End-2-End Self-Supervised SLAM

Mian Akbar Shah
shahak@ethz.ch

Selim Kaelin
kaelisel@ethz.ch

Ivan Alberico
ialberico@ethz.ch

Emilk Sempertegui
semilk@ethz.ch

Abstract

In this work we present End-2-End Self-Supervised SLAM, a pipeline combining gradSLAM, which enables SLAM systems to be posed as differentiable computational graphs, and online adaption to boost performance on indoor scenes. We train and test on two different datasets, implement an online adaptation module for refinement, explore the usage of uncertainty predictions and unsupervised scale learning, and supplement our work with experiments on weak supervision. Our framework generalizes well to previously unknown scenes, and through the online adaptation module we successfully address challenges related to indoor self-supervised depth estimation.

1. Introduction

Simultaneous localization and mapping (SLAM) is widely used in robotic perception and state estimation. It proposes frameworks for determining the current pose of the vehicle and reconstructing the surrounding scenes in 3D while navigating a trajectory. While real-time SLAM methods reconstructing depth maps from a moving depth sensors have increased in popularity recently [13, 17, 22], they are limited by the depth sensors' working range and performance under sunlight. Color cameras are ubiquitous and inexpensive, sparking an interest in working with monocular video for dense and semi-dense SLAM [4, 16], where consecutive image pairs are treated as coming from a stereo camera rig. Applicability of monocular SLAM is limited by the inherently ambiguous absolute scale of the reconstruction. To address this issue, the use of deep Convolutional Neural Networks (CNNs) for regressing depth at a relatively high resolution and absolute accuracy has recently been explored [20], where direct monocular depth estimation is combined with depth predictions from a supervised deep neural network. To further facilitate the integration of deep learning in SLAM, a framework called ***gradSLAM*** has recently been proposed. It allows SLAM pipelines to be posed as differentiable computational graphs, making it possible to employ gradient-based learning techniques, on which machine learning depends [12].

A drawback of supervised deep networks, is that they need large labeled datasets to be properly trained. Creating labeled datasets for each of the countless SLAM applications is a laborious and expensive effort, which is why research has recently focused on some form of self-supervision, e.g. synchronized stereo pairs [6, 7], or monocular video [23]. Self-supervision helps to mitigate the necessity for large labeled datasets, which is especially desirable for autonomous systems that are deployed in both out- and indoor environments, between which exists a considerable domain gap and large differences in scale. SLAM is employed both in outdoor and indoor environments, and could therefore greatly benefit from self-supervised depth predictions. Nevertheless, self-supervised depth prediction faces its own challenges. In the context of indoor scenes, with which we are mainly concerned, these include:

- Abundance of non-Lambertian surfaces and low-texture scenes that do not provide meaningful gradients.
- High degree of rotational movement in camera motion, which acts as noise.
- Scale inconsistency of pose estimates over different samples.

Ideally, we would want a system that incorporates SLAM from monocular video and fully self-supervised depth predictions, in order to navigate and map any environment in real time without prior knowledge. For this reason, we propose End-2-End Self-Supervised SLAM, a pipeline that estimates pose and depth from image pairs of a monocular video stream and implements self-supervision by cross-checking, updating and refining the reconstructed scene in a 3D point cloud. This is achieved through a combination of a depth network and ***gradSLAM***. As shown in Fig. 1, our model estimates and refines a depth map for each new key frame through an online adaptation module, which is then fed into the actual SLAM pipeline for reconstruction of a global point cloud and the estimation of the current pose. For the online adaption module, we use ground truth poses to address the issues of self-supervised learning for indoors mentioned before. We train and validate our model on two separate datasets, ICL-NUIM [9] and TUM [19]. Both qualitative as well as quantitative results are in-

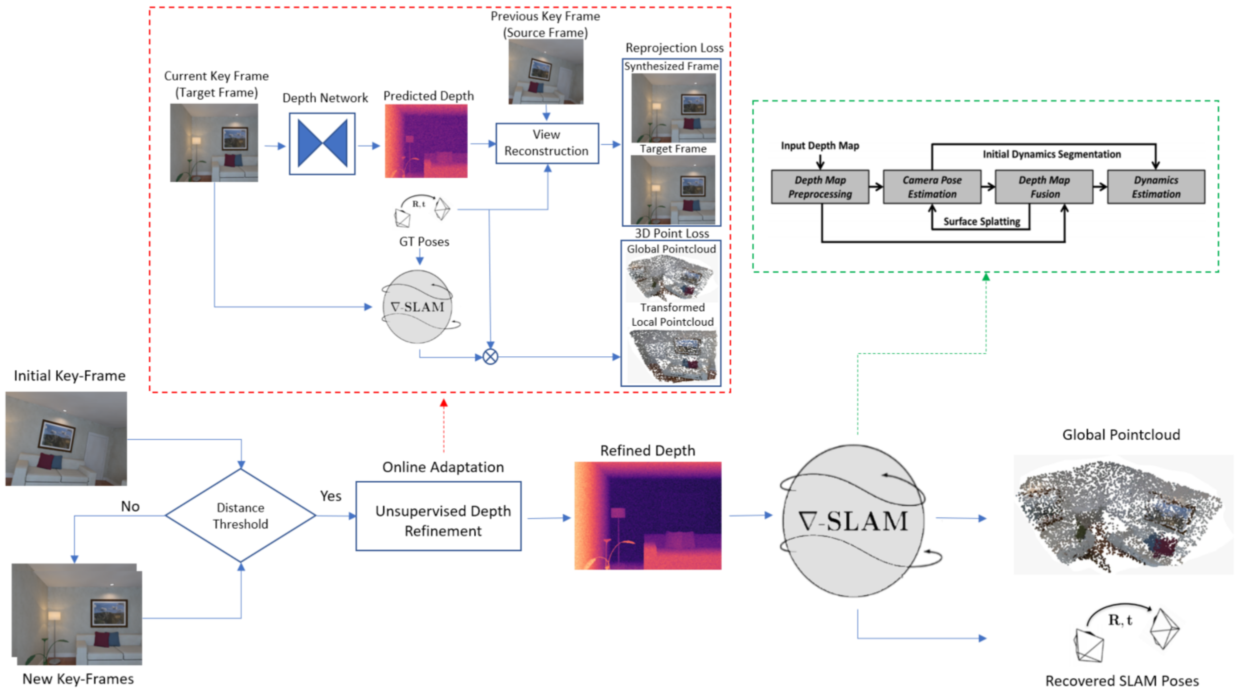

Figure 1. Our proposed End-2-End Self-Supervised SLAM pipeline with online adaptation module.

cluded for both datasets, suggesting that our model generalizes well and benefits from our online adaptation module. We outline the following contributions:

- A self-supervised SLAM system based on *PointFusion* with gradient based learning.
- Implementation of an online adaptation module to refine our self-supervised depth predictions.
- Investigation into uncertainty predictions, scale learning and output fine-tuning to improve depth estimation and reconstruction.

2. Related Work

In this section, we briefly discuss prior works in the fields of SLAM and depth estimation.

2.1. SLAM

SLAM approaches can be classified according to the input data type into depth-camera-based [13, 22, 17] or monocular-camera-based [4, 16], and according to the methodology applied into either feature-based [14, 16] or direct [4, 5].

ORB-SLAM [16] is a feature-based, monocular approach that relies on sparsely extracted ORB features from input images to reconstruction a sparse map of the scene, as well as estimating the camera pose. It employs local bundle adjustment and pose graph optimization. Large Scale Direct(LSD) SLAM [4], on the other hand, is a direct monocular SLAM framework. Instead of keypoints, it operates directly on image intensities for both tracking and mapping. Geometry is represented using semi-dense maps that only include depth values in gradient areas of the input images. This greatly improves efficiency, allowing deployment in real time on a CPU.

Within the context of our work, a recently proposed framework called ***gradSLAM*** [12] is of critical importance. It addresses the issue of differentiability in SLAM pipelines by introduction of a differentiable computational graph methodology. This allows gradient-based learning by providing explicit gradients with respect to input images and depth maps. By integrating ***gradSLAM*** into self-supervised learning models, they can be equipped with a sense of spatial understanding. Furthermore, we depend on the algorithm from the previously mentioned ***PointFusion*** [13] to balance scene reconstruction quality with real-time performance. It is based on a simple point representation, which works directly with input depth or range data. The memory and computational requirements are kept small by integrating new data points into a global model via depth map fusion, and the removal of points considered unstable.

2.2. Monocular Depth Estimation

Classic depth prediction methods employ hand-crafted features and graphical models, using strong assumptions on the scene geometry to yield regularized depth maps. Recently developed deep convolutional neural networks

(CNNs) outperform previous methods in terms of accuracy and robustness. These networks can be classified by their level of supervision into supervised [3] and unsupervised, also called self-supervised [8, 2, 7, 6, 1]. For this discussion we will focus on self-supervised approaches, which are the focus of our work.

GLNet [2] is a self-supervised framework for learning depth, optical flow, camera pose and intrinsics from monocular video. By design of new loss functions capturing multiple geometric constraints, an adaptive photometric loss, model extensions to predict camera intrinsics, and online refinement strategies, the network outperforms previous self-supervised implementations on multiple tasks. ***monodepth2*** [8] demonstrates the effectiveness of a surprisingly simple model including a minimum reprojection loss to handle occlusions, a full-resolution multi-scale sampling method to reduce visual artifacts, and an auto-masking loss to deal with violated camera motion assumptions, in self-supervised depth estimation.

The work of Godard *et al.* [7] proposed a stereo-based model to do single image depth estimation in the absence of ground-truth depth data. This is accomplished by exploiting epipolar geometry constraints, and through a novel training loss that enforces consistency between disparities relative to both left and right images. Especially relevant to our model is the work of Bian *et al.* [1], which establishes that degenerate camera motion in indoor handheld video, with a large amount of rotation that acts as noise, is a critical obstacle for unsupervised depth learning. They propose weak image rectification to bridge the domain gap between out- and indoor environments, allowing existing unsupervised models trained outdoors to be applied on indoor scenes. In our own framework we apply their pre-trained indoor model for depth estimation.

3. Methods

In this section we provide a detailed insight into our online adaption module which is basically self-supervised depth estimation through novel view synthesis. We first explain the basic building blocks for self-supervised depth estimation and then present self-supervision from the global point cloud created by the *PointFusion* method.

3.1. Novel View Synthesis

Novel view synthesis synthesizes a novel image of a scene from a new camera pose given another view of the same scene as input. Given a target image I_t we pass it through the depth prediction network to obtain the target's depth $\hat{D}_t$. Using the camera intrinsics matrix K, the points of the target image, p_t, can be projected into 3D. Homogeneous points are then transformed according to the camera motion profile $\hat{T}_{t\rightarrow s}$ from the target view to the source view $\hat{p}_s$. These operations can be summarized as:

$$\hat{p}_s \sim KT_{t\rightarrow s}\hat{D}_t\left(p_t\right)K^{-1}p_t \tag{1}$$

The projected coordinates $\hat{p}_s$ are continuous values. To obtain points from the source image $I_s(p_s)$ for populating the synthesized frame $\hat{I}_s(\hat{p}_t)$, we employ a differentiable bilinear sampling mechanism [11]. This mechanism linearly interpolates on the 4 neighbouring pixels (top-left, top-right, bottom-left, bottom-right) of p_s such that:

$$\hat{I}_s\left(p_t\right) = I_s\left(p_s\right) = \sum_{i\in\{t,b\},j\in\{l,r\}} w^{ij} I_s\left(p_s^{ij}\right) \tag{2}$$

where w^{ij} is linearly proportional to the spatial proximity between p_s and p_s^{ij}, and sums up to 1 for the four neighbouring pixels.

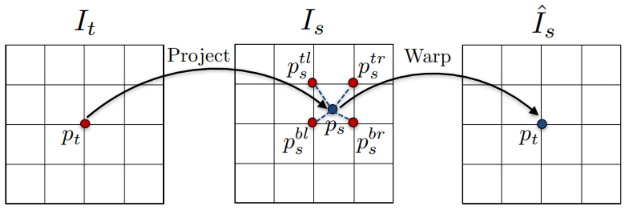

Figure 2. Differentiable image warping. Each point p_t in the target image is projected onto the source image using the predicted target depth and corresponding pose. Bilinear interpolation with four neighbouring pixels of p_s gives $\hat{p}_t$ Image Credits: [23]

This process relies on monocular temporal sequence of images where the target and source frames are temporal neighbours i.e. individual frames in a monocular video. The training procedure is driven by the minimization of the photometric error with the assumption of a static scene, no occlusion/disocclusion between the target and source views, and constant lighting conditions. The photometric error is a combination of SSIM and L_1 loss as:

$$L_{PE} = \frac{\alpha}{2}\left(1 - \mathrm{SSIM}\left(I_t, \hat{I}_s\right)\right) + (1-\alpha)\left\|I_t - \hat{I}_s\right\|_1 \tag{3}$$

where the constant α is denotes the weighting of each individual term. Even though the L_1 loss is robust in the presence of outliers, it is not invariant to changes in illumination of real-world scenes. The SSIM loss can handle complex illumination changes because it normalizes pixel illuminations. Invalid points are masked with photometric error masking technique i.e. pixels outside the projection flow. Using this entire process, we can train a model to produce depth values that agree with the synthesis process. Since we train for a number of refinement steps, our online adaption module is prone to overfit. In order to prevent overfitting to a single image pair, we enforce a regularization on the depth estimates:

$$R_{depth} = \gamma||D_0 - D_i|| \quad i = 0, 1, ...N \tag{4}$$

where γ is a weighting function to control the amount of regularization and N represents the total refinement steps.

3.2. End-2-End Point Self-Supervision

Due to the differentiable nature of gradSLAM[12], we can leverage the global reconstructed pointcloud to refine our depth maps in an online fashion. This task can be easily coupled with the novel view synthesis framework from the previous section. However, since we will compare pointclouds, this supervision incurs additional cost at each refinement step. The global pointcloud G contains points that are updated after online adaption has been applied to the previous key-frames. Subsequently, these points are also refined through depth map fusion within the PointFusion based SLAM. Therefore, these points can be used to supervise future key-frames. At each refinement step, along with novel view synthesis, we pass the I_t, D_t, K, P_t through PointFusion to obtain a local pointcloud of the target frame L_t. We then transform L_t to the source frame's view using the transform $T_{t\to s}$ to obtain $L_{t\to s}$. Finally, the distance between L_t and G is computed such that:

$$L_{CD}\left(G, L_t\right) = \sum_{x\in G} \min_{y\in L_t} \|x-y\|_2^2 + \sum_{y\in L_t} \min_{x\in G} \|x-y\|_2^2 \tag{5}$$

where x corresponds to the points in the global pointcloud and y corresponds to the points in the local pointcloud. Points are padded where one pointcloud has more points than the other and difference is taken with the nearest neighbour. Our final online adaption module minimizes the loss function given as:

$$L_{online} = L_{PE} + L_{CD} + R_{depth} \tag{6}$$

3.3. Uncertainty Prediction

Inspired by the idea of weighting pixels in either the photometric error [23, 8] or depth maps [20], we aim to leverage on predicted uncertainty maps that serve as a weighting mask for the self-supervised loss. As discussed by Poggi et al. [18], the uncertainty maps can be considered to represent the variance $\sigma(d)$ of the target distribution to be learned, where the predicted depth values correspond to the mean of the distribution $\mu(d)$. In this sense, depth maps are inferred together with uncertainty maps via negative log-likelihood minimization, which under the assumption of a Laplacian distribution leads to minimizing a loss function given by

$$L_{log} = \frac{|\mu(d) - d^*|}{\sigma(d)} + \log\sigma(d) \tag{7}$$

where d^* stands for ground truth depth values. However, it has been shown [15] that for the self-supervised case Eq. 7 becomes

$$L_{log} = \frac{\min_{i\in[0..K]} F(\tilde{I}_i(q), I(q))}{u_{Log}} + \log u_{Log} \tag{8}$$

where $\min_{i\in[0..K]} F(\tilde{I}_i(q), I(q))$ stands for the minimum reprojection loss between target $I(q)$ and synthesized frame $\tilde{I}_i(q)$. Note that the $\log u_{Log}$ term in Eq. 8 acts as a regularizer to avoid the trivial solution of setting u_{Log} to infinity. Moreover, the network is trained to predict the log-uncertainty ($\log u_{Log}$) as to prevent instability during training due to divisions by 0. It should be noted that this training procedure is independent from the online adaptation module since the uncertainty head is pre-trained and not modified during the online refinements.

4. Results

In this section we provide experimental results to validate the effectiveness of online refinement module along with the corresponding 3D reconstruction from *PointFusion*. Note that all experiments were carried out with median scaling unless otherwise specified.

4.1. Implementation details

We use sequences from the TUM RGB-D SLAM dataset [19] and the ICL-NUIM dataset [9], with an image size of 320×256. In our experiments, we used the weights of the *SC-SfMLearner* model [1] pretrained on the indoor sequences of the rectified NYU Depth v2 dataset, in order to have a reasonable starting point for our depth predictions. The SSIM to L_1 loss weighting parameter, α, in Eq. 3 is set to 0.15 and the depth regularization parameter, γ, in Eq. 4 is set to $1.e^{-2}$. In order to quantify our results, we report the *Abs-Rel*, *RMSE*, and δ_1 metrics for the predicted and refined depth predictions.

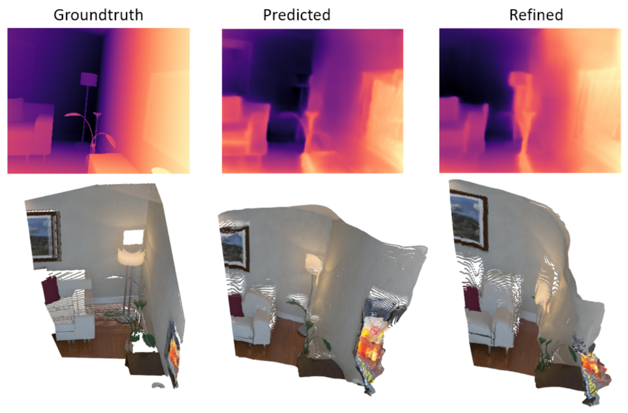

Figure 3. Depth predictions and reconstructed point clouds of a pair of key-frames taken from the ICL dataset. **Left:** Ground truth **Center:** No online-refinement **Right:** Online-refinement

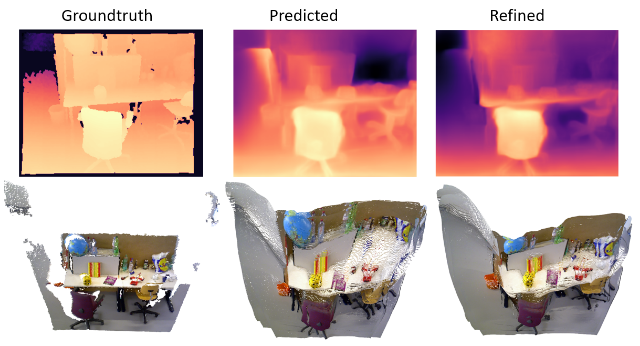

Figure 4. Depth predictions and reconstructed point clouds of a pair of key-frames taken from the TUM dataset. **Left:** Ground truth **Center:** No online-refinement **Right:** Online-refinement

4.2. Online Adaptation Module

We present experimental evaluation to validate the contributions of our method. As discussed before, self-supervised indoor depth estimation is a challenging task. We address these challenges by using the ground truth poses rather than relying on PointFusion's camera pose estimation. Our main assumption behind this is the ubiquitous availability of pose information on modern devices. Additionally, our self-supervised depth model is pre-trained on weakly rectified NYU dataset which addressed the issue of rotation noise [1]. The weak rectification acts as a pre-processing step on the training data, which finds pair of images with moderate translation and removes their relative rotation for effective training. Although prior works have shown to use geometric losses [1] to ensure scale consistency, we find that the scale is still not consistent enough for our application. Another issue is that using the geometric loss [1] in our online adaption module often leads to depth converging to a single value. Using ground truth poses, our model slowly adapts towards produce a consistent scale factor after a few refinement steps since the given pose information is scale consistent. Since we use median scaling, our depth estimates and the ground truth pose are on the same order of scale.

In order to show the effectiveness of our online adaption module, we refine over a pair of key-frames for 25 steps. In Fig. 3 and Fig. 4 it can be noticed that the initial depth predictions are plausible due to pre-training, however, our online adaption module is able to improve these predictions further (see Tab. 1). Refining for numerous steps on a single pair of key-frames can easily lead to overfitting. But we emphasize that these particular results are meant to show the effectiveness of our online adaption module. The depth regularization term, Eq. 4, plays a key role to handle the aforementioned problem of overfitting.

Dataset	Steps	Abs-Rel	RMSE	δ_1
ICL	0	0.234	0.555	0.567
ICL	25	**0.128**	**0.504**	**0.804**
TUM	0	0.168	0.899	0.708
TUM	25	**0.0893**	**0.810**	**0.889**

Table 1. Quantitative evaluation of the online adaptation after applying 25 refinement steps.

4.3. Weak supervision

An addition to our method consists of integrating weak-supervision into the pipeline through ground truth depth information. We wanted to investigate the effectiveness of having pseudo-labels from old classic techniques, that could act as a supervisory signal for the depth predictions. However, due to time constraints we relied on sparse ground truth depth labels. In our case we used only 1% of the total amount of pixels. The weak supervision is integrated in our model through an L_1 loss computed between the sparse ground truth values and the depth predictions from our network. As it can be seen in Fig. 5 and Fig. 6, introducing weak supervision inside our model results in more accurate depth predictions and reconstructed pointclouds. The disadvantage of this method is that it requires some ground-truth information. A remedy to this, which could be investigated in some future works, would be relying on noisy labels instead, possibly calculated via semi-global matching technique.

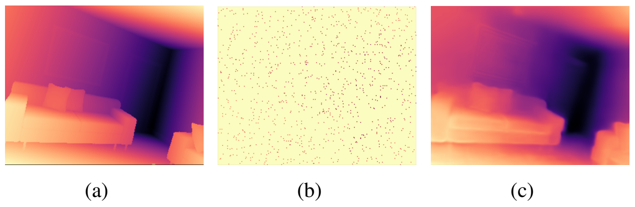

Figure 5. The sequence of images shows a comparison between the ground truth depth map and the depth prediction after including weak-supervision into the pipeline (1% of the pixels). The selected keyframe is taken from the ICL-NUIM dataset. (a) Ground truth depth map (b) Sparse ground truth values (c) Predicted depth map.

4.4. Uncertainty Prediction

We leveraged on the open-source implementation of our rectified-NYU pre-trained model [1] and extended it as to include the uncertainty prediction head, which corresponds to a 3×3 conv layer that is applied to the decoder feature maps with the largest scale. Following the discussion in [18] the extended network was trained from scratch applying an ImageNet initialization to the depth encoder. This model was trained for 17 epochs, batch size 16, and learning rate 0.0001. The original base model implementation con-

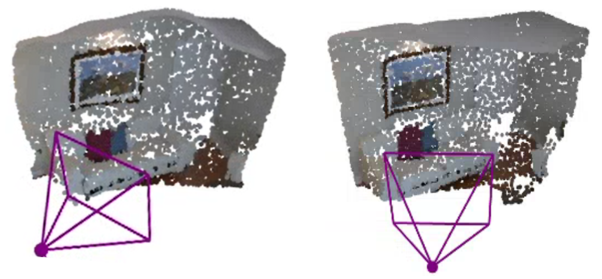

Figure 6. Comparison between global pointcloud reconstructions without (left) and with (right) weak supervision of a sequence of frames taken from the ICL-NUIM dataset.

sidered masking techniques for handling moving objects in the scenes, as well as the addition of the smoothness and geometry consistency terms to the training loss. However, we found that when combining these with the uncertainty related terms the training becomes unstable due to exploding gradients in the early epochs. Thus, the model was trained based on the minimization of the loss given by Eq. 8.

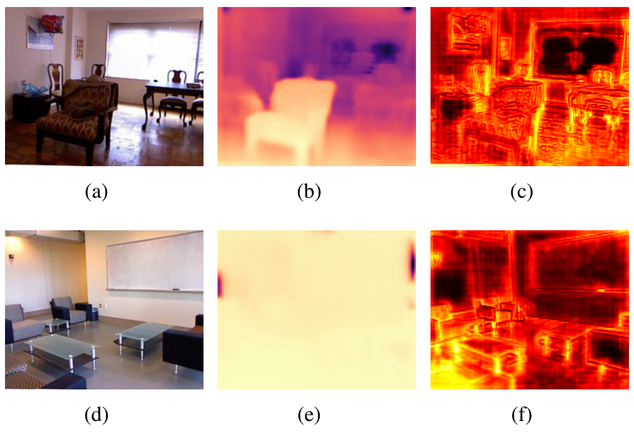

Figure 7. Predictions from the augmented model with uncertainty head. Each row contains a RGB frame (left) and its corresponding depth (center) and uncertainty (right) predictions. Uncertainty color band maps low-medium-high uncertainty values to black-red-yellow colors, respectively.

The resulting predictions are shown in Figure 7. Considering 7(a) consistent depth 7(b) and uncertainty 7(c) predictions are obtained, where the highest uncertainty is assigned typically to edges in the scene, which usually correspond to regions with the largest photometric error. On the other hand, by looking into 7(d) it can be observed that erroneous depth predictions 7(e) are obtained with overconfident estimates 7(f) in most parts of the scene. We believe this behavior is related with the dominance of low texture regions affecting the self-supervised loss; notice that in 7(f) the high confidence (low uncertainty) regions correspond mainly to the white walls, homogeneous floor and white board in 7(d). We found that the latter set of predictions was far more common during the training procedure, hence we did not combined the learned uncertainty head with the online adaptation module.

5. Discussion

In our framework, the online adaption module faces two issues:

- Since we train at test-time, we are prone to overfitting to the given scene.
- GPU intensive operations and large model size

An important hyperparameter is the number of refinement steps per key frame. Typically, in the same scene numerous training samples look similar to each other, therefore we are prone to overfitting to the training objective. Large learning rates ($1e^{-4}$) are desirable to have less number of refinement steps, however, this can also lead to overfitting. While refining a single image pair, we noticed that there was trade off between accuracy, number of refinement steps, and the learning rate. The depth regularization term, Eq. 4, plays a key role to handle both aforementioned problems that can lead to overfitting.

Since we are refining at test-time, we need a GPU that is sufficiently large enough to handle our ResNet-18 based model and the corresponding optimization process. Another bottleneck is storing the global pointcloud which increases as we move through the scene. A smaller model size that gives similar depth accuracy would therefore be a promising future work, since we can also reconstruct larger scenes with the same amount of memory.

The aforementioned problems with indoor self-supervised depth estimation were handled by leveraging ground truth poses, however, in the future we would like to be able to use SLAM poses in order to leverage the full potential of SLAM. In addition, we noticed that the scale inconsistency even after median scaling leads to bad overlap between key-frames in the global pointcloud. We hope to tackle this in upcoming future.

6. Conclusion

In this work we presented End-2-End Self-Supervised SLAM to perform 3D reconstruction. We generalize to new scenes using the online adaptation module that continuously refines depth maps in a self-supervised way. Improvement in this particular topic is heavily dependent on the challenges of indoor self-supervised depth estimation. Nonetheless, we improved the capabilities of online adaption module through removing border artifacts, prevented overfitting via depth regularization and most importantly supervision from the global pointcloud. Poses recovered from SLAM proved to be a challenge to use for online adaption, and still remains as open work.

References

[1] J.-W. Bian, H. Zhan, N. Wang, T.-J. Chin, C. Shen, and I. Reid. Unsupervised depth learning in challenging indoor video: Weak rectification to rescue, 2020.

[2] Y. Chen, C. Schmid, and C. Sminchisescu. Self-supervised learning with geometric constraints in monocular video: Connecting flow, depth, and camera. *CoRR*, abs/1907.05820, 2019.

[3] D. Eigen, C. Puhrsch, and R. Fergus. Depth map prediction from a single image using a multi-scale deep network. *CoRR*, abs/1406.2283, 2014.

[4] J. Engel, T. Schöps, and D. Cremers. Lsd-slam: Large-scale direct monocular slam. In *European Conference on Computer Vision (ECCV)*, September 2014.

[5] J. Engel, J. Sturm, and D. Cremers. Semi-dense visual odometry for a monocular camera. In *IEEE International Conference on Computer Vision (ICCV)*, December 2013.

[6] R. Garg, V. K. BG, G. Carneiro, and I. Reid. Unsupervised cnn for single view depth estimation: Geometry to the rescue, 2016.

[7] C. Godard, O. M. Aodha, and G. J. Brostow. Unsupervised monocular depth estimation with left-right consistency, 2017.

[8] C. Godard, O. M. Aodha, M. Firman, and G. Brostow. Digging into self-supervised monocular depth estimation, 2019.

[9] A. Handa, T. Whelan, J. McDonald, and A. Davison. A benchmark for RGB-D visual odometry, 3D reconstruction and SLAM. In *IEEE Intl. Conf. on Robotics and Automation, ICRA*, Hong Kong, China, May 2014.

[10] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *CoRR*, abs/1502.01852, 2015.

[11] M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu. Spatial transformer networks, 2016.

[12] K. M. Jatavallabhula, S. Saryazdi, G. Iyer, and L. Paull. gradslam: Automagically differentiable slam, 2020.

[13] 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.

[14] G. Klein and D. Murray. Improving the agility of keyframe-based slam. volume 5303, pages 802–815, 10 2008.

[15] M. Klodt and A. Vedaldi. *Supervising the New with the Old: Learning SFM from SFM: 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part X*, pages 713–728. 09 2018.

[16] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós. Orb-slam: A versatile and accurate monocular slam system. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015.

[17] 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, 2011.

[18] M. Poggi, F. Aleotti, F. Tosi, and S. Mattoccia. On the uncertainty of self-supervised monocular depth estimation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[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] K. Tateno, F. Tombari, I. Laina, and N. Navab. Cnn-slam: Real-time dense monocular slam with learned depth prediction. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6565–6574, 2017.

[21] J. Watson, M. Firman, G. J. Brostow, and D. Turmukhambetov. Self-supervised monocular depth hints. In *The International Conference on Computer Vision (ICCV)*, October 2019.

[22] T. Whelan, M. Kaess, H. Johannsson, M. Fallon, J. Leonard, and J. McDonald. Real-time large scale dense RGB-D SLAM with volumetric fusion. *Intl. J. of Robotics Research, IJRR*, 34(4-5):598–626, Apr. 2015.

[23] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe. Unsupervised learning of depth and ego-motion from video. In *CVPR*, 2017.

A. Contributions

Our source code was written with the help of the *gradSLAM* library, *monodepth2* [8] and Bian et al. [1]. *PointFusion* and visualization functions were also taken from *gradSLAM* [12]. The novel view synthesis, SSIM and smoothness loss functions were taken from *monodepth2* [8]. The depth estimation network and geometric loss function implementations were taken from Bian et al. [1]. The rest of the source code was implemented by the team members as mentioned in the contributions section below.

B. Additional Works

B.1. Gradient Experiments

gradSLAM [12] is a fully differentiable dense SLAM framework that allows gradient flow from the 3D reconstruction of the environment all the way through to inputs i.e. color and depth images. In order to investigate how effective these gradients are, we carried out an experiment to recover the RGB-D input from the 3D reconstruction. The problem setup was designed as follows: given a sequence of four RGB-D pairs along with corresponding camera intrinsics and poses, we corrupted the fourth pair. The task was to reconstruct the fourth pair by leveraging gradients from comparison with the ground truth reconstruction (see Fig. 8). The fourth pair was corrupted in different ways i.e. adding Gaussian noise, removing a patch from the centre of the image, and replacing the entire image by a constant value.

We used the Chamfer distance loss from Eq. 5 to minimize the distance between the corrupted and ground truth reconstruction. For color loss, we simply took the L_1 distance between the point's color values in the corrupted and ground truth reconstruction. The final loss function was defined as:

$$L_{total} = L_{color} + L_{CD} \quad (9)$$

The fourth RGB-D pair itself is used as the parameters that we optimize through gradient descent. We use the Adam optimizer with a learning rate of 0.0001 and train for 80 steps. The experimental results verified the gradient propagation capabilities of *gradSLAM*. Gradients from comparison of the pointclouds were able to reconstruct a corrupted image through backpropagation. This analysis helped us to formulate the End-2-End Point Self-Supervision (see Sec. 3.2), such that we can leverage the gradients from comparison of the transformed local pointcloud with the global pointcloud to enhance our online refinement module.

B.2. Unsupervised Scale Learning

To leverage on ground truth poses our method needs to refine the global scale of the predicted maps as to bring the predicted depth values closer to the scale range of the motion profile. One technique to change the scale of depth predictions, commonly used in the literature for the purpose of evaluating results against ground truth depths, is median scaling [23], where the scale factor is computed as $\hat{s} = median(D_{gt})/median(D_{pred})$ thus relying in ground truth depth maps. Given that our proposed approach aims to avoid the use of ground truth depth we investigated the feasibility of predicting an approximate scale through minimization of the unsupervised loss in Eq. 3.
For this purpose an additional head was introduced in the unsupervised depth refinement model, which takes as input the predicted depth maps and transforms them by applying an affine transformation, i.e. each pixel's depth value is multiplied by a constant scale factor and added an offset. The transformed depth maps are utilized for the view synthesis process. In terms of implementation, the scaling head consists of a 1×1 convolutional layer with a bias term, where the weight and bias represent the scale multiplier and offset, respectively. Nevertheless, the implementation of the transformation through a convolutional layer allows to recover the traditional linear scaling technique, such as median scaling, by simply disabling the bias term.

The scale prediction head was trained following the online refinement workflow on a defined sequence of three frames from the ICL dataset. However, in order to mainly assess the effect of the parameters in the scaling conv layer the depth prediction network was frozen after initializing it with the pretrained model. Four different initial configurations of the head were considered, each with a predefined initialization value for the weight w whereas the bias b was initialized according to the He initialization scheme [10]. The model was trained to optimize the defined sequence of frames until a plateau in the loss was reached, with a learning rate of 0.04. Table 2 summarizes the resulting behaviour with the different initial configurations. The performance of the learned parameters with the smallest *Abs-Rel* error was evaluated through 30 steps of the online refinement task in the same sequence of frames considered during training and the corresponding results are shown in Table 3. The model using the learned scale parameters performs poorly in terms of *Abs-Rel* and *RMSE* when compared to using the median scaling factor. This suggests that the scale head is not capable of fully capturing the depth estimation task in hand and instead its training dynamics are misguided by the aforementioned difficulties found during self-supervised depth estimation, e.g. low-texture regions of indoor scenes. Note, however, that this behaviour is consistent with the local minima problem commonly found when using photometric reprojection as the training loss [21].

Init Params (w, b)	Photo	Abs-Rel	RMSE	Final Params (w, b)
3, -0.392	0.1127	0.6962	1.8643	1.06, 0.294
6, 0.831	0.1120	0.6507	1.6271	5.38, -0.949
7, -0.804	0.1119	0.6478	1.6194	5.45, -0.962
8, 0.322	**0.1119**	**0.6383**	**1.5940**	**5.68, -1.014**

Table 2. Training loss (Photo), metrics and learned parameters for scale learning with different initial parameters configurations.

Scale	Abs-Rel	RMSE
Median	**0.2908**	**0.8788**
Learned	0.6730	1.8993

Table 3. Resulting metrics after 30 steps of online refinement using median scale factor = 7.3586 and learned scale parameters.

B.3. Output Fine-tuning

Output fine-tuning[2] is the idea of refining the predicted depth maps rather than the depth prediction model through the proposed loss function (Eq. 6). During training, the depth prediction model's weights will be frozen since the initial depth network is pre-trained and produces plausible results. The benifits of this method are that it updates the depth predictions directly which contains much less parameters ($< 10k$) compared to the depth prediction model (ResNet18 with $11M$). In terms of refinement speed, this method is one order ($10\times$) faster. Our investigation showed this speed up claim to hold true, however, the depth predictions were not improved significantly compared to our online adaption module. Nevertheless, it is promising research direction that can be beneficial to investigate in the future.

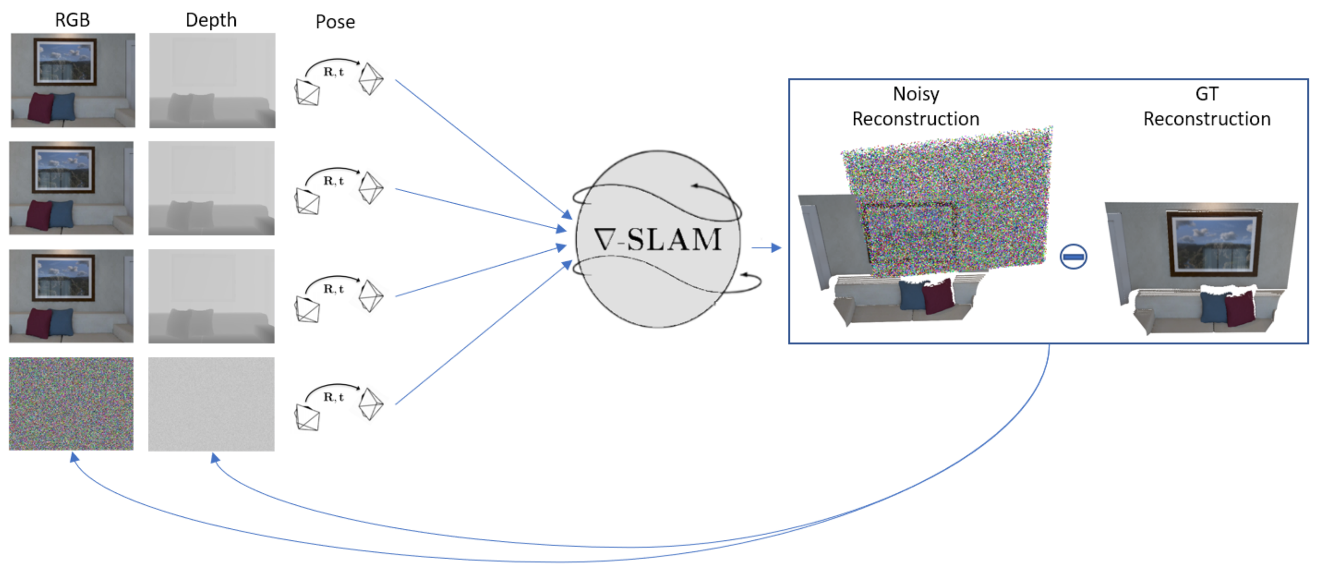

Figure 8. *gradSLAM* image reconstruction gradients experiment setup.

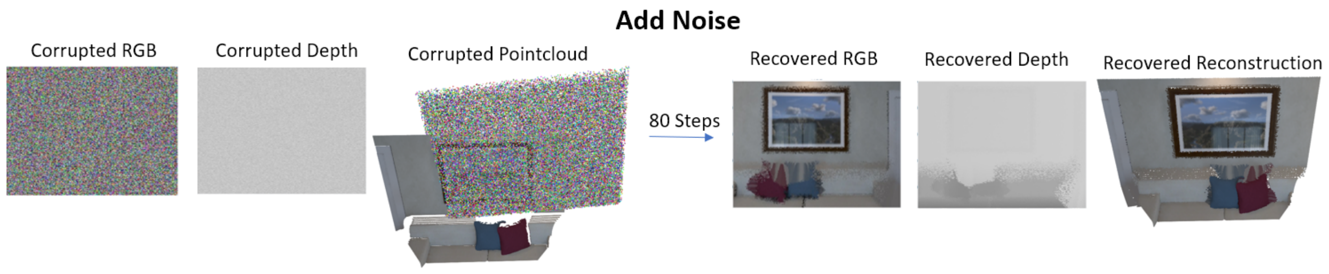

Figure 9. Qualitative results for gradient experiment when the fourth input is corrupted by adding noise.

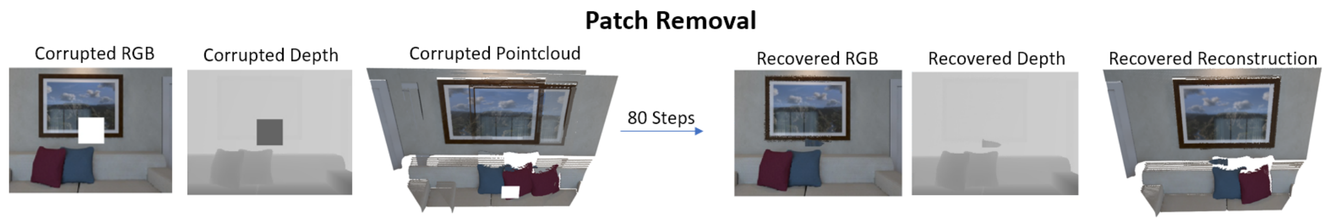

Figure 10. Qualitative results for gradient experiment when the fourth input is corrupted by removing a patch from the middle of the RGB-D frames.

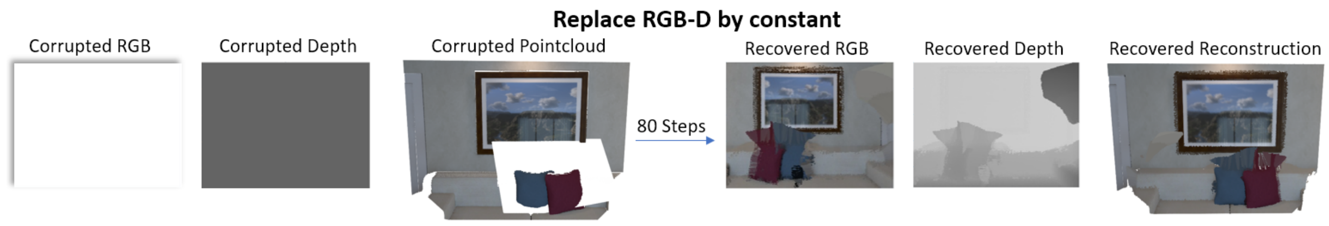

Figure 11. Qualitative results for gradient experiment when the fourth input is replaced by a constant value.