Temporal Upsampling of Depth Maps Using a Hybrid Camera

Mingze Yuan, Lin Gao, Hongbo Fu, and Shihong Xia

Abstract—In recent years consumer-level depth sensors have been adopted in various applications. However, they often produce depth maps at a not very fast frame rate (around 30 frames per second), preventing them from being used for applications like digitizing human performance involving fast motion. On the other hand there are available low-cost faster-rate video cameras. This motivates us to develop a hybrid camera that consists of a high-rate video camera and a low-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 frames and estimates scene flow simultaneously. We have tested 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 purely tracking based method and the state-of-the-art techniques.

Index Terms—Depth Camera, Scene Flow, Joint Reconstruction, Human Motion Capture, Stereoscopic Image Generation, Depth-of-Field Effects.

✦

1 INTRODUCTION

In recent years low-cost depth cameras like 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 sacrifice for the low cost. For example the latest Kinect sensor for Xbox One is able to capture only 512×424 depth frames at 30 Hz. While such specifications might be enough for certain applications, they are not sufficient for applications involving fast motions and higher frame rate video. Furthermore, with recent advancements in photo sensors, consumer high resolution, high frame rate and low cost video cameras like GoPro have also opened up many ways in computer graphics, such as outdoor motion capture [4], structure from-motion (SfM) and dynamic hair capture [5]. Video cameras have many advantages over depth cameras in frame rate and resolution. However, the lack of depth information prevents video camera from applications such as image-base rendering [6] and image processing [7].

Observing that high-resolution video cameras are cheap and everywhere, several techniques (e.g., [8], [9]) have been proposed to use a hybrid camera model, i.e., a high-resolution video camera and a low-resolution depth sensor, to perform *spatial upsampling* of depth maps. The Kinect for Xbox One itself is already such a hybrid camera. However, the problem of low-rate capturing with existing depth sensors is largely unexplored, and is the focus of our work.

Motivated by the existing hybrid cameras for super-resolution of depth maps and the available high-rate but low-cost video cameras like GoPro cameras (with 240 FPS), we propose a hybrid camera to achieve temporal upsampling of depth frames (Fig. 1). Our hybrid camera consists of a low-rate depth sensor camera and a synchronized high-rate video camera. The key challenge is how to effectively extract the fast motion information captured by a high-rate video camera and then use it to guide the interpolation of depth frames. A straightforward solution is to first compute 2D optical flow [10] between consecutive images captured by the high-rate camera and then employ the resulting motion flow to estimate intermediate frames between a pair of original depth frames. However, this simple solution works well only for translational motions.

Another possible solution is based on scene flow [11]. However, the traditional scene flow estimation methods require color and depth images acquired at roughly the same 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 frames jointly: the estimated scene flow to guide the generation of intermediate depth frames, which in turn help refine the scene flow estimation. We derive data constraints from the high-rate camera images and enforce spatiotemporal regularization based on the shortest motion path and the isometric deformation assumption.

We have tested our hybrid camera on various examples with fast motions of single or multiple objects and humans. In these challenging examples with non-rigid motions and topological changes, our method has clear advantages over purely tracking based method and the state of the art [12]. We show that our joint optimization framework can be reduced for scene flow estimation. Compared to the state-of-the-art techniques [10], [13], [14], our method achieves comparable or even better performance on the MPI Sintel Dataset [15], and is able to capture motions more accurately or more detailed for multiple real-world cases.

- *M. Yuan, L. Gao and S. Xia are with the Advanced Computing Research Laboratory, Institute Of Computing Technology, Chinese Academy Of Sciences, Beijing, 100190, China.*
  E-mail:{yuanmingze,gaolin,xsh}@ict.ac.cn.
- *H. Fu is with the School of Creative Media, City University of Hong Kong.*
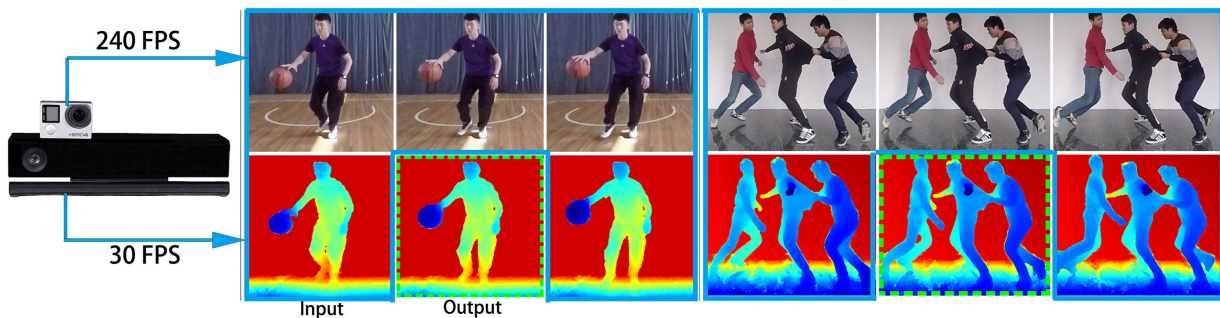 *E-mail:hongbofu@cityu.edu.hk*

Fig. 1. Our technique takes the input captured by a hybrid camera and is able to temporally upsample the depth maps captured by a low-rate depth sensor with the help of the color images captured by a high-rate video camera. Please see the accompanying video for better examination of the reconstruction results.

2 RELATED WORK

Depth sensors are often used together with video cameras to capture RGB-D images. Therefore the idea of hybrid camera is not new for 3D imaging. In fact, consumer-level depth sensors like Kinect are essentially hybrid cameras. It is well known that the depth data produced by low-cost depth sensors is often noisy and low resolution. A common approach to enhance depth maps in the spatial domain is to couple a low-resolution depth map with a high-resolution RGB image. Various solutions based on optimization (e.g., [8], [16]), joint edge-preserving upsampling filters (e.g., [17], [18], [19]), spatialtemporal filtering [9], [12], or shading cues [20], [21] have been explored to increase the spatial resolution of depth maps. All the methods above assume the same frame rates of color and depth maps. The exceptional case is the work by Dolson et al. [12]. However, their technique has mainly focuses on the interpolation of depth maps given sparse depth data with respect to camera frames, and has been applied to moving rigid objects only. Instead, our focus is on the estimation of new depth frames involving more complex motions. See the comparison between these two approaches in Sec. 5.1.

Hybrid cameras have also been used for motion deblurring [22], [23], [24]. For this application, at least one high-speed but often low-resolution video camera is needed to remove motion blur in color images captured by a low-speed high-resolution camera. Li et al. [23] use two low-resolution high-speed cameras as a stereo pair to reconstruct a low-resolution depth map, whose spatial resolution is then enhanced by using joint bilateral filters. Besides reducing motion blur, the approach of Tai et al. [24] 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, Wang et al.'s concurrent work [25] uses a hybrid camera system consisting of a 3 FPS light filed camera and a 30 FPS video camera to reconstruct 30 FPS light field images. The difference between Wang et al.s and our method is that, they use a learning-based method and up-sample light field images.

While consumer-level depth sensors are able to capture the depth at only a limited frame rate, high-speed depth cameras that 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., [26], [27], [28], [29], [30], [31]) 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 special illuminator. Compared with those approaches, our solution can be seen as a post-processing technique, and thus applicable to different types of depth sensors. Stühmer et al. [32] proposed to modify a typical Time-of-Flight (ToF) camera (e.g., Kinect v2) for model-based tracking at high frame rate (300Hz). However, their solution is limited to tracking of objects with rigid motion. Our work shares close resemblance to that by Kim and Kim [33], which uses multi-view hybrid cameras (consisting of eight high-speed RGB cameras and six ToF cameras) for motion capture. However, their technique is highly dependent on skeleton tracking, and thus is only suitable for articulated motion.

Our joint optimization to fuse the color and depth information and estimate the motion field yields a novel scene flow estimation method. Scene flow estimation for depth cameras is an active research topic recently. For example, Herbst et al. [34] extended the Horn-Schunck method [35] to the depth cameras with the depth data term for estimating the scene flow from a consumer depth camera. Jaimez et al. [36] proposed a total variation regularization term for RGB-D flow estimation at real time. Piecewise rigid motion priors have been added to the scene flow estimation in [14]. Jaimez et al. [37] estimate the scene flow with the joint optimization of motion and segmentation. Their method segments the scene with rigid motions. Sun et al. [10] order each depth frame into layers and assume the motion field in a single layer to be in the small range around the mean rigid rotation. When the objects in the same depth layer have large and different motions, this method would introduce artifacts (see Sec. 5.1).

As shown in [14], [37], the piecewise rigid regularization term of the motion field enhances the precision compared with the methods like [34], [36]. Our work employs not only a local rigid prior but also an isometric deformation prior, which corresponds to very common deformations in real scenarios. We follow the as-rigid-as-possible energy to model isometric deformations, which have been demonstrated for various graphics applications, such as shape in-
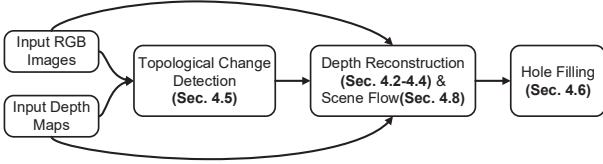
Fig. 2. Pipeline of our system.

(a) $C_{t(k)}$ (b) $C_{t(k)+1}$ (c) $C_{t(k)+2}$ (d) $C_{t(k)+3}$ (e) $C_{t(k)+4}$

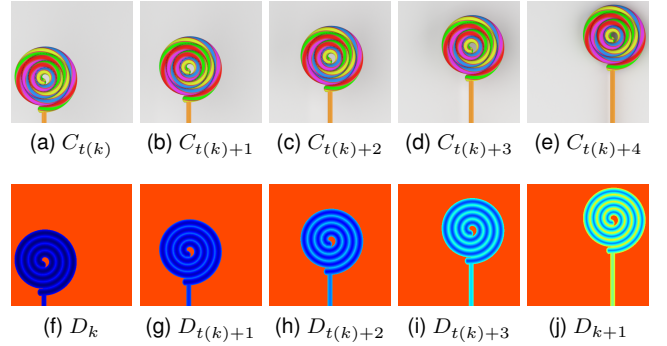(f) D_k (g) $D_{t(k)+1}$ (h) $D_{t(k)+2}$ (i) $D_{t(k)+3}$ (j) D_{k+1}

Fig. 3. Illustration of our main idea on the synthetic data in chronological order. Given the input color images (a-e) and depth maps (f) and (j), we temporally upsample depth frames by reconstructing intermediate depth maps (g-i).

TABLE 1
Notations.

Notations	Exposition
k	the index of captured depth maps
$\mathbf{D}_k, \mathbf{D}_{k+1}$	the first and second ones of a pair of consecutive depth maps captured by the Kinect
$\mathbf{I}_s$	the RGB images captured by GoPro in 240 fps at the interval between $\mathbf{D}_k$ and $\mathbf{D}_{k+1}$, $0 \leq s \leq g$
g	the ratio of the color camera's FPS to depth camera's FPS
$\mathbf{C}_{t(k)+s}$	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 $\mathbf{D}_{k+1}$
$\mathbf{D}_{t(k)+s}$	a depth map to be reconstructed corresponding to $\mathbf{C}_{t(k)+s}$, $1 \leq s \leq g$
$\mathbf{p}_{s,i}$	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 i-th pixel in $C_{t(k)+s}$, $0 \leq s \leq g$. $\mathbf{v}_s$ is the optical flow for $C_{t(k)+s}$.
$\mathbf{M}_k, \mathbf{M}_{k+1}$	point cloud generated by $\mathbf{D}_k$ and $\mathbf{D}_{k+1}$

terpolation [38], shape deformation [39], [40] and 3D shape tracking [41]. In our work, the as-rigid-as-possible energy is employed for the first time for the 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 Microsoft Kinect V2 RGB-D camera. The GoPro camera captures color images at 240 FPS with WVGA format, while Kinect is able to capture depth maps of resolution 512×424 at 30 FPS. As shown in Fig. 1, the GoPro is placed above the Kinect, such that Kinect's depth sensor is vertically aligned with the GoPro's color camera.

Calibration and alignment. The intrinsic and extrinsic parameters of the GoPro and Kinect cameras are calibrated by the method of [42]. The lens distortion of the color images from GoPro is then corrected according to the intrinsic parameters. Let $\mathbf{T}(\mathbf{R}, \mathbf{t})$ denote the transformation matrix, which is composed of the relative rotation and translation from the Kinect camera to the GoPro camera. This defines a continuous mapping function from the Kinect camera coordinate system to the GoPro's. $\psi(\cdot) = \mathbf{K}_g(\mathbf{T}(\cdot))$ is defined to represent this mapping, where $\mathbf{K}_g$ is the calibration matrix of GoPro. The two cameras are synchronized in the temporal domain by flashlight. More specifically, multiple flashlights captured to minimize time misalignment by detecting the first highlight images of both the color and depth cameras. After transforming the depth maps to the GoPro camera's coordinate system, the color images and aligned depth maps are cropped to retain the interesting part of the images.

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-rate video camera, as illustrated in Fig. 3. The core of our algorithm takes as input a pair of consecutive depth frames $\mathbf{D}_k$ and $\mathbf{D}_{k+1}$, and all the color images captured during the interval between $\mathbf{D}_k$ and $\mathbf{D}_{k+1}$. Our algorithm reconstructs a depth map for each color image without the associate captured depth frame, and also outputs the corresponding 3D scene flow. The key of our optimization is to exploit the dense 2D motion information from the color images (Sec. 4.2). The optical flow term is employed to model the 2D motion information that constrains the projection of the 3D scene flow. The projection term is used to build the connection between the 3D scene flow and the 2D optical flow. The depth maps $\mathbf{D}_k$ and $\mathbf{D}_{k+1}$ constrain the start and end position of the reconstructed depth maps. These constraints are modeled by the popular point-to-point and point-to-plane terms. With the assumption that shapes move isometrically, we employ an as-rigid-as-possible deformation term to model isometric deformations (Sec. 4.3). The total squared variation term is employed to regularize the scene flow. Considering that occlusion is unavoidable in the motion, we apply occlusion detection to avoid the artifact due to the occluded region(Sec. 4.4). And to solve the topological changes of the point cloud, we employ the pre-calculated optical flow to detect the topological changes(Sec. 4.5). We use both forward and backward reconstruction to exploit the geometry and point cloud information in $\mathbf{D}_k$ and $\mathbf{D}_{k+1}$ (Sec. 4.6). We incorporate the above data constraints and spatial-temporal regularization terms into a joint optimization (Sec. 4.7). The pipeline of our system show as Fig. 2.

4.1 Notations

Given a pair of consecutive depth frames $\mathbf{D}_k$ and $\mathbf{D}_{k+1}$, as illustrated in Fig. 3, let $\mathbf{C}_{t(k)}$ denote the color image corresponding to $\mathbf{D}_k$, $\mathbf{C}_{t(k)+g}$ the color image corresponding to $\mathbf{D}_{k+1}$, and $\mathbf{C}_{t(k)+s}$ ($s = 1, \cdots, g-1$) the intermediate color images captured between $\mathbf{C}_{t(k)}$ and $\mathbf{C}_{t(k)+g}$. $g = 4$ for the

synthetic example in Fig. 3, while g is roughly equal to 8 given our real inputs of color images captured at 240 FPS and depth maps captured at 30 FPS. Our ultimate goal is thus to reconstruct a depth map $\mathbf{D}_{t(k)+s}$ corresponding to $\mathbf{C}_{t(k)+s}, s \in [1, g]$. Our underlying optimization will also reconstruct $\mathbf{D}_{t(k)+g}$ (at the same time position as $\mathbf{D}_{k+1}$) to use $\mathbf{D}_{k+1}$ as boundary constraints for depth reconstruction, though $\mathbf{D}_{t(k)+g}$ will finally be discarded. The point cloud $\mathbf{M}_k$ is generated byteaser projecting $\mathbf{D}_k$ to the 3D space with the intrinsic parameters of Kinect's depth camera. The above and additional notations are summarized in Tab. 1.

In the time period during which two depth maps captured, the GoPro will capture g color images. The first and last frame of color images are corresponding to the two adjacent depth maps. As show in Fig. 1, in our setting of 240 FPS capturing rate for GoPro and of 30 FPS capturing rate for Kinect, the value of 'g' is 8. As show in Fig. 3, the index of color image which has corresponding depth map $\mathbf{D}_k$ is represented as $t(k)$, the corresponding color image at the same time is represented as $\mathbf{C}_{t(k)}$. For the next depth frame $\mathbf{D}_{k+1}$, the index of the color image captured at the same time is $t(k) + g$. For color image $\mathbf{C}_{t(k)+s}, s \in [1, g]$ we will reconstruct the corresponding depth map $\mathbf{D}_{t(k)+s}$. But the last reconstructed depth map $\mathbf{D}_{\mathbf{t(k)+g}}$ will be replaced by the depth data $\mathbf{D}_{k+1}$ whose depth information is accurate.

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 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:

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

where $\mathbf{I}_s$ is the intensity of the s-th color image (i.e., $\mathbf{C}_{t(k)+s}$), and $\mathbf{v_{s,x}}$ and $\mathbf{v_{s,y}}$ are the 2D motion flow in the images along x-axis and y-axis, respectively. λ_{opti} is the weight of the 2D optical flow term and is chosen as $\lambda_{opti} = 1$ in the implementation. $\rho(r) = \sqrt{r^2 + \varepsilon^2}$ is the kernel function that defines the robust metric for handing noise and outliers ($\varepsilon = 10^{-4}$ in the implementation) [43], [44]. Furthermore, to improve the optical flow value we employ the weighted median filter to avoid over-smoothing along object edges [45].

The previous scene flow estimation methods [10], [14], [36] 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 measuring the distance 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 for measuring the geometric distance. $\psi(\cdot)$ (defined in Sec. 3) projects depth data $\mathbf{D}_k$ to

3D space to generate a point cloud. The ith pixel in $\mathbf{D}_k$ is projected to $\mathbf{p}_i$ in 3D. Connecting relationship between pixel i and its adjacent points is created for the spatial coherency.

We employ the following point-to-point term and point-to-plane term [46] for the geometry constraints of the depth map $\mathbf{D}_{t(k)+g}$ to be reconstructed:

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

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

where $\tilde{\mathbf{p}}_{k+1,i}$ is the closest point to $\mathbf{p}_{k,i}$ in the $\mathbf{D}_{k+1}$ depth map, $\mathbf{n}_{k+1,i}$ is normal vector of $\tilde{\mathbf{p}}_{k+1,i}$. The energy weights for the point-to-point term and point-to-plane term are set as $\lambda_{point} = 0.2$ and $\lambda_{plane} = 0.8$, respectively. Since the optical flow is essentially the projection of the scene flow introduced by the reconstructed depth maps, with the projection function $\psi(\cdot)$ introduced in Sec. 3. We model this constraint as follows:

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

and use the projection term to connect the 2D optical flow and 3D point cloud. The energy weight λ_{proj} is set as 1.0. $\mathbf{O}(\cdot)$ is the function to indicate whether this point is occluded by other depth pixels in motion (Sec. 4.4).

4.3 Spatial and Temporal Regularization

With the assumption that objects move under isometric deformations, we add a local rigid regularization term for each reconstructed depth map in the spatial domain.

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

where N_i denotes connected points to the i-th pint in point cloud. It is more likely that the connected points with similar depth and color appearance would share similar locally rigid motions. λ_{iso} is set to 3.0. These weights are defined as $w_{ij} = w_{c,ij} \cdot w_{d,ij} \cdot w_{t,ij}$, with the depth coherence $w_{d,ij} = \exp(-\|\mathbf{p}_{k,i} - \mathbf{p}_{k,j}\|^2/\sigma_d^2)$, color coherence $w_{c,ij} = \exp(-\|\mathbf{I}(i) - \mathbf{I}(j)\|^2/\sigma_c^2)$, and topological 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 $\mathbf{M}'_k$ (see Tab. 1 and Sec. 4.5). The connection of occlusive points is generally less reliable. When at least one of p_i and 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 empirical $\sigma_t = 0.015$, $\sigma_c = 1$ and $\sigma_d = 0.015$. The total quadratic variations are employed to regularize the motion field and are defined as follows:

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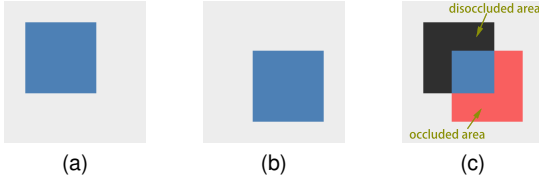\tag{6}
$$

Fig. 4. The blue block in (a) moves to lower right, resulting in (b). (a) is warped by the motion flow to generate (c). The black region in (c) is the disoccluded area of (a) and the red part is the occluded area. The red region in (c) appears in (a) but is covered in (b), while the black area in (c) is in (b) but is covered in (a).
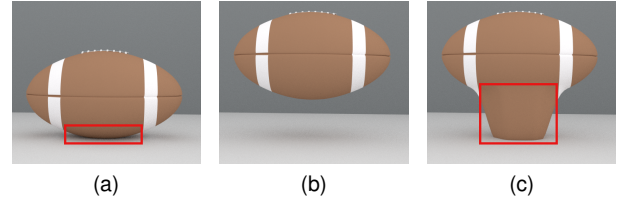

Fig. 5. Topological change detection. Football in (b) is football in (a) bounced from the ground. (a) is warped by optical flow to (c). In the red inset of (a) and (c), they are the topological changing area.

Here the weight of this term is set as $\lambda_{reg} = 0.8$, and d_A is a constant parameter chosen as 10 for all the examples.

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 summarization of the Euclidean distance of the depth positions of the corresponding pixels in two consecutive point cloud.

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

where λ_{short} is set to 0.5.

4.4 Occlusion Detection

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

First, the occlusion results in the projection relationship error in E_{proj}, since the objects that are occluded would cut down the relation between the 3D scene flow and the optical flow. To remove the mismatch of projection relation, we adopt the idea in [47] to detect the occlusion. The $\mathbf{O}(\mathbf{v}_{s,i}, I)$ function used in Eq. 4 is defined as $\exp(-\|\mathbf{div}(s,i)\|^2/\sigma_1^2) \cdot \exp(-\|\mathbf{I}(s,i) - \mathbf{I}(s+1, i+\mathbf{v}_{s,i})\|^2/\sigma_2^2)$, where $i+\mathbf{v}_{s,i}$ is the index of pixel obtained by applying translation of $\mathbf{v}_{s,i}$ to the index i [48]. The $\mathbf{div}(s,i)$ is the divergence of optical flow, and $\mathbf{I}(s,i)$ is the color of the i-th pixel in the s-th image. Based on our experience, we set $\sigma_1 = 1.0$ and $\sigma_2 = 20$. The variation of the pixel motion can be obtained from divergence of optical flow. This will affect the projection term because some motion would be occluded. The occluded depth pixels without the optical flow information will be derived by the motion of its connected points.

Second, the occlusion also produces outlier of E_{opti} due to color pixel mismatch [49]. To deal with the outlier, we use the robust kernel function $\rho(r)$ in Sec. 4.2. However, the outlier still makes optical flow over-smoothed at motion boundaries, i.e., the edges between two objects with different speeds. This problem can be further alleviated by applying the weighted media filter [45].

Third, the occlusion generates holes in the reconstructed depth maps. The occluded surfaces are divided into occlusion and disocclusion as illustrated in Fig. 4 [50]. The disoccluded objects can be detected by inversing time. In other words, the shape of the disoccluded objects can be recovered from the next frames in the reversed timeline. We will show how these holes can be filled in Sec. 4.6.

4.5 Topological Change Detection

The connection between the neighboring pixels describes an object's topology. Topology changes often occur in the scene objects interacting with each other such as clap, rebound and handshake. In such interacting examples, the points which are joined together in the previous frame are separated, or separated points might join together in the next frame are topological change. Both the cases will change the connection of points in the objects boundary.

The topological change due to merging of separated objects is solved by the point cloud merging, with the information of optical flow and geometry constraints (Sec. 4.2). We detect the topological changes of separating objects by getting distance change with the adjacent points, between point cloud and its warped version, which is obtained by warping the original point cloud using the rough motion flow evaluated from the corresponding color images ($\mathbf{C}_{t(k)+g}$), as shown in Fig. 5.

Optical flow is the motion flow of 3D point cloud projected to 2D space. Let $\mathbf{V}_k$ be the accumulated optical flow of $\mathbf{V}_{t(k)+s}$, $s \in [0, g-1]$. So we can estimate 3D point cloud motion flow by projecting the 2D optical flow to 3D space with $\mathbf{K}_g$ and $\mathbf{D}_k$ as the approximation of $\mathbf{D}_{k+1}$. Each motion flow value of $\mathbf{F}_v$ which is the accumulation of $\mathbf{F}_k$, $k \in [1, g)$, corresponds to a point in $\mathbf{M}_k$. $\mathbf{F}_v$ is applied to $\mathbf{M}_k$ to obtain $\mathbf{M}'_k$.

However, the $\mathbf{F}_v$ is only the approximative motion information without the depth change or occlusion detection, in order to keep the geometric information of $\mathbf{M}'_k$ consistent with $\mathbf{M}_k$, we use E_{point}, E_{plane}, and E_{iso} to get a coarse result. We define a weight of topological change $w_{t,ij}$ to represent how topological changes in a pair of points (i, j) (Sec. 4.3). In Fig. 5(c) the relationship of connection which be elongated is detected as topological change.

4.6 Hole Filling

In order to reconstruct the depth more completely, we fill the holes in the depth map. Holes are due to either the occlusion or the imaging principle of Kinect. In Fig. 6 (h), the blue pixels is an example of the first kind of holes, which are caused by the occlusion in the reconstruction. And the yellow pixels is an example of the second kind of holes, which natively exist in the input range data. We use different approaches to deal with these two kinds of holes. The workflow is shown in Fig. 6.

To deal with the first kind of holes, we use the forward and backward reconstruction information together [51]. During the forward reconstruction, the depth data of the
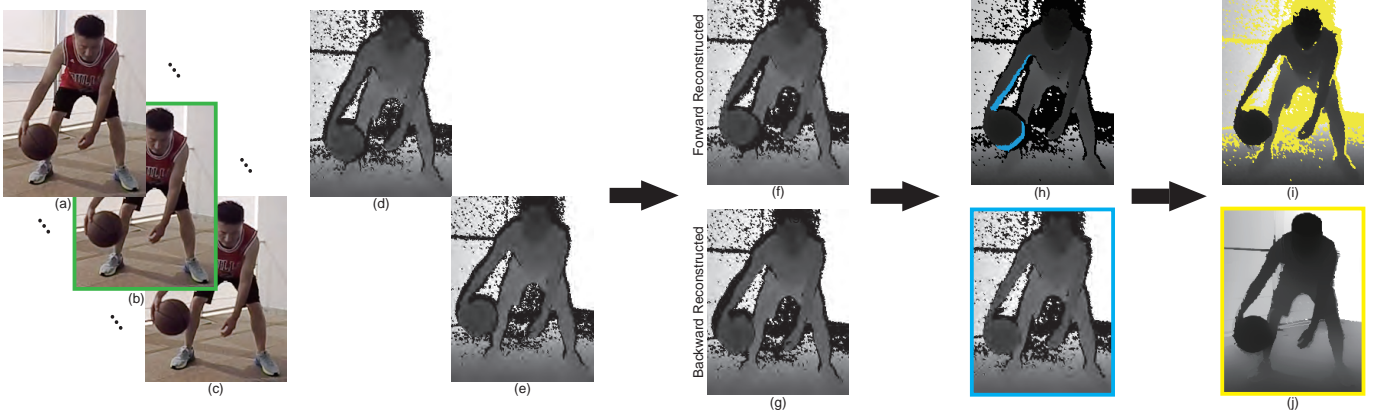
Fig. 6. Hole filling. (a-c) are the input color images. (d) and (e) are the input depth maps corresponding to (a) and (c), respectively. (f) is the forward reconstructed depth corresponding to (b), and (g) is backward reconstructing result. Holes exist in the reconstructed depth (f) and (g) due to either occlusion or the imaging principles. In (h) the pixels in blue are generated by occlusion. (i) is the depth map fixed by forward and backward reconstructed depth, and the yellow pixels are holes due to the imaging artifacts. (j) the results by filling all the holes.

occluded portion in the initial frame is also missing in the next $g - 1$ forward reconstructed depth frames $\mathbf{D}_{t(k)+s}$. The occluded part of the reconstructed depth data in the current frame is thus the accumulated occluded part of the previous depth frames of the current sequence in the forward reconstruction. On the other hand, the occluded part of the forward reconstruction is the disoccluded part of backward reconstruction. So the missing depth information can be achieved from in the backward reconstruction's depth data. 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 get the corresponding relationship between the disocclusive depth data and the pixels in the backward reconstructed depth map, and thus fill this kind of holes, as shown in Fig. 6(i).

The second kind of holes appears due to two main reasons: imperfect alignment of the depth and color images; environmental interference such as hair, glare, motion blur and reflectivity of objects. Parts of the second kind of hole can also be partially filled use forward and backward reconstructed depth. This is because that the missing depth data of the second kind of holes, which is generated by motion blur and interference, can be obtained from adjacent depth frames captured by Kinect. We use the bilateral filter to fill the residual missing depth data with the help of color information [18].

4.7 Energy Minimization

At every interval between two consecutive depth frames 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:

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

The Gaussian-Newton is employed to solve this optimization [52]. Eq. 8 can be rewritten as the following summation of squared residues: $\mathbf{E}(x) = \sum_i g_i(\mathbf{x})^2 = \|\mathbf{g}(\mathbf{x})\|^2$,
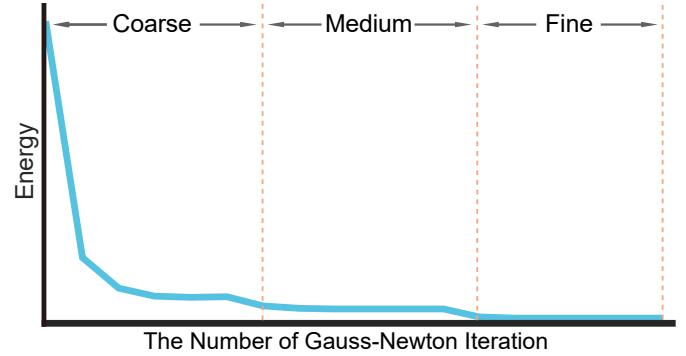

Fig. 7. The energy convergency of each optimization after searching for the closest point. We use three hierarchical levels and run five Gauss-Newton iterations over each level. In every Gauss-Newton iteration, there are ten PCG iterations to minimize the energy.

where $\mathbf{x}$ is an unknown vector variable and $\mathbf{g}(\mathbf{x})$ is a vector function with its element functions $g_i(\mathbf{x})$. In each iteration, the update equation of the variables is $\mathbf{x}_{k+1} = \mathbf{x}_k + \Delta\mathbf{x}$. For the energy, using the first-order Taylor approximation, it will introduce the following least squares problem. $\mathbf{E}(\mathbf{x}_{k+1}) = \sum_i g_i(\mathbf{x}_{k+1})^2 \approx \|\mathbf{g}(\mathbf{x}_k) + \mathbf{J}(\mathbf{x}_k)\Delta\mathbf{x}\|^2$. The $\Delta\mathbf{x}$ is satisfied with the equation $\mathbf{J}^T(\mathbf{x}_k)\mathbf{J}(\mathbf{x}_k)\Delta\mathbf{x} = -\mathbf{J}^T(\mathbf{x}_k)\mathbf{g}(\mathbf{x}_k)$, $\mathbf{J}(\mathbf{x}_k)$ is the Jacobian matrix with the value at $\mathbf{x}_k$. For solving this least squares optimization, we apply the preconditioned conjugate gradient (PCG) solver with several CUDA kernels [41]. The optical flow $\mathbf{V}$ is initialized by the GPU-base method [53]. In this optimization, we use three hierarchical levels. During the prolongation from the coarse level to the fine level, the bi-linear interpolation is employed to the optical flow $\mathbf{V}$ and points positions $\mathbf{P}$. The rigid rotation $\mathbf{R}$ will be the same in the corresponding position in the coarse levels. As it's shown in Fig.7 the outer loop takes five Gauss Newtons steps and the inner loop takes ten PCG steps.

4.8 Scene Flow Estimation

This joint optimization framework would introduce the depth maps and the 3D 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
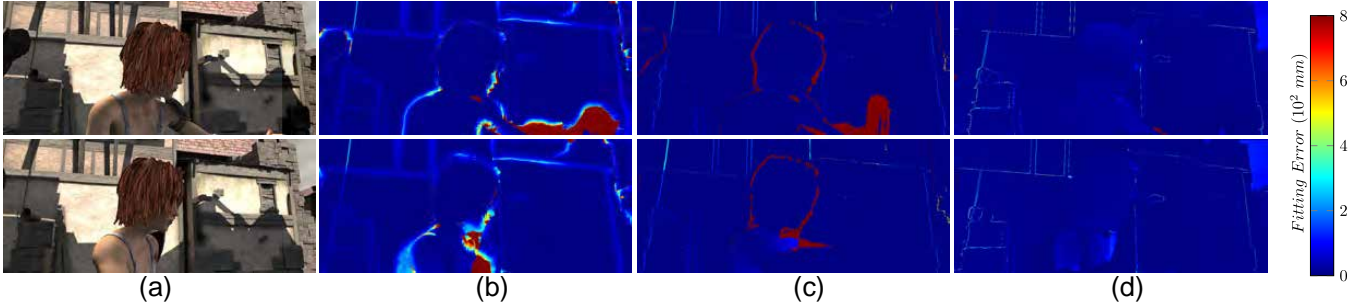
Fig. 8. First row: several color images of alley_1 sequence of MPI Sintel dataset. Second to fourth columns: corresponding depth reconstruction error images of Dolson et al.'s [12], the purely tracking based method, and ours.

estimation method. In Sec. 5.1, we will give quantitative evaluations of our scene flow estimation method and show its advantage over the current state-of-the-art works [10], [14], [36].

5 RESULTS AND DISCUSSIONS

We have tested our algorithm on various scenes' data captured by our hybrid camera. The scenes include playing basketball, jumping, playing games and so on. Occlusion often occurs in motion especially in basketball scenes, such as breaking through where players often occlude each other. Topological change is also a common phenomenon in real-world cases such as dribbling basketball, where the connecting relationship of points cloud between hand and ball change frequently. Even more, we capture some dynamic scenarios with moving camera to make the whole scene moving. To prove the robustness of our hybrid system we evaluate our method on several real-world scenarios and dataset, and compare result with other depth reconstruct methods and scene flow methods. The result of the comparison are show that our method's reconstructed depth is more accurate the purely tracking based method and Dolson et al's method. In this section, we present representative results, quantitative and qualitative evaluations, and possible applications. Please see the accompanying video for better examining the results.

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

Parameter settings. We set a weight for each term of Eq. 8. The input data could be noise-less under the situation when it is captured indoors with plenty of sunlight. We use those weight discussed Sec. 4.2 and Sec. 4.3 to reconstruct depth data. If there are much noise in captured depth data,

TABLE 2
Quantitative Evaluation of Scene Flow Estimation on the MPI Sintel Dataset. The Lower Error Values, the Better Performance

EPE	PD-Flow	SR-Flow	Layered-Flow	Our
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	-	**0.82**
AAE	PD-Flow	SR-Flow	Layered-Flow	Our
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	-	**0.28**
RMSE	PD-Flow	SR-Flow	Layered-Flow	Our
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	-	**2.96**

we set λ_{reg} larger to suppress the noise in point cloud. If there is loud noise in RGB data which make optical flow unfaithful, we make weight of λ_{opti} and λ_{proj} smaller to reduce the impact.

5.1 Quantitative Evaluation

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. Here we evaluate this scene flow estimation method on the MPI Sintel dataset [15], which contains the synthetic data, and provides the ground truth of optical flow and depth data for each RGB frame. There are several continuous video sequences, each of episode has approximately 30 frames. Our approach is compared with the state-of-the-art techniques for scene flow estimation, including *Layered-Flow* [10], *SR-Flow* [14] and *PD-Flow* [36]. The Average Root Mean Squared Error (**RMSE**), the Average Angular Error (**AAE**), and End Point Error (**EPE**) are used as error metrics [14]. The quantitative evaluation results on the MPI Sintel dataset are given in Tab. 2. The numerical results show that under all the error metrics and for almost all the sequences our method outperforms the other scene flow estimation methods and produces results closer to the ground truth. The artifact in the results by *Layered-Flow* [10] is possibly due to the shared depth layer of the poster and background. Layered-Flow fails to deal with invalid depth data on depth map. So it

Fig. 9. Reconstructed depth map of real-world scenarios. From top to down rows: input color images, depth maps reconstructed by the purely tracking based method, results by Dolson et al.'s method [12], and results by our method.
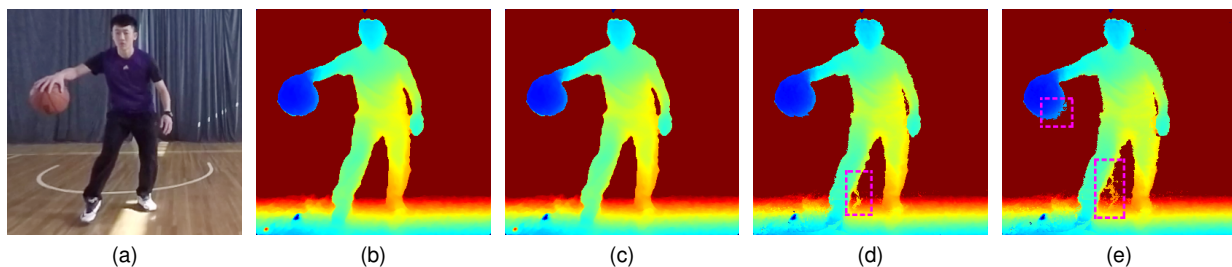

Fig. 10. Downsampling result. (a) is RGB image captured by GoPro corresponding to reconstructed depth map, (b) (c) (d) (e) is reconstructed depth data with the ratio between the number of depth frames and the number of color frames as $1:8$, $1:16$, $1:24$, $1:32$ respectively.

does not exist result in the sequence of $market_2$ of Tab.2 which contain invalid depth data.

Depth reconstruction. To evaluate the precision of the reconstructed depth maps we use the same MPI Sintel flow dataset [15] instead of real-world data which contains random noise and invalid pixels. We compare our method with the state-of-the-art work [12], which is, to the best of our knowledge, the only existing method for temporal upsampling of depth maps given the inputs similar to ours. Their method also uses the color information to interpolate the depth data to recover more depth information. In the original dataset, each synthetic color image has its corresponding depth map. To evaluate the ability of reconstruction, we temporally downsample the depth frames, resulting in the 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. 3. The result show that the reconstructed depth maps by our method are more accurate than those by [12]. This advantage can be visually examined in the error images in Fig. 8.

TABLE 3
Quantitative Evaluation of Depth Reconstruction on the MPI Sintel Dataset.

RMSE	[12]	Purely Tracking Based Method	Ours
$bamboo\_1$	614.3	784.4	**238.3**
$alley\_1$	420.3	1048.5	**61.8**
$sleeping\_2$	70.0	46.9	**26.1**
$bandage\_1$	31.7	87.84	**8.3**

5.2 Qualitative Evaluation

In this subsection, we will evaluate our method on the data of real-world scenes, compared to the technique of Dolson et al. [12] and purely tracking based method, which performs non-rigid registration between $\mathbf{D}_k$ and $\mathbf{D}_{k+1}$ (without considering the input color images, and thus without using E_{opt} and E_{proj} item in the energy equation), and uses interpolation to generated intermediate depth maps.

We have made the comparisons on multiple challenging examples (see the accompanying video). Fig. 9 gives representative results. Since purely tracking based method does not take into the color information into account, it may fail
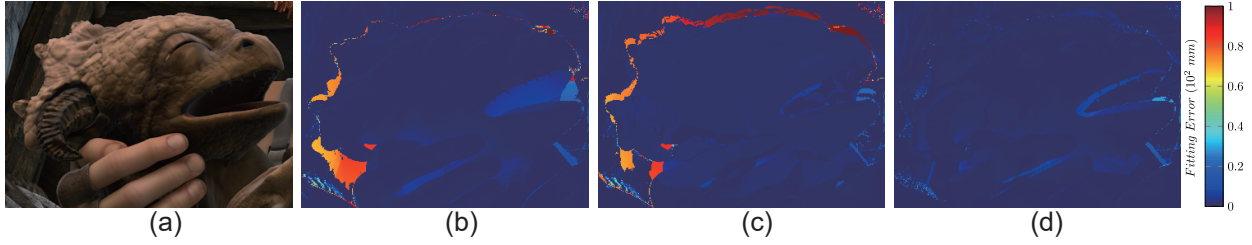
Fig. 11. Component evaluation. (a) is RGB images corresponding to the reconstructed depth maps. (b) (c) (d) are reconstructed depth maps with full energy terms, without E_{point} or E_{plane}, and without E_{opti}, respectively.

to reconstruct the depth information for fast motion. In contrast, our method takes advantage of the color information to evaluate the motion flow information, which helps our method obtain more accurate correspondence across frames.

In contrast, the method of [12] heavily depends on the color information. It is based on the assumption that depth data in terms of color and timestamp. As shown in the second column of Fig. 9, therefore their approach might produce results with noise-like artifacts when the input data contains noisy or even invalid depth data, like those from the Kinect sensor. This problem is even more serious for the example in the fourth row of Fig. 9, because the gym is too large (more than 6 meter) for the Kinect to get correct depth information. Our approach achieves much better results, mainly because of our careful consideration of the color information (for optical flow) as well as the topological and geometric relationships of points in the point cloud. The colors of the foreground and background are similar, the approach of Dolson et al. [12] will easily get confused and produce poor depth reconstruction results.

The frame ratio between the number of depth frames captured by Kinect and the number of color frames captured by GoPro is $1 : 8$. We also tested our method with $1 : 16$, $1 : 24$ and $1 : 32$ frame ratios, which are simulated by extracting the Kinect's depth maps by interval 2, 3 and 4 frames. It is expected and also shown in Fig. 10 the higher ratio might lead to more artifacts (e.g., on the legs in this example). Such artifacts are mainly due to the errors of topological change detection caused by cumulative error of optical flow.

5.3 Impact of Individual Components

In this work, multiple terms works together to estimate the motion and reconstruct the depth frame. To evaluate the impact of energy items, we drop out the optical flow item (E_{opti}) and closet point items(E_{point}, E_{plane}) respectively when reconstructing the depth data. A representative result is shown as Fig. 11.

Importance of E_{point} and E_{plane}. These two terms exploit from the input consecutive depth maps the 3D motion information, which is useful to estimate motion. Fig. 11(c) shows the reconstructed depth data without E_{point} or E_{plane} terms. Compared to the results with the full terms as shown in Fig. 11(b), it can be easily seen that there are more artifacts in the boundary and geometry of objects.

Importance of E_{opti}. E_{opti} takes full advantage of 2D motion information from RGB images. Due to the higher frame rate there is less ambiguous and more robust motion information in continuous RGB frames. As shown in

Fig. 11(d), when E_{opti} is omitted, there are more artifacts than the reconstructed results without E_{point} or E_{plane} terms. This is mainly because there are more miss-matching nearest point relationship in E_{point} and E_{plane} without using optical flow information.

5.4 Applications

Various applications could benefit from this work with reconstructed high frame rate depth. Here, we demonstrate three applications including fast human motion capture and rendering stereoscopic images for VR environment and depth-of-field effect.

Human Motion Capture. The state-of-the-art human motion capture works include [1], [54], [55]. In those works the input depth maps are from the Kinect with 30FPS. In order to capture fast human motion, we make use of the depth frames from Kinect with our system. We apply the full-body motion capture algorithm [54] on the depth maps from our hybrid system to capture the 3D skeletal poses of the fast human movement. As shown in Fig. 12, thanks to the high-rate and accurate depth maps from 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 [56], [57], [58]. In our system, the stereoscopic video can be easily acquired with the captured RGB and reconstructed depth maps by Depth image-based rendering (DIBR) [59]. DIBR is a technology to synthesize virtual views of scene with monocular RGB images and depth maps. With the help of DIBR we synthesize an RGB video for left eye and use the original RGB video for right eye. Fig. 13 shows an example of the resulting stereoscopic images (the method of de Albuquerque Azevedo et al. [59]). Please also find the rendered stereoscopic video in the accompanying video.

Depth-of-Field (DoF). DoF effect which makes object acceptably sharp and others blur in image is a common skill in photography to emphasize a subject. Meanwhile, DoF is a commonly used render effect applied in image or animation [60]. The usual method to producing a DoF effect in captured images is using light field cameras which capture images at three frame rate per second and expensive. We use image-based algorithm to rendering DoF effect with RGBD images. By the help of reconstructed depth, we can easily simulate DoF effect and refocus [7] in acquired fast frame rate RGB images. We implement Kraus er al. [3] proposed
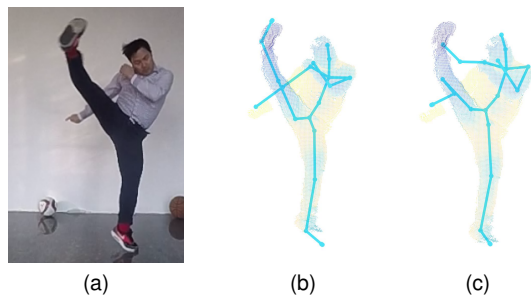
Fig. 12. Human motion tracking. (a) is RGB image captured by GoPro. Corresponding point cloud and tracked post are shown in (b) and (c), which are tracked with reconstructed high-frame rate depth data and Kinect captured depth data, respectively.
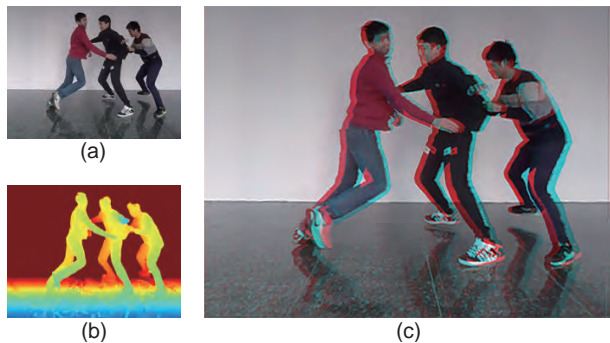

Fig. 13. Stereoscopic video rendering with reconstructed depth maps and RGB images captured by GoPro. (a) is RGB image captured by Go-Pro, (b) is reconstructed depth map, (c) is stereoscopic image rendered by de Albuquerque Azevedo et al.'s method [59].
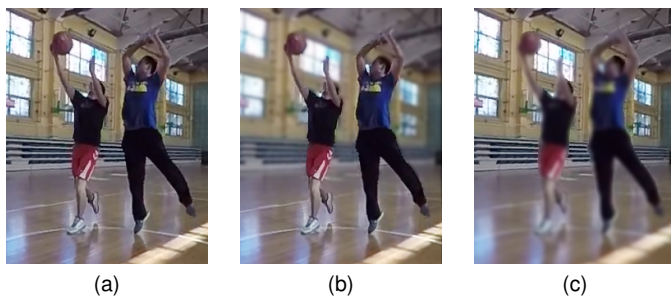

Fig. 14. Depth of field effect: (a) is entire sharp RGB image captured with GoPro. To isolate player from the background, (b) and (c) render with small depth-of-field, make the background and players out of focus respectively.

method based on sub-images to change the DoF of acquired images. As shown in Fig. 14, with the reconstructed dense depth, we rendering DoF effect in GoPro captured images, and refocus on players and background respectively to emphasize different subjects.

5.5 Limitations and Future work

Our work has taken the first step to address 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 CPU to GPU.
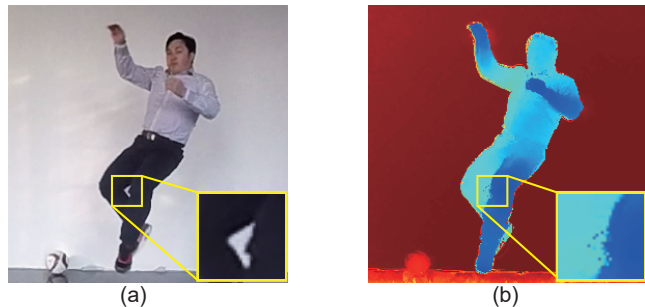

Fig. 15. Failure case. (a) is an input RGB frame, (b) is the corresponding reconstructed depth map. Our method fails to reconstruct depth data in the highlighted yellow box, because of occlusion and fast motion.

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, which uses D_{k+1} to reconstruct previous depth frame $D_{t(k)+s}$. The reconstructed result has a delay of $3 \sim 4$ milliseconds, even the framework achieve real time.

Finally, our system does not reconstruct depth faithfully when motion is too fast and occlusion is too serious. As shown in Fig. 15, in the gap between two legs where the occlusion is serious and the legs' motion is very fast our reconstruction result suffers from errors. The less accurate depth is due to hole filling process error (Sec. 4.6), because there is too few depth information of the gap. It can be mitigated by more powerful hole filling methods, e.g. data-driven based method [61] and deep learning based method [62].

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," in *ACM Transactions on Graphics (TOG)*, vol. 30, no. 4. ACM, 2011, p. 31.

[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 (TOG)*, vol. 33, p. 6, 2014.

[6] 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.

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

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

[9] 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.

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

[11] S. Vedula, S. Baker, P. Rander, R. Collins, and T. Kanade, "Three-dimensional scene flow," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, vol. 2, 1999, pp. 722–729.

[12] 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 Computer Society, 2010, pp. 1141–1148.

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

[14] J. Quiroga, T. Brox, F. Devernay, and J. L. Crowley, "Dense semi-rigid scene flow estimation from rgbd images." in *Proceedings of the European Conference on Computer Vision*, ser. Lecture Notes in Computer Science, D. J. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds., vol. 8695, 2014, pp. 567–582.

[15] 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*, ser. Part IV, LNCS 7577, A. Fitzgibbon et al. (Eds.), Ed. Springer-Verlag, Oct. 2012, pp. 611–625.

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

[17] 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 Computer Society, 2007.

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

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

[20] 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, pp. 200:1–200:10, 2014.

[21] e. a. Zollhöfer, "Shading-based refinement on volumetric signed distance functions," *ACM Transactions on Graphics*, vol. 34, no. 4, pp. 96:1–96:14, 2015.

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

[23] F. Li, J. Yu, and J. Chai, "A hybrid camera for motion deblurring and depth map super-resolution," in *Proceedings of Conference on Computer Vision and Pattern Recognition*, 2008.

[24] 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.

[25] 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 (TOG)*, vol. 36, no. 4, 2017.

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

[27] 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.

[28] 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.

[29] 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.

[30] 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.

[31] 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.

[32] 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 and Pattern Recognition*, 2015.

[33] 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.

[34] 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.

[35] B. K. P. Horn and B. G. Schunk, "Determining Optical Flow," *Artificial Intelligence*, vol. 17, pp. 185–203, 1981.

[36] M. Jaimez, M. Souiai, J. González-Jiménez, 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*, may 2015, pp. 98–104.

[37] M. Jaimez, M. Souiai, J. Stueckler, 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*, Oct. 2015.

[38] 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*, 2000, pp. 157–164.

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

[40] 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.

[41] 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, 2014.

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

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

[44] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert, "High accuracy optic flow estimation based on a theory for warping," in *Proceedings of the European Conference on Computer Vision*, vol. 3024, 2004, pp. 25–36.

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

[46] 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.

[47] P. Sand and S. Teller, "Particle video: Long-range motion estimation using point trajectories," *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, vol. 2, no. 1, pp. 2195–2202, 2006.

[48] 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.

[49] J. Xiao, H. Cheng, H. Sawhney, C. Rao, and M. A. Isnardi, "Bilateral filtering-based optical flow estimation with occlusion detection," *European Conference on Computer Vision*, pp. 211–224, 2006.

[50] 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*, 1991, pp. 296–302.

[51] 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 and Pattern Recognition*, 2015.

[52] 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.

[53] I. Ltkebohle, "BWorld Robot Control Software," http://docs.opencv.org/3.0-beta/modules/cudaoptflow/doc/optflow.html, 2008, [Online; accessed 19-July-2008].

[54] X. Wei, P. Zhang, and J. Chai, "Accurate realtime full-body motion capture using a single depth camera," *ACM Transactions on Graphics (TOG)*, vol. 31, no. 6, p. 188, 2012.

[55] 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.

[56] 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.

[57] 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.

[58] S.-P. Du, S.-M. Hu, and R. Martin, "Changing perspective in stereoscopic images," *IEEE Transactions on Visualization and Computer Graphics*, vol. 19, no. 8, pp. 1288–1297, 2013.

[59] 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.

[60] 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.

[61] H. Kwon, Y.-W. Tai, and S. Lin, "Data-driven depth map refinement via multi-scale sparse representation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 159–167.

[62] F. Liu, C. Shen, and G. Lin, "Deep convolutional neural fields for depth estimation from a single image," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 5162–5170.