OcclusionFusion: Occlusion-aware Motion Estimation for Real-time Dynamic 3D Reconstruction - Supplementary Material -

Wenbin Lin Chengwei Zheng Jun-Hai Yong* Feng Xu*
School of Software and BNRist, Tsinghua University
lwb20@mails.tisnghua.edu.cn, {zhengcw18, xufeng2003}@gmail.com, yongjh@tsinghua.edu.cn

A. Network Details

We first show the details in the network architecture, including the LSTM-based temporal motion encoding and the graph neural network. Then, we show the details in the network training. The neural network is implemented using the PyTorch [2] and PyTorch Geometric [1] libraries.

A.1. Architecture Details

LSTM-based Temporal Motion Encoding. The input and output motion of the LSTM module are represented using Gaussian distribution with diagonal covariance $\mathcal{N}(\boldsymbol{\mu}, \sigma^2 \boldsymbol{I})$, where $\boldsymbol{\mu} \in \mathbb{R}^3, \sigma \in \mathbb{R}$. The LSTM module is a standard two-layer LSTM with a hidden feature dimension of 32. Then we use a fully connected layer to predict the $\boldsymbol{\mu}'$ and σ' based on the output feature of the LSTM.

Graph Neural Network. We show the architecture of the graph neural network in Fig. 1 and the architectures of the graph pyramid convolution and the GConv block in Fig. 2 and Fig. 3. The graph transformer is proposed by Shi et al. [3]. For all the dropout blocks, the drop probability is 0.1. N_i denotes the number of nodes in the ith level of the graph pyramid. The dimensionality of the input node feature is 11. It contains three dimensions of node position, three dimensions of node motion of the current frame, one dimension of the visibility, and four dimensions of the output $\boldsymbol{\mu}'$ and σ' from the LSTM module. Then the output motion vectors ($\boldsymbol{\mu}$ and σ) of the graph neural network will be used as the input historical motion vectors for future frames.

*This work was supported by Beijing Natural Science Foundation (JQ19015), the NSFC (No.61727808, 62021002), the National Key R&D Program of China (2018YFA0704000, 2019YFB1405703) and TC190A4DA/3. This work was supported by THUIBCS, Tsinghua University and BLBCI, Beijing Municipal Education Commission. Jun-Hai Yong and Feng Xu are corresponding authors.

A.2. Training Details

Loss Function. The log-likelihood loss based on per node Gaussian distribution can be transformed as follow:

$$\begin{aligned}\mathcal{L} &= -\frac{1}{N}\sum_{i=1}^{N}\log\left(\mathcal{N}\left(\boldsymbol{y}_i \mid \boldsymbol{\mu}_i, \sigma_i^2\boldsymbol{I}\right)\right) \\ &= \frac{1}{2N}\sum_{i=1}^{N}\left(\log 2\pi + \log\sigma_i + \frac{\|\boldsymbol{y}_i-\boldsymbol{\mu}_i\|_2^2}{\sigma_i^2}\right) \\ &= C_1\sum_{i=1}^{N}\left(\log\sigma_i + \frac{\|\boldsymbol{y}_i-\boldsymbol{\mu}_i\|_2^2}{\sigma_i^2}\right) + C_2,\end{aligned} \tag{1}$$

where C_1, C_2 are constants. Therefore the loss function can be simplified as:

$$\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N}\left(\log\sigma_i + \frac{\|\boldsymbol{y}_i-\boldsymbol{\mu}_i\|_2^2}{\sigma_i^2}\right). \tag{2}$$

Thus, if the σ_i is a fixed value, the loss is equivalent to the mean-squared error (MSE) loss.

Training Procedure. The whole network is trained end-to-end. The LSTM module takes the historically predicted motion as input, while the predicted motion is not accurate enough at the beginning. So we use the ground truth historical motion with $\sigma = 0$ to warm up the network training. We first train the network using ground truth historical motion for 200 epochs. Then, we switch the input historical motion to the network's output and train for another 1200 epochs. Besides, the input historical motion vectors are treated as undifferentiable constants and detached from the computation graph. Thus the gradients of the LSTM module do not flow back to the graph neural network. We train the network using the Adam optimizer with a learning rate of 0.001 and a batch size of 64. The whole training process takes about four days using an NVIDIA RTX GeForce 2080Ti GPU.

To quantitatively evaluate the generalization ability of our network, we train the model on the humanoids subset

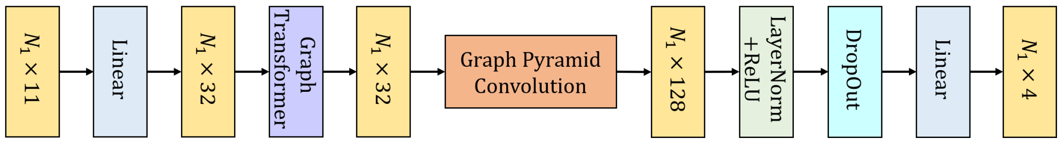

Figure 1. Architecture of the graph neural network.

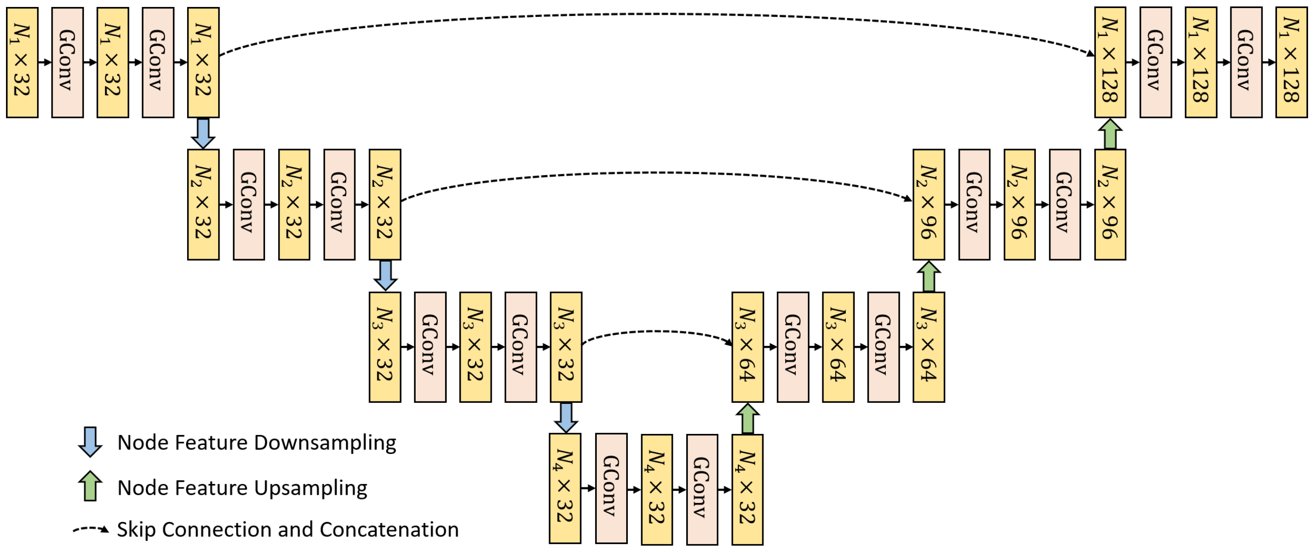

Figure 2. Architecture of the Graph Pyramid Convolution in Fig. 1.

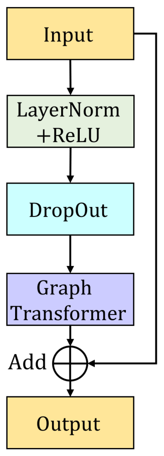

Figure 3. Architecture of the GConv block.

and test it on the animal subset for motion estimation in Sec. 4.2. For other experiments, we use both subsets for training to achieve better reconstruction results.

In addition, as the estimated 3D motion of the visible part in the real world is not as perfect as the synthetic dataset, we add random Gaussian noise to the visible motion during network training. The noise is represented as $\mathcal{N}(\boldsymbol{\mu}, \sigma^2 \boldsymbol{I})$, where $\boldsymbol{\mu} = [0, 0, 0]$ and σ is sampled from a uniform distribution $\mathcal{U}(-0.4, 0.4)$. However, in the motion prediction evaluation in Sec. 4.2 we do not add any noise.

B. Details in Multi-scale Graph Pyramid Construction

We construct a 4-level multi-scale graph for message passing among nodes. Considering that the node connectivity of the first level is computed based on the Euclidean distance, which may lead to misconnections between unrelated parts, we discard edges by temporal consistency. More precisely, if the distance between two nodes changes more than a threshold, the edge between them will be discarded.

The node features downsampling between adjacent levels is performed directly by copying the node features to the higher level, as the nodes in the $(l+1)$th level are a subset of the nodes in the lth level. In feature upsampling, the node features of the $(l+1)$th level are assigned using the feature of the nearest neighbor in the lth level.

The intervals between nodes from the first to the fourth level are set to $\{4\text{cm}, 8\text{cm}, 16\text{cm}, 32\text{cm}\}$ and the neighbor amounts are $\{8, 6, 4, 3\}$. Besides, the distance change threshold is set to 4cm.

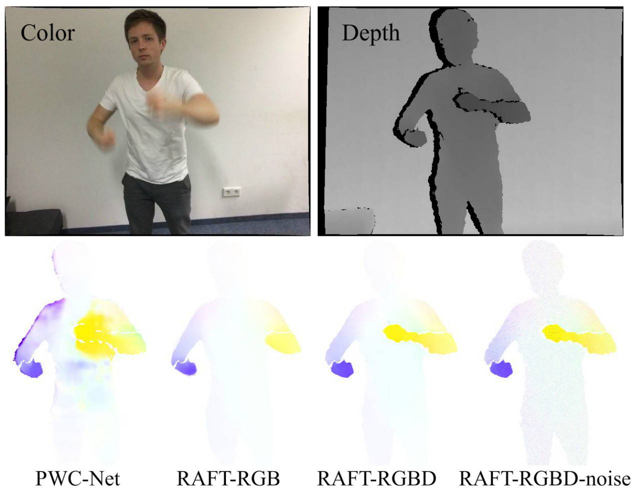

Figure 4. Results of different optical flow methods when motion blur occurs.

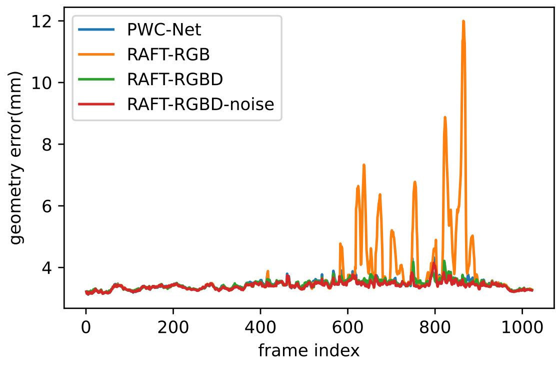

Figure 5. Geometry errors on the top sequence of Fig. 6 in the main paper. The average geometry errors over the whole sequence are 3.44mm, 3.79mm, 3.45mm, and 3.41mm from PWC-Net to RAFT-RGBD-noise.

C. RGB-D Based Optical Flow Prediction

We use the RAFT [5] network to estimate 2D optical flow. The original RAFT is trained with RGB images. We change the input dimensionality from 3 to 4 and retrain the network using RGB-D images as input. For the depth channel, we use the inverse depth (the reciprocal of depth) as input.

To speed up the 2D optical flow estimation in the 3D reconstruction system, we resize the input images from 640×480 to 320×240 to compute the optical flow and upsample the optical flow to the original resolution by bilinear interpolation.

D. Choice of Optical Flow

To test the robustness to flow estimation, we evaluate our system using different optical flow settings PWC-Net [4], RAFT-RGB and RAFT-RGBD. Besides, we further add Gaussian noise of $\mathcal{N}(0, 4)$ pixels on x and y axes to our RAFT-RGBD optical flow. We use the top sequence of Fig. 6 in the main paper for evaluation. The optical flow results at the 833rd frame of the sequence are shown in Fig. 4. We can see severe motion blur occurs on the fast swinging arm in the color image, and significant errors appear in the optical flow of RGB-based methods (PWC-Net and RAFT-RGB). However, since depth images do not suffer the blur artifacts much and provide geometric information, RAFT-RGBD generates reliable optical flow. This indicates the benefit of involving depth in flow estimation.

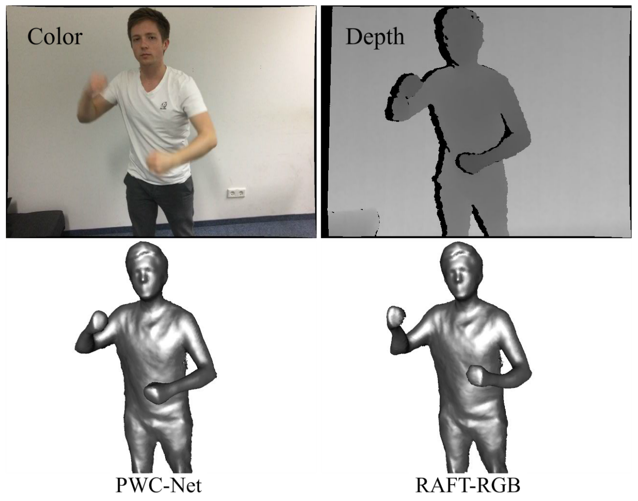

Figure 6. Reconstruction results based on PWC-Net and RAFT-RGB.

For quantitative evaluation, we show the geometry errors of different optical flow methods over the whole sequence in Fig. 5. We can see that the geometry errors of our reconstruction method based on PWC-Net, RAFT-RGBD, and RAFT-RGBD-noise are all low and close to each other, which indicates that our method is robust to noise in the optical flow. Only the reconstruction result based on the RAFT-RGB optical flow provides a large geometry error, which is caused by the tracking failure of the fast swinging arm. We show the reconstruction results using PWC-Net and RAFT-RGB at frame 837 (4 frames after the optical flow shown in Fig. 4) in Fig. 6. The reconstruction result of PWC-Net is better than RAFT-RGB, although they both provide a larger optical flow error. We believe the reason for this is that the optical flow errors of PWC-Net appear mainly on the torso, while those of RAFT-RGB appear on the anterior segment of the arm. In addition, errors on the torso are more easily corrected by the learned motion prior and the regularization term in the optimization, because there is more motion information around the torso region where the predicted optical flow is wrong.

References

[1] Matthias Fey and Jan E. Lenssen. Fast graph representation learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019. 1

[2] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Z. Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems(NeurIPS)*, pages 8024–8035, 2019. 1

[3] Yunsheng Shi, Zhengjie Huang, Shikun Feng, Hui Zhong, Wenjing Wang, and Yu Sun. Masked label prediction: Unified message passing model for semi-supervised classification. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence (IJCAI-21)*, pages 1548–1554, 2021. 1

[4] Deqing Sun, Xiaodong Yang, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *Proceedings of the IEEE Conference on computer Vision and Pattern Recognition (CVPR)*, pages 8934–8943, 2018. 3

[5] Zachary Teed and Jia Deng. RAFT: recurrent all-pairs field transforms for optical flow. In *European Conference on Computer Vision (ECCV)*, volume 12347, pages 402–419, 2020. 3