Layered RGBD Scene Flow Estimation

Deqing Sun[1] Erik B. Sudderth[2] Hanspeter Pfister[1]
[1]Harvard University [2]Brown University

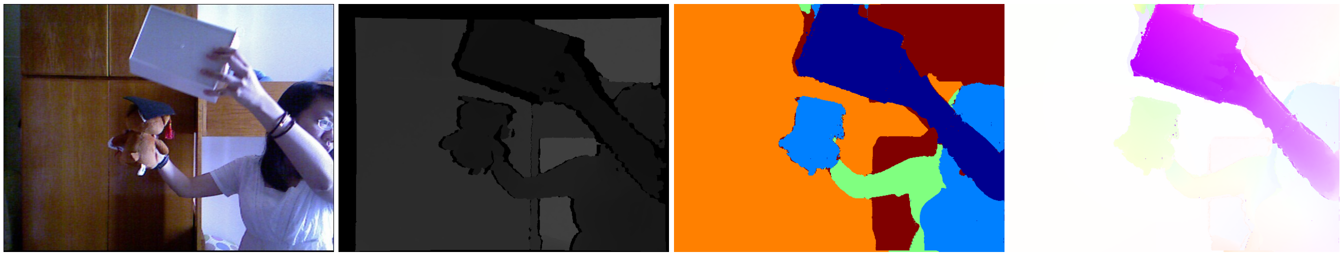

(a) First RGB image (b) First depth (c) Segmentation (d) 2D motion

Figure 1. Our layered approach can handle multiple moving objects and reliably estimate their motion in occlusion regions. Our key observation is that depth provides the depth ordering information, thereby solving a computational bottleneck for previous RGB layered methods (please see Figure 6 for our detected occlusions and the estimated motion by the recent semi-rigid scene flow (SRSF) method [27]).

Abstract

As consumer depth sensors become widely available, estimating scene flow from RGBD sequences has received increasing attention. Although the depth information allows the recovery of 3D motion from a single view, it poses new challenges. In particular, depth boundaries are not well-aligned with RGB image edges and therefore not reliable cues to localize 2D motion boundaries. In addition, methods that extend the 2D optical flow formulation to 3D still produce large errors in occlusion regions. To better use depth for occlusion reasoning, we propose a layered *RGBD scene flow method that jointly solves for the scene segmentation and the motion. Our key observation is that the noisy depth is sufficient to decide the depth ordering of layers, thereby avoiding a computational bottleneck for RGB layered methods. Furthermore, the depth enables us to estimate a per-layer 3D rigid motion to constrain the motion of each layer. Experimental results on both the Middlebury and real-world sequences demonstrate the effectiveness of the layered approach for RGBD scene flow estimation.*

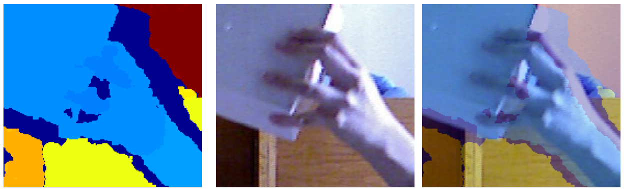
1. Introduction

Estimating the 3D motion of a dynamic scene is a fundamental challenge in computer vision and has many applications. This so-called scene flow estimation problem [42] usually requires multiple, calibrated RGB cameras to jointly solve for stereo and motion. As consumer depth sensors become widely adopted, *e.g*. Microsoft Kinect, the depth information has brought breakthroughs to several vision tasks, such as pose estimation [31], intrinsic image decomposition [6], and object detection [34]. The idea of leveraging depth cues to allow monocular scene flow estimation has also received increasing attention.

Although single-view depth cues give information about 3D motion, the noise in these depth estimates poses new challenges for RGBD scene flow estimation. As shown in Figure 2, depth boundaries are not well-aligned with RGB image edges, and thus we cannot accurately localize 2D motion boundaries using depth. Furthermore, depth is missing around occlusion and disocclusion regions. Occlusions are a major source of errors for both the 2D optical flow formulation and its 3D extension to RGBD data.

Layered models are a promising approach to model occlusions in RGB image sequences [44]. We can decompose a scene into moving layers ordered in depth. Occlusions become a signal produced by the motion of foreground layers. Recent RGB layered methods [40, 41] have obtained promising results in the presence of occlusions, but are computationally expensive. One bottleneck is to infer the depth ordering of layers, which often requires searching over a combinatorial space ($K!$ possibilities for K layers).

We propose using the depth information from RGBD data to solve the depth ordering problem for layered models. Our work is based on a surprisingly simple observation that *depth determines depth ordering*, as shown in Figure 1. The depth further allows us to estimate the 3D rotation and translation for each moving layer. The resultant 3D rigid motion



(a) Color-coded depth (b) Image (c) Overlaid

Figure 2. Crop of region around the left hand in Figure 1. The depth is noisy and has missing values (dark blue). In particular, the depth boundaries are not well-aligned with object boundaries in the 2D image plane. Our observation is that the depth is sufficient to decide the depth ordering of the foreground and the background.

can serve as a stronger prior to constrain the per-layer motion than pairwise Markov random field (MRF) models.

We evaluate our method using several datasets. On the widely used Middlebury benchmark, our motion estimation errors are about half those of the state-of-the-art semi-rigid scene flow (SRSF) method [27]. On real sequences from [27] and [33], our method obtains visually comparable or better results, particularly for scenes with multiple moving objects and large occlusions. To further innovations, we make our MATLAB code publicly available [1].

2. Previous Work

There is a vast literature on optical flow estimation, layered models, RGBD data processing, and scene flow estimation. We briefly review the most relevant work and refer the interested readers to the references cited therein.

Optical flow estimation. Stimulated by several recent benchmarks [5, 11, 13], the field of optical flow estimation has made rapid progress. Most recent methods build on the variational framework by Horn and Schunck [19]. Some notable variants include image-dependent, non-local motion priors to recover sharp motion boundaries [38, 47, 51], fusion moves to reach stronger local minima [24], embedding feature matching to track small, fast-moving objects [10, 45, 49], exploiting the epipolar constraint for egocentric scenes [50], and real-time implementations on GPU [48]. Despite the progress, large regions of occlusions remain a challenge, because occlusions violate the fundamental constancy assumption of optical flow.

RGB layered models. To deal with occlusions, Wang and Adelson [44] popularized the layered approach for motion analysis. Numerous papers have been published on RGB layered models, such as [4, 12, 22, 23, 46]. Recent methods [39, 41] have obtained promising results on optical flow benchmarks [5, 11], but are computationally expensive. One major bottleneck is to infer for the depth ordering of the layers, which requires a search over combinatorial possibilities. There are $K!$ possible layer orderings for K layers. For each depth ordering, we need to solve an energy minimization problem.

To avoid the global depth ordering problem, the local layering approach [26, 36] decides occlusions locally on a per occurrence basis. We can reason about the occlusion relationships between local image regions. From the local occlusion relationships, we can further obtain a pseudo depth [36] to visualize the local depth ordering. The true depth, however, is available in RGBD images. This observation motivates us to adopt a layered representation for RGBD scene flow estimation.

Single-frame RGBD data processing. Depth has simplified some common challenges in vision and brought breakthroughs to several problems, particularly for a single RGBD image. Shotton *et al.* [31] develop a random forest classifier [3, 8] to predict 3D positions of body joints from a single depth image. Barron *et al.* [6] leverage the depth for intrinsic image decomposition of natural scenes. Song and Xiao [34] show that depth can avoid many difficulties for object detection from RGB images, such as variations in illumination, shape, and viewpoint. In particular, they adopt the Truncated Signed Distance Function (TSDF) as a cue for self-occlusion.

Multi-camera scene flow estimation. Vedula *et al.* [42] introduce the 3D scene flow estimation problem and present a framework to compute scene flow from optical flow. Vogel *et al.* [43] represent a scene with a collection of locally planar, rigidly moving planes and obtain top-ranking results on the KITTI benchmark [13]. However, their method is more suitable for rigidly moving scenes.

RGBD scene flow estimation. Gottfried *et al.* [15] propose the first RGBD scene flow method for Kinect data by extending the optical flow formulation of Horn and Schunck [19]. They mask off invalid depth measurements to deal with noisy and unstable depth estimates. Herbst *et al.* [18] use a depth-modulated spatial regularizer and allow motion boundaries to occur along depth boundaries. Hadfield *et al.* [17] estimate the motion of sparse points and perform postprocessing to obtain a dense motion field. Quigoro *et al.* [28] combine both local and global approaches for RGBD scene flow estimation.

Quigoro *et al.* [27] propose a per-pixel overparameterized representation using rotation and translation to capture the global rigid motion of the scene while allowing local deviations. Their method is more suited for a single moving object. By comparison our method captures the global 3D motion for each individually moving object and can handle multiple moving objects. Ghuffar *et al.* [14] first estimate the scene flow and then perform motion segmentation. Their motion estimation method does not model occlusions. The errors in the estimated motion may propagate to the segmentation results, similar to the motion-based segmentation approach for RGB image sequences [16]. We jointly solve for the motion and the segmentation to avoid error propagation.

3. Model and Inference

Given a sequence of images and depth $\{\mathbf{I}_t, \mathbf{z}_t\}, 1 \leq t \leq T$, we want to segment the scene into moving layers and estimate the motion for each layer. We assume that the scene consists of K independently moving layers, ordered in depth [35, 39]. To model the layer segmentation, we use $K-1$ continuous support functions for the first $K-1$ layers. The support function for the kth layer at frame t, $\mathbf{g}_{tk}$, encodes how likely a pixel belongs to that layer at frame t. The Kth layer is background and has no support function. The motion of the kth layer includes both the 2D motion in the image plane $\{\mathbf{u}_{tk}, \mathbf{v}_{tk}\}$ and the change in depth $\mathbf{w}_{tk}$.

Probabilistically we want to solve for the most likely layer support functions and motion fields given the image evidences. We can decompose the posterior distribution, using Bayesian rules, as

$$
\begin{aligned}
p(\mathbf{u}_t, \mathbf{v}_t, \mathbf{w}_t, \mathbf{g}_t | \mathbf{I}_t, \mathbf{I}_{t+1}, \mathbf{z}_t, \mathbf{z}_{t+1}, \mathbf{g}_{t+1}) \propto & \qquad (1)\\
p(\mathbf{I}_{t+1} | \mathbf{I}_t, \mathbf{u}_t, \mathbf{v}_t, \mathbf{g}_t) p(\mathbf{z}_{t+1} | \mathbf{z}_t, \mathbf{u}_t, \mathbf{v}_t, \mathbf{w}_t, \mathbf{g}_t) & \\
p(\mathbf{u}_t, \mathbf{v}_t, \mathbf{w}_t | \mathbf{g}_t, \mathbf{I}_t) p(\mathbf{g}_{t+1} | \mathbf{g}_t, \mathbf{u}_t, \mathbf{v}_t) p(\mathbf{g}_t | \mathbf{I}_t), &
\end{aligned}
$$

where the first/second term describes how the next image/depth depends on the current image/depth, the motion, and the layer support. The third term describes how the motion depends on the layer support and the image, the fourth term describes how the layer support evolves over time, and the last term describes how the layer support depends on the image. We solve for the layer support and motion by maximizing the posterior distribution. It is equivalent to minimizing its negative log function, the energy function, given in Eq. (2). Next we will explain each term of the energy function in Eq. (2) and its assumptions, starting from the prior term for the layer support.

Spatio-temporally coherent layer support. We capture the spatial coherence of the layer support function via Gaussian conditional random field in which edge weights are modulated by differences in color vectors:

$$
E_{\text{spa-g}}(\mathbf{g}_{tk}) = \sum_{\mathbf{x}} \sum_{\mathbf{x}' \in \mathcal{N}_{\mathbf{x}}} \big(g_{tk}(\mathbf{x}) - g_{tk}(\mathbf{x}')\big)^2 \omega_{\mathbf{x}'}^{\mathbf{x}}, \qquad (3)
$$

$$
\omega_{\mathbf{x}'}^{\mathbf{x}} = \max \left\{ \exp\{-\frac{|\mathbf{I}_t(\mathbf{x}) - \mathbf{I}_t(\mathbf{x}')|^2}{\sigma_I^2}\}, \omega_0 \right\}, \qquad (4)
$$

where the set $\mathcal{N}_{\mathbf{x}}$ contains the four nearest spatial neighbors of a pixel $\mathbf{x} = (x, y)^T$, and ω_0 is a constant to add robustness to large color variations in texture regions.

We encourage the temporal coherence of surfaces via a Gaussian MRF:

$$
E_{\text{time}}(\mathbf{g}_{tk}, \mathbf{g}_{t+1,k}, \mathbf{u}_{tk}, \mathbf{v}_{tk}) = \sum_{\mathbf{x}} \big(g_{tk}(\mathbf{x}) - g_{t+1,k}(\tilde{\mathbf{x}})\big)^2, \qquad (5)
$$

where $\tilde{\mathbf{x}} = (x + u_{tk}(\mathbf{x}), y + v_{tk}(\mathbf{x}))^T$ is the corresponding pixel at the next frame according to the 2D motion field of the kth layer at the current frame $\{\mathbf{u}_{tk}, \mathbf{v}_{tk}\}$.

We obtain the layer ownership by sequentially thresholding the layer support functions

$$
s_{tk}(\mathbf{x}) = \prod_{l=1}^{k-1} \delta\big(g_{tl}(\mathbf{x}) < 0\big) \delta\big(g_{tk}(\mathbf{x}) \geq 0\big), \qquad (6)
$$

where $\delta(\cdot)$ is an indicator function. $s_{tk}(\mathbf{x}) = 1$ means that the pixel $\mathbf{x}$ is visible at the kth layer. Pixels invisible at the first $K-1$ layers belong to the last, background layer.

Globally rigid, locally flexible motion. The temporal coherence term of the layer support in Eq. (5) uses the motion field of each layer to non-rigidly align layers across time. Now we will explain our semi-parametric motion model. To capture the global behavior of the motion fields, we assume that pixels in the kth layer share a common 3D rotation $\mathbf{R}_{tk}$ and translation τ_{tk} relative to the camera. For a pixel $\mathbf{x} = (x, y)^T$ with depth z, its 3D position is

$$
\mathbf{X} = \left(\frac{x - c_x}{f_x} z, \frac{y - c_y}{f_y} z, z \right)^T, \qquad (7)
$$

where c_x and c_y are the camera centers, and f_x and f_y are the product of the focal length with the scale parameters in the horizontal and vertical directions. Under the rotation $\mathbf{R}_{tk}$ and translation τ_{tk}, the new 3D position of $\mathbf{x}$ is

$$
(X_2, Y_2, z_2)^T = \mathbf{R}_{tk} \mathbf{X} + \tau_{tk}, \qquad (8)
$$

and the corresponding 2D position is

$$
\left(f_x \frac{X_2}{z_2} + c_x, f_y \frac{Y_2}{z_2} + c_y \right)^T. \qquad (9)
$$

The scene flow resulting from $\mathbf{R}_{tk}$ and τ_{tk} is

$$
u_{tk}^R(\mathbf{x}) = f_x \frac{X_2}{z_2} + c_x - x, \qquad (10)
$$

$$
v_{tk}^R(\mathbf{x}) = f_y \frac{Y_2}{z_2} + c_y - y, \qquad (11)
$$

$$
w_{tk}^R(\mathbf{x}) = z_2 - z, \qquad (12)
$$

where X_2, Y_2, and z_2 are defined in Eq. (8). We use this rigid motion field $(\mathbf{u}_{tk}^R, \mathbf{v}_{tk}^R, \mathbf{w}_{tk}^R)$ as the mean motion for every layer. To model fine details, we allow the motion of each layer to have small deviations from its mean [21]. Our semi-parametric model for the horizontal motion is

$$
\begin{aligned}
E_{\text{spa-u}}(\mathbf{u}_{tk}, \mathbf{R}_{tk}, \tau_{tk}) = \sum_{\mathbf{x}} \lambda_{\text{b}} \rho_b\big(u_{tk}(\mathbf{x}) - u_{tk}^R(\mathbf{x})\big) & \qquad (13)\\
+ \sum_{\mathbf{x}' \in \mathcal{N}_{\mathbf{x}}} \rho_u \Big(\big(u_{tk}(\mathbf{x}) - u_{tk}^R(\mathbf{x})\big) - \big(u_{tk}(\mathbf{x}') - u_{tk}^R(\mathbf{x}')\big) \Big), &
\end{aligned}
$$

$$E(\mathbf{u},\mathbf{v},\mathbf{w},\mathbf{g},\mathbf{R},\tau) = \sum_{t=1}^{T-1} \left\{ \sum_{k=1}^{K} \Big\{ E_{\text{data}}(\mathbf{u}_{tk},\mathbf{v}_{tk},\mathbf{g}_t) + \lambda_{\text{depth}} E_{\text{depth}}(\mathbf{u}_{tk},\mathbf{v}_{tk},\mathbf{w}_{tk}) + \lambda_{\text{motion}} \big\{ E_{\text{spa-u}}(\mathbf{u}_{tk}, R_{tk}, \tau_{tk}) + \right.$$

$$\left. E_{\text{spa-v}}(\mathbf{v}_{tk}, R_{tk}, \tau_{tk}) + E_{\text{spa-w}}(\mathbf{w}_{tk}, R_{tk}, \tau_{tk}) \big\} \Big\} + \sum_{k=1}^{K-1} \lambda_{\text{time}} E_{\text{time}}(\mathbf{g}_t, \mathbf{g}_{t+1}, \mathbf{u}_{tk}, \mathbf{v}_{tk}) \right\} + \sum_{t=1}^{T} \sum_{k=1}^{K-1} \lambda_{\text{support}} E_{\text{spa-g}}(\mathbf{g}_{tk}). \quad (2)$$

where ρ_b and ρ_u are robust penalty function. We assume that the deviations of the horizontal motion, vertical motion, and depth change are independent and define the energy functions of the latter two similarly.

Data constancy and occlusion reasoning. To reason about occlusions for pixel $\mathbf{x}$ at layer k, we examine the layer assignment at the current frame $s_{tk}(\mathbf{x})$ and at the corresponding pixel at the next frame $\tilde{\mathbf{x}} = (x + u_{tk}(\mathbf{x}), y + v_{tk}(\mathbf{x}))^T$. If both $s_{tk}(\mathbf{x})$ and $s_{t+1,k}(\tilde{\mathbf{x}})$ are one, the pixel is visible at both frames and should have constant appearance. Otherwise, occlusion happens and we should disable the constancy term. Let $\tilde{\mathbf{I}}(\mathbf{x})$ be an observed image feature for pixel $\mathbf{x}$. Our data constancy term is

$$E_{\text{data}}(\mathbf{u}_{tk},\mathbf{v}_{tk},\mathbf{g}_t) = \sum_{\mathbf{x}} s_{tk}(\mathbf{x})\big(1 - s_{t+1,k}(\tilde{\mathbf{x}})\big)\lambda_o + s_{tk}(\mathbf{x}) s_{t+1,k}(\tilde{\mathbf{x}}) \rho_c\big(\tilde{\mathbf{I}}_t(\mathbf{x}) - \tilde{\mathbf{I}}_{t+1}(\tilde{\mathbf{x}})\big), \quad (14)$$

where ρ_c a robust penalty function, and λ_o is a constant penalty for being occluded. Note that the data term for the kth layer depends on support functions of the first k layers.

RGBD data has another constraint on the variations of the depth over time. For pixels with valid depth measurement, the depth variation constraint is

$$E_{\text{depth}}(\mathbf{u}_{tk},\mathbf{v}_{tk},\mathbf{w}_{tk}) = \sum_{\mathbf{x}} s_{tk}(\mathbf{x})\big(1 - s_{t+1,k}(\tilde{\mathbf{x}})\big)\lambda_d + s_{tk}(\mathbf{x}) s_{t+1,k}(\tilde{\mathbf{x}}) \rho_d\big(z_{t+1}(\tilde{\mathbf{x}}) - z_t(\mathbf{x}) - w_{tk}(\mathbf{x})\big), \quad (15)$$

where ρ_d is a robust penalty function and λ_d is again a constant penalty for being occluded.

Inference. We perform coordinate descent to minimize the energy function in Eq. (2). First, given the support function, we compute the occlusion regions and use an incremental warping based scheme to optimize the motion [9]. Next, given the motion, we replace the thresholding function in Eq. (6) by a sigmoid function and jointly optimize all the support functions using conjugate gradient descent [29, 37]. As for the initial value of the segmentation, we start from a K-means clustering of the depth values. The average depth values of the layers decide the depth ordering. We also compute optical flow from RGB images. The optical flow provides a correspondence between pixels at adjacent frames. We can project these corresponding pixels into 3D. We estimate the global 3D rotation and translation between these two sets of corresponding 3D points using Horn's quaternion-based method [2, 20].

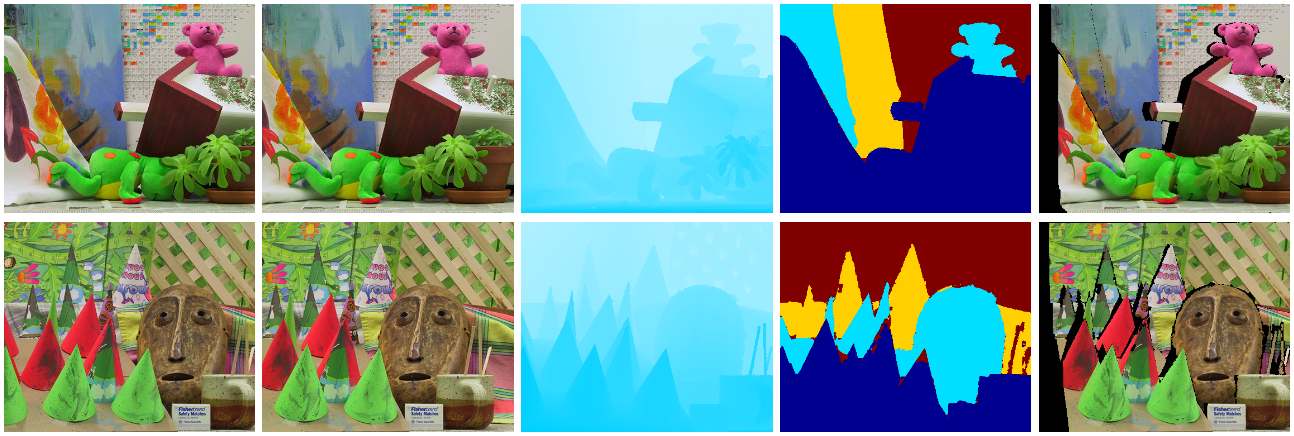
For the RGB layered methods, inferring the depth ordering of layers is computationally expensive, because it requires an exhaustive search over $K!$ possibilities for K layers. For each depth ordering, we need to minimize the energy function of the motion and the segmentation. The depth ordering that gives the lowest energy solution is then selected. For RGBD layered methods, the depth directly provides the depth ordering information, and we just need to perform the energy minimization once.

4. Experimental Results

We evaluate our method on the widely used Middlebury benchmark and several real-world datasets. These datasets have different characteristics, such as the intrinsic camera parameter, the image and depth quality, and the depth range. Following [27], we use different parameter settings for different datasets. Please refer to the reference code for the exact parameter settings [1]. Future work will consider automatically adapting the parameters to a new dataset. We use the colorization scheme [25, 32] to inpaint missing depth values. The initial optical flow is computed using the "Classic+NLP" method [38].

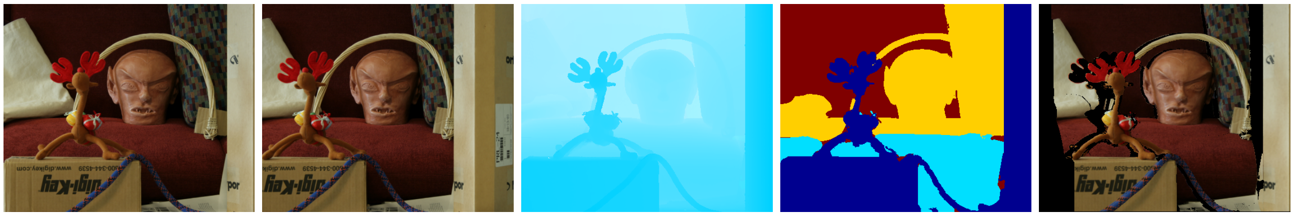
Middlebury 2002 dataset. We first test methods on the Middlebury stereo 2002 dataset [30], which has been commonly used for evaluating RGBD scene flow methods. We follow the setup in [27] and evaluate the accuracy of the 2D optical flow. Table 1 summarizes the numerical results. Our method outperforms the semi-rigid scene flow (SRSF) method [27], which achieves the best results on this benchmark in the literature. Compared with the RGB layered model [41], the improvement by RGBD methods could be up to an order of magnitude. Figure 3 shows the estimated flow fields by the proposed method. The available high quality depth (disparity) allows our method to recover very fine motion details.

Middlebury 2005 dataset. We further test the methods using the Middlebury 2005 dataset. These sequences are more challenging than the 2002 dataset, as shown in Table 2. On average, our method has about half the error of that by the SRSF method. Visually, the warped image



(a) First frame (b) Second frame (c) Estimated 2D motion (d) Segmentation (e) Warped image

Figure 3. Estimated optical flow on the Middlebury dataset. We encode the motion encoded using the color scheme [5] and the segmentation using the scheme [41] (blue to red: near to far). Top: "Teddy". Bottom: "Cones". The black pixels in the warped image correspond to the detected occlusions.



(a) First frame (b) Second frame (c) Estimated 2D motion (d) Segmentation (e) Warped image

Figure 4. Estimated optical flow on "Reindeer" from the Middlebury 2005 dataset. Detected occlusions are marked as black in the warped image.

Table 1. Average root mean squared (RMS) error and average angular error (AAE) results on Middlebury dataset (* means the method uses only RGB images).

	Teddy		Cones	
	RMS	AAE	RMS	AAE
Proposed ($K = 4$)	**0.09**	**0.17**	**0.12**	**0.13**
SRSF [27]	0.49	0.46	0.45	0.37
Hadfield and Bowden [17]	0.52	1.36	0.59	1.61
Basha *et al.* [7] (2 views)	0.57	1.01	0.58	0.39
Quiroga *et al.* [28]	0.94	0.84	0.79	0.52
FC-2Layers-FF [41]*	1.90	0.24	2.80	0.18
Classic+NLP [38]*	2.00	0.24	2.79	0.20
LDOF [10] + depth	2.11	0.43	2.30	0.52

Table 2. Average root mean squared (RMS) error and average angular error (AAE) results on Middlebury 2005 dataset (* means the method uses only RGB images).

	Quiroga *et al.* [28]		Proposed (K=4)	
	RMS	AAE	RMS	AAE
Art	2.33	1.10	1.69	0.77
Books	1.64	0.74	0.39	0.18
Dolls	0.63	0.25	0.17	0.13
Laundry	5.44	0.80	4.24	0.54
Moebius	1.08	0.44	0.13	0.15
Reindeer	3.04	0.97	0.53	0.38
Average	2.36	0.72	**1.19**	**0.36**

look close to the first image except in occlusion regions, as shown in Figure 4.

SRSF dataset. As discussed in [27], the 3D motion field on the Middlebury data mainly results from a single global camera translation. The data thus does not fully evaluate general-purpose RGBD scene flow methods. Thus, we also qualitatively test our method on real-world sequences.

Figure 5 shows results on sequences from [27]. Our method obtains similar results as SRSF [27][1]. Note that SRSF only outputs results for pixels within a specified depth range. Our results compare favorably with the valid output by SRSF and look reasonable for other pixels. The second row in Figure 5 reveals some problems of our method. The foreground and the background share similar appearance.

[1] Results of SRSF are obtained using the public software from the authors' website.

Table 3. Average root mean squared (RMS) error on Middlebury dataset for different numbers of layers. The performance slightly drops with more clusters. There is a single camera translation of the scene. Using more layers means each layers has less pixels to recover the shared motion.

K	2	3	4	5	6
Teddy	0.081	0.090	0.091	0.106	0.107
Cones	0.111	0.116	0.118	0.120	0.122

Our image-based prior merges some foreground pixels into the background. However, our method recovers finer motion details than SRSF, for example, the motion between the fingers at the last row in Figure 5.

RGBD tracking dataset. We further test the methods using sequences from the RGBD tracking dataset [33]. These sequences contain several independently moving objects and also large occlusions. In addition, the depth and the image are not well aligned, as shown in Figure 2. On the "bear_back" sequence shown in Figures 1 and 6, large regions of occlusions cause errors to the SRSF method. SRSF also produces errors by imposing a single rotation and translation to two oppositely moving arms. Our layered approach obtains reasonable results and the predicted occlusions match the image evidence. Our method has some errors in the isolated region at the top middle of the image, which could likely be reduced via a fully-connected prior model for layer support [41]. On the "Flower_red_occ" and "Bag1" sequences in Figures 8 and 7, the motion, segmentation, and occlusion results are qualitatively reasonable and match the scene structures. On all cases, the noisy depth is sufficient to decide the depth ordering.

Effect of layer number. Table 3 shows performance for varying numbers of layers on the "Teddy" and "Cones" sequences. The performance drops with more layers because the only motion is due to camera translation, and with more layers, each layer has less pixels to estimate the same 3D rotation and translation parameters. On the 'Flower_red_occ" sequence [33], two or three-layer models merge objects that move in different directions and produces errors (see Figure 8). Automatically adapting the model complexity to the structure of each scene is an important research direction.

Running time. Given motion computed by "Classic+NLP" [38], the two-frame 640 × 480 "bear_back" sequence with 4 layers takes our unoptimized MATLAB code about 9 minutes on an Intel Xeon 2.4 GHz Windows desktop. There are many opportunities for parallelization.

5. Conclusions

We have presented a layered RGBD scene flow estimation method for joint 3D motion estimation and segmentation. Our key observation is that even noisy depth cues provide reliable layer ordering information. This enables us to handle occlusions using the layered approach without paying a high computational cost. Depth also allows us to estimate a common 3D rigid motion to regularize the interpretation of each layer. Our layered method obtains encouraging results on both the Middlebury benchmark and real-world sequences, particularly those with independently moving objects and large occlusions. Our work suggests that it is promising to explore the rich information in depth data for 3D motion analysis. Our MATLAB code is publicly available at the first author's webpage [1].

Acknowledgement This work has been partially supported by NSF grant OIA 1125087, and by ONR Award No. N00014-13-1-0644. DS would like to thank Julian Quiroga for providing help with his code, Michelle Borkin, Jason Pacheco, and James Tompkin for proofreading.

References

[1] http://people.seas.harvard.edu/~dqsun/. 2, 4, 6

[2] http://www.mathworks.com/matlabcentral/fileexchange/26186-absolute-orientation-horn-s-method. 4

[3] Y. Amit and D. Geman. Shape quantization and recognition with randomized trees. *Neural computation*, 9(7):1545–1588, 1997. 2

[4] S. Ayer and H. S. Sawhney. Layered representation of motion video using robust maximum-likelihood estimation of mixture models and MDL encoding. In *IEEE International Conference on Computer Vision*, 1995. 2

[5] S. Baker, D. Scharstein, J. P. Lewis, S. Roth, M. J. Black, and R. Szeliski. A database and evaluation methodology for optical flow. *IJCV*, 92(1):1–31, March 2011. 2, 5

[6] J. T. Barron and J. Malik. Intrinsic scene properties from a single rgb-d image. *IEEE Conference on Computer Vision and Pattern Recognition*, 2013. 1, 2

[7] T. Basha, Y. Moses, and N. Kiryati. Multi-view scene flow estimation: A view centered variational approach. *IJCV*, 101(1):6–21, 2013. 5

[8] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001. 2

[9] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert. High accuracy optical flow estimation based on a theory for warping. In *European Conference on Computer Vision*, pages 25–36, 2004. 4

[10] T. Brox and J. Malik. Large displacement optical flow: Descriptor matching in variational motion estimation. *IEEE TPAMI*, 33(3):500–513, Mar. 2011. 2, 5

[11] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. A naturalistic open source movie for optical flow evaluation. In *European Conference on Computer Vision*, IV, pages 611–625, 2012. 2

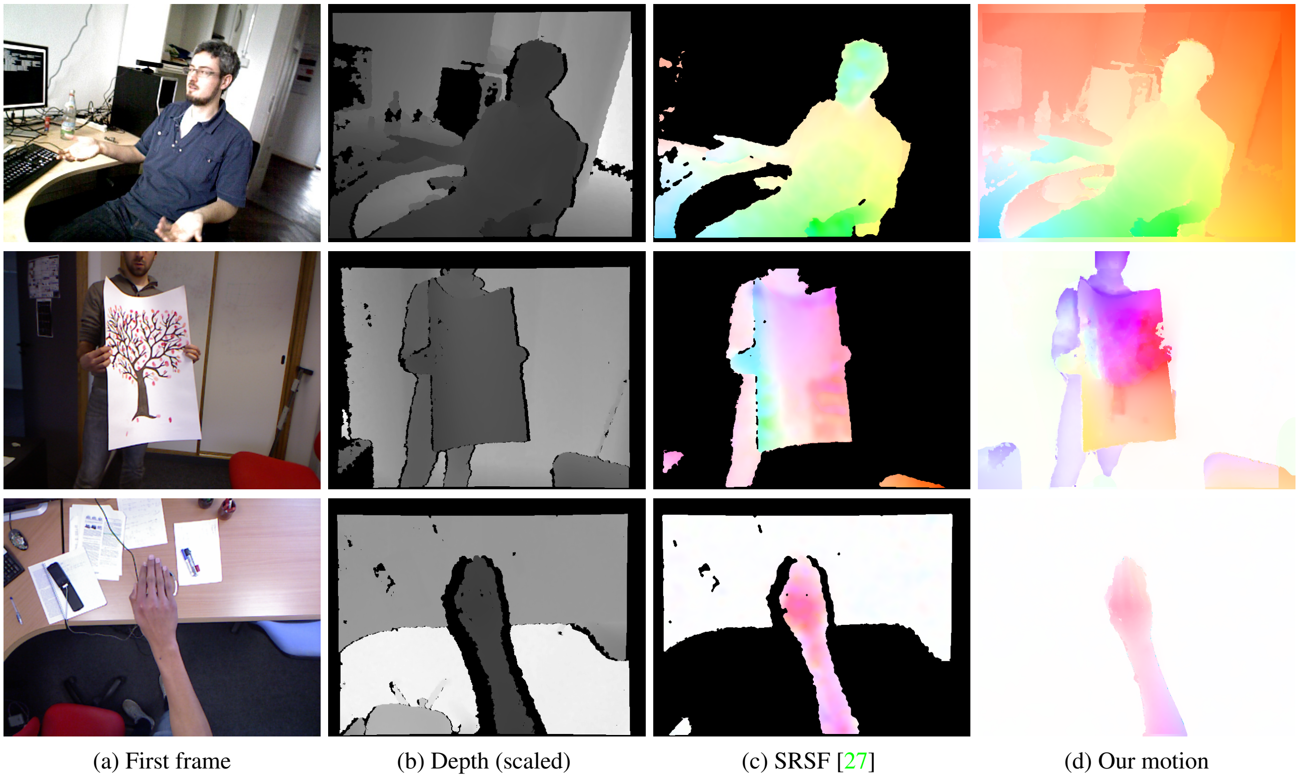


Figure 5. Results on sequences from [27]. SRSF has no output for the black region that has been identified as invalid. Top row: Our method obtains similar motion for the foreground person while producing reasonable estimate of the background motion. Middle row: our method obtains similar results but also has some artifacts; the painting has similar appearance as the background, which misleads our image-based prior to merge some foreground regions into the background. Bottom row: our method better captures the **motion details between fingers**.

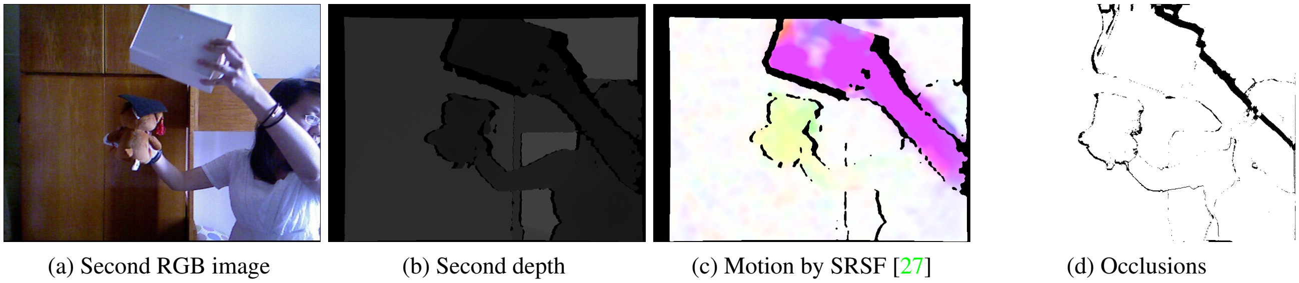


Figure 6. Depth input and our segmentation and occlusion for "bear_back" [33] in Figure 1. Note that the depth naturally tells the ordering of the layers, avoiding a computational bottleneck for layered models.

[12] T. Darrell and A. Pentland. Cooperative robust estimation using layers of support. *IEEE TPAMI*, 17(5):474–487, May 1995. 2

[13] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2012. 2

[14] S. Ghuffar, N. Brosch, N. Pfeifer, and M. Gelautz. Motion estimation and segmentation in depth and intensity videos. *Integrated Computer-Aided Engineering*, 2014. 2

[15] J.-M. Gottfried, J. Fehr, and C. S. Garbe. Computing range flow from multi-modal kinect data. In *Advances in Visual Computing*, pages 758–767. Springer, 2011. 2

[16] M. Grundmann, V. Kwatra, M. Han, and I. Essa. Efficient hierarchical graph-based video segmentation. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2141–2148, 2010. 3

[17] S. Hadfield and R. Bowden. Scene particles: Unregularized particle based scene flow estimation. *IEEE TPAMI*, 36:3, 2013. 2, 5

[18] E. Herbst, X. Ren, and D. Fox. Rgb-d flow: Dense 3-d motion estimation using color and depth. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 2276–2282. IEEE, 2013. 2

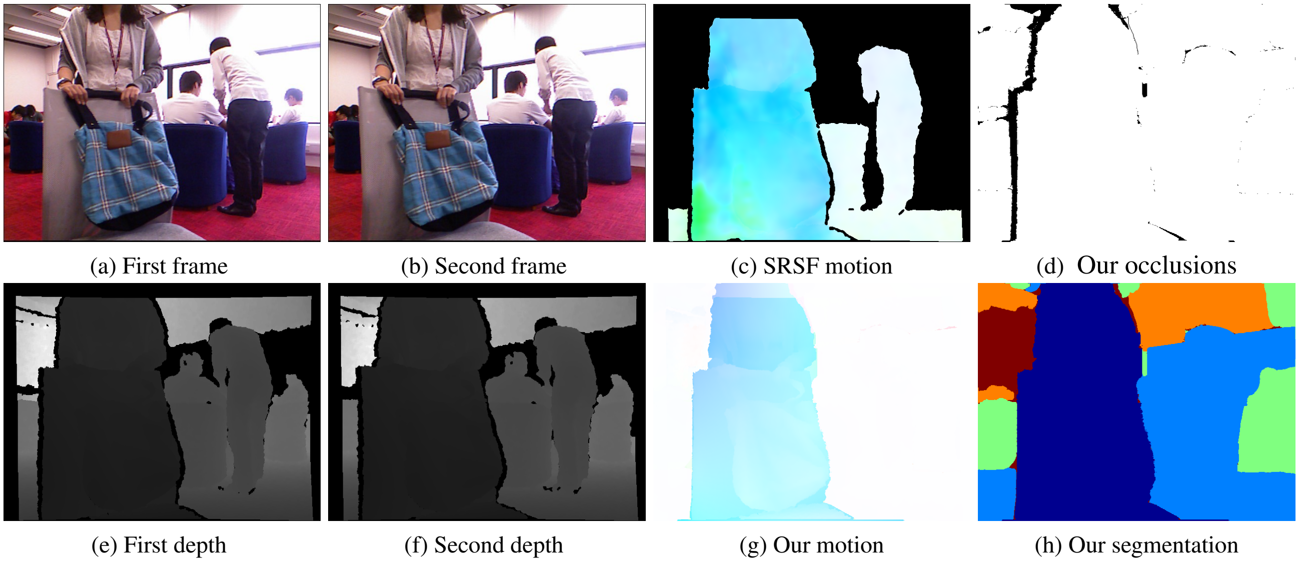


Figure 7. Results on "Bag1" [33]. Our results compare favorably with the valid output of SRSF. Note the detection occlusion match well with the image evidence.

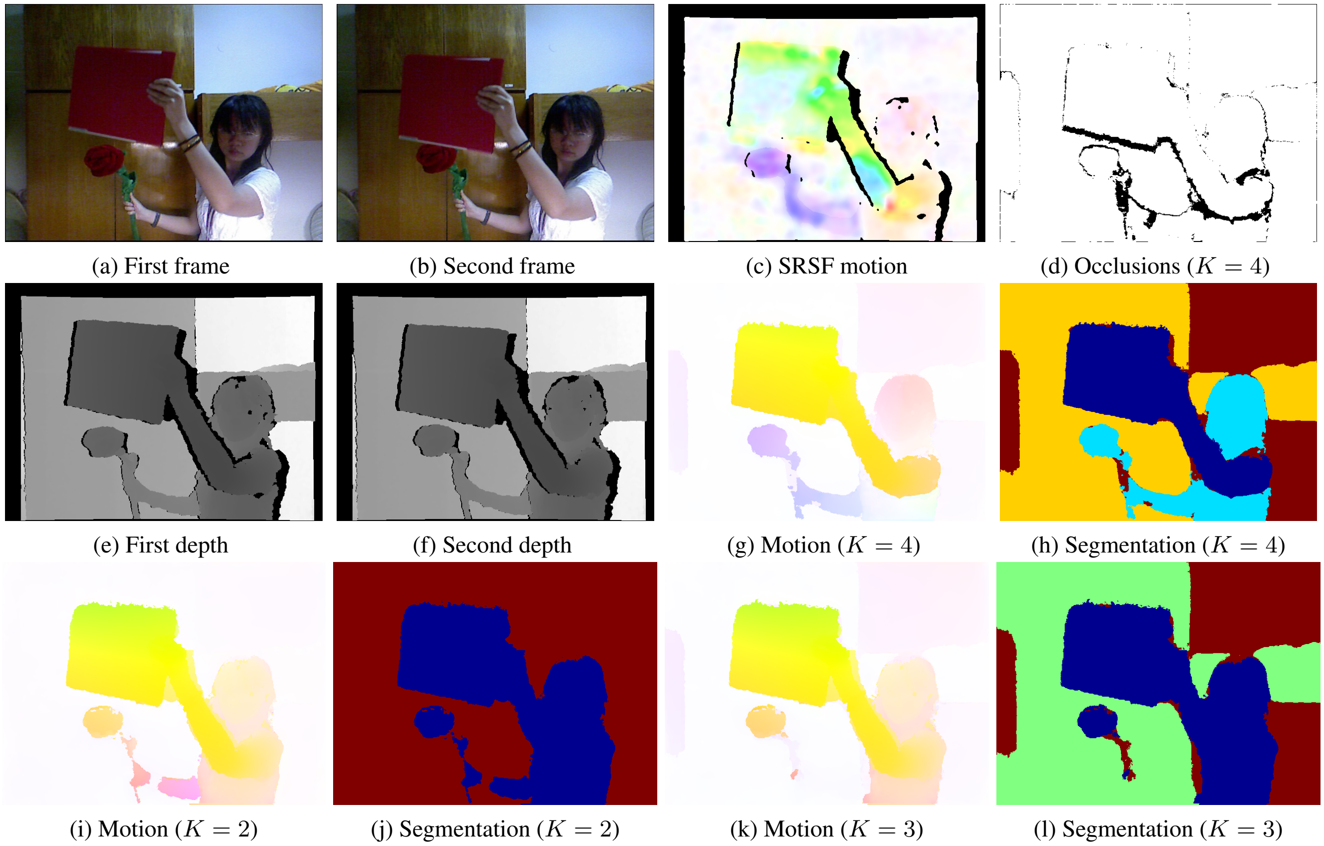


Figure 8. Results on "Flower_red_occ" [33]. Our method produces promising motion, segmentation, and occlusion results compared with the state of the art. Note that our method well captures the global motion by the left arm and the book. With two or three layers, our method produces large errors. It merges the flower with the book that are moving in opposite directions

[19] B. Horn and B. Schunck. Determining optical flow. *Artificial Intelligence*, 16:185–203, Aug. 1981. 2

[20] B. K. Horn. Closed-form solution of absolute orientation using unit quaternions. *JOSA A*, 4(4):629–642, 1987. 4

[21] M. Irani, P. Anandan, and D. Weinshall. From reference frames to reference planes: Multi-view parallax geometry and applications. In *European Conference on Computer Vision*, 1998. 3

[22] A. Jepson and M. J. Black. Mixture models for optical flow computation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 760–761, 1993. 2

[23] M. Kumar, P. Torr, and A. Zisserman. Learning layered motion segmentations of video. *IJCV*, 76(3):301–319, March 2008. 2

[24] V. Lempitsky, S. Roth, and C. Rother. FusionFlow: Discrete-continuous optimization for optical flow estimation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2008. 2

[25] A. Levin, D. Lischinski, and Y. Weiss. Colorization using optimization. In *ACM TOG*, volume 23, pages 689–694. ACM, 2004. 4

[26] J. McCann and N. S. Pollard. Local layering. *Siggraph*, 28(3), Aug. 2009. 2

[27] J. Quiroga, T. Brox, F. Devernay, and J. Crowley. Dense semi-rigid scene flow estimation from rgbd images. In *European Conference on Computer Vision*, 2014. 1, 2, 4, 5, 7

[28] J. Quiroga, F. Devernay, J. L. Crowley, et al. Local/global scene flow estimation. In *ICIP-IEEE International Conference on Image Processing*, 2013. 2, 5

[29] C. E. Rasmussen. http://www.gaussianprocess.org/gpml/code/matlab/util/minimize.m. 4

[30] D. Scharstein and R. Szeliski. High-accuracy stereo depth maps using structured light. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages I–195. IEEE, 2003. 4

[31] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. Real-time human pose recognition in parts from single depth images. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1297–1304. IEEE, 2011. 1, 2

[32] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus. Indoor segmentation and support inference from rgbd images. In *European Conference on Computer Vision*, pages 746–760, 2012. 4

[33] S. Song and J. Xiao. Tracking revisited using rgbd camera: Unified benchmark and baselines. In *IEEE International Conference on Computer Vision*, pages 233–240. IEEE, 2013. 2, 6, 7, 8

[34] S. Song and J. Xiao. Sliding shapes for 3d object detection in depth images. In *European Conference on Computer Vision*, pages 634–651. Springer, 2014. 1, 2

[35] E. Sudderth and M. Jordan. Shared segmentation of natural scenes using dependent Pitman-Yor processes. In *Neural Information Processing Systems*, pages 1585–1592, 2009. 3

[36] D. Sun, C. Liu, and H. Pfister. Local layering for joint motion estimation and occlusion detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2014. 2

[37] D. Sun, S. Roth, and M. J. Black. Secrets of optical flow estimation and their principles. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2432–2439, 2010. 4

[38] D. Sun, S. Roth, and M. J. Black. A quantitative analysis of current practices in optical flow estimation and the principles behind them. *IJCV*, 106(2):15–137, 2014. 2, 4, 5, 6

[39] D. Sun, E. B. Sudderth, and M. J. Black. Layered image motion with explicit occlusions, temporal consistency, and depth ordering. In *Neural Information Processing Systems*, pages 2226–2234, 2010. 2, 3

[40] D. Sun, E. B. Sudderth, and M. J. Black. Layered segmentation and optical flow estimation over time. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1768–1775, 2012. 1

[41] D. Sun, J. Wulff, E. B. Sudderth, H. Pfister, and M. J. Black. A fully-connected layered model of foreground and background flow. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2451–2458, 2013. 1, 2, 4, 5, 6

[42] S. Vedula, S. Baker, P. Rander, R. Collins, and T. Kanade. Three-dimensional scene flow. In *IEEE International Conference on Computer Vision*, volume 2, pages 722–729. IEEE, 1999. 1, 2

[43] C. Vogel, K. Schindler, and S. Roth. Piecewise rigid scene flow. In *IEEE International Conference on Computer Vision*, December 2013. 2

[44] J. Y. A. Wang and E. H. Adelson. Representing moving images with layers. *IEEE TIP*, 3(5):625–638, Sept. 1994. 1, 2

[45] P. Weinzaepfel, J. Revaud, Z. Harchaoui, and C. Schmid. DeepFlow: Large displacement optical flow with deep matching. In *IEEE International Conference on Computer Vision*, 2013. 2

[46] Y. Weiss. Smoothness in layers: Motion segmentation using nonparametric mixture estimation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 520–526, 1997. 2

[47] M. Werlberger, T. Pock, and H. Bischof. Motion estimation with non-local total variation regularization. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2010. 2

[48] M. Werlberger, W. Trobin, T. Pock, A. Wedel, D. Cremers, and H. Bischof. Anisotropic Huber-L1 optical flow. In *BMVC*, pages 108.1–108.11, 2009. 2

[49] L. Xu, J. Jia, and Y. Matsushita. Motion detail preserving optical flow estimation. *IEEE TPAMI*, 34(9):1744–1757, 2012. 2

[50] K. Yamaguchi, D. A. McAllester, and R. Urtasun. Robust monocular epipolar flow estimation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1862–1869, 2013. 2

[51] H. Zimmer, A. Bruhn, J. Weickert, L. Valgaerts, A. Salgado, B. Rosenhahn, and H.-P. Seidel. Complementary optic flow. In *EMMCVPR*, 2009. 2