Epistemic Uncertainty in State Estimation and Belief Space Planning with Learning-Based Perception Systems

Keiko Nagami
Department of Aeronautics and Astronautics
Stanford University, United States
knagami@stanford.edu

Mac Schwager
Department of Aeronautics and Astronautics
Stanford University, United States
schwager@stanford.edu

Abstract: Learning-based models for robot perception are known to suffer from two distinct sources of error: aleatoric and epistemic. Aleatoric uncertainty arises from inherently noisy training data and is easily quantified from residual errors during training. Conversely, epistemic uncertainty arises from a lack of training data, appearing in out-of-distribution operating regimes, and is difficult to quantify. In this work, we propose: (i) an epistemic Kalman filter (EpiKF) to incorporate epistemic uncertainty into state estimation with learned perception models, and (ii) an epistemic belief space planner (EpiBSP) that builds on the EpiKF to plan trajectories to avoid areas of high epistemic and aleatoric uncertainty. Our key insight is to train a generative model that predicts measurements from states, "inverting" the learned perception model that predicts states from measurements.

Keywords: Learned Perception, Belief Space Planning, Out of Distribution Data

1 Introduction

We consider a common architecture for vision based navigation in which a learning-based perception model ingests raw RGB images and outputs pseudo-measurements of the robot's state vector. Our research is motivated by two key shortcomings in existing methods: (i) learning covariance as an output of a perception model fails to quantify epistemic uncertainty, and (ii) learning-based perception models do not allow for projecting uncertainty into the future, posing a significant challenge to planners that reason about future uncertainty, risk, or the belief space.

To overcome these problems, we train a generative model in conjunction with the perception model. The generator takes as input the robot's state vector and outputs a predicted image of what the robot would see at that state. This is the inverse of the perception model, which takes in an image, and outputs a predicted state of where the agent took the image from.

We use the composition of the image generator and perception module in a sampling scheme to obtain an online covariance estimate for the pseudo-measurement, and apply this to the update step of a Gaussian belief filter, which we call the Epistemic Kalman filter (EpiKF). We then incorporate this filter within an optimization-based belief space planning algorithm to plan trajectories that balance efficiency with state estimation uncertainty, which we call the Epistemic Belief Space Planner (EpiBSP). The EpiBSP naturally avoids areas of high sensor noise (aleatoric uncertainty), as well as areas with sparse training data (epistemic uncertainty), without having access to the training data itself. Fig. 1 demonstrates the use of our generative model denoted $g_\phi(\cdot)$ and perception model $p_\theta(\cdot)$ to give a measurement uncertainty signal for both the current and predicted future states.

7th Conference on Robot Learning (CoRL 2023), Atlanta, USA.

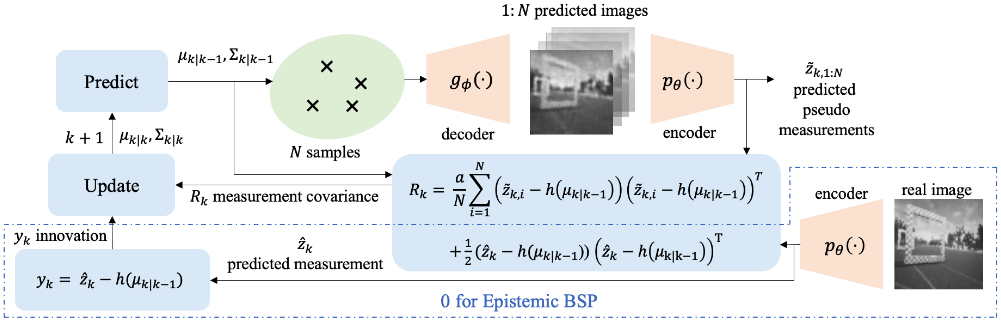
1: N predicted images
Predict
$\mu_{k|k-1}, \Sigma_{k|k-1}$
N samples
$g_\phi(\cdot)$
decoder
$p_\theta(\cdot)$
encoder
$\tilde{z}_{k,1:N}$ predicted pseudo measurements
$k+1$
$\mu_{k|k}, \Sigma_{k|k}$
Update
R_k measurement covariance
$R_k = \frac{a}{N}\sum_{i=1}^{N}\left(\tilde{z}_{k,i} - h(\mu_{k|k-1})\right)\left(\tilde{z}_{k,i} - h(\mu_{k|k-1})\right)^T + \frac{1}{2}(\hat{z}_k - h(\mu_{k|k-1}))\left(\hat{z}_k - h(\mu_{k|k-1})\right)^T$
encoder
real image
$p_\theta(\cdot)$
y_k innovation
$\hat{z}_k$ predicted measurement
$y_k = \hat{z}_k - h(\mu_{k|k-1})$
0 for Epistemic BSP

Figure 1: In both the epistemic Kalman filter and epistemic belief space planner, N measurements are sampled from the predicted distribution, and passed through the neural network generator $g_\phi(\cdot)$. N predicted images are forwarded to the neural network perception model $p_\theta(\cdot)$. The resulting output of predicted pseudo-measurements are used to compute the measurement covariance matrix.

2 Related Work

A common approach to anticipating both aleatoric and epistemic uncertainty from neural networks is using ensemble methods [1], [2], [3], [4]. However, ensemble methods require training of multiple neural networks to obtain disagreement across the models as an OOD metric. Alternative approaches for OOD detection include the methods in [5], [6], [7], and [8], however these approaches are unsuitable for our problem, since they cannot anticipate measurement uncertainty at future states as required in a belief space planner. This motivates our approach of training a measurement generator in conjunction with the perception model.

Active perception or perception-aware planning methods generally require a metric to define the quality of the perception system's performance at future states. Methods that use Visual-Inertial Odometry focus largely on aleatoric uncertainty, where the perception system's performance is tied to the amount of features in frame, as done in [9], [10], and [11], as well as their velocities or errors, as is done in [12], and [13]. Other methods in active perception have also used Neural Networks as part of the planning and control pipelines [14], [15]. However, these methods do not handle epistemic uncertainty, which is crucial in learned perception systems.

Belief space planning considers the evolution of the whole belief along future trajectories, and optimizes plans to balance trajectory efficiency with information richness. A common approach is to formulate the problem as a Partially Observable Markov Decision Process (POMDP) [16]. However, with continuous belief states, actions, and observations, the problem is typically intractable. An alternative solution is to use sampling based approaches ([17], [18], [19], [11], [20]). Another method is to use local trajectory optimization approaches ([21], [22], [23], [24]). None of these methods consider the epistemic uncertainty that arises with learned perception models.

Our work differs significantly from all of these works in the way we quantify measurement noise at future states in the plan. We use two neural networks to define our measurement uncertainty. When we compose these models, we find that the error between the input state to the generator and the output state from the perception model gives a signal that is well-correlated with epistemic uncertainty. The use of this anticipated measurement uncertainty enables us to formulate an epistemic belief space planner (EpiBSP) built on an epistemic Kalman filter (EpiKF), to account for degraded state estimation performance in the regions of high epistemic and aleatoric uncertainty.

3 Methods

We train a neural network perception system that takes an image input $\mathcal{I}$ and outputs a predicted pseudo-measurement of the state, $\hat{\mathbf{z}} = p_\theta(\mathcal{I})$, where θ represents network parameters. We also train an image generator to take a pseudo-measurement $\mathbf{z}$ and output a predicted image, $\hat{\mathcal{I}} = g_\phi(\mathbf{z})$, where ϕ represents the network parameters. p_θ and g_ϕ are trained so their composition is close to the identify map on the training data, $p_\theta(g_\phi(\mathbf{z})) \approx \mathbf{z}$. We find that the error $(p_\theta(g_\phi(\mathbf{z})) - \mathbf{z})$ is a good proxy for the prediction error $(\mathbf{z} - p_\theta(\mathcal{I}))$, but has the key advantage that it does not require access to an image $\mathcal{I}$, so it can be used for planning over future states where the image is unknown. We use the pseudo-measurement to filter the state with a Kalman filter, and use this prediction error proxy to give a measurement error for the filter, to quantify OOD sources of error in the pseudo-measurement.

In the traditional extended Kalman filter equations for nonlinear systems, we have predict and update steps as follows:

$$\begin{aligned}
\text{Predict:} \quad \boldsymbol{\mu}_{k+1|k} &= f(\boldsymbol{\mu}_{k|k}, \mathbf{u}_k) && (1)\\
\boldsymbol{\Sigma}_{k+1|k} &= \mathbf{F}_k \boldsymbol{\Sigma}_{k|k} \mathbf{F}_k^T + \mathbf{Q} && (2)\\
\text{Update:} \quad \mathbf{y}_{k+1} &= \hat{\mathbf{z}}_{k+1} - h(\boldsymbol{\mu}_{k+1|k}) && (3)\\
\mathbf{S}_{k+1} &= \mathbf{H}_{k+1} \boldsymbol{\Sigma}_{k+1|k} \mathbf{H}_{k+1}^T + \mathbf{R}_{k+1} && (4)\\
\mathbf{K}_{k+1} &= \boldsymbol{\Sigma}_{k+1|k} \mathbf{H}_{k+1}^T \mathbf{S}_{k+1}^{-1} && (5)\\
\boldsymbol{\mu}_{k+1|k+1} &= \boldsymbol{\mu}_{k+1|k} + \mathbf{K}_{k+1} \mathbf{y}_{k+1} && (6)\\
\boldsymbol{\Sigma}_{k+1|k+1} &= (\mathbf{I} - \mathbf{K}_{k+1} \mathbf{H}_{k+1}) \boldsymbol{\Sigma}_{k+1|k}, && (7)
\end{aligned}$$

where subscript k indicates the time index, $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are the mean and covariance of the estimated state, and, $\mathbf{H}$ is the Jacobian of the measurement function $h(\cdot)$, $\mathbf{F}$ is the Jacobian of the dynamics equation $f(\cdot)$, $\mathbf{I}$ is an identity matrix, and $\mathbf{R}$ is the measurement noise covariance matrix. We propose to obtain $\mathbf{R}$ in (4) by first sampling pseudo-measurement vectors from the prediction distribution, passing these sampled vectors through the generator network to produce predicted images, and then passing the images through the perception network to attempt to reproduce the state vector,

$$\begin{aligned}
\mathbf{z}_{k,i} &\sim \mathcal{N}(\mathbf{H}_k \boldsymbol{\mu}_{k|k-1}, \mathbf{H}_k \boldsymbol{\Sigma}_{k|k-1} \mathbf{H}_k^T), \quad \forall i = 1:N && (8)\\
\tilde{\mathbf{z}}_{k,i} &= p_\theta(g_\phi(\mathbf{z}_{k,i})). && (9)
\end{aligned}$$

This sampling procedure is done for N samples, where i is the index of the sample. With the sampled measurement vectors $\tilde{\mathbf{z}}_{k,i}$, we compute their empirical covariance matrix, and combine it with the error of the estimate from the real image from the predicted mean,

$$\mathbf{R}_k = \frac{1}{2N} \sum_{i=1}^{N} (\tilde{\mathbf{z}}_{k,i} - \mathbf{H}_k \boldsymbol{\mu}_{k|k-1})(\tilde{\mathbf{z}}_{k,i} - \mathbf{H}_k \boldsymbol{\mu}_{k|k-1})^T + \frac{1}{2} (\hat{\mathbf{z}}_k - \mathbf{H}_k \boldsymbol{\mu}_{k|k-1})(\hat{\mathbf{z}}_k - \mathbf{H}_k \boldsymbol{\mu}_{k|k-1})^T. \tag{10}$$

This matrix $\mathbf{R}_k$ is then used as the measurement noise covariance matrix in the EKF equations. This overall pipeline is shown in Fig. 1. With this formulation we are able to get a measurement uncertainty matrix that reflects the predicted performance of the neural network perception model at the estimated state in the state space.

The trajectory optimization problem we seek to solve takes the following form,

$$\begin{aligned}
\underset{\mathbf{u}_{k=1:K-1}}{\text{minimize}} \quad & c_1 \text{tr}(\boldsymbol{\Sigma}_K) + c_2 \bar{\boldsymbol{\mu}}_K^T \mathbf{C}_s \bar{\boldsymbol{\mu}}_K + c_3 \mathbf{u}^T \mathbf{C}_u \mathbf{u} && (11)\\
\text{s.t. } \boldsymbol{\mu}_{k+1} &= f(\boldsymbol{\mu}_k, \mathbf{u}_k), \quad \forall k = 1 : K-1 && (12)\\
\boldsymbol{\Sigma}_{k+1} &= \mathbf{F}_k \boldsymbol{\Sigma}_k \mathbf{F}_k^T + \mathbf{Q} && (13)\\
\mathbf{y}_{k+1} &= \tilde{\mathbf{z}}_{k+1} - h(\boldsymbol{\mu}_{k+1}) && (14)\\
\mathbf{S}_{k+1} &= \mathbf{H}_{k+1} \boldsymbol{\Sigma}_{k+1} \mathbf{H}_{k+1}^T + \mathbf{R}_{k+1} && (15)\\
\mathbf{K}_{k+1} &= \boldsymbol{\Sigma}_{k+1} \mathbf{H}_{k+1}^T \mathbf{S}_{k+1}^{-1} && (16)\\
\boldsymbol{\mu}_{k+1} &= \boldsymbol{\mu}_{k+1} + \mathbf{K}_{k+1} \mathbf{y}_{k+1} && (17)\\
\boldsymbol{\Sigma}_{k+1} &= (\mathbf{I} - \mathbf{K}_{k+1} \mathbf{H}_{k+1}) \boldsymbol{\Sigma}_{k+1}, && (18)
\end{aligned}$$

where K is the terminal time index, $\bar{\boldsymbol{\mu}}_K$ is the difference between the mean of the estimate $\boldsymbol{\mu}_K$ and the goal state $\mathbf{s}_g$, $\mathbf{C}_s$ is a cost matrix on the terminal state's goal, $\mathbf{C}_u$ is a cost matrix on the

Table 1: Comparison Results for Full Racetrack

	Gate Collisions	Error to Ground Truth	Total Time
Our Method	0	0.3074 ± 0.2315	2 min 35.94 sec
Baseline	1	0.4047 ± 0.4453	2 min 38.68 sec

control action, c_1, c_2, and c_3 are constants, $\tilde{\mathbf{z}}$ is the predicted measurement, and $\mathbf{R}$ is the predicted measurement uncertainty. For the predicted measurement uncertainty, we follow a similar procedure to the measurement uncertainty in the EpiKF. However, the key challenge in the belief space planner is the lack of access to the collected image measurements from the robot. As such, we use only the first term in Equation 10 to define the measurement uncertainty in the belief space planner:

$$\mathbf{R}_k = \frac{1}{N}\sum_{i=1}^{N}(\tilde{\mathbf{z}}_{k,i} - \mathbf{H}_k\boldsymbol{\mu}_{k|k-1})(\tilde{\mathbf{z}}_{k,i} - \mathbf{H}_k\boldsymbol{\mu}_{k|k-1})^T. \tag{19}$$

Since there is no real image collected for the planner, we use a Maximum Likelihood Observation. With this formulation, our belief state dynamics are a function of the neural network generator and perception model. For each sample, we use Pytorch [25] to compute gradients of the cost with respect to the decision variables to obtain an optimized trajectory.

4 Track Simulation

To showcase both the EpiKF and EpiBSP working together, we apply both to a case where a drone is flying through a racetrack with multiple gates. We use the generator and perception neural networks trained with epistemic uncertainty, where data from one side of the gate is withheld. For planning, we apply the EpiBSP by solving for a trajectory at each gate transition. Once the trajectory is optimized, we track this trajectory in a model-predictive control (MPC) loop. The resulting trajectories and light-dark domain are shown in Fig. 2. We additionally compare our method to a baseline similar to that of [26], where we use a neural network that estimates a pseudo-measurement and an associated covariance for estimation. For baseline planning, we use a trajectory from an IPOPT solver. The trajectory for three laps is also shown in Fig. 2.

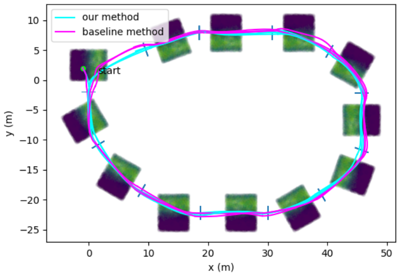

Figure 2: Trajectories of our method and baseline method through the Airsim racetrack for three laps, with light-dark domain ahead of each gate.

To quantify the performance of our method, we compare the number of gate collisions, error of the state estimate mean to the ground truth, and the total number of time steps taken to reach the goal. These metrics are shown for our method and the baseline in Table 1. We find that our method is able to complete three laps with fewer collisions, and maintain a lower error from the ground truth than the baseline approach.

5 Conclusion

In this paper we demonstrate a novel approach to quantifying neural network uncertainty, and use this metric to anticipate regions of high uncertainty with belief space planning and Kalman filtering. We found that our method is able to avoid regions of both aleatoric and epistemic uncertainty to produce trajectories where the perception model is more likely to succeed. In future work, real-time belief space planning trajectories can be prioritized by applying a sampling based planning approach to reduce computation time required for the presented local optimization approach, which would enable real-world experiments.

References

[1] K. Chua, R. Calandra, R. McAllister, and S. Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. *Advances in neural information processing systems*, 31, 2018.

[2] B. Lakshminarayanan, A. Pritzel, and C. Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, 30, 2017.

[3] F. K. Gustafsson, M. Danelljan, and T. B. Schon. Evaluating scalable bayesian deep learning methods for robust computer vision. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 318–319, 2020.

[4] A. Loquercio, M. Segu, and D. Scaramuzza. A general framework for uncertainty estimation in deep learning. *IEEE Robotics and Automation Letters*, 5(2):3153–3160, 2020. doi:10.1109/LRA.2020.2974682.

[5] A. Sharma, N. Azizan, and M. Pavone. Sketching curvature for efficient out-of-distribution detection for deep neural networks. In *Uncertainty in Artificial Intelligence*, pages 1958–1967. PMLR, 2021.

[6] R. Luo, R. Sinha, A. Hindy, S. Zhao, S. Savarese, E. Schmerling, and M. Pavone. Online distribution shift detection via recency prediction. *arXiv preprint arXiv:2211.09916*, 2022.

[7] A. Angelopoulos, S. Bates, J. Malik, and M. I. Jordan. Uncertainty sets for image classifiers using conformal prediction. *arXiv preprint arXiv:2009.14193*, 2020.

[8] Y. Romano, M. Sesia, and E. Candes. Classification with valid and adaptive coverage. *Advances in Neural Information Processing Systems*, 33:3581–3591, 2020.

[9] L. Bartolomei, L. Teixeira, and M. Chli. Perception-aware path planning for uavs using semantic segmentation. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5808–5815. IEEE, 2020.

[10] M. Jacquet, G. Corsini, D. Bicego, and A. Franchi. Perception-constrained and motor-level nonlinear mpc for both underactuated and tilted-propeller uavs. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4301–4306. IEEE, 2020.

[11] M. W. Achtelik, S. Lynen, S. Weiss, M. Chli, and R. Siegwart. Motion-and uncertainty-aware path planning for micro aerial vehicles. *Journal of Field Robotics*, 31(4):676–698, 2014.

[12] X. Wu, S. Chen, K. Sreenath, and M. W. Mueller. Perception-aware receding horizon trajectory planning for multicopters with visual-inertial odometry. *arXiv preprint arXiv:2204.03134*, 2022.

[13] D. Falanga, P. Foehn, P. Lu, and D. Scaramuzza. Pampc: Perception-aware model predictive control for quadrotors. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1–8. IEEE, 2018.

[14] Y. Song, K. Shi, R. Penicka, and D. Scaramuzza. Learning perception-aware agile flight in cluttered environments. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1989–1995. IEEE, 2023.

[15] J. Tordesillas and J. P. How. Deep-panther: Learning-based perception-aware trajectory planner in dynamic environments. *IEEE Robotics and Automation Letters*, 8(3):1399–1406, 2023.

[16] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101(1):99–134, 1998. ISSN 0004-3702. doi:https://doi.org/10.1016/S0004-3702(98)00023-X. URL https://www.sciencedirect.com/science/article/pii/S000437029800023X.

[17] S. Prentice and N. Roy. The belief roadmap: Efficient planning in belief space by factoring the covariance. *The International Journal of Robotics Research*, 28(11-12):1448–1465, 2009.

[18] A. Bry and N. Roy. Rapidly-exploring random belief trees for motion planning under uncertainty. In *2011 IEEE International Conference on Robotics and Automation*, pages 723–730, 2011. doi:10.1109/ICRA.2011.5980508.

[19] J. Van Den Berg, P. Abbeel, and K. Goldberg. Lqg-mp: Optimized path planning for robots with motion uncertainty and imperfect state information. *The International Journal of Robotics Research*, 30(7):895–913, 2011.

[20] Z. Sunberg and M. Kochenderfer. Online algorithms for pomdps with continuous state, action, and observation spaces. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 28, pages 259–263, 2018.

[21] R. Platt Jr, R. Tedrake, L. Kaelbling, and T. Lozano-Perez. Belief space planning assuming maximum likelihood observations. 2010.

[22] J. Van Den Berg, S. Patil, and R. Alterovitz. Motion planning under uncertainty using iterative local optimization in belief space. *The International Journal of Robotics Research*, 31(11): 1263–1278, 2012.

[23] M. Rafieisakhaei, S. Chakravorty, and P. Kumar. T-lqg: Closed-loop belief space planning via trajectory-optimized lqg. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 649–656. IEEE, 2017.

[24] K. Sun and V. Kumar. Belief space planning for mobile robots with range sensors using ilqg. *IEEE Robotics and Automation Letters*, 6(2):1902–1909, 2021.

[25] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[26] E. Kaufmann, M. Gehrig, P. Foehn, R. Ranftl, A. Dosovitskiy, V. Koltun, and D. Scaramuzza. Beauty and the beast: Optimal methods meet learning for drone racing. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 690–696. IEEE, 2019.