Temporally Consistent Depth Estimation in Videos with Recurrent Architectures

Denis Tananaev[1,2], Huizhong Zhou[1(✉)], Benjamin Ummenhofer[1], and Thomas Brox[1]

[1] University of Freiburg, Freiburg im Breisgau, Germany
{zhouh,ummenhof,brox}@cs.uni-freiburg.de
[2] Robert Bosch GmbH, Stuttgart, Germany
Denis.Tananaev@de.bosch.com

Abstract. Convolutional networks trained on large RGB-D datasets have enabled depth estimation from a single image. Many works on automotive applications rely on such approaches. However, all existing methods work on a frame-by-frame manner when applied to videos, which leads to inconsistent depth estimates over time. In this paper, we introduce for the first time an approach that yields temporally consistent depth estimates over multiple frames of a video. This is done by a dedicated architecture based on convolutional LSTM units and layer normalization. Our approach achieves superior performance on several error metrics when compared to independent frame processing. This also shows in an improved quality of the reconstructed multi-view point clouds.

Keywords: Convolutional LSTM · Recurrent networks · Depth estimation · Video processing

1 Introduction

Triggered by the seminal work of Eigen et al. [6] the estimation of depth maps from just a single image has become a popular tool in computer vision. Depth estimation in a single image is well known to be highly ambiguous and only works due to strong conditional priors learned from previously seen data. The work from Eigen et al. [6] demonstrated the superiority of deep networks over previous attempts with hand-crafted features [23]. The priors learned by a deep network yield an unprecedented accuracy and generality compared to all previous approaches on single-view depth estimation. Of course, the accuracy is far from being competitive with multi-view reconstruction, which becomes evident when visualizing the depth maps in the form of a point cloud. Nevertheless, estimating the depth from single images is no longer a toy problem but is used in places, where dense multi-view reconstruction is not directly applicable, for instance to initialize a monocular SLAM method [26] or for rough but dense depth estimates in autonomous driving [28].

L. Leal-Taixé and S. Roth (Eds.): ECCV 2018 Workshops, LNCS 11131, pp. 689–701, 2019.
https://doi.org/10.1007/978-3-030-11015-4_52

Often in these applications, an image sequence rather than a single image is available, yet these additional images are typically not exploited. There are two sources of information that are inherent to a sequence of images: (1) the motion parallax by a moving camera; (2) the temporal consistency of successive frames. Exploiting the motion parallax has been approached in Ummenhofer et al. [29] for two frames. The motion parallax was used also in Zhou et al. [32] for unsupervised learning of depth from single image. However, there is no approach yet, that exploits the temporal consistency of successive frames.

In this work, we propose a network architecture based on convolutional LSTMs to capture temporal information from previous frames and to enforce temporally consistent depth estimates in a video. We show that such a network improves over independent frame processing, both relative to a single-frame baseline and relative to the state of the art. While it is well-known that temporal consistency has only relatively small effects on standard performance metrics based on average statistics, the qualitative improvement is much higher due to more stable estimates that do not flicker; see the supplemental video. The temporal consistency is also advantageous when combining multiple depth maps to a joint point cloud. As the depth estimates of successive frames agree more, the resulting point cloud is more consistent, too.

2 Related Work

End-to-end depth estimation from a single image with convolutional neural networks was introduced by Eigen [5,6]. These works were introduced before the today most common convolutional encoder-decoder architectures, such as FCN [19] and U-Net [22] were available. They use a multi-scale architecture for depth estimation at different spatial resolutions. Joint depth and normal prediction improved the depth estimation results.

Liu et al. [18] combined convolutional networks with superpixel-based conditional random fields. Chackrabarti et al. [3] generate a midlevel representation of the depth with a deep network to find the best matches for the depth values in a post processing step. At the present, Laina et al. [16] yields the best results on benchmarks. This was achieved by replacing the standard convolutional encoder by a ResNet-50 architecture [11] and by a set of computationally efficient up-sampling blocks.

The methods above process each frame independently, even when processing a video. We propose the use of LSTM units [9,12] to relate the intermediate representation in the network across frames in order to obtain temporally consistent outputs. In particular, we use a convolutional LSTM architecture [31]. The convolutional version of LSTMs captures the spatial context of the input tensor and keeps the number of parameters limited.

3 Network Architecture

The architecture in Fig. 1 integrates a typical convolutional encoder-decoder structure with convolutional LSTM layers (in brown) to analyze the image at

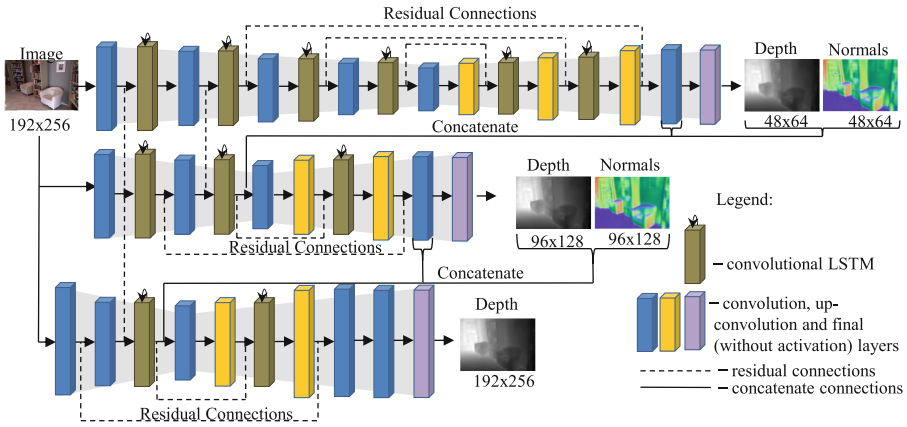

Fig. 1. Overview of the multi-resolution recurrent network architecture. The architecture consists of three encoder-decoder networks that predict the depth at different resolution levels. The low-resolution prediction is fed as input to the next, higher-resolution stream. The first two streams also estimate the normals in addition to the depth. The convolutional LSTM layers carry the state from the previous frame. There are residual connections within each encoder-decoder stream and also between the streams.

various levels of abstraction, where for each level, the state representation from the previous frame is carried over to the present frame. This combination with the previous state enforces the temporal consistency of the states and, consequently, also of the output. Like all common encoder-decoder networks, the architecture has residual connections to directly propagate high-resolution features from the encoder to the respective layer in the decoder. The spatial resolution of all convolutional filters is 3×3. The filters for the up-convolution have size 4×4 and for the LSTM filters we use 5×5 filters.

In addition to the multi-scale analysis due to the encoder-decoder architecture, the architecture includes a coarse-to-fine refinement strategy, which first produces the output depth map at a lower resolution (with a loss applied during training). The low-resolution result is successively refined by the next encoder-decoder stream of the multi-resolution architecture until the resolution of the input image is obtained at the output. This coarse-to-fine strategy efficiently implements the network stacking idea, which has been successful for optical flow estimation [13] and depth from two views [29]. We also added recurrent connections between the layers of the different streams. For the intermediate resolutions, the network also computes the surface normals (with a loss applied during training), which is helpful to learn the depth representation for the surfaces in the scene.

3.1 Convolutional LSTM with Leaky ReLU

Recurrent neural network architectures have shown to be able to leverage temporal data for tasks such as language processing [4] and video captioning [2].

We build upon the Long Short Term Memory (LSTM) unit [12] to enable temporally consistent video depth prediction.

In convolutional LSTMs the input tensor h_t^{l-1} is concatenated with the hidden state tensor h_{t-1}^l before the convolution operation is applied. The leaky ReLU has shown improved performance compared to the tanh activation function in many previous works. Thus, we use it also for the present work. However, our experiments showed that the convolutional LSTM with leaky ReLU is less stable than the LSTM with tanh activation and numerically explodes when processing longer sequences during testing; see Fig. 2. The problem of stability is solved by adding a layer normalization [1] on the cell state c_t^l. Moreover, the normalization layer also allows for faster convergence.

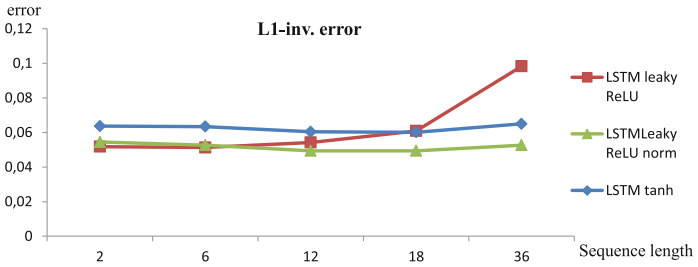

Fig. 2. Network stability at test time with different activation functions used in the LSTM unit. The leaky ReLU activation yields better results than the tanh activation. However, the network becomes unstable over time. Adding layer normalization yields the good performance of the leaky ReLU while being as stable as the tanh activation.

Formally, the convolutional LSTM with leaky ReLU and layer normalization reads:

$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \text{leakyReLU} \end{pmatrix} \circ W^l * \begin{pmatrix} h_t^{l-1} \\ h_{t-1}^l \end{pmatrix}, \tag{1}$$

$$c_t^l = f \cdot c_{t-1}^l + i \cdot g, \tag{2}$$

$$\hat{c}_t^l = \gamma \left(\frac{c_t^l - \mu(c_t^l)}{\sqrt{\text{var}(c_t^l)}} \right) + \beta, \tag{3}$$

$$h_t^l = o \cdot \text{leakyReLU}(\hat{c}_t^l), \tag{4}$$

where i, f, o, g are the input, forget, output gates and new input, respectively; c_t^l and c_{t-1}^l are the cell states for the current and previous time steps; γ and β are the learned parameters of the layer normalization [1], and μ and var are the mean and variance of the argument over each single data sample.

4 Training Procedure

We train the recurrent network with a batch size of 2 and a sequence length of 7 by unrolling the network. For each frame the network generates a depth map; we apply a loss on each of the outputs.

4.1 Datasets

We train and evaluate our recurrent network on static and dynamic indoor sequences. For the static sequences we use the NYUv2 [24] and SUN3D [30] datasets. Both datasets feature indoor video sequences of offices, living rooms, etc. filmed with a structured light sensor. We use the raw depth from the sensor for NYUv2 and the TSDF fused depth provided by SUN3D. The NYUv2 dataset has 249 training video sequences and 215 test sequences. We use the SUN3D dataset split proposed by Ummenhofer et al. [29] which has 253 sequences for training and 16 for testing.

For dynamic scene experiments we use the Princeton Tracking Benchmark [25]. The dataset consists of indoor scenes with dynamic objects captured with the Kinect sensor. We use 96 sequences for training and four sequences for test.

4.2 Initialization and Training Strategy

We initialize the network weights using Xavier initialization [8] with modifications proposed by [10] for ReLU functions. We normalize the input image values to the range $[-0.5, 0.5]$ and use inverse depth values $\xi = 1/z$ for parameterizing the depth values. Inverse depth emphasizes distances to close objects, yielding more precise predictions for those objects and allows us to represent points at infinity.

For training we use nearest neighbour sampling to resize the ground truth depth maps to 256×192. On NYUv2 we first crop images to 561×427 before downsampling. The output depth has the same resolution as input.

We use ADAM [15] with restarts. The restart technique was proposed for SGD optimization and allows to achieve faster convergence of the network compared to the fixed learning rate schedules [20]. The starting learning rate for each restart is 10^{-4} and it drops to 10^{-6} at the end of each period. The first restart interval is 10000 iterations and it increases by factor 1.5 at each restart.

To avoid overfitting to very long sequences in the training set, we iterate in random order over the set of sequences and sample from each sequence a random segment with 7 frames. This allows the network to see an equal number of training samples from each sequence. Further, we augment the segments by randomly skipping frames with a probability of 0.5.

4.3 Loss Functions

On the depth output, we combine two loss functions. First, we use L1 loss for the inverse depth

$$L_{\text{depth}} = \sum_{i,j} |\xi(i,j) - \hat{\xi}(i,j)|, \tag{5}$$

where $\hat{\xi}(i,j)$ is the ground truth inverse depth. Second, we compute the scale-invariant gradient loss [29] for the inverse depth

$$L_{\text{grad}} = \sum_{h \in 1,2,4,8,16} \sum_{i,j} ||g_h[\xi](i,j) - g_h[\hat{\xi}](i,j)||_2 \tag{6}$$

where $g_h[\xi](i,j)$ is the discrete scale invariant gradient:

$$g_h[f](i,j) = \left(\frac{f(i+h,j)-f(i,j)}{|f(i+h,j)|+|f(i,j)|}, \frac{f(i,j+h)-f(i,j)}{|f(i,j+h)|+|f(i,j)|} \right). \tag{7}$$

In (6) we sum the gradient for five different discretization widths h to cover gradients with different slopes. The gradient loss significantly improves the smoothness of the depth values while preserving sharp depth edges.

The loss on the normals is the non-squared L2 norm

$$L_{\text{normal}} = \sum\nolimits_{i,j} ||n(i,j) - \hat{n}(i,j)||_2, \tag{8}$$

where $n(i,j)$ is the normal predicted by the network and $\hat{n}(i,j)$ is the ground truth normal, which we derive from the ground truth depth maps.

To balance the importance of the loss functions we use different weights. We assign the weight 300 for the $L1$ depth loss and 1500 for the scale invariant gradient loss and 100 for the loss on the normals. The weights were set empirically.

For the first 10000 iterations we set the weight for the gradient loss to zero, because the scale invariance of L_{grad} can cause instabilities directly after weight initialization. During training and evaluation, we do not consider pixels with invalid depth values.

4.4 Error Metrics

To quantify the quality of the predicted depth maps we compute several common error metrics:

$L1$ inverse error: $L1 - inv(z,\hat{z}) = \frac{1}{N}\sum_i \left| \frac{1}{z_i} - \frac{1}{\hat{z}_i} \right|$,

The mean root squared error (RMS): $RMS(z,\hat{z}) = \sqrt{\frac{1}{N}\sum_i (z_i - \hat{z}_i)^2}$

Average log_{10} error (log10): $log10(z,\hat{z}) = \frac{1}{N}\sum_i |\log z_i - \log \hat{z}_i|$

Percentage of pixels below a ratio treshold θ: $\max\left(\frac{\hat{z}_i}{z_i}, \frac{z_i}{\hat{z}_i}\right) = \delta(z,\hat{z}) < \theta$.

Here z_i is the depth prediction, $\hat{z}_i$ is the ground truth depth, and N is the number of valid pixels in a depth map.

5 Experiments

5.1 Choice of the Activation Function

To quantify the behaviour of the LSTM with different activation functions and normalization strategies we first ran experiments with a simplified network architecture, which consists of only one encoder with 10 layers and one decoder with 13 layers. We also compared the recurrent network with the LSTM layers removed, i.e., that network does not take information from previous frames into account. Results on SUN3D dataset for a sequence length of 6 are shown in Table 1. The recurrent architecture improves over the single-frame baseline on all metrics, and the leaky ReLU activation unit always outperforms tanh activation. Thus, for the following experiments, we always used leaky ReLU activation with normalization.

Table 1. Comparison of the recurrent architecture to its non-recurrent baseline for two different activation functions in the LSTM unit on the SUN3D dataset. The recurrent architectures improve over the single-frame baseline on all metrics. The leaky ReLU activation unit always outperforms tanh activation.

Metrics	L1-inv	log10	RMS	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Single frame	0.0763	0.170	0.577	73.1%	90.8%	98.2%
LSTM tanh	0.0637	0.156	0.474	80.0%	92.8%	99.1%
LSTM leaky ReLU+norm	**0.0518**	**0.140**	**0.398**	**80.1%**	**94.3%**	**99.6%**

5.2 Comparison to the State of the Art

In Table 2, we compare the full network architecture to the state of the art in depth estimation from single image. We trained the network for the first six restarts on the SUN3D dataset and then fine-tuned it for the last restart on the NYUv2 training data. For the evaluation, we randomly sampled 50 sequences from the NYUv2 test set and evaluated on the first 50 frames of each sequence. We evaluated only in the regions with valid depth values.

The use of temporal consistency with LSTM yields state-of-the-art results in several metrics. Moreover, the version with temporal consistency outperforms the baseline without LSTM units on all metrics. This clearly shows the benefit of taking previous frames into account.

A qualitative comparison is shown in Fig. 3. Our results have sharper boundaries than the single-frame methods, and there is no flickering in the depth maps estimated over time, as can be seen in the supplemental video.

Table 2. Comparison to the state of the art on 50 sequences from the NYUv2 dataset with a length of 50 frames each. The use of temporal consistency with LSTM yields state-of-the-art results on most metrics. The runtime performance of the methods (frames per second) was estimated on the NVidia Geforce Titan X (Maxwell architecture).

Metrics	L1-inv	log10	RMS	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$	fps
Liu [18]	0.155	0.133	0.995	56.1%	81.1%	88.7%	0.07
Eigen [5]	0.130	0.108	0.885	69.0%	86.9%	91.6%	0.33
Chakrabarti [3]	0.116	0.095	**0.808**	77.1%	88.3%	91.2%	0.04
Laina [16]	0.114	0.093	0.823	77.9%	88.4%	**91.7**%	8.25
Our single-frame	0.119	0.101	0.878	75.4%	87.7%	91.2%	**8.33**
Our LSTM	**0.111**	**0.092**	0.824	**79.6**%	**88.9**%	91.4%	4.54

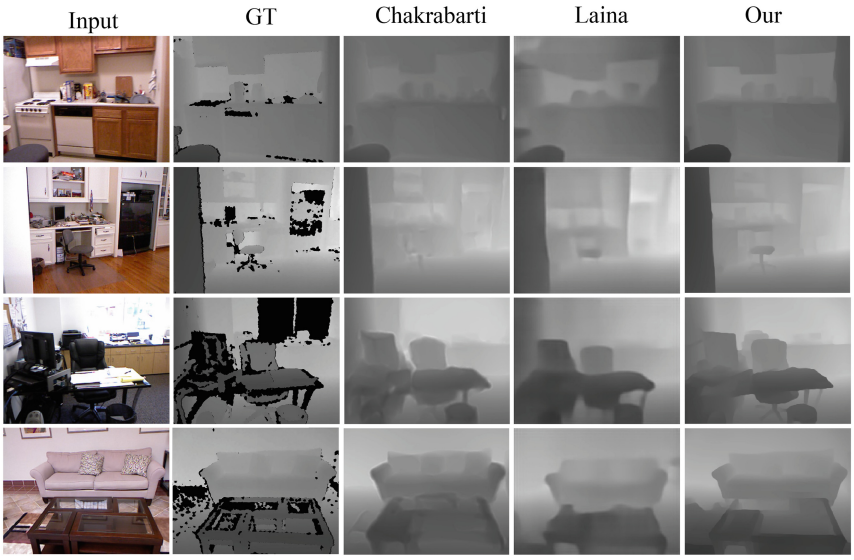

Fig. 3. Qualitative comparison of our LSTM based network to the independent frame processing of Chakrabarti et al. [3] and Laina et al. [16]. The result of the last image in each 50-frame sequence is shown. The proposed architecture with multi-frame processing yields sharper edges and captures more details.

5.3 Temporal Consistency

We show that our LSTM network learns to predict temporally consistent depth maps. We validate this by comparing depth predictions of our LSTM-based architecture with state of the art single-frame depth estimation networks. In Fig. 4 we show the depth trajectory of a point on a 50 frames sequence from the NYUv2 dataset. And in Fig. 5 we further show the temporal consistency comparison with the average depth change over all pixels.

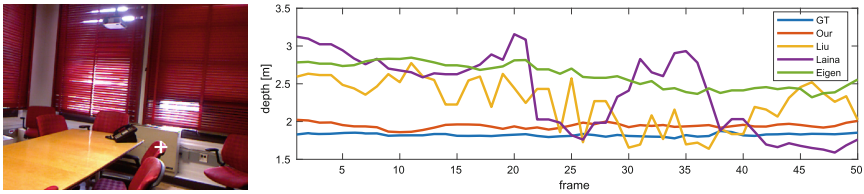

Fig. 4. We track the depth of a single point over time. We use the KLT tracker [21,27] to track the point in the image sequence and plot the depth over time. Our LSTM-based architecture is not only more accurate but also more temporally consistent and therefore suited for processing video streams.

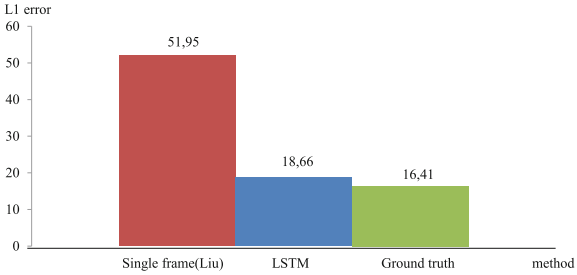

Fig. 5. Comparison of the temporal consistency with single frame Liu et al. [17], our LSTM network, and ground truth. The bars represent the average depth change of corresponding points between consecutive depth frames of a sequence of size 10. In order to compute point correspondences we use Farneback optical flow [7]. Since optical flow estimation introduces additional errors we also show the ground truth results.

Temporal consistency is clearly advantageous in static images. In case of dynamic scenes, temporal consistency, which effectively induces smoothing over time, could have negative effects. It is worth noting, though, that the proposed approach does not smooth the resulting depth map, but the intermediate state representation, i.e., the network can learn to smooth along the point trajectories and consider motion boundaries and occlusion areas. To verify the performance of the recurrent architecture in dynamic scenes we compare it to the single-frame baseline on the Princeton tracking benchmark, which comprises dynamic scenes. We evaluated on four sequences with 50 frames each.

The results in Table 3 show that there is still a small advantage for the LSTM-based architecture, yet it is smaller than in static cases. This indicates that the network cannot learn all of the effects mentioned above, but at least it alleviates most of the negative effects.

A qualitative result is shown in Fig. 6. The single-frame baseline has problems with the shape of the moving object, while the recurrent network can exploit the additional information from previous frames. The effect is strongest when the object gets occluded and is partially not visible in the single frame.

Table 3. Depth prediction on the dynamic scenes of the Princeton tracking benchmark. The results do not suffer from temporal consistency despite the motion of objects. In the contrary, the results with temporal consistency are even a little better.

Metrics	L1-inv	log10	RMS	$\delta < 1.25$	$\delta < 1.25^2$	$\delta < 1.25^3$
Our single-frame	**0.150**	**0.126**	1.698	79.9%	81.8%	82.9%
Our LSTM	**0.150**	**0.126**	**1.672**	**80.0%**	**82.0%**	**83.2%**

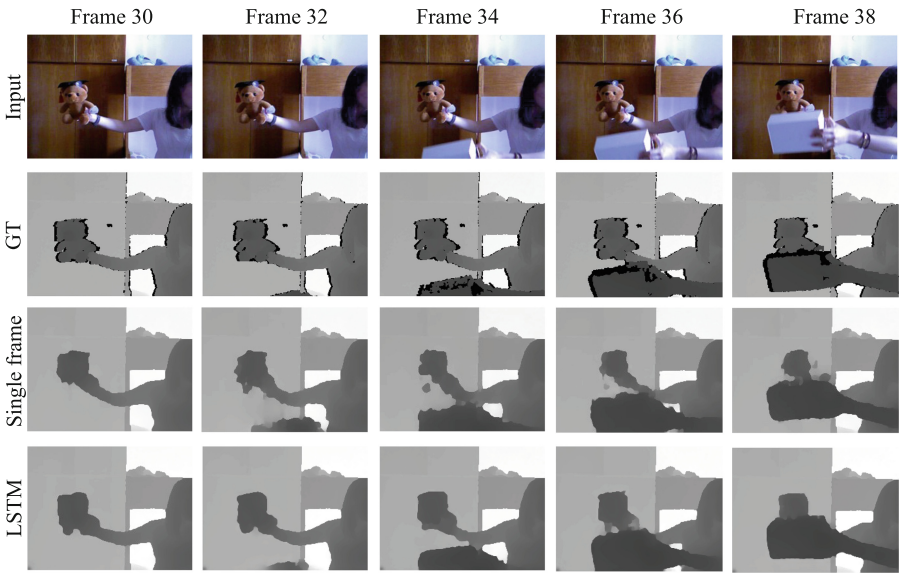

Fig. 6. Dynamic scene from the Princeton tracking benchmark. The temporally consistency due to the LSTM helps to reconstruct the precise depth near boundaries of a moving object.

5.4 3D Reconstruction

We compared the quality of the predicted depth maps also in a full scene reconstruction context, where the depth maps were used as depth channel in an RGB-D SLAM approach [30] to reconstruct a 3D scene from a video sequence. Figure 7 shows the 3D reconstructions.

The temporally consistent depth maps help improve the reconstructed 3D scene, since the variation of the same surface points over time is much reduced. Thus, the point cloud is less noisy which leads to better 3D reconstruction. Also there are less severe misalignments in the scan, since the temporally consistent depth maps are easier to register for the SLAM method.

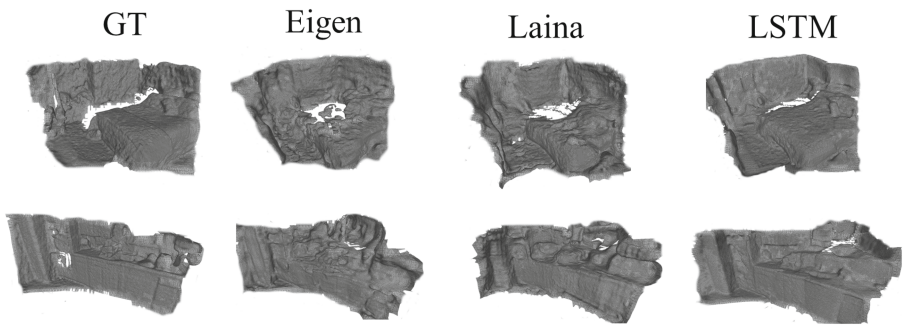

Fig. 7. The result of the RGB-D structure from motion [30] with depth from the neural network of Eigen and Fergus [5], Laina [16] and our recurrent network. For each reconstruction we use a sequence of 25 frames. We use Poisson surface reconstruction to generate the meshes [14]. Inconsistent depth estimates for Eigen and Laina lead to reconstruction artifacts in the surface mesh. The reconstructions from our depth predictions show less artifacts and have more details.

6 Conclusions

In this work, we have introduced the first depth estimation network that optimizes the temporal consistency of the estimated depth map over multiple frames in a video. We have shown that the LSTM with leaky ReLU yields better results than the traditional convolutional LSTM with tanh activation. In this context, we have also shown the importance of layer normalization for the stability of the recurrent network. The experimental results with the proposed multi-frame processing consistently outperformed those with frame-independent processing both in static and dynamic scenes.

Acknowledgements. This project was partially funded by the EU Horizon 2020 project Trimbot2020. We also thank Facebook for their P100 server donation and gift funding.

References

1. Ba, J.L., Kiros, J.R., Hinton, G.E.: Layer normalization. arXiv preprint arXiv:1607.06450 (2016)
2. Ballas, N., Yao, L., Pal, C., Courville, A.: Delving deeper into convolutional networks for learning video representations. In: International Conference on Learning Representations (ICLR 2016) (2016)
3. Chakrabarti, A., Shao, J., Shakhnarovich, G.: Depth from a single image by harmonizing overcomplete local network predictions, pp. 2658–2666 (2016)
4. Cho, K., et al.: Learning phrase representations using RNN encoder-decoder for statistical machine translation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 1724–1734. Association for Computational Linguistics, Doha, Qatar, October 2014. http://www.aclweb.org/anthology/D14-1179

5. Eigen, D., Fergus, R.: Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2650–2658 (2015)
6. Eigen, D., Puhrsch, C., Fergus, R.: Depth map prediction from a single image using a multi-scale deep network. In: Advances in Neural Information Processing Systems, pp. 2366–2374 (2014)
7. Farnebäck, G.: Two-frame motion estimation based on polynomial expansion. In: Bigun, J., Gustavsson, T. (eds.) SCIA 2003. LNCS, vol. 2749, pp. 363–370. Springer, Heidelberg (2003). https://doi.org/10.1007/3-540-45103-X_50
8. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: Aistats, vol. 9, pp. 249–256 (2010)
9. Greff, K., Srivastava, R.K., Koutník, J., Steunebrink, B.R., Schmidhuber, J.: LSTM: a search space odyssey. IEEE Trans. Neural Networks Learn. Syst. **28**(10), 2222–2232 (2017)
10. He, K., Zhang, X., Ren, S., Sun, J.: Delving deep into rectifiers: surpassing human-level performance on imagenet classification. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1026–1034 (2015)
11. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
12. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
13. Ilg, E., Mayer, N., Saikia, T., Keuper, M., Dosovitskiy, A., Brox, T.: Flownet 2.0: evolution of optical flow estimation with deep networks. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), July 2017. http://lmb.informatik.uni-freiburg.de//Publications/2017/IMKDB17
14. Kazhdan, M., Bolitho, M., Hoppe, H.: Poisson surface reconstruction. In: Proceedings of the Fourth Eurographics Symposium on Geometry Processing, SGP 2006, pp. 61–70. Eurographics Association, Aire-la-Ville, Switzerland (2006)
15. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. In: Proceedings of the 3rd International Conference on Learning Representations (ICLR 2015) (2015)
16. Laina, I., Rupprecht, C., Belagiannis, V., Tombari, F., Navab, N.: Deeper depth prediction with fully convolutional residual networks. In: 2016 Fourth International Conference on 3D Vision (3DV), pp. 239–248, October 2016. https://doi.org/10.1109/3DV.2016.32
17. Liu, F., Shen, C., Lin, G.: Deep convolutional neural fields for depth estimation from a single image. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2015). http://arxiv.org/abs/1411.6387
18. Liu, M., Salzmann, M., He, X.: Structured depth prediction in challenging monocular video sequences. arXiv preprint arXiv:1511.06070 (2015)
19. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3431–3440 (2015)
20. Loshchilov, I., Hutter, F.: SGDR: stochastic gradient descent with restarts. In: 5th International Conference on Learning Representations (ICLR 2017) (2017)
21. Lucas, B.D., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: Proceedings of the 7th International Joint Conference on Artificial Intelligence, IJCAI 1981, vol. 2, pp. 674–679. Morgan Kaufmann Publishers Inc., San Francisco (1981)

22. Ronneberger, O., Fischer, P., Brox, T.: U-Net: convolutional networks for biomedical image segmentation. In: Navab, N., Hornegger, J., Wells, W.M., Frangi, A.F. (eds.) MICCAI 2015. LNCS, vol. 9351, pp. 234–241. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24574-4_28
23. Saxena, A., Sun, M., Ng, A.Y.: Make3D: learning 3D scene structure from a single still image. IEEE Trans. Pattern Anal. Mach. Intell. **31**(5), 824–840 (2009)
24. Silberman, N., Hoiem, D., Kohli, P., Fergus, R.: Indoor segmentation and support inference from RGBD images. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012. LNCS, vol. 7576, pp. 746–760. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-33715-4_54
25. Song, S., Xiao, J.: Tracking revisited using RGBD camera: unified benchmark and baselines. In: 2013 IEEE International Conference on Computer Vision, pp. 233–240, December 2013. https://doi.org/10.1109/ICCV.2013.36
26. Tateno, K., Tombari, F., Laina, I., Navab, N.: CNN-SLAM: real-time dense monocular SLAM with learned depth prediction. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 6565–6574, July 2017. https://doi.org/10.1109/CVPR.2017.695
27. Tomasi, C., Kanade, T.: Detection and tracking of point features. Int. J. Comput. Vis. **9**(3), 137–154 (1991). Technical report
28. Uhrig, J., Cordts, M., Franke, U., Brox, T.: Pixel-level encoding and depth layering for instance-level semantic labeling. In: Rosenhahn, B., Andres, B. (eds.) GCPR 2016. LNCS, vol. 9796, pp. 14–25. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-45886-1_2
29. Ummenhofer, B., et al.: DeMoN: depth and motion network for learning monocular stereo. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
30. Xiao, J., Owens, A., Torralba, A.: SUN3D: a database of big spaces reconstructed using SfM and object labels. In: 2013 IEEE International Conference on Computer Vision (ICCV), pp. 1625–1632. IEEE, December 2013. https://doi.org/10.1109/ICCV.2013.458
31. Xingjian, S., Chen, Z., Wang, H., Yeung, D.Y., Wong, W.K., Woo, W.-c.: Convolutional LSTM network: a machine learning approach for precipitation nowcasting. In: Advances in Neural Information Processing Systems, pp. 802–810 (2015)
32. Zhou, T., Brown, M., Snavely, N., Lowe, D.G.: Unsupervised learning of depth and ego-motion from video. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)