PRISM: Probabilistic Real-Time Inference in Spatial World Models

Atanas Mirchev[1], **Baris Kayalibay**[1], **Ahmed Agha**[1],
Patrick van der Smagt[1], **Daniel Cremers**[2], **Justin Bayer**[1]

[1]Machine Learning Research Lab, Volkswagen Group, [2]Technical University of Munich
atanas.mirchev@argmax.ai

Abstract: We introduce PRISM, a method for real-time filtering in a probabilistic generative model of agent motion and visual perception. Previous approaches either lack uncertainty estimates for the map and agent state, do not run in real-time, do not have a dense scene representation or do not model agent dynamics. Our solution reconciles all of these aspects. We start from a predefined state-space model which combines differentiable rendering and 6-DoF dynamics. Probabilistic inference in this model amounts to simultaneous localisation and mapping (SLAM) and is intractable. We use a series of approximations to Bayesian inference to arrive at probabilistic map and state estimates. We take advantage of well-established methods and closed-form updates, preserving accuracy and enabling real-time capability. The proposed solution runs at 10Hz real-time and is similarly accurate to state-of-the-art SLAM in small to medium-sized indoor environments, with high-speed UAV and handheld camera agents (Blackbird, EuRoC and TUM-RGBD).

Keywords: generative model, SLAM, Bayes filter, uncertainty, diff. rendering

1 Introduction

Moving agents perceive streams of information, typically a mix of RGB images, depth and inertial measurements. Probabilistic generative models [1] are a principled way to formalise the *synthesis* of this data, and from these models inference can be derived through Bayes' rule. We focus on exactly such inference and target the agent states and the scene map, a problem known as simultaneous localisation and mapping (SLAM). We treat it as a posterior approximation for a given state-space model, such that the combination is useful for model-based control: the posterior inference serves as a state estimator and the predictive state-space model as a simulator with which to plan ahead [2].

To pave the way towards decision making, we believe an inference method should have:

- a compatible predictive model for both RGB-D images and 6-DoF dynamics;
- principled state and map uncertainty;
- real-time performance on commodity hardware;
- state-of-the-art localisation accuracy.

We motivate these requirements further in appendix J. Prominent methods like LSD-SLAM [3], ORB-SLAM [4], DSO [5] have propelled visual SLAM forward, with heavy focus on large-scale localisation. The core of modern large-scale SLAM is maximum a-posteriori (MAP) smoothing in a probabilistic factor graph [6, 7]. At present this demands sparsity assumptions for computational feasibility, which obstructs the tight integration of dense maps and rendering. Nonetheless, for smaller scenes the recent popularity of neural models (e.g. NERF [8]) has sparked interest in inference through a renderer (e.g. [9, 10, 11]), but dynamics modelling and uncertainty have remained out of scope. Conversely, classical filtering comes with dynamics and uncertainty in real-time (e.g. [12, 13, 14]),

6th Conference on Robot Learning (CoRL 2022), Auckland, New Zealand.

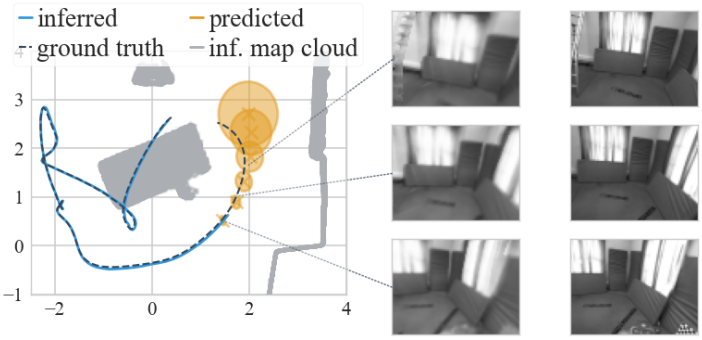


Figure 1: Inference is tailored to the depicted predictive model. Predicting future rollouts, as shown, is required for optimal control. Ground-truth trajectory in *black*, inferred trajectory from past data in *blue*. In *orange*, we see uncertainty envelopes for the predicted future states. On the right, we see predicted and ground-truth future images. Visualised in 2D for clarity, our method operates in 3D.

but over time has given way to large-scale smoothing [6] and to our knowledge has not been well explored for the integration of dense differentiable rendering and dynamics on a moderate scale.

Overall, we find there is a need for a cohesive inference solution that satisfies our requirements. We thus contribute by meeting all the above goals, emphasising the link to a predictive model (fig. 1).

We start from the generative model of Mirchev et al. [15], who combine differentiable rendering and agent dynamics in a probabilistic framework. The authors considered stochastic variational inference for this model, applying it off-line with runtime orders of magnitude too long for on-line use. We pursue an alternative route for real-time inference: from the generative assumptions we derive approximations to the true marginal filters over the last state and map [16]. By focusing on recursive filtering updates, we identify where established probabilistic inference and computer vision techniques can be used, putting emphasis on fast closed-form updates. We find this divide-and-conquer strategy is a good compromise for achieving the aforementioned objectives under computational constraints.

We evaluate the proposed solution on two unmanned aerial vehicle (UAV) data sets [17, 18] and on TUM-RGBD [19]. Our method PRISM runs at 10 Hz real-time with similar localisation accuracy to state-of-the-art SLAM in moderately-sized indoor environments. It provides uncertainty estimates and features a predictive distribution that can both render images and forecast the agent's movement.

2 Related Work

Generative models Generative state-space models simulate the formation of observed data over time in a Markov chain [1, 12, 20, 21, 22, 23], serving as *world* models [24, 25]. With their agent dynamics and state-to-observation emission models we can imagine future rollouts for planning [2, 26, 27, 28, 29, 30, 31, 32, 33]. We abide by this framework and design a posterior inference for a *spatial* state-space model, to enable on-line control. Among such models (e.g. [34, 35, 36, 37, 38, 39, 40]), we tailor our inference to the model of Mirchev et al. [15]. It scales to 3D with rendering and 6-DoF dynamics. We contribute a real-time inference that fits its probabilistic formulation.

SLAM through image synthesis The assumed generative model renders RGB-D images, which is related to SLAM through full-image synthesis. Traditional methods feature varied maps, from volumetric to surfels (e.g. [41, 42, 43, 44, 45, 46, 47, 48, 49]), and commonly estimate new camera poses by aligning new observations to a rendered image with variants of point-to-plane ICP with photometric consistency [50, 51, 52, 53, 44]. We extend this optimisation with dynamics in our approximate state filter [54]. A recent trend is to use implicit scene representations like NERF (e.g. [8, 55, 56]) with high rendering fidelity. Gradient-based pose inference through NERF-like rendering has received attention [57, 58], with iMAP [11] and NICE-SLAM [9] being two real-time

solutions. The mapping runtime of such methods is weighed down by optimisation through the renderer. Rendering can be sped up by decomposing parameters over space, e.g. by using voxels or primitives [59, 60, 61, 62, 63], but how to update neural maps in closed form remains unclear. Therefore, we rely on vanilla voxel grid maps [15, 64], as their probabilistic treatment and closed-form updates are straightforward, leaving implicit representations for future work. We note that none of the aforementioned methods incorporate dynamics and uncertainty, which distinguishes our approach.

Probabilistic SLAM inference SLAM filters are thoroughly explored for flat 2D modelling [65, 12, 66, 13, 67, 68, 69], but have been superseded by MAP smoothing in modern visual SLAM (e.g. [3, 4, 5, 70, 71, 72, 73]), primarily due to scalability concerns [6, 74]. However, as of now smoothing is not computationally feasible without sparsity assumptions. We therefore reexamine filtering for differentiable rendering, as we aim to obtain a dense map posterior with uncertainty in real-time (see appendix J for further motivation). Filters may benefit from the dense modelling of observations [74], which aligns with our objective, and we will demonstrate they can be a feasible solution for moderately-sized indoor environments. For the states, we use a Laplace approximation [75] and velocity updates similar to those in extended Kalman filters [12]. For the map, occupancy grids are a common probabilistic choice [64, 66, 76] and closed-form mapping has been used in that context [69]. To enable rendering we provide a similar derivation, but for a signed distance function (SDF), which is related. Probabilistic SDF mapping dates back to Curless and Levoy [41], and SDF updates have a well-known probabilistic interpretation [77, 78]. We use these approximations to arrive at a holistic probabilistic solution that scales to dense 3D modelling in real-time.

3 Overview

We approach on-line SLAM inference with two aims in mind. First, we want to harmonise our map and state estimation with a predictive model. Second, we want to quantify uncertainty: estimates and predictions should account for modelling inaccuracies as well as measurement and process noise. Both are important for autonomous decision making. To achieve this, we derive a Bayesian posterior in the probabilistic model of Mirchev et al. [15], to ensure that inference matches the forward model. Before we delve into our proposed solution, we present a practical summary. At every time step:

1. we point-estimate the agent's pose using gradient descent, involving geometry and dynamics.
2. we extend the pose with a Gaussian covariance matrix through a Laplace approximation.
3. with the pose, we estimate the agent's current velocity in closed form.
4. with the pose and the current observation, we update the map in closed form.

We use well-established methods for the above. In 1. we combine assumed density filtering [79], point-to-plane ICP [50] and photometric alignment [51, 52]. In 2. we use a Laplace approximation [75, 80]. In 3. we use linear-Gaussian updates, akin to Kalman filters [12]. In 4. we first derive generic closed-form map updates, which boil down to SDF updates [41] for our generative assumptions.

We contribute by deriving a holistic Bayesian inference from the generative model we started with. In doing so, we identify where traditional techniques are applicable to make a practical algorithm.

4 Methods

In the following we will denote generative distributions, true posterior distributions and conditionals with $p(\cdot)$. Respectively, approximate distributions will be denoted with $q(\cdot)$. Approximation steps will be indicated by $\approx$ in equations. We use $q^{\boldsymbol{\phi}}(\cdot)$ to subsume estimated distribution parameters into $\boldsymbol{\phi}$. A subscript $\cdot_t$ indicates that a variable or a distribution is different at every time step.

4.1 Background

We start with an overview of the generative model of Mirchev et al. [15] from which we will derive the inference. We assume a sequence of RGB-D observations $\mathbf{x}_{1:T}$ and a sequence of agent states

$\mathbf{z}_{1:T}$ driven by controls $\mathbf{u}_{1:T-1}$ form a Markovian state-space model. Each observation is constructed from a respective state with a rendering emission model $p(\mathbf{x}_t \mid \boldsymbol{\mathcal{M}}, \mathbf{z}_t)$, where $\boldsymbol{\mathcal{M}}$ is a global latent random variable for a dense map. A transition model $p(\mathbf{z}_t \mid \mathbf{z}_{t-1}, \mathbf{u}_{t-1})$ accounts for the agent dynamics, where $\mathbf{u}_t$ are known acceleration controls. Assuming $\mathbf{z}_1$ is given, the joint distribution is:

$$p(\boldsymbol{\mathcal{M}}, \mathbf{z}_{2:T}, \mathbf{x}_{1:T} \mid \mathbf{u}_{1:T-1}, \mathbf{z}_1) = p(\boldsymbol{\mathcal{M}})p(\mathbf{x}_1 \mid \boldsymbol{\mathcal{M}}, \mathbf{z}_1) \prod_{t=2}^{T} p(\mathbf{z}_t \mid \mathbf{z}_{t-1}, \mathbf{u}_{t-1})p(\mathbf{x}_t \mid \mathbf{z}_t, \boldsymbol{\mathcal{M}}).$$

The map is a 3D voxel grid of occupancy and color–each cell contains four values. The emission is fully-differentiable and performs volumetric raymarching, searching for a unique hit position at a surface along each ray [81]. The transition performs Euler integration, using the acceleration controls and maintained velocity from the latent state. Appendix B and the original paper have the details.

4.2 Posterior Choice

First we need to choose which posterior to approximate. For example, Mirchev et al. [15] approximate the full posterior over the map and *all* states $p(\boldsymbol{\mathcal{M}}, \mathbf{z}_{2:T} \mid \mathbf{x}_{1:T}, \mathbf{u}_{1:T-1}, \mathbf{z}_1)$ with variational inference [82]. While generic, this approach is slowed down by rendering at every optimisation step [54], and the inevitable stochastic optimisation demands multiple steps until convergence. In addition, estimating the posterior over all states scatters the optimisation budget across the whole trajectory.

To enable real-time inference we target an alternative posterior, the filter $p(\boldsymbol{\mathcal{M}}, \mathbf{z}_t \mid \mathbf{x}_{1:t}, \mathbf{u}_{1:t-1}, \mathbf{z}_1)$, as the last state belief is enough for planning ahead [2]. Since filters can be updated recursively [16, 80], we can use closed-form updates for fast inference. Still, maintaining the joint distribution is too costly because of the large dense 3D map $\boldsymbol{\mathcal{M}}$.[1] Instead, we approximate the two marginal filters:

$$q_t^{\phi}(\boldsymbol{\mathcal{M}}) \approx p(\boldsymbol{\mathcal{M}} \mid H_t) = p(\boldsymbol{\mathcal{M}} \mid \mathbf{x}_{1:t}, \mathbf{u}_{1:t-1}, \mathbf{z}_1)$$
$$q_t^{\phi}(\mathbf{z}_t) \approx p(\mathbf{z}_t \mid H_t) = p(\mathbf{z}_t \mid \mathbf{x}_{1:t}, \mathbf{u}_{1:t-1}, \mathbf{z}_1),$$

where $H_t = \mathbf{x}_{1:t}, \mathbf{u}_{1:t-1}, \mathbf{z}_1$. More details about this modelling choice can be found in appendix A. We draw attention to the shorthand notation $p(\ \cdot\ \mid H_t)$, which will appear again in the following.

4.3 Approximate Filtering

For both marginal filters, we will arrive at adequate approximations by reusing the following equation:

$$p(\boldsymbol{\mathcal{M}}, \mathbf{z}_t \mid H_t) \propto p(\mathbf{x}_t \mid \mathbf{z}_t, \boldsymbol{\mathcal{M}}) \int p(\mathbf{z}_t \mid \mathbf{z}_{t-1}, \mathbf{u}_{t-1}) p(\boldsymbol{\mathcal{M}}, \mathbf{z}_{t-1} \mid H_{t-1}) d\mathbf{z}_{t-1}, \tag{1}$$

This is a classic recursive expression of the Bayes filter [16]. Starting from each true marginal posterior, we will first expand the joint, then use eq. (1) and apply a set of approximations. Next we will discuss our final result, we defer the detailed derivation of both filters to appendices C and E.

4.3.1 Marginal Map Filter

We begin with the map approximation, starting from the true marginal Bayes filter:

$$\begin{aligned} p(\boldsymbol{\mathcal{M}} \mid H_t) &= \int p(\boldsymbol{\mathcal{M}}, \mathbf{z}_t \mid H_t)\, \mathrm{d}\mathbf{z}_t \\ &\propto \int p(\mathbf{x}_t \mid \mathbf{z}_t, \boldsymbol{\mathcal{M}}) \int p(\mathbf{z}_t \mid \mathbf{z}_{t-1}, \mathbf{u}_{t-1}) p(\boldsymbol{\mathcal{M}}, \mathbf{z}_{t-1} \mid H_{t-1})\, \mathrm{d}\mathbf{z}_{t-1}\, \mathrm{d}\mathbf{z}_t \\ &\approx p(\mathbf{x}_t \mid \hat{\mathbf{z}}_t, \boldsymbol{\mathcal{M}}) \times q_{t-1}^{\phi}(\boldsymbol{\mathcal{M}}) \qquad (2) \\ &\approx q(\boldsymbol{\mathcal{M}} \mid \mathbf{x}_t, \hat{\mathbf{z}}_t) \times q_{t-1}^{\phi}(\boldsymbol{\mathcal{M}}) =: q_t^{\phi}(\boldsymbol{\mathcal{M}}). \qquad (3) \end{aligned}$$

Equations (2) and (3) hide a few approximations detailed in appendix C. The resulting solution takes a nominal state sample $\hat{\mathbf{z}}_t$, with which a map update $q(\boldsymbol{\mathcal{M}} \mid \mathbf{x}_t, \hat{\mathbf{z}}_t)$ is applied to the previous map belief $q_{t-1}^{\phi}(\boldsymbol{\mathcal{M}})$. We set $\hat{\mathbf{z}}_t$ to the mean of the current state belief $q_t^{\phi}(\mathbf{z}_t)$. Accepting some bias, we do this for speed as it is our best guess for $\mathbf{z}_t$ without extra computation.[2] Intuitively, the map update

[1] E.g. the size of full-covariance Gaussian representations [12] or carrying multiple maps in parallel for a Rao-Blackwellised particle filter [13, 14] become prohibitive.

[2] Appendix F discusses this approximation further.

$q(\boldsymbol{\mathcal{M}} \mid \mathbf{x}_t, \hat{\mathbf{z}}_t)$ populates the map such that the observation $\mathbf{x}_t$ can be reconstructed. Our derivation of the updates is similar to the one by Grisetti et al. [69] for 2D occupancy maps, but now applied to 3D.

The above approximation is generic, agnostic to the specific map and rendering assumptions. In practice, we need a closed-form map update $q(\boldsymbol{\mathcal{M}} \mid \mathbf{x}_t, \hat{\mathbf{z}}_t)$ that is faithful to the emission $p(\mathbf{x}_t \mid \hat{\mathbf{z}}_t, \boldsymbol{\mathcal{M}})$. In this work, we follow Mirchev et al. [15] and use a Gaussian map that factorises over voxels:

$$q_t^{\boldsymbol{\phi}}(\boldsymbol{\mathcal{M}}) = \prod_{ijk} \mathcal{N}(\boldsymbol{\mathcal{M}}_{ijk} \mid \boldsymbol{\mu}_{ijk,t}^{\boldsymbol{\mathcal{M}}}, \mathrm{diag}((\boldsymbol{\sigma}_{ijk,t}^{\boldsymbol{\mathcal{M}}})^2)).$$

Here the indices ijk run over voxels in a 3D grid. For this specific representation and the assumed surface-based rendering, we identify that the map update $q(\boldsymbol{\mathcal{M}} \mid \mathbf{x}_t, \hat{\mathbf{z}}_t)$ can be implemented as a probabilistic signed distance function (SDF) update [41]. We provide the technical details in appendix D. SDF updates for voxel maps are a traditional concept in computer vision, and prior work has considered their probabilistic interpretation before [77, 78]. We contribute by identifying the place of such updates in a probabilistic filter that follows the generative model of [15]. A detailed discussion of how the above relates to classical SDF update equations can be found in appendix D.

The above approximations are motivated by the real-time constraint. For example, one could optimise eq. (2) directly with gradient descent through the renderer, but evaluating the emission is expensive and hinders accurate convergence on a budget. This is particularly true when uncertainty estimates are desirable, as optimisation would then be stochastic and gradients noisy [83]. In contrast, the derived one-shot map updates are meant to have a cost similar to emitting just once, while capturing uncertainty as well. We show some of the differences between the two approaches in section 5.3.

4.3.2 Marginal State Filter

Similarly, for the state filter we start from the true marginal and arrive at approximations via eq. (1):

$$\begin{aligned}
p(\mathbf{z}_t \mid H_t) &= \int p(\boldsymbol{\mathcal{M}}, \mathbf{z}_t \mid H_t)\, \mathrm{d}\boldsymbol{\mathcal{M}} \\
&\propto \int p(\mathbf{x}_t \mid \mathbf{z}_t, \boldsymbol{\mathcal{M}}) \int p(\mathbf{z}_t \mid \mathbf{z}_{t-1}, \mathbf{u}_{t-1}) p(\boldsymbol{\mathcal{M}}, \mathbf{z}_{t-1} \mid H_{t-1})\, \mathrm{d}\mathbf{z}_{t-1}\, \mathrm{d}\boldsymbol{\mathcal{M}} \\
&\approx p\Big(\mathbf{x}_t \mid \mathbf{z}_t^{\mathrm{pose}}, \hat{\boldsymbol{\mathcal{M}}}\Big) q_t(\mathbf{z}_t^{\mathrm{pose}} \mid \mathbf{u}_{t-1}, H_{t-1}) q_t(\mathbf{z}_t^{\mathrm{vel}} \mid \mathbf{z}_t^{\mathrm{pose}}, \mathbf{u}_{t-1}, H_{t-1}) && (4) \\
&\approx q_t^{\boldsymbol{\phi}}(\mathbf{z}_t^{\mathrm{pose}}) \times q_t(\mathbf{z}_t^{\mathrm{vel}} \mid \mathbf{z}_t^{\mathrm{pose}}, \mathbf{u}_{t-1}, H_{t-1}) =: \; q_t^{\boldsymbol{\phi}}(\mathbf{z}_t). && (5)
\end{aligned}$$

We detail all the approximations that lead to eq. (4) in appendix E. In eq. (4) we have three terms: an image reconstruction likelihood, a Gaussian pose prior and a linear Gaussian velocity conditional given a pose. The latter two we obtain analytically with a linear approximation of the transition model and the previous Gaussian belief $q_{t-1}^{\boldsymbol{\phi}}(\mathbf{z}_{t-1})$ (c.f. appendix E). First, using the first two terms of eq. (4) we define a maximum a-posteriori (MAP) objective for pose optimisation:

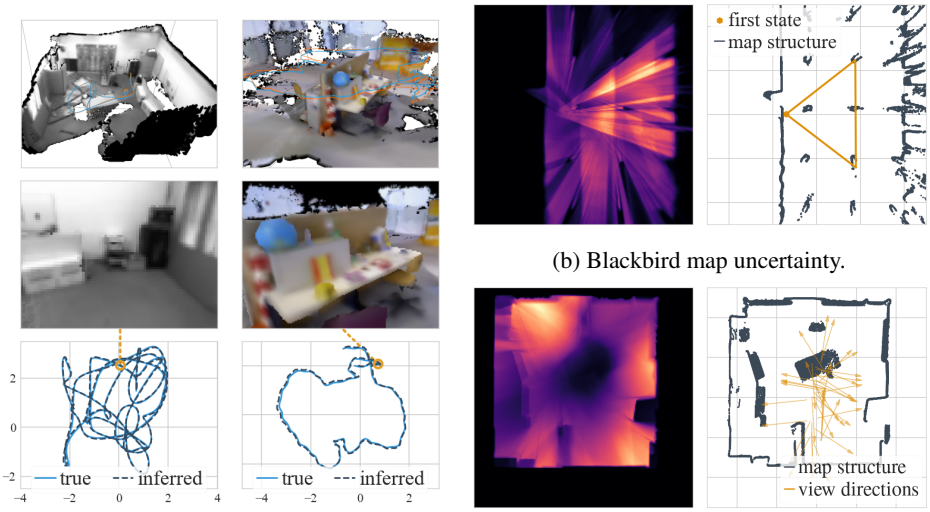
$$\arg\max_{\mathbf{z}_t^{\mathrm{pose}}} \; \log p\Big(\mathbf{x}_t \mid \hat{\boldsymbol{\mathcal{M}}}, \mathbf{z}_t^{\mathrm{pose}}\Big) + \log q_t(\mathbf{z}_t^{\mathrm{pose}} \mid \mathbf{u}_{t-1}, H_{t-1}).$$

Here, $\hat{\boldsymbol{\mathcal{M}}}$ is a nominal map sample set to the mean of the previous map belief $q_{t-1}^{\boldsymbol{\phi}}(\boldsymbol{\mathcal{M}})$.[3] The term $\log q_t(\mathbf{z}_t^{\mathrm{pose}} \mid \mathbf{u}_{t-1}, H_{t-1})$ is an approximate dynamics prior over the current pose, it makes the pose respect the transition model. The term $\log p(\mathbf{x}_t \mid \hat{\boldsymbol{\mathcal{M}}}, \mathbf{z}_t^{\mathrm{pose}})$ represents reconstructing the current observation, optimising it for the current pose will align the observation to the map. However, evaluating this rendering term in every gradient step is inefficient. Because of this, we replace it with the prediction-to-observation objective used by Kayalibay et al. [54], Nießner et al. [45], Newcombe et al. [84]. We refer to [54] for further motivation and we list the technical details in appendix E.

The above optimisation gives us a MAP pose estimate, which we denote with $\boldsymbol{\mu}_t^{\mathrm{pose}}$. Next, we apply a Laplace approximation [75] around it to obtain a full covariance matrix $\boldsymbol{\Sigma}_t^{\mathrm{pose}}$ which captures the curvature of the objective. This leaves us with a full Gaussian belief over the current pose:

$$q_t^{\boldsymbol{\phi}}(\mathbf{z}_t^{\mathrm{pose}}) = \mathcal{N}(\mathbf{z}_t^{\mathrm{pose}} \mid \boldsymbol{\mu}_t^{\mathrm{pose}}, \boldsymbol{\Sigma}_t^{\mathrm{pose}}).$$

[3] Appendix F discusses this approximation further.



(a) Example mapping and localisation

(b) Blackbird map uncertainty.

(c) EuRoC map uncertainty.

Figure 2: (a) 3D reconstruction, example emission and inferred trajectory for EuRoC/V102 and TUM-RGBD fr3/office. (b) Blackbird experiment. Top-down map uncertainty on the left, *black* is uncertain, *orange* is precise. Precision is highest in a triangle around the center, which is the camera frustum where the agent remains sitting on a platform for a long time, see the orange triangle amidst the map point cloud on the right. (c) Analogous EuRoC experiment. Map uncertainty is high outside of the room, at the center and behind the two structures on the left due to occlusion. The uncertainty in the center is high because the agent primarily looks outwards (view directions in the right image).

Finally, we can combine this Gaussian with the Gaussian velocity conditional $q_t(\mathbf{z}_t^{\text{vel}} \mid \mathbf{z}_t^{\text{pose}}, \mathbf{u}_{t-1}, H_{t-1})$ (the third term in eq. (4)) into a full-state belief in closed form:

$$q_t^{\phi}(\mathbf{z}_t) = \mathcal{N}(\mathbf{z}_t \mid \boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t) = \mathcal{N}(\mathbf{z}_t^{\text{pose}} \mid \boldsymbol{\mu}_t^{\text{pose}}, \boldsymbol{\Sigma}_t^{\text{pose}})\mathcal{N}\big(\mathbf{z}_t^{\text{vel}} \mid \mathbf{D}_t \mathbf{z}_t^{\text{pose}} + \mathbf{e}_t, \boldsymbol{\Sigma}_t^{\text{vel}}\big).$$

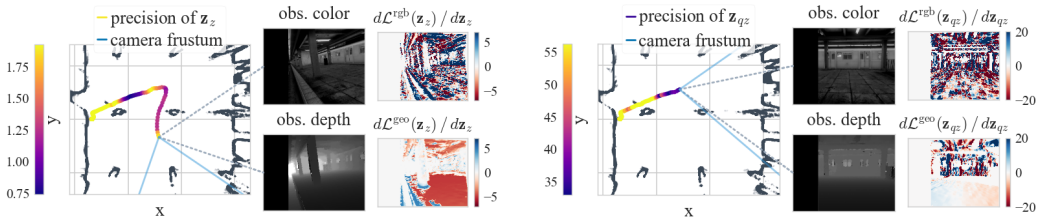
This is approximate, as $q_t(\mathbf{z}_t^{\text{vel}} \mid \mathbf{z}_t^{\text{pose}}, \mathbf{u}_{t-1}, H_{t-1})$ is not informed of the latest observation $\mathbf{x}_t$. We do it for speed and find it does not harm localisation in practice. Appendix E has the details.

5 Experiments

Originally we set out with a few goals: the inference method should be faithful to the generative assumptions, it should quantify uncertainty and it should run in real-time. What follows is an empirical analysis of these aspects. We evaluate on the EuRoC [17], Blackbird [18] and TUM-RGBD [19] data sets. The agent in the former two is an unmanned aerial vehicle (UAV), with speed of up to 4 m/s. For Blackbird, we use Semi-Global Block Matching (SGBM) for stereo depth estimation [85]. For EuRoC, we use the ground-truth Leica MS50 depth readings provided by [10]. We pretend the IMU readings from these data sets are our control inputs. For TUM-RGBD we do not feed in any controls and assume a constant-velocity transition. All experimental details are in appendix G.

5.1 Inference Through a Probabilistic Generative Model

First we look into the synergy between the inference and the generative assumptions. In fig. 2a we see mapping and localisation examples. The inferred scenes are consistent, with no dramatic offsets in geometry. More importantly, rendering from the inferred map using the emission $p(\mathbf{x}_t \mid \mathbf{z}_t, \boldsymbol{\mathcal{M}})$ works as expected (see middle row), indicating that map updates are consistent with the generative assumptions. This is evident from the accuracy of the inferred state trajectories as well (last row), as the pose optimisation objective from section 4.3.2 uses rendered images at every filtering step. A potential discrepancy between the inference and the generative assumptions would lead to errors that would accumulate over time, which is not the case.



(a) x location uncertainty

(b) $\mathbf{q}_z$ (yaw) orientation uncertainty

Figure 3: Inferred state uncertainty. Inferred trajectories are colored by precision (inverse uncertainty) of a certain state dimension, followed by observations, followed by columns of the tracking Jacobian for that same state dimension. (a) Here the precision in z (vertical movement) is high (*yellow*), because the z-orthogonal floor produces a consistent Jacobian (bottom right). (b) Here the precision in $\mathbf{q}_z$ orientation (yaw, azimuth) is low (*violet*), as there are no orthogonal surfaces (i.e. facing sideways). Note the low Jacobian magnitude of the horizontal floor this time (bottom right).

Map uncertainty The inferred map uncertainty is determined by the map updates. We show its interpretable effects in figs. 2b and 2c for two examples, one from Blackbird and another from EuRoC. Our map updates are akin to traditional SDF updates and the main factor that decides whether a map region is certain is how often it was observed. Regions that were occluded by objects, are behind walls or were rarely in view remain uncertain, e.g. as seen in fig. 2c. In contrast, if the agent spends a lot of time looking at a certain map region, the uncertainty there decreases, as seen in fig. 2b.

State uncertainty In fig. 3 we analyse state uncertainty by looking at the variance for individual dimensions. We notice that state uncertainty changes along the trajectory. Uncertainty is determined by what the agent currently sees, based on the geometric relationship between the agent movement and the observed scene (e.g. fig. 3a and fig. 3b). This effect can be explained if we examine the Laplace approximation used to estimate pose covariances. At any given time step, we set the covariance to $\mathbf{\Sigma}_t^{\text{pose}} \approx -\mathbf{H}^{-1} \approx -\left(2\mathbf{J}^T\mathbf{J}\right)^{-1}$. Here $\mathbf{H}$ is the Hessian of the tracking objective at the mean pose estimate and $\mathbf{J}$ is the Jacobian. The Jacobian connects the pose to all image pixel errors. The more consistent Jacobian entries are for a given pose dimension, the smaller the variance for that dimension will be. We refer to appendix H for more details about the map and state uncertainty quality.

5.2 Localisation Accuracy

We compare PRISM's localisation to state-of-the-art methods in moderately-sized indoor environments. We consider both baselines with dense maps (TANDEM [10], VSSM-LM [15], iMAP [11], NICE-SLAM [9], CodeVIO [86]) and sparse methods without rendering (ORB-SLAM2 [4], VINS [71], VIMO [70]). The results are in table 1. For the considered trajectories accuracy is comparable to the baselines, with differences of a few centimeters. At the same time, our inference boasts a predictive state-space model with both rendering and dynamics as well as uncertainty estimates, which is not common in the dense visual SLAM literature. Finally, in fig. 4 we see example inferred agent velocities, noting the uncertainty bands. This is possible because we model the agent dynamics.

Our localisation accuracy on Blackbird is better than the off-line variational inference results of VSSM-LM presented by Mirchev et al. [15], and at the same time our solution runs in real-time and also captures uncertainty. This shows the advantages of the proposed divide-and-conquer filtering.

5.3 Approximations for Runtime Improvement

All of our approximations are motivated by the real-time constraint, dictating the need for closed-form map updates, a Laplace approximation, linearisation assumptions and a surrogate pose optimisation objective. Figure 5 shows a runtime breakdown for different image resolutions, measured on an

[4]Last 10 s are skipped, as the drone hits the ground during landing.

Table 1: Localisation absolute error RMSE in meters on EuRoC [17], Blackbird [18] and TUM-RGBD [19].

Trajectory	Ours	Code VIO	TANDEM	ORB SLAM2
EuRoC/V101	0.041 ($\pm$ 0.002)	0.05	0.09	**0.031**
EuRoC/V102	0.035 ($\pm$ 0.002)	0.07	0.17	**0.02**
EuRoC/V103	**0.042 ($\pm$ 0.002)**	0.07	-	0.048
EuRoC/V201	**0.037 ($\pm$ 0.001)**	0.10	0.09	**0.037**
EuRoC/V202	**0.035 ($\pm$ 0.003)**	0.06	0.12	**0.035**
EuRoC/V203	x	**0.275**	-	x
Trajectory	**Ours**	**VSSM LM**	**VIMO**	**VINS**
picasso, 1 m/s	0.064 ($\pm$ 0.003)	0.139	**0.055**	0.097
picasso, 2 m/s	0.053 ($\pm$ 0.003)	0.136	**0.040**	0.043
picasso, 3 m/s	0.061 ($\pm$ 0.003)	0.120	**0.043**	0.045
picasso, 4 m/s	0.079 ($\pm$ 0.005)[4]	0.174	**0.049**	0.056
star, 1 m/s	0.089 ($\pm$ 0.007)[4]	0.137	**0.088**	0.102
star, 2 m/s	0.111 ($\pm$ 0.009)	0.163	**0.082**	0.133
star, 3 m/s	**0.115 ($\pm$ 0.012)**	0.281	0.183	0.235
star, 4 m/s	**0.153 ($\pm$ 0.015)**[4]	0.156	x	x
Trajectory	**Ours**	**iMAP**	**NICE SLAM**	**ORB SLAM2***
fr1/desk	0.053 ($\pm$ 0.003)	0.049	0.027	**0.016**
fr2/xyz	0.029 ($\pm$ 0.001)	0.02	**0.018**	0.04
fr3/office	0.083 ($\pm$ 0.001)	0.058	0.03	**0.01**

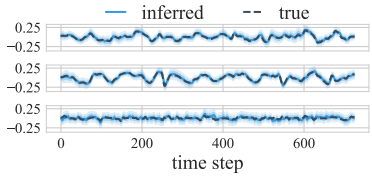


Figure 4: Inferred xyz-velocity.

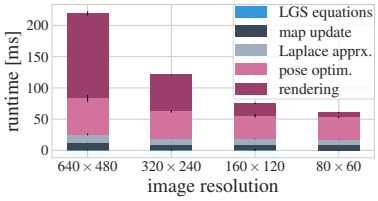


Figure 5: Runtime breakdown.

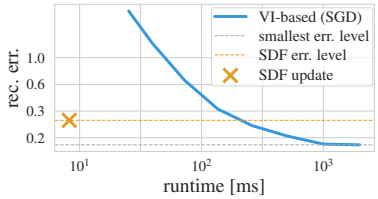


Figure 6: Mapping comparison.

NVIDIA 1080 Ti GPU and an Intel(R) Xeon(R) W-2123 CPU at 3.6 GHz. The heaviest operations are rendering and the gradient-based pose optimisation. Based on movement speed, rendering can happen periodically, whenever a new anchor image prediction for pose optimisation is needed. This leaves us with a total runtime of 10 Hz to 15 Hz, updating the map and state at every data step. In fig. 6 we also compare closed-form map updates to map inference via gradient-descent (e.g. as in [15, 8, 11, 9]). While gradient-descent is more accurate on a bigger budget, it is much more expensive. For example, to match the accuracy of the closed-form updates, which take less than 10 ms, one would need ca. 250 ms of optimisation, which is impractical. These runtimes are for a voxel grid that is significantly faster than neural representations [54], which would only exacerbate the problem.

6 Limitations and Conclusion

SDF voxel grids allow for closed-form updates, but their memory footprint limits the maximum resolution and scene size. Voxel hashing [45] or octrees [87] can directly replace them for memory efficiency. Neural maps and dynamically changing maps have remained out of our scope. Their probabilistic formulation and closed-form updates require further investigation. Our map factorises over voxels with no inter-region correlation, which could also be improved. PRISM provides interpretable uncertainty in real-time, but estimation is approximate. Obtaining perfectly calibrated uncertainty on a budget remains an open question (see appendix H). While filtering works for our generative assumptions indoors, filters cannot revisit past errors and can drift in large scenes with high levels of exploration [6]. We leave large-scale inference considerations for future work.

We have introduced PRISM, a method for probabilistic filtering in a predefined spatial state-space model. Our solution runs in real-time, provides state and map uncertainty, and infers a dense map and a 6-DoF state trajectory with velocities. It is comparably accurate to state-of-the-art SLAM in indoor environments. To the best of our knowledge this is the first real-time fully-probabilistic solution for SLAM that combines differentiable rendering and agent dynamics. We validated our method on three challenging data sets, featuring unmanned aerial vehicles and a handheld camera. The results are promising, establishing PRISM as a viable state estimator for downstream model-based control.

Acknowledgments

We thank our reviewers for the thoughtful discussion, it helped us to better position our contribution.

References

[1] D. Koller and N. Friedman. *Probabilistic Graphical Models - Principles and Techniques*. MIT Press, 2009.

[2] D. P. Bertsekas. *Dynamic programming and optimal control, 3rd Edition*. Athena Scientific, 2005. ISBN 1886529264. URL http://www.worldcat.org/oclc/314894080.

[3] J. Engel, T. Schöps, and D. Cremers. Lsd-slam: Large-scale direct monocular slam. In *European conference on computer vision*, pages 834–849. Springer, 2014.

[4] R. Mur-Artal and J. D. Tardós. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE Transactions on Robotics*, 33(5):1255–1262, 2017.

[5] J. Engel, V. Koltun, and D. Cremers. Direct sparse odometry. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Mar. 2018.

[6] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Transactions on robotics*, 32(6):1309–1332, 2016.

[7] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on information theory*, 47(2):498–519, 2001.

[8] 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. In *ECCV*, 2020.

[9] Z. Zhu, S. Peng, V. Larsson, W. Xu, H. Bao, Z. Cui, M. R. Oswald, and M. Pollefeys. Nice-slam: Neural implicit scalable encoding for slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2022.

[10] L. Koestler, N. Yang, N. Zeller, and D. Cremers. Tandem: Tracking and dense mapping in real-time using deep multi-view stereo. In *Conference on Robot Learning (CoRL)*, 2021.

[11] E. Sucar, S. Liu, J. Ortiz, and A. J. Davison. imap: Implicit mapping and positioning in real-time, 2021.

[12] R. E. Kalman et al. A new approach to linear filtering and prediction problems [j]. *Journal of basic Engineering*, 82(1):35–45, 1960.

[13] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. Fastslam: A factored solution to the simultaneous localization and mapping problem. In *AAAI/IAAI*, 2002.

[14] G. Grisetti, C. Stachniss, and W. Burgard. Improved techniques for grid mapping with rao-blackwellized particle filters. *IEEE transactions on Robotics*, 23(1):34–46, 2007.

[15] A. Mirchev, B. Kayalibay, P. van der Smagt, and J. Bayer. Variational state-space models for localisation and dense 3d mapping in 6 dof. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=XAS3uKeFWj.

[16] S. Särkkä. *Bayesian Filtering and Smoothing*. Number 3. Cambridge University Press, USA, 2013. ISBN 1107619289.

[17] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart. The euroc micro aerial vehicle datasets. *The International Journal of Robotics Research*, 35 (10):1157–1163, 2016.

[18] A. Antonini, W. Guerra, V. Murali, T. Sayre-McCord, and S. Karaman. The blackbird uav dataset. *The International Journal of Robotics Research*, 0(0):0278364920908331, 2020. doi:10.1177/0278364920908331.

[19] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of rgb-d slam systems. In *Proc. of the International Conference on Intelligent Robot Systems (IROS)*, Oct. 2012.

[20] L. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989. doi:10.1109/5.18626.

[21] M. Karl, M. Soelch, J. Bayer, and P. van der Smagt. Deep variational bayes filters: Unsupervised learning of state space models from raw data. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

[22] R. G. Krishnan, U. Shalit, and D. Sontag. Deep kalman filters. *arXiv preprint arXiv:1511.05121*, 2015.

[23] J. Chung, K. Kastner, L. Dinh, K. Goel, A. C. Courville, and Y. Bengio. A recurrent latent variable model for sequential data. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015. URL https://proceedings.neurips.cc/paper/2015/file/b618c3210e934362ac261db280128c22-Paper.pdf.

[24] S. Chiappa, S. Racanière, D. Wierstra, and S. Mohamed. Recurrent environment simulators. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=B1s6xvqlx.

[25] D. Ha and J. Schmidhuber. Recurrent world models facilitate policy evolution. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 2450–2462. Curran Associates, Inc., 2018.

[26] M. Deisenroth and C. E. Rasmussen. Pilco: A model-based and data-efficient approach to policy search. In *Proceedings of the 28th International Conference on machine learning (ICML-11)*, pages 465–472. Citeseer, 2011.

[27] D. Hafner, T. Lillicrap, I. Fischer, R. Villegas, D. Ha, H. Lee, and J. Davidson. Learning latent dynamics for planning from pixels. In *International Conference on Machine Learning*, pages 2555–2565. PMLR, 2019.

[28] D. Hafner, T. Lillicrap, J. Ba, and M. Norouzi. Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=S1lOTC4tDS.

[29] A. X. Lee, A. Nagabandi, P. Abbeel, and S. Levine. Stochastic latent actor-critic: Deep reinforcement learning with a latent variable model. *Advances in Neural Information Processing Systems*, 33:741–752, 2020.

[30] P. Becker-Ehmck, M. Karl, J. Peters, and P. van der Smagt. Learning to fly via deep model-based reinforcement learning, 2020.

[31] M. Karl, M. Soelch, P. Becker-Ehmck, D. Benbouzid, P. van der Smagt, and J. Bayer. Unsupervised real-time control through variational empowerment. *arXiv preprint arXiv:1710.05101*, 2017.

[32] D. Corneil, W. Gerstner, and J. Brea. Efficient model-based deep reinforcement learning with variational state tabulation. volume 80 of *Proceedings of Machine Learning Research*, pages 1049–1058, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018. PMLR.

[33] K. Chua, R. Calandra, R. McAllister, and S. Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. In *Advances in Neural Information Processing Systems*, pages 4754–4765, 2018.

[34] M. Fraccaro, D. J. Rezende, Y. Zwols, A. Pritzel, S. M. A. Eslami, and F. Viola. Generative temporal models with spatial memory for partially observed environments. *CoRR*, abs/1804.09401, 2018.

[35] S. M. A. Eslami, D. Jimenez Rezende, F. Besse, F. Viola, A. S. Morcos, M. Garnelo, A. Ruderman, A. A. Rusu, I. Danihelka, K. Gregor, D. P. Reichert, L. Buesing, T. Weber, O. Vinyals, D. Rosenbaum, N. Rabinowitz, H. King, C. Hillier, M. Botvinick, D. Wierstra, K. Kavukcuoglu, and D. Hassabis. Neural scene representation and rendering. *Science*, 360(6394):1204–1210, 2018. doi:10.1126/science.aar6170. URL https://science.sciencemag.org/content/360/6394/1204.

[36] K. Gregor, D. J. Rezende, F. Besse, Y. Wu, H. Merzic, and A. van den Oord. Shaping belief states with generative environment models for rl. In *Advances in Neural Information Processing Systems*, pages 13475–13487, 2019.

[37] A. Mirchev, B. Kayalibay, M. Soelch, P. van der Smagt, and J. Bayer. Approximate bayesian inference in spatial environments. In *Proceedings of Robotics: Science and Systems*, Freiburgim-Breisgau, Germany, June 2019.

[38] S. Gupta, V. Tolani, J. Davidson, S. Levine, R. Sukthankar, and J. Malik. Cognitive mapping and planning for visual navigation, 2019.

[39] E. Parisotto and R. Salakhutdinov. Neural map: Structured memory for deep reinforcement learning. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=Bk9zbyZCZ.

[40] D. S. Chaplot, D. Gandhi, S. Gupta, A. Gupta, and R. Salakhutdinov. Learning to explore using active neural slam. In *International Conference on Learning Representations (ICLR)*, 2020.

[41] B. Curless and M. Levoy. A volumetric method for building complex models from range images. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 303–312, 1996.

[42] J. C. Carr, R. K. Beatson, J. B. Cherrie, T. J. Mitchell, W. R. Fright, B. C. McCallum, and T. R. Evans. Reconstruction and representation of 3d objects with radial basis functions. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 67–76, 2001.

[43] R. A. Newcombe, S. Lovegrove, and A. J. Davison. Dtam: Dense tracking and mapping in real-time. *2011 International Conference on Computer Vision*, pages 2320–2327, 2011.

[44] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *2011 10th IEEE international symposium on mixed and augmented reality*, pages 127–136. IEEE, 2011.

[45] M. Nießner, M. Zollhöfer, S. Izadi, and M. Stamminger. Real-time 3d reconstruction at scale using voxel hashing. *ACM Transactions on Graphics (ToG)*, 32(6):1–11, 2013.

[46] M. Keller, D. Lefloch, M. Lambers, S. Izadi, T. Weyrich, and A. Kolb. Real-time 3d reconstruction in dynamic scenes using point-based fusion. In *2013 International Conference on 3D Vision - 3DV 2013*, pages 1–8, 2013. doi:10.1109/3DV.2013.9.

[47] T. Whelan, S. Leutenegger, R. F. Salas-Moreno, B. Glocker, and A. J. Davison. Elasticfusion: Dense slam without a pose graph. In *Robotics: Science and Systems*, 2015.

[48] Y.-P. Cao, L. P. Kobbelt, and S. Hu. Real-time high-accuracy three-dimensional reconstruction with consumer rgb-d cameras. *ACM Transactions on Graphics (TOG)*, 37:1 – 16, 2018.

[49] Y. Xu, L. Nan, L. Zhou, J. Wang, and C. C. Wang. Hrbf-fusion: Accurate 3d reconstruction from rgb-d data using on-the-fly implicits. *ACM Transactions on Graphics (TOG)*, 41(3):1–19, 2022.

[50] Y. Chen and G. Medioni. Object modelling by registration of multiple range images. *Image and vision computing*, 10(3):145–155, 1992.

[51] F. Steinbrücker, J. Sturm, and D. Cremers. Real-time visual odometry from dense rgb-d images. In *2011 IEEE international conference on computer vision workshops (ICCV Workshops)*, pages 719–722. IEEE, 2011.

[52] C. Audras, A. Comport, M. Meilland, and P. Rives. Real-time dense appearance-based slam for rgb-d sensors. In *Australasian Conf. on Robotics and Automation*, volume 2, pages 2–2, 2011.

[53] C. Kerl, J. Sturm, and D. Cremers. Dense visual slam for rgb-d cameras. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2100–2106. IEEE, 2013.

[54] B. Kayalibay, A. Mirchev, P. van der Smagt, and J. Bayer. Tracking and planning with spatial world models. *arXiv preprint arXiv:2201.10335*, 2022.

[55] V. Sitzmann, J. Martel, A. Bergman, D. Lindell, and G. Wetzstein. Implicit neural representations with periodic activation functions. *Advances in Neural Information Processing Systems*, 33, 2020.

[56] J. Ortiz, A. Clegg, J. Dong, E. Sucar, D. Novotny, M. Zollhoefer, and M. Mukadam. isdf: Real-time neural signed distance fields for robot perception. *arXiv preprint arXiv:2204.02296*, 2022.

[57] L. Yen-Chen, P. Florence, J. T. Barron, A. Rodriguez, P. Isola, and T.-Y. Lin. Inerf: Inverting neural radiance fields for pose estimation, 2021.

[58] Z. Wang, S. Wu, W. Xie, M. Chen, and V. A. Prisacariu. Nerf–: Neural radiance fields without known camera parameters, 2021.

[59] C. Reiser, S. Peng, Y. Liao, and A. Geiger. Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps, 2021.

[60] S. Lombardi, T. Simon, J. Saragih, G. Schwartz, A. Lehrmann, and Y. Sheikh. Neural volumes. *ACM Transactions on Graphics*, 38(4):1–14, Jul 2019. ISSN 1557-7368. doi:10.1145/3306346.3323020. URL `http://dx.doi.org/10.1145/3306346.3323020`.

[61] L. Liu, J. Gu, K. Z. Lin, T.-S. Chua, and C. Theobalt. Neural sparse voxel fields. *NeurIPS*, 2020.

[62] S. Lombardi, T. Simon, G. Schwartz, M. Zollhoefer, Y. Sheikh, and J. Saragih. Mixture of volumetric primitives for efficient neural rendering. *ACM Trans. Graph.*, 40(4), jul 2021. ISSN 0730-0301. doi:10.1145/3450626.3459863. URL `https://doi.org/10.1145/3450626.3459863`.

[63] T. Müller, A. Evans, C. Schied, and A. Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 41(4):102:1–102:15, July 2022. doi:10.1145/3528223.3530127. URL `https://doi.org/10.1145/3528223.3530127`.

[64] H. Moravec and A. Elfes. High resolution maps from wide angle sonar. In *Proceedings. 1985 IEEE International Conference on Robotics and Automation*, volume 2, pages 116–121. IEEE, 1985.

[65] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005. ISBN 0262201623.

[66] K. P. Murphy. Bayesian map learning in dynamic environments. In *Advances in Neural Information Processing Systems 12, [NIPS Conference, Denver, Colorado, USA, November 29 - December 4, 1999]*, pages 1015–1021, 1999.

[67] D. Hahnel, W. Burgard, D. Fox, and S. Thrun. An efficient fastslam algorithm for generating maps of large-scale cyclic environments from raw laser range measurements. In *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003) (Cat. No.03CH37453)*, volume 1, pages 206–211 vol.1, 2003. doi:10.1109/IROS.2003.1250629.

[68] M. Montemerlo, S. Thrun, D. Koller, B. Wegbreit, et al. Fastslam 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges. In *IJCAI*, volume 3, pages 1151–1156, 2003.

[69] G. Grisetti, C. Stachniss, and W. Burgard. Improving grid-based slam with rao-blackwellized particle filters by adaptive proposals and selective resampling. *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pages 2432–2437, 2005.

[70] B. Nisar, P. Foehn, D. Falanga, and D. Scaramuzza. Vimo: Simultaneous visual inertial model-based odometry and force estimation. In *Proceedings of Robotics: Science and Systems*, FreiburgimBreisgau, Germany, June 2019.

[71] T. Qin, P. Li, and S. Shen. Vins-mono: A robust and versatile monocular visual-inertial state estimator. *IEEE Transactions on Robotics*, 34(4):1004–1020, 2018.

[72] A. Rosinol, T. Sattler, M. Pollefeys, and L. Carlone. Incremental visual-inertial 3d mesh generation with structural regularities. In *2019 International Conference on Robotics and Automation (ICRA)*, 2019.

[73] A. Rosinol, M. Abate, Y. Chang, and L. Carlone. Kimera: an open-source library for real-time metric-semantic localization and mapping. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2020. URL `https://github.com/MIT-SPARK/Kimera`.

[74] H. Strasdat, J. M. Montiel, and A. J. Davison. Visual slam: why filter? *Image and Vision Computing*, 30(2):65–77, 2012.

[75] P. S. Laplace. Memoir on the probability of the causes of events. *Statistical Science*, 1(3): 364–378, 1986. ISSN 08834237. URL `http://www.jstor.org/stable/2245476`.

[76] R. Senanayake and F. Ramos. Bayesian hilbert maps for dynamic continuous occupancy mapping. In *Proceedings of the 1st Annual Conference on Robot Learning*, volume 78 of *Proceedings of Machine Learning Research*, pages 458–471. PMLR, 13–15 Nov 2017.

[77] C. Hernández, G. Vogiatzis, and R. Cipolla. Probabilistic visibility for multi-view stereo. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2007.

[78] W. Dong, Q. Wang, X. Wang, and H. Zha. Psdf fusion: Probabilistic signed distance function for on-the-fly 3d data fusion and scene reconstruction. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 701–717, 2018.

[79] M. Opper and O. Winther. A bayesian approach to on-line learning. 1999.

[80] C. M. Bishop. *Pattern recognition and machine learning*. springer, 2006.

[81] S. Parker, P. Shirley, Y. Livnat, C. Hansen, and P.-P. Sloan. Interactive ray tracing for isosurface rendering. In *Proceedings Visualization'98 (Cat. No. 98CB36276)*, pages 233–238. IEEE, 1998.

[82] D. P. Kingma and M. Welling. Auto-encoding variational bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014. URL http://arxiv.org/abs/1312.6114.

[83] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe. Variational inference: A review for statisticians. *Journal of the American statistical Association*, 112(518):859–877, 2017.

[84] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohli, J. Shotton, S. Hodges, and A. W. Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. *2011 10th IEEE International Symposium on Mixed and Augmented Reality*, pages 127–136, 2011.

[85] H. Hirschmuller. Stereo processing by semiglobal matching and mutual information. *IEEE Transactions on pattern analysis and machine intelligence*, 30(2):328–341, 2007.

[86] X. Zuo, N. Merrill, W. Li, Y. Liu, M. Pollefeys, and G. Huang. Codevio: Visual-inertial odometry with learned optimizable dense depth. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 14382–14388. IEEE, 2021.

[87] F. Steinbrucker, C. Kerl, and D. Cremers. Large-scale multi-resolution surface reconstruction from rgb-d sequences. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3264–3271, 2013.

[88] J. Bradbury, R. Frostig, P. Hawkins, M. J. Johnson, C. Leary, D. Maclaurin, G. Necula, A. Paszke, J. VanderPlas, S. Wanderman-Milne, and Q. Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL http://github.com/google/jax.

[89] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

[90] L. V. Jospin, H. Laga, F. Boussaid, W. Buntine, and M. Bennamoun. Hands-on bayesian neural networks—a tutorial for deep learning users. *IEEE Computational Intelligence Magazine*, 17 (2):29–48, 2022.

[91] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. MIT press, 2018.