Temporal Upsampling of Depth Maps Using a Hybrid Camera

Ming-Ze Yuan, Lin Gao*, Hongbo Fu, and Shihong Xia*

Abstract—In recent years, consumer-level depth cameras have been adopted for various applications. However, they often produce depth maps at only a moderately high frame rate (approximately 30 frames per second), preventing them from being used for applications such as digitizing human performance involving fast motion. On the other hand, low-cost, high-frame-rate video cameras are available. This motivates us to develop a hybrid camera that consists of a high-frame-rate video camera and a low-frame-rate depth camera and to allow temporal interpolation of depth maps with the help of auxiliary color images. To achieve this, we develop a novel algorithm that reconstructs intermediate depth maps and estimates scene flow simultaneously. We test our algorithm on various examples involving fast, non-rigid motions of single or multiple objects. Our experiments show that our scene flow estimation method is more precise than a tracking-based method and the state-of-the-art techniques.

Index Terms—Hybrid Camera, Scene Flow Estimation, Depth Upsampling

1 Introduction

In recent years, low-cost depth cameras such as Microsoft Kinect and Intel RealSense have been popular and employed for various computer graphics applications, including motion capture [1], scene reconstruction [2], and image-based rendering [3]. For such cameras, the resolution and speed of depth acquisition are sacrificed to achieve a low cost. For example, the latest Microsoft Kinect depth camera for Xbox One (Kinect V2) is able to capture depth frames with only 512 $\times$ 424 resolution at 30 frames per second (FPS). While such specifications might be sufficient for certain applications, they are not sufficient for applications involving fast motions and higher frame-rate video. On the other hand, with recent advancements in imaging sensors, high-resolution, high-frame-rate and low-cost video cameras such as GoPro have also opened up many possibilities in computer graphics, such as outdoor motion capture [4], structure from motion (SfM) and dynamic hair capture [5].

Video cameras have their advantages over depth cameras in terms of frame rate and resolution. Observing that high-resolution video cameras are cheap and available anywhere, several techniques (e.g., [6], [7]) have been proposed to use a hybrid camera, i.e., a high-resolution video camera and a low-resolution depth camera, to perform spatial upsampling of depth maps. Many applications, such as image-based rendering [8], [9] and image processing [10], can benefit from additional depth information. The Kinect V2 itself is already such a hybrid camera. However, the low-frame-rate capture problem of existing depth cameras is largely unexplored and thus is the focus of our work.

Motivated by the existing hybrid cameras for obtaining the spatial super-resolution of depth maps and the available high-frame-rate, low-cost video cameras, such as GoPro (with 240 FPS), we propose a hybrid camera to achieve temporal upsampling of depth maps (Fig. 1). Our hybrid camera consists of a low-frame-rate depth camera and a synchronized high-frame-rate video camera. The key challenge is to effectively extract fast motion information from color images using a high-frame-rate video camera and then use it to guide the interpolation of depth maps. A straightforward solution is to first compute the 2D optical flow [11] between consecutive images using the high-frame-rate camera and then employ the resulting motion flow to estimate intermediate depth maps between a pair of original depth maps. However, this simple solution works well only for translational motions.

Another possible solution is based on scene flow [12]. However, the traditional methods for scene flow estimation require both color images and depth maps acquired at roughly the same frame rate and thus cannot be directly used for temporal upsampling. To address this problem, we formulate an optimization to estimate the scene flow and intermediate depth maps jointly; the estimated scene flow is used to guide the interpolation of intermediate depth maps, which in turn help refine the scene flow estimation. We derive data constraints from the high-frame-rate color images and enforce spatiotemporal regularization based on the shortest motion path and the locally rigid deformation assumption.

We test our hybrid camera on various examples with quickly moving single or multiple objects and humans.

- *Corresponding Author: Lin Gao (gaolin@ict.ac.cn) and Shihong Xia (xsh@ict.ac.cn)*
- *M.-Z. Yuan is with the Beijing Key Laboratory of Mobile Computing and Pervasive Device, Institute of Computing Technology, Chinese Academy of Sciences, and also with the University of Chinese Academy of Sciences, Beijing, 100190, China. E-mail: yuanmingze@ict.ac.cn.*
- *L. Gao is with the Beijing Key Laboratory of Mobile Computing and Pervasive Device, Institute of Computing Technology, Chinese Academy of Sciences, Beijing, 100190, China. E-mail: gaolin@ict.ac.cn.*
- *H. Fu is with the School of Creative Media, City University of Hong Kong. E-mail: hongbofu@cityu.edu.hk*
- *S. Xia is with the Beijing Key Laboratory of Mobile Computing and Pervasive Device, Institute of Computing Technology, Chinese Academy of Sciences, Beijing, 100190, China. E-mail: xsh@ict.ac.cn.*

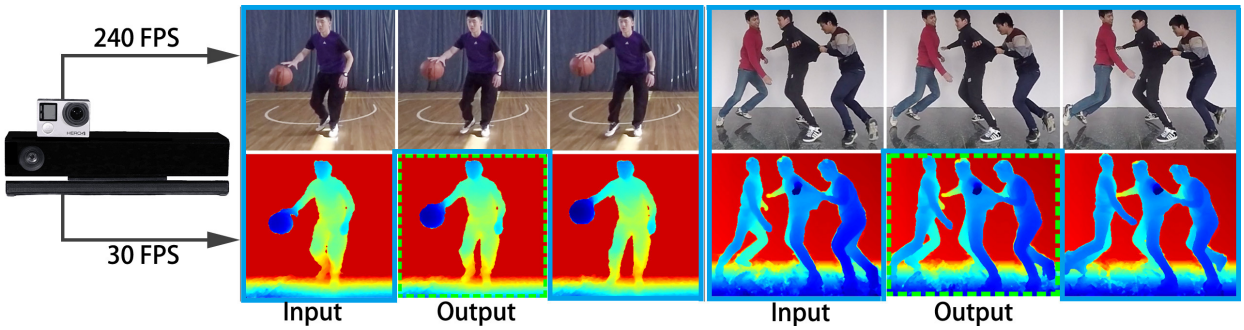


Fig. 1. Our technique obtains the input via a hybrid camera and is able to temporally upsample the depth maps using a low-frame-rate depth camera with the help of the color images taken by a high-frame-rate video camera. Images surrounded by a cyan line and green dotted line represent the input and output of our method, respectively.

In these challenging examples, which possess non-rigid motions and topology changes, our method has clear advantages over a tracking-based method and the state of the art [13]. We show that our joint optimization framework can be reduced for scene flow estimation. Compared to the state-of-the-art scene flow methods [11], [14], [15], our method achieves comparable or even better performance on the MPI Sintel dataset [16] and Middlebury stereo dataset [17].

2 Related Work

Depth cameras are often used together with video cameras to capture RGB-D images. Therefore, the idea of a hybrid camera is not new for 3D imaging. In fact, consumer-level depth cameras such as Kinect V2 are essentially hybrid cameras. It is well known that the depth maps produced by low-cost depth cameras are often noisy and of low resolution. A common approach to enhancing depth maps in the spatial domain is to couple a low-resolution depth map with a high-resolution color image. Various solutions based on optimization (e.g., [6], [18]), joint edge-preserving upsampling filters (e.g., [19], [20], [21]), spatiotemporal filtering (e.g., [7], [13]), or shading cues (e.g., [22], [23]) have been explored to increase the spatial resolution of depth maps. All the above methods assume color images and depth maps with the same frame rates. The exceptional case is the work by Dolson et al. [13]. Our experimental results show that our method is more accurate than [13] in reconstructing intermediate depth frames. Their method estimates the depth of each pixel by using the time, space, and color information but ignores the depth relationship between adjacent pixels. In contrast, our method not only considers the time, space, and color information but also regularizes the relationship between the depth values of adjacent pixels via the locally rigid priori to approximate the relationship of input depth maps as closely as possible. Further discussions are given in Sec. 5.1 and 5.2.

Hybrid cameras have also been used for motion deblurring [24], [25], [26]. For this application, at least one high-speed but often low-resolution video camera is needed to remove motion blur in color images taken by a low-speed, high-resolution camera. Li et al. [25] used two low-resolution, high-speed cameras as a stereo pair to reconstruct a low-resolution depth map, the spatial resolution of which was then enhanced by using joint bilateral filters. In addition to reducing motion blur, the approach of Tai et al. [26] is also used to estimate new high-resolution color images at a higher frame rate. This task of temporal upsampling is similar to ours, but temporal upsampling of depth maps is generally more difficult. Furthermore, the concurrent work of Wang et al. [27] used a hybrid camera system consisting of a 3 FPS light field camera and a 30 FPS video camera to reconstruct 30 FPS light field images. The difference between the work of Wang et al. and our method is that they used a learning-based method and upsampled light field images.

While consumer-level depth cameras are able to capture depth maps at only a limited frame rate, high-speed depth cameras, which already reach hundreds or even thousands of frames per second, have been explored in the fields of computer vision and optical engineering in recent years. Among various solutions, structured light illumination (e.g., [28], [29], [30], [31], [32], [33]) is the most popular technique, which requires a DLP video projector and a synchronized video camera to acquire structured patterns (e.g., fringe images) projected by a special illuminator. Compared with these approaches, our solution can be regarded as a post-processing technique and is thus applicable to different types of depth cameras. Stühmer et al. [34] proposed modifying a typical Time-of-Flight (ToF) camera (e.g., Kinect v2) for model-based tracking at a high frame rate (300 Hz). However, their solution is limited to tracking objects with rigid motion. Our work closely resembles that of Kim and Kim [35], who used multiview hybrid cameras (consisting of eight high-frame-rate video cameras and six ToF cameras) for motion capture. However, their technique is highly dependent on skeleton tracking and thus is suitable only for articulated motion.

Our joint optimization, which is performed to fuse the color and depth information and estimate the motion field, yields a novel scene flow method. Scene flow estimation for depth cameras is a recent active research topic. For example, Herbst et al. [36] extended the Horn-Schunck method [37] to depth cameras with the depth data term for estimating the scene flow from a consumer-level depth camera. Jaimez et al. [14] proposed a total variation regularization term for RGB-D flow estimation in real time. Piecewise rigid motion priors were added to the scene flow estimation in [15]. Jaimez et al. [38] estimated the scene flow with the joint optimization of motion and segmentation. Their method

segments a scene with rigid motions. Sun et al. [11] ordered each depth map into layers and assumed the motion field in a single layer to be within the small range around the mean rigid rotation. When objects in the same depth layer have large and different motions, this method will introduce artifacts (see Sec. 5.1).

As shown in [15], [38], the piecewise rigid regularization term of the motion field enhances the precision compared with methods such as [14], [36]. We follow the as-rigid-as-possible energy to model isometric deformations, which have been demonstrated for various graphics applications, such as shape interpolation [39], shape deformation [40], [41], [42] and 3D shape tracking [43]. In our work, the as-rigid-as-possible energy is employed for the first time for scene flow estimation with the assumption of nearly isometric deformations.

3 Hardware Setup

Our hybrid camera system is composed of two consumer-level cameras, namely, a GoPro HERO 4 video camera and a Kinect V2 RGB-D camera. The GoPro camera captures color images of WVGA resolution at 240 FPS, while Kinect V2 is able to capture depth maps of 512×424 resolution at 30 FPS. As shown in Fig. 1, the GoPro is placed above the Kinect, such that Kinect's depth camera is vertically aligned with the GoPro.

Calibration and alignment. The intrinsic and extrinsic parameters of the GoPro and Kinect V2's depth camera are calibrated by the method of [44]. The lens distortion of the color images from GoPro is corrected according to the intrinsic parameters of GoPro. We transform the depth maps from the depth camera plane to the color camera plane by using the method introduced by Park et al. [6]. After aligning the depth maps with the undistorted color images, we crop the images to retain the part that needs reconstruction. The two cameras are synchronized in the temporal domain by a flashlight. More specifically, we capture the flashlights and then identify the first highlighted images from the color and depth sequences that refer to the same point in time. Finally, we acquire the aligned and synchronized depth maps, color images, and camera intrinsic parameters of the cropped and aligned images, which are denoted by $\mathbf{D}$, $\mathbf{C}$ and $\mathbf{A}$, respectively.

4 Temporal Upsampling of Depth Maps

Our main goal is to temporally upsample the depth maps by estimating a depth map corresponding to each color image from a higher-frame-rate video camera, as illustrated in Fig. 2. The optical flow term is employed to exploit the dense 2D motion information from the color images. To connect between the 3D motion and 2D optical flow, a projection term is employed. We use the popular point-to-point and point-to-plane terms to exploit the start and end positions of the depth maps reconstructed from the two consecutive depth maps captured using Kinect V2 (Sec. 4.2). We also employ regularization terms to enforce the local rigidity and shortest path in the motion flow (Sec. 4.3). Considering occlusion in the motion, we apply occlusion detection to avoid artifacts resulting from the occluded regions (Sec. 4.4).

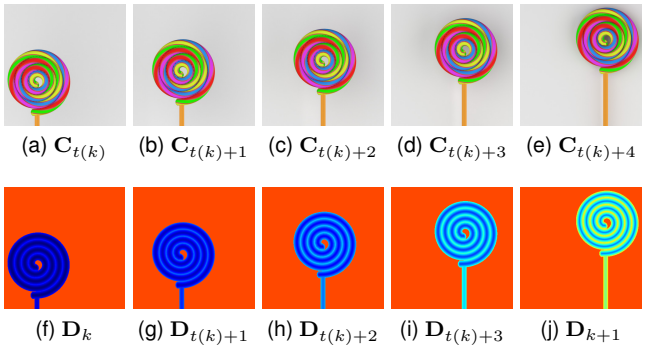


Fig. 2. Our main idea illustrated on synthetic chronological data. Given input color images (a-e) and depth maps (f) and (j), we temporally upsample depth frames by reconstructing intermediate depth maps (g-i).

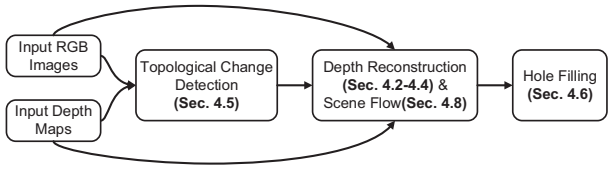


Fig. 3. Pipeline of our system.

The precalculated optical flow is used to detect the topology changes (Sec. 4.5). Moreover, to fill the remaining holes, we use both forward and backward reconstruction and a bilateral filter (Sec. 4.6). We use a joint optimization to determine the above data constraints and spatiotemporal regularization terms (Sec. 4.7). Finally, the framework is reduced to a scene flow method with pairs of color images and depth maps (Sec. 4.8). The pipeline of our system is shown in Fig. 3. All the above components work together to reconstruct the depth maps and estimate the scene flow. The importance of each component will be evaluated in Sec. 5.3.

4.1 Notations

Given a pair of consecutive depth frames $\mathbf{D}_k$ and $\mathbf{D}_{k+1}$, as illustrated in Fig. 2, let $\mathbf{C}_{t(k)}$ denote the color image corresponding to $\mathbf{D}_k$, $\mathbf{C}_{t(k)+g}$ denote the color image corresponding to $\mathbf{D}_{k+1}$, and $\mathbf{C}_{t(k)+s}$ $(s{=}1, \cdots, g{-}1)$ denote the intermediate color images captured between $\mathbf{C}_{t(k)}$ and $\mathbf{C}_{t(k)+g}$. $g{=}4$ for the synthetic example in Fig. 2. Our ultimate goal is thus to reconstruct a depth map $\mathbf{D}_{t(k)+s}$ corresponding to $\mathbf{C}_{t(k)+s}, s \in [1, g{-}1]$. Our underlying optimization will also reconstruct $\mathbf{D}_{t(k)+g}$ to exploit the boundary constraints from $\mathbf{D}_{k+1}$ in depth reconstruction. $\mathbf{D}_{t(k)+g}$ will be replaced with $\mathbf{D}_{k+1}$, the depth information of which is accurate. The point cloud $\mathbf{M}_k$ is generated by projecting $\mathbf{D}_k$ to the 3D space with the intrinsic parameters $\mathbf{A}$. The above and additional notations are summarized in Tab. 1. In our setting of the capture rate (240 FPS) for the GoPro and that (30 FPS) for Kinect V2, the value of g is 8.

4.2 Data Terms

To exploit the motion information from the color images, we employ the optical flow data term, point-to-point term, point-to-plane term and projection term as the motion estimation constraints. To estimate the optical flow at the

TABLE 1
Notations.

Notation	Exposition
k	the index of the captured depth maps
g	the ratio of the color camera's FPS to the depth camera's FPS
$\mathbf{D}_k$, $\mathbf{D}_{k+1}$	the first and second of a pair of consecutive depth maps generated by Kinect V2
$\mathbf{D}_{t(k)+s}$	a depth map to be reconstructed corresponding to $\mathbf{C}_{t(k)+s}$, $1 \leq s \leq g$
$\mathbf{C}_{t(k)+s}$	color images captured at the interval between $\mathbf{D}_k$ and $\mathbf{D}_{k+1}$, $0 \leq s \leq g$, with $\mathbf{C}_{t(k)}$ corresponding to $\mathbf{D}_k$ and $\mathbf{C}_{t(k)+g}$ to $\mathbf{D}_{k+1}$
$\mathbf{p}_{s,i}$	the position of the i-th point in the point cloud corresponding to $\mathbf{D}_{t(k)+s}$; $\mathbf{P} = \{\mathbf{p}_{s,i}\}$
$\mathbf{R}_{s,i}$	a rotation matrix associated with the i-th point in the point cloud corresponding to $\mathbf{D}_{t(k)+s}$; $\mathbf{R} = \{\mathbf{R}_{s,i}\}$
$\mathbf{v}_{s,i}$	optical flow at the i-th pixel in $\mathbf{C}_{t(k)+s}$, $0 \leq s \leq g$; $\mathbf{v}_s$ is the optical flow for $\mathbf{C}_{t(k)+s}$
$\mathbf{M}_k$,$\mathbf{M}_{k+1}$	point cloud generated by $\mathbf{D}_k$ and $\mathbf{D}_{k+1}$

interval between the consecutive depth maps $\mathbf{D}_k$ and $\mathbf{D}_{k+1}$, we use the color images $\mathbf{C}_{t(k)+s}$, $s \in [0, g]$, to recover the 2D motion flow between $\mathbf{D}_k$ and $\mathbf{D}_{k+1}$. The optical flow data term is shown below:

$$\begin{aligned} &E_{opti}(\mathbf{v}_s) \\ &= \lambda_{opti} \sum_{\mathbf{s}\in[0,g-1]} \sum_{\mathbf{x},\mathbf{y}} \rho(\mathbf{C}_{t(k)+s}(\mathbf{x}+\mathbf{v}_{\mathbf{s},\mathbf{x}}, \mathbf{y}+\mathbf{v}_{\mathbf{s},\mathbf{y}}) \\ &- \mathbf{C}_{t(k)+s+1}(\mathbf{x},\mathbf{y})), \end{aligned} \tag{1}$$

where $\mathbf{v}_{\mathbf{s},\mathbf{x}}$ and $\mathbf{v}_{\mathbf{s},\mathbf{y}}$ are the optical flow in the images along the x-axis and y-axis, respectively. λ_{opti} is the weight of the optical flow term and is chosen to be 8 in the implementation. $\rho(r)=\sqrt{r^2+\varepsilon^2}$ is the kernel function that defines the robust metric for addressing noise and outliers (ε=10^{-4} in the implementation) [45], [46]. Furthermore, to improve the optical flow value, we employ the weighted median filter to avoid over-smoothing along object edges [47].

The previous RGB-D scene flow methods [11], [14], [15] use local or global smooth terms to solve the undetermined problem in the depth map field. In this work, we lift the depth similarity constraint to the 3D space where the geometry information can be explored better [2]. The advantage of such a distance metric in the 3D Euclidean space instead of in the depth difference is that not only the local geometric distance but also the surface normal information can be employed to measure the geometric distance. We project depth map $\mathbf{D}_k$ to the 3D space to generate a point cloud $\mathbf{M}_K$. The i-th pixel in $\mathbf{D}_k$ is projected to $\mathbf{p}_i$ in 3D. The connecting relationship between pixel i and its adjacent pixels is created for the spatial coherency.

We employ the following point-to-point term and point-to-plane term [48] to reconstruct the geometry constraints of the depth map $\mathbf{D}_{t(k)+g}$:

$$E_{point}(\mathbf{p}_{g,i}) = \lambda_{point} \sum_{i\in V} \|\mathbf{p}_{g,i} - \tilde{\mathbf{p}}_{k+1,i}\|_2^2, \tag{2}$$

$$E_{plane}(\mathbf{p}_{g,i}) = \lambda_{plane} \sum_{i\in V} \|\mathbf{n}_{k+1,i}^T(\mathbf{p}_{g,i} - \tilde{\mathbf{p}}_{k+1,i})\|_2^2, \tag{3}$$

where $\tilde{\mathbf{p}}_{k+1,i}$ is the closest point to $\mathbf{p}_{k,i}$ in $\mathbf{M}_{k+1}$ and $\mathbf{n}_{k+1,i}$ is the normal vector of $\tilde{\mathbf{p}}_{k+1,i}$. The energy weights for the point-to-point term λ_{point} and point-to-plane term λ_{plane} are set to 9 and 10, respectively. Since the optical flow is essentially the projection of the scene flow introduced by the reconstructed depth maps, the projection function $\psi(\cdot)$ can project the point to the video camera's plane by $\mathbf{A}$ [6]. We model this constraint as follows:

$$\begin{aligned} &E_{proj}(\mathbf{p}_{s,i}, \mathbf{v}_{s,i}) \\ &= \lambda_{proj} \sum_{s\in[1,g-1]} \sum_{i\in V} \mathbf{O}(\mathbf{v}_{s,i}, \mathbf{C}_{t(k)+s}) \|\psi(\mathbf{p}_{s+1,i} - \mathbf{p}_{s,i}) - \mathbf{v}_{s,i}\|_2^2 \end{aligned} \tag{4}$$

and use the projection term to connect the optical flow and 3D point cloud. The energy weight λ_{proj} is set to 5. $\mathbf{O}(\cdot)$ is a function that indicates whether this point is occluded in motion (Sec. 4.4).

4.3 Spatial and Temporal Regularization

With the observation that most of the objects in real-world scenes move in a rigid or locally rigid fashion, we employ the following locally rigid regularization term:

$$\begin{aligned} &E_{rigid}(\mathbf{p}_{s,i}, \mathbf{R}_{s,i}) \\ &= \lambda_{rigid} \sum_{s\in[1,g]} \sum_{i\in V} \sum_{j\in N_i} w_{ij} \|(\mathbf{p}_{s,i} - \mathbf{p}_{s,j}) - \mathbf{R}_{s,i}(\mathbf{p}_{k,i} - \mathbf{p}_{k,j})\|_2^2, \end{aligned} \tag{5}$$

where N_i denotes points connected to the i-th point in the point cloud and λ_{rigid} is set to 16. It is more likely that the connected points with similar depth and color would share similar locally rigid motions. These weights w_{ij} are defined as $w_{c,ij} \cdot w_{d,ij} \cdot w_{t,ij}$, with the depth coherence $w_{d,ij} = \exp(-\|\mathbf{p}_{k,i} - \mathbf{p}_{k,j}\|^2/\sigma_d^2)$, color coherence $w_{c,ij} = \exp(-\|\mathbf{C}_{t(k),i} - \mathbf{C}_{t(k),j}\|^2/\sigma_c^2)$, and topology change $w_{t,ij} = \exp(-\|\mathbf{e}_{i',j'} - \mathbf{e}_{i,j}\|^2/\sigma_t^2)$, where $\mathbf{e}_{i,j}$ and $\mathbf{e}_{i',j'}$ are the Euclidean distance of the corresponding point pair (i,j) in $\mathbf{M}_k$ and warped $\mathbf{M}_k$, respectively (see Tab. 1 and Sec. 4.5). The connection of occlusive points is generally less reliable. When at least one of $\mathbf{p}_i$ and $\mathbf{p}_j$ is occluded or out of the image boundary and thus does not have the corresponding point pair in $\mathbf{M}_{k+1}$, $w_{t,ij}$ is set to 1. In practice, $\sigma_t = 0.015$, $\sigma_c = 1$ and $\sigma_d = 0.015$. The total quadratic variations are employed to regularize the motion field and prevent an artifact being generated in large-scale deformation [49]. This is defined as follows:

$$E_{reg}(\mathbf{R}_{s,i}) = \lambda_{reg} \sum_{s\in[1,g]} \sum_{i\in V} \sum_{j\in N_i} w_{ij} \|\mathbf{R}_{s,i} - \mathbf{R}_{s,j}\|_2^2, \tag{6}$$

where λ_{reg} is set to 0.8.

The temporal regularization term is employed to reduce the uncertainty of the solution and to favor the solution with the shortest path. This term is defined as the sum of the Euclidean distances of the corresponding points in the consecutive point cloud:

$$E_{short}(\mathbf{p}_{s,i}) = \lambda_{short} \sum_{s\in[1,g]} \sum_{i\in V} \|\mathbf{p}_{s,i} - \mathbf{p}_{s-1,i}\|_2^2, \tag{7}$$

where λ_{short} is set to 7.

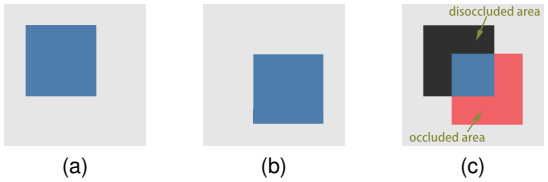


Fig. 4. The blue block in (a) moves to the lower right, resulting in (b). (a) is warped by the optical flow; thus, (c) is generated. The black region in (c) is the disoccluded area of (a), while the red region is the occluded area. The red region in (c) appears in (a) but is covered in (b), while the black area in (c) is present in (b) but is covered in (a).

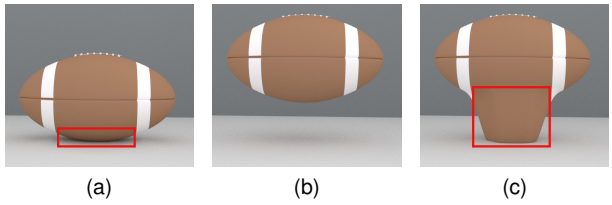


Fig. 5. Topology change detection. The football in (b) is the football in (a) after having bounced from the ground. (a) is warped by 3D motion flow to (c). The red insets of (a) and (c) present the areas in which the topology changes.

4.4 Occlusion Detection

Due to different objects or different parts of the same object moving with different speeds, there often exist occlusions in the camera view. The occlusion leads to three problems.

First, the occlusion results in the projection relationship error in E_{proj}, as the occlusion degrades the relation between the 3D motion flow and the optical flow. To reduce the impact of the mismatch of the projection relation, we adopt the idea in [50] to detect the occlusion based on flow divergence and the pixel difference. The $\mathbf{O}(\mathbf{v}_{s,i}, \mathbf{C}_{t(k)+s})$ function used in Eq. 4 is defined as $\exp(-\|\mathbf{div}(s,i)\|^2/\sigma_1^2) \cdot \exp(-\|\mathbf{C}_{t(k)+s,i} - \mathbf{C}_{t(k)+s+1,i+\mathbf{v}_{s,i}}\|^2/\sigma_2^2)$, where $i + \mathbf{v}_{s,i}$ is the index of the pixel obtained by applying the translation of $\mathbf{v}_{s,i}$ to the index i [51]. $\mathbf{div}(s,i)$ is the divergence of the optical flow. Based on our experience, we set $\sigma_1 = 1$ and $\sigma_2 = 20$. The motion of its connected points will yield the occluded depth pixels without the need for optical flow information.

Second, the occlusion also produces an outlier of E_{opti} due to color pixel mismatch [52]. To address the outlier, we use the robust kernel function $\rho(r)$ in Sec. 4.2. However, the outlier still makes the optical flow over-smoothed at the boundaries [47]. This problem can be further alleviated by applying the weighted media filter [47].

Third, the occlusion generates holes in the reconstructed depth maps. The occluded surfaces are divided into occlusion and disocclusion areas, as illustrated in Fig. 4 [53]. The disoccluded objects can be detected in the reversed timeline. In other words, the portion of the disoccluded objects can be recovered from the corresponding frames in the backward-reconstructed depth maps sequence. We will show how these holes can be filled in Sec. 4.6.

4.5 Topology Change Detection

The connection between neighboring pixels describes an object's topology. Topology changes often occur in scene objects that interact with each other via a clap, rebound or handshake. In such interactions, the separation of points, or the combination of points, is regarded as a topology change. Both cases will change the connection of points at an object boundary.

The topology changes due to the merging of separated objects are solved by the point cloud stitching, with use of the information regarding the optical flow and geometry constraints (Sec. 4.2). We detect the topology changes of separating objects by computing the distance change, with respect to the adjacent points, between the point cloud and the warped one. The latter is obtained by warping the original point cloud using the rough motion flow in 3D space from $\mathbf{D}_k$ to $\mathbf{D}_{k+1}$, as shown in Fig. 5. To estimate the proximate motion flow, we project the optical flow that is accumulated from $\mathbf{C}_{t(k)}$ to $\mathbf{C}_{t(k)+g-1}$ to 3D space with $\mathbf{A}$ and $\mathbf{D}_k$. To keep the geometric information of the warped point cloud consistent with that of the original one, we use E_{point}, E_{plane}, and E_{rigid} to obtain a coarse result. As mentioned in Sec. 4.3, we define a weight of topology change $w_{t,ij}$ to represent topology changes between a pair of points (i,j).

4.6 Hole Filling

To reconstruct the depth maps more completely, we fill the holes in the depth maps. Holes occur due to either the occlusion or the imaging principle of a depth camera. In Fig. 6 (h), the cyan pixels exemplify the first type of holes, which are caused by occlusion. The yellow pixels exemplify the second type of holes, which natively exist in the input depth maps. We use different approaches to fill these two kinds of holes. The workflow is shown in Fig. 6.

To address the first type of holes, we use the forward and backward reconstruction information together [54]. During the forward reconstruction, the depth data of the occluded portion in the initial frame are also missing in the next $g-1$ forward reconstructed depth maps. The occluded part of the reconstructed depth data in the current frame is thus the accumulated occluded part of the previous depth frames in the current forward reconstructed sequence. On the other hand, the occluded part of the forward reconstruction is the disoccluded part of backward reconstruction. Thus, the missing depth information of forward reconstructed maps can be recovered from the backward reconstruction depth maps. By comparing the final depth frame of forward reconstruction ($\mathbf{D}_{t(k)+g}$) with the initial depth map of backward reconstruction ($\mathbf{D}_{k+1}$), we can obtain the corresponding relationship between the disocclusive depth data and the pixels in the backward reconstructed depth map and thus fill in the holes of this type, as shown in Fig. 6(i).

The second type of holes appears due to two main reasons: imperfect alignment of the depth and color images and environmental interference, such as hair, glare, motion blur and the reflectivity of objects. Some of the holes of this second type can also be filled using forward- and backward-reconstructed depth maps. This is because the missing depth data of the second type of holes can be obtained from adjacent depth maps by Kinect V2. We use the bilateral filter,

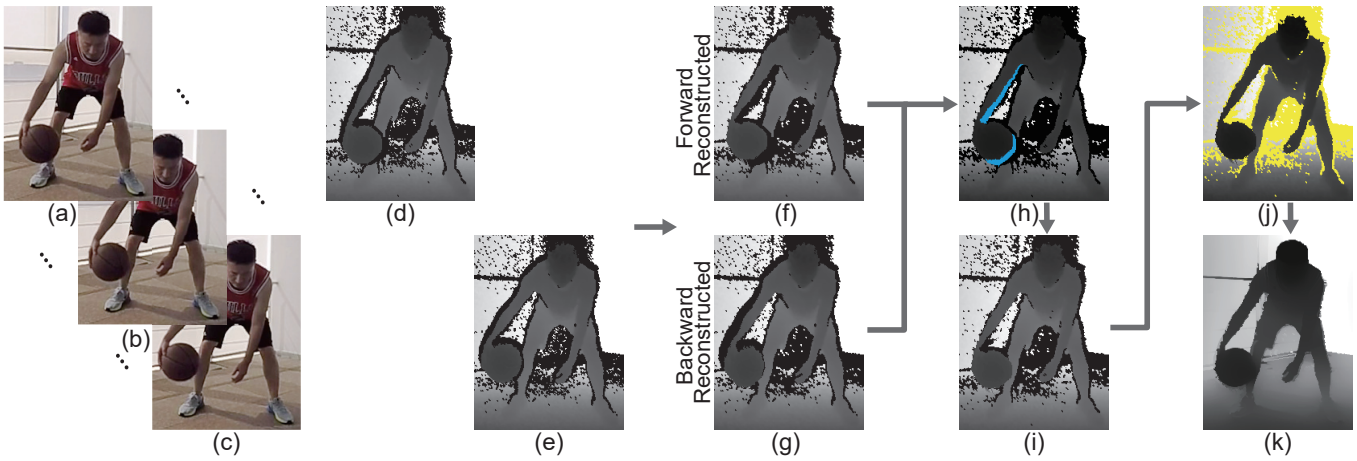


Fig. 6. Hole filling. (a-c) are the input color images. (d) and (e) are the input depth maps corresponding to (a) and (c), respectively. (f) is the forward-reconstructed depth map corresponding to (b), and (g) is the backward-reconstructed result. Holes exist in the reconstructed depth maps (f) and (g) due to either occlusion or the imaging principles. In (h), the pixels in cyan represent the first type of holes generated by occlusion. (i) is the depth map fixed using forward- and backward-reconstructed depth maps. The yellow pixels of (j) are the second type of holes, which occur due to the imaging artifacts in (i). (k) is the result after filling all the holes.

with the help of the color information, to fill in the residual missing depth data [20].

4.7 Energy Minimization

At every interval between two consecutive Kinect V2 captured depth maps, we reconstruct the intermediate depth maps by optimizing the following global energy Eq. 8, which consists of the energy terms introduced in the previous section:

$$\begin{aligned} & E(\mathbf{P},\mathbf{V},\mathbf{R}) \\ &= E_{opti}(\mathbf{v}_s) + E_{point}(\mathbf{p}_{g,i}) + E_{plane}(\mathbf{p}_{g,i}) \\ &\quad + E_{proj}(\mathbf{p}_{s,i},\mathbf{v}_{s,i}) + E_{rigid}(\mathbf{p}_{s,i},\mathbf{R}_{s,i}) \\ &\quad + E_{reg}(\mathbf{R}_{s,i}) + E_{short}(\mathbf{p}_{s,i}). \end{aligned} \tag{8}$$

This equation can be rewritten as the following summation of squared residues: $\mathbf{E}(x)=\sum_i g_i(\mathbf{x})^2=\|\mathbf{g}(\mathbf{x})\|_2^2$, where $\mathbf{x}$ is a vector of all variables and $\mathbf{g}(\mathbf{x})$ is a vector function, with its element function denoted by $g_i(\mathbf{x})$. Minimizing $\mathbf{E}(x)$ is a least squares problem. The Gauss-Newton method is employed to solve this optimization [55]. In the k-th Gauss-Newton iteration, we update the variables according to $\mathbf{x}_{k+1}=\mathbf{x}_k+\Delta\mathbf{x}$. $\Delta\mathbf{x}$ is satisfied by the equation $\mathbf{J}^T(\mathbf{x}_k)\mathbf{J}(\mathbf{x}_k)\Delta\mathbf{x}=-\mathbf{J}^T(\mathbf{x}_k)\mathbf{g}(\mathbf{x}_k)$, where $\mathbf{J}(\mathbf{x}_k)$ is the Jacobian matrix at $\mathbf{x}_k$. To solve these linear equations, we apply the preconditioned conjugate gradient (PCG) solver with several CUDA kernels [43].

The optical flow is initialized by the GPU-based method [56]. In this optimization, we use three hierarchical levels. During the prolongation from the coarse level to the fine level, the bilinear interpolation is applied to the optical flow and point cloud. The rigid rotation in $\mathbf{E}_{rigid}$ remains the same in the corresponding position in the coarse levels.

4.8 Scene Flow Estimation

This joint optimization framework will introduce the depth maps and the scene flow simultaneously. If our input is a set of color images, each of which has its corresponding depth map, our method reduces to a standard scene flow method. In Sec. 5.1, we give quantitative evaluations of our scene flow method and show its advantage over the current state-of-the-art works [11], [14], [15].

5 Results and Discussions

In this section, we make both qualitative and quantitative evaluations of our approach. Our technique was tested on real-world complex scenes such as those depicting basketball & games. There are many challenges in real-world scenes, including occlusion, topology changes and moving cameras. These challenges are demonstrated in different cases in the following sections. From the visual results, it is clear that our method performs better than the state-of-the-art methods. These real-world scenes lack the ground truth of scene flow and depth maps. To perform the quantitative evaluations, the MPI Sintel dataset [16] and Middlebury stereo dataset [17] with ground truth are employed.

Performance. The performance of our hybrid camera system was measured on an Intel Xeon E5-2520 CPU with 32 GB of RAM and a single NVIDIA Titan X. Between every pair of successive depth frames, we reconstructed the missing depth maps (8 frames in total) according to the color images taken by the GoPro. This whole process took approximately 315.6 seconds on average for all the tested sequences. At the coarse, medium, and fine levels, the duration of an iteration was approximately 2 seconds, 3 seconds, and 15 seconds, respectively. At the coarse level, the total time of more than ten iterations was 30 seconds. This time was 45 seconds for the medium level and 255 seconds for the fine level. All three levels included 5 Gauss-Newton iterations (within each iteration, there were 10 PCG iterations). The average time for reconstructing one depth frame was 39.5 seconds.

Parameters. We fine tune the optimal values of our important parameters via a quantitative analysis, as shown in Fig. 7. The x-axis is the range of each parameter, and the y-axis is the average accuracy for different input x values of all the sequences of the MPI Sintel dataset [16]

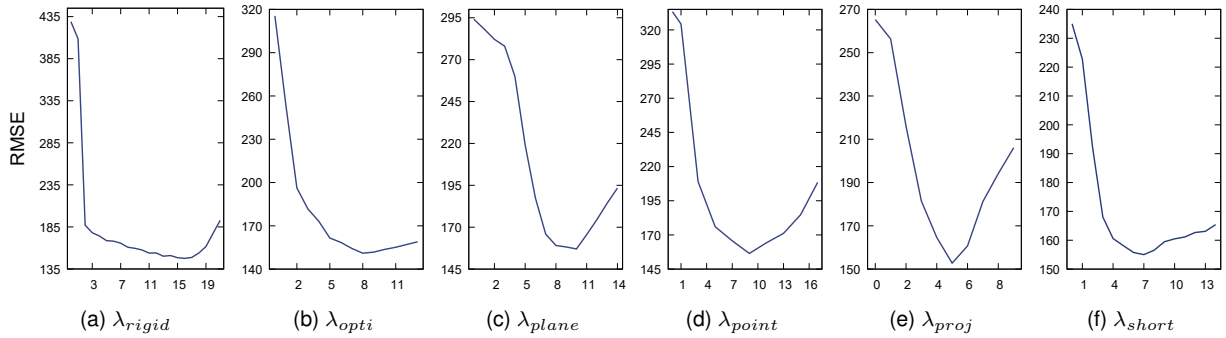


Fig. 7. The root-mean-square error (**RMSE**) of the reconstructed depth maps against different parameter settings in all the sequences of the MPI Sintel [16] and Middlebury stereo datasets [17]. The fixed default values of the weight parameters were selected as follows: λ_{rigid}=16, λ_{opti}=8, λ_{plane}=10, λ_{point}=9, λ_{proj}=5 and λ_{short}=7.

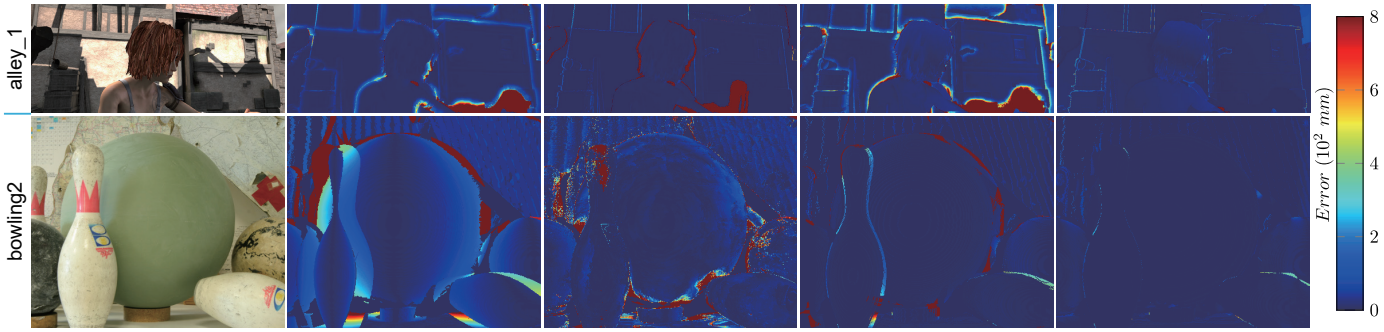


Fig. 8. Visualization of errors between reconstructed depth maps and ground truth. Our method produces more accurate depth maps.

and the Middlebury stereo dataset [17]. We used the fixed default parameter value in all the comparative experiments in Sec. 5.

5.1 Quantitative Evaluation

Depth reconstruction. We compare our method with a tracking-based method and the method of Dolson et al. [13]. The tracking-based method exploits the motion information from depth maps with E_{point}, E_{plane}, and E_{rigid} and regularizes with E_{reg} and E_{short} but does not employ E_{opti} or E_{proj} to take advantage of the color images (Sec. 4.2, 4.3). The post-process, solver (Sec. 4.7) and the weights of individual terms are the same as those of our method. Meanwhile, we also make a comparison with the method by removing components for occlusion detection and topology change detection. In the hole filling post-process, the holes generated by occlusion are not filled. The state-of-the-art work [13] is, to the best of our knowledge, the only existing method for the temporal upsampling of depth maps given inputs similar to ours. Their method also uses the color information to interpolate the depth maps to recover more depth information. In the original datasets, each synthetic color image has its corresponding depth map. To evaluate the ability of reconstruction, we temporally downsample the depth frames, resulting in inputs similar to those from our hybrid camera. To evaluate the precision, we compute the **RMSE** of the reconstructed depth maps against the ground truth, as summarized in Tab. 2. The results show that the reconstructed depth maps of our method are more accurate than those of others. This advantage can be seen based on the visualized errors in Fig. 8.

TABLE 2
Quantitative evaluation of the depth reconstruction on the MPI Sintel dataset [16] and Middlebury stereo dataset [17]. As shown in this table, our method achieves the highest accuracy.

RMSE	Dolson et al. [13]	Tracking-Based Method	W/O occlusion detection and topology change detection	Ours
bamboo_1	614.3	784.4	263.33	**238.3**
alley_1	420.3	1048.5	403.05	**61.8**
sleeping_2	81.35	79.39	87.54	**26.1**
bandage_1	54.44	87.84	45.12	**8.3**
wood1	159.29	273.85	88.23	**81.93**
bowling2	482.82	433.17	174.23	**155.92**

Scene flow estimation. As discussed in Sec. 4.8, our joint optimization method can be used to estimate a scene flow given a set of input color images and corresponding depth maps captured at the same frame rate. We evaluate this scene flow method on the MPI Sintel dataset and Middlebury stereo dataset. Our approach is compared with the state-of-the-art techniques for scene flow estimation, including *Layered-Flow* [11], *SR-Flow* [15] and *PD-Flow* [14]. The **RMSE**, average angular error (**AAE**), and end point error (**EPE**) are used as the error metrics [15]. The quantitative

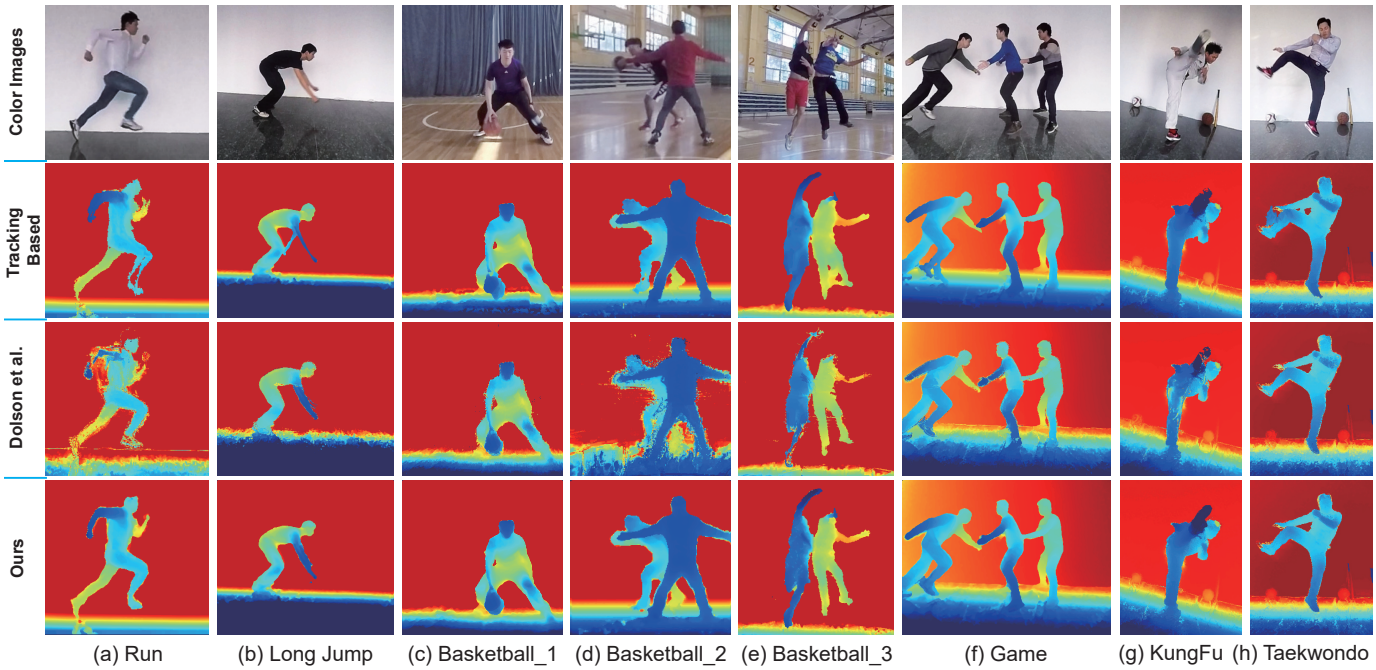


Fig. 9. Reconstructed depth maps of real-world scenes. From top to bottom, the rows present the following: input color images, depth maps reconstructed by the tracking-based method, results of the method of Dolson et al. [13], and results of our method, respectively.

evaluation results are given in Tab. 3. The numerical results show that under all the error metrics and for almost all the sequences, our method outperforms the other scene flow methods and produces results closer to the ground truth.

TABLE 3
Quantitative evaluation of scene flow estimation on the MPI Sintel dataset [16] and Middlebury stereo dataset [17]. The lower the error values, the better the performance.

EPE	PD-Flow	SR-Flow	Layered-Flow	Ours
alley_1	9.07	2.76	1.01	**0.13**
ambush_5	34.1	4.59	2.73	**0.84**
cave_2	147.08	26.83	11.67	**2.38**
market_2	53.22	5.20	4.21	**0.82**
wood1	49.87	9.72	10.09	**0.50**
bowling2	109.60	10.82	5.84	**0.54**
AAE	PD-Flow	SR-Flow	Layered-Flow	Ours
alley_1	1.53	**0.98**	2.10	1.85
ambush_5	1.43	1.19	1.11	**0.74**
cave_2	1.60	1.44	1.57	**1.30**
market_2	0.99	0.75	1.24	**0.28**
wood1	1.12	1.25	1.44	**0.03**
bowling2	1.30	1.37	0.21	**0.02**
RMSE	PD-Flow	SR-Flow	Layered-Flow	Ours
alley_1	10.56	4.03	3.39	**0.52**
ambush_5	71.65	5.98	5.21	**2.01**
cave_2	235.49	29.81	14.86	**5.93**
market_2	73.03	9.36	7.48	**2.96**
wood1	125.44	10.09	30.20	**0.77**
bowling2	356.65	10.75	18.56	**1.86**

5.2 Qualitative Evaluation

In this subsection, we will evaluate our method on the data of real-world scenes and compare it to the technique of Dolson et al. [13] and the tracking-based method. We have made the comparisons on multiple challenging examples (see the accompanying video). Fig. 9 gives representative

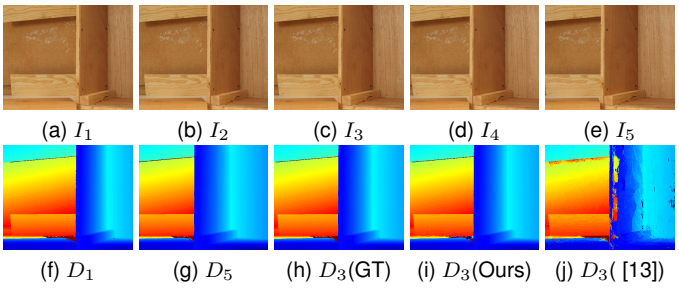
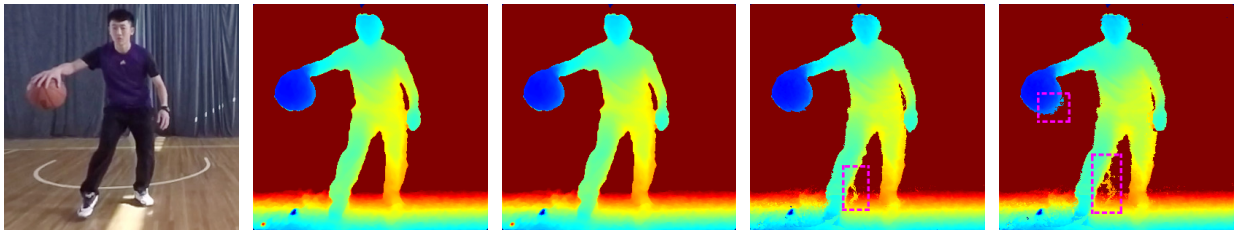


Fig. 10. Results for the frames of sequence $wood1$ in the Middlebury stereo dataset [17]. (a)-(e) are input color images. (f) and (g) are input depth maps. (h) is the ground truth (GT) depth map corresponding to (c). (i) and (j) are the respective depth maps reconstructed by our method and [13] corresponding to (c).

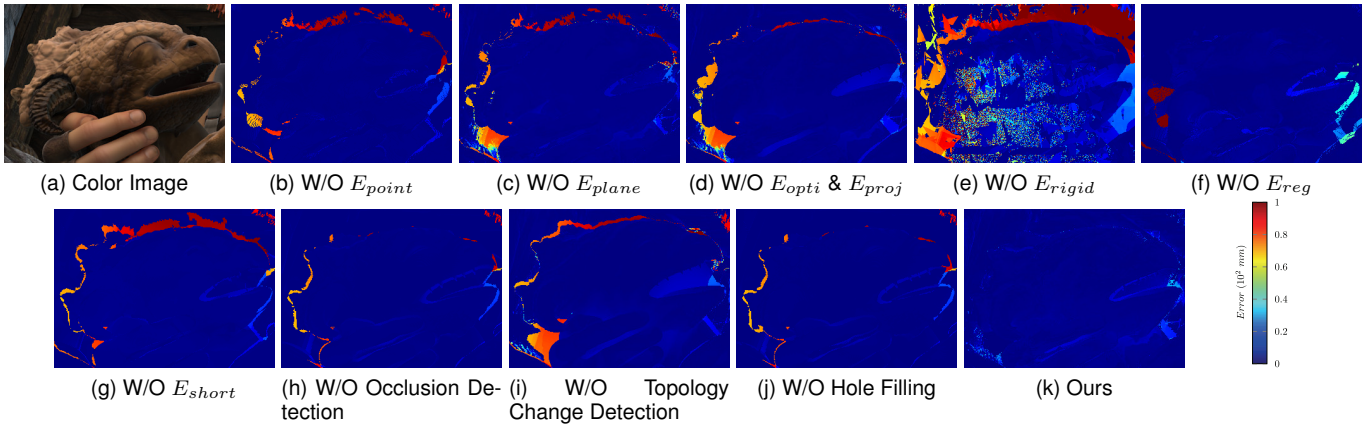
results. Since the tracking-based method does not take the color information into account, it may fail to reconstruct the depth information for a scene with fast motion. In contrast, our method employs the color information to evaluate the motion flow information, thus obtaining more accurate correspondence across frames.

The method of Dolson et al. [13] employs a d-dimensional Gaussian filtering framework to interpolate depth maps by encoding color, time, depth and location. However, their method does not take geometry into account, which is important to maintain the structures of objects. Without the constraint of this geometry prior, their method will easily choose incorrect reference depth data to interpolate intermediate depths. As shown in Fig. 9, their method causes artifacts with serious noise, as it patches the depth maps of foreground and background objects with the background, foreground or invalid data without the restraint of geometry. This problem is obvious in Fig. 9(d) and (e), as



(a) Color Image (b) 1:8 (c) 1:16 (d) 1:24 (e) 1:32

Fig. 11. Downsampling results. (a) is the color image corresponding to the reconstructed depth map. (b) (c) (d) and (e) are reconstructed depth maps with frame-rate ratios of the depth maps generated by Kinect V2 to the color images taken by the GoPro of 1:8, 1:16, 1:24, and 1:32, respectively.



(a) Color Image (b) W/O E_{point} (c) W/O E_{plane} (d) W/O E_{opti} & E_{proj} (e) W/O E_{rigid} (f) W/O E_{reg}

(g) W/O E_{short} (h) W/O Occlusion Detection (i) W/O Topology Change Detection (j) W/O Hole Filling (k) Ours

Fig. 12. An evaluation of the importance of individual components. (a) is an input color image whose corresponding depth map needs to be reconstructed. (b)-(j) are the visualizations of error (the difference between the reconstructed depth maps and the ground truth) in our ablation study.

the gym is so deep (more than 6 meters) that the depth difference between the foreground and the background is very large. Our approach achieves much better results, mainly because of our careful consideration of the color (for optical flow), space and time information, as well as the topology and geometric relationships. As shown in another example in Fig. 10, when the colors of the foreground and background are similar, without geometry regularization, the approach of Dolson et al. [13] easily produces artifacts.

The frame-rate ratio of the depth maps generated by Kinect V2 to the color images taken by the GoPro is 1:8. We also tested our method on 1:16, 1:24 and 1:32 ratios, which are simulated by extracting the Kinect's depth maps at intervals of 2, 3 and 4 frames, respectively. It is expected and shown in Fig. 11 that a higher ratio will lead to more artifacts (e.g., on the legs in this example). These artifacts are mainly due to the errors of topology change detection caused by the cumulative error of optical flow.

5.3 Impact of Individual Components

To evaluate the importance of each term, we conduct an ablation study of our framework. We evaluate using representative frames from the MPI Sintel dataset, which involve topology change, occlusion, and rigid and non-rigid transformation. In Fig. 12, we visualize the errors against the ground truth when one component is removed. Moreover, we quantify the results using the **RMSE** between the reconstructed depth maps and ground truth, as shown in Tab. 4. Our method with all the components achieves the highest accuracy. It is shown that each component of our method is important, with E_{rigid} being the most important term.

Importance of E_{point} and E_{plane}. These two terms take both corresponding point-pairs across the point clouds and the normal of a point cloud into account. Dropping either of them increases the reconstruction error, as shown in (b) and (c) of Fig. 12.

Importance of E_{opti} and E_{proj}. E_{opti} takes full advantage of 2D motion information from color images. E_{proj} is used to connect 2D optical flow with 3D point cloud movement. As shown in Fig. 12(d), when they are omitted, there are more artifacts in the reconstructed depth map. This is mainly because there are more mismatched nearest point relations in E_{point} and E_{plane} when the optical flow information is not used.

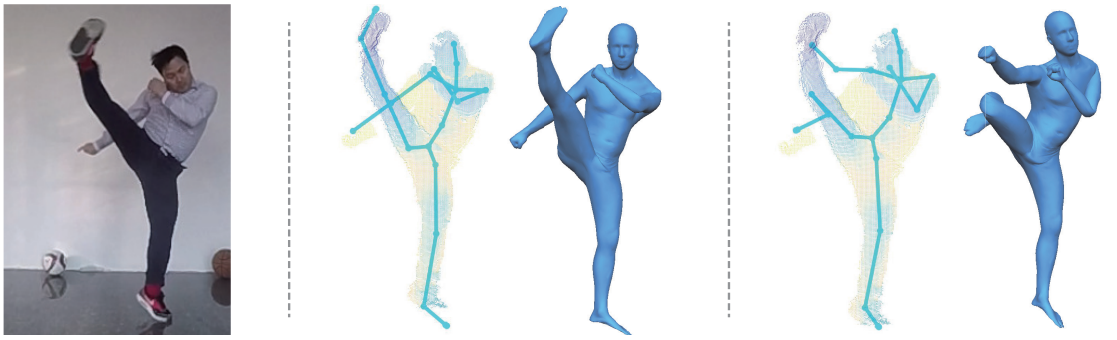
Importance of E_{rigid}. This term is used to regularize the motion such that it is as locally rigid as possible. Since the local rigidity of the motion is very common in real-world scenes, this term plays a central role, as evidenced by the most serious errors in Fig. 12(e).

Importance of E_{reg}. E_{reg} is employed to prevent the artifact generated in large-scale deformation [49]. The result without E_{reg} is shown in Fig. 12(f).

Importance of E_{short}. This term is based on a temporal prior that enforces temporal smoothness and penalizes the jitter of a point cloud over time. Based on the observation that the speed of an object is essentially constant over a very short time, the E_{short} term constrains the solution to a

TABLE 4
Quantitative evaluation of component importance on the MPI Sintel dataset [16] and Middlebury stereo dataset [17]. Depth maps reconstructed by our full components method achieve the lowest **RMSE**.

RMSE	W/O E_{rigid}	W/O E_{opti} & E_{proj}	W/O E_{plane}	W/O E_{point}	W/O E_{reg}	W/O E_{short}	W/O Occlusion Detection	W/O Hole Filling	W/O Topology Change Detection	With all the components
alley_1	901.46	784.4	429.72	379.45	340.14	315.84	323.41	316.46	253.59	**238.3**
wood1	357.8	344.4	215.32	99.21	101.2	103.44	99.17	95.18	83.46	**81.93**
bowling2	432.91	461.64	339.73	230.73	347.07	279.71	194.48	181.69	169.76	**155.92**



(a) Color Images (b) With reconstructed high-frame-rate (240 FPS) depth maps (c) With Kinect-captured (30 FPS) depth maps

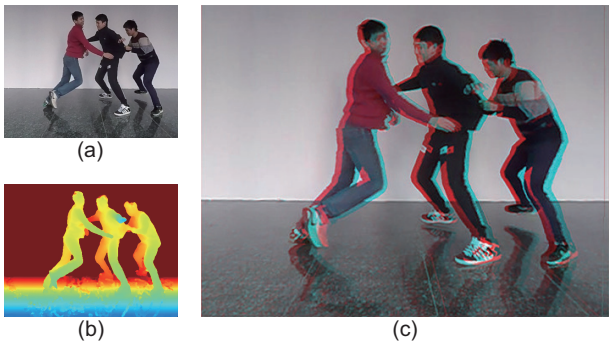
Fig. 13. Human motion capture. (a) is a color image taken by the GoPro. The corresponding point cloud and captured pose are shown in (b) and (c), respectively, which are captured from reconstructed high-frame-rate depth maps and Kinect-captured depth maps, respectively. The meshes in (b) and (c) are reconstructed via linear blending skinning (LBS) based on the currently captured 3D skeletal pose. The mesh in (b) is more reasonable than that in (c) and closer to the pose of the character in (a).

lower-dimensional subspace. The error against the ground truth is shown in Fig. 12(g).

Importance of Occlusion Detection. Occlusion will cause a mismatched correspondence relationship of E_{proj}, causing outliers of the optical flow and creating holes at the edge of the object. To disable the occlusion detection, we set $\mathbf{O}(\mathbf{v}_{s,i}, \mathbf{C}_{t(k)+s})$ to 1 (Sec. 4.1) and do not fill the holes caused by occlusion. Fig. 12(h) and (j) are almost the same, as the holes are generated by occlusion.

Importance of Topology Change Detection. As shown in Fig. 12(i), when topology changes are not considered, obvious artifacts occur on the horn and mouth of the dragon.

Importance of Hole Filling. The hole-filling post-processing patches the invalid data in depth maps generated by occlusion and Kinect imaging. The dominant invalid depth data in the synthetic data is generated by occlusion. If we do not carry out hole filling, clear artifacts occur around the moving objects, as shown in Fig. 12(j).



(a) (b) (c)

Fig. 14. Stereoscopic video rendering based on reconstructed depth maps and color images taken by the GoPro. (a) is the color image taken by the GoPro, (b) is the reconstructed depth map, and (c) is the stereoscopic image rendered by the method of de Albuquerque Azevedo et al. [57].

5.4 Applications

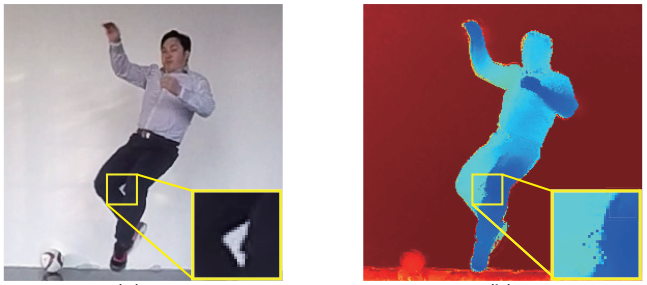
Various applications could benefit from our system with its reconstructed high-frame-rate depth maps. Here, we demonstrate three applications: fast human motion capture, rendering of stereoscopic images for a VR environment and the depth-of-field effect.

Human Motion Capture. The state-of-the-art human motion capture studies include [1], [58], [59]. In those works, the input depth maps were generated by a Kinect at 30 FPS. To capture fast human motion, we make use of the depth maps from our system. We apply the full-body motion capture algorithm [58] to the depth maps from our hybrid system to capture the 3D skeletal poses of the fast human movement. As shown in Fig. 13, thanks to the high frame rate and accurate depth maps of our hybrid system, the capturing algorithm performs better than when using the input from the Kinect directly.

Stereoscopic Image Rendering. Stereoscopic videos provide more immersive experiences for virtual reality, such as 3DTV and head-mounted display. There are several methods that employ depth data to generate and edit stereoscopic images [60], [61], [62]. In our system, a stereoscopic video can be easily acquired from the captured color images and reconstructed depth maps via a depth image-based rendering (DIBR) [57]. DIBR is a technology that synthesizes virtual views of a scene using monocular color images and depth maps. With the help of DIBR, we synthesize a color

(a) Color Image (b) Focus on Players (c) Focus on Background

Fig. 15. Depth-of-field effect: (a) is the entire sharp color image captured using the GoPro. To isolate a player from the background, (b) and (c) render with a small depth-of-field, causing the background and the players to be out of focus, respectively.



(a) (b)

Fig. 16. Failure case. (a) is an input color image, and (b) is the corresponding reconstructed depth map. Our method fails to reconstruct depth data in the highlighted yellow box because of occlusion and fast motion.

video for the left eye and use the original color video for the right eye. Fig. 14 shows an example of the resulting stereoscopic images (the method of de Albuquerque Azevedo et al. [57]). Please also find the rendered stereoscopic video in the accompanying video.

Depth-of-Field (DoF). The DoF effect, which makes some objects in an image acceptably sharp and others blurry, is a common technique in photography used to emphasize a subject. Meanwhile, DoF is a render effect that is commonly applied to images or animation [63]. The usual method for producing a DoF effect in captured images is to use light field cameras, which capture images at a rate of three frames per second and are expensive. We use an image-based algorithm to render the DoF effect in RGB-D images. With the help of a reconstructed depth map, we can easily simulate the DoF effect and carry out refocusing in color images acquired with fast frame rates [10]. We implement the method proposed by Kraus et al. [3] based on sub-images to change the DoF of the acquired images. As shown in Fig. 15, with the help of a reconstructed depth map, we render the DoF effect in GoPro-captured color images and refocus on players and the background separately to emphasize different subjects.

5.5 Limitations and Future Work

Our work has taken the first step in addressing the interesting issues of hybrid cameras in a temporal domain. Our technique can be improved in multiple aspects. First, our current unoptimized implementation is still too slow to support real-time performance capture. The bottleneck of our program is the transmission of data from the CPU to the GPU. We will completely implement the algorithm with CUDA to reduce the overhead of transfer and improve the throughput. Meanwhile, our framework reconstructs the sequence of depth maps together, using $\mathbf{D}_{k+1}$ to reconstruct previous depth frame $\mathbf{D}_{t(k)+s}$. The reconstructed result has a delay of $3 \sim 4$ milliseconds, even if the framework achieves real-time performance.

Finally, our system does not reconstruct depth faithfully when motion is too fast and occlusion is too serious. As shown in Fig. 16, in the gap between the two legs, where the occlusion is serious, and when the leg motion is very fast, our reconstruction result suffers from errors. The less accurate depth is due to a hole filling process error (Sec. 4.6), as there is too little depth information regarding the gap. This problem can be mitigated by using more powerful hole filling methods, e.g., data-driven-based method [64] and deep-learning-based method [65].

Acknowledgments

This work was supported by grants from the National Natural Science Foundation of China (No.61502453, No.61772499 and No.61611130215), Royal Society-Newton Mobility Grant (No. IE150731), the Science and Technology Service Network Initiative of Chinese Academy of Sciences (No. KFJ-STS-ZDTP-017), the Knowledge Innovation Program of the Institute of Computing Technology of the Chinese Academy of Sciences under Grant No.ICT20166040, the Hong Kong Research Grants Council (No.CityU CityU 11237116) and ACIM-SCM.

References

[1] Y. Chen, Z.-Q. Cheng, C. Lai, R. R. Martin, and G. Dang, "Realtime reconstruction of an animating human body from a single depth camera," *IEEE Transactions on Visualization and Computer Graphics*, vol. 22, no. 8, pp. 2000–2011, 2016.

[2] K. Chen, Y.-K. Lai, and S.-M. Hu, "3d indoor scene modeling from rgb-d data: a survey," *Computational Visual Media*, vol. 1, no. 4, pp. 267–278, 2015.

[3] M. Kraus and M. Strengert, "Depth-of-field rendering by pyramidal image processing," *Computer Graphics Forum*, vol. 26, no. 3, pp. 645–654, 2007.

[4] T. Shiratori, H. S. Park, L. Sigal, Y. Sheikh, and J. K. Hodgins, "Motion capture from body-mounted cameras," *ACM Transactions on Graphics*, vol. 30, no. 4, p. 31, 2011.

[5] Z. Xu, H.-T. Wu, L. Wang, C. Zheng, X. Tong, and Y. Qi, "Dynamic hair capture using spacetime optimization," *ACM Transactions on Graphics*, vol. 33, p. 6, 2014.

[6] J. Park, H. Kim, Y.-W. Tai, M. S. Brown, and I. Kweon, "High quality depth map upsampling for 3d-tof cameras," in *Proceedings of the IEEE International Conference on Computer Vision*. IEEE, 2011, pp. 1623–1630.

[7] C. Richardt, C. Stoll, N. A. Dodgson, H.-P. Seidel, and C. Theobalt, "Coherent spatiotemporal filtering, upsampling and rendering of rgbz videos," *Computer Graphics Forum*, vol. 31, no. 2pt1, pp. 247–256, 2012.

[8] J. Lee, Y. Kim, S. Lee, B. Kim, and J. Noh, "High-quality depth estimation using an exemplar 3d model for stereo conversion," *IEEE Transactions on Visualization and Computer Graphics*, vol. 21, no. 7, pp. 835–847, 2015.

[9] P. Bhat, C. L. Zitnick, N. Snavely, A. Agarwala, M. Agrawala, M. Cohen, B. Curless, and S. B. Kang, "Using photographs to enhance videos of a static scene," in *Proceedings of the 18th Eurographics conference on Rendering Techniques*. Eurographics Association, 2007, pp. 327–338.

[10] F. Moreno-Noguer, P. N. Belhumeur, and S. K. Nayar, "Active refocusing of images and videos," *ACM Transactions on Graphics*, vol. 26, no. 3, p. 67, 2007.

[11] D. Sun, E. B. Sudderth, and H. Pfister, "Layered rgbd scene flow estimation," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*. IEEE, 2015, pp. 548–556.

[12] S. Vedula, P. Rander, R. Collins, and T. Kanade, "Three-dimensional scene flow," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 3, pp. 475–480, 2005.

[13] J. Dolson, J. Baek, C. Plagemann, and S. Thrun, "Upsampling range data in dynamic environments," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*. IEEE, 2010, pp. 1141–1148.

[14] M. Jaimez, M. Souiai, J. Gonzalez-Jimenez, and D. Cremers, "A primal-dual framework for real-time dense rgb-d scene flow," in *Proceedings of The International Conference on Robotics and Automation*. IEEE, 2015, pp. 98–104.

[15] J. Quiroga, T. Brox, F. Devernay, and J. Crowley, "Dense semi-rigid scene flow estimation from rgbd images," in *Proceedings of the European Conference on Computer Vision*. Springer, 2014, pp. 567–582.

[16] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black, "A naturalistic open source movie for optical flow evaluation," in *Proceedings of the European Conference on Computer Vision*. Springer, 2012, pp. 611–625.

[17] D. Scharstein and R. Szeliski, "High-accuracy stereo depth maps using structured light," in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, June 2003, pp. 195–202 vol.1.

[18] J. Diebel and S. Thrun, "An application of markov random fields to range sensing," in *Proceedings of Advances in Neural Information Processing Systems*, 2006, pp. 291–298.

[19] Q. Yang, R. Yang, J. Davis, and D. Nistér, "Spatial-depth super resolution for range images," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*. IEEE, 2007, pp. 1–8.

[20] D. Chan, H. Buisman, C. Theobalt, and S. Thrun, "A noise-aware filter for real-time depth upsampling," in *Workshop on Multi-camera and Multi-modal Sensor Fusion Algorithms and Applications*, A. Cavallaro and H. Aghajan, Eds., Marseille, France, 2008, pp. 1–12.

[21] Y. Song, D.-W. Shin, E. Ko, and Y.-S. Ho, "Real-time depth map generation using hybrid multi-view cameras," in *Signal and Information Processing Association Annual Summit and Conference*. IEEE, 2014, pp. 1–4.

[22] C. Wu, M. Zollhöfer, M. Niessner, M. Stamminger, S. Izadi, and C. Theobalt, "Real-time shading-based refinement for consumer depth cameras," *ACM Transactions on Graphics*, vol. 33, no. 6, p. 200, 2014.

[23] M. Zollhöfer, A. Dai, M. Innmann, C. Wu, M. Stamminger, C. Theobalt, and M. Nießner, "Shading-based refinement on volumetric signed distance functions," *ACM Transactions on Graphics*, vol. 34, no. 4, p. 96, 2015.

[24] S. K. Nayar and M. Ben-Ezra, "Motion-based motion deblurring," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 6, pp. 689–698, 2004.

[25] F. Li, J. Yu, and J. Chai, "A hybrid camera for motion deblurring and depth map super-resolution," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*. IEEE, 2008, pp. 1–8.

[26] Y.-W. Tai, H. Du, M. S. Brown, and S. Lin, "Correction of spatially varying image and video motion blur using a hybrid camera," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 6, pp. 1012–1028, 2010.

[27] T.-C. Wang, J.-Y. Zhu, N. K. Kalantari, A. A. Efros, and R. Ramamoorthi, "Light field video capture using a learning-based hybrid imaging system," *ACM Transactions on Graphics*, vol. 36, no. 4, p. 133, 2017.

[28] S. Zhang and P. S. Huang, "High-resolution, real-time three-dimensional shape measurement," *Optical Engineering*, vol. 45, no. 12, 2006.

[29] S. G. Narasimhan, S. J. Koppal, and S. Yamazaki, "Temporal dithering of illumination for fast active vision," in *Proceedings of the European Conference on Computer Vision*. Springer, 2008, pp. 830–844.

[30] K. Liu, Y. Wang, D. L. Lau, Q. Hao, and L. G. Hassebrook, "Dual-frequency pattern scheme for high-speed 3-d shape measurement," *Optics Express*, vol. 18, no. 5, pp. 5229–5244, 2010.

[31] Y. Gong and S. Zhang, "Ultrafast 3-d shape measurement with an off-the-shelf dlp projector," *Optics Express*, vol. 18, no. 19, pp. 19 743–19 754, 2010.

[32] L. Ekstrand, N. Karpinsky, Y. Wang, and S. Zhang, "High-resolution, high-speed, three-dimensional video imaging with digital fringe projection techniques," *Journal of Visualized Experiments*, no. 82, 2013.

[33] R. Sagawa, R. Furukawa, and H. Kawasaki, "Dense 3d reconstruction from high frame-rate video using a static grid pattern," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 9, pp. 1733–1747, 2014.

[34] J. Stühmer, S. Nowozin, A. Fitzgibbon, R. Szeliski, T. Perry, S. Acharya, D. Cremers, and J. Shotton, "Model-based tracking at 300hz using raw time-of-flight observations," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 3577–3585.

[35] J. Kim and M. Kim, "Motion capture with high-speed rgb-d cameras," in *Proceedings of the IEEE International Conference on Information and Communication Technology Convergence*. IEEE, 2014, pp. 394–395.

[36] E. Herbst, X. Ren, and D. Fox, "Rgb-d flow: Dense 3-d motion estimation using color and depth," in *Proceedings of the International Conference on Robotics and Automation*. IEEE, 2013, pp. 2276–2282.

[37] B. K. P. Horn and B. G. Schunk, "Determining optical flow," *Artificial Intelligence*, vol. 17, no. 1-3, pp. 185–203, 1981.

[38] M. Jaimez, M. Souiai, J. Stückler, J. Gonzalez-Jimenez, and D. Cremers, "Motion cooperation: Smooth piece-wise rigid scene flow from rgb-d images," in *Proceedings of the IEEE International Conference on 3D Vision*. IEEE, 2015, pp. 64–72.

[39] M. Alexa, D. Cohen-Or, and D. Levin, "As-rigid-as-possible shape interpolation," in *Proceedings of the Conference on Computer Graphics and Interactive Techniques*. ACM Press/Addison-Wesley Publishing Co., 2000, pp. 157–164.

[40] O. Sorkine and M. Alexa, "As-rigid-as-possible surface modeling," in *Proceedings of Symposium on Geometry Processing*, vol. 4, 2007, pp. 109–116.

[41] L. Gao, Y.-K. Lai, D. Liang, S.-Y. Chen, and S. Xia, "Efficient and flexible deformation representation for data-driven surface modeling," *ACM Transactions on Graphics*, vol. 35, no. 5, pp. 158:1–17, 2016.

[42] S.-Y. Chen, L. Gao, Y.-K. Lai, and S. Xia, "Rigidity controllable as-rigid-as-possible shape deformation," *Graphical Models*, vol. 91, pp. 13–21, 2017.

[43] M. Zollhöfer, M. Nießner, S. Izadi, C. Rhemann, C. Zach, M. Fisher, C. Wu, A. Fitzgibbon, C. Loop, C. Theobalt, and M. Stamminger, "Real-time non-rigid reconstruction using an rgb-d camera," *ACM Transactions on Graphics*, vol. 33, no. 4, p. 156, 2014.

[44] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 11, pp. 1330–1334, 2000.

[45] M. J. Black and P. Anandan, "A framework for the robust estimation of optical flow," in *Proceedings of the IEEE International Conference on Computer Vision*. IEEE, 1993, pp. 231–236.

[46] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert, "High accuracy optical flow estimation based on a theory for warping," in *Proceedings of the European Conference on Computer Vision*. Springer, 2004, pp. 25–36.

[47] D. Sun, S. Roth, and M. J. Black, "Secrets of optical flow estimation and their principles," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*. IEEE, 2010, pp. 2432–2439.

[48] H. Pottmann, Q.-X. Huang, Y.-L. Yang, and S.-M. Hu, "Geometry and convergence analysis of algorithms for registration of 3d shapes," *International Journal of Computer Vision*, vol. 67, no. 3, pp. 277–296, 2006.

[49] Z. Levi and C. Gotsman, "Smooth rotation enhanced as-rigid-as-possible mesh animation," *IEEE Transactions on Visualization and Computer Graphics*, vol. 21, no. 2, pp. 264–277, 2015.

[50] P. Sand and S. Teller, "Particle video: Long-range motion estimation using point trajectories," *International Journal of Computer Vision*, vol. 80, no. 1, p. 72, 2008.

[51] D. Baričević, T. Höllerer, P. Sen, and M. Turk, "User-perspective ar magic lens from gradient-based ibr and semi-dense stereo," *IEEE Transactions on Visualization and Computer Graphics*, vol. 23, no. 7, pp. 1838–1851, 2017.

[52] J. Xiao, H. Cheng, H. Sawhney, C. Rao, and M. Isnardi, "Bilateral filtering-based optical flow estimation with occlusion detection," *Proceedings of the European Conference on Computer Vision*, pp. 211–224, 2006.
[53] M. J. Black and P. Anandan, "Robust dynamic motion estimation over time," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*. IEEE, 1991, pp. 296–302.
[54] K. Guo, F. Xu, Y. Wang, Y. Liu, and Q. Dai, "Robust non-rigid motion tracking and surface reconstruction using l0 regularization," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 3083–3091.
[55] H. Wu, Z. Wang, and K. Zhou, "Simultaneous localization and appearance estimation with a consumer rgb-d camera," *IEEE Transactions on Visualization and Computer Graphics*, vol. 22, no. 8, pp. 2012–2023, 2016.
[56] I. Ltkebohle, "BWorld Robot Control Software," http://docs.opencv.org/3.0-beta/modules/cudaoptflow/doc/optflow.html, 2008, [Online; accessed 19-July-2008].
[57] R. G. de Albuquerque Azevedo, F. Ismério, A. B. Raposo, and L. F. G. Soares, "Real-time depth-image-based rendering for 3dtv using opencl," in *International Symposium on Visual Computing*. Springer, 2014, pp. 97–106.
[58] X. Wei, P. Zhang, and J. Chai, "Accurate realtime full-body motion capture using a single depth camera," *ACM Transactions on Graphics*, vol. 31, no. 6, p. 188, 2012.
[59] Z. Liu, L. Zhou, H. Leung, and H. P. Shum, "Kinect posture reconstruction based on a local mixture of gaussian process models," *IEEE Transactions on Visualization and Computer Graphics*, vol. 22, no. 11, pp. 2437–2450, 2016.
[60] T.-J. Mu, J.-H. Wang, S.-P. Du, and S.-M. Hu, "Stereoscopic image completion and depth recovery," *The Visual Computer*, vol. 30, no. 6-8, pp. 833–843, 2014.
[61] S.-J. Luo, Y.-T. Sun, I.-C. Shen, B.-Y. Chen, and Y.-Y. Chuang, "Geometrically consistent stereoscopic image editing using patch-based synthesis," *IEEE Transactions on Visualization and Computer Graphics*, vol. 21, no. 1, pp. 56–67, 2015.
[62] S.-P. Du, S.-M. Hu, and R. R. Martin, "Changing perspective in stereoscopic images," *IEEE Transactions on Visualization and Computer Graphics*, vol. 19, no. 8, pp. 1288–1297, 2013.
[63] S. Lee, G. J. Kim, and S. Choi, "Real-time depth-of-field rendering using anisotropically filtered mipmap interpolation," *IEEE Transactions on Visualization and Computer Graphics*, vol. 15, no. 3, pp. 453–464, 2009.
[64] H. Kwon, Y.-W. Tai, and S. Lin, "Data-driven depth map refinement via multi-scale sparse representation," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*. IEEE, 2015, pp. 159–167.
[65] F. Liu, C. Shen, and G. Lin, "Deep convolutional neural fields for depth estimation from a single image," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*. IEEE, 2015, pp. 5162–5170.

Ming-Ze Yuan received a BS degree in computer science from the University of Electronic Science and Technology of China and a Master's degree in computer science from the North China Institute of Computing Technology. He is currently working toward a PhD degree at the Institute of Computing Technology, Chinese Academy of Sciences. His research interests include computer graphics and virtual reality.

Lin Gao received a BS degree in mathematics from Sichuan University and a PhD degree in computer science from Tsinghua University. He is currently an Associate Professor at the Institute of Computing Technology, Chinese Academy of Sciences. His research interests include computer graphics and geometric processing.

Hongbo Fu received a BS degree in information sciences from Peking University, China, in 2002 and a PhD degree in computer science from the Hong Kong University of Science and Technology in 2007. He is an Associate Professor at the School of Creative Media, City University of Hong Kong. His primary research interests fall in the fields of computer graphics and human computer interaction. He has served as an Associate Editor of The Visual Computer, Computers & Graphics, and Computer Graphics Forum.

Shihong Xia is a Professor associated with the Beijing Key Laboratory of Mobile Computing and Pervasive Device, Institute of Computing Technology, Chinese Academy of Sciences. He received a BS degree in mathematics from Sichuan Normal University, China, in 1996 and a PhD degree in computer software and theory from the University of Chinese Academy of Sciences in 2002. His research interests include computer graphics, virtual reality and artificial intelligence.