Monocular Piecewise Depth Estimation in Dynamic Scenes by Exploiting Superpixel Relations

Yan Di[1], Henrique Morimitsu[1], Shan Gao[2] and Xiangyang Ji[1]

[1]Department of Automation, Tsinghua University, Beijing, China
[2]Unmanned System Research Institute, Northwestern Polytechnical University, Xi'an, China

*{diy17@mails, hmorimitsu@mail}.tsinghua.edu.cn gaoshan@nwpu.edu.cn xyji@tsinghua.edu.cn

1. Details of Motion Selection

After superpixel relations analysis, we get homogrpahies H and spatial relations R_s. We decompose each homography model and generate $2n$ hypotheses of camera rotation R, translation t, plane norm n and inverse depth d, up to scale [6].

In this step, we follow [12] to identify camera motion $\{R_0, t_0\}$ and select static superpixels S_t. This task can be considered as a labelling problem on superpixel-level graph G_s, while [12] is developed on a pixel-level graph. The label set $\mathfrak{L} = \{\{R, t\}, l_\phi\}$ consists of motion hypotheses $\{R, t\}$ and a discard label l_ϕ. Since we mainly focus on mostly rigid scenes, for fast running speed, we only select 5 motion hypotheses for motion selection. Firstly, we transform the rotation matrices R into quaternions. Then concatenating rotation quaternions and the corresponding translations, we obtain a 7-dimensional feature vector for each superpixel. We intuitively select motion hypotheses by density in the 7-dimensional feature space. The feature with highest density is selected as the first motion hypothesis. Then removing feature vectors that are similar to the first selected feature, we select the feature with highest density in the remaining feature space as the second motion hypothesis. We repeat this process to select 5 motion hypotheses.

The energy function of motion selection is defined to fit a PEaRL framework,

$$E_{sel}(l) = \sum_{i \in S} D_i(l_i) + \lambda_s \sum_{(i,j) \in E} \omega_{ij} \delta(l_i \neq l_j) + \lambda_L |L_1|, \tag{1}$$

where l is the label field and λ_s, λ_L are weighting constants. The data term $D_i(l_i)$ is defined as follows,

$$D_i(l_i) = \frac{1}{|Z_i|} \sum_{p_k \in S_u(S_i)} \omega_c(S_i, p_k) \cdot min(d_k, \tau_s^2), \tag{2}$$

Figure 1. Results of the motion selection step. (a) demonstrates the selected static superpixels S_t, which are colored blue. (b) is the diagram of camera motion $\{R_0, t_0\}$.

where τ_s is a threshold. If the label l_i indicates a pure rotation, d_k is defined as the symmetric transfer error (STE) [6] and the Sampson distance [6] otherwise. We set $\omega_{ij} = 1$ in our experiment. Finally, the label term $|L_1|$ denotes the number of selected motion hypotheses. We introduce this term to force the algorithm to select as few motion hypotheses as possible. After this labeling process, we select the motion hypothesis that associates with most superpixels as the camera motion $\{R_0, t_0\}$ and the corresponding superpixels as static superpixels S_t. We demonstrate an example in Figure 1.

2. Details of propagation based optimization.

The fast propagation method is first proposed and successfully used in RicFlow [7]. We briefly summarize the method in 3 steps,

1. Random initialization. For each superpixel, the model (affine, homograpy or plane parameters) is initialized locally.

2. Hypothesis propagation. For each superpixel, its spatially neighboring superpixels that have been processed in this iteration propagate their models to the reference superpixel and help it to figure out the fittest

model.

3. Random test. Pick up several matches randomly to generate model hypotheses and then the current best model will be improved by testing the new hypotheses.

However, we cannot directly apply this fast propagation method to optimize the energy function E_{sra} , since E_{sra} has a pairwise term E_{pair}, while fast propagation only works with the data term. We slightly improve the fast propagation method to optimize E_{sra}. The **Random initialization** and **Random test** are the same. In hypothesis propagation, we process each superpixel in 2 steps,

1. Updating support neighbors. For superpixel S_i, the energy function that needs to be optimized in each iteration is defined as:

$$E_{iter} = \sum_{pl \in Su(S_i)} E_{data}(H_i) + \sum_{(i,j) \in E} E_{pair}(H_i), \quad (3)$$

where S_j has been processed in this iteration. We can update the support neighbors of S_i, then the pairwise term can be integrated into the data term. Specifically, if the relation between S_i and its neighbor S_j is hinge, then for each pixel p_k on the shared boundary, point match $(p_k, H_j P_k)$ is added to the set of support neighbors of S_i, denoted as $Su(S_i)$. If the relation is coplanar, for all pixels in S_j, the corresponding point matches under H_j are added to $Su(S_i)$. To balance locality and smoothness, we set the number of added matches from hinge and coplanar relations should be less than $2/5$ of initial support neighbors of S_i. Once exceeding the limit, we randomly select point matches to the number of the threshold.

2. Model propagation. With the new set of support neighbors, superpixel S_i receives models from its surrounding superpixels that have been processed in this iteration and figure out the best model with only the E_{data} term.

For the energy function E_{re} in reconstruction, we also use a block coordinate descent algorithm to optimize. We introduce the optimization process in 4 steps,

1. We first initialize the plane parameters θ and scale s of each superpixel in S_r. For superpixels in S_t, their scales are fixed to 1. We initialize their plane parameters locally with the E_{fit} term. For superpixels in $\{S_r \setminus S_t\}$, we initialize their plane parameters and scales by minimizing $E_{fit} + E_{rel} + E_{occ}$. After determining the parameters (θ, s) of superpixels in S_t, we propagate parameters from S_t to $\{S_r \setminus S_t\}$. Then we can gradually initialize the parameters (θ, s) of superpixels in $\{S_r \setminus S_t\}$ from near S_t to far from S_t.

2. After initialization, we optimize $E_{fit} + E_{rel} + E_{occ}$ with the improved fast propagation method. We propagate parameters (θ, s) among spatially neighboring superpixels in S_r.

3. Next, we determine the parameters of superpixels that are not in S_r by minimizing the $E_{fit} + E_{pri} + E_{occ}$ term. We just check the superpixels in $\{S \setminus S_r\}$ that are adjacent to superpixels in S_r. We change the relations from crack to hinge between the superpixel pairs that can minimize the energy term.

4. Finally, all superpixels are optimized together by minimizing $E_{fit} + E_{rel} + E_{occ} + E_{pri}$ with the improved fast propagation method.

The fast propagation based method is the basis of our fast reconstruction pipeline. Although it can not guarantee optimal results, it tends to output reasonable depth map.

3. Details of superpixel relations

We exploit superpixel relations to solve the inherent relative scale ambiguity (RSA) problem in dynamic reconstruction. We further explain our motivation in detail. As demonstrated in Figure 2, in dynamic scenes, foreground objects are usually supported by the surrounding environment. In the first row of Figure 2, the girl's right foot adheres to the ground while running. In the second row of Figure 2, the girl's right hand and right foot stick to the wall while climbing. In the third and fourth row of Figure 2, the girl's trunk is static and only her arms move freely. These examples show clearly that surrounding environment around moving objects usually indicates their spatial positions. DMDE [11] directly introduces motion segmentation and an ordering term which captures the assumption that moving objects occlude surrounding environment. However, it doesn't analyze the relations between foreground objects and the background and thus loses much information. Meanwhile, accurately segmenting moving objects is not easy.

We instead propose a unified method for dynamic reconstruction, requiring no motion segmentation. We analyze two kinds of superpixel relations between neighboring superpixels: motion relations R_e and spatial relations R_s. We assume $R_e = R_s$ in most cases. However, since crack relations provide no constraints, we can only determine the plane parameters of superpixels that are either in S_t or connects to S_t. As shown in the fifth row of Figure 2, the motion relations between superpixels on the car and superpixels on the surrounding road are crack, while the corresponding real spatial relations are hinge. We resolve this problem by introducing a complementary term E_{pri} in reconstruction, which aims to change sereval crack relations into hinge relations under the same assumption used in DMDE. Then

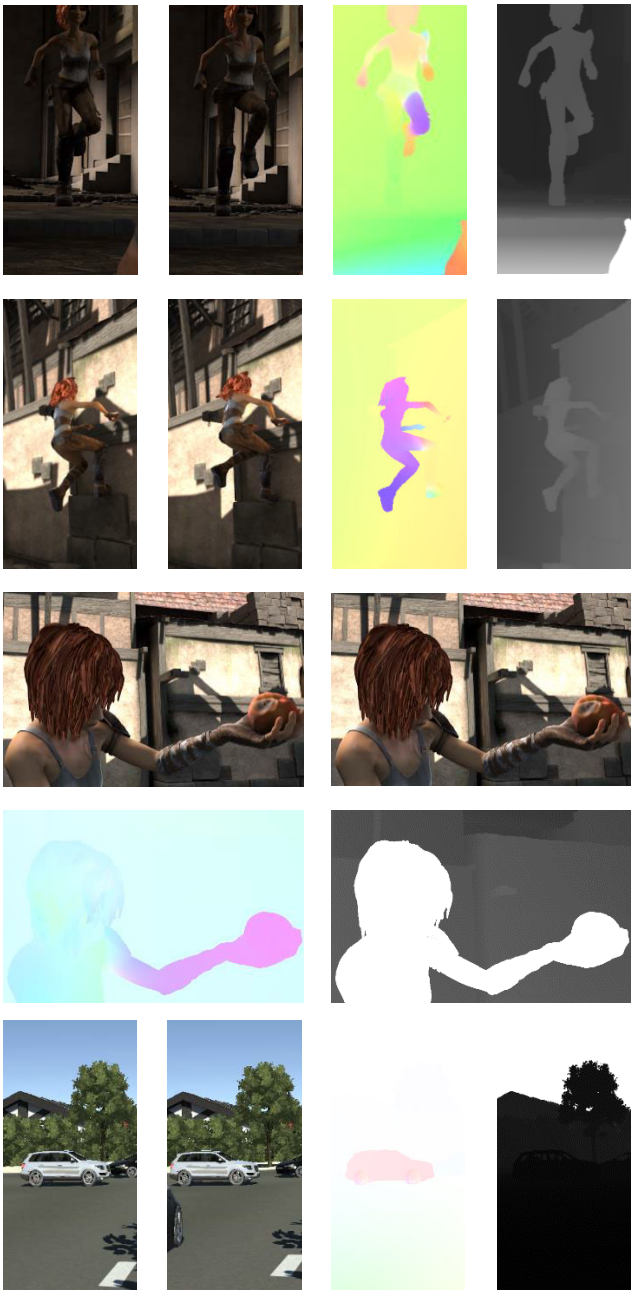

Figure 2. Demonstration of different dynamic cases. We propose our method by analyzing these scenes. For each example, we show the image pair, optical flow field and the depth map

we can reconstruct the scene with spatial relations. In summary, for superpixels in S_t or connecting to S_t (usually account for more than 90% of total superpixels), we directly reconstruct them with the assumption $R_e = R_s$. For the remaining superpixels, we change partial crack relations to hinge relations and then reconstruct the scene similarly.

4. Training details of learning-based methods

SfMLearner [10] and GeoNet [14] are deep learning-based methods that require training. The networks learn to predict the camera motion and depth map in an unsupervised setting, in which the loss is computed based on the photometric difference between a pair of images. The pretrained models provided by the authors were trained on the KITTI dataset [4], which contains sequences of urban scenes captured by a moving car. The frames in these sequences are composed mostly of static scenes (background) with only relatively small areas of moving objects (other vehicles). Therefore, depth estimation on these images are usually more accurate, since the majority of the scene can be reconstructed based solely on the camera motion. The MPI Sintel dataset [1], on the other hand, presents much more challenging dynamic scenes, in which a large part of the image may contain moving objects, requiring more robust depth estimation approaches.

In the paper, we compare our method against SfMLearner and GeoNet on four datasets. For the datasets containing urban driving scenes (KITTI [5, 3], Virtual KITTI [2], and SYNTHIA [13]), we report the results obtained directly from the pretrained model provided by the authors.

For MPI Sintel, we attempt to finetune the models before collecting the results. We follow the split proposed in [9], by retaining five sequences from the MPI Sintel training set as a validation set. We finetune both models by using the default configurations provided by respective the authors. We experimented reducing the learning rate by up to two orders of magnitude during the finetuning. We also tested two different image size: 416×128 (default configuration) and 512×192 which is more similar to the original MPI Sintel image size. The MRE (see Sec. 5 in the paper) training and validation plots are shown in Fig. 3.

As it can be seen, both methods fail to generalize well on the MPI Sintel dataset, presenting poor validation errors after finetuning. Over several runs, we observe some random minor improvement in the first iterations, which is always followed by a sudden increase in the error, corresponding to the network adapting to the new data. We can see that SfMLearner can improve on the training set, but not on the validation. GeoNet, on the other hand, shows unstable results during both stages, indicating that it cannot adapt to the dynamic scenes in MPI Sintel, at least with the default settings. Notice that the authors of both methods do not report results on MPI Sintel, which further indicates that they are not able to properly handle this type of scene.

5. Results on different datasets

We provide reconstruction results on different datasets, including MPI Sintel, KITTI, Virtual KITTI and SYNTHIA. Figure 4 and 6 compare the results of our method

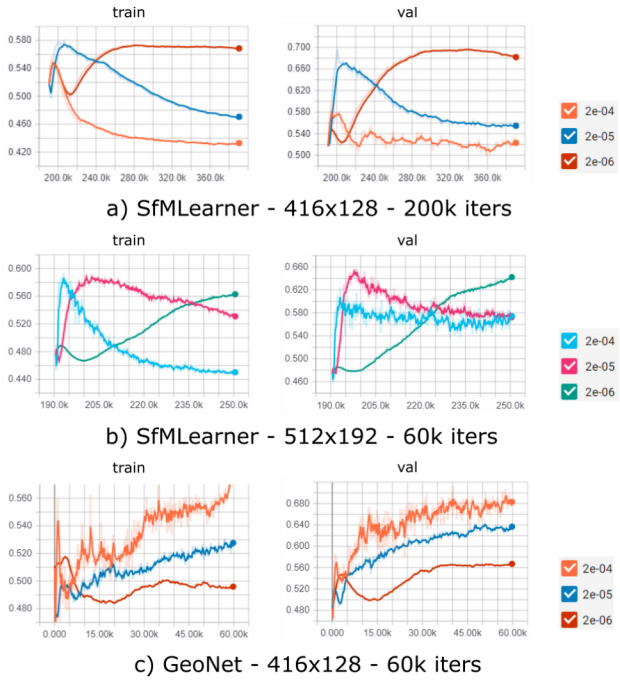

Figure 3. Plot of MRE during training and validation stages.

with competing methods. Figure 5 and 7 demonstrate the result of Ours+CPM (T). Our method may fail when the foreground objects do not connect to surrounding environment. As shown in Figure 7 Row 2, the depth of the flying duck is estimated incorrectly.

References

[1] Daniel J. Butler, Jonas Wulff, Garrett B. Stanley, and Michael J. Black. A naturalistic open source movie for optical flow evaluation. In A. Fitzgibbon et al. (Eds.), editor, *ECCV*, Part IV, LNCS 7577, pages 611–625. Springer-Verlag, Oct. 2012.

[2] Adrien Gaidon, Qiao Wang, Yohann Cabon, and Eleonora Vig. Virtual worlds as proxy for multi-object tracking analysis. In *CVPR*, 2016.

[3] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The KITTI dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013.

[4] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the KITTI vision benchmark suite. In *CVPR*, pages 3354–3361. IEEE, 2012.

[5] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the KITTI vision benchmark suite. In *CVPR*, 2012.

[6] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[7] Yinlin Hu, Yunsong Li, and Rui Song. Robust interpolation of correspondences for large displacement optical flow. In *CVPR*, 2017.

[8] Junhwa Hur and Stefan Roth. Mirrorflow: Exploiting symmetries in joint optical flow and occlusion estimation. In *ICCV*, pages 312–321, 2017.

[9] Suryansh Kumar, Yuchao Dai, and Hongdong Li. Monocular dense 3D reconstruction of a complex dynamic scene from two perspective frames. In *ICCV*, pages 4649–4657, 2017.

[10] Reza Mahjourian, Martin Wicke, and Anelia Angelova. Unsupervised learning of depth and ego-motion from monocular video using 3D geometric constraints. In *CVPR*, pages 5667–5675, 2018.

[11] Rene Ranftl, Vibhav Vineet, Qifeng Chen, and Vladlen Koltun. Dense monocular depth estimation in complex dynamic scenes. In *CVPR*, pages 4058–4066, 2016.

[12] Carolina Raposo and Joao P Barreto. π match: Monocular vSLAM and piecewise planar reconstruction using fast plane correspondences. In *ECCV*, pages 380–395. Springer, 2016.

[13] German Ros, Laura Sellart, Joanna Materzynska, David Vazquez, and Antonio M. Lopez. The SYNTHIA dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *CVPR*, June 2016.

[14] Zhichao Yin and Jianping Shi. Geonet: Unsupervised learning of dense depth, optical flow and camera pose. In *CVPR*, volume 2, 2018.

[15] Tinghui Zhou, Matthew Brown, Noah Snavely, and David G Lowe. Unsupervised learning of depth and ego-motion from video. In *CVPR*, volume 2, page 7, 2017.

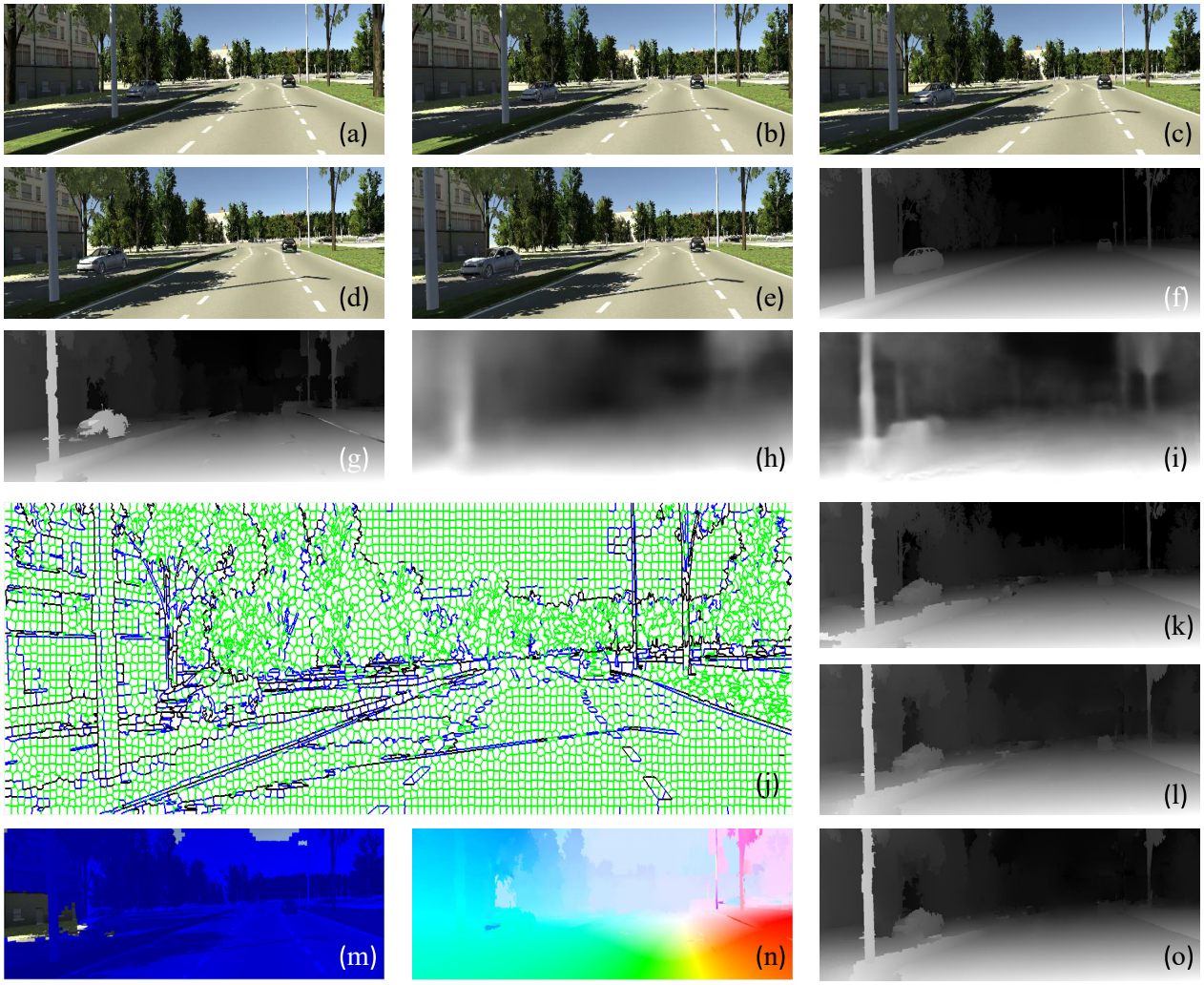

Figure 4. Our main results on Virtual KITTI dataset. (a)-(e): input image sequence. (f): ground truth depth. (g): result of MVG [6]. (h) result of SFMLearner [15]. (i): result of Geonet[14]. (j): spatial relations. (k): result of Ours+CPM (T). (l): result of Ours+CPM (M). (m):selected static superpixels . (n): optical flow estimated by MirrorFlow [8]. (o) result of Ours+MirrorFlow (T).

Figure 5. Demonstration of depth map estimated by Ours+CPM (T) in traffic scenes. **From left to right:** reference frame I_t, next frame I_{t+1}, ground truth depth map and the result of Ours+CPM (T).

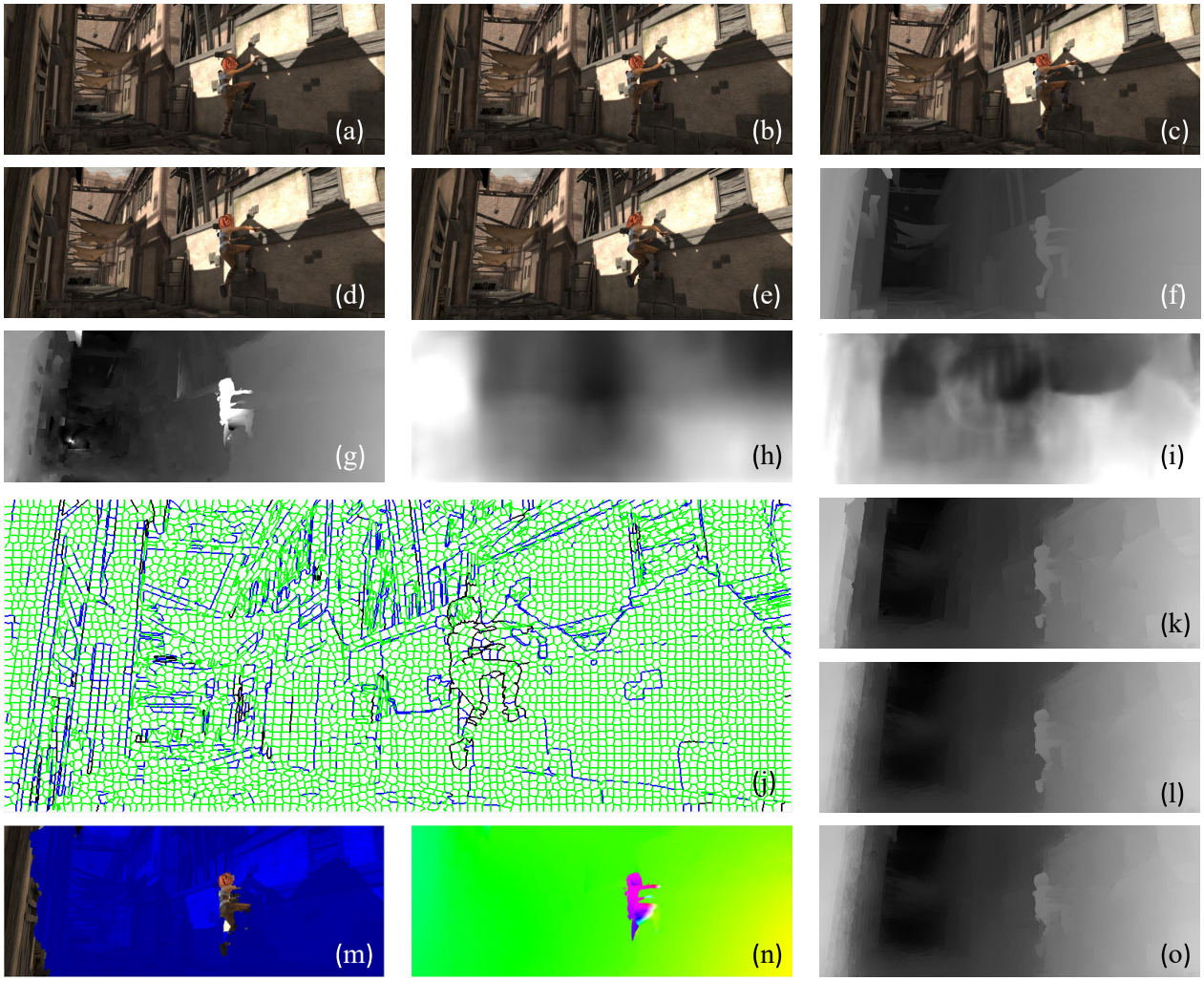

Figure 6. Our main results on MPI Sintel dataset. The meaning of each picture is the same with Figure 4

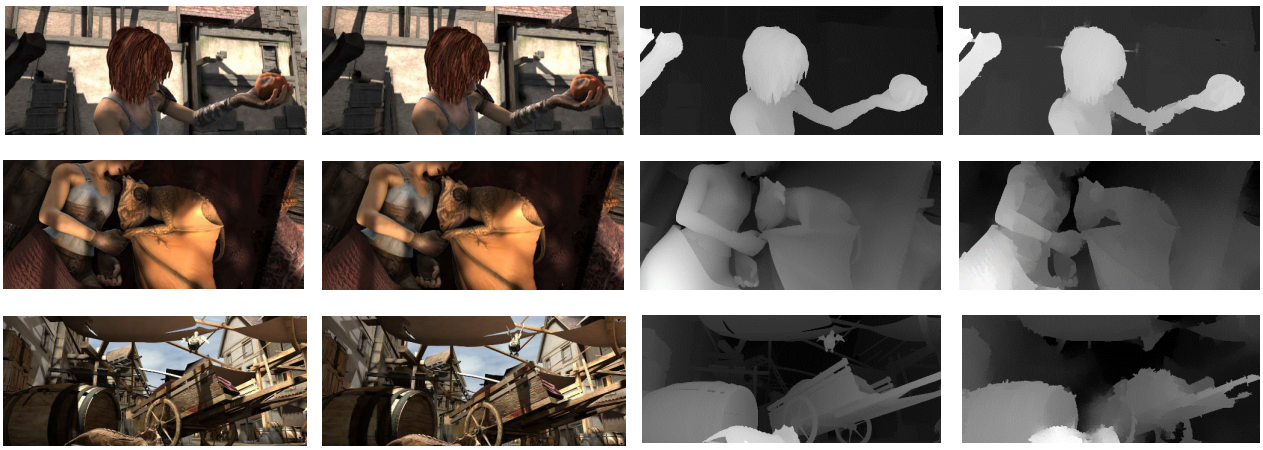

Figure 7. Demonstration of depth map estimated by Ours+CPM (T) on MPI Sintel dataset. **From left to right:** reference frame I_t, next frame I_{t+1}, ground truth depth map and the result of Ours+CPM (T).