Uncertainty Quantification of Neural Radiance Fields for Enhanced Safety Validation

Jacob Frausto
Stanford University
Stanford, CA 94305
jfrausto@stanford.edu

Harrison Delecki*
Stanford University
Stanford, CA 94305
hdelecki@stanford.edu

Mykel J. Kochenderfer*
Stanford University
Stanford, CA 94305
mykel@stanford.edu

Abstract

This work aims to augment the safety validation of autonomous systems by integrating Uncertainty Quantification (UQ) of Neural Radiance Fields (NeRFs) into a safety validation framework and dynamics simulator. By quantifying the uncertainty of the model, we aspire to provide a more granular understanding of the surrogate model's performance and behavior guiding a simulated quadcopter agent, thereby informing decision-making in safety-critical scenarios. Our approach involves two methods: a Gaussian Approximation and a Bayesian Laplace Approximation, both of which estimate the uncertainty in volumetric density predictions. The data for this approach are disturbance vectors that lead to failure modes, generated through sampling and optimization algorithms within our simulation framework. Despite some limitations, the results indicate that incorporating uncertainty into a comprehensive safety validation framework can enhance its robustness.

1. Introduction

The input to our approach is a 3D scene represented as a Neural Radiance Field (NeRF), along with associated color values, density values, and rendered color. We then use a Gaussian approximation and a Bayesian Laplace approximation to model the uncertainty in the density values and the parameters of the NeRF model, respectively. The output of our approach is a quantification of this uncertainty, which can be used to inform the sampling of disturbance vectors in safety validation. This allows us to more thoroughly test the safety of autonomous systems in various scenarios, taking into account the uncertainty in the scene representation.

1.1. Neural Radiance Fields

The advent of Neural Radiance Fields has revolutionized the field of 3D scene representation, demonstrating remarkable capabilities in applications such as view synthesis [12], depth estimation [5], and localization [14]. NeRFs represent continuous volumetric density and RGB values in a neural network and generate photo-realistic images from unseen camera viewpoints through ray tracing. However, a critical limitation of current NeRF-based methods is their inability to quantify the uncertainty associated with the learned appearance and geometry of the scenes they represent. This information is paramount in real-world applications such as medical diagnosis [15] or autonomous driving [11], where confidence in model outputs must be considered in the decision-making process.

1.2. Safety Validation

Safety validation is a critical process in the development and deployment of autonomous systems. It involves testing, performance evaluation, and interpretation of the failure modes of a system to ensure its safe operation [4]. This is particularly crucial for autonomous systems operating in safety-critical domains such as autonomous driving, aircraft collision avoidance, and healthcare. A failure to perform adequate safety validation can risk property damage or even loss of human life.

Safety validation can be incorporated at various stages of the development of an autonomous system. It starts with defining a complete set of realistic and safe requirements. Safety can also be incorporated directly into the design of the system through techniques such as safety-masked reinforcement learning (RL) where an agent learns how to operate under the constraint that it only takes actions that have a minimal likelihood of causing a collision [2].

In the context of safety validation, the adversary plays a crucial role [3]. The adversary seeks to find disturbances that lead to failure. The exact goal of the adversary may be one of the following:

1. **Falsification:** Find any disturbance that leads to a failure.

*Non-CS231n contributor

2. **Most likely failure analysis:** Find the most likely disturbance that leads to a failure.

3. **Estimation of the probability of failure:** Determine how likely it is that any failure will occur based on some prior knowledge.

1.3. Prior Work

In the context of our work, we're dealing with a quadcopter drone that's tasked with navigating through a 3D environment to reach a specified goal while avoiding obstacles. Failures, in this case, are defined as collisions with the environment. In previous work, we explored the use of Neural Radiance Fields (NeRFs) in safety validation, specifically in stress testing a simulated quadcopter agent with a vision-based controller. The challenge lies in determining the most probable sequence of transitions leading to failure. We compared failure modes discovered by the stress testing agent in simulations conducted with the baseline renderer (e.g., Blender) and the NeRF-based simulator. Two stress testing methods were explored: the Monte Carlo (MC) algorithm and the Cross Entropy Method (CEM). The use of a NeRF-powered simulator was found to potentially reduce the computational cost of safety validation, thereby increasing safety and expediting the deployment of autonomous systems.

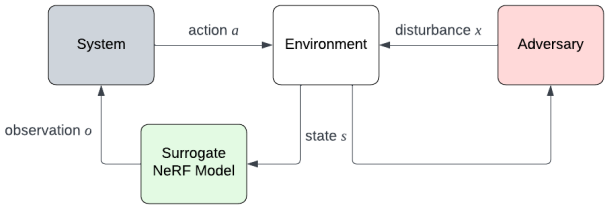

Figure 1: The problem formulation for safety validation using a surrogate NeRF model

We leveraged the NeRF as a surrogate model, which allowed for a comprehensive exploration of potential failure modes and vulnerabilities in safety validation. The detailed and realistic scene representations provided by NeRFs aligned with the objectives of creating a controlled and authentic testing environment. We also created a collision-detection algorithm by generating a collision map and implementing a marching cubes algorithm on the Blender mesh with a granularity of 40 grid cells per world space meter. An image transform was then used to turn the collision map into a signed distance field (SDF). The SDF facilitates the assessment of drone crashes and the collision risk of trajectories, indicating the proximity to collisions in world space meters throughout the trajectory. This previous work provides a solid foundation for further exploration and development in the field of safety validation using Neural Radiance Fields. The methods and findings can serve as valuable context and background for this work.

1.4. Related Work

Recent research efforts aimed at addressing this issue have primarily relied on empirical methods or auxiliary networks.

BayesRays [8] introduces a post-hoc framework to evaluate epistemic uncertainty in any pre-trained NeRF without modifying the training process. Their method establishes a volumetric uncertainty field using spatial perturbations and a Bayesian Laplace approximation, but is limited by its dependency on specific NeRF implementations and the need for a large number of perturbations to achieve accurate uncertainty estimates.

In contrast, the work by Lee et al. [10] presents the Bayesian Neural Radiance Field, which explicitly quantifies uncertainty in geometric volume structures without the need for additional networks. Their method learns a distribution over all possible radiance fields, which is used to quantify the uncertainty associated with the modeled scene. This approach could potentially provide highly accurate and reliable uncertainty estimates. Furthermore, neither method provides a principled way to incorporate uncertainty information during the training of the NeRF. This is a significant limitation as incorporating uncertainty during training could allow the model to better account for potential errors or inaccuracies in the data.

In this work, we aim to address this critical aspect of autonomous systems safety validation by incorporating two UQ methods alongside a surrogate NeRF model in a dynamics simulator framework. This approach encompasses two methods: the Gaussian Approximation, and the Bayesian Laplace Approximation. Our approach will leverage the strengths of these aforementioned methods while also adhering to their limitations. This allows us to quantify the uncertainty in the model's predictions, which is crucial for many applications, especially those involving safety-critical decisions [7, 9, 6].

Our work also serves as a natural progression of a pioneering effort that revolves around NeRFs and their application in vision-based robot navigation and trajectory optimization within 3D environments [1] [13].

2. Dataset

The dataset for this work is composed of disturbance vectors, which are sampled using MC and CEM. Each disturbance vector is representative of an 18-dimensional state space, encapsulating the comprehensive state of our system at any given time step. This state space comprises four key components: position, velocity, rotation, and angular velocity. These disturbance vectors are parameterized by a Multi-

variate Gaussian distribution where μ is a 12-D state vector mean (in this case, initialized to all zeros) and Σ is a 12x12 covariance matrix containing the variances of disturbance value samples.

The data collection process involves conducting a large number of simulations, each introducing a disturbance vector $\mathbf{x}$ at every step. For every trajectory, its likelihood is calculated by summing up the log-likelihood of each of the steps:

$$\log p(\mathbf{x}) = \sum_{t=1}^{T} \log p\left(\mathbf{x}_t \mid \mu, \Sigma\right). \tag{1}$$

The trajectories that failed are then filtered out. An example of the drone's bounding box colliding with a monolith is shown in Figure 2. With these failures, the likelihoods of the trajectories that led to failures in the NeRF simulator are compared to those from the Blender simulator.

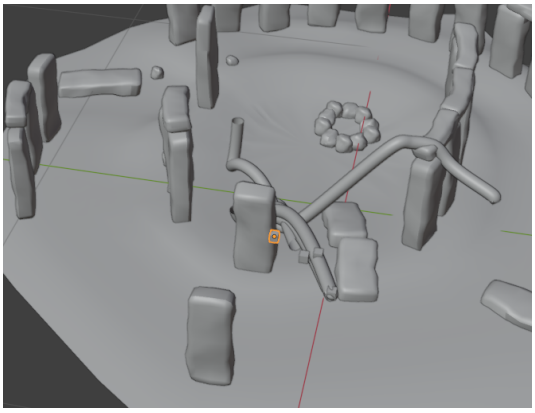

Figure 2: Example of a failure

The CEM accelerates the identification of potential failure modes by maintaining a proposal distribution, a probability distribution of the design space. This distribution is updated using "elite" samples, which are the best samples drawn from the distribution. These samples represent the disturbance values that led to a trajectory and are selected based on their score, the minimum distance to a collision in the trajectory. Weights are assigned to each sample according to the difference in log probabilities of each elite disturbance under the prior and proposal distributions. After normalization, these weights are used to calculate a new proposal distribution, favoring likely disturbances and downplaying unlikely ones. This process helps identify path-dependent vulnerabilities in autonomous systems.

Finally, we created three NeRF-specific image datasets in the typical canonical format, captured via the Blender raytracing engine. These each contain 200 training images, 150 validation images, and 150 test images paired with the camera intrinsics for all.

3. Problem Statement & Technical Approach

The primary objective of this work is to quantify the uncertainty in volumetric density predictions within Neural Radiance Fields. The problem can be addressed by two methods: the Gaussian Approximation and the Bayesian Laplace Approximation. Both methods aim to quantify the uncertainty of the volumetric density that the NeRF predicts at various steps in the simulated trajectories that the drone navigates through. These methods provide a measure of how confident we can be in the predictions made by the NeRF at each step of the drone's trajectory. This measure is then used in a reward function so that the framework continuously samples more likely and certain disturbance vectors, leading to more realistic failure modes.

Gaussian Approximation

The Gaussian Approximation method operates by formulating an objective function for the Maximum Likelihood Estimation (MLE) and then optimizing this function to find the parameters (mean and standard deviation) that minimize it. The objective function is based on the color values, density values, and the rendered color of the NeRF images captured.

We denote the color values as $\mathbf{c}$, the density values as $\mathbf{d}$, and the rendered color as r. The objective function $f(\mu_d, \sigma_d)$ for the MLE can be defined as:

$$f(\mu_d, \sigma_d) = \log\left(\sum_{i=1}^{N_s} \mathbf{c}_i^2 \mathbf{d}_i^2 \sigma_d^2\right) + \frac{\left(r - \sum_{i=1}^{N_s} \mathbf{c}_i \mu_d \mathbf{d}_i\right)^2}{\sum_{i=1}^{N_s} \mathbf{c}_i^2 \sigma_d^2 \mathbf{d}_i^2} \tag{2}$$

where μ_d and σ_d are the mean and standard deviation of the volume density, and N_s is the number of samples. The optimization problem can then be formulated as:

$$\min_{\mu_d, \sigma_d} f(\mu_d, \sigma_d) \tag{3}$$

Bayesian Laplace Approximation

The Bayesian Laplace Approximation method operates by approximating the posterior distribution of the parameterized NeRF. The method starts with a prior distribution for the model parameters and then updates this distribution based on the data (using the likelihood function).

We denote the model parameters as θ, the data as X and y, and the prior distribution as $p(\theta)$.

The likelihood function $p(y|X, \theta)$ can be defined as:

$$p(y|X, \theta) = \prod_{i=1}^{N} p(y_i|x_i, \theta) \tag{4}$$

where N is the number of data points, and x_i and y_i are the input and output of the i-th data point. The posterior distribution $p(\theta|X, y)$ can then be calculated using Bayes' theorem:

$$p(\theta|X,y) = \frac{p(y|X,\theta)p(\theta)}{p(y|X)} \tag{5}$$

where $p(y|X)$ is the marginal likelihood, which can be calculated by integrating over all possible values of θ:

$$p(y|X) = \int p(y|X,\theta)p(\theta)d\theta \tag{6}$$

The Bayesian Laplace Approximation approximates the posterior distribution with a Gaussian distribution centered at the mode of the posterior distribution. The mode $\hat{\theta}$ can be found by maximizing the log posterior:

$$\hat{\theta} = \arg\max_{\theta} \log p(\theta|X,y) \tag{7}$$

The covariance of the Gaussian approximation can then be estimated as the inverse of the approximated Hessian of the log posterior at $\hat{\theta}$. The Hessian, denoted as H, is approximated using the Levenberg-Marquardt method:

$$H \approx J^T J \tag{8}$$

where:

- J is the Jacobian matrix of the function f at $\mathbf{x}_k$
- J^T is the transpose of the Jacobian matrix.

The update rule in the Levenberg-Marquardt method can be written in terms of the Hessian:

$$\mathbf{x}_{k+1} = \mathbf{x}_k - (H + \lambda I)^{-1} g \tag{9}$$

where:

- $\mathbf{x}_k$ is the parameter vector at the k-th iteration
- g is the gradient vector of the function f at $\mathbf{x}_k$
- λ is the damping factor (set to 0.01 in our framework)

This change allows us to calculate the Hessian without the need for an analytical form, which can be beneficial when the analytical form is difficult to derive or compute. This is the case in this work because of the high dimensionality of the state space.

To further improve the robustness of our method, we introduce spatial perturbations to the input data during the fitting process. Specifically, we generate a set of perturbed versions of the input data by adding small random noise. This results in a set of slightly different versions of the original input, which we denote as $X_{\text{perturbed}}$. For each perturbed input X_p in $X_{\text{perturbed}}$, we perform the optimization process described above to find the corresponding maximum a posteriori (MAP) estimate $\hat{\theta}_p$. We then select the $\hat{\theta}_p$ that results in the minimum loss as our final MAP estimate $\hat{\theta}$. This process of introducing spatial perturbations helps to ensure that our model is not overly sensitive to small changes in the input data, enhancing its generalization ability. This is particularly important in our case, given that images have a high dimensionality state space.

4. Experiments & Results

We conducted experiments with a NeRF trained on a simulated Stonehenge environment requiring the quadcopter drone to navigate around a monolith by following a simple six-step trajectory 3. We selected this path to optimize computational efficiency. The performance of each method was evaluated by comparing it to a baseline method (MC/CEM without a reward function) and by comparing the simulator results directly against the Blender simulator, which served as the ground truth for failure modes. The ideal method is expected to result in an increased frequency of collisions, preferably at earlier time steps, and trajectories that exhibit low uncertainty from the NeRF and have a higher likelihood of resulting in a collision.

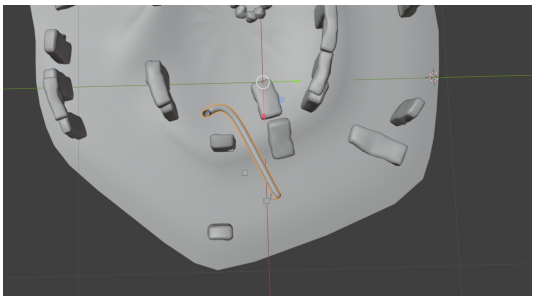

Figure 3: Trajectory of drone path used in experiments

4.1. Gaussian Approximation

The Gaussian Approximation (GA) method was implemented in two stages: an offline stage for testing and verification, and an online stage for integration into the safety validation framework.

4.1.1 Offline stage

In the offline stage, we loaded the corresponding camera parameters and rendered the image using NeRF for each image in our training dataset. We then extracted the color and density values from the output and used these to optimize the parameters of our Gaussian Approximation.

Our optimization resulted in optimized mean and standard deviation values for the volume density for each image. The mean represents the central tendency or the average of the volume density within an image. The standard deviation measures the amount of variation or dispersion in the volume density values. Higher values for either metric indicate a greater degree of uncertainty about the volumetric density of an image based on the rendered color and the color predicted by the NeRF model.

We found that there were several images for which the optimized standard deviation was either absolutely certain

($\sigma_d \leq 0$) or absolutely uncertain ($\sigma_d \geq 3$). These were not included in our results.

The results from the offline Gaussian Approximation method are shown in figure 4, are promising. In our case, the optimized standard deviations (σ_d) were all low ($\leq 1.1\sigma$) to zero for the train, validation, and test sets. The optimized mean and standard deviation values are also shown in relation to each other in figure 5. Both figures provide clear visual representations of the uncertainty in our volume density, which seem to indicate that the model has a high level of confidence in its predictions. This makes sense, as it was the same canonical data used to train and evaluate the model.

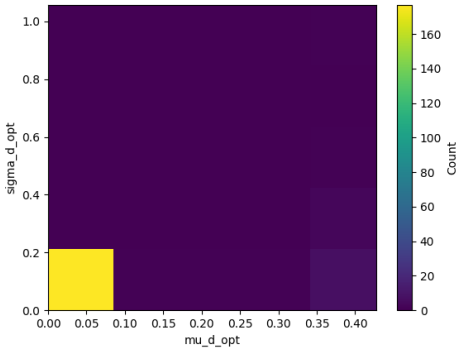

Figure 5: Uncertainty for all data with GA method

4.1.2 Online stage

In the online stage, we integrated the Gaussian Approximation method into the safety validation framework.

A key part of this process is the use of a reward function that adjusts the disturbance vectors in the sampling/stress tests directly. The reward is directly proportional to the likelihood and decreases with increasing uncertainty. All hyperparameters were chosen practically as a result of experimentation and observation.

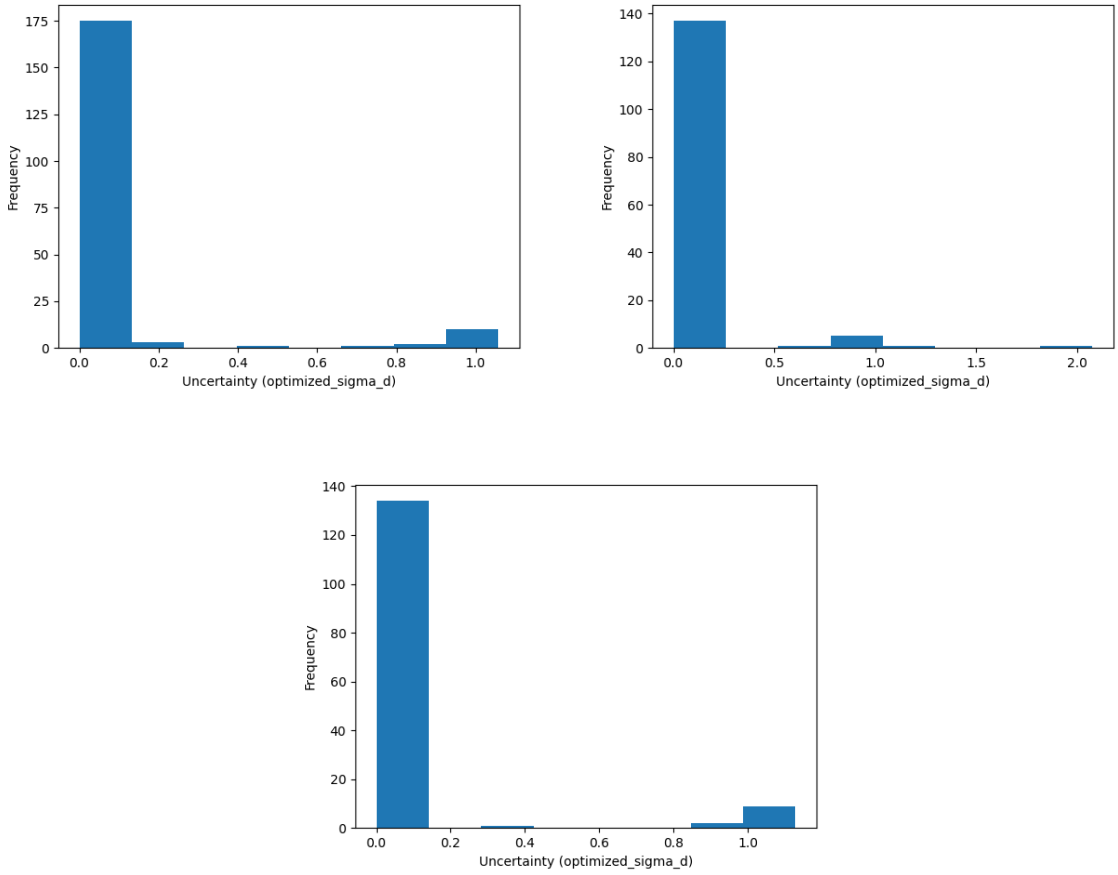

Figure 4: Histograms of σ_d values for train (top left), validation (top right), and test (bottom)

The reward function is defined as follows:

$$\text{reward} = \left(\text{likelihood} - \text{penalty} \times \sigma_{d_{\text{opt}}}\right) \tag{10}$$

where:

- likelihood is the likelihood of the trajectory
- $\sigma_{d_{\text{opt}}}$ is the optimized standard deviation of the volume density used as a measure of uncertainty
- $penalty$ is a predefined constant (set to 36 in our context)

After 100 simulations, our results are illustrated in table 1. Figure 6 compares the NeRF simulator results directly against the ground truth Blender simulator.

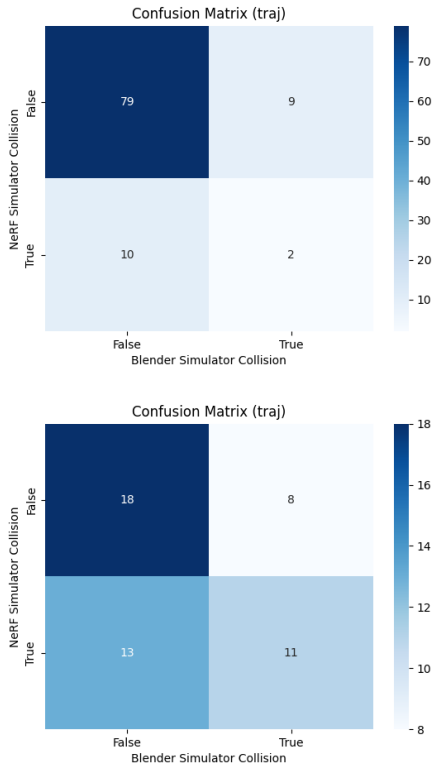

Figure 6: NeRF w/ GA: MC (top) & CEM (bottom)
Accuracy: 81% & 58%
Precision: 0.1667 & 0.4583
Recall: 0.8876 & 0.5789

4.2. Bayesian Laplace Approximation

The Bayesian Laplace Approximation (BLA) method was also implemented in two stages: an offline stage for testing and verification, and an online stage for integration into the safety validation framework.

4.2.1 Offline stage

In the offline stage, we applied a Bayesian Laplace Approximation within the simulator and updated the distribution based on the data using the likelihood function (4). We then calculated the posterior distribution using Bayes' theorem and approximated the posterior distribution with a Gaussian distribution centered at the mode of the posterior distribution. The mode was found by maximizing the log posterior, and the covariance of the approximation was estimated as the inverse of the approximated Hessian of the log posterior at the mode.

Our optimization yielded an optimized posterior mean and covariance matrix per image. We used these to compute two uncertainty metrics: the normalized trace of the covariance matrix, and the normalized root mean variance (rmv). These metrics aggregate the model's prediction uncertainty. The covariance matrix's trace, the sum of its diagonal elements, measures total system variance. In a NeRF model, a larger trace implies higher overall uncertainty in the sigma network parameters, indicating a broad range of plausible values given the data. The root mean variance, the square root of the covariance matrix diagonal elements' average, measures the parameters' average standard deviation. In a NeRF model, a larger rmv suggests higher average uncertainty in individual sigma network parameters.

In the Bayesian Laplace Approximation, uncertainty is contained within the covariance matrix. This is high-dimensional and difficult to interpret directly. By simplifying our posterior covariance matrix into two metrics, it becomes easier to understand and communicate the model's uncertainty.

The results from the offline Bayesian Laplace Approximation method are also promising. The method produced only one significant outlier for our canonical data in the form of an outlier in our test set. The image for that data point included very little of the scene, with most of the camera's focus on the transparent background. This explains the incredibly high degree of uncertainty.

Sampling Method	w/ Rewards	Collision Rate	Avg. Collision Step	Avg. Traj. Likelihood	Avg. Uncertainty
MC	No	12%	4.0	34.13032267	1.009453161
CEM	No	44%	4.0	**39.73473478**	0.966691219
MC	Yes	12%	**3.083333333**	38.15115893	0.980460885
CEM	Yes	**48%**	3.833333333	39.22802489	**0.938138706**

Table 1: Simulation Results for Gaussian Approximation Method

The trace and root mean variance values were low, clustered, and very tightly together. Figure 7 illustrates this phenomenon. This is reflective of the same high degree of confidence seen in the Gaussian Approximation method. Figure 8 indicates a generally strong correlation between the trace and root mean variance.

4.2.2 Online stage

In the online stage, we integrated the Bayesian Laplace Approximation method into the safety validation framework.

Similar to the Gaussian Approximation method, we also use a reward function in the Bayesian Laplace Approximation method. The reward is directly proportional to the likelihood and decreases with increasing uncertainty, which is the product of the trace and root mean variance of the diagonal elements of the posterior covariance matrix. All hyperparameters were chosen practically as a result of experimentation and observation.

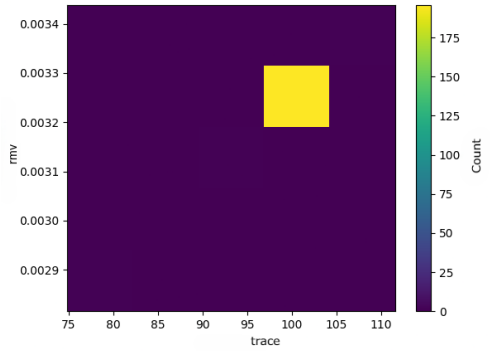

Figure 8: Uncertainty for all data with BLA method

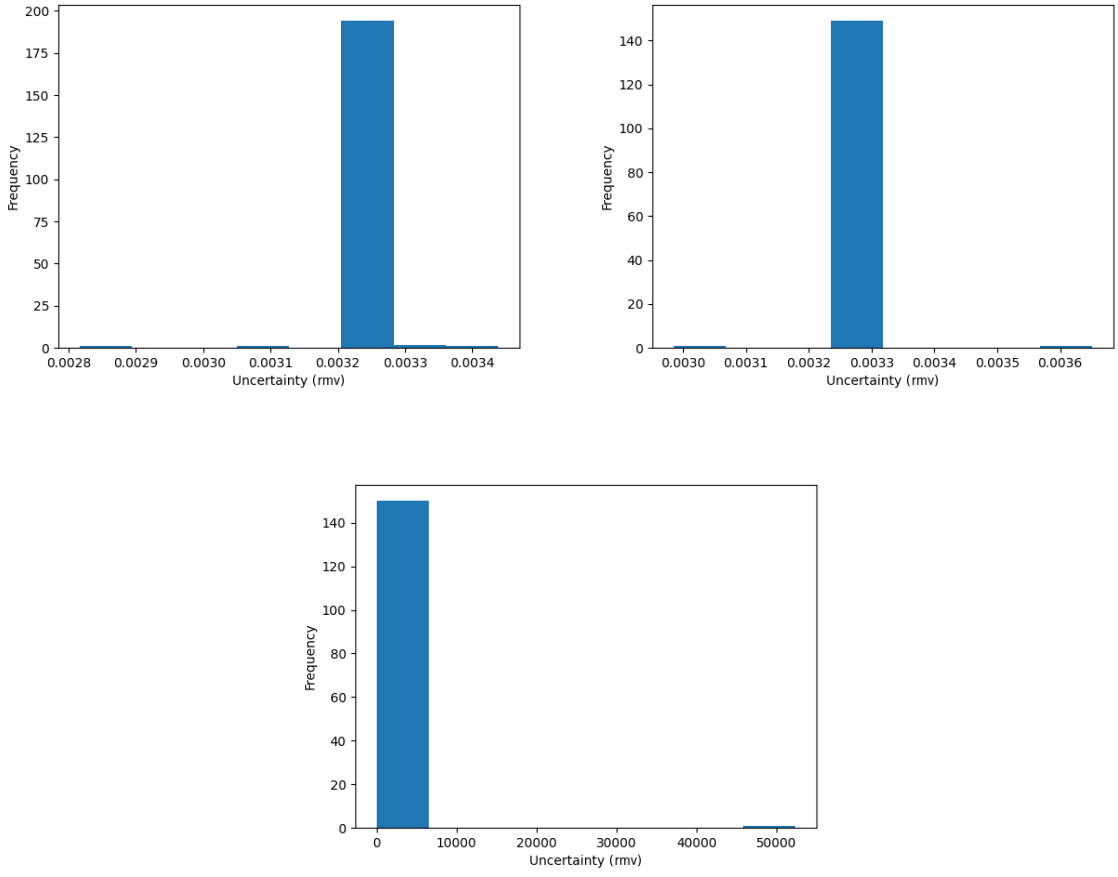

Figure 7: Histograms of σ values for train (top left), validation (top right), and test (bottom)

Sampling Method	w/ Rewards	Collision Rate	Avg. Collision Step	Avg. Traj. Likelihood	Avg. Uncertainty
MC	No	12%	**4.0**	34.13032267	0.00325468
CEM	No	**44%**	**4.0**	**39.73473478**	0.003235783
MC	Yes	13%	4.153846154	33.9025961	**0.003194839**
CEM	Yes	**44%**	**4.0**	**39.73473478**	0.003253748

Table 2: Simulation Results for Bayesian Laplace Approximation Method

The reward function is defined as follows:

$$\text{reward} = (\text{likelihood} - \text{penalty} \times \sigma \times \text{trace} \times \text{n}) \quad (11)$$

where:

- likelihood is the likelihood of the trajectory
- rmv is the square root of the mean of diagonal elements in the posterior covariance matrix
- trace is the trace of the covariance matrix
- n is the number of spatial perturbations (3 in our context)
- $penalty$ is a predefined constant (set to 36 in our context)

After 50 simulations (5 populations with 10 simulations and 5 elite samples each), our results are illustrated in table 2. Figure 9 compares the NeRF simulator results directly against the ground truth Blender simulator.

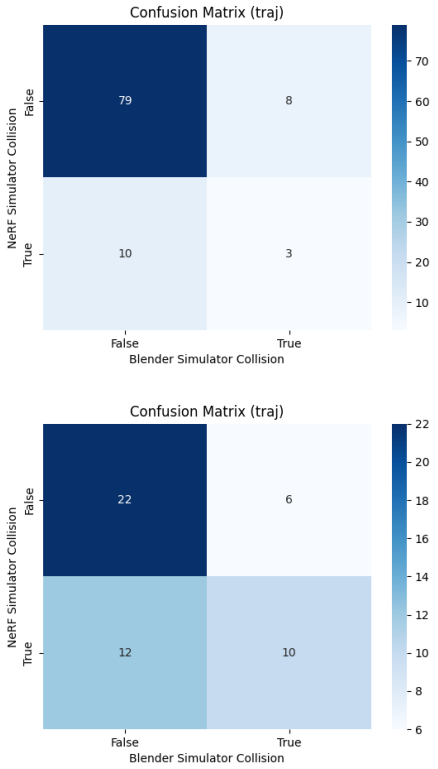

Figure 9: NeRF w/ BLA: MC (top) & CEM (bottom)
Accuracy: 82% & 64%
Precision: 0.2308 & 0.4545
Recall: 0.2727 & 0.6250

5. Conclusion

5.1. Summary

In this work, we have presented a novel approach to safety validation using NeRFs. The results from both the offline Gaussian Approximation and Bayesian Laplace Approximation methods are promising, showing a high degree of confidence in the predictions made by the NeRF model. Online methods also show potential through simulations that integrate uncertainty into the safety validation framework through a reward function. These simulations exhibit slight metric improvements compared to those that disregard uncertainty.

In conclusion, this work represents a significant step forward in the field of safety validation for autonomous systems. By leveraging NeRFs as surrogate models and incorporating uncertainty quantification into the safety validation process, we have demonstrated the potential for more robust and comprehensive safety validation for autonomous systems.

5.2. Limitations & Future Work

This work has several limitations that open avenues for future research. The NeRF model's architecture may not capture high-frequency details in complex scenes, affecting the accuracy of failure modes.

Our safety validation's reliance on a simulated environment creates a gap with real-world scenarios. The fixed parameters in our reward function and the computational cost of the Gaussian Approximation and Bayesian Laplace Approximation methods limit their efficacy. A lookup table from offline methods could enhance simulations by storing previous computations. The use of aggregate metrics overlooks pixel-specific variations, limiting the precision of safety validation. High uncertainty values in some images indicate struggles with unfamiliar viewpoints and poses, which could be mitigated by including more close-up views in the training data. Future work also could explore the application of these methods in different domains, such as autonomous driving or robotic manipulation.

6. Acknowledgements

I would like to express my gratitude to Prof. Mykel Kochenderfer for providing me with the opportunity to explore this intriguing research problem as a part of the Stanford Intelligent Systems Laboratory (SISL). The study of autonomous systems and safety validation has been both enriching and enjoyable. Having previously worked with NeRFs, it's been incredibly exciting to see how they can be applied, and our preliminary results have provided future direction for this work. I'm also thankful to have had the privilege of witnessing some of the innovative work being conducted and presented by other members of SISL, serving as a source of constant inspiration. Lastly, I would like to extend my heartfelt thanks to Harrison Delecki for his invaluable mentorship throughout this work. His guidance through discussion was instrumental in navigating the nuanced complexities faced. Finally, I also want to acknowledge Yash Kadadi for his exceptional contributions to the validation framework codebase used in this work.

References

[1] M. Adamkiewicz, T. Chen, A. Caccavale, R. Gardner, P. Culbertson, J. Bohg, and M. Schwager. Vision-only robot navigation in a neural radiance world, 2022.

[2] A. Corso and M. J. Kochenderfer. Interpretable safety validation for autonomous vehicles. In *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, pages 1–6, 2020.

[3] A. Corso and M. J. Kochenderfer. Transfer learning for efficient iterative safety validation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(8):7125–7132, May 2021.

[4] A. Corso, R. Moss, M. Koren, R. Lee, and M. Kochenderfer. A survey of algorithms for black-box safety validation of cyber-physical systems. *Journal of Artificial Intelligence Research*, 72, Oct. 2021.

[5] K. Deng, A. Liu, J.-Y. Zhu, and D. Ramanan. Depth-supervised nerf: Fewer views and faster training for free, 2022.

[6] Y. Gal and Z. Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. *arXiv preprint arXiv:1506.02142*, 2016.

[7] Z. Ghahramani. Probabilistic machine learning and artificial intelligence. *Nature*, 521(7553):452–459, 2015.

[8] L. Goli, C. Reading, S. Sellán, A. Jacobson, and A. Tagliasacchi. Bayes' rays: Uncertainty quantification for neural radiance fields, 2023.

[9] A. Kendall and Y. Gal. What uncertainties do we need in bayesian deep learning for computer vision? *arXiv preprint arXiv:1703.04977*, 2017.

[10] S. Lee, K. Kang, and H. Yu. Bayesian nerf: Quantifying uncertainty with volume density in neural radiance fields, 2024.

[11] C. Lindström, G. Hess, A. Lilja, M. Fatemi, L. Hammarstrand, C. Petersson, and L. Svensson. Are nerfs ready for autonomous driving? towards closing the real-to-simulation gap, 2024.

[12] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. Nerf: Representing scenes as neural radiance fields for view synthesis, 2020.

[13] J. Tang. Torch-ngp: a pytorch implementation of instant-ngp, 2022. https://github.com/ashawkey/torch-ngp.

[14] W. Wang, V. Cai, and S. Gil. Mulan-wc: Multi-robot localization uncertainty-aware active nerf with wireless coordination, 2024.

[15] X. Wang, S. Hu, H. Fan, H. Zhu, and X. Li. Neural radiance fields in medical imaging: Challenges and next steps, 2024.