Supplementary Materials for SLAIM: Robust Dense Neural SLAM for Online Tracking and Mapping

Vincent Cartillier[1], Grant Schindler[2], Irfan Essa[1,2]
[1]Georgia Tech [2] Google Research
vcartillier3@gatech.edu
vincentcartillier.github.io/slaim.html

1. Additional implementation details

1.1. Derivation of the KL regularizer

In order to model the ray termination distribution we start by representing the density $\tilde{\sigma}$ as a narrow bell shaped function using the $sech^2$ function. We use the $sech^2$ over a Gaussian for its simplicity when computing integrals. We define the estimated density as follow:

$$\tilde{\sigma}(r) = S \times sech^2\left(\frac{(r-d')}{\sigma_d}\right) \tag{1}$$

With d' and σ_d the mean and variance and S a scale factor. The final ray termination distribution is then computed using the following equation: $\tilde{w}(r) = T(r)\tilde{\sigma}(r)$, with $T(r) = exp(-\int_0^r \tilde{\sigma}(s)ds)$. We intent to have an analytical formulation of $\tilde{w}(r)$ so it is easier and faster to compute the resulting KL regularizer. In order to do this we need to compute the integral in $T(r)$, $\int_0^r \tilde{\sigma}(s)ds$. We start with the following,

$$\begin{aligned}\int_0^r \tilde{\sigma}(s)ds &= \int_0^r S \times sech^2\left(\frac{(s-d')}{\sigma_d}\right) ds \\ &= S\sigma_d \left[tanh\left(\frac{(s-d')}{\sigma_d}\right)\right]_0^r \\ &= S\sigma_d \left[tanh\left(\frac{(r-d')}{\sigma_d}\right) - tanh\left(\frac{-d'}{\sigma_d}\right)\right]\end{aligned} \tag{2}$$

We can plug the result of Eq 2 into our custom ray termination distribution and express $\tilde{w}$ analytically as follow

$$\begin{aligned}\tilde{w}(r) &= \tilde{\sigma}(r)T(r) \\ &= \tilde{\sigma}(r)e^{-S\sigma_d\left[tanh\left(\frac{(r-d')}{\sigma_d}\right) - tanh\left(\frac{-d'}{\sigma_d}\right)\right]} \\ &= S.sech^2\left(\frac{(r-d')}{\sigma_d}\right) e^{-S\sigma_d\left[tanh\left(\frac{(r-d')}{\sigma_d}\right) - tanh\left(\frac{-d'}{\sigma_d}\right)\right]}\end{aligned} \tag{3}$$

We can now use that estimation of a ray termination distribution in a KL regularizer.

$$\mathcal{L}_{KL} = -\frac{1}{N}\sum_{n=1}^{N}\sum_k log(w(r_k))\tilde{w}(r_k)\Delta_r \tag{4}$$

With N the number of rays per batch, Δ_r the ray marching step size and $\tilde{w}$ the predicted ray termination distribution. Note here that although a we could have represented the $\tilde{\sigma}$ function with a Gaussian, we wouldn't have been able to compute an analytical formulation of the integral on Eq. 2, and estimating such integral would have been computationally costly.

1.2. Expectation of the custom ray termination distribution

It is important that the expectation of the ray termination distribution be centered on the depth measurement d in order to ensure a good quality of reconstruction. The custom ray termination distriubtion involves a mean d' and variance σ_d parameters. Naively setting $d' = d$ would lead to a bias estimator. We can compute the expected value of the distribution $\tilde{w}$ in order to set the parameter d'.

$$E_r(\tilde{w}) = \int_{-\infty}^{\infty} r.\tilde{w}(r)dr \tag{5}$$

When reusing the results in Eq. 3 and by setting $y = \frac{(r-d')}{\sigma_d}$ and $C = tanh\left(\frac{-d'}{\sigma_d}\right)$ we have,

$$\begin{aligned}E_r(\tilde{w}) &= \int_{-\infty}^{\infty} \sigma_d(y\sigma_d + d').S.sech^2(y)e^{-S\sigma_d[tanh(y)-C]}dy \\ &= S\sigma_d^2 \int_{-\infty}^{\infty} y.sech^2(y)e^{-S\sigma_d[tanh(y)-C]}dy \\ &+ S\sigma_d d' \int_{-\infty}^{\infty} sech^2(y)e^{-S\sigma_d[tanh(y)-C]}dy \\ &= S\sigma_d^2 \times \mathcal{A} + S\sigma_d d' \times \mathcal{B}\end{aligned} \tag{6}$$

With $\mathcal{A}$ and $\mathcal{B}$ the two integrals. In addition, we can notice that $d' \gg \sigma_d$ therefore we can simplify $C = tanh\left(\frac{-d'}{\sigma_d}\right) \approx -1$. We can now rewrite the two integrals $\mathcal{A}$ and $\mathcal{B}$ as two functions that do not depend on d'.

$$\begin{aligned} \mathcal{A} &= \int_{-\infty}^{\infty} y.sech^2(y) e^{-S\sigma_d[tanh(y)+1]} dy \\ \mathcal{B} &= \int_{-\infty}^{\infty} sech^2(y) e^{-S\sigma_d[tanh(y)+1]} dy \end{aligned} \tag{7}$$

In practice computing these integrals is difficult. However, we only need to evaluate them once. Therefore, we estimate these integrals using a sampling based approach at the beginning of each run. We can then infer the parameter d' by setting the expectation value to the depth measurement d, i.e. $E_r(\tilde{w}) = d$. We then find,

$$d' = \frac{d - S\sigma_d^2 \times \mathcal{A}}{S\sigma_d \times \mathcal{B}} \tag{8}$$

1.3. Hyperparameters

We detail here the hyperparameters and network structure used in SLAIM. We build our code upon the existing Instant-NGP framework [3]. Our NeRF model uses hash-grid features with 16 levels and 2 features per levels. The base resolution of the feature grid is set to $R_{min} = 16$, and we set the max resolution R_{max} to the next power of 2 value such that it corresponds to a resolution of 1 cm for Replica [5], 2cm for TUM [6] and 4cm for ScanNet [1]. We do not use the ray direction as input to our network. Both MLP decoders have one hidden layer with 32 neurons. We use the Adam optimizer [2] with a learning rate of $1.e^{-2}$ without any scheduler. In TUM and ScanNet experiments we use a subset of $M = 5$ keyframes for local bundle adjustment. We do not perform local bundle adjustment in Replica.

TUM settings. We use an $S = 10000$ and $\sigma_d = 0.02$ in the custom ray termination distribution. The camera pose parameters' learning rate is set to $1.e^{-3}$ during tracking and $5.e^{-4}$ during mapping. We perform 15 tracking iterations with a GPL of 2: 5 iterations at each levels 2, 1 and 0. During the mapping phase we perform 15 iterations of local bundle-adjustment first and 15 additional global bundle adjustment iterations using a GPL of 2 (ie. 5 iterations at each levels 2 to 0). We sample $N_t = N_m = 2048$ rays for both mapping and tracking, we use $\lambda_d = 1$ and $\lambda_{KL} = 10.$.

ScanNet settings. We use an $S = 5000$ and $\sigma_d = 0.04$ in the custom ray termination distribution. The camera pose parameters' learning rate is set to $1.e^{-3}$ during tracking and $5.e^{-4}$ during mapping. We perform 15 tracking iterations with a GPL of 2: 5 iterations at each levels 2, 1 and 0. During the mapping phase we perform 15 iterations of local bundle-adjustment first and 15 additional global bundle adjustment iterations using a GPL of 2 (ie. 5 iterations at each levels 2 to 0). We sample $N_t = 2048$ and $N_m = 4096$ rays for tracking and mapping respectively, and we use $\lambda_d = 1$ and $\lambda_{KL} = 1.$.

Replica settings. We use an $S = 10000$ and $\sigma_d = 0.01$ in the custom ray termination distribution. The camera pose parameters' learning rate is set to $1.e^{-3}$ during tracking and $5.e^{-4}$ during mapping. We perform 10 tracking iterations with a GPL of 1: 5 iterations at each levels 1 and 0. During the mapping phase we perform 20 iterations of global bundle-adjustment only using a GPL of 1 (ie. 10 iterations at each levels 1 and 0). We sample $N_t = N_m = 4096$ rays for tracking and mapping, and we use $\lambda_d = 1$ and $\lambda_{KL} = 1.$.

1.4. Camera tracking evaluation protocol

To evaluate the camera tracking performances we first perform a global alignment between the generated camera poses and the ground-truth ones to alleviate any downgrading effects due to global shifts. This is a common practice in evaluation of SLAM systems [4, 7–9]. We further compute the absolute translation error (ATE) as the RMSE between the ground-truth poses and the aligned generated ones.

1.5. 3D reconstruction elvaluation protocole

In the realm of neural implicit reconstruction and SLAM, it is necessary to perform an additional mesh culling step to limit the extrapolation capabilities of NeRf when rendering a mesh outside of the camera view frustrum. We adopt the same strategy as in Co-SLAM [8]. We compute the global camera viewing frustrum given the set of camera poses plus additional poses generated to handle occlusions. We then remove any vertices outside of that viewing frustrum. This is a simple yet effective technique to limit reconstruction artifacts outside of the targeted region.

2. Additional details on the $Ours_{MG}$ experiments.

In our experiments, we compared our SLAIM model to another baseline, that we refer to as $SLAIM_{MG}$ (MG stands for max-grid). This baseline also does coarse-to-fine tracking by rendering blurry images in the early tracking iterations. To construct blurry images $SLAIM_{MG}$ renders images using different grid-resolution features. We use a max-grid-level parameter $mgl \leq L$ that we use to set a max resolution the network is allowed to use to construct the position embedding y. Grid features for higher levels are set to 0, and the final positional encoding can be written as $y = \left[h_\beta^1(x), ..., h_\beta^{mgl}(x), 0\right]$. This implementation is interesting because it requires no extra training steps or additional FLOPs operations during tracking or mapping. However, from our results on the TUM dataset [6], we observe that this baseline performs worse than previous baselines. We hypothesis that the reason is because the generated blurry

images contain reconstruction artifacts. We show some of these reconstructions defaults in Fig. 1. For instance in the lowest resolution we observe a blue area near the top left part of the image which is inconsistent with the original image. We believe that this phenomena is due to the MLP decoder trying to overcompensate the lack of high-frequency features.

References

[1] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5828–5839, 2017. 2

[2] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 2

[3] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *arXiv preprint arXiv:2201.05989*, 2022. 2

[4] Raul Mur-Artal and Juan 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. 2

[5] Julian Straub, Thomas Whelan, Lingni Ma, Yufan Chen, Erik Wijmans, Simon Green, Jakob J Engel, Raul Mur-Artal, Carl Ren, Shobhit Verma, et al. The replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797*, 2019. 2

[6] Jürgen Sturm, Nikolas Engelhard, Felix Endres, Wolfram Burgard, and Daniel Cremers. A benchmark for the evaluation of rgb-d slam systems. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pages 573–580. IEEE, 2012. 2

[7] Edgar Sucar, Shikun Liu, Joseph Ortiz, and Andrew J Davison. imap: Implicit mapping and positioning in real-time. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6229–6238, 2021. 2

[8] Hengyi Wang, Jingwen Wang, and Lourdes Agapito. Co-slam: Joint coordinate and sparse parametric encodings for neural real-time slam. *arXiv preprint arXiv:2304.14377*, 2023. 2

[9] Zihan Zhu, Songyou Peng, Viktor Larsson, Weiwei Xu, Hujun Bao, Zhaopeng Cui, Martin R Oswald, and Marc Pollefeys. Nice-slam: Neural implicit scalable encoding for slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12786–12796, 2022. 2

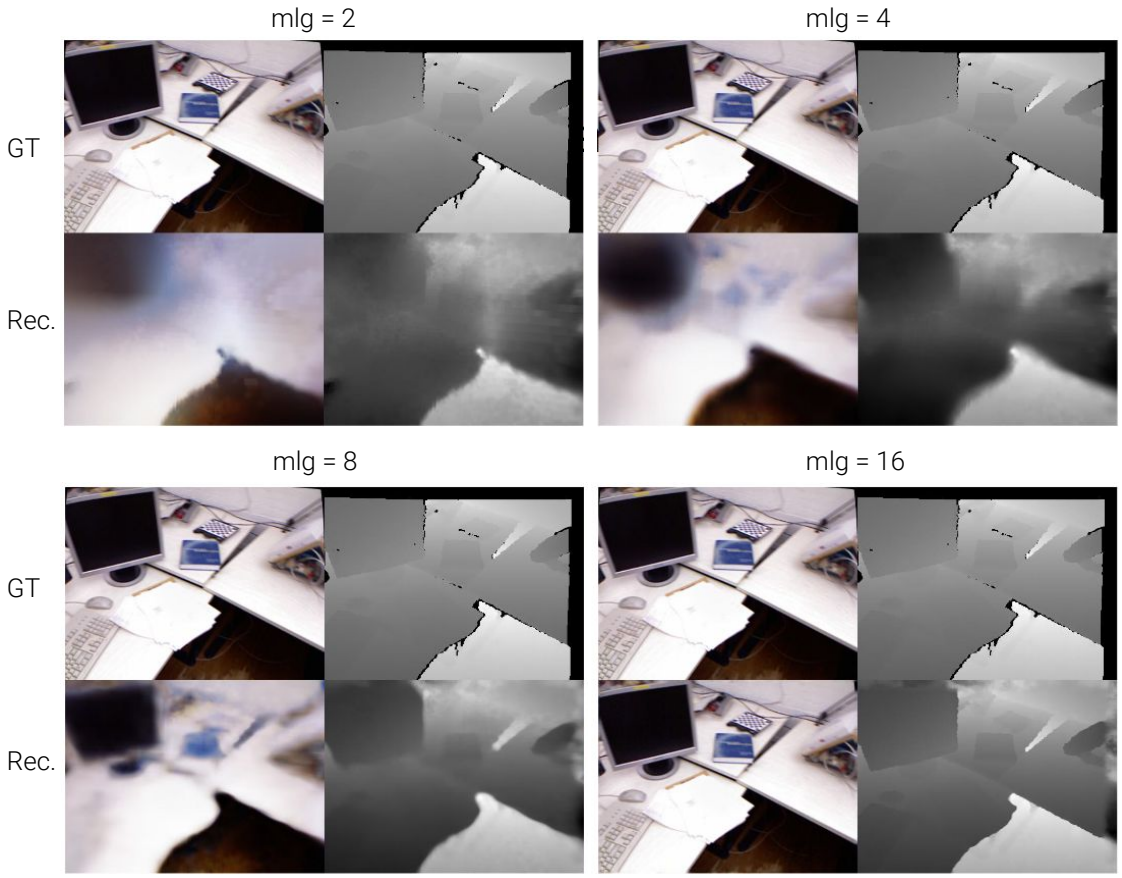

Figure 1. Visualization of 2D NeRF reconstructions of the $SLAIM_{MG}$ model with different mgl levels. Each sub-figures has ground-truth views on the top and reconstructions at the bottom. We show different reconstructions with mgl levels 2, 4, 8, and 16. We observe that the low resolution reconstruction contain artifacts such as the blue area under the monitor in the $mgl = 2$ figure. In addition the table edge appears more blurry in $mgl = 4$ compared to $mgl = 2$ which is counter intuitive.