Disocclusion-Reducing Geometry for Multiple RGB-D Video Streams

Jaesuk Lee* Youngwook Kim* Jehyeong Yun* Joungil Yun† Won-Sik Cheong† Insung Ihm*
*Dept. of Computer Sci. and Eng., Sogang University, Korea †Electronics and Telecommunications Research Institute (ETRI), Korea

Abstract

Depth-image-based rendering (DIBR) is a key method for synthesizing virtual views from multiple RGB-D video streams. A challenging issue inherent in this approach is the disocclusion that occurs as the virtual viewpoint moves away from a reference view. In this work, we present a technique for extracting 3D geometric data, called the *disocclusion-reducing geometry*, from the input video streams. This auxiliary information, represented as a 3D point cloud, can then be combined easily with a conventional DIBR pipeline to reduce the disoccluded region as much as possible during the view warping process, eventually decreasing the visual artifacts by the subsequent hole-filling process.

Index Terms: Computing methodologies—Computer graphics—Image manipulation—Image-based-rendering

1 Introduction

Depth-image-based rendering paradigm [1] is used to build a real-time virtual-reality (VR) system that enables free navigation in a 3D world created from multiple RGB-D video streams. One of the critical issues is the effective handling of the disocclusions that occur when the virtual viewpoint moves away from a reference view. Several effective hole-filling algorithms have been proposed that aim to reconstruct such missing regions in a visually plausible way. However, particularly when applied in real-time VR environments that require a sufficiently high rendering speed to satisfy the display refresh rate of current VR headsets (80 Hz or above), they often exhibit annoying visual artifacts during reconstruction over increasing viewing baselines between the virtual and reference cameras. In our approach, we first preprocess the input video streams to generate a supplementary dataset, called the *disocclusion-reducing geometry (DRG)* through visibility analysis. This 3D geometric information, represented as a point cloud, is then fed into our real-time virtual-view synthesis pipeline to mitigate the resulting holes as much as possible at little extra computation cost. By reducing the size of disocclusions before a hole-filling algorithm is applied, the visual artifacts can be reduced substantially in the final rendering.

2 System Overview and Problem Description

Fig. 1 shows our real-time view-synthesis system, which produces a stream of stereo images in response to the 6-degrees-of-freedom poses of a user wearing a VR head-mounted display (HMD). Our system assumes that the size of the entire input RGB-D video streams is too large for the GPU memory to hold at one time, forcing it to be kept in the CPU memory. For example, the Technicolor-Museum (TM) dataset was created using 24 reference cameras to generate 300 time frames [2]. Each RGB-D image was captured at 2048×2048 pixels and stored using a 10-bit YUV color format (at 12 MB per image) and a 16-bit grayscale depth (at 8 MB per image), requiring 140.63 GB of memory space for the entire RGB-D stream. In our implementation, the dataset was compressed to 56.25 GB using quality-preserving encoding methods (DXT1 for color and BC4 for depth), but the resulting dataset remained too large for the GPU memory system.

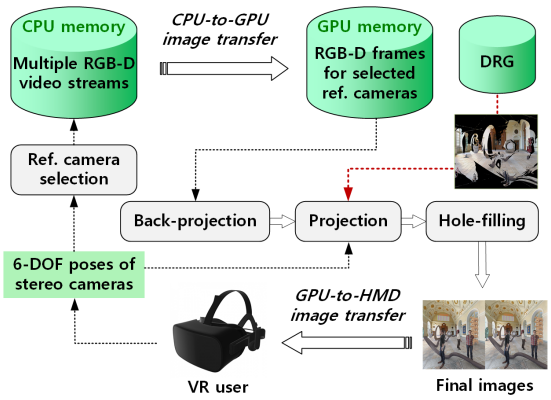

Figure 1: The real-time virtual-view synthesis pipeline for multiple RGB-D video streams. Using the view warping (Back-projection followed by Projection), the points from selected reference views are projected onto virtual view planes. In addition, those 3D points in the DRG that could possibly be occluded with respect to some reference camera in some time frame are directly projected onto the virtual views. As a result, the sizes of any disocclusions occurring in the warping process are reduced significantly at little extra computational cost, thereby substantially enhancing the quality of the final synthesis.

An important aim in visualizing such large RGB-D video streams in real time is to minimize the (on-the-fly) data transfer between the CPU and the GPU. Although using more reference views for each time frame would increase the rendering quality, it also increases the size of the RGB-D images to be transferred, possibly preventing the achievement of a desired frame rate. Our system therefore uses a fixed number of reference cameras, selecting those whose field of view overlaps the most with views from the user's current stereo viewpoints. For the TM dataset, we found that using six to eight out of the 24 reference cameras was optimal when balancing rendering quality against speed. However, such a restriction on the use of 3D information will necessarily exacerbate the disocclusion problem, demanding extra efforts to prevent the occurrence of holes as much as possible during the view warping process.

3 Creation of Disocclusion Reducing Geometry

Consider the hole, marked as Region (A) in Fig. 2, that occurs when the two reference cameras, Camera 5 and Camera 7, are selected for the current time frame. Although this region will be occluded using standard view synthesis, some part of it, marked as Region (B), might be visible from either an unselected camera (e.g., Camera 3) or a selected camera (e.g., Camera 5) in a different time frame (because foreground objects move). The fundamental idea of DRG is to collect 3D points in the world space that might be occluded with respect to some reference camera in some time frame, and use them to reduce the hole occurrence as much as possible.

Fig. 3 illustrates the step-by-step process used to generate the DRG from the input RGB-D video streams. In the first step, we

*{luka756, kimyu7, dudrms5975, ihm}@sogang.ac.kr
†{sigipus, wscheong}@etri.re.kr

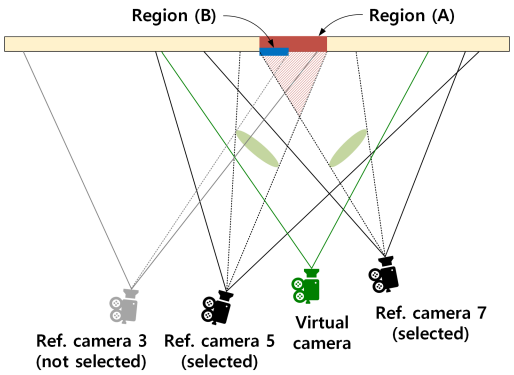

Figure 2: Occurrence of holes in virtual view synthesis.

generate an RGB-D image, called the *maximum-depth image (MDI)*, for every reference camera, each of whose pixels stores the color and depth of the pixel that has the largest depth value among the corresponding pixels in the video stream from the same camera. The pixels of each MDI are then back-projected into the 3D space and accumulated into a point cloud, called the *maximum-depth points (MDPs)*. Intuitively, the MDPs represent the background regions in the 3D space where most disocclusions occur, thereby becoming candidate regions for the DRG. After obtaining the MDPs, from the input video streams, we extract the *possibly missing points (PMPs)* from them. Note that if a 3D point is occluded by a foreground object in at least one time frame of a reference camera, it is considered as *occluded by the camera*.

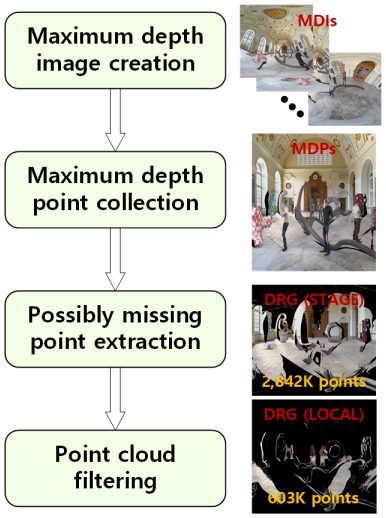

Figure 3: Generation of DRG.

In constructing the PMPs, our system supports two different modes, namely the STAGE mode and the LOCAL mode. The STAGE mode targets those VR applications that need to render standing-scale content, in which any combination of reference cameras may be selected for a given virtual camera. In this case, a 3D point in the MDPs passes the *occlusion test* if it is occluded by any reference camera. In the LOCAL mode, which needs to render seated-scale content, only a predetermined set of combinations of reference cameras are used for view synthesis, enabling the PMPs to be created more compactly for each combination. In the occlusion test, a 3D point passes if it is either occluded by all cameras or outside the combined view volume of all reference cameras. The points that pass both the occlusion test and an extra *background test* now form the set of PMPs. Finally, the points in the PMPs are filtered by averaging the sets of similar points to build a compact DRG dataset.

4 Results

Fig. 4 shows the view synthesis results, where each pair of frame rates, measured on a PC with an Intel Core i7-9800X CPU and dual Nvidia GeForce RTX 2080 Ti GPUs, indicates the total time taken until the created stereo images of 2560×1440 pixels were displayed on the PC and an Oculus HMD with a refresh rate of 80 Hz. The TechnicolorHijack (TH) dataset also used in the test was produced by 10 reference cameras looking in the same direction for a period of 300 time frames, with each RGB-D frame captured at 4096×4096 pixels [2], but downscaled to 2048×2048 for our experimentation. When all reference views (24 for TM and 10 for TH) were considered in the view synthesis, the frame rates on average were 31.3 fps/16.4 fps and 57.6 fps/37.9 fps, respectively. However, using fewer reference views (6 for TM and 5 for TH) together with their DRGs, comprising 2,842,292 and 3,032,664 3D points, respectively, we achieved both the desired frame rates and images that were hard to differentiate from (or often better than) those obtained using all reference views.

(a) 146.3 fps/78.4 fps (b) 138.7 fps/78.7 fps (c) 91.6 fps/62.8 fps

(d) 137.9 fps/78.7 fps (e) 128.7 fps/77.1 fps (f) 89.8 fps/61.7 fps

(g) 194.9fps/ 75.0 fps (h) 183.2 fps/75.3 fps (i) 116.1 fps/78.2 fps

Figure 4: View synthesis results for TM (rows 1 and 2) and TH (row 3). (a), (d), and (g) show the results of the original view-warping process applied to the respective virtual views. (b), (e), and (h) demonstrate how effectively the sizes of holes (in light green) decreased through the help of the DRGs. The remaining holes were then filled more easily by a real-time hole-filling algorithm, as shown in (c), (f), and (i). Please refer to the attached video.

Acknowledgments

This work was supported by the National Research Foundation of Korea (NRF) (No. 2020R1A2C2011709) and Institute for Information & Communications Technology Planning & Evaluation (IITP) (No. 2017-0-00072) grants funded by the Korea government (MSIT).

References

[1] C. Fehn. Depth-image-based rendering (DIBR), compression, and transmission for a new approach on 3D-TV. In *Proc. Stereoscopic Displays and Virtual Reality Systems XI*, vol. 5291, pp. 93 – 104. SPIE, 2004.

[2] ISO/IEC JTC1/SC29/WG11. Common test conditions for immersive video. MPEG/N18563, May 2019.