum_{i=1}^{k} \frac{\alpha_i}{\sum_{j=1}^{k} \alpha_j} \mathcal{N}(\mu_i, \sigma_i^2). \tag{15}$$

This is a multimodal Gaussian distribution where each mode is weighted relative to the recursively updated terrain class likelihood. Note that (15) can be extended to use terrain property models other than the unimodal Gaussian distribution.

VII. Implementation

This section describes the implementation of our algorithm. Algorithm 1 is implemented in C++ and includes a Robot Operating System (ROS) interface[1]. Our implementation features noise models for the Realsense RGB-D camera and an interface to include additional sensor noise models. We evaluated our method on a desktop with a 3.1GHz Ryzen 3600 processor, 32GB of RAM and an Nvidia RTX 2080 Ti GPU.

We use (14) to estimate terrain properties from semantically segmented RGB-D images. This requires a model relating terrain class to terrain properties. The dataset published in [22] is insufficient to compute a probabilistic model as it only contains approximately three friction measurements per terrain class, and neither the Gaussian friction model proposed in [5] nor their friction data is currently publicly available. To compute a probabilistic model we introduce a novel dataset of friction measurements across ten terrain classes and make this data publicly accessible.[2] We discuss the steps for data collection and subsequent model fitting in the following paragraphs.

Our primary focus in this paper is on estimating the coefficient of friction. We built a device to measure the coefficient of friction using the pulling force measured using a load cell, the known weight of the device, and $g = 9.81 \frac{m}{s^2}$:

$$\mu = \frac{F_{pull}}{mg}. \tag{16}$$

Approximately ten thousand data samples were collected and the data was post-processed using a low-pass filter to remove measurement noise from the load cell. To model $f(\boldsymbol{\psi} \mid z = i)$, we fit the unimodal Gaussian, Weibull, and log-normal distributions, and we assessed the goodness-of-fit for each distribution using the Kolmogorov-Smirnov test [23]. The unimodal Gaussian distribution had the highest average Kolmogorov-Smirnov score across all terrain classes demonstrating that the Gaussian model generalized the best over the entire dataset. We therefore use the unimodal Gaussian model to model $f(\boldsymbol{\psi} \mid z = i)$. Table I contains the mean and variance parameters of each unimodal Gaussian distribution for each terrain class of interest.

[1]https://github.com/roahmlab/sel_map

[2]https://github.com/roahmlab/terrain_friction_dataset

Terrain Class	Coefficient of Friction Gaussian Parameters	
	μ	σ
Concrete	0.543	0.065
Grass	0.577	0.077
Pebbles	0.428	0.059
Rocks	0.478	0.113
Wood	0.372	0.055
Rubber	0.616	0.048
Rug	0.583	0.068
Snow	0.390	0.071
Ice	0.192	0.046
Laminated Flooring	0.311	0.045

TABLE I: Unimodal Gaussian parameters computed from coefficient of friction data collected across multiple terrain classes.

VIII. Results

We evaluate the performance of our mapping framework in the Carla simulation environment [24] and on a physical legged robot. In simulation, we compare our method against two baselines representing state-of-the-art terrain property estimation methods and illustrate that our method outperforms both baselines. We also demonstrate our method in real-world indoor and outdoor environments on a quadruped robot and compare it to a state-of-the-art traversability estimation method. A supplementary video demonstrates the proposed mapping framework on the Spot quadruped.

A. Computational Performance Evaluation

We ran Alg. 1 using a 1m×1m mesh and varied mesh element lengths with random input images and associated ground-truth semantic segmentations to evaluate the computational speed and memory requirements. Approximately 45-55MB of memory is required to store the mesh. With 1cm mesh element lengths, Algorithm 1 takes 527ms to run, of which the semantic segmentation network from [25] takes 477ms (Alg. 1 Lines 6-7), and the elevation map and terrain property update takes 50ms (Alg. 1 Lines 8-10). These computation times were computed by averaging across 300 trials. The computation time for semantic segmentation is network dependent, and using Fast-SCNN [26] the total computation time is reduced to approximately 200ms. A thorough evaluation of the computational times for Algorithm 1 with two semantic segmentation networks, [25] and [26], and varying mesh element lengths is given in Figure 4.

B. Simulation

We evaluate our terrain property estimation method in the Carla simulation environment [24] where ground truth terrain property information is provided on a per-class basis. The ground truth distribution for the coefficient of friction for each class is a unimodal Gaussian using the coefficient of friction models computed in Section VII. Within Carla, we collect RGB-D information from a camera mounted on the front of a car. To estimate terrain class, we use the pre-trained semantic segmentation network presented in [25] and trained on the ADE20K dataset.

We use coefficient of friction estimates to compare our method against two baselines representing the state of the art in the terrain property estimation literature. The first baseline, denoted as the *Unimodal Non-Recursive* method, estimates the

Method	KL Score (↓)	Average Precision (↑)	Average Accuracy (↑)
Uni-Modal Non-Recursive	42.3	0.59	0.58
Multi-Modal Non-Recursive	3.7	**0.99**	0.93
Ours	**2.4**	**0.99**	**0.95**

TABLE II: The Kullback-Leibler divergence scores, average precision, and average accuracy of the two baselines and our method when applied to the Carla simulation environment. Arrows depict whether a high (↑) or low (↓) score is desired. A bolded score indicates the best performing method in each criteria.

coefficient of friction by taking the most likely terrain class for a given mesh element at each instance in time and uses the unimodal Gaussian model with parameters from Table I. This baseline is representative of methods such as those presented in [12], [6], [27], which estimate the expected value of the coefficient of friction using convolutional neural networks. The second baseline, denoted as the *Multimodal Non-Recursive* method, uses the full categorical distribution of a given semantically segmented mesh element to estimate the coefficient of friction as a multi-modal Gaussian distribution. This equates to using (15) to compute the coefficient of friction directly from the pixel-wise categorical scores outputted from the semantic segmentation network. This baseline is representative of the state of the art methods [13], [5] that use the terrain class to estimate terrain properties from RGB-D images. These methods do not employ a recursive framework to update belief in terrain classifications. We ran our algorithm and the baselines offline using the data collected within Carla and compared these estimates with the ground-truth distributions using a precision-recall curve (Fig. 3) and their Kullback-Leibler divergence scores (Table II).

Note, the lower the Kullback-Leibler divergence score, the more similar two distributions are. From Table II, one can see that the first baseline performed poorly for the Carla dataset. The score for the second baseline is lower than the first's, indicating the coefficient of friction distribution estimate of the second baseline is more similar to the ground-truth distribution. Lastly, our proposed method performed best on the Kullback-Leibler divergence test and demonstrates that the coefficient of friction distribution estimated using our proposed method is the most similar to the ground-truth distribution.

The precision-recall curve summarizes the trade-off between the true positive rate and the positive predicted value and is used to evaluate the performance of a multi-class classifier. We use the average precision to evaluate the performance on the precision-recall curve as seen in Figure 3. A higher average precision indicates a more accurate classifier. For this evaluation, we divide the range of coefficient of friction values into low friction ($\mu \leq 0.5$) and high friction ($\mu > 0.5$) categories and compare the ability of our method and and the *Multimodal Non-Recursive* method to correctly predict whether a given mesh element falls within the low or high friction category. The results for *Unimodal Non-Recursive* method is omitted due to poor performance.

Table II includes the performance of all methods using the three quantitative metrics. Due to class imbalance within the

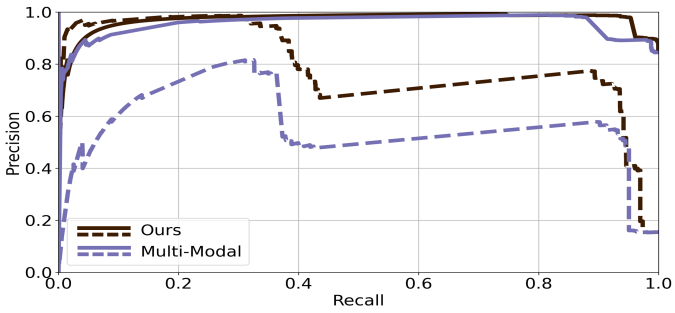

Fig. 3: The Precision-Recall curve for terrain property estimation within the Carla simulator. We compare our method against the *Multimodal Non-Recursive* baseline for regions of high friction coefficients ($\mu > 0.5$), plotted using a solid line, and low friction coefficients ($\mu \leq 0.5$), plotted using a dashed line. Our method's performance is comparable to the baseline for regions of high friction, however, for regions of low friction our method significantly outperforms the baseline.

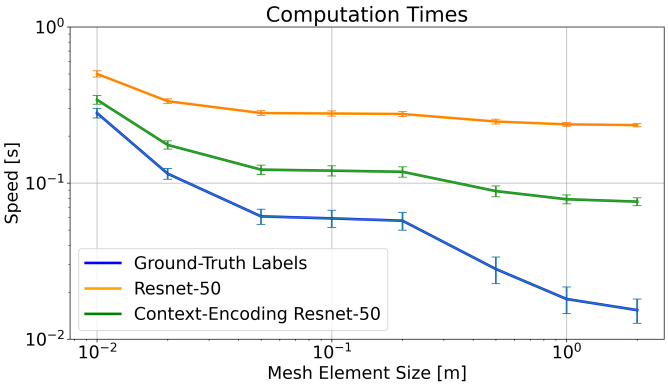

Fig. 4: Computation times of Algorithm 1 for a 10m $\times$10m mesh with varying mesh element lengths using two off-the-shelf semantic segmentation networks, Resnet-50 [25] and Context-Encoding Resnet-50 [28], as well as the baseline algorithm assuming ground-truth semantically segmented images. The ground-truth label experiments use a pre-generated semantic segmentation image, representing the speed of Algorithm 1 without considering the time required for semantic segmentation (Line 6) and represents an lower bound on the speed of our algorithm. Error bars represent one standard deviation.

simulation environment, more high-friction terrain classes are present in the data. The baselines perform better for high-friction classes, but even with this class imbalance our method matches or outperforms both baselines across all evaluation criteria. This shows our method is able to better predict the terrain friction properties than previous terrain estimation methods from the literature.

C. Real-World

We ran our method on the Spot quadruped using an on-board Realsense D435 RGB-D camera. Experiments were conducted both indoors and outdoors with a variety of terrain classes. We compared our method to a state-of-the-art traversability mapping framework [11] to demonstrate the utility of our semantic mapping algorithm when compared to a traversability estimation algorithm. Figure 5 illustrates the performance of both algorithms on a variety of examples across different terrains. On an icy surface (Fig. 5a), for instance, our method is able to predict the low friction of the surface, while the traversability estimate assumes the surface is safe to walk on and provides no additional information regarding the surface. Similarly, Figs. 5b and 5c illustrate

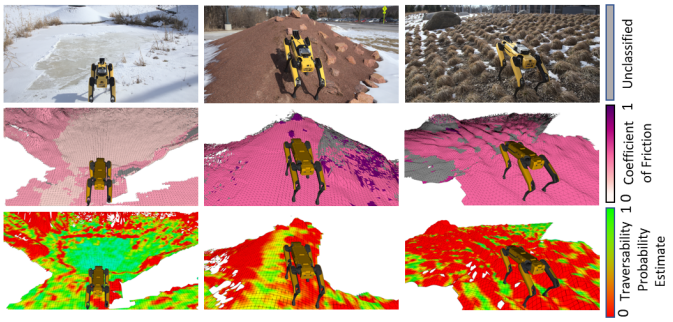

Fig. 5: Each column depicts the performance of our proposed mapping algorithm (second row, the first column uses the Context-Encoding ResNet-50 trained on the Pascal dataset while the remaining columns use the RenNet-50 trained on ADE20K for terrain classification) when compared to a traversability estimation algorithm [11] (third row) applied on the scenes depicted in the top row. Traversability estimation sometimes believes that a region is traversable when it is not, such as an icy surface (Fig. 5a). In other scenarios, it believes that an area is intraversable when it is traversable such as on hills (Figs. 5b) and near low vegetation (Fig. 5c). Our method makes no claims about traversability, instead it estimates the probability distribution of terrain properties for each mesh element along with the terrain geometry.

that the traversability estimation incorrectly classifies regions which are traversable as intraversable while our method is able to predict the terrain geometry and properties. When no terrain class from Table I is estimated within a mesh element, we make no friction estimate and color the mesh element grey (second row, Fig. 5). Traversability depends on the means of robot locomotion and other robot-dependent factors. In an effort to generalize, traversability estimation methods often over- or under-approximate traversable regions, supporting the conclusions reached by [15]. In contrast, our method provides more information than binary traversability labels which better informs robots about their environment.

IX. Conclusions

We propose a Bayesian inference framework for real-time elevation mapping and terrain property estimation from RGB-D images. Our method outperforms other algorithms both in simulation and the real-world. Unlike traversability methods, our algorithm provides terrain property information that can enable robots to adjust their locomotion to traverse regions of low friction rather than just avoid them.

References

[1] C. Mastalli, *et al.*, "Trajectory and foothold optimization using low-dimensional models for rough terrain locomotion," in *2017 IEEE International Conference on Robotics and Automation*, 2017, pp. 1096–1103.

[2] P. Fankhauser, M. Bloesch, and M. Hutter, "Probabilistic terrain mapping for mobile robots with uncertain localization," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3019–3026, 2018.

[3] M. Neunert, *et al.*, "Whole-body nonlinear model predictive control through contacts for quadrupeds," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1458–1465, 2018.

[4] T. Nguyen, F. Verdoja, F. Abu-Dakka, and V. Kyrki, "Probabilistic surface friction estimation based on visual and haptic measurements," *IEEE Robotics and Automation Letters*, vol. PP, pp. 1–1, 02 2021.

[5] D. Noh, *et al.*, "Surface material dataset for robotics applications (smdra): A dataset with friction coefficient and rgb-d for surface segmentation," in *International Conference on Pattern Recognition*, 2021, pp. 6275–6281.

[6] M. Brandão, K. Hashimoto, and A. Takanishi, "Friction from vision: A study of algorithmic and human performance with consequences for robot perception and teleoperation," in *International Conference on Humanoid Robots*, 2016, pp. 428–435.

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

[8] H. Oleynikova, *et al.*, "Signed distance fields: A natural representation for both mapping and planning," in *RSS Workshop: Geometry and Beyond-Representations, Physics, and Scene Understanding for Robotics*. University of Michigan, 2016.

[9] J. Zienkiewicz, A. Tsiotsios, A. Davison, and S. Leutenegger, "Monocular, real-time surface reconstruction using dynamic level of detail," in *International Conference on 3D Vision*, 2016, pp. 37–46.

[10] B. Kuipers and Y.-T. Byun, "A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations," *Robotics and Autonomous Systems*, vol. 8, no. 1, pp. 47–63, 1991.

[11] L. Gan, *et al.*, "Multi-task learning for scalable and dense multi-layer bayesian map inference," *ArXiv*, vol. abs/2106.14986, 2021.

[12] S. Wang, *Road Terrain Classification Technology for Autonomous Vehicle*. Springer Singapore, 01 2019.

[13] A. Angelova, L. Matthies, D. Helmick, and P. Perona, "Slip prediction using visual information," in *Robotics: Science and Systems*, 08 2006.

[14] P. Papadakis, "Terrain traversability analysis methods for unmanned ground vehicles: A survey," *Engineering Applications of Artificial Intelligence*, vol. 26, no. 4, pp. 1373–1385, 2013.

[15] D. Kim, J. Sun, S. M. Oh, J. M. Rehg, and A. F. Bobick, "Traversability classification using unsupervised on-line visual learning for outdoor robot navigation," in *International Conference on Robotics and Automation*. IEEE, 2006, pp. 518–525.

[16] J. Warren, S. Schaefer, A. Hirani, and M. Desbrun, "Barycentric coordinates for convex sets," *Adv. Comput. Math.*, vol. 27, pp. 319–338, 10 2007.

[17] J. Zhang, *et al.*, "Local barycentric coordinates," *ACM Transactions on Graphics*, vol. 33, no. 6, pp. 1–12, 2014.

[18] A. Garcia-Garcia, *et al.*, "A survey on deep learning techniques for image and video semantic segmentation," *Applied Soft Computing*, vol. 70, pp. 41–65, 2018.

[19] M. S. Nixon and A. S. Aguado, "Feature extraction & image processing for computer vision (third edition)," in *Feature Extraction & Image Processing for Computer Vision (Third Edition)*, third edition ed., M. S. Nixon and A. S. Aguado, Eds. Oxford: Academic Press, 2012, pp. 489–518.

[20] S. Tu, "The dirichlet-multinomial and dirichlet-categorical models for bayesian inference," *Computer Science Division, UC Berkeley*, vol. 2, 2014.

[21] T. Soler and J. Marshall, "Rigorous transformation of variance–covariance matrices of gps-derived coordinates and velocities," *GPS Solutions*, vol. 6, pp. 76–90, 11 2002.

[22] G. Panahandeh, E. Ek, and N. Mohammadiha, "Road friction estimation for connected vehicles using supervised machine learning," in *Intelligent Vehicles Symposium*. IEEE, 2017, pp. 1262–1267.

[23] F. J. Massey Jr, "The kolmogorov-smirnov test for goodness of fit," *Journal of the American statistical Association*, vol. 46, no. 253, pp. 68–78, 1951.

[24] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in *Conference on Robot Learning*, 2017, pp. 1–16.

[25] B. Zhou, *et al.*, "Semantic understanding of scenes through the ade20k dataset," *International Journal on Computer Vision*, 2018.

[26] R. P. K. Poudel, S. Liwicki, and R. Cipolla, "Fast-scnn: Fast semantic segmentation network," 2019.

[27] M. Procopio, J. Mulligan, and G. Grudic, "Learning terrain segmentation with classifier ensembles for autonomous robot navigation in unstructured environments," *Journal of Field Robotics*, vol. 26, pp. 145 – 175, 02 2009.

[28] H. Zhang, *et al.*, "Context encoding for semantic segmentation," in *Conference on Computer Vision and Pattern Recognition*, June 2018.