CNN-SLAM: Real-time dense monocular SLAM with learned depth prediction (2017)

Authors: Tateno, Tombari, Laina, Navab

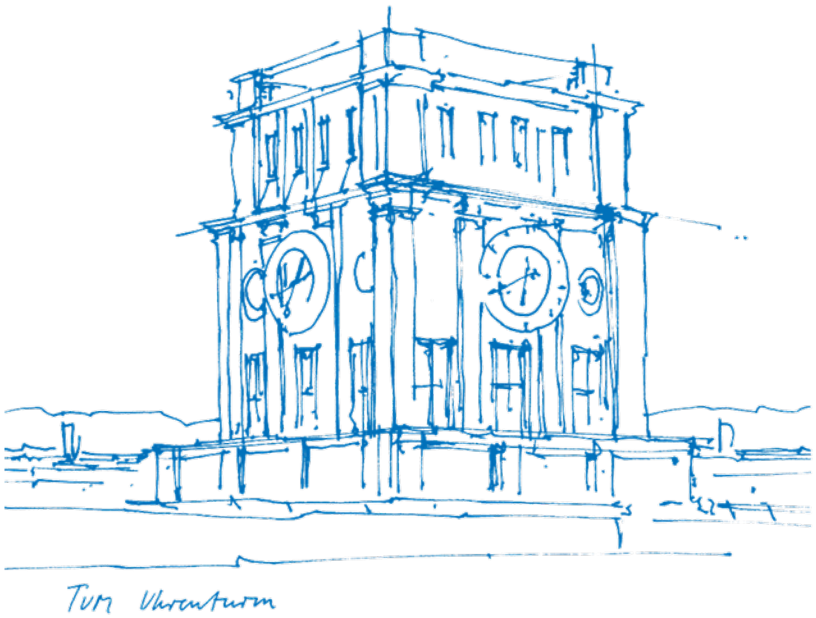
Seminar: The Evolution of Motion Estimation and Real-time 3D Reconstruction

Chair of Computer Vision & Artificial Intelligence

Technische Universität München

Juan Díez

January 12, 2021



Outline

1. Introduction
2. Related work
3. Method description
4. Experiments and results
5. Personal comments
6. Summary

1. Introduction

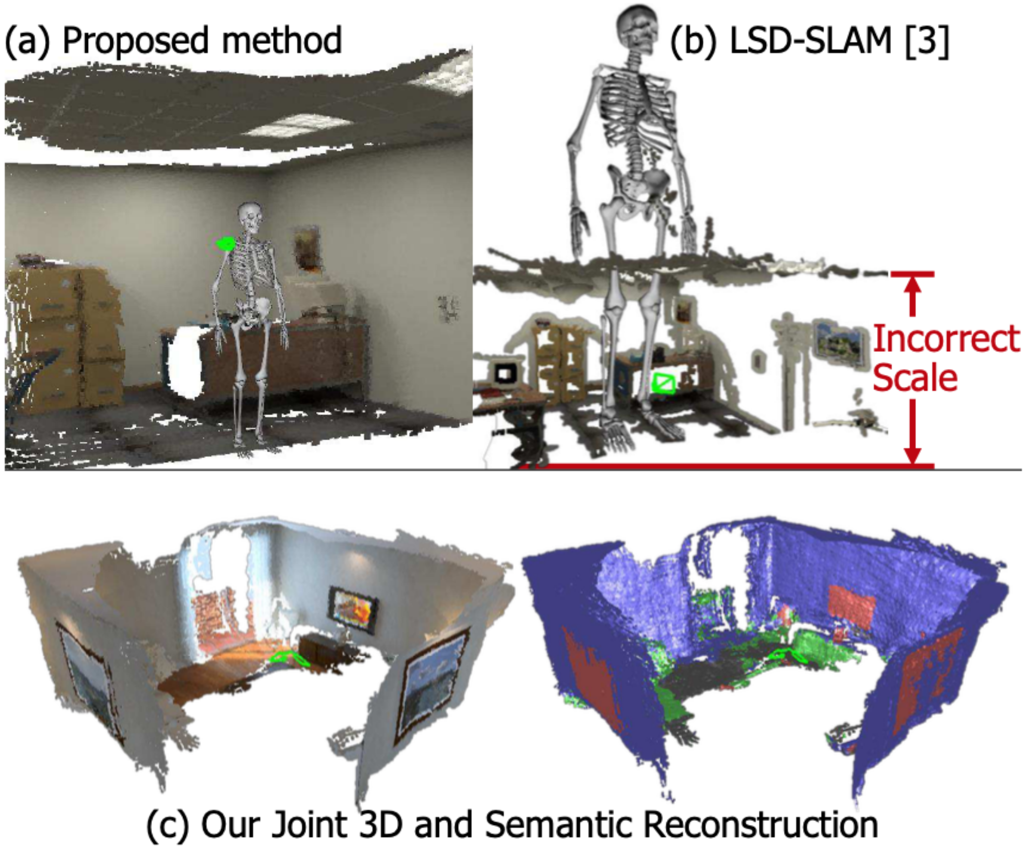
1. Introduction

1. Introduction

Main contributions of the paper:

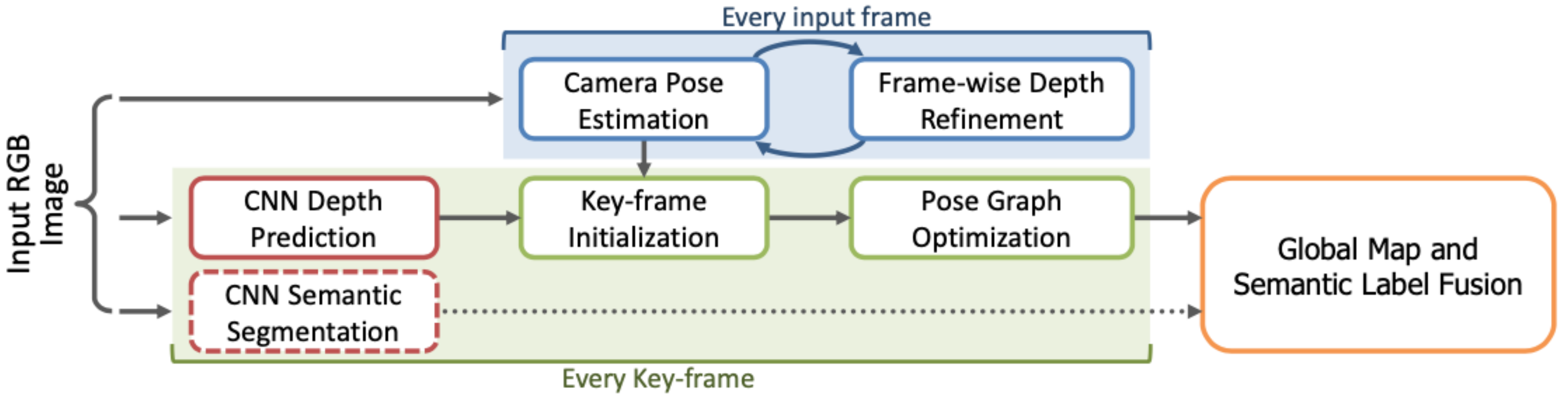
- Combine depth prediction via Convolutional Neural Network (CNN) with small baseline stereo depth prediction (best of both worlds).
- Solve the scale ambiguity issue via domain specific knowledge learned by the CNN (estimate absolute scale of reconstruction).
- Semantic labeling via CNN seamlessly integrated with dense SLAM (joint 3D and semantic reconstruction).
- Significantly outperform state-of-the-art methods in the benchmarks.

1. Introduction



2. Related work

- Engel et al. (2014) (LSD-SLAM) inspired, **but** CNN-SLAM has dense depth map.
- Laina et al. (2014) already used CNN, **but** without refinement (blurring artifacts, lacking shape details). CNN-SLAM uses the same network architecture.
- Engel et al. (2013) frame-wise depth refinement scheme is used, **but** CNN-SLAM refines every element of the key-frame (dense depth map).
- Semantic label fusion similar to Tateno et al. (2015).



3.1. Camera Pose Estimation

- Weighted Gauss-Newton optimization on the objective function

$$E(\boldsymbol{T}_t^{k_i}) = \sum_{\tilde{\boldsymbol{u}} \in \Omega} \rho\left(\frac{r\left(\tilde{\boldsymbol{u}}, \boldsymbol{T}_t^{k_i}\right)}{\sigma\left(r\left(\tilde{\boldsymbol{u}}, \boldsymbol{T}_t^{k_i}\right)\right)}\right)$$

with r, the photometric residual, defined as

$$r(\tilde{\boldsymbol{u}}, \boldsymbol{T}_t^{k_i}) = \mathcal{I}_{k_i}(\tilde{\boldsymbol{u}}) - \mathcal{I}_t\left(\pi\left(\boldsymbol{K}\boldsymbol{T}_t^{k_i}\tilde{\mathcal{V}}_{k_i}(\tilde{\boldsymbol{u}})\right)\right)$$

and $\mathcal{V}_{k_i}$, the 3D vertex map, as

$$\mathcal{V}_{k_i}(\boldsymbol{u}) = \boldsymbol{K}^{-1}\dot{\boldsymbol{u}}\mathcal{D}_{k_i}(\boldsymbol{u})$$

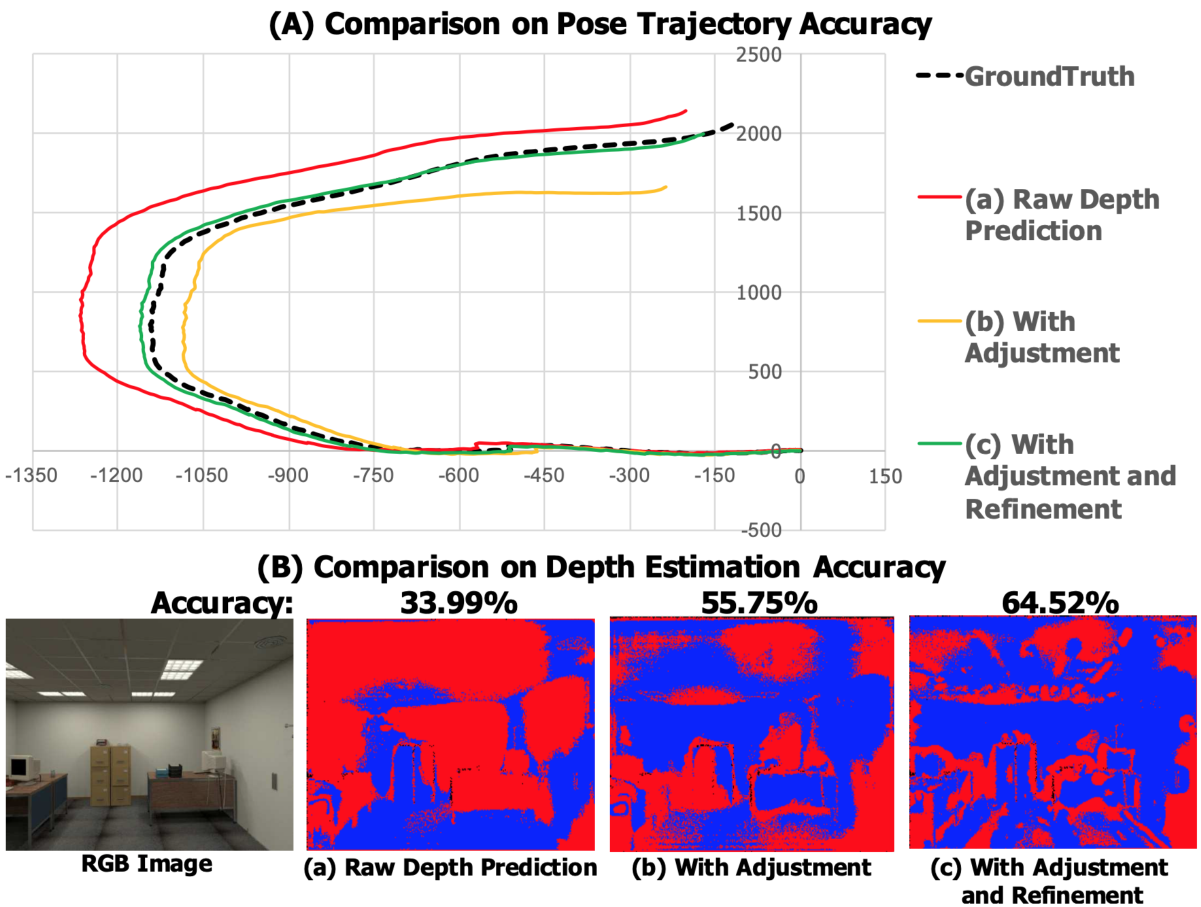
- σ is a function measuring the residual uncertainty:

$$\sigma = \sqrt{\sigma_{\mathcal{I}}^2 + \left(\frac{\partial r(\tilde{\boldsymbol{u}}, \boldsymbol{T}_t^{k_i})}{\partial \mathcal{D}_{k_i}(\tilde{\boldsymbol{u}})}\right)^2 \sigma_{\mathcal{D}_{k_i}}(\tilde{\boldsymbol{u}})}$$

3.2. CNN-based Depth Prediction and Semantic Segmentation

- CNN architecture for depth prediction:
  - Based on ResNet-50 and initialized with pre-trained weights on ImageNet dataset.
  - Decapitate ResNet-50 last pooling and fully connected layers.
  - Replace them by up-sampling blocks: combination of unpooling and convolutional layers.
  - Result: a Fully Convolutional Network (FCN) with all the layers of the first part (before up-sampling) already trained.
  - Retrain for depth prediction on the NYU Depth v2 dataset.
- Same CNN architecture for semantic segmentation except for:
  - Last layer that has as many output channels as number of categories.
  - Soft-max layer at the end to get the mode of a nice probability distribution.
  - Cross-entropy loss minimized by Stochastic Gradient Descent (SGD).

3.3. Key-frame Creation and Pose Graph Optimization



3.3. Key-frame Creation and Pose Graph Optimization

- Adjust depth estimation by ratio of focal length:

$$\mathcal{D}_{k_i}(\boldsymbol{u}) = \frac{f_{cur}}{f_{tr}}\tilde{\mathcal{D}}_{k_i}(\boldsymbol{u})$$

- Uncertainty map associated to depth map of key-frame k_i w.r.t nearest key-frame k_j:

$$\mathcal{U}_{k_i}(\boldsymbol{u}) = \left(\mathcal{D}_{k_i}(\boldsymbol{u}) - \mathcal{D}_{k_j}\left(\pi\left(\boldsymbol{K}\boldsymbol{T}_{k_j}^{k_i}\mathcal{V}_{k_i}(\boldsymbol{u})\right)\right)\right)^2$$

- Propagated uncertainty map from nearest key-frame k_j:

$$\tilde{\mathcal{U}}_{k_j}(\boldsymbol{v}) = \frac{\mathcal{D}_{k_j}(\boldsymbol{v})}{\mathcal{D}_{k_i}(\boldsymbol{u})}\mathcal{U}_{k_j}(\boldsymbol{v}) + \sigma_p^2$$

with $\boldsymbol{v} = \pi\left(\boldsymbol{K}\boldsymbol{T}_{k_j}^{k_i}\tilde{\mathcal{V}}_{k_i}(\tilde{\boldsymbol{u}})\right)$, and σ_p^2 is Gaussian noise.

3.3. Key-frame Creation and Pose Graph Optimization

- Fuse depth maps according to weighted scheme:

$$\mathcal{D}_{k_i}(\boldsymbol{u}) = \frac{\tilde{\mathcal{U}}_{k_j}(\boldsymbol{v}) \cdot \mathcal{D}_{k_i}(\boldsymbol{u}) + \mathcal{U}_{k_i}(\boldsymbol{u}) \mathcal{D}_{k_j}(\boldsymbol{v})}{\mathcal{U}_{k_i}(\boldsymbol{u}) + \tilde{\mathcal{U}}_{k_j}(\boldsymbol{v})}$$

$$\mathcal{U}_{k_i}(\boldsymbol{u}) = \frac{\tilde{\mathcal{U}}_{k_j}(\boldsymbol{v}) \cdot \mathcal{U}_{k_i}(\boldsymbol{u})}{\mathcal{U}_{k_i}(\boldsymbol{u}) + \tilde{\mathcal{U}}_{k_j}(\boldsymbol{v})}$$

- Furthermore, the pose graph is updated at each new key-frame (edge creation based on small relative pose).

- And the pose of key-frames is refined via pose graph optimization.

3.4. Frame-wise Depth Refinement

- Refinement of key-frame depth map (and uncertainty map) via small-baseline stereo matching with every new frame.
- $\mathcal{D}_t$ and $\mathcal{U}_t$ are computed by enforcing color consistency minimization between a key-frame and associated input frames (5-pixel matching along the epipolar line).
- Update key-frame depth map and uncertainty map:

$$\mathcal{D}_{k_i}(\boldsymbol{u}) = \frac{\mathcal{U}_t(\boldsymbol{u}) \cdot \mathcal{D}_{k_i}(\boldsymbol{u}) + \mathcal{U}_{k_i}(\boldsymbol{u}) \mathcal{D}_t(\boldsymbol{u})}{\mathcal{U}_{k_i}(\boldsymbol{u}) + \mathcal{U}_t(\boldsymbol{u})}$$

$$\mathcal{U}_{k_i}(\boldsymbol{u}) = \frac{\tilde{\mathcal{U}}_t(\boldsymbol{u}) \cdot \mathcal{U}_{k_i}(\boldsymbol{u})}{\mathcal{U}_{k_i}(\boldsymbol{u}) + \tilde{\mathcal{U}}_t(\boldsymbol{u})}$$

- $\mathcal{D}_t$ and $\mathcal{U}_t$ are already aligned with the key-frame based on the camera pose $\boldsymbol{T}_t^{k_i}$.

3.5. Global Model and Semantic Label Fusion

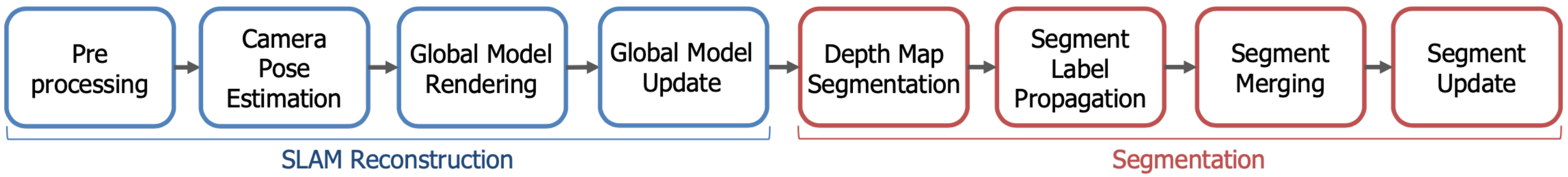


Fig. 2. Flow diagram of the proposed incremental segmentation pipeline applied at each input depth map.

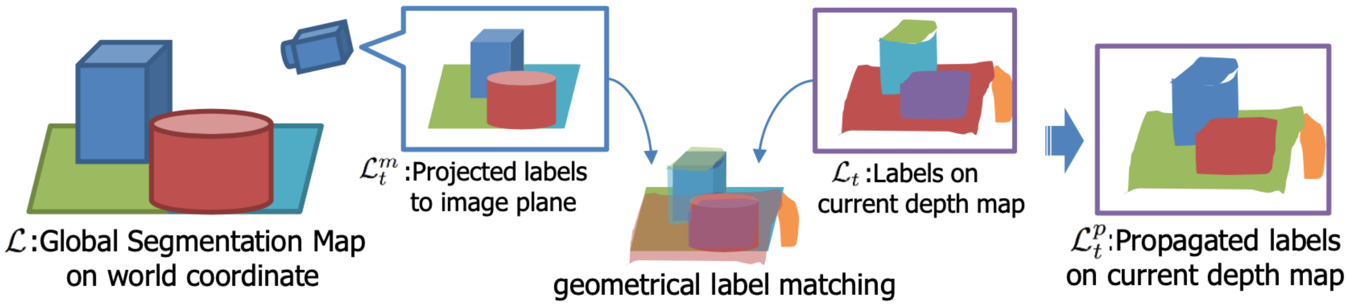
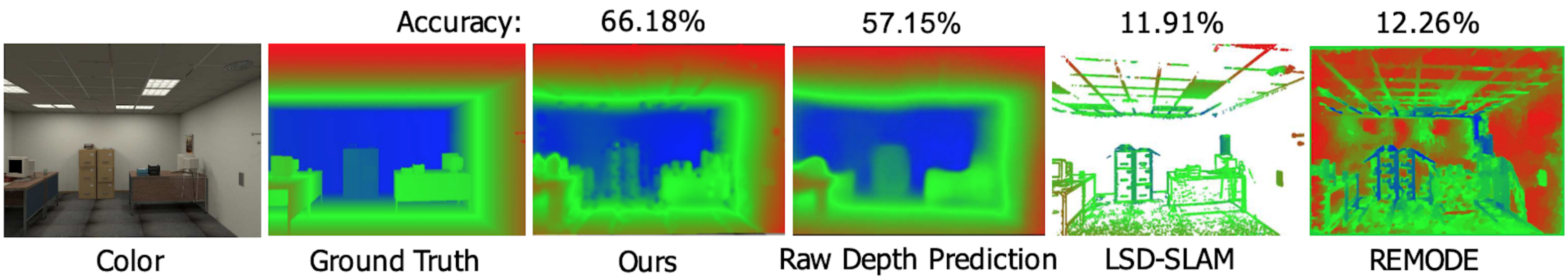


Fig. 4. Toy example explaining the proposed Segment Propagation stage: first segments from the GSM are re-projected onto the current camera plane; then, they are compared with those of the current depth map, so to identify corresponding segments between GSM and the camera plane.

4. Experiments and results



- LSD-SLAM provides very nice boundaries/shape details (high gradient information), but it is very sparse.
- Raw Depth Prediction via CNN is dense but very blurry.
- With the proposed refinement approach (Ours) we can achieve both dense and detailed depth prediction (best of both worlds).

4. Experiments and results

Table 1. Comparison in terms of Absolute Trajectory Error [m] and percentage of correctly estimated depth on ICL-NUIM and TUM datasets (TUM/seq1: *fr3/long_office_household*, TUM/seq2: *fr3/nostructure_texture_near_withloop*, TUM/seq3: *fr3/structure_texture_far*.

	Abs. Trajectory Error					**Perc. Correct Depth**					
	Our Method	LSD-BS [4]	LSD [4]	ORB [20]	Laina [16]	Our Method	LSD-BS [4]	LSD [4]	ORB [20]	Laina [16]	Remode [23]
ICL/office0	**0.266**	0.587	0.528	0.430	0.337	**19.410**	0.603	0.335	0.018	17.194	4.479
ICL/office1	**0.157**	0.790	0.768	0.780	0.218	**29.150**	4.759	0.038	0.023	20.838	3.132
ICL/office2	0.213	**0.172**	0.794	0.860	0.509	**37.226**	1.435	0.078	0.040	30.639	16.7081
ICL/living0	**0.196**	0.894	0.516	0.493	0.230	12.840	1.443	0.360	0.027	**15.008**	4.479
ICL/living1	**0.059**	0.540	0.480	0.129	0.060	**13.038**	3.030	0.057	0.021	11.449	2.427
ICL/living2	0.323	**0.211**	0.667	0.663	0.380	26.560	1.807	0.167	0.014	**33.010**	8.681
TUM/seq1	**0.542**	1.717	1.826	1.206	0.809	12.477	3.797	0.086	0.031	**12.982**	9.548
TUM/seq2	0.243	**0.106**	0.436	0.495	1.337	**24.077**	3.966	0.882	0.059	15.412	12.651
TUM/seq3	0.214	**0.037**	0.937	0.733	0.724	**27.396**	6.449	0.035	0.027	9.450	6.739
Avg.	**0.246**	0.562	0.772	0.643	0.512	**22.464**	3.032	0.226	0.029	18.452	7.649

4. Experiments and results

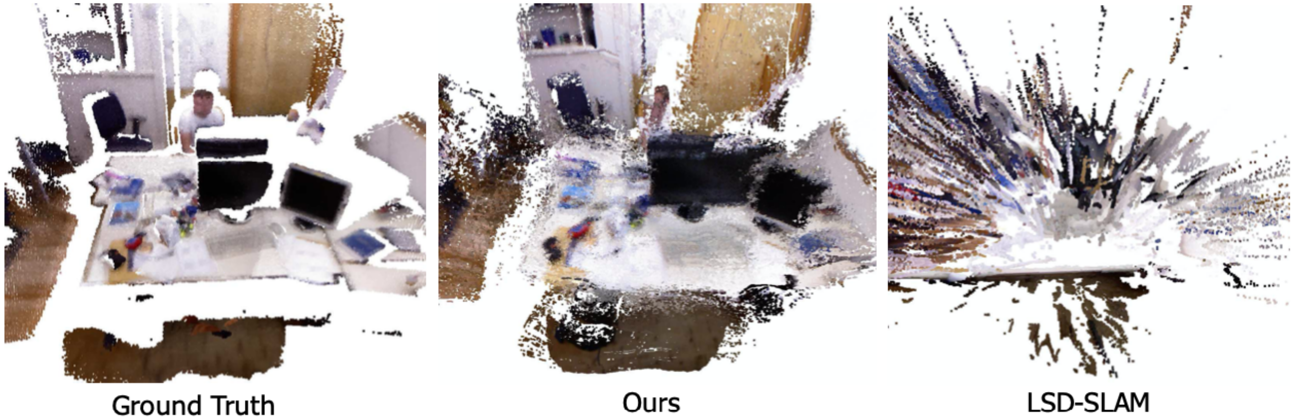


Figure 5. Comparison on a sequence that includes mostly pure rotational camera motion between the reconstruction obtained by ground truth depth (left), proposed method (middle) and LSD-SLAM [4] (right).

4. Experiments and results

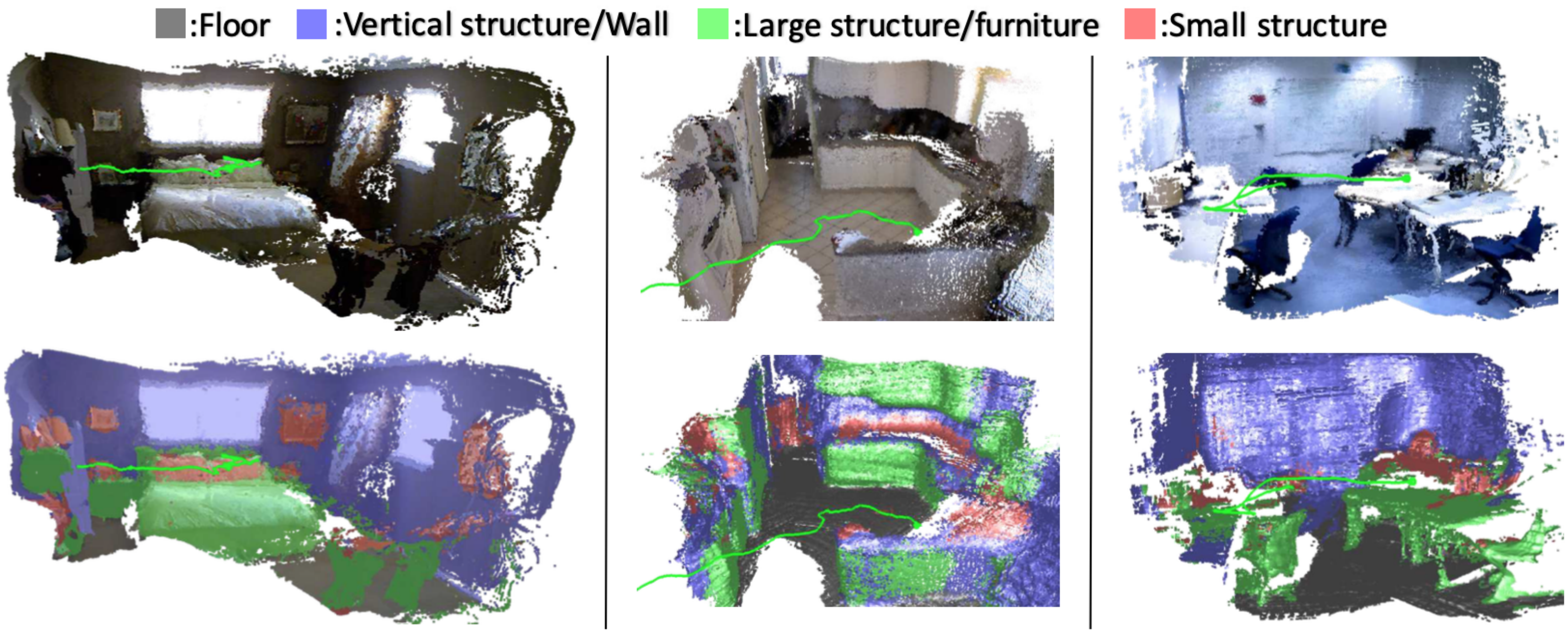


Figure 6. The results of reconstruction and semantic label fusion on the office sequence (top, acquire by our own) and one sequence (*kitchen_0046*) from the *NYU Depth V2* dataset [25] (bottom). Reconstruction is shown with colors (left) and with semantic labels (right).

5. Personal comments

- Absolute scale information allows avoiding scale-drift. It would be nice to compare results with the ones obtained with ORB-SLAM for specific hard sequences. I expect CNN-SLAM performs significantly better.
- Focal length adjustment is very easy to implement, generalize to different cameras and provides additional accuracy.
- The system seems quite big, I can imagine it is hard to implement/maintain/update. On the other hand, the modular architecture helps to alleviate this issue and the system makes a good use of resources when deployed (CPU + GPU).
- At some point the authors claim the training set and validation set are completely different. I am skeptical about this.
- Careful not to forget/ignore the assumptions the system makes, e.g., camera velocity, texture of the world, brightness consistency or representativity of the training set.
- One could argue comparison to other SLAM systems may be a little unfair, since this one has additional information about absolute scale. However, this is precisely the point of introducing the CNN.

6. Summary

- The proposed system solves both KSLAM and Semantic Segmentation.
- Depth prediction via CNN (FCN based on Resnet50) on keyframes, and via small baseline stereo on other frames.
- Absolute scale information is incorporated into the model by the CNN, overcoming major limitation of traditional SLAM systems.
- Blurry areas of CNN depth maps refined with weighted scheme integrating information from nearby frames.
- The refinement further improves the accuracy.
- Output of KSLAM is used as input for Semantic Segmentation.
- Same CNN architecture used for Semantic Segmentation, only change last layers.
- Real-time capable system using both CPU and GPU.

References

Engel, J., Schöps, T., & Cremers, D. (2014, September). LSD-SLAM: Large-scale direct monocular SLAM. In *European conference on computer vision (eccv).*

Engel, J., Sturm, J., & Cremers, D. (2013). Semi-dense visual odometry for a monocular camera. In *2013 ieee international conference on computer vision* (p. 1449-1456). doi: 10.1109/ICCV.2013.183

Long, J., Shelhamer, E., & Darrell, T. (2014). Fully convolutional networks for semantic segmentation. *CoRR*, *abs/1411.4038*. Retrieved from `http://arxiv.org/abs/1411.4038`

Tateno, K., Tombari, F., Laina, I., & Navab, N. (2017). CNN-SLAM: real-time dense monocular SLAM with learned depth prediction. *CoRR*, *abs/1704.03489*. Retrieved from `http://arxiv.org/abs/1704.03489`

Tateno, K., Tombari, F., & Navab, N. (2015). Real-time and scalable incremental segmentation on dense slam. In *2015 ieee/rsj international conference on intelligent robots and systems (iros)* (p. 4465-4472). doi: 10.1109/IROS.2015.7354011

Questions

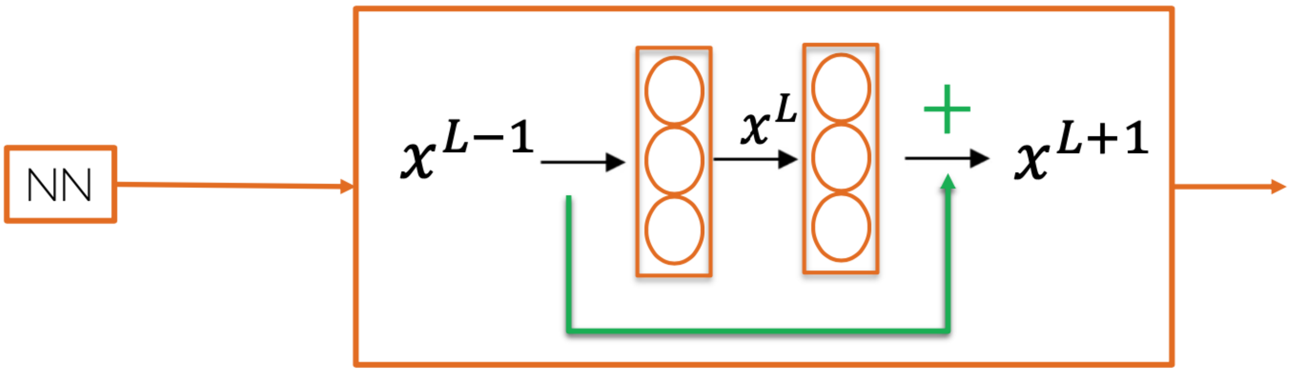
?

Notation

- $\pi : \mathbb{R}^3 \to \mathbb{R}^2, \pi([xyz]^T) = (x/z, y/z)^T$ (homogeneous coord. to Cartesian coord. projection).
- $\boldsymbol{K}$ (camera intrinsic matrix).
- ρ (Huber norm) (robust average of L1 and L2 norms).
- $\mathcal{K} = \{k_1, \dots, k_n\}, n \in \mathbb{N}$ (set of key-frames).
- $\boldsymbol{u} = (x, y) \in \Omega$ (generic depth map element).
- $\dot{\boldsymbol{u}} \in \mathbb{R}^3$ (homogeneous representation of $\boldsymbol{u}$).
- $\tilde{\boldsymbol{u}} \subset \boldsymbol{u} \in \Omega$ (image domain subset with high color gradients).
- $t \in \mathbb{N}$ (time step).
- $\boldsymbol{R}_t \in \mathbb{SO}(3)$ (rotation matrix in the 3D Special Orthogonal group).
- $\boldsymbol{t}_t \in \mathbb{R}^3$ (translation vector).
- $\boldsymbol{T}_t^{k_i} = [\boldsymbol{R}_t, \boldsymbol{t}_t] \in \mathbb{SE}(3)$ (transformation between nearest key-frame k_i and frame t, in the 3D Rigid Body Transformations group).
- $\mathcal{I}_t : \Omega \to \mathbb{R}^3$ (intensity image of frame t).
- $\mathcal{D}_{k_i}$ (depth map of key-frame k_i, computed by CNN).

ResNet

Why do ResNets work?



- The identity is easy for the residual block to learn
- Guaranteed it will not hurt performance, can only improve

I2DL: Prof. Niessner, Dr. Dai 65

ResNet

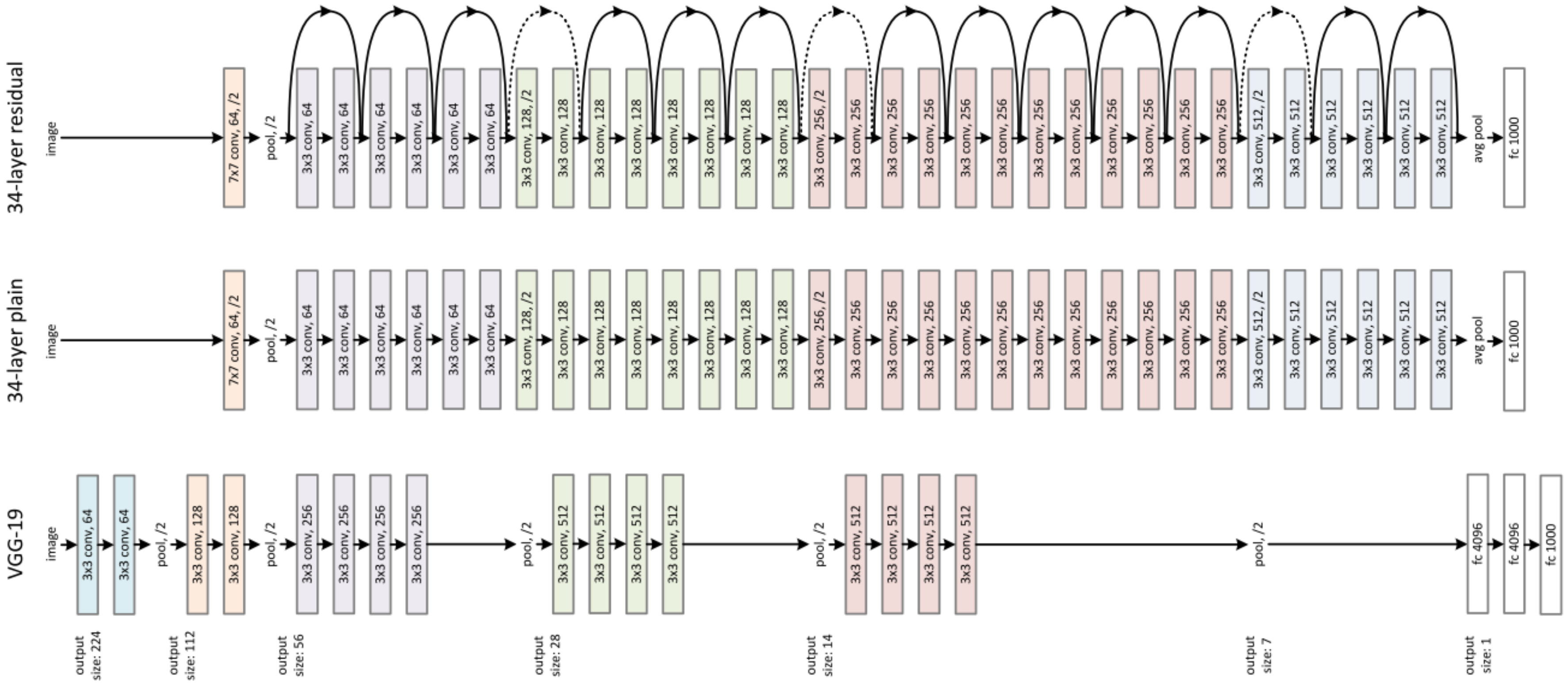
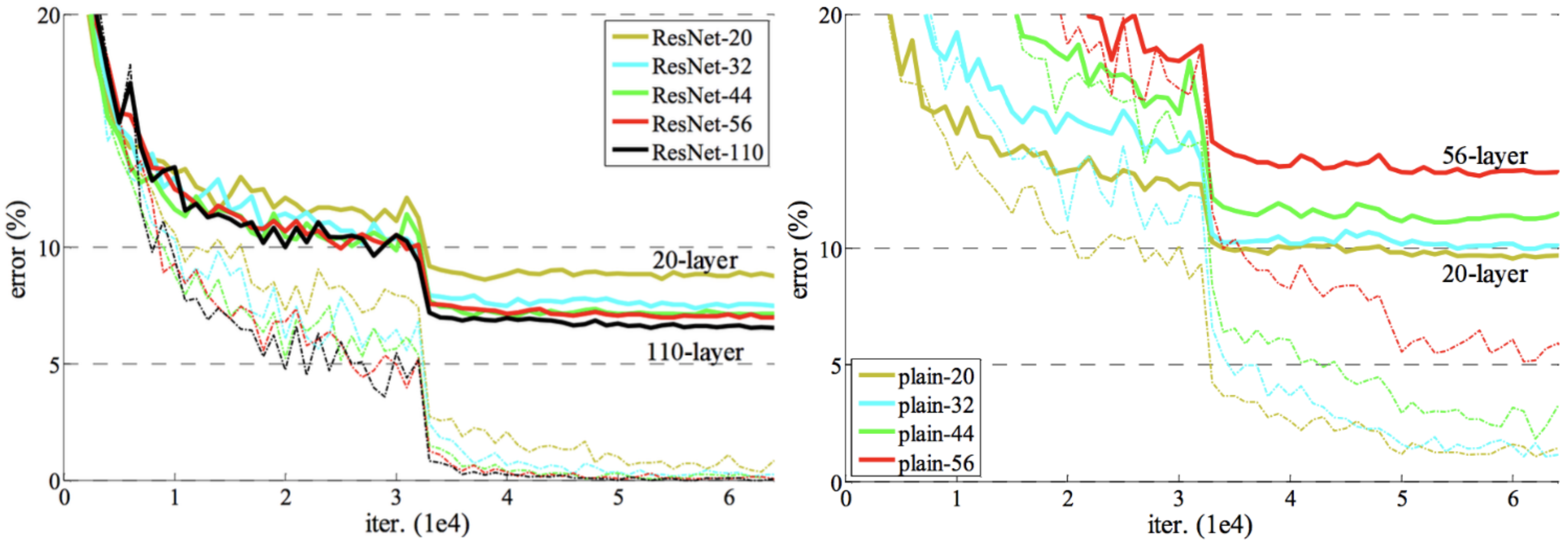


Figure: https://www.kaggle.com/keras/resnet50

ResNet

ResNet

- If we make the network deeper, at some point performance starts to degrade



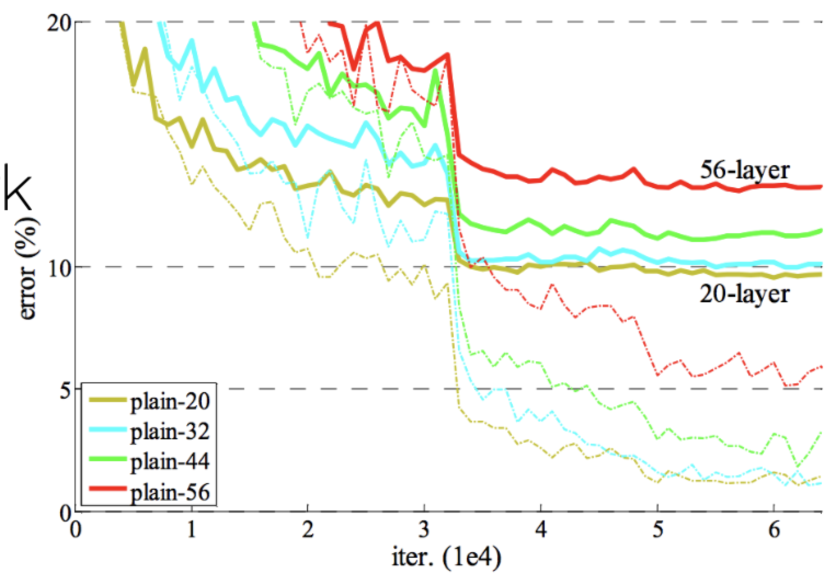
I2DL: Prof. Niessner, Dr. Dai

61

ResNet

ResNet

- If we make the network deeper, at some point performance starts to degrade
- Too many parameters, the optimizer cannot properly train the network



I2DL: Prof. Niessner, Dr. Dai 60

FCN

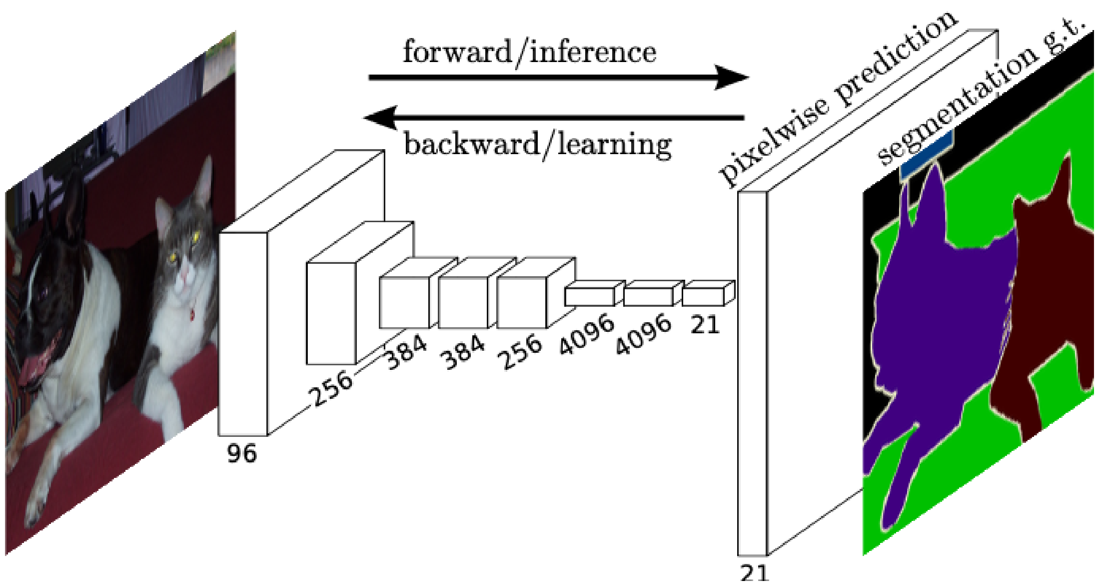


Figure 1. Fully convolutional networks can efficiently learn to make dense predictions for per-pixel tasks like semantic segmentation.

Figure: https://arxiv.org/pdf/1411.4038.pdf