Received: 19 October 2020 | Revised: 13 April 2021 | Accepted: 15 July 2021

DOI: 10.1002/navi.445

ORIGINAL ARTICLE

WILEY ION

Data-driven protection levels for camera and 3D map-based safe urban localization

Shubh Gupta | **Grace Gao**

Stanford University

Correspondence
Grace Xingxin Gao, Stanford University.
Email: gracegao@stanford.edu

Funding information
NSF, Grant/Award Number: #2006162

Abstract

Reliably assessing the error in an estimated vehicle position is integral for ensuring the vehicle's safety in urban environments. Many existing approaches use GNSS measurements to characterize protection levels (PLs) as probabilistic upper bounds on position error. However, GNSS signals might be reflected or blocked in urban environments, and thus additional sensor modalities need to be considered to determine PLs. In this paper, we propose an approach for computing PLs by matching camera image measurements to a LiDAR-based 3D map of the environment. We specify a Gaussian mixture model probability distribution of position error using deep neural-network-based data-driven models and statistical outlier weighting techniques. From the probability distribution, we compute PL by evaluating the position error bound using numerical line-search methods. Through experimental validation with real-world data, we demonstrate that the PLs computed from our method are reliable bounds on the position error in urban environments.

KEYWORDS

deep learning, image registration, integrity monitoring, localization safety, protection level, vision-based localization

1 | INTRODUCTION

In recent years, research on autonomous navigation for urban environments has been garnering increasing attention. Many publications have targeted different aspects of navigation such as route planning (Delling et al., 2017), perception (Jensen et al., 2016), and localization (Caselitz et al., 2016; Wolcott & Eustice, 2017). For trustworthy operation in each of these aspects, assessing the level of safety of the vehicle from potential system failures is critical. However, few works have examined the problem of safety quantification for autonomous vehicles.

In the context of satellite-based localization, safety is typically addressed via integrity monitoring (IM) (Spilker Jr. et al., 1996). Within IM, protection levels (PLs) specify a statistical upper bound on the error in an estimated position of the vehicle, which can be trusted to enclose the position errors with a required probabilistic guarantee. To detect an unsafe estimated vehicle position, these protection levels are compared with the maximum allowable position error value, known as the alarm limit.

Various methods (Cezón et al., 2013; Jiang & Wang, 2016; Tran & Lo Presti, 2019) have been proposed over the years for computing protection levels, however, most of these approaches focus on GNSS-only navigation. These approaches do not directly apply to GNSS-denied urban environments, where visual sensors are becoming increasingly preferred (Badue et al., 2021). Although various options in visual sensors exist in the market, camera sensors are inexpensive, lightweight, and have been widely employed in industry. For quantifying localization safety in GNSS-denied urban

environments, there is thus a need to develop new ways of computing protection levels using camera image measurements.

Since protection levels are bounds over position error, computing them from camera image measurements requires a model that relates the measurements to position error in the estimate of the vehicle location. Furthermore, since the lateral, longitudinal, and vertical directions are well-defined with respect to a vehicle's location on the road, the model must estimate the maximum position error in each of these directions for computing protection levels (Reid et al., 2019). However, characterizing such a model is not so straightforward. This is because the relation between a vehicle location in an environment and the corresponding camera image measurement is complex, depending on identifying and matching structural patterns in the measurements with prior known information about the environment (Caselitz et al., 2016; Kim et al., 2018; Taira et al., 2021; Wolcott & Eustice, 2017).

Recently, data-driven techniques based on deep neural networks (DNN) have demonstrated state-of-the-art performance in determining the state of the camera sensor, comprised of its position and orientation, by identifying and matching patterns in images with a known map of the environment (Cattaneo et al., 2019; Lyrio et al., 2015; Oliveira et al., 2020) or an existing database of images (Sarlin et al., 2019; Taira et al., 2021).

By leveraging data sets consisting of multiple images with known camera states in an environment, these approaches can train a DNN to model the relationship between an image and the corresponding state. However, the model characterized by the DNN can often be erroneous or brittle. For instance, recent research has shown that the output of a DNN can change significantly with minimal changes to the inputs (Recht et al., 2019). Thus, for using DNNs to determine position error, uncertainty in the output of the DNN must also be addressed.

DNN-based algorithms consider two types of uncertainty (Kendall & Gal, 2017; Loquercio et al., 2020). *Aleatoric* or statistical uncertainty results from the noise present in the inputs to the DNN, because of which a precise output cannot be produced. For camera image inputs, sources of noise include illumination changes, occlusion, or the presence of visually ambiguous structures, such as windows tessellated along a wall (Kendall & Gal, 2017). On the other hand, *epistemic* or systematic uncertainty exists within the model itself. Sources of epistemic uncertainty include poorly determined DNN model parameters as well as external factors that are not considered in the model (Kiureghian & Ditlevsen, 2009), such as environmental features which might be ignored by the algorithm while matching the camera images to the environment map.

While aleatoric uncertainty is typically modeled as the input-dependent variance in the output of the DNN (Kendall & Gal, 2017; McAllister et al., 2017; Yang et al., 2020), epistemic uncertainty relates to the DNN model and, therefore, requires further deliberation. Existing approaches approximate epistemic uncertainty by assuming a probability distribution over the weight parameters of the DNN to represent ignorance about the correct parameters (Blundell et al., 2015; Gal & Ghahramani, 2016; Kendall & Cipolla, 2016).

However, these approaches assume that a correct value of the parameters exists and that the probability distribution over the weight parameters captures the uncertainty in the model, both of which do not necessarily hold up in practice (Smith & Gal, 2018). This inability of existing DNN-based methods to properly characterize uncertainty limits their applicability to safety-critical applications, such as the localization of autonomous vehicles.

In this paper, we propose a novel method for computing protection levels associated with a given vehicular state estimate (position and orientation) from camera image measurements and a 3D map of the environment. This work is based on our recent ION GNSS+ 2020 conference paper (Gupta & Gao, 2020) and includes additional experiments and improvements to the DNN training process.

Recently, high-definition 3D environment maps in the form of LiDAR point clouds have become increasingly available through industry players such as HERE, TomTom, Waymo, and NVIDIA, as well as through projects such as USGS 3DEP (Lukas & Stoker, 2016) and OpenTopography (Krishnan et al., 2011). Furthermore, LiDAR-based 3D maps are more robust to noise from environmental factors, such as illumination and weather, than image-based maps (Wang et al., 2020). Hence, we use LiDAR-based 3D point cloud maps in our approach.

Previously, CMRNet (Cattaneo et al., 2019) has been proposed as a DNN-based approach for determining the vehicular state from camera images and a LiDAR-based 3D map. In our approach, we extend the DNN architecture proposed in Cattaneo et al. (2019) to model the position error and the covariance matrix (aleatoric uncertainty) in the vehicular state estimate.

To assess the epistemic uncertainty in position error, we evaluate the DNN position error outputs at multiple candidate states in the vicinity of the state estimate, and combine the outputs into samples of the state estimate position error. Figure 1 shows the architecture of our proposed approach.

Given a state estimate, we first select multiple candidate states from its neighborhood. Using the DNN, we then evaluate the position error and covariance for each candidate state by comparing the camera image measurement

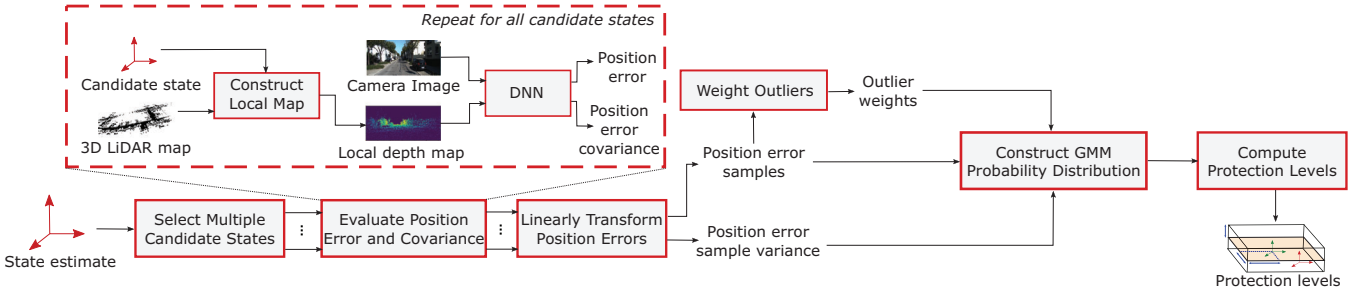

FIGURE 1 Architecture of our proposed approach for computing protection levels. Given a state estimate, multiple candidate states are selected from its neighborhood and the corresponding position error and the covariance matrix for each candidate state are evaluated using the DNN. The position errors and covariance are then linearly transformed to obtain samples of the state estimate position error and variance, which are then weighted to determine outliers. Finally, the position error samples, outlier weights, and variance are combined to construct a Gaussian mixture model probability distribution, from which the lateral, longitudinal, and vertical protection levels are computed through numerical evaluation of its probability intervals [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com and www.ion.org]

with a local map constructed from the candidate state and 3D environment map. Next, we linearly transform the position error and covariance outputs from the DNN with relative positions of candidate states into samples of the state estimate position error and variance. We then separate these samples into the lateral, longitudinal, and vertical directions and weight the samples to mitigate the impact of outliers in each direction. Subsequently, we combine the position error samples, outlier weights, and variance samples to construct a Gaussian mixture model probability distribution of the position error in each direction, and numerically evaluate its intervals to compute protection levels.

Our main contributions are as follows:

1. We extend the CMRNet (Cattaneo et al., 2019) architecture to model both the position error in the vehicular state estimate and the associated covariance matrix. Using the 3D LiDAR-based map of the environment, we first construct a local map representation with respect to the vehicular state estimate. Then, we use the DNN to analyze correspondence between the camera image measurement and the local map for determining the position error and the covariance matrix
2. We develop a novel method for capturing epistemic uncertainty in the DNN position error output. Unlike existing approaches which assume a probability distribution over DNN weight parameters, we directly analyze different position errors that are determined by the DNN for multiple candidate states selected from within a neighborhood of the state estimate. The position error outputs from the DNN corresponding to the candidate states are then linearly combined with the candidate states' relative position from the state estimate to obtain an empirical distribution of the state estimate position error
3. We design an outlier weighting scheme to account for possible errors in the DNN output at inputs that differ from the training data. Our approach weighs the position error samples from the empirical distribution using a robust outlier detection metric known as a robust Z-score (Iglewicz & Hoaglin, 1993), along the lateral, longitudinal, and vertical directions individually
4. We construct the lateral, longitudinal, and vertical protection levels as intervals over the probability distribution of the position error. We model this probability distribution as a Gaussian mixture model (Lindsay, 1995) from the position error samples, DNN covariance, and outlier weights
5. We demonstrate the applicability of our approach in urban environments by experimentally validating the protection levels computed from our method using real-world data with multiple camera images and different state estimates

The remainder of this paper is structured as follows: Section 2 discusses related work; Section 3 formulates the problem of estimating protection levels; Section 4 describes the two types of uncertainties considered in our approach; Section 5 details our algorithm; Section 6 presents the results from experimentation with real-world data; and we conclude the paper in Section 7.

2 | RELATED WORK

Several methods have been developed over the years which characterize protection levels in the context of GNSS-based urban navigation. Jiang and Wang (2016) computed horizontal protection levels using an iterative search-based method and test statistic based on the bivariate normal distribution. Cezón et al. (2013) analyzed methods which

utilize the isotropy of residual vectors from the least-squares position estimation to compute the protection levels. Tran and Lo Presti (2019) combined advanced receiver autonomous integrity monitoring (ARAIM) with Kalman filtering, and computed the protection levels by considering the set of position solutions which arise after excluding faulty measurements.

These approaches compute the protection levels by deriving the mathematical relation between measurement and position domain errors. However, such a relation is difficult to formulate with camera image measurements and a LiDAR-based 3D map, since the position error in this case depends on various factors such as the structure of buildings in the environment, available visual features, and illumination levels.

Previous works have proposed IM approaches for LiDAR- and camera-based navigation where the vehicle is localized by associating identified landmarks with a stored map or a database. Joerger and Pervan (2019) developed a method to quantify integrity risk for LiDAR-based navigation algorithms by analyzing failures of feature extraction and data association subroutines. Zhu et al. (2020) derived a bound on the integrity risk in camera-based navigation using EKF caused by incorrect feature associations.

However, these IM approaches have been developed for localization algorithms based on data association and cannot be directly applied to many recent camera and LiDAR-based localization techniques which use deep learning to model the complex relation between measurements and the stored map or database. Furthermore, these IM techniques do not estimate protection levels, which are the focus of our work.

Deep learning has been widely applied to determine position information from camera images. Kendall et al. (2015) trained a DNN using images from a single environment to learn the relationship between image and the camera 6-DOF pose. Taira et al. (2021) learned image features using a DNN to apply feature extraction and matching techniques to estimate the 6-DOF camera pose relative to a known 3D map of the environment. Sarlin et al. (2019) developed a deep learning-based 2D-3D matching technique to obtain a 6-DOF camera pose from images and a 3D environment model. However, these approaches do not model the corresponding uncertainty associated with the estimated camera pose, or account for failures in DNN approximation (Smith & Gal, 2018), which is necessary for characterizing safety measures such as protection levels.

Some recent works have proposed to estimate the uncertainty associated with deep learning algorithms. Kendall and Cipolla (2016) estimate the uncertainty in DNN-based camera pose estimation from images by evaluating the network multiple times through dropout (Gal & Ghahramani, 2016). Loquercio et al. (2020) propose a general framework for estimating uncertainty in deep learning as variance computed from both aleatoric and epistemic sources. McAllister et al. (2017) suggest using Bayesian deep learning to determine uncertainty and quantify safety in autonomous vehicles by placing probability distributions over DNN weights to represent the uncertainty in the DNN model. Yang et al. (2020) jointly estimate the vehicle odometry, scene depth, and uncertainty from sequential camera images.

However, the uncertainty estimates from these algorithms do not take into account the inaccuracy of the trained DNN model, or the influence of the underlying environment structure on the DNN outputs. In our approach, we evaluate the DNN position error outputs at inputs corresponding to multiple states in the environment, and utilize these position errors for characterizing uncertainty both from inaccuracy in the DNN model as well as from the environment structure around the state estimate.

To the best of our knowledge, our approach is the first that applies data-driven algorithms for computing protection levels by characterizing the uncertainty from different error sources. The proposed method seeks to leverage the high-fidelity function modeling capability of DNNs and combine it with techniques from robust statistics and integrity monitoring to compute robust protection levels using camera image measurements and 3D maps of the environment.

3 | PROBLEM FORMULATION

Consider the scenario of a vehicle navigating in an urban environment using measurements acquired by an onboard camera. The 3D LiDAR map of the environment $\mathcal{M}$ that consists of points $\mathbf{p} \in \mathbb{R}^3$ is assumed to be pre-known from either openly available repositories (Krishnan et al., 2011; Lukas & Stoker, 2016) or simultaneous localization and mapping algorithms (Cadena et al., 2016).

The vehicular state $\mathbf{s}_t = [\mathbf{x}_t, \mathbf{o}_t]$ at time t is a seven-element vector comprising of its 3D position $\mathbf{x}_t = [x_t, y_t, z_t]^\top \in \mathbb{R}^3$ along x, y, and z dimensions as well as 3D orientation unit quaternion $\mathbf{o}_t = [o_{1,t}, o_{2,t}, o_{3,t}, o_{4,t}] \in \mathrm{SU}(2)$. The vehicle state estimates over time are denoted as $\{\mathbf{s}_t\}_{t=1}^{T_{\max}}$ where $T_{\max}$ denotes the total time in a navigation sequence. At each time t, the vehicle captures an RGB camera image $I_t \in \mathbb{R}^{l \times w \times 3}$ from the onboard camera where l and w denote pixels along length and width dimensions, respectively.

Given an integrity risk specification IR, our objective is to compute the lateral protection level $PL_{lat,t}$, longitudinal protection level $PL_{lon,t}$, and vertical protection level $PL_{vert,t}$ at time t, which denote the maximal bounds on the

position error magnitude with a probabilistic guarantee of at least $1 - IR$. Considering x, y, and z dimensions in the rotational frame of the vehicle:

$$PL_{lat,t} = \sup\left\{\rho \mid \mathbb{P}\left(|x_t - x_t^*| \leq \rho\right) \leq 1 - IR\right\}$$
$$PL_{lon,t} = \sup\left\{\rho \mid \mathbb{P}\left(|y_t - y_t^*| \leq \rho\right) \leq 1 - IR\right\}$$
$$PL_{vert,t} = \sup\left\{\rho \mid \mathbb{P}\left(|z_t - z_t^*| \leq \rho\right) \leq 1 - IR\right\} \quad (1)$$

where $\mathbf{x}_t^* = [x_t^*, y_t^*, z_t^*]$ denotes the unknown true vehicle position at time t.

4 | TYPES OF UNCERTAINTY IN POSITION ERROR

Protection levels for a state estimate $\mathbf{s}_t$ at time t depend on the uncertainty in determining the associated position error $\Delta\mathbf{x}_t = [\Delta x_t, \Delta y_t, \Delta z_t]$ between the state estimate position $\mathbf{x}_t$ and the true position $\mathbf{x}_t^*$ from the camera image I_t and the environment map $\mathcal{M}$. We consider two different kinds of uncertainty, which are categorized by the source of inaccuracy in determining the position error $\Delta\mathbf{x}_t$: aleatoric uncertainty and epistemic uncertainty.

4.1 | Aleatoric uncertainty

Aleatoric uncertainty refers to the uncertainty from noise present in the camera image measurements I_t and the environment map $\mathcal{M}$, due to which a precise value of the position error $\Delta\mathbf{x}_t$ cannot be determined. Existing DNN-based localization approaches model the aleatoric uncertainty as a covariance matrix with only diagonal entries (Kendall & Gal, 2017; McAllister et al., 2017; Yang et al., 2020) or with both diagonal and off-diagonal terms (Liu et al., 2018; Russell & Reale, 2019).

Similar to the existing approaches, we characterize the aleatoric uncertainty by using a DNN to model the covariance matrix Σ_t associated with the position error $\Delta\mathbf{x}_t$. We consider both nonzero diagonal and off-diagonal terms in Σ_t to model the correlation between x-, y-, and z-dimension uncertainties, such as along the ground plane.

Aleatoric uncertainty by itself does not accurately represent the uncertainty in determining position error. This is because aleatoric uncertainty assumes that the noise present in training data also represents the noise in all future inputs and the DNN approximation is error-free. These assumptions fail in scenarios when the input at evaluation time is different from the training data or when the input contains features that occur rarely in the real world (Smith & Gal, 2018). Thus, relying purely on aleatoric uncertainty can lead to overconfident estimates of the position error uncertainty (Kendall & Gal, 2017).

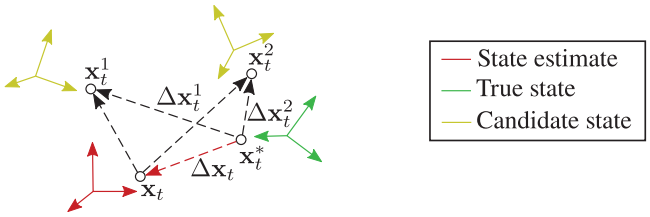

FIGURE 2 Position error $\Delta\mathbf{x}_t$ in the state estimate position $\mathbf{x}_t$ is a linear combination of the position error $\Delta\mathbf{x}_t^i$ in position $\mathbf{x}_t^i$ of any candidate state s_t^i and the relative position vector between $\mathbf{x}_t^i$ and $\mathbf{x}_t$ [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com and www.ion.org]

4.2 | Epistemic uncertainty

Epistemic uncertainty relates to the inaccuracies in the model for determining the position error $\Delta\mathbf{x}_t$. In our approach, we characterize the epistemic uncertainty by leveraging a geometrical property of the position error $\Delta\mathbf{x}_t$, where for the same camera image I_t, $\Delta\mathbf{x}_t$ can be obtained by linearly combining the position error $\Delta\mathbf{x}_t'$ computed for any candidate state $\mathbf{s}_t'$ and the relative position of $\mathbf{s}_t'$ from the state estimate $\mathbf{s}_t$ (Figure 2). Hence, using known relative positions and orientations of N_C candidate states $\{\mathbf{s}_t^1, \ldots, \mathbf{s}_t^{N_C}\}$ from $\mathbf{s}_t$, we transform the different position errors $\{\Delta\mathbf{x}_t^1, \ldots, \Delta\mathbf{x}_t^{N_C}\}$ determined for the candidate states into samples of the state estimate position error $\Delta\mathbf{x}_t$. The empirical distribution comprised of these position error samples characterizes the epistemic uncertainty in the position error estimated using the DNN.

5 | DATA-DRIVEN PROTECTION LEVELS

This section details our algorithm for computing data-driven protection levels for the state estimate $\mathbf{s}_t$ at time t, using the camera image I_t and environment map $\mathcal{M}$. First, we describe the method for generating local representations of the 3D environment map $\mathcal{M}$ with respect to the state estimate $\mathbf{s}_t$. Then, we illustrate the architecture of the DNN. Next, we discuss the loss functions used in DNN training. We then detail the method for selecting multiple candidate states from the neighborhood of the state estimate $\mathbf{s}_t$.

Using the position errors and covariance matrix evaluated from the DNN for each of these candidate states, we then illustrate the process for transforming the candidate state position errors into multiple samples of the state estimate position error. To mitigate the impact of outliers on the computed position error samples in each of the lateral, longitudinal, and vertical directions, we then detail the procedure for computing outlier weights. Next, we

characterize the probability distribution over position error in lateral, longitudinal, and vertical directions. Finally, we detail the approach for determining protection levels from the probability distribution by numerical methods.

5.1 | Local map construction

A local representation of the 3D LiDAR map of the environment captures the environment information in the vicinity of the state estimate $\mathbf{s}_t$ at time t. By comparing the environment information captured in the local map with the camera image $I_t \in \mathbb{R}^{l \times w \times 3}$ using a DNN, we estimate the position error $\Delta\mathbf{x}_t$ and covariance Σ_t in the state estimate $\mathbf{s}_t$.

For computing local maps, we utilize the LiDAR-image generation procedure described in Cattaneo et al. (2019). Similar to their approach, we generate the local map $L(\mathbf{s}, \mathcal{M}) \in \mathbb{R}^{l \times w}$ associated with vehicle state $\mathbf{s}$ and LiDAR environment map $\mathcal{M}$ in two steps.

1. First, we determine the rigid-body transformation matrix $H_\mathbf{s}$ in the special Euclidean group SE(3) corresponding to the vehicle state $\mathbf{s}$:

$$H_\mathbf{s} = \begin{bmatrix} R_\mathbf{s} & T_\mathbf{s} \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix} \in \mathrm{SE}(3) \tag{2}$$

   where
   - $R_\mathbf{s}$ denotes the rotation matrix corresponding to the orientation quaternion elements $\mathbf{o} = [o_1, o_2, o_3, o_4]$ in the state $\mathbf{s}$
   - $T_\mathbf{s}$ denotes the translation vector corresponding to the position elements $\mathbf{x} = [x, y, z]$ in the state $\mathbf{s}$

   Using the matrix $H_\mathbf{s}$, we rotate and translate the points in the map $\mathcal{M}$ to the map $\mathcal{M}_\mathbf{s}$ in the reference frame of the state $\mathbf{s}$:

$$\mathcal{M}_\mathbf{s} = \{\begin{bmatrix} I_{3\times 3} & \mathbf{0}_{3\times 1} \end{bmatrix} \cdot H_\mathbf{s} \cdot \begin{bmatrix} \mathbf{p}^\top & 1 \end{bmatrix}^\top \mid \mathbf{p} \in \mathcal{M}\} \tag{3}$$

   where I denotes the identity matrix. For maintaining computational efficiency in the case of large maps, we use the points in the LiDAR map $\mathcal{M}_\mathbf{s}$ that lie in a subregion around the state $\mathbf{s}$ and in the direction of the vehicle orientation.
2. In the second step, we apply the occlusion estimation filter presented in Pintus et al. (2011) to identify and remove occluded points along rays from the camera center. For each pair of points $(\mathbf{p}^{(i)}, \mathbf{p}^{(j)})$ where $\mathbf{p}^{(i)}$ is closer to the state $\mathbf{s}$, $\mathbf{p}^{(j)}$ is marked occluded if the angle between the ray from $\mathbf{p}^{(j)}$ to the camera center and the line from $\mathbf{p}^{(j)}$ to $\mathbf{p}^{(i)}$ is less than the threshold. Then, the remaining points are projected to the camera image frame using the camera projection matrix K to generate the local depth map $L(\mathbf{s}, \mathcal{M})$. The i-th point $\mathbf{p}^{(i)}$ in $\mathcal{M}_\mathbf{s}$ is projected as:

$$\begin{aligned} [p_x \ p_y \ c]^\top &= K \cdot \mathbf{p}^{(i)} \\ [L(\mathbf{s}, \mathcal{M})]_{(\lceil p_x/c \rceil, \lceil p_y/c \rceil)} &= [0\ 0\ 1] \cdot \mathbf{p}^{(i)} \end{aligned} \tag{4}$$

   where
   - p_x, p_y denote the projected 2D coordinates with scaling term c
   - $[L(\mathbf{s}, \mathcal{M})]_{(p_x, p_y)}$ denotes the (p_x, p_y) pixel position in the local map $L(\mathbf{s}, \mathcal{M})$

The local depth map $L(\mathbf{s}, \mathcal{M})$ for state $\mathbf{s}$ visualizes the environment features that are expected to be captured in a camera image obtained from the state $\mathbf{s}$. However, the obtained camera image I_t is associated with the true state $\mathbf{s}_t^*$ that might be different from the state estimate $\mathbf{s}_t$. Nevertheless, for reasonably small position and orientation differences between the state estimate $\mathbf{s}_t$ and true state $\mathbf{s}_t^*$, the local map $L(\mathbf{s}, \mathcal{M})$ contains features that correspond with some of the features in the camera image I_t that we use to estimate the position error.

5.2 | DNN architecture

We use a DNN to estimate the position error $\Delta\mathbf{x}_t$ and associated covariance matrix Σ_t by implicitly identifying and comparing the positions of corresponding features in camera image I_t and the local depth map $L(\mathbf{s}_t, \mathcal{M})$ associated with the state estimate $\mathbf{s}_t$.

The architecture of our DNN is given in Figure 3. Our DNN is comprised of two separate modules: one for estimating the position error $\Delta\mathbf{x}_t$ and other for the parameters of the covariance matrix Σ_t. The first module for estimating the position error $\Delta\mathbf{x}_t$ is based on CMRNet (Cattaneo et al., 2019).

CMRNet was originally proposed as an algorithm to iteratively determine the position and orientation of a vehicle using a camera image and 3D LiDAR map, starting from a provided initial state. For determining position error $\Delta\mathbf{x}_t$ using CMRNet, we use the state estimate $\mathbf{s}_t$ as the provided initial state and the corresponding DNN translation $\Delta\tilde{\mathbf{x}}_t$ and rotation $\Delta\tilde{\mathbf{r}}$ error output for transforming the state $\mathbf{s}_t$ towards the true state $\mathbf{s}_t^*$. Formally, given any state $\mathbf{s}$ and camera image I_t at time t, the translation error $\Delta\tilde{\mathbf{x}}$ and rotation error $\Delta\tilde{\mathbf{r}}$ are expressed as:

$$\Delta\tilde{\mathbf{x}}, \Delta\tilde{\mathbf{r}} = \mathrm{CMRNet}(I_t, L(\mathbf{s}, \mathcal{M})) \tag{5}$$

CMRNet estimates the rotation error $\Delta\tilde{\mathbf{r}}$ as a unit quaternion. Furthermore, the architecture determines both the translation error $\Delta\tilde{\mathbf{x}}$ and rotation error $\Delta\tilde{\mathbf{r}}$ in the reference

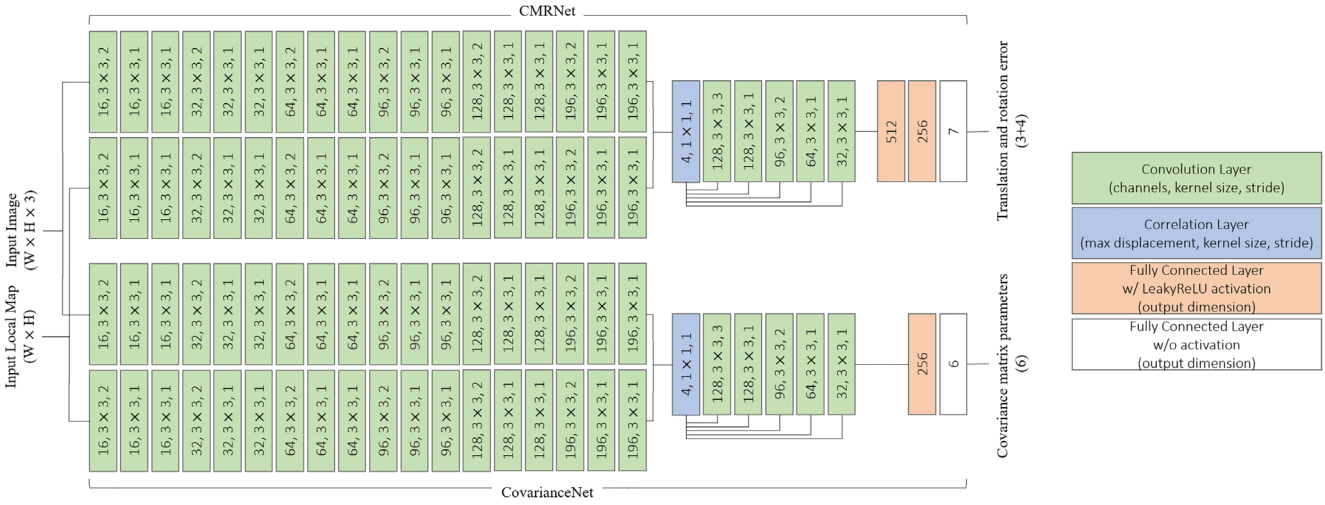

FIGURE 3 Architecture of our deep neural network (DNN) for estimating translation and rotation errors as well as parameters of the covariance matrix. The translation and rotation errors are determined using CMRNet (Cattaneo et al., 2019), and employs correlation layers (Dosovitskiy et al., 2015) for comparing feature representations of the camera image and the local depth map. Using a similar architecture, we design CovarianceNet which produces parameters of the covariance matrix associated with the translation error output [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com and www.ion.org]

frame of the state $\mathbf{s}$. Since the protection levels depend on the position error $\Delta\mathbf{x}$ in the reference frame from which the camera image I_t is captured (the vehicle reference frame), we transform the translation error $\Delta\tilde{\mathbf{x}}$ to the vehicle reference frame by rotating it with the inverse of $\Delta\tilde{\mathbf{r}}$:

$$\Delta\mathbf{x} = -\tilde{R}^{\top} \cdot \Delta\tilde{\mathbf{x}} \tag{6}$$

where $\tilde{R}$ is the 3×3 rotation matrix corresponding to the rotation error quaternion $\Delta\tilde{\mathbf{r}}$.

In the second module, we determine the covariance matrix Σ associated with $\Delta\mathbf{x}$ by first estimating the covariance matrix $\tilde{\Sigma}$ associated with the translation error $\Delta\tilde{\mathbf{x}}$ obtained from CMRNet and then transforming it to the vehicle reference frame using $\Delta\tilde{\mathbf{r}}$.

We model the covariance matrix $\tilde{\Sigma}$ by following a similar approach to Russell & Reale (2019). Since the covariance matrix is both symmetric and positive-definite, we consider the decomposition of $\tilde{\Sigma}$ into diagonal standard deviations $\boldsymbol{\sigma} = [\sigma_1, \sigma_2, \sigma_3]$ and correlation coefficients $\boldsymbol{\eta} = [\eta_{21}, \eta_{31}, \eta_{32}]$:

$$\begin{aligned} [\tilde{\Sigma}]_{ii} &= \sigma_i^2 \\ [\tilde{\Sigma}]_{ij} &= [\Sigma]_{ji} = \eta_{ij}\sigma_i\sigma_j \end{aligned} \tag{7}$$

where $i, j \in \{1, 2, 3\}$ and $j < i$. We estimate these terms using our second DNN module (referred to as CovarianceNet) which has a similar network structure as CMRNet, but with 256 and 6 artificial neurons in the last two fully connected layers to prevent overfitting.

For stable training, CovarianceNet produces a logarithm of the standard deviation output, which is converted to the standard deviation by then taking the exponent. Additionally, we use `tanh` function to scale the correlation coefficient outputs $\boldsymbol{\eta}$ in CovarianceNet between ± 1. Formally, given a vehicle state $\mathbf{s}$ and camera image I_t at time t, while the standard deviation $\boldsymbol{\sigma}$ and correlation coefficients $\boldsymbol{\eta}$ is approximated as:

$$\boldsymbol{\sigma}, \boldsymbol{\eta} = \text{CovarianceNet}(I_t, L(\mathbf{s}, \mathcal{M})) \tag{8}$$

Using the constructed $\tilde{\Sigma}$ from the obtained $\boldsymbol{\sigma}, \boldsymbol{\eta}$, we obtain the covariance matrix Σ associated with $\Delta\mathbf{x}$ as:

$$\Sigma = \tilde{R}^{\top} \cdot \tilde{\Sigma} \cdot \tilde{R} \tag{9}$$

We keep the aleatoric uncertainty restricted to position domain errors in this work for simplicity, and thus treat $\Delta\tilde{\mathbf{r}}$ as a point estimate. The impact of errors in estimating $\Delta\tilde{\mathbf{r}}$ on protection levels is taken into consideration as epistemic uncertainty and discussed in more detail in Sections 5.5 and 5.7.

The feature extraction modules in CovarianceNet and CMRNet are separate since the two tasks are complementary; for estimating position error, the DNN must learn features that are robust to noise in the inputs while the variance in the estimated position error depends on the noise itself.

5.3 | Loss functions

The loss function for training the DNN must penalize position error outputs that differ from the corresponding

ground truth present in the dataset, as well as penalize any covariance that overestimates or underestimates the uncertainty in the position error predictions. Furthermore, the loss must incentivize the DNN to extract useful features from the camera image and local map inputs for predicting the position error. Hence, we consider three additive components in our loss function $\mathcal{L}(\cdot)$:

$$\mathcal{L} = \alpha_{\text{Huber}}\mathcal{L}_{\text{Huber}}(\Delta\tilde{\mathbf{x}}^*, \Delta\tilde{\mathbf{x}}) + \alpha_{\text{MLE}}\mathcal{L}_{\text{MLE}}(\Delta\tilde{\mathbf{x}}^*, \Delta\tilde{\mathbf{x}}, \tilde{\Sigma}) + \alpha_{\text{Ang}}\mathcal{L}_{\text{Ang}}(\Delta\tilde{\mathbf{r}}^*, \Delta\tilde{\mathbf{r}}) \tag{10}$$

where:

- $\Delta\tilde{\mathbf{x}}^*, \Delta\tilde{\mathbf{r}}^*$ denotes the vector-valued translation and rotation error in the reference frame of the state estimate $\mathbf{s}$ to the unknown true state $\mathbf{s}^*$
- $\mathcal{L}_{\text{Huber}}(\cdot)$ denotes the Huber loss function (Huber, 1992)
- $\mathcal{L}_{\text{MLE}}(\cdot)$ denotes the loss function for the maximum likelihood estimation of position error $\Delta\mathbf{x}$ and covariance $\tilde{\Sigma}$
- $\mathcal{L}_{\text{Ang}}(\cdot)$ denotes the quaternion angular distance from Cattaneo et al. (2019)
- $\alpha_{\text{Huber}}, \alpha_{\text{MLE}}, \alpha_{\text{Ang}}$ are coefficients for weighting each loss term

We employ the Huber loss $\mathcal{L}_{\text{Huber}}(\cdot)$ and quaternion angular distance $\mathcal{L}_{\text{Ang}}(\cdot)$ terms from Cattaneo et al. (2019). The Huber loss term $\mathcal{L}_{\text{Huber}}(\cdot)$ penalizes the translation error output $\Delta\tilde{\mathbf{x}}$ of the DNN:

$$\mathcal{L}_{\text{Huber}}(\Delta\tilde{\mathbf{x}}^*, \Delta\tilde{\mathbf{x}}) = \sum_{X=x,y,z} D_{\text{Huber}}(\Delta\tilde{X}^*, \Delta\tilde{X})$$

$$D_{\text{Huber}}(a^*, a) = \begin{cases} \frac{1}{2}(a-a^*)^2 & \text{for } |a-a^*| \le \delta \\ \delta \cdot \left(|a-a^*| - \frac{1}{2}\delta\right) & \text{otherwise} \end{cases} \tag{11}$$

where δ is a hyperparameter for adjusting the penalty assignment to small error values. In this paper, we set $\delta = 1$. Unlike the more common mean squared error, the penalty assigned to higher error values is linear in Huber loss instead of quadratic. Thus, Huber loss is more robust to outliers and leads to more stable training as compared to squared error. The quaternion angular distance term $\mathcal{L}_{\text{Ang}}(\cdot)$ penalizes the rotation error output $\Delta\tilde{\mathbf{r}}$ from CMRNet:

$$\mathcal{L}_{\text{Ang}}(\Delta\tilde{\mathbf{r}}^*, \Delta\tilde{\mathbf{r}}) = D_{\text{Ang}}(\Delta\tilde{\mathbf{r}}^* \times \Delta\tilde{\mathbf{r}}^{-1})$$

$$D_{\text{Ang}}(\mathbf{q}) = \text{atan2}\left(\sqrt{q_2^2 + q_3^2 + q_4^2}, |q_1|\right) \tag{12}$$

where:

- q_i denotes the i-th element in quaternion $\mathbf{q}$
- $\Delta\mathbf{r}^{-1}$ denotes the inverse of the quaternion $\Delta\mathbf{r}$
- $\mathbf{q} \times \mathbf{r}$ here denotes element-wise multiplication of the quaternions $\mathbf{q}$ and $\mathbf{r}$
- atan2$(\cdot)$ is the two-argument version of the arctangent function.

Including the quaternion angular distance term $\mathcal{L}_{\text{Ang}}(\cdot)$ in the loss function incentivizes the DNN to learn features that are relevant to the geometry between the camera image and the local depth map. Hence, it provides additional supervision to the DNN training as a multi-task objective (Zeng & Ji, 2015), and is important for the stability and speed of the training process.

The maximum likelihood loss term $\mathcal{L}_{\text{MLE}}(\cdot)$ depends on both the translation error $\Delta\tilde{\mathbf{x}}$ and covariance matrix $\tilde{\Sigma}$ estimated from the DNN. The loss function is analogous to the negative log-likelihood of the Gaussian distribution:

$$\mathcal{L}_{\text{MLE}}(\Delta\tilde{\mathbf{x}}^*, \Delta\tilde{\mathbf{x}}, \tilde{\Sigma}) = \frac{1}{2}\log|\tilde{\Sigma}| + \frac{1}{2}(\Delta\tilde{\mathbf{x}}^* - \Delta\tilde{\mathbf{x}})^\top \cdot \tilde{\Sigma}^{-1} \cdot (\Delta\tilde{\mathbf{x}}^* - \Delta\tilde{\mathbf{x}}) \tag{13}$$

If the covariance output from the DNN has small values, the corresponding translation error is penalized much more than the translation error corresponding to a large valued covariance. Hence, the maximum likelihood loss term $\mathcal{L}_{\text{MLE}}(\cdot)$ incentivizes the DNN to output small covariance only when the corresponding translation error output has high confidence, and otherwise output large covariance.

5.4 | Multiple candidate state selection

To assess the uncertainty in the DNN-based position error estimation process as well as uncertainty from environmental factors, we evaluate the DNN output at N_C candidate states $\{\mathbf{s}_t^1 \dots, \mathbf{s}_t^{N_C}\}$ in the neighborhood of the state estimate $\mathbf{s}_t$.

For selecting the candidate states $\{\mathbf{s}_t^1 \dots, \mathbf{s}_t^{N_C}\}$, we randomly generate multiple values of translation offset $\{\mathbf{t}^1, \dots, \mathbf{t}^{N_C}\}$ and rotation offset $\{\mathbf{r}^1, \dots, \mathbf{r}^{N_C}\}$ about the state estimate $\mathbf{s}_t$, where N_C is the total number of selected candidate states. The i-th translation offset $\mathbf{t}^i \in \mathbb{R}^3$ denotes translation in x, y, and z dimensions and is sampled from a uniform probability distribution between a specified range $\pm t_{max}$ in each dimension.

Similarly, the i-th rotation offset $\mathbf{r}^i \in \text{SU}(2)$ is obtained by uniformly sampling between $\pm r_{max}$ angular deviations about each axis and converting the resulting rotation to

a quaternion. The i-th candidate state $\mathbf{s}_t^i$ is generated by rotating and translating the state estimate $\mathbf{s}_t$ by $\mathbf{r}^i$ and $\mathbf{t}^i$, respectively. Corresponding to each candidate state $\mathbf{s}_t^i$, we generate a local depth map $L(\mathbf{s}_t^i, \mathcal{M})$ using the procedure laid out in Section 5.1.

5.5 | Linear transformation of position errors

Using each local depth map $L(\mathbf{s}_t^i, \mathcal{M})$ and camera image I_t for the i-th candidate state $\mathbf{s}_t^i$ as inputs to the DNN in Section 5.2, we evaluate the candidate state position error $\Delta\mathbf{x}_t^i$ and covariance matrix Σ_t^i. From the known translation offset $\mathbf{t}^i$ between the candidate state $\mathbf{s}_t^i$ and the state estimate $\mathbf{s}_t$ and the DNN-based rotation error $\Delta\tilde{\mathbf{r}}_t$ in $\mathbf{s}_t$, we compute the transformation matrix $H_{\mathbf{s}_t^i \to \mathbf{s}_t}$ for converting the candidate state position error $\Delta\mathbf{x}_t^i$ to the state estimate position error $\Delta\mathbf{x}_t$ in the vehicle reference frame:

$$H_{\mathbf{s}_t^i \to \mathbf{s}_t} = \begin{bmatrix} I_{3\times3} & -\tilde{R}_t^\top \mathbf{t}^i \end{bmatrix} \tag{14}$$

where $I_{3\times3}$ denotes the identity matrix and $\tilde{R}_t$ is the 3×3 rotation matrix computed from the DNN-based rotation error $\Delta\tilde{\mathbf{r}}_t$ between the state estimate $\mathbf{s}_t$ and the unknown true state $\mathbf{s}_t^*$. Note that the rotation offset $\mathbf{r}^i$ is not used in the transformation, since we are only concerned with the position errors from the true state $\mathbf{s}_t^*$ to the state estimate $\mathbf{s}_t$, which are invariant to the orientation of the candidate state $\mathbf{s}_t^i$. Using the transformation matrix $H_{\mathbf{s}_t^i \to \mathbf{s}_t}$, we obtain the i-th sample of the state estimate position error $\Delta\mathbf{x}_t^{(i)}$:

$$\Delta\mathbf{x}_t^{(i)} = H_{\mathbf{s}_t^i \to \mathbf{s}_t} \cdot \begin{bmatrix} \Delta\mathbf{x}_t^i & 1 \end{bmatrix}^\top = \Delta\mathbf{x}_t^i - \tilde{R}_t^\top \mathbf{t}^i \tag{15}$$

We use parentheses in the notation $\Delta\mathbf{x}_t^{(i)}$ for the transformed samples of the position error between the true state $\mathbf{s}_t^*$ and the state estimate $\mathbf{s}_t$ to differentiate from the position error $\Delta\mathbf{x}_t^i$ between $\mathbf{s}_t^*$ and the candidate state $\mathbf{s}_t^i$. Next, we modify the candidate state covariance matrix Σ_t^i to account for uncertainty in DNN-based rotation error $\Delta\tilde{\mathbf{r}}_t$. The resulting covariance matrix $\Sigma_t^{(i)}$ in terms of the covariance matrix Σ_t^i for $\Delta\mathbf{x}_t^i$, $\tilde{R}_t$ and $\mathbf{t}^i$ is:

$$\Sigma_t^{(i)} = \Sigma_t^i + \mathrm{Var}\left[\tilde{R}_t^\top \mathbf{t}^i\right] \tag{16}$$

Assuming small errors in determining the true rotation offsets between state estimate $\mathbf{s}_t$ and the true state $\mathbf{s}_t^*$, we consider the random variable $R'\tilde{R}_t^\top \mathbf{t}^i$ where R' represents the random rotation matrix corresponding to small angular deviations (Barfoot et al., 2011). Using $R'\tilde{R}_t^\top \mathbf{t}^i$, we approximate the covariance matrix $\Sigma_t^{(i)}$ as:

$$\begin{aligned}
\Sigma_t^{(i)} &\approx \Sigma_t^i + \mathbb{E}\left[(R' - I)\left(\tilde{R}_t^\top \mathbf{t}^i\right)\left(\tilde{R}_t^\top \mathbf{t}^i\right)^\top (R' - I)^\top\right] \\
\left[\Sigma_t^{(i)}\right]_{i'j'} &\approx \left[\Sigma_t^i\right]_{i'j'} + \mathbb{E}\left[\left(\mathbf{r}_{i'}'\right)^\top \left(\tilde{R}_t^\top \mathbf{t}^i\right)\left(\tilde{R}_t^\top \mathbf{t}^i\right)^\top \left(\mathbf{r}_{j'}'\right)\right] \\
&= \left[\Sigma_t^i\right]_{i'j'} + \mathrm{Tr}\left(\left(\tilde{R}_t^\top \mathbf{t}^i\right)\left(\tilde{R}_t^\top \mathbf{t}^i\right)^\top \mathbb{E}\left[\left(\mathbf{r}_{i'}'\right)\left(\mathbf{r}_{j'}'\right)^\top\right]\right) \\
&= \left[\Sigma_t^i\right]_{i'j'} + \mathrm{Tr}\left(\left(\tilde{R}_t^\top \mathbf{t}^i\right)\left(\tilde{R}_t^\top \mathbf{t}^i\right)^\top Q_{i'j'}\right)
\end{aligned} \tag{17}$$

where $\left(\mathbf{r}_i'\right)^\top$ represents the i-th row vector in $R' - I$. Since errors in $\tilde{R}$ depend on the DNN output, we specify R' through the empirical distribution of the angular deviations in $\tilde{R}$ as observed for the trained DNN on the training and validation data, and precompute the expectation $Q_{i'j'}$ for each (i', j') pair.

The samples of state estimate position error $\{\Delta\mathbf{x}_t^{(1)}, \ldots, \Delta\mathbf{x}_t^{(N_C)}\}$ represent both inaccuracy in the DNN estimation as well as uncertainties due to environmental factors.

If the DNN approximation fails at the input corresponding to the state estimate $\mathbf{s}_t$, the estimated position errors at candidate states would lead to a wide range of different values for the state estimate position errors. Similarly, if the environment map $\mathcal{M}$ near the state estimate $\mathbf{s}_t$ contains repetitive features, the position errors computed from candidate states would be different and hence indicate high uncertainty.

5.6 | Outlier weights

Since the candidate states $\{\mathbf{s}_t^1 \ldots, \mathbf{s}_t^{N_C}\}$ are selected randomly, some position error samples may correspond to the local depth map and camera image pairs for which the DNN performs poorly. Thus, we compute outlier weights $\{\mathbf{w}_t^{(1)}, \ldots, \mathbf{w}_t^{(N_C)}\}$ corresponding to the position error samples $\{\Delta\mathbf{x}_t^{(1)}, \ldots, \Delta\mathbf{x}_t^{(N_C)}\}$ to mitigate the effect of these erroneous position error values in determining the protection levels.

We compute outlier weights in each of the x, y, and z dimensions separately, since the DNN approximation might not necessarily fail in all of its outputs. An example of this scenario would be when the input camera image and local map contain features such as building edges that can be used to robustly determine errors along certain directions but not others.

For computing the outlier weights $\mathbf{w}_t^{(i)} = [w_{x,t}^{(i)}, w_{y,t}^{(i)}, w_{z,t}^{(i)}]$ associated with the i-th position error

value $\Delta\mathbf{x}_t^{(i)} = [\Delta x_t^{(i)}, \Delta y_t^{(i)}, \Delta z_t^{(i)}]$, we employ the robust Z-score-based outlier detection technique (Iglewicz & Hoaglin, 1993). The robust Z-score is used in a variety of anomaly detection approaches due to its resilience to outliers (Rousseeuw & Hubert, 2018). We apply the following operations in each dimension $X = x, y$, and z:

1. We compute the median absolute deviation statistic (Iglewicz & Hoaglin, 1993) MAD_X using all position error values $\{\Delta X_t^{(1)}, \dots, \Delta X_t^{(N_C)}\}$:

$$MAD_X = \text{median}\left(|\Delta X_t^{(i)} - \text{median}\left(\Delta X_t^{(i)}\right)|\right) \quad (18)$$

2. Using the statistic MAD_X, we compute the robust Z-score $\mathcal{Z}_X^{(i)}$ for each position error value $\Delta X_t^{(i)}$:

$$\mathcal{Z}_X^{(i)} = \frac{|\Delta X_t^{(i)} - \text{median}\left(\Delta X_t^{(i)}\right)|}{MAD_X} \quad (19)$$

   The robust Z-score $\mathcal{Z}_X^{(i)}$ is high if the position error $\Delta\mathbf{x}^{(i)}$ deviates from the median error with a large value when compared with the median deviation value.
3. We compute the outlier weights $\{w_X^{(1)}, \dots, w_X^{(N_C)}\}$ from the robust Z-scores $\{\mathcal{Z}_X^{(1)}, \dots, \mathcal{Z}_X^{(N_C)}\}$ by applying the softmax operation (Goodfellow et al., 2016) such that the sum of weights is unity:

$$w_{X,t}^{(i)} = \frac{e^{-\gamma \cdot \mathcal{Z}_X^{(i)}}}{\sum_{j=1}^{N_C} e^{-\gamma \cdot \mathcal{Z}_X^{(j)}}} \quad (20)$$

   where γ denotes the scaling coefficient in the softmax function. We set $\gamma = 0.6745$ as the approximate inverse of the standard normal distribution evaluated at 3/4 to make the scaling in the statistic consistent with the standard deviation of a normal distribution (Iglewicz & Hoaglin, 1993). A small value of outlier weight $w_{X,t}^{(i)}$ indicates that the position error $\Delta X_t^{(i)}$ is an outlier.

For brevity, we extract the diagonal variances associated with each dimension for all position error samples:

$$\begin{aligned}\left(\sigma_{x,t}^2\right)^{(i)} &= \left[\Sigma_t^{(i)}\right]_{11} \\ \left(\sigma_{y,t}^2\right)^{(i)} &= \left[\Sigma_t^{(i)}\right]_{22} \\ \left(\sigma_{z,t}^2\right)^{(i)} &= \left[\Sigma_t^{(i)}\right]_{33}\end{aligned} \quad (21)$$

5.7 | Probability distribution of position error

We construct a probability distribution in each of the $X = x, y$, and z dimensions from the previously obtained samples of position errors $\Delta X_t^{(i)}$, variances $\left(\sigma_{X,t}^2\right)^{(i)}$, and outlier weights $w_{X,t}^{(i)}$. We model the probability distribution using the Gaussian mixture model (GMM) distribution (Lindsay, 1995):

$$\mathbb{P}(\rho_{X,t}) = \sum_{i=1}^{N_C} w_{X,t}^{(i)} \mathcal{N}\left(\Delta X_t^{(i)}, \left(\sigma_{X,t}^2\right)^{(i)}\right) \quad (22)$$

where:

- $\rho_{X,t}$ denotes the position error random variable
- $\mathcal{N}(\mu, \sigma^2)$ is the Gaussian distribution with mean μ and variance σ^2

The probability distributions $\mathbb{P}(\rho_{x,t})$, $\mathbb{P}(\rho_{y,t})$ and $\mathbb{P}(\rho_{z,t})$ incorporate both aleatoric uncertainty from the DNN-based covariance and epistemic uncertainty from the multiple DNN evaluations associated with different candidate states. Both the position error and covariance matrix depend on the rotation error point estimate from CMR-Net for transforming the error values to the vehicle reference frame.

Since each DNN evaluation for a candidate state estimates the rotation error independently, the epistemic uncertainty incorporates the effects of errors in DNN-based estimation of both rotation and translation. The epistemic uncertainty is reflected in the multiple GMM components and their weight coefficients, which represent the different possible position error values that may arise from the same camera image measurement and the environment map. The aleatoric uncertainty is present as the variance in each possible value of the position error is represented by the individual components.

5.8 | Protection levels

We compute the protection levels along the lateral, longitudinal, and vertical directions using the probability distributions obtained in the previous section. Since the position errors are in the vehicle reference frame, the x, y, and z dimensions coincide with the lateral, longitudinal, and the vertical directions, respectively. First, we obtain the cumulative distribution function CDF(·) for each

probability distribution:

$$\mathrm{CDF}(\rho_{X,t}) = \sum_{i=1}^{N_C} w_{X,t}^{(i)} \Phi\left(\frac{\rho_{X,t} - \Delta X_t^{(i)}}{(\sigma_{X,t})^{(i)}} \right) \tag{23}$$

where $\Phi(\cdot)$ is the cumulative distribution function of the standard normal distribution.

Then, for a specified value of the integrity risk IR, we compute the protection level PL in lateral, longitudinal, and vertical directions from Equation 1 using the CDF as the probability distribution. For numerical optimization, we employ a simple interval halving method for line search or the bisection method (Burden & Faires, 2011). To account for both positive and negative errors, we perform the optimization both using CDF (supremum) and $1 - \mathrm{CDF}$ (infemum) with $IR/2$ as the integrity risk and use the maximum absolute value as the protection level.

The computed protection levels consider heavy-tails in the GMM probability distribution of the position error that arise because of the different possible values of the position error that can be computed from the available camera measurements and environment map. Our method computes large protection levels when many different values of position error may be equally probable from the measurements, resulting in larger tail probabilities in the GMM, and small protection levels only if the uncertainty from both aleatoric and epistemic sources is small.

6 | EXPERIMENTAL RESULTS

6.1 | Real-world driving dataset

We use the KITTI visual odometry dataset (Geiger et al., 2012) to evaluate the performance of the protection levels computed by our approach. The dataset was recorded around Karlsruhe, Germany, over multiple driving sequences and contains images recorded by multiple onboard cameras, along with ground truth positions and orientations.

Additionally, the dataset contains LiDAR point cloud measurements which we use to generate the environment map corresponding to each sequence. Since our approach for computing protection levels just requires a monocular camera sensor, we use the images recorded by the left RGB camera in our experiments. We use the sequences 00, 03, 05, 06, 07, 08, and 09 from the dataset based on the availability of a LiDAR environment map. We use sequence 00 for validation of our approach and the rest of the sequences are utilized in training our DNN. The experimental parameters are provided in Table 1.

TABLE 1 Experimental parameters

Parameter	Value
Integrity risk IR	0.01
Candidate state maximum translation offset t_{max}	1.0 m
Candidate state maximum rotation offset r_{max}	5°
Number of candidate states N_C	24
Lateral alarm limit AL_{lat}	0.85 m
Longitudinal alarm limit AL_{lon}	1.50 m
Vertical alarm limit AL_{vert}	1.47 m

6.2 | LiDAR environment map

To construct a precise LiDAR point cloud map $\mathcal{M}$ of the environment, we exploit the openly available position and orientation values for the dataset computed via simultaneous localization and mapping (Caselitz et al., 2016). Similar to Cattaneo et al. (2019), we aggregate the LiDAR point clouds across all time instances. Then, we detect and remove sparse outliers within the aggregated point cloud by computing the Z-score (Iglewicz & Hoaglin, 1993) of each point in a 0.1 m local neighborhood. We discarded the points which had a higher Z-score than 3. Finally, the remaining points are down sampled into a voxel map of the environment $\mathcal{M}$ with resolution of 0.1 m. The corresponding map for sequence 00 in the KITTI dataset is shown in Figure 4. For storing large maps, we divide the LiDAR point cloud sequences into multiple overlapping parts and construct separate maps of roughly 500 megabytes each.

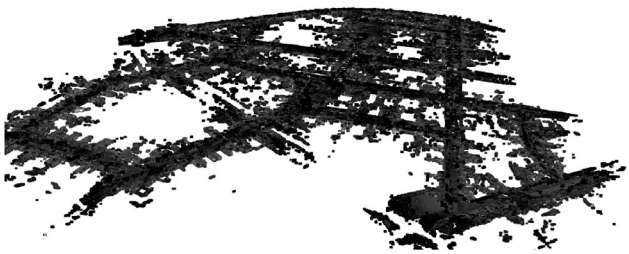

FIGURE 4 3D LiDAR environment map from KITTI dataset sequence 00 (Geiger et al., 2012)

6.3 | DNN training and testing datasets

We generate the training dataset for our DNN in two steps. First, we randomly select a state estimate s_t at time t from within a 2 m translation and a 10° rotation of the ground truth positions and orientations in each driving sequence. The translation and rotation used for generating the state estimate is utilized as the ground truth position error $\Delta\mathbf{x}_t^*$ and orientation error $\Delta\mathbf{r}_t^*$.

Then, using the LiDAR map $\mathcal{M}$, we generate the local depth map $L(\mathbf{s}_t, \mathcal{M})$ corresponding to the state estimate $\mathbf{s}_t$ and use it as the DNN input along with the camera image

I_t from the driving sequence data. The training dataset is comprised of camera images from 11,455 different time instances, with the state estimate selected at runtime so as to have different state estimates for the same camera images in different epochs.

Similar to the data augmentation techniques described in Cattaneo et al. (2019), we:

1. Randomly changed contrast, saturation, and brightness of images
2. Applied random rotations in the range of $\pm 5°$ to both the camera images and local depth maps
3. Horizontally mirrored the camera image and computed the local depth map using a modified camera projection matrix

All three of these data augmentation techniques are used in training CMRNet in the first half of the optimization process. However, for training CovarianceNet, we skip the contrast, saturation, and brightness changes during the second half of the optimization so that the DNN can learn real-world noise features from camera images.

We generate the validation and test datasets from sequence 00 in the KITTI odometry dataset, which is not used for training. We follow a similar procedure as the one for generating the training dataset, except we do not augment the data. The validation dataset comprised of randomly selected 100 time instances from sequence 00, while the test dataset contains the remaining 4,441 time instances in sequence 00.

6.4 | Training procedure

We train the DNN using stochastic gradient descent. Directly optimizing via the maximum likelihood loss term $\mathcal{L}_{\text{MLE}}(\cdot)$ might suffer from instability caused by the interdependence between the translation error $\Delta\tilde{\mathbf{x}}$ and covariance $\tilde{\Sigma}$ outputs (Skafte et al., 2019). Therefore, we employ the mean-variance split training strategy proposed in Skafte et al. (2019): First, we set $(\alpha_{\text{Huber}} = 1, \alpha_{\text{MLE}} = 1, \alpha_{\text{Ang}} = 1)$ and only optimize the parameters of CMRNet until validation error stops decreasing. Next, we set $(\alpha_{\text{Huber}} = 0, \alpha_{\text{MLE}} = 1, \alpha_{\text{Ang}} = 0)$ and optimize the parameters of CovarianceNet. We alternate between these two steps until validation loss stops decreasing.

Our DNN is implemented using the PyTorch library (Paszke et al., 2019) and takes advantage of the open-source implementation available for CMRNet (Cattaneo et al., 2019) as well as the available pre-trained weights for initialization. Similar to CMRNet, all the layers in our DNN use the leaky RELU activation function with a negative slope of 0.1. We train the DNN on using a single NVIDIA Tesla P40 GPU with a batch size of 24 and learning rate of 10^{-5} selected via grid search.

6.5 | Metrics

We evaluated the lateral, longitudinal, and vertical protection levels computed with our approach using the following three metrics (with subscript t dropped for brevity):

1. **Bound gap** measures the difference between the computed protection levels $PL_{lat}, PL_{lon}, PL_{vert}$, and the true position error magnitude during nominal operations (protection level is less than the alarm limit and greater than the position error):

$$\begin{aligned} BG_{lat} &= \text{avg}(PL_{lat} - |\Delta x^*|) \\ BG_{lon} &= \text{avg}(PL_{lon} - |\Delta y^*|) \\ BG_{vert} &= \text{avg}(PL_{vert} - |\Delta z^*|) \end{aligned} \tag{24}$$

   where:
   - BG_{lat}, BG_{lon}, and BG_{vert} denote bound gaps in lateral, longitudinal, and vertical dimensions respectively
   - avg$(\cdot)$ denotes the average computed over the test dataset for which the value of protection level is greater than the position error and less than the alarm limit

   A small bound gap value BG_{lat}, BG_{lon}, and BG_{vert} is desirable because it implies that the algorithm both estimates the position error magnitude during nominal operations accurately and has low uncertainty in the prediction. We only consider the bound gap for nominal operations since the estimated position is declared unsafe when the protection level exceeds the alarm limit.
2. **Failure rate** measures the total fraction of time instances in the test data sequence for which the computed protection levels PL_{lat}, PL_{lon}, and PL_{vert} are smaller than the true position error magnitude:

$$\begin{aligned} FR_{lat} &= \frac{1}{T_{\max}} \sum_{t=1}^{T_{\max}} \mathbb{m}1_t(PL_{lat} < |\Delta x^*|) \\ FR_{lon} &= \frac{1}{T_{\max}} \sum_{t=1}^{T_{\max}} \mathbb{m}1_t(PL_{lon} < |\Delta y^*|) \\ FR_{vert} &= \frac{1}{T_{\max}} \sum_{t=1}^{T_{\max}} \mathbb{m}1_t(PL_{vert} < |\Delta z^*|) \end{aligned} \tag{25}$$

   where:
   - FR_{lat}, FR_{lon}, and FR_{vert} denote failure rates for lateral, longitudinal, and vertical protection levels, respectively

- $\mathbb{m}1_t(\cdot)$ denotes the indicator function computed using the protection level and true position error values at time t. The indicator function evaluates to 1 if the event in its argument holds true, and otherwise evaluates to 0
- $T_{\max}$ denotes the total time duration of the test sequence

The failure rate FR_{lat}, FR_{lon}, and FR_{vert} should be consistent with the specified value of the integrity risk IR to meet the safety requirements.

3. **False alarm rate** is computed for a specified alarm limit AL_{lat}, AL_{lon}, and AL_{vert} in the lateral, longitudinal, and vertical directions and measures the fraction of time instances in the test data sequence for which the computed protection levels PL_{lat}, PL_{lon}, and PL_{vert} exceed the alarm limit AL_{lat}, AL_{lon}, and AL_{vert} while the position error magnitude is within the alarm limits. We first define the following integrity events:

$$\begin{aligned}
\Omega_{lat,PL} &= (PL_{lat} > AL_{lat}) \\
\Omega_{lat,PE} &= (|\Delta x^*| > AL_{lat}) \\
\Omega_{lon,PL} &= (PL_{lon} > AL_{lon}) \\
\Omega_{lon,PE} &= (|\Delta y^*| > AL_{lon}) \\
\Omega_{vert,PL} &= (PL_{vert} > AL_{vert}) \\
\Omega_{vert,PE} &= (|\Delta z^*| > AL_{vert})
\end{aligned} \tag{26}$$

The complement of each event is denoted by $\bar{\Omega}$. Next, we define the counts for false alarms $N_{X,FA}$, true alarms $N_{X,TA}$, and the number of times the position error exceeds the alarm limit $N_{X,PE}$ with $X = lat, lon$, and $vert$:

$$\begin{aligned}
N_{X,FA} &= \sum_{t=1}^{T_{\max}} \mathbb{m}1_t \left(\Omega_{X,PL} \cap \overline{\Omega}_{X,PE}\right) \\
N_{X,TA} &= \sum_{t=1}^{T_{\max}} \mathbb{m}1_t \left(\Omega_{X,PL} \cap \Omega_{X,PE}\right) \\
N_{X,PE} &= \sum_{t=1}^{T_{\max}} \mathbb{m}1_t \left(\Omega_{X,PE}\right)
\end{aligned} \tag{27}$$

Finally, we compute the false alarm rates FAR_{lat}, FAR_{lon}, and FAR_{vert} after normalizing the total number of position error magnitudes lying above and below the alarm limit AL:

$$FAR_X = \frac{N_{X,FA} \cdot (T_{\max} - N_{X,PE})}{N_{X,FA} \cdot (T_{\max} - N_{X,PE}) + N_{X,TA} \cdot N_{X,PE}} \tag{28}$$

6.6 | Results

Figure 5 shows the lateral and longitudinal protection levels computed by our approach on two 200 s subsets of the

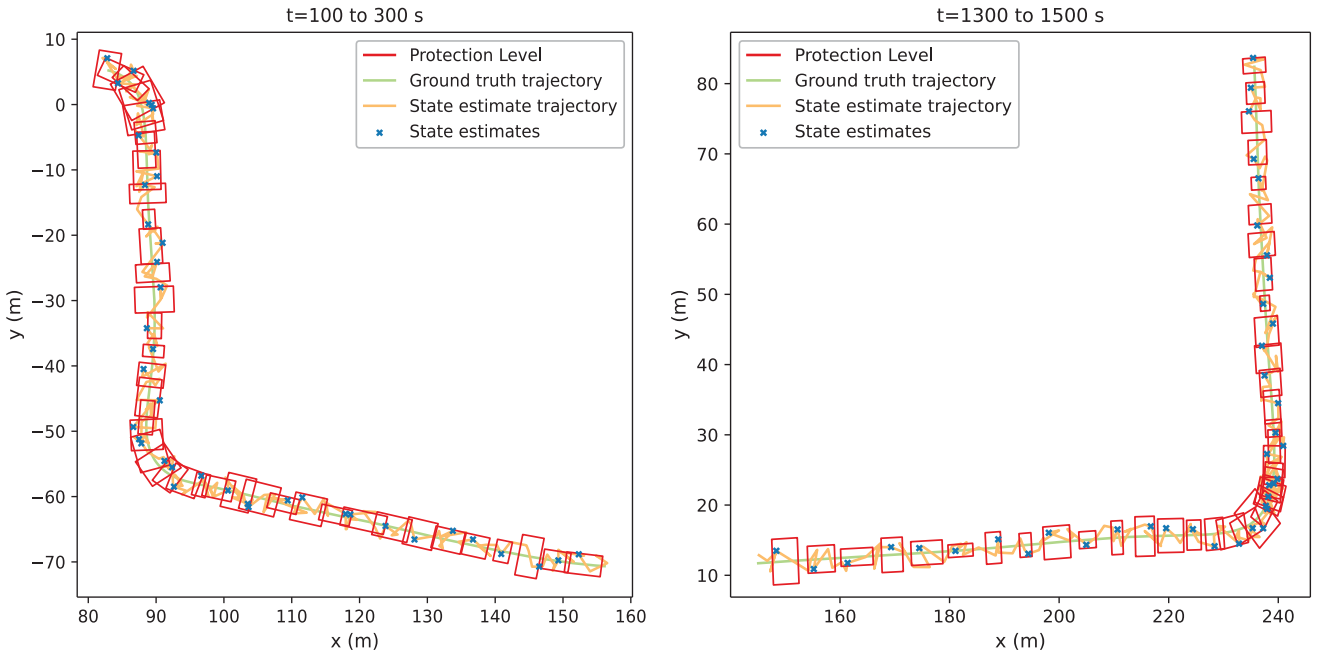

FIGURE 5 Lateral and longitudinal protection level results on the test sequence in real-world dataset. We show protection levels for two subsets of the total sequence, computed at 5 s intervals. The protection levels successfully enclose the state estimates in ~ 99% of the cases [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com and www.ion.org]

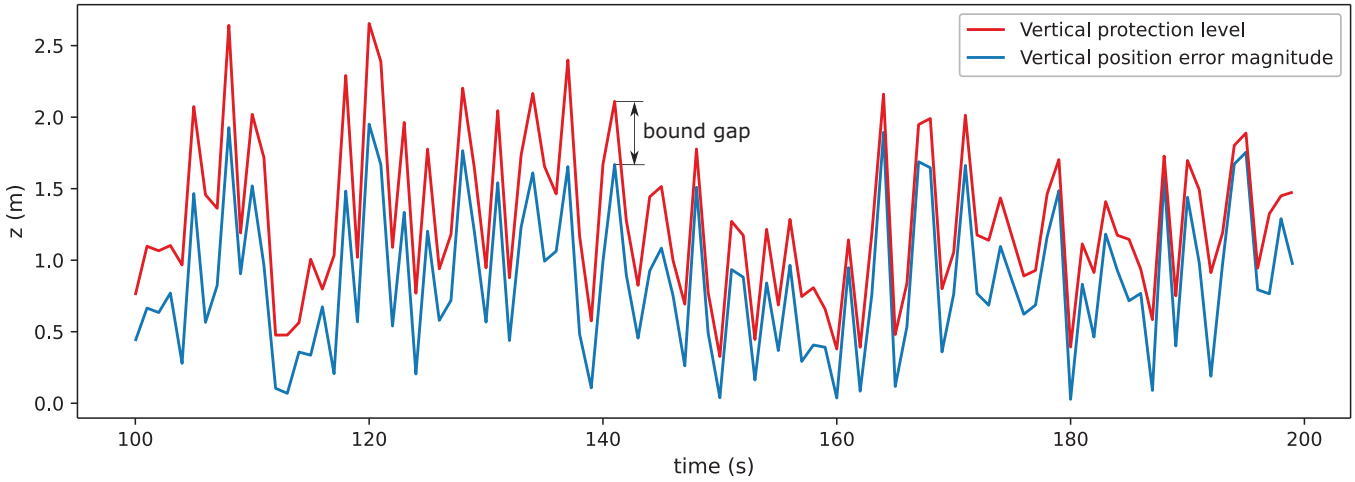

FIGURE 6 Vertical protection level results on the test sequence in real-world dataset. We show protection levels for a subset of the total sequence. The protection levels successfully enclose the position error magnitudes with a small bound gap [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com and www.ion.org]

test sequence. For clarity, protection levels are computed at every 5th time instance. Similarly, Figure 6 shows the vertical protection levels along with the vertical position error magnitude in a subset of the test sequence.

As can be seen from both the figures, the computed protection levels successfully enclose the position error magnitudes at a majority of the points ($\sim$ 99%) in the visualized subsequences. Furthermore, the vertical protection levels are observed to be visually closer to the position error as compared to the lateral and longitudinal protection levels. This is due to the superior performance of the DNN in determining position errors along the vertical dimension, which is easier to determine since all the camera images in the dataset are captured by a ground-based vehicle.

Figure 7 displays the integrity diagrams generated after the Stanford-ESA integrity diagram proposed for SBAS integrity (Tossaint et al., 2007). The diagram is generated from 15,000 samples of protection levels corresponding to randomly selected state estimates and camera images within the test sequence.

For protection levels in each direction, we set the alarm limit (Table 1) based on the specifications suggested for mid-size vehicles in Reid et al. (2019), beyond which the state estimate is declared unsafe to use. The lateral, longitudinal, and vertical protection levels are greater than the position error magnitudes in $\sim$ 99% cases, which is consistent with the specified integrity requirement. Furthermore, a large fraction of the failures is in the region where the protection level is greater than the alarm limit and thus the system has been correctly identified to be under unsafe operation.

We conducted an ablation study to numerically evaluate the impact of our proposed epistemic uncertainty measure and outlier weighting method in computing protection levels. We evaluated protection levels in three different cases: Incorporating DNN covariance, epistemic uncertainty, and outlier weighting (VAR+EO); incorporating just the DNN covariance and epistemic uncertainty with equal weights assigned to all position error samples (VAR+E); and only using the DNN covariance (VAR).

For VAR, we constructed a Gaussian distribution using the DNN position error output and diagonal variance entries in each dimension. Then, we computed protection levels from the inverse cumulative distribution function of the Gaussian distribution corresponding to the specified value of integrity risk IR. Table 2 summarizes our results.

Incorporating the epistemic uncertainty in computing protection levels improved the failure rate from 0.05 in lateral protection levels, 0.05 in longitudinal protection levels, and 0.03 in vertical protection levels to within 0.01 in all cases. This is because the covariance estimate from the DNN provides an overconfident measure of uncertainty, which is corrected by our epistemic uncertainty measure. Furthermore, incorporating outlier weighting reduced the average nominal bound gap by about 0.02 m in lateral protection levels, 0.05 m in longitudinal protection levels, and 0.05 m in vertical protection levels as well as false alarm rate by about 0.02 for each direction while keeping the failure rate within the specified integrity risk requirement.

The mean bound gap between the lateral protection levels computed from our approach and the position error magnitudes in the nominal cases is smaller than a quarter of the width of a standard US lane. In the longitudinal direction, the bound gap is somewhat larger since fewer visual features are present along the road for determining the position error using the DNN. The corresponding value in the vertical dimension is smaller, owing to the DNN's superior performance in determining position errors and

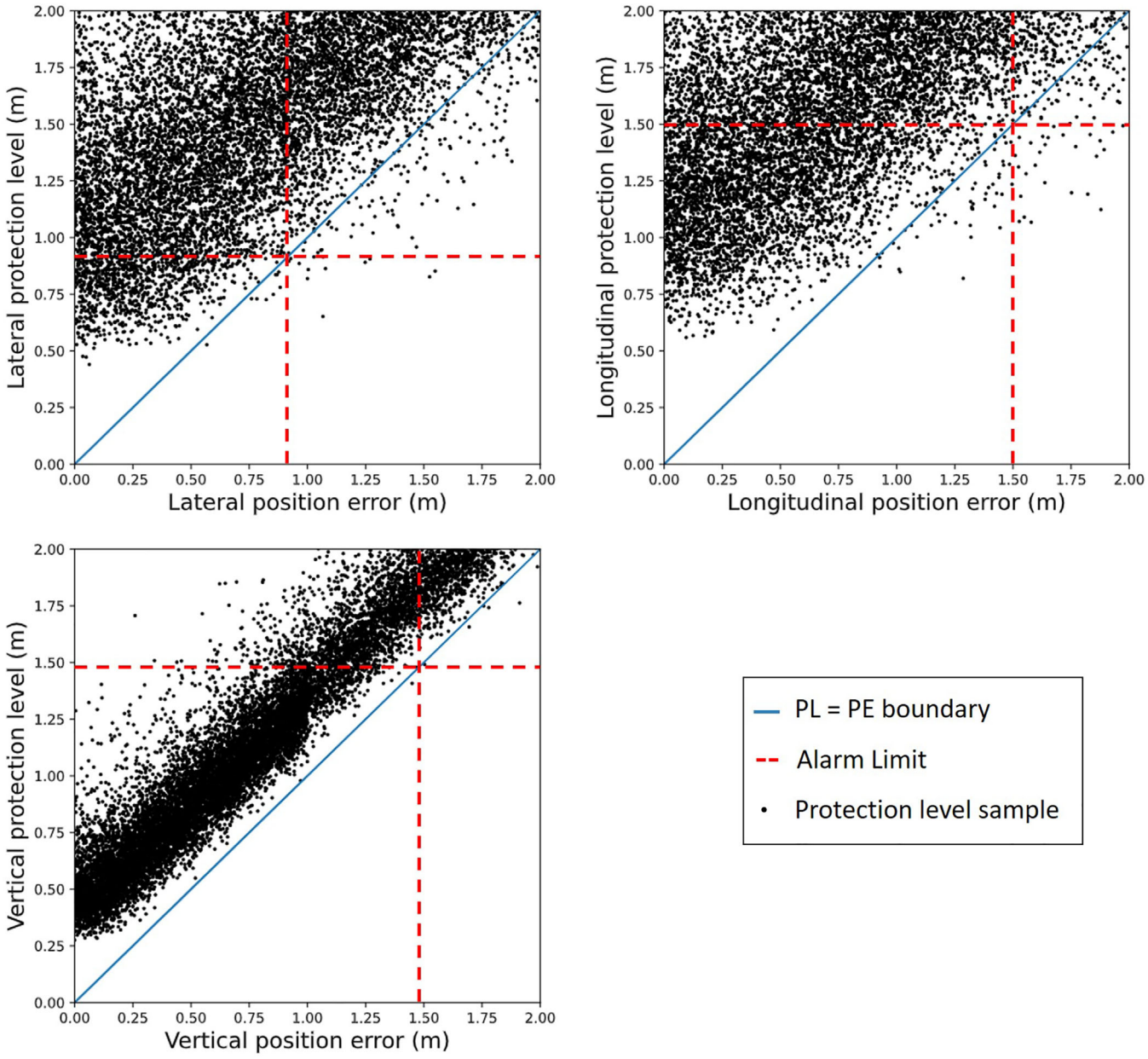

FIGURE 7 Integrity diagram results for the lateral, longitudinal, and vertical protection levels. The diagram contains protection levels evaluated across 15,000 different state estimates and camera images randomly selected from the test sequence. A majority of the samples are close to and greater than the position error magnitude, validating the applicability of the computed protection levels as a robust safety measure [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com and www.ion.org]

TABLE 2 Evaluation of lateral, longitudinal, and vertical protection levels from our approach. We compare protection levels computed by our trained model using DNN covariance, epistemic uncertainty, and outlier weighting (VAR+EO); DNN covariance and epistemic uncertainty (VAR+E); and only using the DNN covariance (VAR). Incorporating epistemic uncertainty results in lower failure rate while incorporating outlier weights reduces bound gap and false alarm rate

	Lateral PL			**Longitudinal PL**			**Vertical PL**		
	***BG*(m)**	***FR***	***FAR***	***BG*(m)**	***FR***	***FAR***	**BG(*m*)**	***FR***	***FAR***
VAR+EO	0.49	0.01	0.47	0.77	0.01	0.40	0.38	< 0.01	0.14
VAR+E	0.51	0.01	0.49	0.82	0.01	0.43	0.43	< 0.01	0.16
VAR	0.42	0.05	0.45	0.64	0.05	0.36	0.30	0.02	0.12

uncertainty in the vertical dimension. This demonstrates the applicability of our approach to urban roads.

For an integrity risk requirement of 0.01, the protection levels computed by our method demonstrate a failure rate equal to or within 0.01 as well. However, further lowering the integrity risk requirement during our experiments either did not similarly improve the failure rate or caused a significant increase in the bound gaps and the false alarm rate.

A possible reason is that the uncertainty approximated by our approach through both the aleatoric and epistemic measures fails to act as an accurate uncertainty representation for smaller integrity risk requirements than 0.01. Future research would consider more and varied training data, better strategies for selecting candidate states, and different DNN architectures to meet smaller integrity risk requirements.

A shortcoming of our approach is the large false alarm rate exhibited by the computed protection levels shown in Table 2. The large value results both from the inherent noise in the DNN-based estimation of position and rotation error as well as from frequently selecting candidate states that result in large outlier error values. A future work direction for reducing the false alarm rate is to explore strategies for selecting candidate states and mitigating outliers.

A key advantage offered by our approach is its application to scenarios where a direct analysis of the error sources in the state estimation algorithm is difficult, such as when feature rich visual information is processed by a machine learning algorithm for estimating the state. In such scenarios, our approach computes protection levels separately from the state estimation algorithm by both evaluating a data-driven model of the position error uncertainty and characterizing the epistemic uncertainty in the model outputs.

7 | CONCLUSION

In this work, we presented a data-driven approach for computing lateral, longitudinal, and vertical protection levels associated with a given state estimate from camera images and a 3D LiDAR map of the environment. Our approach estimates both aleatoric and epistemic measures of uncertainty for computing protection levels, thereby providing robust measures of localization safety.

We demonstrated the efficacy of our method on real-world data in terms of bound gap, failure rate, and false alarm rate. Results show that the lateral, longitudinal, and vertical protection levels computed from our method enclose the position error magnitudes with 0.01 probability of failure and less than 1 m bound gap in all directions, which demonstrates that our approach is applicable to GNSS-denied urban environments.

ACKNOWLEDGEMENT

This material is based upon work supported by the National Science Foundation under award #2006162.

ORCID

Grace Gao https://orcid.org/0000-0002-1807-8637

REFERENCES

Badue, C., Guidolini, R., Carneiro, R. V., Azevedo, P., Cardoso, V. B., Forechi, A., Jesus, L., Berriel, R., Paixão, T. M., Mutz, F., de Paula Veronese, L., Oliveira-Santos, T., & De Souza, A. F. (2021). Self-driving cars: A survey. *Expert Systems with Applications*, *165*. https://doi.org/10.1016/j.eswa.2020.113816

Barfoot, T., Forbes, J. R., & Furgale, P. T. (2011). Pose estimation using linearized rotations and quaternion algebra. *Acta Astronautica*, *68*(1-2), 101–112. ISSN 00945765. https://doi.org/10.1016/j.actaastro.2010.06.049

Blundell, C., Cornebise, J., Kavukcuoglu, K. & Wierstra, D. (2015). Weight Uncertainty in Neural Network. In F. Bach, & D. Blei (Eds.) *Proc. of the 32nd International Conference on Machine Learning*, vol. 37 of Proceedings of Machine Learning Research (pp. 1613–1622). Lille, France: PMLR.

Burden, R. L., & Faires, J. D. (2011). *Numerical Analysis*. Cengage Learning.

Cadena, C., Carlone, L., Carrillo, H., Latif, Y., Scaramuzza, D., Neira, J., Reid, I., & Leonard, J. J. (2016). Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Transactions on Robotics*, *32*(6), 1309–1332. https://doi.org/10.1109/TRO.2016.2624754. ArXiv: 1606.05830.

Caselitz, T., Steder, B., Ruhnke, M. & Burgard, W. (2016). Monocular camera localization in 3D LiDAR maps. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Daejeon, South Korea: IEEE, 1926-1931. https://doi.org/10.1109/IROS.2016.7759304

Cattaneo, D., Vaghi, M., Ballardini, A. L., Fontana, S., Sorrenti, D. G. & Burgard, W. (2019). CMRNet: Camera to LiDAR-Map Registration. *2019 IEEE Intelligent Transportation Systems Conference (ITSC)* (pp. 1283–1289). https://doi.org/10.1109/ITSC.2019.8917470

Cezón, A., Cueto, M. & Fernández, I. (2013). Analysis of Multi-GNSS Service Performance Assessment: ARAIM vs. IBPL Performances Comparison. *Proc. of the 26th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2013)*, Nashville, TN, 2654–2663. https://www.ion.org/publications/abstract.cfm?articleID=11407

Delling, D., Goldberg, A. V., Pajor, T., & Werneck, R. F. (2017). Customizable Route Planning in Road Networks. *Transportation Science*, *51*(2), 566–591. https://doi.org/10.1287/trsc.2014.0579

Dosovitskiy, A., Fischer, P., Ilg, E., Hausser, P., Hazirbas, C., Golkov, V., Smagt, P. v. d., Cremers, D. & Brox, T. (2015). FlowNet: Learning Optical Flow with Convolutional Networks. *2015 IEEE International Conference on Computer Vision (ICCV)*, Santiago, 2758–2766. https://doi.org/10.1109/ICCV.2015.316

Gal, Y. & Ghahramani, Z. (2016). Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. In M. F. Balcan, & K. Q. Weinberger (Eds.) *Proc. of The 33rd International Conference on Machine Learning*, Proceedings of Machine Learning Research Vol. 48, New York, NY, 1050–1059. https://dl.acm.org/doi/10.5555/3045390.3045502

Geiger, A., Lenz, P. & Urtasun, R. (2012). Are we ready for autonomous driving? The KITTI vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, Providence, RI, 3354-3361. https://doi.org/10.1109/CVPR.2012.6248074

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press.

Gupta, S. & Gao, G. X. (2020). Data-Driven Protection Levels for Camera and 3D Map-based Safe Urban Localization. *Proc. of the 33rd International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2020)*, 2483–2499. https://doi.org/10.33012/2020.17698

Huber, P. J. (1992). Robust Estimation of a Location Parameter. In S. Kotz, & N. L. Johnson (Eds.) *Breakthroughs in Statistics*. Springer Series in Statistics, New York, NY: 492–518. https://doi.org/10.1007/978-1-4612-4380-9_35

Iglewicz, B., & Hoaglin, D. C. (1993). *How to Detect and Handle Outliers*. The ASQC Basic References in Quality Control: Statistical Techniques ASQC Quality Press.

Jensen, M. B., Philipsen, M. P., Mogelmose, A., Moeslund, T. B., & Trivedi, M. M. (2016). Vision for Looking at Traffic Lights: Issues, Survey, and Perspectives. *IEEE Transactions on Intelligent Transportation Systems*, *17*(7), 1800–1815. https://doi.org/10.1109/TITS.2015.2509509

Jiang, Y., & Wang, J. (2016). A New Approach to Calculate the Horizontal Protection Level. *The Journal of Navigation*, *69*(1), 57–74. https://doi.org/10.1017/S0373463315000545

Joerger, M., & Pervan, B. (2019). Quantifying Safety of Laser-Based Navigation. *IEEE Transactions on Aerospace and Electronic Systems*, *55*(1), 273–288. IEEE Transactions on Aerospace and Electronic Systems. https://doi.org/10.1109/TAES.2018.2850381

Kendall, A. & Cipolla, R. (2016). Modelling uncertainty in deep learning for camera relocalization. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, Stockholm, Sweden, 4762–4769. https://doi.org/10.1109/ICRA.2016.7487679

Kendall, A. & Gal, Y. (2017). What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision? https://arxiv.org/abs/1703.04977

Kendall, A., Grimes, M. & Cipolla, R. (2015). PoseNet: A Convolutional Network for Real-Time 6-DOF Camera Relocalization. *2015 IEEE International Conference on Computer Vision (ICCV)*, Santiago, Chile, 2938-2946. https://doi.org/10.1109/ICCV.2015.336

Kim, Y., Jeong, J. & Kim, A. (2018). Stereo Camera Localization in 3D LiDAR Maps. *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Madrid, Spain, 1-9. https://doi.org/10.1109/IROS.2018.8594362

Kiureghian, A. D., & Ditlevsen, O. (2009). Aleatory or epistemic? Does it matter? *Structural Safety*, *31*(2), 105–112. https://doi.org/10.1016/j.strusafe.2008.06.020

Krishnan, S., Crosby, C., Nandigam, V., Phan, M., Cowart, C., Baru, C. & Arrowsmith, R. (2011). OpenTopography: a services oriented architecture for community access to LIDAR topography. *Proc. of the 2nd International Conference on Computing for Geospatial Research & Applications (COM.Geo '11)*, Washington, DC, 1–8. https://doi.org/10.1145/1999320.1999327

Lindsay, B. G. (1995). *Mixture Models: Theory, Geometry, and Applications*. NSF-CBMS Regional Conference Series in Probability and Statistics IMS.

Liu, K., Ok, K., Vega-Brown, W. & Roy, N. (2018). Deep Inference for Covariance Estimation: Learning Gaussian Noise Models for State Estimation. *2018 IEEE International Conference on Robotics and Automation (ICRA)*, Brisbane, QLD, 1436-1443, https://doi.org/10.1109/ICRA.2018.8461047

Loquercio, A., Segu, M., & Scaramuzza, D. (2020). A General Framework for Uncertainty Estimation in Deep Learning. *IEEE Robotics and Automation Letters*, *5*(2), 3153–3160. https://doi.org/10.1109/LRA.2020.2974682

Lukas, V. & Stoker, J. M. (2016). 3D Elevation Program—Virtual USA in 3D. US Geological Survey, Reston, VA. https://doi.org/10.3133/fs20163022

Lyrio, L. J., Oliveira-Santos, T., Badue, C. & De Souza, A. F. (2015). Image-based mapping, global localization and position tracking using VG-RAM weightless neural networks. *2015 IEEE International Conference on Robotics and Automation (ICRA)*, Seattle, WA, 3603–3610. https://doi.org/10.1109/ICRA.2015.7139699

McAllister, R., Gal, Y., Kendall, A., van der Wilk, M., Shah, A., Cipolla, R. & Weller, A. (2017). Concrete Problems for Autonomous Vehicle Safety: Advantages of Bayesian Deep Learning. *Proc. of the 26th International Joint Conference on Artificial Intelligence*, Melbourne, Australia, 4745–4753. https://doi.org/10.24963/ijcai.2017/661

Oliveira, G. L., Radwan, N., Burgard, W., & Brox, T. (2020). Topometric Localization with Deep Learning. In N. M. Amato, G. Hager, S. Thomas, & M. Torres-Torriti (Eds.) *Robotics Research Vol. 10*. The 18th International Symposium ISRR: Springer Proceedings in Advanced Robotics. https://doi.org/10.1007/978-3-030-28619-4_38

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., … Chintala, S. (2019). PyTorch: An Imperative Style, High-Performance Deep Learning Library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d. Alché-Buc, E. Fox, & R. Garnett (Eds.) *Advances in Neural Information Processing Systems, vol. 32*. Curran Associates, Inc.

Pintus, R., Gobbetti, E. & Agus, M. (2011). Real-time rendering of massive unstructured raw point clouds using screen-space operators. *Proc. of the 12th International conference on Virtual Reality*, Archaeology and Cultural Heritage, 105–112. https://dl.acm.org/doi/10.5555/2384495.2384513

Recht, B., Roelofs, R., Schmidt, L. & Shankar, V. (2019). Do ImageNet Classifiers Generalize to ImageNet? In K. Chaudhuri, & R. Salakhutdinov (Eds.) *Proc. of the 36th International Conference on Machine Learning*, Vol. 97 of Proceedings of Machine Learning Research, 5389–5400.

Reid, T. G. R., Houts, S. E., Cammarata, R., Mills, G., Agarwal, S., Vora, A., & Pandey, G. (2019). Localization Requirements for Autonomous Vehicles. *SAE International Journal of Connected and Automated Vehicles*, *2*(3), 173–190. https://doi.org/10.4271/12-02-03-0012.

Rousseeuw, P. J., & Hubert, M. (2018). Anomaly detection by robust statistics. *WIREs Data Mining and Knowledge Discovery*, *8*(2). https://doi.org/10.1002/widm.1236

Russell, R. L. & Reale, C. (2021). Multivariate Uncertainty in Deep Learning. https://arxiv.org/pdf/1910.14215.pdf

Sarlin, P., Cadena, C., Siegwart, R. & Dymczyk, M. (2019). From Coarse to Fine: Robust Hierarchical Localization at Large Scale. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 12708–12717. https://doi.org/10.1109/CVPR.2019.01300

Skafte, N., Jørgensen, M. & Hauberg, S. r. (2019) Reliable training and estimation of variance networks. *33rd Conference on Neural Information Processing Systems (NeurIPS 2019)*, Vancouver, Canada, 6326–6336.

Smith, L. & Gal, Y. (2018). Understanding Measures of Uncertainty for Adversarial Example Detection. https://arxiv.org/pdf/1803.08533.pdf

Spilker Jr., J. J., Axelrad, P., Parkinson, B. W., & Enge, P. (Eds.) (1996). *Global Positioning System: Theory and Applications, Volume I*. Washington, DC, American Institute of Aeronautics and Astronautics. https://doi.org/10.2514/4.866388

Taira, H., Okutomi, M., Sattler, T., Cimpoi, M., Pollefeys, M., Sivic, J., Pajdla, T., & Torii, A. (2021). InLoc: Indoor Visual Localization with Dense Matching and View Synthesis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *43*(4), 1293–1307. https://doi.org/10.1109/TPAMI.2019.2952114

Tossaint, M., Samson, J., Toran, F., Ventura-Traveset, J., Hernandez-Pajares, M., Juan, J., Sanz, J., & Ramos-Bosch, P. (2007). The Stanford - ESA Integrity Diagram: A New Tool for The User Domain SBAS Integrity Assessment. *NAVIGATION*, *54*(2), 153–162. https://doi.org/10.1002/j.2161-4296.2007.tb00401.x

Tran, H. T., & Lo Presti, L. (2019). Kalman filter-based ARAIM algorithm for integrity monitoring in urban environment. *ICT Express*, *5*(1), 65–71. https://doi.org/10.1016/j.icte.2018.05.002

Wang, C., Wen, C., Dai, Y., Yu, S., & Liu, M. (2020). Urban 3D modeling with mobile laser scanning: a review. *Virtual Reality & Intelligent Hardware*, *2*(3), 175–212. https://doi.org/10.1016/j.vrih.2020.05.003

Wolcott, R. W., & Eustice, R. M. (2017). Robust LIDAR localization using multiresolution Gaussian mixture maps for autonomous driving. *The International Journal of Robotics Research*, *36*(3), 292–319. https://doi.org/10.1177/0278364917696568

Yang, N., Stumberg, L., Wang, R. & Cremers, D. (2020). D3VO: Deep Depth, Deep Pose and Deep Uncertainty for Monocular Visual Odometry. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Seattle, WA, 1278–1289. https://doi.org/10.1109/CVPR42600.2020.00136

Zeng, T. & Ji, S. (2015). Deep Convolutional Neural Networks for Multi-instance Multi-task Learning. *2015 IEEE International Conference on Data Mining*, Atlantic City, NJ, 579–588. https://doi.org/10.1109/ICDM.2015.92

Zhu, C., Joerger, M. & Meurer, M. (2020, April). Quantifying Feature Association Error in Camera-based Positioning. In *2020 IEEE/ION Position, Location and Navigation Symposium (PLANS)* (pp. 967–972). ISSN: 2153-3598. https://doi.org/10.1109/PLANS46316.2020.9109919

How to cite this article: Gupta S, Gao G. Data-driven protection levels for camera and 3D map-based safe urban localization. *NAVIGATION*. 2021;*68*:643–660. https://doi.org/10.1002/navi.445