IEEE Robotics and Automation Letters (RAL) paper presented at the
2019 International Conference on Robotics and Automation (ICRA)
Palais des congres de Montreal, Montreal, Canada, May 20-24, 2019

Sparse2Dense: From direct sparse odometry to dense 3D reconstruction

Jiexiong Tang[1], John Folkesson[1] and Patric Jensfelt[1]

***Abstract*— In this paper, we proposed a new deep learning based dense monocular SLAM method. Compared to existing methods, the proposed framework constructs a dense 3D model via a sparse to dense mapping using learned surface normals. With single view learned depth estimation as prior for monocular visual odometry, we obtain both accurate positioning and high quality depth reconstruction. The depth and normal are predicted by a single network trained in a tightly coupled manner. Experimental results show that our method significantly improves the performance of visual tracking and depth prediction in comparison to the state-of-the-art in deep monocular dense SLAM.**

I. Introduction

SLAM is a key building block in most mobile autonomous systems. Much of the recent research addresses the SLAM problem with a single camera. A solution with a single camera would be very competitive in many applications as a camera is relatively inexpensive and already present in most mobile devices. In this paper we investigate direct methods for SLAM. Impressive semi-dense/sparse tracking and mapping results have been achieved. LSD-SLAM [1] and the more recent DSO [2] define the state of the art in these domains. However, they are not able to overcome the intrinsic problem of monocular visual positioning, that scale is not observable. With the recent advances in deep learning, this issue is now being tackled by using learning based depth estimators. The idea is to use a network to predict the depth from a monocular image and use this as a prior in a SLAM or visual odometry (VO) system. Recent works [3], [4] show that the absolute position error can be greatly reduced in this way. This is the approach we take in this paper as well.

Our long term goal is accurate and dense 3D reconstruction of scenes. Such models could, for example, support advanced predictions of the effect of certain physical interactions. We make several important contributions in this paper. At a high level, we propose a deep learning based dense monocular SLAM method capable of real-time performance. The most related work to ours is CNN-SLAM [3]. A key insight in our work is that we should combine the ability of the CNN to generate dense depth predictions with the ability of a visual tracking system to generate highly accurate but sparse points through optimization. We use these sparse but accurate points to correct the dense depth predictions from the CNN. In particular, we leverage normals and an assumption about local planar structures. Depth and normals are predicted by a single CNN for efficiency. The network has been trained in a novel coupled manner, optimized for the sparse-to-dense reconstruction task. The CNN thus contributes to the tracking system by providing the true scale and the tracking system helps improve the accuracy of the dense depths. In our work, the sparse point clouds are generated from the active window keyframes of DSO [2], initialised by depth priors from the CNN. After the sparse to dense reprojection, the keyframes are sent to the backend which includes keyframe wise refinement and a fusion based mapper. We choose a fusion based mapper for the further enhancement of the global 3D model consistency. The backend also provides our system with loop closure abilities. An example is shown in Fig. 1.

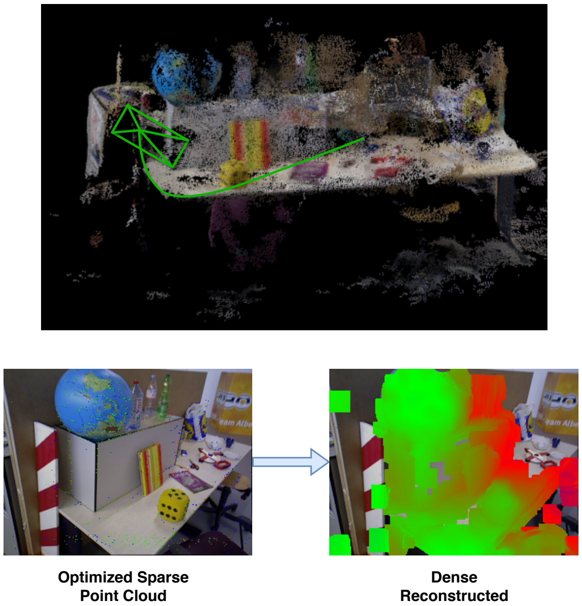


Fig. 1: An example of our proposed method, S2D, for dense reconstruction. Top: The reconstructed 3D scene. Left: The optimized sparse point cloud overlaid on the corresponding image frame. Right: The densely reconstructed point cloud. Note how the depth of the points in the local region around the sparse points are refined.

In summary, the key feature of our system is learning based sparse to dense mapping for 3D reconstruction. We denote our method S2D (Sparse2Dense). In the remainder of the paper, we firstly review the related work. Secondly, we provide an overall system overview to better explain the details. Then, in the following two sections, we present our approach to coupled training and reconstruction. Finally, we show experimental results, discuss these and outline

This work was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) and the SSF project FACT.

[1]The authors are all with the Centre for Autonomous Systems at KTH Royal Institute of Technology, Stockholm, SE-10044, Sweden `jiexiong@kth.se`

directions for future work.

II. Related Works

Single View Depth and Normal Estimation. Deep learning methods have achieved great advances in the area of single view depth and/or normal estimation and have largely replaced classical methods such as [5] and [6].

Eigen *et al.* [7] train a two scale CNN to predict depth from single images. Liu *et al.* [8] use a CNN to learn unary and pairwise potentials for a continuous CRF for depth estimation. Laina *et al.* [9] propose a fully convolutional [10] residual network [11] (FCRN) with up-projection based upsampling using interleaved convolution. In addition, there are many supervised deep learning methods for monocular depth estimation [12], [13], [14], [15] showing good performances.

Another recent trend, are approaches which train the CNN to predict the depth in a self-supervised way [16], [17], [18], [19] or in an unsupervised way [20]. They use an image reconstruction loss without the supervision of ground truth depth. This is well suited for scenarios where the depth ground truth is hard or expensive to be collected, e.g., recorded video and outdoor. Recent methods in [21], [4] show that combining supervised learning using depth ground truth and self-supervised learning achieves better performance.

For single view normal prediction, Wang *et al.* [22] developed CNNs that operate both locally and globally on the image. The resulting predictions are combined with evidence from vanishing points to produce the final prediction. In [23] both depth and normal predictions are performed using a multi-scale deep network. Li *et al.* [24] use hierarchical CRFs to estimate depth and normals from monocular images. Bansal *et al.*introduce a skip-network model in [25] and in [26] a model for stratified sampling of pixels that can be used for normal prediction. In work [27], a CRF with a 4-stream CNN is designed to improve the consistency of predicted depth and surface normals in planar regions. GeoNet, proposed in recent work [28], consists of two streams of CNNs that have been jointly optimized to predict depth and normal through depth-to-normal and normal-to-depth mappings.

In our setting, an indoor scenario, relative large amounts of labeled samples is available, such as NYUv2 [29], SUN-3D [30], etc. Thus, we trained the network in a supervised manner. Furthermore, we trained the CNN to predict both depth and normal in a coupled way similar to [28]. The reconstructed depth and normals are used as strong regularization and can be seen as a pre-optimization for the sparse to dense reconstruction. Further details can be found in Sec. IV.

Monocular VO and SLAM Impressive progress has been made in visual odometry and SLAM methods. A common way to categorize different approaches is to use direct / indirect and dense / sparse. In direct methods, image frames are aligning directly based on pixel intensities and in indirect methods by first extracting features. Sparse and dense methods differ by how much of the image information is used. ORB-SLAM [31] defines the state-of-the-art in indirect sparse methods. When speed is of the essence SVO2 [32], using a semi-direct approach, offers frame rates of hundreds of Hz. LSD-SLAM [1] was one of the first direct semi-dense methods. The more recent DSO [2] is a direct and sparse method that adds joint optimization of all model parameters.

Scale drift [33] is an error which cannot be removed easily in a principled way with traditional methods when using a single camera[1]. Traditional, non-deep, methods are therefore gradually being challenged by learning based methods. Recent deep learning based mapping systems [3], [4] reduce the scale drift error by incorporating deep learning based single view depth estimation. In CNN-SLAM [3], a CNN is used to predict single view depth, which is fed into LSD-SLAM to achieve dense reconstruction. The depth is refined by using Bayesian filtering from [1], [34]. In DVSO [35], a virtual stereo view similar to [18] is predicted for the depth. This is jointly optimized for high accuracy tracking using DSO. Yin *et al.* [36] improve the performance of the depth estimation by using two consecutive frames and estimate ego-motion with refined depth. In CodeSLAM [37], a compact learned representation from conditioned auto-encoding is optimized to obtain a dense reconstruction with camera pose.

End-to-end training is a general trend. Here ego-motion estimation is performed directly, either supervised with ground truth or unsupervised [20], [38] using image reconstruction loss. However, as shown in [35], the performance of the end-to-end ego-motion is not on par with geometrical optimization based methods yet.

Our work is tightly related to deep learning based VO/SLAM and single view depth/normal estimation. Our method, S2D, is built on top of the direct monocular VO method DSO[2]. Depth and normals are predicted by a jointly optimized CNN. The learning based depth prior is used in the geometric optimization to reduce scale drift and achieve accurate monocular camera pose estimation. This results in sparse but optimized depth estimates. Finally, surface normal based geometrical reconstruction is conducted to rebuild a dense point cloud from the optimized sparse depth estimates.

III. System Overview

The overview of proposed S2D system is shown in Fig. 2. The overall framework can be divided into four major stages: learning based prior generation for depth/normal, visual tracking using direct alignment, geometrical sparse to dense reconstruction and lastly fusion based mapping. The main contributions in this paper are made in stage one and three. Examples of intermediate results in our pipeline are illustrated in Fig. 3.

Before we dive into the fine technical details, we provide a brief overview of these four stages and how they are connected. We use DSO for the visual tracking. Whenever a new keyframe is created by DSO, we use a single network

[1] Observing objects with known sizes has been one way to overcome scale-drift.

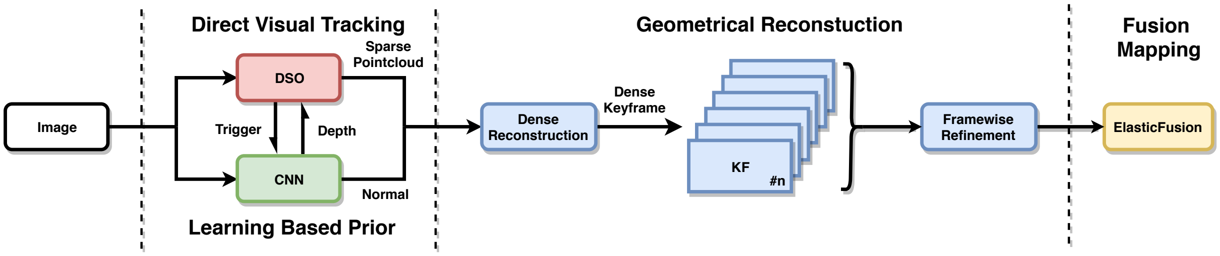


Fig. 2: An overview of our sparse to dense mapping framework, S2D.

to infer the depth/normals from the image data. When DSO is in the initialization stage, we directly assign depth priors to the new immature points to be optimized. If, on the other hand, DSO has been initialized, we project all mature points from active keyframes in the optimization window to the new keyframe. This sparse optimized depth cloud is used for: (1) a global scale correction for the depth prior of the new immature points and (2) a sparse to dense recasting, where the optimized depth is propagated to co-planar neighbouring pixels using the predicted normal. The reconstructed depth images of theses keyframes are refined using Bayesian estimation. Finally, the dense refined depth images are fed into a fusion based mapper, built on ElasticFusion, which generates a consistent global 3D model and handles loop closures.

IV. Coupled Depth and Normal Prediction

In this section, we first introduce the network structure deployed in S2D. Then, we illustrate the training scheme for the tightly coupled depth and normal prediction.

A. Network Structure

In CNN-SLAM, the network FCRN with an encoder-decoder structure is used for monocular depth estimation. The encoder part is built on ResNet50. For the decoder, a sequence of residual upsampling blocks, composed of interleaved convolution and up-projection, are used for recovering the feature maps at the targeted resolution. In S2D, various modifications have been made for more stable training and better performance. A source of inspiration is [18], which present a structure that obtain better performance than related methods (including FCRN) in outdoor scenarios. With respect to FCRN, we replace ResNet50 by the Dialated Residual Network (DRN) [39] in the encoder part. The feature maps of the DRN have higher resolution, which is better suited for generating more detailed depth/normal. For the decoder, besides residual upsampling, we train the network to predict depth/normal at three different scales. The depth prediction at lower resolution are upsampled and aggregated with the higher resolution one during the decoding. The overall structure is shown in Fig. 4. We denote the modified network structure FCDRN, to highlight the interleaved upsampling block from FCRN and the use of DRN.

B. Training Scheme

Inspired by GeoNet [28], we use a tightly coupled two-way reconstruction training scheme: depth-to-normal and normal-to-depth. [28] showed impressive depth and normal estimation quality compared with other existing methods. GeoNet uses two CNNs to predict the depth/normal. It runs at around 1Hz on a desktop with modern CPU and GPU, which is much more efficient than other related deep learning based methods (more than 10 times faster). However, an online SLAM system can still not afford this computational cost.

In S2D, the depth/normal are predicted in one single network (FCDRN introduced in Sec. IV-A) using input images with resolution of 320×240. This is similar to the resolution of 304×228 used in FCRN. The output, the depth/normal predictions, is 160×120. We do not use depth-to-normal mapping as post-processing, as we found its main effect to be to regularize the depth with geometrical structure during training, but not to improve the quality of the predicted depth.

We implement the normal-to-depth conversion in C++ with CUDA to allow it to run in real time together with the whole system. Note that the overhead of computing depth/normal only appears when making a new keyframe.

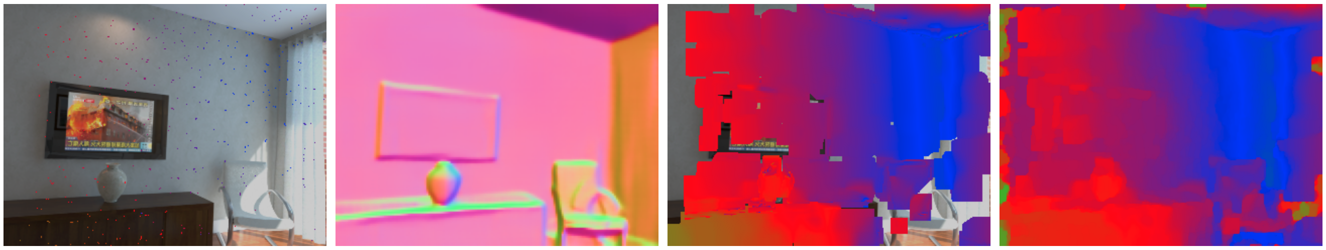
Focal Length Adaption. The main challenge when training a network to predict depth from RGB-D images captured by a single RGB-D camera is: if the testing is conducted using another sensor, the change in focal length brings in an error in the scale of the estimated depth. To reduce this effect and make the trained network generalize better, in CNN-SLAM [3], the depth $\hat{\mathbf{Z}}$ generated by the CNN is adjusted as follows:

$$\mathbf{Z}(\mathbf{u}_i) = \frac{f_{test}}{f_{train}} \hat{\mathbf{Z}}(\mathbf{u}_i) \tag{1}$$

where f_{train} and f_{test} are the focal lengths of the cameras used for training and testing respectively. This rescaling is performed as a separate post-processing step, which is not related to the training of the network. In S2D, we choose the disparity as the target for the CNN to regress rather than the depth. By doing so, we embed the scale correction into the training to better diminish the effect mentioned above. The depth is calculated from the disparity $\hat{\mathbf{D}}$ as follows:

$$\mathbf{Z}(\mathbf{u}_i) = \frac{B\ f_{train}}{\hat{\mathbf{D}}(\mathbf{u}_i)} \tag{2}$$

where B is a hyperparameter that can be seen as a “virtual” baseline. It controls the range of the depth to be regressed and is set to 0.1m in our implementation. Note that the disparity is linearly dependent on the inverse depth, which is well-known to have various statistical advantages and also



(a) (b) (c) (d)

Fig. 3: The sparse to dense procedure, figures above progressively shows the intermediate outputs: (a) optimized sparse depth image using CNN depth as prior; (b) CNN normal; (c) dense reconstruction using (a) and (b); (d) after (c) has been refined with adjacent keyframes.

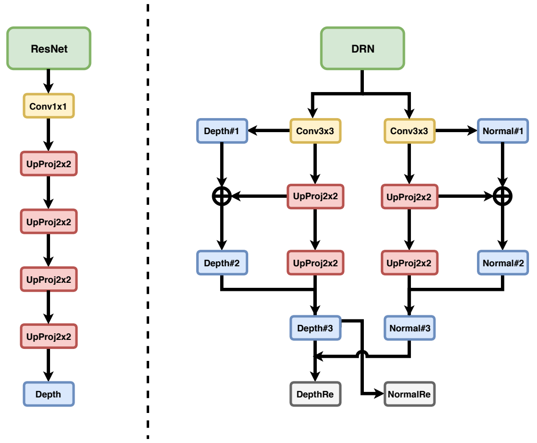


Fig. 4: Network structure comparison of the original FCRN (left) and our FCDRN (right).

converge better in our optimization. By using Eq.2, we decouple the focal length from the training. More importantly, the network now predicts the disparity from a fixed base line camera rather than a camera with a fixed focal length.

Depth-to-Normal. The depth-to-normal conversion is straightforward. As mentioned, the original least mean square estimation is slow. To accelerate the training, we adopt a common approach from SLAM and compute the normal using the cross product [40]:

$$\mathbf{N}_{re}(\mathbf{u}_i) = \nu[\mathbf{V}(u_i+1, v_i) - \mathbf{V}(u_i, v_i)) \times (\mathbf{V}(u_i, v_i+1) - \mathbf{V}(u_i, v_i))]$$
$$\mathbf{V}(u_i, v_i) = [x_i, y_i, z_i]^T = [\tfrac{z_i(u_i - c_x)}{f_x}, \tfrac{z_i(v_i - c_y)}{f_y}, z_i]^T \tag{3}$$

where $\nu[\mathbf{n}] = \mathbf{n}/||\mathbf{n}||_2$ and $\mathbf{V}(u, v)$ is the 3D vertex unprojected from image plane. Compared to the least mean square estimation used in [28], we did not find a notable difference in the quality of depth/normal estimation.

Normal-to-Depth. The normal-to-depth conversion is based on the assumption that points used to reproject the depth to the current position are locally on the same surface tangent:

$$n_{xi}(x - x_i) + n_{yi}(y - y_i) + n_{zi}(z - z_i) = 0 \tag{4}$$

Rearranging the equation and substituting x with a coplanar point x_j:

$$z_{ij} = \frac{n_{xj}x_j + n_{yj}y_j + n_{zj}z_j}{\frac{(u_i - c_x)n_{xj}}{f_x} + \frac{(v_i - c_y)n_{yj}}{f_y} + n_{zj}} \tag{5}$$

The above equation shows how the depth can be reprojected by using the normal and depth of neighbouring points. The final depth can then be computed using the weighted sum of every depth reprojected by the neighbouring points. The weighted sum for each pixel corresponds to a spatial filter for which the kernel weights are given by the inner products of normals of neighbouring points:

$$\mathbf{Z}_{re}(\mathbf{u}_i) = \frac{\sum_{j \in \mathcal{C}_i} \mathbf{n}_j^T \mathbf{n}_i z_{ij}}{\sum_{j \in \mathcal{C}_i} \mathbf{n}_j^T \mathbf{n}_i} \tag{6}$$

where $\mathcal{C}_i$ contains the pixels around u_i meeting the following condition:

$$\mathcal{C}_i = \{(x_j, y_j, z_j) | \mathbf{n}_j^T \mathbf{n}_i > \psi, |u_i - u_j| < \sigma, |v_i - v_j| < \sigma\} \tag{7}$$

where ψ is a threshold to remove non-coplanar points and σ is the spatial distance in the image plane. To be consistent with [28], they are set as 0.95 and 5 respectively.

Objective Functions. The overall objective function is as follows:

$$\mathcal{L}(\mathbf{u}_i) = \underbrace{\alpha\left(||\hat{\mathbf{D}}(\mathbf{u}_i) - \mathbf{D}^{gt}(\mathbf{u}_i)||_\epsilon + ||\hat{\mathbf{N}}(\mathbf{u}_i) - \mathbf{N}^{gt}(\mathbf{u}_i)||_\epsilon\right)}_{\text{Supervised Regression}}$$
$$\underbrace{+\beta\, ||\mathbf{D}_{re}(\mathbf{u}_i) - \mathbf{D}^{gt}(\mathbf{u}_i)||_\epsilon + \gamma\, ||\mathbf{N}_{re}(\mathbf{u}_i) - \mathbf{N}(\mathbf{u}_i)||_1}_{\text{Coupled Refinement}} \tag{8}$$

where $\mathbf{D}_{re}$ is computed from $\mathbf{Z}_{re}$ via Eq. 2 to be in the same scale as $\hat{\mathbf{D}}$ and $\mathbf{D}^{gt}/\mathbf{N}^{gt}$ are the ground truth disparity and normals respectively, $||.||_\epsilon$ is the huber loss and $||.||_1$ is the L1 loss. The threshold for the huber is set as 0.1 relative to the maximum absolute error. The hyper-parameters $\{\alpha, \beta, \gamma\}$ control the weights of the different terms and are set to $\{1.0, 0.1, 0.05\}$ in our implementation. The supervised term is the ordinary regression using ground truth disparity and normal. The coupled terms consists of the penalty on the reconstructed normal and depth. The supervised term is given higher weight because the corresponding information is more reliable. Henceforth, we denote the depth and normal predicted by the CNN as CNN depth and CNN normal to avoid confusion.

V. SLAM

In this section, we introduce the tracking frontend, deep sparse visual odometry, and the mapping backend, dense global fusion.

A. Deep Sparse Visual Odometry

For the visual odometry, our implementation is based on DSO [2][2]. In S2D, the CNN depth is used as a prior for the initialization of sparse points. We do not force DSO to initialize densely for the following reason: even if points in flat regions are activated using CNN depth, the uncertainty of those are unlikely to be reduced during the tracking since the gradients they contribute are relative small. In fact, we found that the tracking performance degrades if large amounts of low-gradient points are forced into the joint optimization.

Online Scale Correction. When a new key frame is required by DSO, we warp all visible mature points from active keyframes into the current image plane:

$$\begin{aligned} \mathbf{Z}_{opt}(\mathbf{u}_i) &= z_i^* \\ [u_i^*, v_i^*, z_i^*]^T &= V^{-1}(\mathbf{R}\,\mathbf{V}(u_i, v_i) + \mathbf{t}) \end{aligned} \tag{9}$$

where $[\mathbf{R}, \mathbf{t}] \in \mathbb{SE}(3)$ are the relative transforms between active keyframes and the new keyframe. They are estimated by direct image alignment of DSO. $\mathbf{Z}_{opt}$ is the warped optimized sparse depth image, it has been corrected in range and structure by tracking. We perform online scale corrections to the CNN depths using the scale changes observed in the sparsely optimized point:

$$\mathbf{Z}_{cor}(\mathbf{u}_i) = \mathbf{Z}(\mathbf{u}_i)\frac{\sum_{j\in\boldsymbol{\Omega}} B_j^{rel}\ z_j^*/z_j}{\sum_{j\in\boldsymbol{\Omega}} B_j^{rel}} \tag{10}$$

where B^{rel} is the maximum relative baseline from which the point has been observed. $\boldsymbol{\Omega}$ includes all mature points belonging to active keyframes and visible in the new keyframe. Z_{cor} is the rescaled CNN depth.

Sparse to Dense Filtering. During the training we used Eq. 6 to recast the depth of the points using depth and normal of other pixels around them inside a windows with predefined size. However, this approach is error prone when the input depth image is sparse. For example, in a scene where a desk stands on a flat floor, the depth of the edges on the desk can be propagated from the floor since they are equally "flat" in the same 3D direction. If so, the depth is recomputed from a wrong parallel surface rather than the actual coplanar tangent. To avoid this, a fast pre-segmentation into super-pixels is performed. The depth will only be filled by reprojecting from adjacent pixels within the same super-pixel. The new filtering criteria $\tilde{\mathcal{C}}$ is defined as:

$$\tilde{\mathcal{C}}_i = \{(x_j, y_j, z_j) | \mathbf{n}_j^T \mathbf{n}_i > \psi, c_i = c_j\} \tag{11}$$

where c is the label assigned by the super-pixel segmentation.

The overall filtering based sparse to dense reconstruction can be summarized into three steps: (1) filter $\mathbf{Z}_{opt}$ with CNN normal using Eq. 6 with the new criteria $\tilde{\mathcal{C}}$; (2) filter the updated $\mathbf{Z}_{opt}$ with bilateral filtering, the kernel weight is based on the color difference and spatial distance; (3) a wrap up filtering with CNN normal using Eq. 6 with original criteria $\mathcal{C}$. As mentioned in the previous section, the filtering is parallelized and performed on GPU, which allows us to meet the requirement of real time.

Step (1) can effectively diminish the incorrect depth reprojection. The downside of this is that it results in no value exchange between blobs. To tackle with this issue, step (2), a classical bilateral filtering is conducted. However, as the color based smoothing is not as reliable as the normal, we perform step (3) for further regularization. We denote the final reconstructed depth as $\mathbf{Z}_{dense}$ to distinguish it from the intermediate output $\mathbf{Z}_{re}$ (only used for training).

B. Dense Global Fusion

Keyframe-wise Refinement. In CNN-SLAM, based on LSD-SLAM [1], an uncertainty based update is used for dense depth refinement. We build on DSO instead. DSO uses a window based optimization scheme, containing a bundle of active keyframes for more robust estimation. It is very expensive to associate and update the dense depth and uncertainty using every single frame. However, the depth uncertainty has already been greatly reduced by the dense reconstruction which directly propagates the low uncertainty points using the geometrical structure. In Fig. 1, we see that a 3D reconstruction can be performed even **without** the refinement. That said, the refinement helps reject outliers and grant additional baseline stimulus in a dense manner and we therefore include it in our pipeline. However, we only perform the refinement between keyframes, and not between every frame. Specifically, we use the Bayesian Estimation based on REMODE [41][3] and estimate the uncertainty for each pixel based on the difference between updated depth and scale fixed CNN depth (Eq. 10).

As a final step in our SLAM system we deploy a fusion based method to build a global 3D model consisting of surfel splats. It is fed our refined dense depth images. Our implementation is based on ElasticFusion [40][4] with frame-to-frame tracking disabled since we only use it as an advanced mapper. As the point cloud is fused into the global model, transient noise can be further rejected and loop closures are handled.

VI. Experiments

In this section, we evaluate the effectiveness of our S2D framework on the TUM [42] and ICL-NUIM [43] RGB-D datasets. The Absolute Trajectory Error (ATE) and Percentage of Correct Depth (PCD) (also used in [3]) are used as metrics to compare with other learning/non-learning based monocular VO and SLAM systems.

The training of FCDRN is conducted using a desktop with an Intel i7-4790 processor and dual Nvidia 1080 graphic cards. The testing is done with a laptop with Intel i7-7700HQ and mobile version Nvidia 1070. The core of the

[2] `https://github.com/JakobEngel/dso`

[3] `https://github.com/uzh-rpg/rpg_open_remode`

[4] `https://github.com/mp3guy/ElasticFusion`

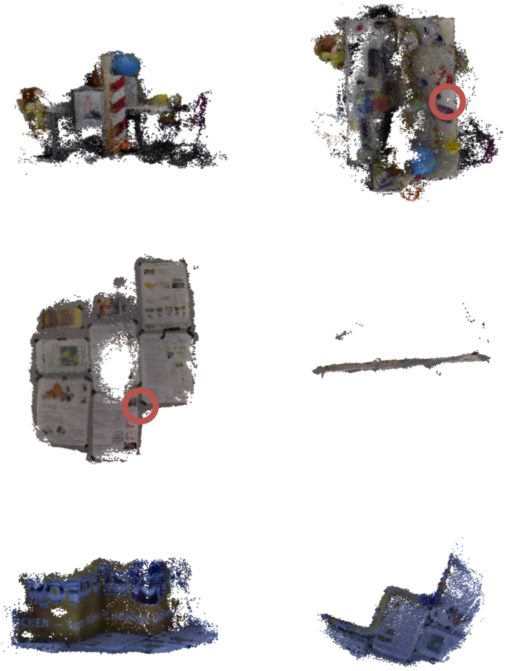


Fig. 5: Raw point cloud reconstruction examples of TUM seq1, 2 and 3 (top to bottom). The left and right columns show the front and top views, respectively. The red circles mark the area where loop closures are detected.

sparse to dense reconstruction is the normal based spatial filtering together with super-pixel segmentation and color based bilateral filtering. These steps can all be greatly accelerated by GPU computing. In the experiments, we did not find a notable frame drop with our implementation in C++ with CUDA. The running framerate of the overall system on our laptop was more than 23Hz using images of size 320×240. The inference time of FCDRN is around 25Hz (21Hz including copying from CPU to/back GPU) on our laptop.

A. Datasets

Training samples First we pre-trained FCDRN using 44624 RGB-D frames extracted from the SUN-3D [30] dataset. The SUN-3D dataset includes videos recorded in various typical indoor environments. We sampled roughly one frame per second from the video to avoid too repetitive training samples. Noisy images, e.g., mostly occluded or overexposed, cause the training to diverge. We use a standard SIFT keypoint detector to identify images that are likely to be noisy and discard images in which we find less than 50 SIFT keypoints. Then, we trained the network with the SUN-RGBD [44] dataset containing 10k refined RGB-D images which are collected from NYUv2 [29], Berkeley B3DO [45] and SUN-3D [30]. The training of FCDRN is implement using Pytorch. We used the Adam optimizer with learning rate 10^{-4} for the pre-training, and the same learning rate decayed by 2 every 20 epochs for the formal training. The overall training includes 20 epochs over the pre-trained dataset we extracted and 50 epochs using SUN-RGBD.

TABLE I: Absolute Trajectory Error

Datasets	S2D	DDSO	DSO [2]	CNN-SLAM [3]	LSD-B [1]	LSD [1]	ORB [1]	Laina [9]
TUM/seq1	**0.071**	0.552	1.221	0.542	1.717	1.826	1.206	0.809
TUM/seq2	**0.078**	0.203	0.123	0.243	0.106	0.436	0.495	1.337
TUM/seq3	**0.072**	0.335	0.648	0.214	0.037	0.937	0.733	0.724
ICL/office0	**0.132**	0.409	1.118	0.266	0.587	0.528	0.430	0.337
ICL/office1	**0.131**	0.155	0.633	0.157	0.790	0.768	0.780	0.218
ICL/office2	**0.085**	0.456	0.795	0.213	0.172	0.794	0.860	0.509
ICL/living0	**0.137**	0.143	0.404	0.196	0.894	0.516	0.493	0.230
ICL/living1	0.082	**0.028**	0.187	0.059	0.540	0.480	0.129	0.060
ICL/living2	**0.045**	0.162	0.668	0.323	0.211	0.667	0.663	0.380

Test sequences For the testing, we used the same sequences from TUM-RGBD and ICL-NUIM datasets as in the evaluation of CNN-SLAM [3]. The abbreviations TUM seq1 to seq3 refer to *long_office_household, nostructure_texture_near_withloop* and *structure_texture_far* of sensor $fr3$, respectively. From ICL-NUIM, the first 3 office and living room sequences are used for testing.

B. Quantitative Results

Absolute trajectory error (ATE). ATE, is a well-established metric for evaluating the quality of a predicted camera trajectory. It is defined as the root mean square error between the estimated and ground-truth camera trajectories. ATE directly shows the final performance of monocular visual tracking.

Tab. I shows the evaluated results. We compare the performance against the original DSO and DSO using depth from FCRN, denoted as DDSO with focal length adaption as in CNN-SLAM with Eq. 1. These two additional baselines can help further demonstrate the influence of different depth adaptions and the quality of the scale estimation.

Firstly, it can be seen that S2D outperforms the other methods in general. Obvious lower and more stable results have been obtained with S2D in almost all the test cases. In the exception, $ICL/living1$, the depth scale estimated by FCRN is really accurate, and as a result DDSO has the lowest ATE in this case. The overall results demonstrate the high quality of the scale estimated by our network, FCDRN. Our method wrap the focal length adaption into the training and perform more effective online scale correction on the run. In contrast, DDSO using the depth and adaption with CNN-SLAM only works on par with CNN-SLAM.

Percentage of Correct Depth (PCD). PCD is defined as the percentage of depth predictions whose absolute error is smaller than 10% of the ground truth depth. This reveals the quality of final depth of keyframes of our and other methods. The results are shown in Tab. II. We achieve better results than CNN-SLAM in all but two of the sequences and in many the difference is large. The sequences, $ICL/office1$ and $ICL/office2$, where CNN-SLAM is better than S2D, are from the artificially refined ICL dataset. On some of the datasets, e.g., $TUM/seq1$, $TUM/seq2$ and $ICL/living2$, the PCD of our method is more than twice that of CNN-SLAM, illustrating the impact of our geometrical sparse to dense reconstruction using the geometric normal. Not

TABLE II: Percentage of Correct Depth

Datasets	S2D	CNN-SLAM [3]	LSD-B [1]	LSD [1]	ORB [31]	Laina [9]	REMODE [41]
TUM/seq1	**53.287**	12.477	3.797	0.086	0.031	12.982	9.548
TUM/seq2	**66.628**	24.077	3.966	0.882	0.059	15.412	12.651
TUM/seq3	**37.683**	27.396	6.449	0.035	0.027	9.450	6.739
ICL/office0	**27.445**	19.410	0.603	0.335	0.018	17.194	4.479
ICL/office1	19.702	**29.150**	4.759	0.038	0.023	20.838	3.132
ICL/office2	27.059	**37.226**	1.435	0.078	0.040	30.639	16.708
ICL/living0	**19.337**	12.840	1.443	0.360	0.027	15.008	4.479
ICL/living1	**25.090**	13.038	3.030	0.057	0.021	11.449	2.427
ICL/living2	**68.907**	26.560	1.807	0.167	0.014	33.010	8.681

TABLE III: RMSEs on KITTI odometry dataset

	DSO		S2D		S2D fine-tuned	
Sequence No.	t_{rel}	r_{rel}	t_{rel}	r_{rel}	t_{rel}	r_{rel}
00	0.487	0.046	0.213	0.058	0.107	0.055
02	0.640	0.040	0.211	0.044	0.089	0.045
04	0.979	0.035	0.706	0.021	0.035	0.015
06	0.571	0.111	0.136	0.186	0.096	0.129
08	0.570	0.084	0.234	0.099	0.077	0.092
mean	0.649	0.063	0.300	0.082	0.081	0.067

surprising, S2D dramatically outperforms other classical methods shown for completeness in the table.

C. Qualitative Results

Fig. 5 shows top and front views of 3D constructions using S2D on the three TUM sequences. Shown in the top views, sharp edges are well preserved. This is accomplished by the verification via tracking of DSO and good quality of normal prediction for depth reprojecting. The original depth usually suffers from ambiguous boundaries. Both $TUM/seq1$ and $TUM/seq2$ have loop closures that have been detected (marked with red circles in Fig. 5). These closed loops also provide evidence that the monocular depth estimation of our method is consistent. A fusion based mapper requires correct alignment between the active and the global model. On the contrary, a pose graph based key frame management approach only need a minimum of two keyframes to be aligned. However, this does not necessarily mean that loop closure using pose graphs is easier for monocular SLAM. Quality is what matters and the deformation model of ElasticFusion helps to achieve this goal via rejecting outliers based on the surface quality and refining appearance based on the elastic deformative graph.

D. Discussion

In this subsection, we discuss the limitations of S2D. The overall performance of S2D relies on two major factors: the performance of visual tracking and the generalization ability of FCDRN for predicting depth/normals.

In the supplementary material, we present the ATE of S2D for all available TUM/ICL sequences to allow for future comparisons to S2D. We compare it with the results from ElasticFusion [40] which uses the captured depth image as input. S2D works well in general, but for some sequences, both S2D and ElasticFusion lose track. The reasons reported in [40] are valid also for S2D. The sequences in question exhibit a high rate of dropped frames and sudden high angular velocities, which mainly affect the image alignment and thus the tracking. As for the FCDRN, it struggles to generate accurate depth priors when the input image is close to textureless. These two challenges can be addressed, e.g., by using a global shutter camera with higher frame rate [2] and by combining the visual input with inertial data [46].

S2D has been developed for indoor use. To investigate how S2D generalizes to outdoor scenes we performed tests with the KITTI odometry dataset [47]. Tab. III shows the translational, $t_{rel}(ratio)$, and rotational, $r_{rel}(10^{-3} \times deg/m)$, RMSEs for some sequences. The official tool provided by [47] is used for the evaluation. We compared three methods: original DSO, S2D with FCDRN trained by the SUN dataset (indoor as above) and S2D with FCDRN fine-tuned using the same KITTI raw sequences as in [18]. The CNN depth is fine-tuned with the same loss function as in [21], [4]. The CNN normal is fine-tuned based on the coupled refinement term from Eq. 8 as ground truth depth is not available to calculate the normals. Note that we only finetune the CNN in S2D and do not change anything else in the system pipeline. Thanks to the robust tracking of DSO, the rotational errors are low for all three methods (note the scale of r_{rel}). On the other hand, there are significant differences in translational errors. S2D trained on indoor scenes is not working so well outdoors, as can be expected. We still see a clear improvement over DSO. When the CNN in S2D is fine-tuned for the outdoor environment, the translational errors are significantly reduced. The t_{rel} is now on par with results of DSO with state-of-the-art depth priors for single view images, achieving an average of 0.107 according to [4]. To fully convert S2D from indoor use to outdoor use, one should take into account that stereo data is the main source for training in outdoor environments, in contrast to indoor scenes where RGB-D data dominates. One could redesign FCDRN to additionally predict a virtual stereo pair and incorporate the impressive results of DVSO [4]. However, we want to emphasize that the focus in this paper is utilizing the learning based geometrical information to densely reconstruct scenes from corrected sparse depth and that we target indoor scenes.

VII. Summary and Conclusions

In this paper, a new deep learning based monocular SLAM method is proposed. A single CNN has been trained to predict depth and normals in a coupled way. The depth is used in the projective geometric optimization for accurate pose estimation. The normals are utilized for a dense geometrical reconstruction using intermediate sparse optimized point clouds. Experiments demonstrated the effectiveness of our method, S2D. Both improved motion estimation and dense depth reconstruction are achieved in comparison with state-of-the-art deep dense monocular SLAM.

In future work, we plan to investigate including the camera pose estimation in the depth/normal training scheme. S2D is not limited to mapping with a single camera, it can potentially be used for reconstruction with multiple sensors having sparse depth measurements, e.g., camera and Lidar. We also plan to investigate how to support human interaction with the dense 3D reconstruction. To achieve this goal, we will study exploiting the semantics of the environment in the model.

References

[1] J. Engel, T. Schöps, and D. Cremers, "LSD-SLAM: Large-scale direct monocular SLAM," in *"IEEE Intl. Conf. on Computer Vision (ICCV)"*, September 2014.

[2] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE Trans. Pattern Anal. Mach. Intell*, March 2018.

[3] K. Tateno, F. Tombari, I. Laina, and N. Navab, "Cnn-slam: Real-time dense monocular slam with learned depth prediction," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, vol. 2, 2017.

[4] N. Yang, R. Wang, J. Stückler, and D. Cremers, "Deep virtual stereo odometry: Leveraging deep depth prediction for monocular direct sparse odometry," *European Conf. on Computer Vision (ECCV)*, 2018.

[5] A. Saxena, M. Sun, and A. Y. Ng, "Make3d: Learning 3d scene structure from a single still image," *IEEE Trans. Pattern Anal. Mach. Intell*, vol. 31, 2009.

[6] D. Hoiem, A. A. Efros, and M. Hebert, "Automatic photo pop-up," in *ACM Trans. on Graphics*, vol. 24. ACM, 2005.

[7] D. Eigen, C. Puhrsch, and R. Fergus, "Depth map prediction from a single image using a multi-scale deep network," in *Advances in Neural Information Processing Systems (NIPS)*, 2014.

[8] F. Liu, C. Shen, G. Lin, and I. D. Reid, "Learning depth from single monocular images using deep convolutional neural fields." *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, 2016.

[9] I. Laina, C. Rupprecht, V. Belagiannis, F. Tombari, and N. Navab, "Deeper depth prediction with fully convolutional residual networks," in *Intl. Conf. on 3D Vision (3DV)*. IEEE, 2016.

[10] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[11] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[12] M. Liu, M. Salzmann, and X. He, "Discrete-continuous depth estimation from a single image," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[13] B. Liu, S. Gould, and D. Koller, "Single image depth estimation from predicted semantic labels," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2010.

[14] P. Wang, X. Shen, Z. Lin, S. Cohen, B. Price, and A. L. Yuille, "Towards unified depth and semantic prediction from a single image," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[15] W. Zhuo, M. Salzmann, X. He, and M. Liu, "Indoor scene structure analysis for single image depth estimation," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[16] R. Garg, V. K. BG, G. Carneiro, and I. Reid, "Unsupervised cnn for single view depth estimation: Geometry to the rescue," in *European Conf. on Computer Vision (ECCV)*. Springer, 2016.

[17] J. Xie, R. Girshick, and A. Farhadi, "Deep3d: Fully automatic 2d-to-3d video conversion with deep convolutional neural networks," in *European Conf. on Computer Vision (ECCV)*. Springer, 2016.

[18] C. Godard, O. Mac Aodha, and G. J. Brostow, "Unsupervised monocular depth estimation with left-right consistency," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, vol. 2, 2017.

[19] Y. Luo, J. Ren, M. Lin, J. Pang, W. Sun, H. Li, and L. Lin, "Single view stereo matching," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[20] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe, "Unsupervised learning of depth and ego-motion from video," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, vol. 2, 2017.

[21] Y. Kuznietsov, J. Stückler, and B. Leibe, "Semi-supervised deep learning for monocular depth map prediction," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[22] X. Wang, D. Fouhey, and A. Gupta, "Designing deep networks for surface normal estimation," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[23] D. Eigen and R. Fergus, "Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture," in *IEEE Intl. Conf. on Computer Vision (ICCV)*, 2015.

[24] B. Li, C. Shen, Y. Dai, A. Van Den Hengel, and M. He, "Depth and surface normal estimation from monocular images using regression on deep features and hierarchical crfs," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[25] A. Bansal, B. Russell, and A. Gupta, "Marr revisited: 2d-3d alignment via surface normal prediction," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[26] A. Bansal, X. Chen, B. Russell, A. Gupta, and D. Ramanan, "Pixelnet: Representation of the pixels, by the pixels, and for the pixels," *arXiv preprint arXiv:1702.06506*, 2017.

[27] P. Wang, X. Shen, B. Russell, S. Cohen, B. Price, and A. L. Yuille, "Surge: Surface regularized geometry estimation from a single image," in *Advances in Neural Information Processing Systems (NIPS)*, 2016.

[28] X. Qi, R. Liao, Z. Liu, R. Urtasun, and J. Jia, "Geonet: Geometric neural network for joint depth and surface normal estimation," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[29] P. K. Nathan Silberman, Derek Hoiem and R. Fergus, "Indoor segmentation and support inference from rgbd images," in *European Conf. on Computer Vision (ECCV)*, 2012.

[30] J. Xiao, A. Owens, and A. Torralba, "Sun3d: A database of big spaces reconstructed using sfm and object labels," in *IEEE Intl. Conf. on Computer Vision (ICCV)*, 2013.

[31] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: an open-source SLAM system for monocular, stereo and RGB-D cameras," *IEEE Trans. on Robotics*, vol. 33, 2017.

[32] C. Forster, Z. Zhang, M. Gassner, M. Werlberger, and D. Scaramuzza, "Svo: Semidirect visual odometry for monocular and multicamera systems," *IEEE Trans. on Robotics*, vol. 33, 2017.

[33] H. Strasdat, J. Montiel, and A. J. Davison, "Scale drift-aware large scale monocular slam," *Robotics: Science and Systems VI*, vol. 2, 2010.

[34] J. Engel, J. Sturm, and D. Cremers, "Semi-dense visual odometry for a monocular camera," in *European Conf. on Computer Vision (ECCV)*, December 2013.

[35] S. Wang, R. Clark, H. Wen, and N. Trigoni, "Deepvo: Towards end-to-end visual odometry with deep recurrent convolutional neural networks," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*. IEEE, 2017.

[36] X. Yin, X. Wang, X. Du, and Q. Chen, "Scale recovery for monocular visual odometry using depth estimated with deep convolutional neural fields," in *IEEE Intl. Conf. on Computer Vision (ICCV)*, 2017.

[37] M. Bloesch, J. Czarnowski, R. Clark, S. Leutenegger, and A. J. Davison, "Codeslam-learning a compact, optimisable representation for dense visual slam," *arXiv preprint arXiv:1804.00874*, 2018.

[38] R. Li, S. Wang, Z. Long, and D. Gu, "Undeepvo: Monocular visual odometry through unsupervised deep learning," *arXiv preprint arXiv:1709.06841*, 2017.

[39] F. Yu, V. Koltun, and T. Funkhouser, "Dilated residual networks," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[40] T. Whelan, R. F. Salas-Moreno, B. Glocker, A. J. Davison, and S. Leutenegger, "Elasticfusion: Real-time dense slam and light source estimation," *Intl. Journal of Robotics Research*, vol. 35, 2016.

[41] M. Pizzoli, C. Forster, and D. Scaramuzza, "Remode: Probabilistic, monocular dense reconstruction in real time," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*. IEEE, 2014.

[42] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of rgb-d slam systems," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, Oct. 2012.

[43] 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.

[44] S. Song, S. P. Lichtenberg, and J. Xiao, "Sun rgb-d: A rgb-d scene understanding benchmark suite," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[45] A. Janoch, S. Karayev, Y. Jia, J. T. Barron, M. Fritz, K. Saenko, and T. Darrell, "A category-level 3d object dataset: Putting the kinect to work," in *Intl. Conf. on Computer Vision-Workshop on Consumer Depth Cameras for Computer Vision (ICCV)*, 2011.

[46] L. von Stumberg, V. Usenko, and D. Cremers, "Direct sparse visual-inertial odometry using dynamic marginalization," *arXiv preprint arXiv:1804.05625*, 2018.

[47] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," in *The International Journal of Robotics Research*, 2013.