This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCSVT.2015.2416571, IEEE Transactions on Circuits and Systems for Video Technology

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCSVT.2015.2416571, IEEE Transactions on Circuits and Systems for Video Technology

Fast Depth Video Compression for Mobile RGB-D Sensors

Xiaoqin Wang, Y. Ahmet Şekercioğlu, *Senior Member, IEEE,* Tom Drummond, *Member, IEEE,* Enrico Natalizio, *Member, IEEE,* Isabelle Fantoni, *Member, IEEE,* and Vincent Frémont, *Member, IEEE*

***Abstract*—We propose a new method, called *3D Image Warping Based Depth Video Compression* (IW-DVC), for fast and efficient compression of depth images captured by mobile RGB-D sensors.**

The emergence of low-cost RGB-D sensors has created opportunities to find new solutions for a number of computer vision and networked robotics problems, such as 3D map building, immersive telepresence or remote sensing. However, efficient transmission and storage of depth data still presents a challenging task to the research community in these applications. Image/video compression has been comprehensively studied, and several methods have already been developed. But, these methods result in unacceptably suboptimal outcomes when applied to the depth images.

We have designed the IW-DVC method to exploit the special properties of the depth data to achieve a high compression ratio while preserving the quality of the captured depth images. Our solution combines the egomotion estimation and 3D image warping techniques and includes a lossless coding scheme which is capable of adapting to depth data with a high dynamic range. IW-DVC operates in high-speed, suitable for real-time applications, and is able to attain an enhanced motion compensation accuracy compared with the conventional approaches. Also, it removes the existing redundant information between the depth frames to further increase compression efficiency. Our experiments show that IW-DVC attains a very high performance yielding significant compression ratios without sacrificing image quality.

***Index Terms*—Depth map, motion compensation, depth video coding, inter-frame correlation.**

I. Introduction

RECENTLY invented low-cost RGB-D sensors, such as Kinect [1] and PrimeSense Carmine [2], have attracted immense interest from the computer vision, multimedia, and robotics communities [3]. By integrating RGB cameras and infrared sensing, RGB-D sensors provide synchronized color and depth images at high frame rates. The complementary nature of the depth and visual information makes them suitable for applications in various research fields, including three-dimensional (3D) mapping and localization [4], [5], or scene/object reconstruction [6], [7]. In these applications, a mobile RGB-D sensor is used to observe the texture and geometrical structure information of a static scene/object from different viewpoints. Since such applications generate large amount of visual and depth information, which usually needs to be remotely accessed, an efficient compression algorithm is necessary to minimize the memory utilization and communication bit rate as much as possible.

Data compression for visual information is now a well-established technology. There are numerous lossless and lossy compression algorithms for image and video applications, JPEG 2000 [8] and H.264/AVC [9] being the most prominent ones. The coding of depth images, however, is a recent research topic. A depth image, representing the relative distance from the recording camera to an object in the 3D space, usually consists of smooth regions and sharp edges at boundaries between the object and background. A typical way to compress a depth image sequence is by processing each frame as a standard gray scale image for viewing purposes and applying the standard video coding schemes. Redundancy between successive frames can be removed by estimating the motion between frames and then generating the motion vectors (MVs), which describe the motion of the pixel information repeatedly shown in successive frames. In standard video coding techniques, 2D block matching algorithms [10] are used for motion estimation, in which frames are divided into blocks of $M \times N$ pixels, such as 16×16 and 8×8. A search is performed to find a matching block from a frame i in some other frame j. However, this approach is unfortunately very suboptimal for depth image sequences. Large, homogeneous areas on the surface of an object can be divided into small blocks and the sharp discontinuities at object boundaries can be placed into the same block. Because of this, these schemes result in significant coding artifacts along the depth discontinuities in the reconstructed depth images, especially when the compression ratio is high [11], [12].

The above mentioned shortcomings of the standard image and video coding algorithms are the driving force for concentrating the research efforts in developing new data compression schemes by considering the specific properties of depth images. The schemes can be broadly classified under two categories: Approaches developed to remove the (i) temporal redundancies [13], [14], [15], [16], [17], [18], [19], or (ii) inter-view redundancies [20], [21], [22], [23], [24], [25], [26], [27], [28] in depth video images. The starting point of all these schemes is 2D block matching algorithms. In the following paragraphs we provide a brief overview of these methods.

The method proposed in [13] exploits the correspondences between the depth images and the corresponding color frames captured by a texture camera. It adopts a conventional block

This work was supported by the Australian Research Council Centre of Excellence for Robotic Vision (project number CE140100016).

X. Wang, Y. A. Şekercioğlu, and T. Drummond are with the ARC Centre of Excellence for Robotic Vision, Monash University, Victoria, 3800, Australia (e-mail: xiaoqin.wang@monash.edu; ahmet.sekercioglu@monash.edu, tom.drummond@monash.edu).

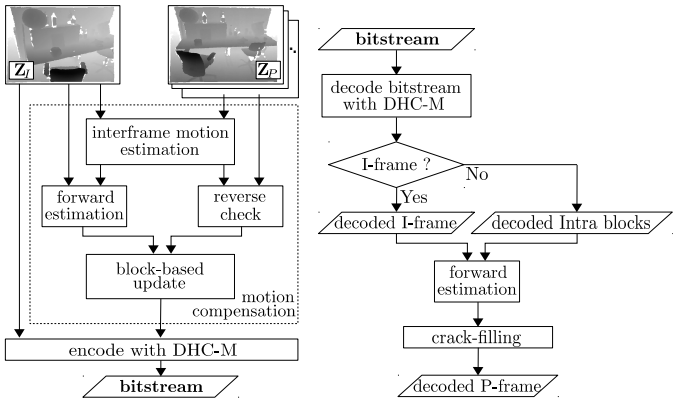
E. Natalizio, I. Fantoni, and V. Frémont are with the Heudiasyc Laboratory, Compiègne University of Technology, Compiègne, 60200, France (e-mail: enrico.natalizio@hds.utc.fr; isabelle.fantoni@hds.utc.fr; vincent.fremont@hds.utc.fr).

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCSVT.2015.2416571, IEEE Transactions on Circuits and Systems for Video Technology

matching approach to determine the MVs according to the texture information. The MVs from the texture information are considered to be encoding both the texture and depth image sequences. A number of techniques [15], [17], [18], [19] are developed based on this concept and provide enhanced performances. Daribo et al. [15] proposed a MV sharing algorithm based on the correlation between the motion of the texture and of the depth. This algorithm considers the motion of a block at the same coordinates in both video texture and depth images and uses a joint distortion criterion to generate common MVs for both texture and depth. Shahriyar et al. [16] proposed an inherently edge-preserving depth-map coding scheme. Their scheme uses the texture motion vectors, avoids distortion on edges, and accurately preserves the depth information on edges. Fan et al. [18] proposed a motion estimation method with various block sizes. It first determines the block size and its corresponding MV using the color information. Then, a z-direction motion estimation is used to correct the depth values in each block. A similar algorithm, proposed in [17], replaces the 2D block matching algorithm with a 3D one operating in horizontal, vertical, and depth dimensions. The method by Nguyen et al. [19] compresses the depth video by using a weighted mode filter to suppress the coding artifacts.

In addition to the temporal redundancy in the frames captured by a single camera at different times, inter-view redundancy also exists in the frames captured from various viewpoints in a multi-camera system. Many research studies have been proposed to remove these inter-view correlations and achieve efficient compression for depth video in multi-view scenarios. The study by Zhang et al. [20] combines multi-view video coding and the prediction of MVs in the depth image sequence from those of the texture image sequence. Ekmekçioğlu et al. [23] proposed a coding scheme in which the bit rate used for multi-view depth image coding is further reduced by skipping one or more temporal layers of selected depth image views. The method proposed by Lee et al. [24] reduces the bit rate by skipping depth blocks under consideration of temporal and inter-view correlations of texture images. The temporal and inter-view correlations are measured from the temporally successive pictures and the neighboring views synthesized through pixel-by-pixel mapping respectively. Lee et al. [26] describe a multi-view depth video coding scheme that incorporates depth view synthesis, H.264/MVC, and additional prediction modes. A depth image-based rendering technique [27] is adopted in this approach to generate an additional reference depth image for the current viewpoint. The inter-view correlation is exploited by using a texture and depth view synthesis approach. A block-based depth image interpolation approach was proposed by Wang et al. [28]. In this scheme, the first and last frames in a texture video are treated as the key frames with known depth images. The remaining depth images corresponding to other texture frames can be recovered by the proposed bidirectional prediction algorithm.

To the best of our knowledge, these represent the most relevant works related to the depth video coding in which the characteristics of depth images are preserved and the temporal and inter-view correlations are exploited. However, all of these



(a) *Encoder* (b) *Decoder*

Fig. 1. Operational overview of the *3D Image Warping Based Depth Video Compression* (IW-DVC) framework.

algorithms focus on compressing the depth images captured by cameras that remain in a fixed position. The situation becomes inevitably much more complicated when moving cameras involved. As the distance between a moving camera and the objects in a scene changes across time, depth values of the same objects change in successive depth frames. Therefore, the coding schemes with motion compensation methods for static cameras become very inaccurate or even become useless with mobile cameras.

In this paper, we consider typical situations in visual-SLAM (simultaneous localization and mapping) or surface reconstruction applications in which a mobile RGB-D sensor collects depth and visual information of a static environment. As the RGB information can be compressed easily by conventional image/video coding schemes, we only focus on developing a fast and efficient coding method for the depth video. An overview of the new scheme we propose, called 3D *Image Warping Based Depth Video Compression* (IW-DVC), is shown in Fig. 1.

We first eliminate the redundant depth data at the encoder side (Fig. 1(a)). Then the bitstream is further compressed to maximize its storage and transmission efficiency. For the elimination of redundant depth data, instead of using conventional motion compensation algorithms with 2D block-based matching, we propose a novel method which combines egomotion estimation of a mobile RGB-D sensor and 3D image warping technique [29]. This approach enables better prediction of frames, and so leading to an enhanced compression performance by taking advantage of the characteristic properties of depth images. Additionally, this motion compensation method is computationally efficient and can operate in real-time, since it does not need to derive the mean squared error (MSE) block by block, and it further can be used as a front-end for other depth image coding schemes to remove the redundancy between consecutive frames prior to further coding.

We have also included our lossless compression scheme, designed particularly for depth images captured by RGB-D sensors, called *Differential Huffman Coding with Multiple Lookup Tables* (DHC-M). In IW-DVC framework, each I-frame is encoded using DHC-M directly. DHC-M is also

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCSVT.2015.2416571, IEEE Transactions on Circuits and Systems for Video Technology

applied on each P-frame after the motion compensation and performs entropy coding allowing for higher compression ratios.

At the decoder side, crack-filling algorithms are adopted to deal with the under-sampling problem [30] in the reconstructed depth images.

The proposed framework can be implemented directly on a mobile RGB-D sensor system to achieve efficient depth video storage and transmission. The received depth information can be used for many applications, such as visual mapping and exploration, 3D scene reconstruction, or free viewpoint video rendering. The main accomplishments of the study presented in this paper can be summarized as:

- Theoretical development and practical implementation of the first motion compensation algorithm for depth video captured by a mobile sensor/camera,
- Thorough performance evaluation of the proposed framework under the various parameter changes to the system, and
- Extensive experiments using multiple datasets to evaluate the coding performance under a variety of scenes.

The rest of the paper is organized as follows. In Section II, the mechanism of the RGB-D sensor is presented and the 3D image warping based depth video compression (IW-DVC) framework is explained. Experimental results are presented and analyzed in Section III, followed by our concluding remarks.

II. 3D Image Warping Based Depth Video Compression Framework

Given a mobile RGB-D sensor which uses depth sensing to explore and map an environment, significant redundancy exists in the captured depth video, as the same surface geometrical information repeatedly appears in the consecutive depth frames, especially at high frame rates. Our objective is to prevent the same depth information from being unnecessarily transmitted and stored by removing this redundancy as much as possible. A general description of the framework we proposed is shown in Fig. 1. The list of symbols used through the paper is given in Table I.

A brief overview of the system is as follows. Let $\mathbf{Z}_I$ and $\mathbf{Z}_P$ denote an I-frame and P-frame in a group of pictures (GoP) in captured depth video (Table I can be referred for the descriptions of the mathematical symbols used throughout the paper). The system consists of two main components: motion compensation algorithm and a lossless coding scheme. Before encoding $\mathbf{Z}_P$ into a bitstream, the differences between $\mathbf{Z}_P$ and $\mathbf{Z}_I$ should be determined in motion compensation first. The first step of the encoding procedure (Fig. 1(a)) is the interframe motion estimation. We estimate the motion of a moving sensor in the time interval of capturing depth frames $\mathbf{Z}_I$ and $\mathbf{Z}_P$. In the second step, $\hat{\mathbf{Z}}_P$, a prediction of $\mathbf{Z}_P$ is generated by using the interframe motion information, forward estimation/reverse check, and block-based update. This procedure estimates the newly observed depth information in $\mathbf{Z}_P$ but not in the reference frame $\mathbf{Z}_I$. Then, only the newly observed information in $\mathbf{Z}_P$ is encoded using an entropy coding scheme.

TABLE I
Mathematical Notation

Notation	Descriptions
$\mathbf{Z}_I$	I-frame in a group of depth frames.
$\mathbf{Z}_P$	P-frame in a group of depth frames.
$\hat{\mathbf{Z}}_P$	Predicted P-frame.
$\mathbf{p}_e$	Vector representing a real world point in Euclidean space.
(i_c, j_c)	Principal point coordinates of the pinhole camera model.
(f_x, f_y)	Focal length of the camera in horizontal and vertical axes.
$\mathbf{M}$	Transformation matrix.
$\mathbf{p}_P^l$	Sampled points on a P-frame.
$\mathbf{p}_I^{l*}$	Corresponding points of $\mathbf{p}_P^l$ on an I-frame.
N_P	Number of sampled points on a P-frame.
S_P	Set of sample points on a P-frame
S_I^*	Set of corresponding points on an I-frame.
$\vec{n}_{l,P}$	Surface normal at point $\mathbf{p}_P^l$.
$w_{l,P}$	Weight parameter for correspondence established between $\mathbf{p}_P^l$ and $\mathbf{p}_I^{l*}$.
$\mathbf{E}$	Update transformation matrix in each iteration.
α_j	An element of a 6D motion vector.
$\mathbf{G}_j$	6D motion generator matrices.
X_i, X_{i-1}	Current/reference pixel.
H_i, H_{i-1}	Current pixel/reference pixel with the invalid depth value (2047 for the sensor we use).
V_i, V_{i-1}	Current pixel/reference pixel with a valid depth value.
$q_h()$	Conditional probability distribution when the reference pixel has the invalid depth value.
$q_v()$	Conditional probability distribution when the reference pixel has a valid depth value.

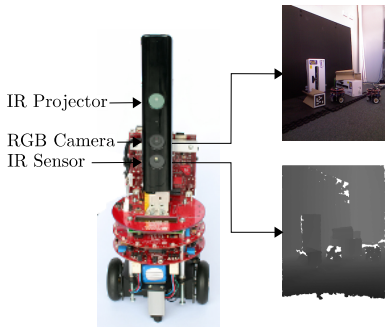


Fig. 2. A Microsoft Kinect RGB-D sensor, shown as installed vertically on Monash University's experimental mobile robot "eyeBug". Samples of captured color and depth images are shown as well.

As a result, the redundancy in the depth video is taken out during the encoding process.

In the decoding process, the received bitstream is decoded with the same codec used at the encoder side. I-frames can be directly decoded from bitstream. Each P-frame has to be reconstructed using the decoded newly observed information (intra blocks) in each P-frame and its corresponding I-frame. Then, a crack-filling approach is used to deal with the under-sampling issue [31] and enhance the quality of the reconstructed P-frames. Detailed explanation of each step is described in the next section preceding with a very brief overview of the RGB-D sensor.

A. RGB-D Sensor Mechanism

Microsoft Kinect (shown installed on WSRNLab's [32] experimental robot eyeBug in Fig. 2), is one of the most popular RGB-D sensors. It contains an RGB camera and an IR projector-camera pair, which can produce color and depth

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCSVT.2015.2416571, IEEE Transactions on Circuits and Systems for Video Technology

images at a rate of 30 fps. As described by its inventors [33], depth is measured via a triangulation process that is based on the detection of transverse shifts of local dot patterns in the IR speckle with respect to its reference patterns at a known distance to the device. This process is repeated for all local regions in the IR speckle and produces a disparity-based depth image. The default RGB video stream provided by Kinect is in 8-bit VGA resolution (640×480 pixels) and the monochrome depth video stream is also in VGA resolution with 11-bit depth, which provides 2,048 levels of sensitivity. If the Kinect cannot determine the distance to a point, it denotes this with an invalid depth value.

Note that all the depth images presented in this paper are scaled down to 8 bits per pixel for the visual inspection purposes.

B. Interframe Motion Estimation

Conventional 2D block-based motion estimation algorithms use 2D block matching approach to estimate the MVs which can map the pixels from the reference frame to the current frame. However, this kind of approaches relies on the assumption that the pixels representing the visual information on the same object's surface stays unchanged even if the object or the camera is in motion. Therefore, it is clear that these 2D block-based motion estimation approaches are not optimal for depth videos as the depth frames represent the distance of objects within a scene relating to the camera position and the depth values change when the camera or object is moving.

Taking advantage of the depth image characteristics, the depth pixels in the reference frame can be mapped to the current frame. Consider the vector $\mathbf{p}_e = [x\ y\ z\ 1]^T$ which represents a real world point in Euclidean space by using homogeneous coordinates. Given the intrinsic parameters of an RGB-D sensor: principal point coordinates $(i_c,\ j_c)$ and focal length of the camera $(f_x,\ f_y)$, $\mathbf{p}_e$ can be estimated from the corresponding pixel in depth image by using the pinhole camera model as

$$\mathbf{p}_e \equiv \frac{1}{z}\begin{bmatrix} x & y & z & 1 \end{bmatrix}^T \equiv \begin{bmatrix} \frac{i-i_c}{f_x} & \frac{j-j_c}{f_y} & 1 & \frac{1}{z} \end{bmatrix}^T, \quad (1)$$

where (i, j) denotes the pixel coordinates of this real world point projection in the depth image, and z is the corresponding depth value reported by the camera.

In the discussion that follows, we assume that $\mathbf{p_e}$ can be observed in $\mathbf{Z}_I$ and $\mathbf{Z}_P$ captured by the mobile RGB-D sensor, and the projections of $\mathbf{p}_e$ are located at pixel coordinates (i_I, j_I) and (i_P, j_P) on the depth frames $\mathbf{Z}_I$ and $\mathbf{Z}_P$, respectively. Also, under the assumption that the world coordinates system is equivalent to the mobile sensor coordinate system, the depth pixel (projection) at (i_I, j_I) in $\mathbf{Z}_I$ can establish a relationship between the depth pixel at (i_P, j_P) in $\mathbf{Z}_P$ as follows,

$$\begin{bmatrix} \frac{i_P-i_c}{f_x} & \frac{j_P-j_c}{f_y} & 1 & \frac{1}{z_P} \end{bmatrix}^T = \mathbf{M}\begin{bmatrix} \frac{i_I-i_c}{f_x} & \frac{j_I-j_c}{f_y} & 1 & \frac{1}{z_I} \end{bmatrix}^T \quad (2)$$

and, to simplify the equation, by doing some rudimentary algebraic substitutions we obtain following equation in inverse depth coordinate,

$$\begin{bmatrix} u_P & v_P & 1 & q_P \end{bmatrix}^T = \mathbf{M}\begin{bmatrix} u_I & v_I & 1 & q_I \end{bmatrix}^T. \quad (3)$$

The transformation matrix $\mathbf{M}$ represents the 6DoF motion model, which describes the motion of the sensor and the transformation of the structure between a pair of depth images. It has the form

$$\mathbf{M} = \begin{bmatrix} & \mathbf{R} & & \mathbf{t} \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad (4)$$

where $\mathbf{R}$ is a 3×3 rotation matrix and $\mathbf{t}$ is a 3×1 translation vector. Therefore, if we have the accurate information about the transformation matrix $\mathbf{M}$, each pixel in the reference depth frame can find its corresponding pixel in the current depth frame.

For RGB-D sensors, several approaches for estimation of $\mathbf{M}$ can be found in the research literature. Their comparative evaluation, and our proposed method for $\mathbf{M}$ estimation can be found in [34], [35]. Our approach is a variant of the Iterative Closest Point (ICP) algorithm [36] which has enhanced performance in circumstances with heavy occlusion. Even though this algorithm is initially developed for estimating the relative pose between two RGB-D sensors, it can also estimate a single sensor's motion during the time period for capturing a pair of depth frames. It estimates $\mathbf{M}$ through explicit registration of surface geometries extracted from two depth frames. The registration problem is approached by iteratively minimizing a cost function whose error metrics are defined based on the bidirectional point-to-plane geometrical relationship.

Suppose that correspondences between $N = N_I + N_P$[1] pairs of points from two depth images $\mathbf{Z}_I$ and $\mathbf{Z}_P$, are established, we can then estimate the transformation matrix $\mathbf{M}$ by minimizing the bidirectional point-to-plane error metric, $\mathcal{C}$, expressed in normal least squares form as

$$\begin{aligned} \mathcal{C} = &\sum_{l=1}^{N_I}\left[w_{l,I}(\mathbf{M}\mathbf{p}_I^l - \mathbf{p}_P^{l^*})\cdot\vec{n}_{l^*,P}\right]^2 + \\ &\sum_{l=1}^{N_P}\left[w_{l,P}(\mathbf{M}^{-1}\mathbf{p}_P^l - \mathbf{p}_I^{l^*})\cdot\vec{n}_{l^*,I}\right]^2, \end{aligned} \quad (5)$$

where $\mathbf{p}_I^l$ and $\mathbf{p}_P^l$ are the sampled points in depth frames $\mathbf{Z}_I$ and $\mathbf{Z}_P$, $\mathbf{p}_P^{l*}$ and $\mathbf{p}_I^{l*}$ are their corresponding points respectively. The variables $w_{l,I}$ and $w_{l,P}$ are weight parameters for correspondences established in opposite directions between pairs. Also, $\vec{n}_{l^*,P}$ and $\vec{n}_{l^*,I}$ are the surface normals at the corresponding points $\mathbf{p}_P^{l^*}$ and $\mathbf{p}_I^{l^*}$ in real world coordinates, and

$$\vec{n}_{l^*,P} = \begin{bmatrix} \alpha_{l^*,P} & \beta_{l^*,P} & \gamma_{l^*,P} & 0 \end{bmatrix}^T, \quad (6)$$

$$\vec{n}_{l^*,I} = \begin{bmatrix} \alpha_{l^*,I} & \beta_{l^*,I} & \gamma_{l^*,I} & 0 \end{bmatrix}^T. \quad (7)$$

The cost function presented in Eq. 5 consists of two parts:

1) the sum of squared distances in the forward direction from depth images $\mathbf{Z}_I$ to $\mathbf{Z}_P$, and

[1]A larger number of sampled points leads to more accurate results. However, the processing time increases at the same time. By considering both time consumption and accuracy, we have chosen $N_I = N_P = 250$ in this paper.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCSVT.2015.2416571, IEEE Transactions on Circuits and Systems for Video Technology

2) the sum of square distances in the backward direction from $\mathbf{Z}_P$ to $\mathbf{Z}_I$.

However, to estimate the 6DoF motion that minimizes Eq. 5 would require partial derivatives of $\mathbf{M}$ to be obtained. In order to make this problem solvable, we simplify Eq. 5 into the following form

$$\mathcal{C} = \sum_{l=1}^{N_I} \left[w_{l,I}(\mathbf{M}\mathbf{p}_I^l - \mathbf{p}_P^{l*}) \cdot \vec{n}_{l^*,P}\right]^2 + \sum_{l=1}^{N_P} \left[w_{l,P}(\mathbf{M}\mathbf{p}_I^{l^*} - \mathbf{p}_P^l) \cdot \vec{n}_{l,P}\right]^2, \tag{8}$$

We can then estimate $\mathbf{M}$ by re-weighting the least squares operation in an ICP framework. Details of the criteria for selecting the weight function can be found in [34]. The weight function chosen for forward direction is

$$w_{l,I} = \begin{cases} c/\left[c + (z_P^{l*} - z_I^l)\right] & \text{if } z_I^l \leq z_P^{l*} \\ c/\left[c + (z_P^{l*} - z_I^l)^2\right] & \text{otherwise,} \end{cases} \tag{9}$$

where z_I^l is the depth value of the sampled point $\mathbf{p}_I^l$, z_k^{l*} is the depth value of the corresponding point $\mathbf{p}_P^{l*}$, c is the mean of deviation between depth values of sampled points and depth values of corresponding points. Likewise, the weight function used in the backward direction is

$$w_{l,P} = \begin{cases} c/\left[c + (z_I^{l*} - z_P^l)\right] & \text{if } z_P^l \leq z_I^{l*} \\ c/\left[c + (z_I^{l*} - z_P^l)^2\right] & \text{otherwise.} \end{cases} \tag{10}$$

Algorithm 1 Inter-frame motion estimation procedure

1: Capture depth frames, $\mathbf{Z}_I$ and $\mathbf{Z}_P$.
2: Initialize the transformation matrix $\mathbf{M}_P$ as identity transformation.
3: **procedure** REPEAT UNTIL CONVERGENCE
4: Update depth frame $\mathbf{Z}_I$ according to $\mathbf{M}$.
5: Randomly sample N_I points from $\mathbf{Z}_I$ to form set S_I, $S_I = \{\mathbf{p}_I^l \in \mathbf{Z}_I, l = 1, \ldots, N_I\}$,
6: Randomly sample N_P points from $\mathbf{Z}_P$ to form set S_P, $S_P = \{\mathbf{p}_P^l \in \mathbf{Z}_P, l = 1, \ldots, N_P\}$.
7: Find the corresponding point set, P_P^*, of S_I in $\mathbf{Z}_P$, $S_P^* = \{\mathbf{p}_P^{l*} \in \mathbf{Z}_P, l = 1, \ldots, N_I\}$; Find the corresponding point set, S_I^*, of S_P in $\mathbf{Z}_I$, $S_I^* = \{\mathbf{p}_I^{l*} \in \mathbf{Z}_I, l = 1, \ldots, N_P\}$. ▷ The correspondences are established using the project and walk method with a neighborhood size of 3×3 based on the nearest neighbor criteria.
8: Apply the weight functions in Eqs. 9 and 10 bidirectionally, $S_I \mapsto S_P^*$, $S_P \mapsto S_I^*$
9: Compute the update matrix $\mathbf{E}$ which can minimize Equation 8 based on current bidirectionally weighted correspondences.
10: $\mathbf{M} \leftarrow \mathbf{E}\mathbf{M}$.
11: **end procedure**

An overview of the entire process is presented in Algorithm 1. In this coarse-to-fine algorithm, each iteration generates an update $\mathbf{E}$ to the sensor's pose which modifies the transformation matrix $\mathbf{M}$. $\mathbf{E}$ takes the same form as $\mathbf{M}$ which may be parameterized by a 6-dimensional motion vector having the elements $\alpha_1, \alpha_2, \ldots, \alpha_6$ via the exponential map and their corresponding group generator matrices $\mathbf{G}_1, \mathbf{G}_1, \ldots, \mathbf{G}_6$ as

$$\mathbf{E} = \exp\left(\sum_{j=1}^{6} \alpha_j \mathbf{G}_j\right), \tag{11}$$

where

$$\mathbf{G}_1 = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \mathbf{G}_2 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \mathbf{G}_3 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\mathbf{G}_4 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \mathbf{G}_5 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \mathbf{G}_6 = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Here $\mathbf{G}_1$, $\mathbf{G}_2$ and $\mathbf{G}_3$ are the generators of translations in x, y and z directions, while $\mathbf{G}_4$, $\mathbf{G}_5$ and $\mathbf{G}_6$ are rotations about x, y and z axes respectively.

Afterwards, the task becomes finding the $\alpha_1, \ldots, \alpha_6$ that describe the inter-frame transformation. Based on Eq. 3, through determining the partial derivatives of u_P, v_P and q_P with respect to the elements of the motion vector $\alpha_1, \ldots, \alpha_6$, the Jacobian matrix for each established corresponding point pair can be obtained from

$$\mathbf{J} = \begin{bmatrix} q_I & 0 & -u_I q_I & -u_I v_I & 1 + u_I^2 & -v_I \\ 0 & q_I & -v_I q_I & -1 - v_I^2 & v_I u_I & u_I \\ 0 & 0 & -q_I^2 & -v_I q_I & u_I q_I & 0 \end{bmatrix}. \tag{12}$$

The six-dimensional motion vector, which minimizes Eq. 5, is then determined iteratively by the least squares solution

$$\mathbf{B} = (\mathbf{K}^T\mathbf{W}\mathbf{K})^{-1}\mathbf{K}^T\mathbf{W}\mathbf{Y} \tag{13}$$

in which $\mathbf{W}$ is a diagonal matrix weighting the bidirectional point-to-plane correspondences based on Eq. 8. $\mathbf{B}$, $\mathbf{Y}$, and $\mathbf{K}$ are matrices

$$\mathbf{B} = \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \alpha_4 \\ \alpha_5 \\ \alpha_6 \end{bmatrix}, \mathbf{Y} = \begin{bmatrix} -(\mathbf{p}_I^1 - \mathbf{p}_P^{1^*}) \cdot \vec{n}_{1^*,P} \\ \vdots \\ -(\mathbf{p}_I^{N_I} - \mathbf{p}_P^{N_I^*}) \cdot \vec{n}_{N_I^*,P} \\ -(\mathbf{p}_I^{1^*} - \mathbf{p}_P^1) \cdot \vec{n}_{1,P} \\ \vdots \\ -(\mathbf{p}_I^{N_P^*} - \mathbf{p}_P^{N_k}) \cdot \vec{n}_{N_P,P} \end{bmatrix}, \tag{14}$$

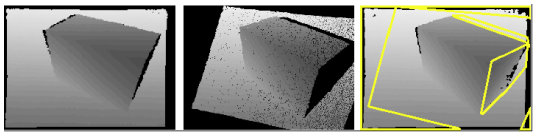
$$\mathbf{K} = \begin{bmatrix} \vec{n}'_{1^*,P}\mathbf{J}_1 \ \ldots \ \vec{n}'_{N_I^*,P}\mathbf{J}_{N_I} \ \vec{n}'_{1,P}\mathbf{J}_{1^*} \ \ldots \ \vec{n}'_{N_P,P}\mathbf{J}_{N_P^*} \end{bmatrix}^T \tag{15}$$

Here, $\vec{n}'_{l,P} = \begin{bmatrix} \alpha_{l,P} & \beta_{l,P} & \gamma_{l,P} \end{bmatrix}^T$ is the surface normal expressed in a slightly different form than the one shown in Eqs. 6 and 7. To detect the convergence of our algorithm, we use the thresholds for the ICP framework presented in [37]. The required number of iterations mostly depends on the motion between two frames. If the motion is small, only a few iterations is required. Otherwise, a large number of iterations is needed. As the speed of the camera is not high, algorithm usually converges in a few iterations. Once the algorithm converges, the registration is considered as completed and the $\mathbf{M}$ is determined.

C. Forward Estimation/Reverse Check and Block-based Update

After determining $\mathbf{M}$, the correspondences between the depth pixels in $\mathbf{Z}_I$ and depth pixels in $\mathbf{Z}_P$ are established by *forward estimation/reverse check* and *block-based update* procedures. They are explained in this section.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCSVT.2015.2416571, IEEE Transactions on Circuits and Systems for Video Technology



$\mathbf{Z}_I$ $\hat{\mathbf{Z}}_P$ $\mathbf{Z}_P$

Fig. 3. An intuitive example of forward estimation. The depth frame $\hat{\mathbf{Z}}_P$ is predicted from $\mathbf{Z}_I$ as the frame captured at P-frame's viewpoint virtually. After comparing $\hat{\mathbf{Z}}_P$ and $\mathbf{Z}_P$, the newly observed information in $\mathbf{Z}_P$ is outlined in yellow.

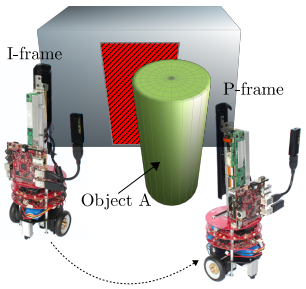


Fig. 4. An example of situations that may lead to forward estimation failures: Object A is not in the field of view of the sensor at I-frame's location. Object enters into the view of the sensor at P-frame's location, and consequently a section of the background surface becomes occluded.

1) Forward Estimation: In the forward estimation, we try to generate a depth frame $\hat{\mathbf{Z}}_P$ which is the prediction of the depth frame $\mathbf{Z}_P$. We first initialize the pixels in $\hat{\mathbf{Z}}_P$ to the invalid depth value. Then, according to Eq. 2, each pixel in depth frame $\mathbf{Z}_I$ is warped to a coordinate in $\hat{\mathbf{Z}}_P$. In this process, it can happen that two or more different depth pixels are warped to the same pixel coordinate in $\hat{\mathbf{Z}}_P$. This over-sampling issue happens because some 3D world points are occluded by the other ones at the new viewpoint. We adopt Z-buffer [38] to solve this problem. By contrast, some regions in $\hat{\mathbf{Z}}_P$ may have no valid depth information as none of the pixels in $\mathbf{Z}_I$ can be warped to these regions. We name these as *hole regions* and pixels in them have invalid depth values. They indicate that some depth information in $\mathbf{Z}_P$ cannot be predicted from $\mathbf{Z}_I$. Therefore, the regions in $\mathbf{Z}_P$ with the same locations of hole regions in $\hat{\mathbf{Z}}_P$ are assumed to be containing the newly observed depth information at time P. An example of this process is shown in Fig. 3. In this example, the regions containing the depth information that can only be observed by $\mathbf{Z}_P$ are outlined in yellow lines.

2) Reverse Check: Though the forward estimation can detect the newly observed depth information in the current frame $\mathbf{Z}_P$ in most cases, it may fail to operate correctly in the situations when some points are occluded by the objects that can only be seen in $\mathbf{Z}_P$. A typical scenario is shown in Fig. 4. In this example, as object A cannot be observed in I-frame, the forward estimation will falsely treat the background in red as the surface that can be observed in $\mathbf{Z}_P$. However, the surface of object A in green is observed in $\mathbf{Z}_P$ instead of the red background surface. Therefore, the forward estimation cannot accurately determine the newly observed depth information of $\mathbf{Z}_P$ in this case.

In order to solve this problem, we introduce a reverse check mechanism. Similar to the warping process in the forward estimation, in the reverse check process the pixels on depth frame $\mathbf{Z}_P$ can be warped to generate a depth frame, $\hat{\mathbf{Z}}_I$, which is captured I-frame's viewpoint virtually. The warping process in the reverse check can be described as

$$\begin{bmatrix} \frac{i_I - i_c}{f_x} & \frac{j_I - j_c}{f_y} & 1 & \frac{1}{z_I} \end{bmatrix}^T = \mathbf{M}^{-1} \begin{bmatrix} \frac{i_P - i_c}{f_x} & \frac{j_P - j_c}{f_y} & 1 & \frac{1}{z_P} \end{bmatrix}^T. \quad (16)$$

Pixel at (i_P, j_P) in $\mathbf{Z}_P$ can be mapped to (i_I, j_I) in $\hat{\mathbf{Z}}_I$. In this process, the pixels representing the range information of the green surface on object A will move out of the image coordinate range and will not be shown in $\hat{\mathbf{Z}}_I$. Thus we need to find the pixels in $\mathbf{Z}_P$ that move out of the image coordinate range in the reverse check process to complete the newly observed information determined by forward estimation.

3) Block-Based Update: In order to identify the newly observed depth information in the current depth frame, forward estimation and reverse check mechanisms are used in combination.

First, based on the transformation matrix $\mathbf{M}$, the forward estimation (Eq. 2) generates a virtual depth frame $\hat{\mathbf{Z}}_P$. Second, with the reverse check mechanism, $\hat{\mathbf{Z}}_I$ is generated from $\mathbf{Z}_P$ (Eq. 16). We record the coordinates of the pixels in $\mathbf{Z}_P$ which are warped out of the image coordinate range of $\hat{\mathbf{Z}}_I$. Third, we use these recorded coordinates to set the values of the corresponding pixels in $\hat{\mathbf{Z}}_P$ to holes.

Then the virtual depth frame $\hat{\mathbf{Z}}_P$ and captured frame $\mathbf{Z}_P$ are uniformly subdivided into 8×8 pixels macro blocks. We search for the blocks in $\hat{\mathbf{Z}}_P$ with the number of pixels with invalid depth values which are above a pre-determined threshold[2]. Blocks in $\hat{\mathbf{Z}}_P$ can then be classified into two groups as follows

if (number of pixels with invalid depth values in the block) $<$ threshold **then**
"*Skip block*": $\mathbf{Z}_I$ has sufficient data to predict the depth information in the block with the same coordinate in $\mathbf{Z}_P$.
else
"*Intra block*": The block with the same coordinate in frame $\mathbf{Z}_P$ contains the depth information that $\hat{\mathbf{Z}}_P$ does not have, and consequently cannot be predicted from $\mathbf{Z}_I$. The depth information in the block with the same coordinate in $\mathbf{Z}_P$ should be included in the encoding process.
end if

D. Differential Huffman Coding with Multiple Lookup Tables (DHC-M)

By using the process explained in Section II-C, we can extract the newly observed depth information from every P-frame in a depth video. However, there is still further room for improvement. As the depth frames usually contain a large number of smooth regions, there is generally a high degree of correlation between adjacent pixels. So, we can assert that a high degree of pixel-to-pixel correlation will result in a high compression ratio in differential coding. As the accurate information in keyframes [39] is the prerequisite of successful operating many applications [40], such as SLAM and 3D reconstruction, we need to keep the full fidelity of these frames

[2]Based on different compression requirements, the value of the threshold can be set as a certain portion of the total number of pixels in one block, such as $1/2$ or $1/6$. A lower threshold leads to lower compression ratios.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCSVT.2015.2416571, IEEE Transactions on Circuits and Systems for Video Technology

which are usually the I-frames in a GoP. Therefore, we propose a lossless coding scheme to encode complete I-frames and only the newly observed information in P-frames.

We designed our coding scheme, called *Differential Huffman Coding with Multiple Lookup Tables (DHC-M)*, to be fast and capable of compressing the depth images in a lossless manner without introducing any artificial refinements. DHC-M scheme can be applied on complete depth frames or individual blocks.

The DHC-M scheme operates by comparing the current depth pixel with its reference pixel (which is the neighboring pixel of the pixel under consideration). If a pixel is at the end of a row, then it is treated as the reference pixel of the pixel at the same column under it (see Fig. 5).

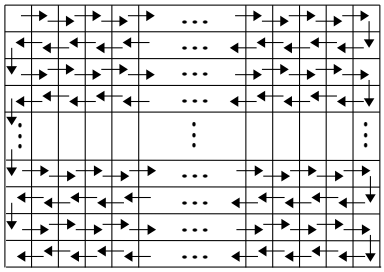


Fig. 5. Depth pixels and their reference pixels in a frame.

The difference between the values of a pixel and its reference pixel is used in the encoding process. However, the depth image captured by RGB-D sensors has a portion of pixels that have invalid depth value (holes). The invalid value is set at 2047 which is the largest number can be represented by 11-bit data. This characteristic make the difference between the current and reference pixels have a high dynamic range. If we directly apply the standard entropy coding on the difference values, the compression ratio will be very limited. It is because the probabilities of difference values are distributed in a very wide range. In order to efficiently encode these difference values, we centralize the probability distribution by classifying the difference values into two groups according to the value of the reference pixel and using two different lookup tables to encode the differences.

Let X_i represent the current pixel. If it has a valid depth value, we use the symbol V_i, or H_i, if it is a hole. Similarly for its reference pixel, we use the symbols either V_{i-1} or H_{i-1} depending on whether it is a hole or not.

The first lookup table is used for the cases when a reference pixel is a hole, and its values are generated according to the conditional probabilities as follows

$$p(X_i|H_{i-1}) = \begin{cases} p(V_i|H_{i-1}) & \text{if the current pixel is valid} \\ p(H_i|H_{i-1}) & \text{otherwise,} \end{cases} \tag{17}$$

where $p(V_i|H_{i-1}) = q_h(V_i)$, and $p(H_i|H_{i-1}) = q_h(H_i - H_{i-1})$. The second lookup table is for the cases that where the reference pixel has a valid depth value, and is established based on the conditional probabilities as follows

$$p(X_i|V_{i-1}) = \begin{cases} p(V_i|V_{i-1}) & \text{if the current pixel is valid} \\ p(H_i|V_{i-1}) & \text{otherwise,} \end{cases} \tag{18}$$

where $p(V_i|V_{i-1}) = q_v(V_i - V_{i-1})$, and $p(H_i|V_{i-1}) = q_v(H_i)$. We use the standard technique of Huffman coding [41] to generate two lookup tables based on the probability distributions of $q_h()$ and $q_v()$. Probability distributions of $q_h()$ and $q_v()$ are determined empirically by collecting information over a number of representative depth images which are captured in scenarios with varying geometrical structures.

In the encoding stage, the scheme first determines whether a reference pixel is a hole. If it is, the first lookup table (generated by using Eq. 17) is used to determine the codeword of the difference value. Otherwise, the second lookup table (Eq. 18) is used instead.

E. Decoding Process and Under-Sampling Problem

At the decoder side, the received bitstream is decoded with the same lookup tables used at the encoder side. In the decoding process, if the reference pixel is a hole, the first lookup table is used to decode the current pixel, otherwise the second table is used. As no motion compensation is applied on I-frame, every I-frame can decoded losslessly. While after motion compensation, only some blocks of each P-frame are transmitted with the transformation matrix. Therefore, each P-frame is lossy encoded and need to be decoded using its corresponding I-frame and transformation matrix besides transmitted blocks. We first apply the transformation matrix on each pixel in I-frame (use Eq. 2) to generate the prediction of the P-frame, $\hat{\mathbf{Z}}_P$. Then, we paste the transmitted blocks on $\hat{\mathbf{Z}}_P$. Eventually, P-frame is reconstructed at decoder side.

Directly applying warping equations may cause some visual artifacts in the synthesized view, like disocclusions and cracks. Some remarkable works [42], [43], [44], [38] have been proposed to reduce these adverse effects, especially the effects of disocclusions. In our system, the disocclusions effects are fixed using information of original P-frames in block-based update process. Here, we only need to deal with the cracks which generated due to the *under-sampling problem*. Under-sampling occurs when the warping process "rotates" a surface in such a way that the viewing angle gets closer to the normal or "reduces" the distance between the sensor and the surface (Fig. 6). In such situations, the original image does not have enough information to predict the same surface from the viewpoint of the generated image.

A distinguishing feature of the pixels in cracks is that they have invalid depth value and their neighboring pixels usually have valid depth values. There are two general approaches to fill the cracks. The first approach is to use the value of the nearest non-hole pixel in the left/right side to fill the pixels in the cracks [31], [24]. This method is inaccurate and not robust for scenes that contain objects with complex geometries and variable camera motions. The second approach uses a median filter to smooth the whole images [38], [43]. This approach exhibits good performance on filling the cracks, but it smoothes the complete image and introduces noise in regions

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCSVT.2015.2416571, IEEE Transactions on Circuits and Systems for Video Technology

with correct depth values, especially on the object boundaries. In order to avoid this adverse effect, we have modified it by using an adaptive median filter. The filter is only applied on the pixels with invalid depth value instead of the whole image. Detailed performance evaluation of these three crack-filling algorithms is presented in Section III-B2.

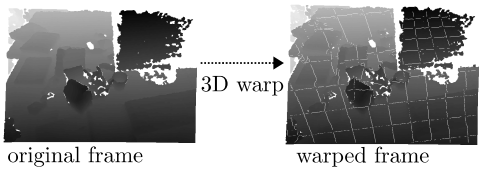


Fig. 6. Cracks can be introduced during the image warping process due to the under-sampling problem.

III. Performance Evaluation

We have implemented the IW-DVC framework in C++ using libCVD [45], OpenCV [46], and OpenKinect [47] libraries on a personal computer with an Intel i7-M620 2.66 GHz processor and 4GB memory. In order to evaluate its performance we have conducted a series of experiments by using datasets selected archive provided by the Computer Vision Group in Technical University of Munich [48]. All datasets in this archive consist of color and depth images captured by a mobile RGB-D sensor. The datasets also provide the intrinsic parameters of the RGB-D sensor used. All depth images were captured at a rate of 30 fps (full frame rate) with a resolution of 640×480 pixels. The pixels are trained to represent the distance in meters. For our experiments, we have selected the data sets containing images captured over static scenes with sufficiently rich geometrical features. The archival names of the selected datasets are included in Table II. Before conducting the experiments, we have converted the depth images in these datasets back into their raw values so that each pixel has 2048 levels of sensitivity.

The two major components of the IW-DVC framework, motion compensation and the lossless coding scheme DHC-M have been tested separately. DHC-M has been tested compared against the JPEG2000 lossless mode and context-adaptive binary arithmetic coding (CABAC) [49]. In the second set of experiments, the performance of the proposed motion compensation approach has been compared to the first motion compensation algorithm for depth video that uses 2D block-based motion vector sharing (2D-BMS) [13] and recently developed 3D block-based motion vector sharing approach (3D-BMS) [18].

There are two main kinds of approaches to evaluate the quality of the reconstructed depth images: (1) subjective, or (2) objective methods, such as peak signal-to-noise ratio (PSNR) and the structural similarity (SSIM). Subjective methods [50], [51] require human viewers to train the assessment program and judge the quality of the reconstructed images. However, different from the conventional 8-bit depth images, the pixels in the depth images captured by RGB-D sensors contain 11-bit data. Original depth information in these images cannot be seen directly. If we downscale them to 8 bits per pixel before using subjective evaluation methods, a certain amount of information is inevitably lost. Therefore, the subjective methods are not applicable in this project, and consequently we have adopted objective methods to evaluate performances.

In the second set of experiments, we have used both PSNR and SSIM to evaluate the quality of the reconstructed depth images. We also have adopted the entropy approach to measure the differences between the reconstructed image and the uncompressed original image. Then, we have analyzed the performance of different crack-filling algorithms. In the third set of the experiments, we have analyzed the encoding complexity based on the processing time for each step and presented time complexity of each step in big O notation. For the latter three sets of experiments, the GoP size was set to 10 and the block size was 8×8.

A. Performance Evaluation of DHC-M Scheme

In the first set of experiments, in order to evaluate its effectiveness, we compared the DHC-M scheme against the standard JPEG2000 lossless mode and CABAC. Here, we have tested it alone without running the motion compensation block (Fig. 1(a)).

DHC-M achieves its best performance when the individualized probability distributions are used to build lookup tables in different scenes. However, DHC-M cannot be universally used in this case. In this experiment, we randomly sampled 30 images in each dataset and generated lookup tables of DHC-M accordingly. So the same lookup tables can be used for all datasets, and DHC-M is able to achieve its near-best performance. The results are presented in Table II.

The size of a single uncompressed depth image is 614.4 KB. It is obvious that the DHC-M scheme outperforms JPEG2000 lossless mode and CABAC in all datasets. The average compression ratio of DHC-M is 18.7% higher than the average compression ratio of JPEG2000 and 127.9% higher than the average compression ratio of CABAC. It is because the images captured by RGB-D sensors have a much higher dynamic range than conventional gray-scale images and the pixels' values switch between the valid values and the invalid value (the largest value) frequently. DHC-M, which takes this feature into account and uses two lookup tables alternatively, centralizes the probability distribution of the difference values and achieves better context modeling. While JPEG2000 and CABAC, which are not specially designed for RGB-D sensor, do not consider this property. Furthermore, in lossless mode, JPEG2000 use a reversible wavelet transform and no quantization is performed. Thus, all bits planes have to be encoded, JPEG2000 can hardly achieve the best performance.

B. Objective Evaluations of Reconstructed Depth Images

PSNR and SSIM can only indicate the degree of closeness between the reconstructed and original depth images. As the depth images captured by RGB-D sensors almost always have a number of pixels with invalid depth value, we need a quality evaluation method which can describe the noises of different kinds of pixels separately. Therefore, we use the entropy of the difference between the reconstructed depth image and the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCSVT.2015.2416571, IEEE Transactions on Circuits and Systems for Video Technology

TABLE II
COMPRESSION PERFORMANCE OF JPEG2000, CABAC, AND DHC-M.

DATASET		JPEG2000		CABAC		DHC-M	
ARCHIVE NAME	ID	AVERAGE SIZE (KB)	COMP. RATIO	AVERAGE SIZE (KB)	COMP. RATIO	AVERAGE SIZE (KB)	COMP. RATIO
freiburg1_plant	1	72.43	8.48	148.48	4.14	58.42	10.52
freiburg2_dishes	2	68.34	9.00	140.84	4.36	54.91	11.19
freiburg3_cabinet	3	60.02	10.24	118.72	5.18	51.40	11.95
freiburg3_large_cabinet	4	57.47	10.69	105.62	5.82	50.74	12.11
freiburg3_structure_texture_far	5	61.60	9.97	123.91	4.96	52.31	11.75
freiburg3_long_office_household	6	68.51	8.97	107.93	5.69	56.23	10.93
freiburg1_xyz	7	62.74	9.79	127.88	4.80	54.76	11.22

TABLE III
QUALITY EVALUATION OF THE RECONSTRUCTED DEPTH IMAGES GENERATED BY IW-DVC FRAMEWORK WITH USING DIFFERENT BLOCK UPDATE THRESHOLDS.

DATA SET	THRES HOLD	COMP. RATIO	PSNR	SSIM	ENTROPY		AVG. NUMBER OF BITS REQUIRED TO CORRECT THE ERRORS IN A RECONSTRUCTED DEPTH IMAGE (ANBR)		
					PIXELS WITH VALID VALUE	HOLES	PIXELS WITH VALID VALUE	HOLES	OVERALL
1	1/2	594	33.53	0.8356	1.49	1.07	3.46×10^5	8.03×10^4	4.36×10^5
	1/3	285	34.79	0.8585	1.45	0.81	3.38×10^5	6.28×10^4	4.00×10^5
	1/6	139	35.91	0.8818	1.44	0.65	3.34×10^5	4.78×10^4	3.82×10^5
2	1/2	590	35.66	0.8780	1.28	1.19	3.29×10^5	5.93×10^4	3.89×10^5
	1/3	374	37.88	0.8942	1.25	0.82	3.22×10^5	4.01×10^4	3.62×10^5
	1/6	256	38.71	0.9010	1.21	0.62	3.12×10^5	3.02×10^4	3.42×10^5
3	1/2	869	27.15	0.7254	1.47	1.92	3.79×10^5	9.53×10^4	4.74×10^5
	1/3	352	29.18	0.8144	1.45	1.75	3.73×10^5	8.73×10^4	4.60×10^5
	1/6	269	31.05	0.8321	1.40	1.53	3.60×10^5	7.73×10^4	4.38×10^5
4	1/2	395	42.87	0.9256	1.16	0.61	2.62×10^5	4.93×10^4	3.12×10^5
	1/3	268	43.26	0.9296	1.12	0.45	2.54×10^5	3.61×10^4	2.90×10^5
	1/6	211	43.88	0.9347	1.07	0.31	2.41×10^5	2.54×10^4	2.67×10^5
5	1/2	829	41.77	0.8770	1.13	0.61	2.99×10^5	2.59×10^4	3.25×10^5
	1/3	415	42.49	0.8883	1.11	0.55	2.93×10^5	2.34×10^4	3.16×10^5
	1/6	291	44.05	0.9040	1.09	0.48	2.88×10^5	2.26×10^4	3.09×10^5
6	1/2	625	46.51	0.9256	1.00	0.93	2.54×10^5	4.89×10^4	3.03×10^5
	1/3	342	48.06	0.9399	0.98	0.75	2.51×10^5	3.91×10^4	2.90×10^5
	1/6	220	49.26	0.9479	0.97	0.64	2.48×10^5	3.33×10^4	2.81×10^5
7	1/2	357	35.49	0.8117	1.25	1.45	2.86×10^5	1.14×10^4	4.01×10^5
	1/3	250	36.09	0.8303	1.20	1.26	2.75×10^5	9.96×10^4	3.74×10^5
	1/6	180	37.63	0.8592	1.13	0.96	2.58×10^5	7.49×10^4	3.33×10^5

original depth image to evaluate the performance of the IW-DVC framework. As GOP size was 10, the system encoded one complete depth frame (I-frame) losslessly in every 10 frames. The redundancies in the other depth frames (P-frames) of the same group were removed by our motion compensation approach presented in Section II-C. Then, only the newly observed depth information in these frames were coded by DHC-M.

1) System without Crack-Filling Algorithm: The objective of this experiment is to evaluate the effects of different block update thresholds on compression ratio and quality of the reconstructed depth image without implementing the crack-filling algorithm at the decoder side. Three thresholds were tested, which are 1/2, 1/3, and 1/6 of the overall number of pixels in one block. Two entropies for holes and pixels with valid values in the reconstructed depth images were derived. The entropies indicate the theoretical number of bits required to describe the difference between one pixel in the reconstructed depth frame and its corresponding pixel in the original uncompressed depth frame. We also derived the average number of bits required to correct the errors in the holes and pixels with valid values in one complete reconstructed depth frame to the original one (ANBR). The smaller ANBR, the higher reconstructed quality is achieved. The average entropies and ANBRs of seven datasets are presented in Table III besides PSNR and SSIM.

According to Table III, the compression ratio decreases when the threshold drops. It is because more intra blocks need to be updated and encoded if the threshold is low. In the meantime, when more blocks in the original depth frame are updated, fewer prediction errors exist in the reconstructed depth image. Thus, both entropies decline along with the threshold, and ANBR decreases simultaneously.

To verify the superiority of our system, we have implemented 2D-BMS [13] and 3D-BMS [18] on DHC-M and tested them on the same datasets. 2D-BMS algorithm uses the motion vectors derived from the texture information to encode both color and depth video. 3D-BMS, developed recently, uses variable block sizes to perform motion estimation. In addition, it estimates motion vector in z direction and reported good reconstruction quality on the depth video captured by a static camera. The results are presented in Tables IV and V.

By comparing the results in Tables III, V, and IV, it is clear that the 3D-BMS has enhanced accuracy than 2D-BMS, and our proposed motion compensation scheme with update threshold at 1/3 and 1/6 outperforms 2D-BMS and 3D-BMS in reconstruction accuracy. Since in each dataset, our system has much smaller entropies and ANBR. PSNR and SSIM

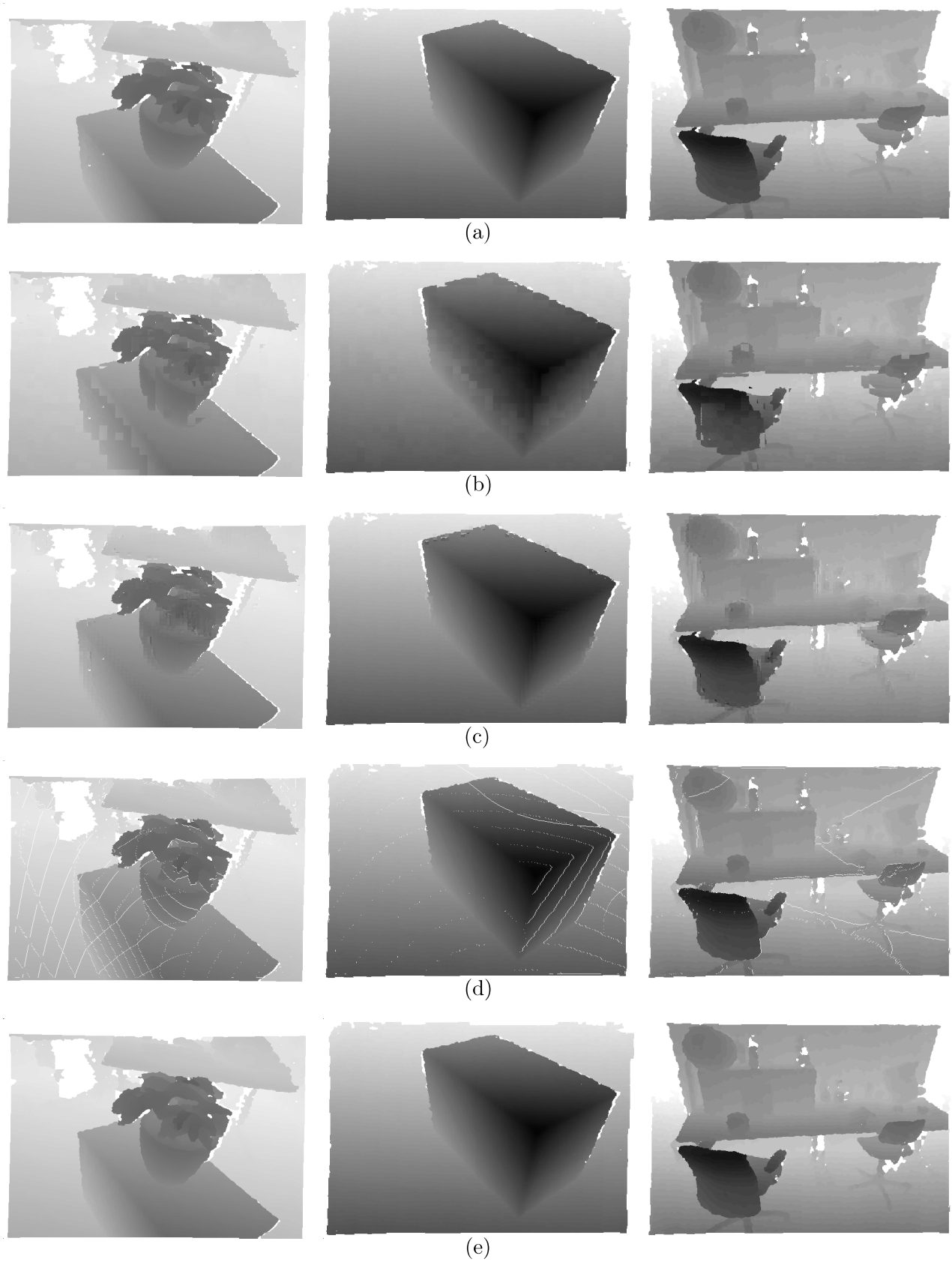


Fig. 7. Examples of the original depth images and their reconstructed counterparts in datasets 1, 3 and 6: (a) Originals; (b) Reconstructed depth images generated by 2D-BMS; (c) Reconstructed depth images generated by 3D-BMS; (d) Reconstructed depth images generated by the IW-DVC framework with an update threshold of $1/3$; (e) Enhancements of depth images in (d) by 3^{rd} crack-filling algorithm.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCSVT.2015.2416571, IEEE Transactions on Circuits and Systems for Video Technology

TABLE IV
QUALITY EVALUATION OF THE RECONSTRUCTED DEPTH IMAGES GENERATED BY 2D-BMS.

DATA SET	COMP. RATIO	PSNR	SSIM	ENTROPY		AVG. NUMBER OF BITS REQUIRED TO CORRECT THE ERRORS IN A RECONSTRUCTED DEPTH IMAGE (ANBR)		
				PIXELS WITH VALID VALUE	HOLES	PIXELS WITH VALID VALUE	HOLES	OVERALL
1	156	23.92	0.7321	2.82	1.84	6.54×10^5	1.39×10^5	7.93×10^5
2	265	24.27	0.8256	1.78	1.66	4.60×10^5	8.27×10^4	5.42×10^5
3	214	21.65	0.5471	1.95	1.46	5.04×10^5	7.13×10^4	5.75×10^5
4	220	35.76	0.8562	1.19	1.10	2.68×10^5	8.99×10^4	3.58×10^5
5	239	24.62	0.7707	2.73	0.77	7.33×10^5	2.95×10^4	7.62×10^5
6	189	42.21	0.9068	1.06	1.06	2.70×10^5	5.65×10^4	3.27×10^5
7	166	26.44	0.7759	2.38	1.36	5.53×10^5	1.02×10^5	6.55×10^5

TABLE V
QUALITY EVALUATION OF THE RECONSTRUCTED DEPTH IMAGES GENERATED BY 3D-BMS.

DATA SET	COMP. RATIO	PSNR	SSIM	ENTROPY		AVG. NUMBER OF BITS REQUIRED TO CORRECT THE ERRORS IN A RECONSTRUCTED DEPTH IMAGE (ANBR)		
				PIXELS WITH VALID VALUE	HOLES	PIXELS WITH VALID VALUE	HOLES	OVERALL
1	118	28.68	0.7894	1.73	1.30	4.04×10^5	9.59×10^4	5.00×10^5
2	136	29.65	0.8730	1.40	0.95	3.64×10^5	4.67×10^4	4.11×10^5
3	149	27.06	0.7125	1.63	1.38	4.22×10^5	6.76×10^4	4.89×10^5
4	124	39.29	0.8723	1.05	1.04	2.37×10^5	8.54×10^4	3.22×10^5
5	125	33.62	0.8107	2.09	0.73	5.60×10^5	2.79×10^4	5.88×10^5
6	134	47.19	0.9184	0.99	0.93	2.51×10^5	5.06×10^4	3.01×10^5
7	115	32.18	0.8360	1.63	1.18	3.80×10^5	8.86×10^4	4.68×10^5

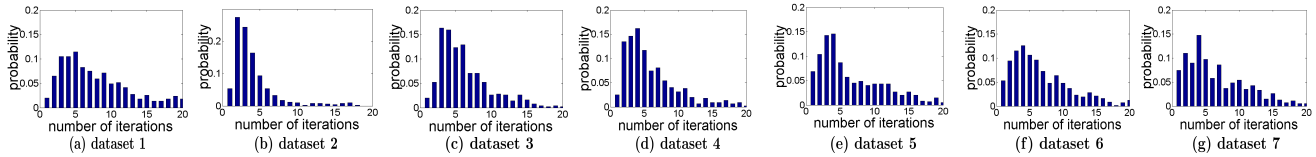


Fig. 8. Probability distributions of the number of iterations required for each of the 7 datasets.

results of our proposed methods are also higher than the other two algorithms. Moreover, our proposed method has higher compression ratios than 2D-BMS and 3D-BMS. One reason is that our approach only needs one 6D motion vector to describe the inter-frame motion. However, the conventional block-based motion compensation methods require a motion vector for each block.

We have selected three representative examples of results with noticeable differences/artifacts (Fig. 7). The scenarios in these three examples contain both smooth surfaces and abruptly changing object boundaries to allow us to check the performance of algorithms in a wide range of conditions. The smooth surfaces in the original images were reconstructed as coarse surfaces with block-artifacts by 2D-BMS (see the surface of the cabinet in the middle and left images of Fig. 7(b)). It is because the depth information of the same surface changes in the consecutive frames captured by a mobile RGB-D sensor, while 2D-BMS does not consider this issue and does not change the values of the pixels representing the same surface in consecutive frames. 3D-BMS, which has an extra motion vector on z direction and uses variable block sizes, achieves relatively good performance in dealing with this problem. However, it still generates block artifacts on object boundaries. It is because different surfaces on the object boundaries can be placed in the same block. The distance changes on different surfaces are different. Then, the same motion vector on z direction is used for all the pixels in the same block, which leads to incorrect prediction on object boundaries. Therefore, we can see the block artifacts on the smooth surfaces are removed generally in the middle image of Fig. 7(c). But a large number of artifacts and blurs appear on the object boundaries in the left and right images of Fig. 7(c), when the scenes have complex geometrical structures. IW-DVC framework, which uses a 6D motion vector, not only maps each pixel from the reference frame to the predicted frame, but also changes its value individually. Therefore, the IW-DVC scheme does not introduce these block-artifacts and nicely preserves the object boundary information (see Fig. 7(d)). The results of this experiment accord our analysis in the beginning of this paper that the block-based motion compensation schemes are suboptimal for depth video captured by a mobile sensor. However, in Fig. 7(d), it is obvious that some cracks regularly appear in the reconstructed depth images. This drawback of our motion compensation algorithm can be overcome easily by a crack-filling algorithm which is addressed in the next section.

2) System with Crack-Filling Algorithm: In this experiment, we focus on determining the best performing crack-

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TCSVT.2015.2416571, IEEE Transactions on Circuits and Systems for Video Technology

TABLE VI
QUALITY EVALUATION OF THE RECONSTRUCTED DEPTH IMAGES ENHANCED BY DIFFERENT CRACK-FILLING ALGORITHMS, UPDATE THRESHOLD $= 1/3$.

DATA SET	METHOD	ACCURACY PERCENT	PSNR	SSIM	ENTROPY		AVG. NUMBER OF BITS REQUIRED TO CORRECT THE ERRORS IN A RECONSTRUCTED DEPTH IMAGE (ANBR)		
					PIXELS WITH VALID VALUE	HOLES	PIXELS WITH VALID VALUE	HOLES	OVERALL
1	1	48.7%	33.21	0.8645	1.62	0.04	3.95×10^5	2.48×10^4	3.97×10^5
	2	81.5%	35.77	0.8956	1.46	0.30	3.45×10^5	2.20×10^4	3.67×10^5
	3	81.5%	36.05	0.8985	1.45	0.23	3.44×10^5	1.67×10^4	3.61×10^5
2	1	20.2%	34.75	0.9286	1.30	0.05	3.43×10^5	2.10×10^3	3.45×10^5
	2	83.0%	39.41	0.9665	1.24	0.26	3.22×10^5	1.23×10^4	3.34×10^5
	3	83.0%	39.87	0.9676	1.25	0.19	3.25×10^5	9.08×10^3	3.33×10^5
3	1	26.6%	29.72	0.7167	1.65	0.09	4.41×10^5	3.64×10^4	4.44×10^5
	2	91.2%	32.67	0.8478	1.51	0.84	3.90×10^5	4.06×10^4	4.31×10^5
	3	91.2%	34.03	0.8574	1.46	0.34	3.79×10^5	1.66×10^4	3.96×10^5
4	1	18.8%	42.01	0.9520	1.17	0.06	2.77×10^5	4.07×10^3	2.82×10^5
	2	88.6%	46.27	0.9744	1.13	0.11	2.60×10^5	8.58×10^3	2.69×10^5
	3	88.6%	48.55	0.9776	1.13	0.07	2.59×10^5	5.47×10^3	2.65×10^5
5	1	32.5%	42.69	0.9179	1.17	0.05	3.18×10^5	1.92×10^3	3.20×10^5
	2	86.4%	43.77	0.9483	1.12	0.18	3.00×10^5	7.38×10^3	3.07×10^5
	3	86.4%	44.92	0.9506	1.12	0.14	2.99×10^5	5.72×10^3	3.05×10^5
6	1	42.4%	44.21	0.9184	1.05	0.06	2.75×10^5	2.60×10^3	2.78×10^5
	2	88.9%	46.99	0.9643	0.99	0.41	2.54×10^5	2.10×10^4	2.75×10^5
	3	88.9%	49.67	0.9679	0.99	0.26	2.54×10^5	1.33×10^4	2.67×10^5
7	1	31.9%	36.98	0.8611	1.32	0.16	3.22×10^5	1.07×10^4	3.33×10^5
	2	85.5%	37.50	0.8970	1.22	0.46	2.88×10^5	3.37×10^4	3.21×10^5
	3	85.5%	38.16	0.8993	1.21	0.41	2.84×10^5	3.00×10^4	3.14×10^5

filling algorithm in recovering depth information of the cracks created due to the under-sampling problem. Based on the results of the experiments presented in Section III-B1, we have chosen 1/3 as the update threshold in this experiment by considering both compression ratio and the quality of the reconstructed depth images. We have tested the three crack-filling algorithms described in Section II-E. Comparison results are presented in Table VI. Percent accuracy of each method has been calculated as follows

$$accuracy = \frac{correctly\_filled\_pixels}{\max(pixels\_need\_filling, filled\_pixels)} \times 100\%.$$

Tables III and VI show that the third method has the highest percent accuracy, PSNR, and SSIM. The first method, which considers the horizontal neighboring pixels, is only suitable for cracks generated by a horizontally rotating and translating camera. However, in these datasets, the mobile camera moves and rotates in all directions. The second method has a similar performance with the third one. But, as it is applied on complete images, it smoothes the overall image and introduces noise in areas without cracks. Because of this, the entropy of holes decreases while the entropy of pixels with valid values increases. An important observation to make here is that the application of third method leads to a significant reduction of prediction errors while maintaining a high compression ratio. The third method has higher ANBR, PSNR, and SSIM values, evan though it uses an update threshold of 1/3 (Table VI) compared to 1/6 (Table III). Thus, we adopt the third crack-filling algorithm at the decoder side to enhance the quality of the reconstructed images and preserve high compression ratios. The corresponding enhanced results are shown in Fig. 7(e).

C. Encoding Complexity Analysis

In this set of experiments, we present time complexity of each step in big O notation and further analyze encoding complexity based on the processing time of each step. We implemented the encoder with the update threshold at 1/3 on our computer. We measured the average processing time of each step required for encoding one frame in milliseconds. The results are summarized in Table VII.

In Fig. 8, we present seven histograms to illustrate probability distributions of the numbers of iterations in different datasets. We conducted extra experiments and found there is not any direct relation between the number of sample points and the required number of iterations. We also list the average number of iterations required for convergence in the process of inter-frame motion estimation in Table VII.

In Table VII, n_w and n_s represent the number of pixels in the searching window and the number of sampled points in inter-frame motion estimation process. N_f represents the number of pixels in one complete frame. As $N_f \gg n_w n_s$, the time complexity of our motion compensation algorithm is $O(N_f)$. The time complexity of 2D-BMS is $O(N_w N_f)$. N_w indicates the number of different block positions in the matching window ($N_w \gg n_w$). The time complexity of 3D-BMS is also $O(N_w N_f)$. Without estimating MVs in a block by block manner, our proposed motion compensation approach with linear time complexity is more efficient than 2D-BMS and 3D-BMS. Moreover, according to Table VII, it is clear that the depth video can be encoded up to around 25 fps in real time on a standard computer.

IV. CONCLUDING REMARKS

We have presented a novel coding framework, called 3D image warping based depth video compression (IW-DVC), for efficiently removing the existing redundancy in the depth video frames captured by a mobile RGB-D sensor. In particular, the motion compensation scheme included in the framework, designed to exploit the unique characteristics of depth images,

TABLE VII
PROCESSING TIME OF THE PROPOSED FRAMEWORK IN EACH STEP FOR VARIOUS DATASETS.

DATASET	NUMBER OF ITERATIONS	INTERFRAME MOTION ESTIMATION	FORWARD ESTIMATION/ REVERSE CHECK	BLOCK UPDATE	DHC-M	OVERALL
1	7.54	14.45 (31.12%)	26.95 (58.04%)	1.15 (2.48%)	3.88 (8.36%)	46.43
2	4.01	12.72 (30.22%)	24.10 (57.26%)	1.14 (2.71%)	4.13 (9.81%)	42.09
3	6.21	11.87 (27.29%)	25.51 (58.66%)	1.58 (3.63%)	4.53 (10.42%)	43.49
4	5.95	11.64 (28.16%)	24.82 (60.05%)	1.20 (2.90%)	3.67 (8.88%)	41.33
5	6.58	10.55 (24.11%)	27.98 (63.94%)	1.12 (2.56%)	4.11 (9.39%)	43.76
6	6.63	9.31 (23.32%)	25.26 (63.28%)	1.18 (2.96%)	4.17 (10.45%)	39.92
7	6.57	14.36 (33.43%)	23.38 (54.20%)	1.32 (3.07%)	3.99 (9.29%)	42.95
Average	6.21	*12.13 (28.31%)*	*25.41 (59.30%)*	*1.24 (2.89%)*	*4.07 (9.50%)*	*42.85*
Time complexity	—	$O(n_w n_s)$	$O(N_f)$	$O(N_f)$	$O(N_f)$	$O(N_f)$

and works cooperatively with the egomotion estimation and 3D image warping.

Experimental results show that our motion compensation method reduces the prediction errors of 2D-BMS and 3D-BMS for depth video captured by a mobile sensor to 55.9% and 72.8% on average, respectively. Also, they demonstrate that IW-DVC framework is capable of keeping the quality of the reconstructed depth image at a high level and can accurately determine the newly observed depth information in each frame. This significantly enhances the compression ratio. Furthermore, the results show that the IW-DVC framework is capable of operating in real time. As a final note, with the losslessly encoded I-frames, IW-DVC is suitable for many applications that have high requirements on the accuracy of keyframes.

REFERENCES

[1] B. Freedman, A. Shpunt, M. Machline, and Y. Arieli, "Depth Mapping Using Projected Patterns," US Patent App. 2010/0118123 A1, 2010.

[2] "PrimeSense: 3D Sensing Technology Solutions," http://www.primesense.com.

[3] E. E. Hitomi, J. V. da Silva, and G. C. Ruppert, "3D Scanning Using RGB-D Imaging Devices: A Survey," *Computational Vision and Medical Image Processing IV: VIPIMAGE 2013*, pp. 197–202, 2013.

[4] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox, "RGB-D Mapping: Using Kinect-Style Depth Cameras for Dense 3D Modeling of Indoor Environments," in *The International Journal of Robotics Research*, vol. 31, 2012, pp. 647–663.

[5] F. Endres, J. Hess, J. Sturm, D. Cremers, and W. Burgard, "3-D Mapping with an RGB-D Camera," *IEEE Transactions on Robotics*, vol. 30, no. 1, pp. 177–187, Feb 2014.

[6] R. A. Newcombe, A. J. Davison, S. Izadi, P. Kohli, O. Hilliges, J. Shotton, D. Molyneaux, S. Hodges, D. Kim, and A. Fitzgibbon, "KinectFusion: Real-Time Dense Surface Mapping and Tracking," in *Proceedings of the IEEE International Symposium on Mixed and Augmented Reality (ISMAR 2011)*, Basel, Switzerland, 2011, pp. 127–136.

[7] S. Kim and J. Kim, "Occupancy Mapping and Surface Reconstruction Using Local Gaussian Processes with Kinect Sensors," *IEEE Transactions on Cybernetics*, vol. 43, no. 5, pp. 1335–1346, 2013.

[8] D. Taubman and M. Marcellin, Eds., *JPEG2000: Image Compression Fundamentals, Standards and Practice*. Springer, 2002.

[9] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H. 264/AVC Video Coding Standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 560–576, 2003.

[10] A. Puri, H.-M. Hang, and D. Schilling, "An efficient Block-Matching Algorithm for Motion-Compensated Coding," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 1987)*, vol. 12, Dallas, USA, 1987, pp. 1063–1066.

[11] R. Krishnamurthy, B.-B. Chai, H. Tao, and S. Sethuraman, "Compression and Transmission of Depth Maps for Image-Based Rendering," in *Proceedings of the International Conference on Image Processing (ICIP 2001)*, vol. 3, Thessaloniki, Greece, 2001, pp. 828–831 vol.3.

[12] B.-B. Chai, S. Sethuraman, H. S. Sawhney, and P. Hatrack, "Depth Map Compression for Real-Time View-Based Rendering ," *Pattern Recognition Letters*, vol. 25, no. 7, pp. 755 – 766, 2004.

[13] S. Grewatsch and E. Miiller, "Sharing of Motion Vectors in 3D Video Coding," in *Proceedings of the International Conference on Image Processing (ICIP 2004)*, vol. 5, Singapore, 2004, pp. 3271–3274.

[14] C. Hewage, S. T. Worrall, S. Dogan, and A. M. Kondoz, "A Novel Frame Concealment Method for Depth Maps Using Corresponding Colour Motion Vectors," in *Proceedings of the 3DTV Conference: The True Vision - Capture, Transmission and Display of 3D Video*, Istanbul, Turkey, 2008, pp. 149–152.

[15] I. Daribo, C. Tillier, and B. Pesquet-Popescu, "Motion Vector Sharing and Bitrate Allocation for 3D Video-Plus-Depth Coding," *EURASIP Journal on Advances in Signal Processing*, vol. 2009, pp. 3:1–3:13, Jan 2009.

[16] S. Shahriyar, M. Murshed, M. Ali, and M. Paul, "Inherently Edge-Preserving Depth-Map Coding Without Explicit Edge Detection and Approximation," in *2014 IEEE International Conference on Multimedia and Expo Workshops*, July 2014, pp. 1–6.

[17] B. Kamolrat, W. A. C. Fernando, M. Mrak, and A. Kondoz, "3D Motion Estimation for Depth Image Coding in 3D Video Coding," *IEEE Transactions on Consumer Electronics*, vol. 55, no. 2, pp. 824–830, 2009.

[18] Y.-C. Fan, S.-F. Wu, and B.-L. Lin, "Three-Dimensional Depth Map Motion Estimation and Compensation for 3D Video Compression," *IEEE Transactions on Magnetics*, vol. 47, no. 3, pp. 691–695, March 2011.

[19] V.-A. Nguyen, D. Min, and M. Do, "Efficient Techniques for Depth Video Compression Using Weighted Mode Filtering," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 23, no. 2, pp. 189–202, 2013.

[20] J. Zhang, M. Hannuksela, and H. Li, "Joint Multiview Video Plus Depth Coding," in *Proceedings of the IEEE International Conference on Image Processing (ICIP 2010)*, Hong Kong, China, 2010, pp. 2865–2868.

[21] H. Schwarz, D. Marpe, and T. Wiegand, "Overview of the Scalable Video Coding Extension of the H.264/AVC Standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 17, no. 9, pp. 1103–1120, 2007.

[22] C. Bilen, A. Aksay, and G. Akar, "A Multi-View Video Codec Based on H.264," in *Proceedings of the IEEE International Conference on Image Processing (ICIP 2006)*, Atlanta, USA, 2006, pp. 541–544.

[23] E. Ekmekçioğlu, S. Worrall, and A. Kondoz, "A Temporal Subsampling Approach for Multiview Depth Map Compression," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 19, no. 8, pp. 1209–1213, 2009.

[24] J. Y. Lee, H.-C. Wey, and D.-S. Park, "A Fast and Efficient Multi-View Depth Image Coding Method Based on Temporal and Inter-View Correlations of Texture Images," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 21, no. 12, pp. 1859–1868, 2011.

[25] C. Fehn, "Depth-Image-Based rendering (DIBR), Compression, and Transmission for a New Approach on 3D-TV," in *Electronic Imaging*. International Society for Optics and Photonics, 2004, pp. 93–104.

[26] C. Lee, B. Choi, and Y.-S. Ho, "Efficient Multiview Depth Video Coding Using Depth Synthesis Prediction," *Optical Engineering*, vol. 50, no. 7, pp. 077 004–077 004–14, 2011.

[27] W. R. Mark, "Post-Rendering 3D Image Warping: Visibility, Reconstruction and Performance for Depth-Image Warping," Ph.D. dissertation, University of North Carolina at Chapel Hill, 1999.

[28] H.-M. Wang, C.-H. Huang, and J.-F. Yang, "Block-Based Depth Maps Interpolation for Efficient Multiview Content Generation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 21, no. 12, pp. 1847–1858, 2011.

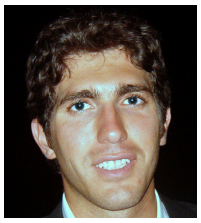
[29] C. Fehn, "Depth-Image-Based Rendering (DIBR), Compression, and

Transmission for a New Approach on 3D-TV," in *Proceedings of SPIE*, vol. 5291, 2004, pp. 93–104.
[30] C. Zhu, Y. Zhao, L. Yu, and M. Tanimoto, "3D-TV System with Depth-Image-Based Rendering: Architectures, Techniques and Challenges," *Springer*, 2013.
[31] W. R. Mark, L. McMillan, and G. Bishop, "Post-Rendering 3D Warping," in *Proceedings of the Symposium on Interactive 3D Graphics*, Providence, USA, 1997, pp. 7–ff.
[32] "Wireless Sensor and Robot Networks Laboratory (WSRNLab)," http://wsrnlab.ecse.monash.edu.au.
[33] B. Freedman, A. Shpunt, M. Machline, and Y. Arieli, "Depth Mapping Using Projected Patterns," USA Application US20 080 240 502 A1, 2007.
[34] X. Wang, Y. A. Şekercioğlu, and T. Drummond, "A Real-Time Distributed Relative Pose Estimation Algorithm for RGB-D Camera Equipped Visual Sensor Networks," in *Proceedings of the 7th ACM/IEEE International Conference on Distributed Smart Cameras (ICDSC 2013)*, Palm Springs, USA, 2013.
[35] X. Wang, Y. A. ekerciolu, and T. Drummond, "Vision-Based Cooperative Pose Estimation for Localization in Multi-Robot Systems Equipped with RGB-D Cameras," *Robotics*, vol. 4, no. 1, pp. 1–22, 2014.
[36] P. Besl and N. D. McKay, "A Method for Registration of 3-D Shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 239–256, 1992.
[37] W. Lui, T. Tang, T. Drummond, and W. H. Li, "Robust Egomotion Estimation Using ICP in Inverse Depth Coordinates," in *2012 IEEE International Conference on Robotics and Automation (ICRA)*, May 2012, pp. 1671–1678.
[38] Y. Mori, N. Fukushima, T. Fujii, and M. Tanimoto, "View Generation with 3D Warping Using Depth Information for FTV," in *3DTV Conference: The True Vision - Capture, Transmission and Display of 3D Video, 2008*, May 2008, pp. 229–232.
[39] G. Klein and D. Murray, "Improving the Agility of Keyframe-Based SLAM," in *European Conference on Computer Vision*, 2008, vol. 5303, pp. 802–815.
[40] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox, "RGB-D Mapping: Using Kinect-Style Depth Cameras for Dense 3D Modeling of Indoor Environments," *International Journal of Robotics Research*, vol. 31, no. 5, pp. 647–663, April 2012.
[41] D. Huffman, "A method for the construction of minimum-redundancy codes," *Proceedings of the IRE*, vol. 40, no. 9, pp. 1098–1101, Sept 1952.
[42] L. Zhang and W. J. Tam, "Stereoscopic image generation based on depth images for 3D TV," *IEEE Transactions on Broadcasting*, vol. 51, no. 2, pp. 191–199, June 2005.
[43] K.-J. Oh, S. Yea, and Y.-S. Ho, "Hole filling method using depth based in-painting for view synthesis in free viewpoint television and 3-D video," in *Picture Coding Symposium*, May 2009, pp. 1–4.
[44] C. Zhu and S. Li, "A New Perspective on Hole Generation and Filling in DIBR Based View Synthesis," in *2013 Ninth International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, Oct 2013, pp. 607–610.
[45] "libCVD - Computer Vision Library," http://www.edwardrosten.com/cvd/.
[46] "OpenCV: Open Source Computer Vision Library," http://opencv.org.
[47] "OpenKinect Library," http://openkinect.org.
[48] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A Benchmark for the Evaluation of RGB-D SLAM Systems," in *Proceedings of the International Conference on Intelligent Robot Systems (IROS 2012)*, Vilamoura, Portugal, Oct. 2012, pp. 573 – 580.
[49] D. Marpe, H. Schwarz, and T. Wiegand, "Context-Based Adaptive Binary Arithmetic Coding in the H.264/AVC Video Compression Standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 620–636, July 2003.
[50] A. S. Umar, R. M. Swash, and A. Sadka, "Subjective quality assessment of 3d videos," in *AFRICON, 2011*, Sept 2011, pp. 1–6.
[51] E. Bosc, P. Hanhart, P. Le Callet, and T. Ebrahimi, "A Quality Assessment Protocol for Free-Viewpoint Video Sequences Synthesized from Decompressed Depth Data," in *2013 Fifth International Workshop on Quality of Multimedia Experience*, July 2013, pp. 100–105.

Xiaoqin Wang graduated with a B.Eng. (Hons.) First Class in Electronic and Communication Engineering from the University of Birmingham, UK. He then obtained M.Sc. in Communication and Signal Processing from Imperial College London, UK, in 2011. He is currently pursuing the Ph.D. degree from Monash University, Melbourne, Australia. His research interests include the fields of computer vision, robotics, and 3D video processing.

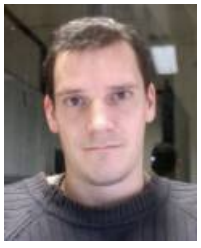
Y. Ahmet Şekercioğlu is a member of the academic staff at the Department of Electrical and Computer Systems Engineering of Monash University, Melbourne, Australia. He established the Monash Wireless Sensor and Robot Networks Laboratory, and currently serves as its director. He completed his Ph.D. degree at Swinburne University of Technology, Melbourne, Australia, and M.Sc. and B.Sc. degrees at Middle East Technical University, Ankara, Turkey (all in Electrical and Electronics Engineering). He leads a number of research projects on distributed algorithms for self-organization in mobile visual sensor, ad hoc, and robot networks.

Tom Drummond studied mathematics at Cambridge University for his first degree before immigrating to Australia in 1990. He worked for CSIRO in Melbourne until 1994 when he went to Perth to do his Ph.D. in Computer Science at Curtin University, Perth, Australia. He then returned to UK to undertake post-doctoral research in Computer Vision at Cambridge University and was appointed a permanent lectureship in 2002. In September 2010, he moved back to Melbourne and took up a professorship at Monash University, Melbourne, Australia. His research interests include real-time computer vision, visually guided robotics, augmented reality, robust methods and SLAM.



Enrico Natalizio is an Assistant Professor at the Universite de Technologie de Compiègne (UTC), France, in the Network & Optimization Group within the Heudiasyc Lab. He obtained his Ph.D. from the Universita della Calabria, Italy and he was a visiting researcher at the Broadband Wireless Networking Laboratory at Georgia Tech in Atlanta, USA. Between 2005 and 2010, he worked as a research fellow and a contract professor at the Universita della Calabria, Italy. From 2010 to 2012 he worked at INRIA Lille as a postdoctoral researcher. His current research interests include group communication in wireless robot and sensor networks and coordination and cooperation among swarm networked devices.

Isabelle Fantoni graduated from the Universite de Technologie de Compiègne (UTC), France, and received the Master of Sciences in Electronic System Design at the University of Cranfield in England (double degree), both in 1996. She received the Master in Control Systems, in 1997 and the Ph.D. degree, in 2000 from the Universite de Technologie de Compiègne (UTC). Since October 2001, she has been a permanent Researcher at Heudiasyc Laboratory, UTC, employed by the French National Foundation for Scientific Research (CNRS).



Vincent Frémont received the M.S. degree in automatic control and computer science from the Ecole Centrale de Nantes, France, in 2000 and the Ph.D. degree in automatic control and computer science from the Ecole Centrale de Nantes, France, in 2003. He joined the Institut de Recherche en Communications et Cybernetique de Nantes in 2000, and the Laboratoire de Vision et Robotique de Bourges, in 2003. Since 2005, he is an Associate Professor at the UTC. His research interests are computer vision for robotic perception with applications to intelligent vehicles.