Robust Multiview Synthesis For Wide-Baseline Camera Arrays

Beerend Ceulemans, *Student Member, IEEE,* Shao-Ping Lu, *Member, IEEE,* Gauthier Lafruit, *Member, IEEE,* and Adrian Munteanu, *Member, IEEE*

***Abstract*—In many advanced multimedia systems, multiview content can offer more immersion compared to classical stereoscopy. The feeling of immersiveness is increased substantially by offering motion-parallax, as well as stereopsis. This drives both the so-called free-navigation and super-multiview technologies. However, it is currently still challenging to acquire, store, process and transmit this type of content. This paper presents a novel multiview-interpolation framework for wide-baseline camera arrays. The proposed method comprises several novel components, including point cloud-based filtering, improved deghosting, multi-reference color blending, and depth-aware MRF-based disocclusion inpainting. The method offers robustness against depth errors caused by quantization and smoothing across object boundaries. Furthermore, the available input color and depth are maximally exploited while preventing propagation of unreliable information to virtual viewpoints. The experimental results show that the proposed method outperforms the state-of-the-art View Synthesis Reference Software (VSRS 4.1) both in objective terms as well as subjectively, based on a visual assessment on a high-end light-field 3D display.**

***Index Terms*—view synthesis, multiview 3D, disocclusion inpainting, depth filtering**

I. Introduction

MANY modern multimedia systems and applications demand high-quality multiview video content. This type of content is found in numerous domains such as 3D media creation, management and distribution, digital signage, augmented and mixed reality, gaming, medical visualization, to name a few. The multiview video format opens the door for a lot of interesting future applications, but it remains a challenge to acquire, store, transmit and process this type of data in an efficient manner.

Views synthesis methods play a major role in this context as they allow for the generation of virtual viewpoints based on only a limited set of input feeds. View synthesis is applied in the creation of super-multiview content starting from a small number of original cameras needed in order to feed autostereoscopic displays [1]–[3]. View synthesis has also the potential to adjust the baseline of stereoscopic video to serve a diverse set of display devices ranging from mobile devices to large cinema screens [4]. Furthermore, view synthesis has already been successfully used in order to create better prediction signals in 3D video coding systems and thereby improve their compression performance [5]–[7]. Recognizing this potential, the MPEG-i community closely monitors any advances in the field of *immersive video*. The MPEG-FTV ad-hoc group, the predecessor of the MPEG-I working group, recently issued a call for evidence on super-multiview and free-navigation technologies [8]. The call targets the design of (i) better compression methods for super-multiview content in dense but not necessarily linear camera setups, and (ii) new view synthesis techniques that can handle large and non-linear camera arrangements. The group also maintains a Depth Estimation Reference Software (DERS) [9] and a View Synthesis Reference Software (VSRS) [10] representing the state-of-the-art in the field.

B. Ceulemans, S-P. Lu, and A. Munteanu are with the Vrije Universiteit Brussel. B. Ceulemans and G. Lafruit are with l'Université Libre de Bruxelles.

Manuscript submitted February 28, 2017, and revised July 2017, accepted November 27 2017. (Corresponding authors: S-P. Lu, and A. Munteanu)

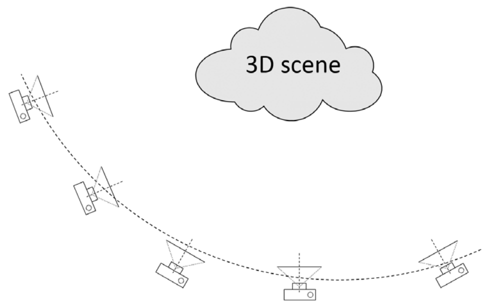


Fig. 1. The considered camera arrays do not have to be linear or ideal arc-shaped. We performed experiments on both ideal and non-ideal arc and linear camera arrangements at various inter-camera distances.

When depth information is available, arbitrary virtual viewpoints can be generated using depth image-based rendering (DIBR) techniques. Using the depth map and the camera calibration matrices, pixels from known reference cameras can be projected onto the imaging plane of a desired virtual camera. However, rendering a virtual viewpoint usually uncovers a part of the scene that was occluded for some or all of the reference cameras. The rendered image will therefore contain holes that need to be concealed in order to provide a pleasant user experience. In image inpainting, many state-of-the-art methods exist that are designed to remove unwanted parts of an image by seamlessly copying existing image structures in their place [11]–[14]. However, directly applying these methods to conceal disoccluded areas usually does not yield acceptable results. Various extensions of these algorithms have been investigated, e.g. the classical PatchMatch algorithm [14] has been applied to view synthesis in [15] and the Markov random field (MRF) -based inpainting method of [13] was adapted for disocclusion filling in [16] and our previous work [17]. Other works focus on graph-based reconstruction techniques [18],

[19] and superpixel segmentation [20].

Many techniques exist in the literature to perform view synthesis. Most methods, however, usually consider either small baseline and/or purely linear camera arrays. In this work, we consider relatively large baselines and arc-shaped camera configurations. These characteristics give rise to some additional challenges. Firstly, the disoccluded areas become larger, which calls for more advanced inpainting methods. Secondly, color differences between corresponding scene points are much more likely to occur and have to be processed properly. In our earlier work [21], [22] we presented methods to correct for significant color differences between cameras in multiview setups. In this work, we assume that the cameras at the input have the desired color characteristics and the aim of the proposed method is to smoothly interpolate viewpoints. It is also important to note that we still consider the camera's to be arranged in a relatively structured way, i.e. all looking towards the scene of interest as shown in Fig. 1.

In this paper, we propose a novel multiview synthesis framework. The proposed method builds further on our earlier work of [17]. We consider the synthesis of virtual viewpoints in a camera array that is not necessarily linear with relatively large baselines between the original cameras. Our method makes use of multiple reference views to reconstruct most of the virtual view's geometry and colors, while occlusions are handled by our MRF-based inpainting algorithm. We also highlight the importance of the depth maps and propose a novel method to detect unreliable depth values. Thorough experimental evaluations demonstrate objective performance gains brought by the proposed method over state-of-the-art. Additionally, we have also performed subjective evaluations on a high-end light-field display (HoloVizio 722RC). The results demonstrate that the proposed method brings statistically-significant subjective quality improvements over state-of-the-art.

In summary, the novel contributions in this paper include:

- Point cloud filtering for multiview sequences
- De-ghosting and depth map filtering
- Depth-aware MRF-based disocclusion inpainting
- Color corrected multi-reference blending
- Subjective assessment and experimental validation on a high-end light-field display

The paper is structured as follows. Section II overviews the different components and the latest advances in multiview video synthesis. The proposed method is presented in detail in section III. The objective and subjective experiments and the comparison against the state of the art are reported in section IV. Finally, section VI draws the conclusions of our work.

II. State-of-the-art

The view synthesis problem originated under the name image-based rendering (IBR). Given a series of 2D projections, the goal is to create a 3D model. This is a very hard problem as there are no assumptions made on the collection of photos that are fed into such a system. Nevertheless, some impressive methods, such as [23], were devised that can solve this problem reasonably well provided that they are given 1) sufficient computation power and time, and more importantly 2) a large amount of photos with a large overlap. The IBR methods then jointly estimate the parameters of the camera associated to each input photo and the 3D coordinates of key-points in the 3D scene that can be tracked in multiple input photos. This results in a sparse mesh that can be densified further by several interpolation methods. View interpolation is usually achieved by morphing some input views to the desired viewpoint. This technique, however, does not work well in large-baseline scenarios.

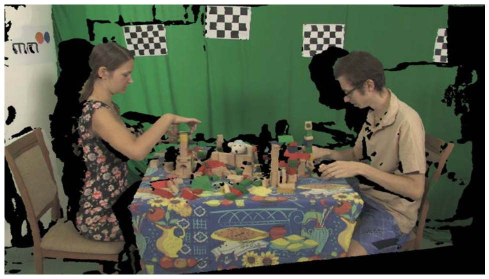


Fig. 2. Pixels from a known camera have been warped on the view of a virtual camera. As the camera movement was to the left, parts of the background that were not visible to the original camera are now disoccluded.

For several applications, the IBR pipeline is often too complex as it requires a lot of processing power and time. In this scenario it is often more desirable to do most of the heavy-lifting off-line. This gave rise to the so-called depth-image-based rendering (DIBR) techniques. Rather than relying solely on 2D color images, DIBR methods also require that each input viewpoint has a depth map associated to it and that the camera parameters are known. This information is then stored and transmitted as a multiview-plus-depth stream. With this kind of format, the view synthesis problem can be addressed in a more realistic manner. However, there still remain difficulties that are not resolved. (I) Due to the materials of objects in the 3D scene and their geometry or due to imperfections in the recording of the multiview data, the colors of different viewpoints can noticeably differ. (II) Scene objects may occlude parts of the scene background that would be visible in a desired virtual perspective. (III) Depth maps are never perfect, so careful processing is required. (IV) Increasing the baseline reduces severely the quality of the synthesized views.

In the following sections, we will discuss existing works in more detail. We categorize them in four categories related to 3D warping, disocclusion inpainting and depth map processing, respectively. Note that this paper does not address the color correction problem. We assume that the input sequences are properly calibrated and aim to provide smooth transitions between the input viewpoints. For more information on this topic, we refer to our recent works in the field [21], [22].

A. 3D-warping

Most DIBR systems follow the same mathematical principle of projective geometry based on the pinhole camera model. They mostly differ in their implementation: some systems perform forward warping with splatting [24], others perform bi-directional warping (also sometimes referred to as inverse mapping) [10], and others employ a mesh-based approach [25]. It is also important to note that when the reference cameras are arranged in a purely linear fashion, the perceived pixel-motion or disparity will also be linear. 3D-warping then simplifies to simple pixel shifts. This is also referred to as disparity-compensation. *Forward warping* refers to the projection from a known camera perspective to the target virtual view. Each pixel of the known camera is projected to the image matrix of the virtual camera. Because of discretization in the image coordinates, the warped pixels do not hit the pixel grid of the virtual camera exactly. The warped values are therefore usually splatted to several nearby virtual pixels. However, even after splatting, the warped pixel grid becomes visible in the missing areas of the virtual view. This can easily be avoided by performing *bi-directional warping*. Most systems first perform forward warping to create the depth map of the virtual view [10], [17]. This depth map is then processed using classical morphological operators and average or median filters are used in order to remove small cracks and outliers. In a backward warp, the cleaned depth map is then used to lookup the correct colors from the known camera using bilinear interpolation.

Other DIBR methods first triangulate the pixel grid of the known camera and warp triangles instead of pixels [25], [26]. In the method of Kopf *et al.* [27], gradients rather than pixels are warped. The actual image is then reconstructed by integrating the Poisson equation [28], regularized with a weak additive bias to compensate for the lack of proper boundary values. Poisson integration also occurs in other works in order to compensate for color differences [29].

Generally, either bi-directional warping or triangle-based warping both yield good objective and subjective results for linear camera arrangements and small baselines between the cameras. The benefit of the bi-directional warping approach is that multiple classical or more advanced image processing algorithms can be tested in order to maximize the quality of the warp. The benefit of a triangle-based approach is that it can rely on well-known methods that are already implemented in graphics hardware and therefore have a lower runtime. It is also possible to obtain computational speedups by performing region-aware 3D-warping that avoids redundant operations [30].

B. Disocclusion handling

Virtual viewpoints generally uncover parts of the scene that were occluded in the reference camera views. This is referred to as disocclusion and results in empty areas surrounding objects in the synthesized image (see Fig. 2). There are generally two ways to fill in disoccluded pixels. Some researchers make sure the virtual depth map contains no holes [31]–[33]. This means that for every virtual pixel, a color can be found in one of the known cameras. Others have developed specialized inpainting algorithms [15], [34], [35], usually inspired by those in single image/video inpainting [36], but taking into account depth information.

1) Avoiding holes: In [31], a regular grid is imposed on the depth map. The warping procedure is then implemented as a deformation of this grid and is constrained in such a way that it does not create any holes. This means that in DIBR systems employing bi-directional warping, a color can be found for every pixel in the virtual image. Alternatively, one can prevent holes in the virtual images by estimating the full depth map of the desired virtual view and then use this depth map as a guide to sample colors from the reference views. In [32] this is implemented by so called plane-sweeping. Every pixel in the virtual view is assigned a tentative depth value. Based on this value, the pixel can be warped on multiple reference images and a cost is computed based on how much the references agree or disagree about the color for a particular pixel. Next, the tentative depth value is adjusted and a new cost is computed. In the end, each pixel in the virtual view is assigned the depth and color on which most of the references agree. This method is inherently massively parallel and therefore very suited for real-time GPU implementation. The plane sweeping method has been shown to deliver good results for free navigation in soccer video [33]. However, when the scene contains complicated textures or the displacement of the virtual camera is either too large or non-linear, methods like [31], [32] are expected to create large blurry areas and ghost edges. In [27], plane-sweeping is employed as a stereo-matcher and an extra global optimization is performed in order to reduce noise and blur artefacts in image regions with non-negligible gradient magnitudes. This is achieved using graph cuts [37] and a gradient-sensitive regularization function [27]. Alternatively, as suggested in [38] one can try to analytically determine optimal camera locations in order to have full visibility of the captured scene.

2) Inpainting: A more common approach for disocclusion handling is to employ inpainting algorithms. Many inpainting methods exist in order to seamlessly remove content in a 2D image or even 2D video. However, directly applying these methods to erase disocclusion areas will not generate satisfactory results given that most patch-based inpainting methods follow a kind of "onion peeling" approach where iteratively, a region of the hole boundary is filled in by a patch that resembles the overlap with the already known area [39]. The filling order is driven by the presence of structures in the image. Disocclusion regions, however, have a clear physical origin and it is known that they should be filled in with only patches that are sampled from the scene's background. Therefore, various view synthesis algorithms in the literature adapt a classical 2D image inpainting algorithm in order to make them depth aware and to avoid the bleeding artifacts that would otherwise occur (see Fig. 4).

A first class of inpainting methods is based on interpolation or diffusion. Smaller holes may be filled in using either Gaussian filtering or median filtering, while for larger holes an iterated diffusion process makes sure that strong contours are extended in the disocclusion hole [40]. These methods are simple and efficient for smaller disocclusion and content with simple textures. For larger disocclusion holes, most researchers tend to opt for a patch-based method, usually based on Criminisi's work [39]. In [39], the border of the hole is referred to as the fill front $\partial\Omega$ (see Fig. 3). In every iteration, the pixel p of the highest priority $P(p)$ is selected and a patch

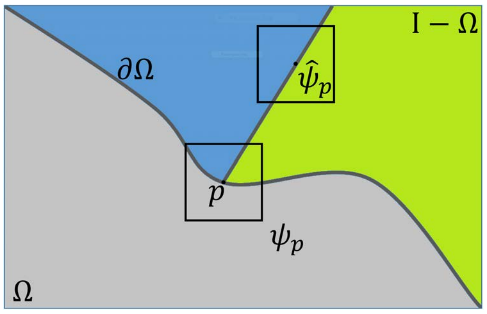


Fig. 3. Inpainting schema: the region Ω needs to be filled by copying and pasting patches from the known region $I - \Omega$.

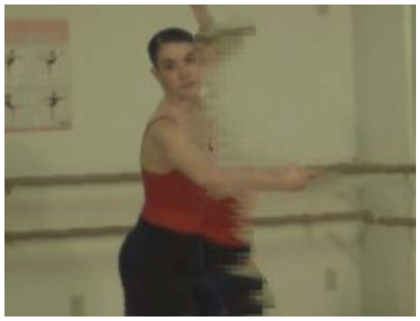


Fig. 4. Bleeding of foreground object when classic 2D image inpainting is used to fill the disoccluded area after 3D warping [16].

$\psi(p)$ around it is examined. Since $p \in \partial\Omega$, $\psi(p)$ overlaps with the known region of the image. Based on this overlap, a patch $\hat{\psi}(p)$ is searched for such that the sum of squared differences $SSD(\psi(p), \hat{\psi}(p))$, computed only on the known pixels of $\psi(p)$, is minimized. The priority function $P(p)$ is constructed in such a way that patches that extend edges are favored over others. Because of their use of patches and the clever choice of $P(p)$, the method of [39] is able to preserve both the texture and structure of the image.

Daribo *et al.* [34] extended Criminisi's work [39] to inpainting of disocclusions, refining the calculation of $P(p)$ and the search for $\hat{\psi}(p)$ by taking the depth information into account. Similarly, Gautier *et al.* [41] used a tensor-based structure propagation approach to compute the priority of structural textures based on their local geometry-based inpainting strategy [42]. Also, [43] employs a tensor-based approach. In these methods, depth-based foreground and background analysis can be used to further guide the inpainting process [35].

The main disadvantage of [39] and its extensions is that they are greedy methods that cannot backtrack if they make a wrong decision at some point. These limitations were tackled by posing the inpainting problem as a global optimization over the entire image. The PatchMatch method [14] operates by estimating a nearest neighbor field (NNF) from Ω to $\mathcal{I} - \Omega$ (following the notations from Fig. 3). The estimation of the NNF and a weighted vote-based image reconstruction step are then alternated in an Expectation Maximization framework that operates on a Gaussian image pyramid. The PatchMatch algorithm has been adapted for disocclusion inpainting by using depth information in our previous works [15], [44].

Komodakis *et al.* [13] pose the inpainting problem as an energy minimization problem on a 2D MRF. [13] proposes a new variant of the classical belief propagation (BP) algorithm and names it priority-BP. The priority function is computed based on the state of the MRF rather than the image structures. However, the original 2D inpainting method of [13] is relatively slow even with clever implementations that use frequency domain computations and multi-scale processing. Moreover, like any regular 2D inpainting method, it generates artifacts by bleeding pixels from foreground objects into the background when applied to fill disocclusions (Fig. 4).

In [16], an extension of Komodakis' method is presented where the computation of unary MRF-potentials at object boundaries is disabled, which prevents color bleeding artefacts. Our MRF-based disocclusion inpainting approach in [17] further improves on this technique by limiting the patch-selection to background regions only and by adopting a new priority function that promotes the information-flow from background to foreground. This results in reduces complexity and improved visual quality. In this work, we further build on our approach [17] by adding the image quilting technique [45] which significantly reduces blocking artefacts. More details are given in section III-D.

C. Depth map processing

Most researchers agree that the quality of the depth maps has an enormous impact on the view synthesis result. In [46], a depth no-synthesis-error model (D-NOSE) is proposed which analyzes the allowable errors that depth maps can have in order to maintain high-quality synthesis. Preprocessing of depth maps is therefore quite common. In [47], a quad-lateral filter is proposed. This filter is an extension of the classical bilateral filter, but it includes some additional information in the form of a spatio-temporal component in the weighted sum. This attempts to suppress common flickering depth pixels while also enhancing the local coherency of surfaces. Emori *et al.* [48] consider mixed-resolution scenarios where depth maps are of lower resolution compared to the corresponding texture image. In [48], various experiments using simple bicubic interpolation or wavelet transforms are presented. Other researchers focus on the actual estimation of depth maps. The MPEG depth estimation reference software (DERS) [9] delivers state-of-the-art depth maps based on image segmentation, epipolar disparity search and graph-cut optimization [37]. However, the software has a high runtime and various parameters to tune. Stankiewicz *et al.* [49] propose a novel method for unsupervised depth estimation based on maximum a posteriori probability (MAP) estimation.

The state-of-the-art in view synthesis is given by the View Synthesis Reference Software (VSRS) [10], maintained by the MPEG-I (previously MPEG-FTV) working group. This method combines bi-directional warping using quarter pixel precision and simple diffusion-based inpainting. The software required exactly two reference views in order to synthesize a virtual viewpoint, i.e. the closest left and right views. In contrast to this approach, the proposed method makes use of

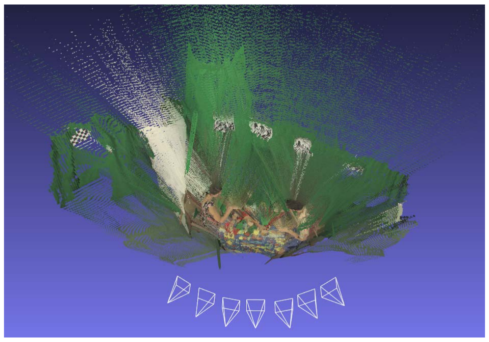


Fig. 5. Top view of Poznan Blocks, showing the camera locations.

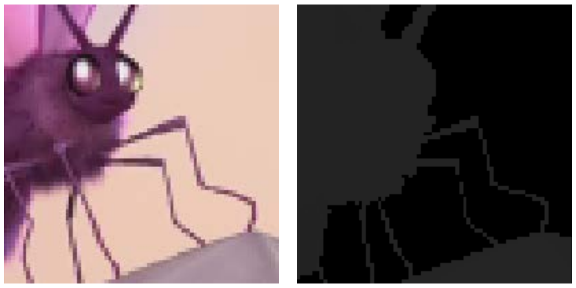


Fig. 6. Both images in this figure are zoomed-in on a critical part of the Butterfly sequence. The zooms are computed using nearest neighbor interpolation. This shows that the legs of the butterfly are only 1 pixel wide in the depth map. In the color image however, the edge is slightly blurred into the background which would result into annoying ghosting artefacts when performing 3D-warping of this view to a virtual viewpoint.

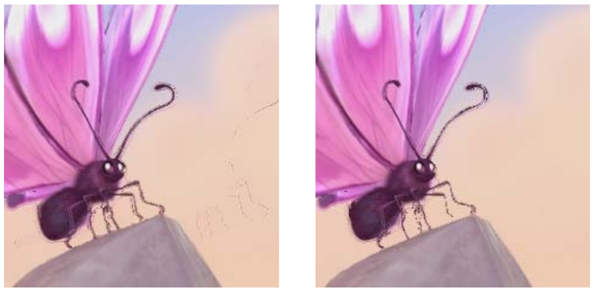
multiple references, performs removal of impossible surfaces and improved depth map filtering, and employs MRF-based disocclusion inpainting.

III. Proposed method

In this section, we will discuss the proposed multiview synthesis framework in more detail. We consider a scenario where multiple reference cameras are available and focus on the case where they are positioned with relatively large inter-camera distances (wide baseline). A top view of a camera array for *Poznan Blocks* is shown in Figure 5. We denote the set of known reference views as a set

$$\mathcal{C}_{ref} = \{(T_1, D_1), (T_2, D_2), \cdots (T_N, D_N)\} \tag{1}$$

This set consists of N cameras denoted as a tuple $R_i = (T_i, D_i)$ of a texture T_i and depth map D_i. Given a target virtual view, we then choose a set of reference cameras $R = R_l \cup R_r$ with $R, R_l, R_r \subseteq \mathcal{C}_{ref}$ where R_l (R_r) is a subset of up to N_l (N_r) reference cameras that are positioned to the directly left (right) of the virtual camera position. We also add the constraint that $abs(|R_l| - |R_r|) \leq 1$, with $|\cdot|$ denoting the cardinality of a set and abs indicating the absolute value. This constraint ensures that virtual camera positions at the edges of the reference camera array are not synthesized from cameras that are too far away and thereby maintain more the characteristics of an interpolation problem rather than an extrapolation. The notions of left and right imply that there needs to be an ordering relation between different cameras.



(a) Visible ghosting artefacts. (b) Proposed warping method.

Fig. 7. Unreliable color pixels were identified and processed more carefully, which greatly reduces ghosting artefacts while preserving fine structures such as the legs of the butterfly.

We allow the camera configuration to be non-linear, but we still assume that it follows a 1-dimensional path, i.e. we do not consider cameras arranged in a 2-D grid.

The high level steps of the proposed multiview synthesis method can be summarized as follows:

- Select the appropriate reference cameras $R \subseteq \mathcal{C}_{ref}$
- Remove unreliable pixels from all $D_i \in R$
- Forward warp and clean all $D_i \in R$ reference depth maps to the desired view
- Backward warp the 2 closest references and perform depth-aware color blending
- Backward warp the $|R| - 2$ remaining references in order to avoid large disocclusion areas
- Perform MRF-based disocclusion inpainting for the remaining hole regions

A. *Warping from known cameras*

In order to warp a known view to the desired virtual view, we employ a bi-directional warping using projective geometry and the pinhole camera model. We experimented with different strategies in terms of the number of references used and the order in which different stages are performed.

1) Number of references: In our experiments, we found that using more cameras greatly helps to avoid large disoccluded areas. However, regions in the scene that are seen by almost all references are very difficult to blend seamlessly as further references are more likely to have subtle color changes and errors in both the estimation and quantization of depth pixels. For this reason we settled for a value of $|R| = 4$ where we use the 2 closest references to reconstruct the majority of the image structure and colors. The remaining cameras are then only used for image regions where they are contributing new information or where the information they are providing is closer to the camera than the already existing values.

2) Multi-source blending: Since our approach is heavily based on bi-directional warping, it is crucial that in the backwards warps, color from different sources are only blended together if they correspond to the same scene-point. On the one hand it seems logical to first create one single depth map for the virtual view - cfr. plane-sweeping [32], [33] - and then process it to be as clean as possible, e.g. by enforcing piece-wise smoothness [37]. However, this depth map then needs to be used to sample the appropriate color information from the reference cameras. If a virtual pixel is reprojected onto the different references, we still need to determine whether all references are seeing the same scene-pixel or if some of them are seeing a background color while the others are seeing foreground. A simple way to do this, is to check whether the depth value of the reprojected virtual pixel is similar to the depth value of the reference pixel of which we are trying to sample the color. If this is not the case, there is an occlusion. Even though the reference depth maps are not fully reliable, this approach works because the forward and backward projection are each others inverse. However, when we generate one single depth map and use this to re-project to multiple references, errors are aggregated and negatively impact the synthesis quality. For this reason, we choose to generate multiple virtual depth maps D_{vi} by separately warping every $D_i \in R$.

Due to spatial quantization in digital images, the color information of pixels near an object edge is usually a mix of the object's color and the background behind it (Fig. 6). However, a depth map usually produces a sharp discontinuity and naively warping all pixels would cause ghosting artefacts (Fig. 7a). These artefacts can be avoided by performing edge detection on the depth maps of the reference cameras and slightly dilating those to produce a reliability mask M_i for every $T_i \in \mathcal{C}_{ref}$. Pixels that fall exactly on the edge are unmasked in order to preserve fine structures such as the legs of the butterfly in the *Butterfly* sequence, and the flower stems and the bunny's whiskers in the *Bunny Flowers* sequence. If in the backwards warping step a masked pixel would be selected, we discard this color contribution if at least one other reference can provide a non-masked color. The result of this, is shown in Fig. 7b.

For every pixel p in the virtual view T_v, the color is computed by considering each warped $D_{vi} \in R$ going from the closest to the furthest reference camera position. Let p_i be the image of p in the i'th reference camera. An initial color is computed based on the two closest references, which are the most reliable as:

$$T_v(p) = \frac{\omega_1 T_1(p_1) + \omega_2 T_2(p_2)}{\omega_1 + \omega_2} \text{ if } M_1(p_1) \wedge M_2(p_2) \quad (2)$$

where M_i denotes the reliability masks and ω_i are weights associated to each reference camera. The weights to perform the color blending are computed as:

$$\omega_i = e^{\frac{-dc_i}{2 \cdot dc_{max}}} + \lambda e^{\frac{-dq_i}{2 \cdot dq_{max}}} \quad (3)$$

In this equation, dc_i is the Euclidean distance between the camera centers of the virtual view and a reference view $(T_i, D_i) \in R$, and $dc_{max} = \max dc_i$. Additionally, dq_i is a distance measure between the camera orientations. Let, $q_{virtual}$ and q_i be the normalized quaternion representation of the orientation of the virtual and i'th reference camera, respectively. We then define the distance between these two orientations based on the inner product $< \cdot, \cdot >$ of these quaternions [50], λ is then a parameter to control the balance between these position and rotation terms.

$$dq_i = 1 - < q_{virtual}, q_i > \quad (4)$$

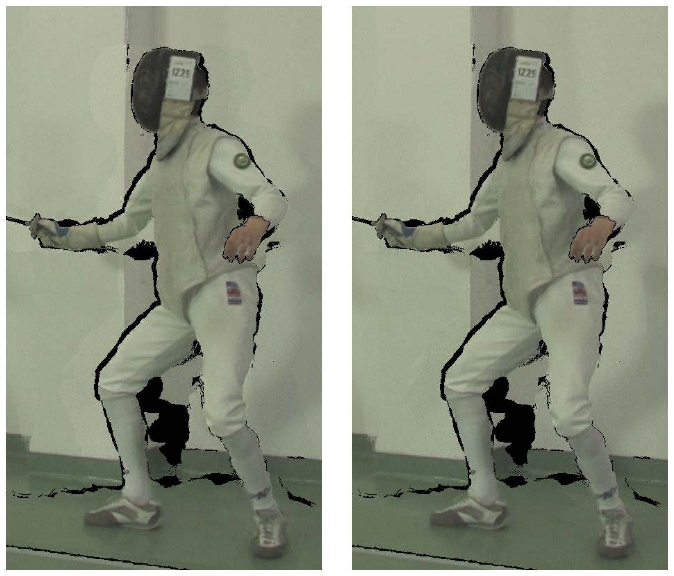

(a) Only weighted blending. (b) Weighted blending and histogram matching.

Fig. 8. The result of the proposed color-blending method. Color correction after establishing the desired color distribution is necessary to avoid ghost-like contours in regions that were only visible for 1 reference camera. This figure is best viewed in color and at high resolution.

The proposed multi-source blending algorithm consists of two passes. In a first pass an initial image is rendered based on pixels that can be reliably sampled from the two closest views. The appropriate color in the virtual view is computed based on the blending formula presented in Eq. 2. After this first pass, the reference images T_i are color-corrected to match the color distribution of the current T_v. In a second pass, the remaining pixels from the closest references (now color-corrected) are warped and remaining occlusion areas are filled using the further references where possible.

The two closest reference views are blended by default. Other views are only used for pixels that are still unknown or pixels that are significantly closer to the virtual camera center, i.e. have a relative z-value decrease of at least 5%. In this work, for efficiency considerations, we apply simple histogram matching [51] to perform color correction. Figure 8 depicts a result of the proposed color blending method.

B. Removal of impossible surfaces

Both in measured and estimated depth maps, some depth pixels contain values that do not correspond to the 3D scene. Sometimes, the values are just a little off due to limited precision, but it can also happen that they are simply plain wrong, i.e. a confusion between foreground and background values. Figure 9a depicts a point cloud rendered by re-projecting all pixels in camera 4 of the Poznan Blocks sequence using the input depth map. Visualizing the data in this manner reveals some artefacts that are less apparent in the 2D projection. For example, the effect of quantization in the coordinates of the vertices becomes visible. However, the most disturbing errors are the *flying pixels* that connect foreground objects to the background. We suspect that these originate from the post-processing of the depth map in order to make it visually pleasant and seemingly coherent. These pixels are very disturbing when performing view synthesis. In this paper, we propose a method to detect these pixels. We then avoid them when performing 3D warping. This is achieved by following a procedure for the denoising of LiDAR-acquired depth maps as proposed in [52].

To detect the so-called *flying pixels*, we analyze the local geometry of the point cloud and identify vertices that are physically not plausible. Since the point cloud is generated from a 3D $\rightarrow$ 2D projective measurement, it is highly unlikely that it could contain surfaces that are aligned with the view frustum of the projective camera. For every vertex $\vec{v_i}$ in the point cloud we estimate its local normal vector in the coordinate frame of the camera using the method of [53]. This normal vector describes the surface around this vertex. We then compute the inner product between this normal vector $\vec{n}_{v_i}$ and the vector $\vec{v_i} - \vec{f}$ which is the vector that points from the camera's optical center to the vertex. When this inner product is lower than a certain threshold τ, the vertex is considered unreliable and discarded from the point cloud. This is implemented in our system by the computation of a depth map mask

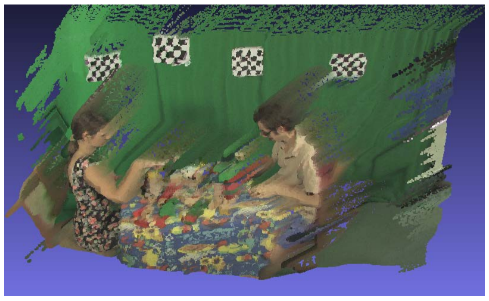
$$D_m = \begin{cases} 0 & \text{if } \langle \vec{n}_{v_i}, \vec{v_i} - \vec{f} \rangle < \tau \\ 1 & \text{otherwise} \end{cases} \tag{5}$$

Figure 9b depicts the same point cloud removing the pixels marked by our filter. Since the rendering of this point cloud is a form of view synthesis, it is easy to see that this pre-processing filter will greatly improve the performance of other view synthesis techniques as well.

C. Filtering after forward warping

In a classical DIBR framework, the depth maps of known camera viewpoints are first projected to the target camera (forward warping). These warped depth maps then undergo some processing and the warped-and-processed depth maps are then used in a backward warp to sample colors from the known cameras and essentially synthesize the desired virtual view. Figure 10a shows a forward warp of a depth map. Red pixel values indicate that the depth value at this location is unknown, either because it was occluded in the view where it was warped from or because we marked it as unreliable in the previous step. In this stage, the effect of quantization is hindering the synthesis result. The procedure can be summarized as follows: for every pixel p in the current warped depth map D, the following steps are performed.

1) Let $W_p = \{p_i | \, ||p - p_i||_1 \leq 1\}$ be the window that contains the pixel p and its 8-connected neighbors.
2) $N_k = \{p_i \in W_p | D(p_i) \text{ is known}\}$ and $N_u = \{p_i \in W_p | D(p_i) \text{ is unknown}\}$
3) If p is known and $|N_u| > 6$, mark p as unknown. Isolated pixels are likely to be wrong.
4) If p is known, but 6 of its neighbors have a value that is closer to the camera, replace p by the median of these neighbors. This pixel is likely to be a background value that is visible through a foreground object.


(a) Raw point cloud with all vertices.

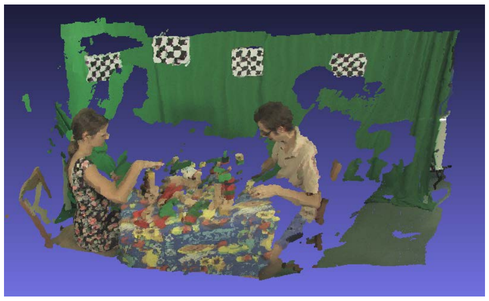

(b) The proposed filter removes unlikely points.

Fig. 9. The proposed filtering approach based on point normals and their orientation with respect to the view vector that connects the camera focal point and the vertex position. The point cloud is rendered in MeshLab [24] using forward warping with splatting.

5) If p is unknown but $|N_k| > 4$, mark p as known and assign it the median value of its known neighbors. This pixel is a thin hole due to quantization.

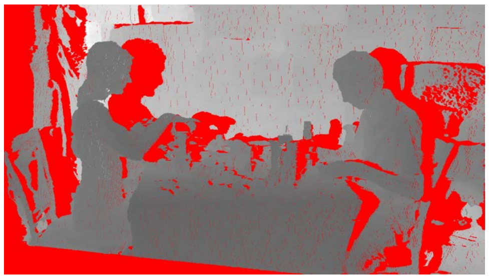
These steps are repeated until the depth map stabilizes. The resulting depth map after processing with the proposed method is depicted in Fig. 10b.

D. MRF-based disocclusion inpainting

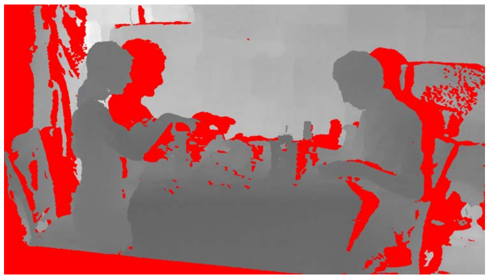
First, a regular grid of overlapping $w \times w$ patches is defined over the entire image. Patches that contain disoccluded pixels are considered to be nodes $p_i \in \mathcal{V}$ in an MRF where $\mathcal{V}$ denotes the set of all nodes. All other patches that contain no missing pixels form the set of possible labels $\mathcal{L}$ that can be used to fill in any missing pixels. MRF nodes are connected in a 4-neighborhood and the set of edges is denoted as $\mathcal{E}$. For every node, we want to find a plausible patch $x_i \in \mathcal{L}$ that contains no missing pixels and can be copied such that all disoccluded pixels are filled in. We call this a labeling for patch p_i and we denote $\vec{x}$ as a labeling for the entire MRF. Finally, we would like the filling of all disoccluded pixels in the image to be globally optimal rather than greedy. We denote the MRF total energy as:

$$E(\vec{x}) = \sum_{p_i \in \mathcal{V}} V_i(x_i) + \sum_{(i,j) \in \mathcal{E}} V_{ij}(x_i, x_j) \tag{6}$$

where $V_i(x_i)$ denotes the label cost for a node p_i and is computed as the sum of squared differences (SSD) between any known pixels in the existing patch at node p_i and the corresponding pixel in the patch x_i that is assigned to fill the missing pixels. Similarly, $V_{ij}(x_i, x_j)$ denotes the SSD within



(a) Forward warping without filtering. Red areas denote unknown pixels. Note that there are thin regions with red values as well as thin regions with background pixels that are visible through what should be a solid foreground object, due to the discretization of the source pixels' coordinates.



(b) The proposed depth map filter successfully fills depth pixels that should be known or corrects depth pixels that are likely to be wrong.

Fig. 10. The effect of our depth map filtering after forward warping.

the overlap when patches x_i and x_j are assigned to nodes p_i and p_j, respectively.

The solution to the inpainting problem, $\vec{x}^*$, is then defined to be the labeling that minimizes this energy:

$$\vec{x}^* = \underset{\vec{x}}{\operatorname{argmin}}\, E(\vec{x}) \tag{7}$$

Patches that consist of only unknown pixels will have a 0 label cost for any label, therefore the label for these patches will be inferred from the information that is propagated in the overlap costs.

1) Energy minimization: Exact minimization of this energy function would require an exhaustive enumeration of all possible combinations and is therefore computationally intractable. For N MRF-nodes and $|\mathcal{L}|$ possible labels, the complexity would be $O(|\mathcal{L}|^N)$. However, a good approximate minimum can be computed in $O(|\mathcal{L}|^2)$ time using a variant of the belief propagation algorithm. We employ the priority-BP (p-BP) algorithm proposed by [13].

In a belief propagation (BP) algorithm, the global energy is minimized by having all MRF-nodes send message vectors to their direct neighbors. A message from node p_i to node p_j is computed as:

$$m_{ij}(x_j) = \min_{x_i \in \mathcal{L}_i} \{V_{ij}(x_i, x_j) + V_i(x_i) \\ \sum_{n \in \mathcal{N}_i \setminus \{j\}} m_{ni}(x_i)\} \tag{8}$$

For every label x_j that MRF-node p_j can take, its neighbor p_i first consults all its own labels $x_i \in \mathcal{L}_i$. Node p_i checks what the cost of choosing this label is and how well it would fit if j chooses x_j. It then also considers the messages that it has received about this label from all its neighbors $n \in \mathcal{N}_i$, except p_j, and computes the minimal message that is achievable for x_j based on the available information.

From these messages, all MRF-nodes can compute a belief value $b_i(x_i)$ for all of their labels:

$$b_i(x_i) = -V_i(x_i) - \sum_{n \in \mathcal{N}_i} m_{ni}(x_i) \tag{9}$$

The order and frequency of message updates depend on the particular variant of the algorithm. In p-BP, nodes are assigned priorities based on the strength of their current belief values. Every iteration, nodes are traversed in order of decreasing priority and messages vectors are sent to nodes that were not yet visited during that iteration. This is referred to as a forward pass. After the forward pass, nodes are visited in reverse order and messages are sent again. A single iteration of the p-BP algorithm consists of a forward pass and a backward pass. After only one iteration, some labels may have become very unlikely for some nodes. This is represented by very low belief values. In a pruning step, these labels are removed from the label-sets of these nodes which significantly speeds up subsequent iterations.

The p-BP algorithm computes patch priorities $P(p_i)$ based on their belief values and additional information from the depth map:

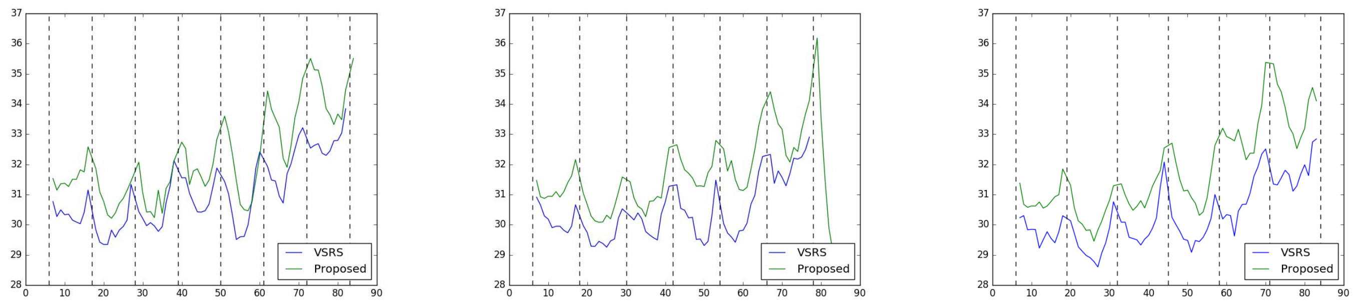
$$P(p_i) = Z\text{-}bonus(p_i) + \frac{1}{|\mathbb{CS}(p_i)|} \tag{10}$$

$$\mathbb{CS}(p_i) = \{x_i \in \mathcal{L}_i | b_i(x_i) - \max_x b_i(x) \geq b_{conf}\} \tag{11}$$

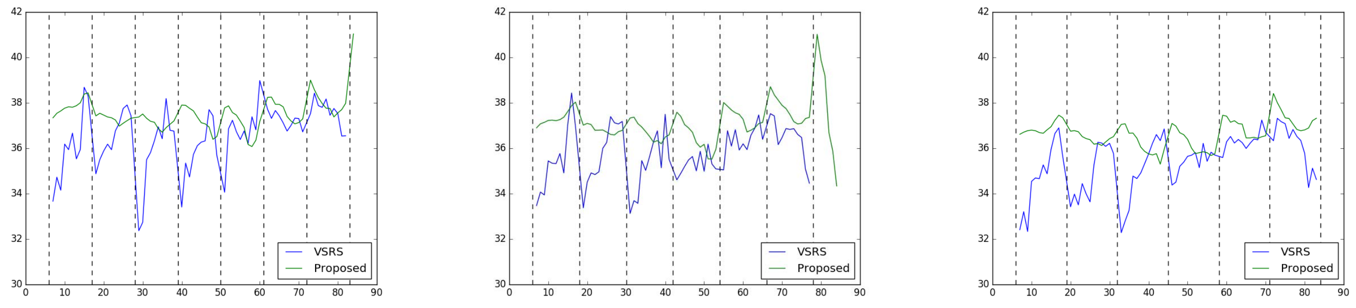
In Eq. 10, $|\mathbb{CS}(p_i)|$ denotes the size of the *confusion set* (Eq. 11) for node p_i. This is the set of labels for which the belief values relative to the maximum value, exceed some predefined threshold b_{conf}. The Z-bonus is a value that is computed based on the depth map, i.e. nodes that intersect the edge of a foreground object are given a 0 value, while nodes that intersect with the background are set to 1. Next, at each message update, the sending node passes 80% of its confidence to its neighbor. This priority function promotes both the flow of important visual structures as well as background-to-foreground propagation. The p-BP algorithm converges after only a few iterations and at this point, every patch will select the label with the highest belief value.

In order to avoid blocking artefacts in the composition of the inpainting result, we employ the image quilting method of [45]. For every pair of overlapping patches, a minimum-energy seam is computed in order to determine which patch contributes which pixel-value. Pixels on and around the seam are slightly feathered in order to hide the seam itself.

2) Candidate patch selection: As mentioned in the previous section, the asymptotic complexity of the p-BP algorithm is $O(|\mathcal{L}|^2)$. We assume here that every MRF-node $p_i \in \mathcal{V}$ has to select a label from the same set of candidates $\mathcal{L}$. This is the case in Komodakis' original work for 2D image inpainting. However, it is possible to further reduce the runtime by having separate candidate label sets $\mathcal{L}_i$. In [54] for example, each MRF-node is assigned a set of candidate labels based on



(a) *Flowers* skipping 10 cameras. (b) *Flowers* skipping 11 cameras. (c) *Flowers* skipping 12 cameras.



(d) *Butterfly* skipping 10 cameras. (e) *Butterfly* skipping 11 cameras. (f) *Butterfly* skipping 12 cameras.

Fig. 11. Objective evaluation of the proposed method on the *Flowers* and *Butterfly* sequences.

texture features. We obtain a similar behavior by taking into account the physical origin of the regions that need inpainting. Since they originated from a disocclusion, it is reasonable to assume that valid candidate patches should contain background information and that the offset to sample these patches from should be mainly horizontal. In our method, this candidate patch selection is implemented by growing a rectangular window around each MRF-node. More precisely, we first walk across the local hole boundary $\partial\Omega$ in order to find the z_{far} value that is the furthest from the virtual camera. We then start with a small window of 150×15 and start selecting fully-known patches for which the average depth is at least a certain percentage of z_{far}. We adjust the size of the window with increments of 10 and 2 pixels in the x and y dimensions, respectively, until each MRF-node has *at least* 50 candidate labels.

3) Implementation considerations: Based on the description in the original paper of [13] the implementation of the p-BP algorithm is relatively straightforward, but there are some points worth mentioning. First of all, disocclusion holes are often disconnected in a full-HD video frame. Therefore it does not make sense to run global optimization on the disconnected MRF as this will consume a lot of memory. We first run a connected components algorithm on the MRF graph and separately optimize these components. Second, if the MRF consists of a large number of nodes, repeatedly finding the one with the highest priority can become a bottleneck so it is worth implementing this part of the algorithm using a max-heap with an update-key method, similar to what one would use for Dijkstra's shortest path algorithm. And finally, when sending messages from a node to its neighbors, one needs to make sure to normalize the message vector by subtracting the minimum element of this vector in order to avoid numerical instability problems and overflows.

Performing frequency domain SSD computations and a multi-scale optimization as suggested in [13] is definitely worthwhile for 2D image inpainting, but not crucial in the context of disocclusion inpainting as the number of labels can be sufficiently constrained with help from the depth map.

E. Post-processing

After disocclusion inpainting, the virtual view is fully computed. However, DIBR-methods tend to render unnaturally sharp edges at the border between foreground objects and the background. As suggested in [25], we track these edges and before writing out the final result, we apply a subtle low-pass filter at these edges in order to avoid objects looking like cardboard cutouts.

IV. Experimental results

We implemented the proposed multiview synthesis system in `C++` and calculations were performed in Lab color space. Various virtual viewpoints were rendered for some of the popular multiview datasets each with their own particular difficulties, including BigBuckBunny Flowers, BigBuckBunny Butterfly, Poznan Blocks, Poznan Fencing, Microsoft Breakdancer. The camera arrangement in *Flowers* and *Butterfly* forms a segment of an Arc. *Poznan Blocks*, *Poznan Fencing*, and *Microsoft Breakdancer* are sequences where the capturing cameras are more or less arranged in an Arc but they are sparse. For these sequences we interpolated a virtual camera path such that the camera center position follows a smooth cubic spline and the virtual camera orientation a spherical linear interpolation. *Poznan Fencing* is actually arranged as 5 stereo pairs. For the interpolation of the virtual camera path we dropped the right camera in each stereo pair, but this was of course still used for the actual synthesis.

To analyze the performance of our synthesis, we compare against the latest version of the MPEG VSRS software (4.1) and conduct both objective and subjective quality analysis. We also considered the view synthesis tools implemented in the 3D-extension of HEVC (HTM-Renderer) [55]. This software has also been demonstrated to yield very good synthesis results. We can however not use it in our use-case as it is not developed to consider *non-linear* camera arrangements.

Our implementation of the proposed method takes roughly 20 seconds to render a frame while VSRS only takes 1 second on the same desktop PC. The most time is spent on the globally optimized disocclusion inpainting. Speed optimizations of the current implementation are expected to reduce the execution time to an acceptable operational range. This can be done by code optimizations and parallelized GPU implementations of belief propagation [56].

A. Objective evaluation

In order to quantitatively assess the performance of the proposed method, we followed the same procedure as [17]. We present results for the *Flowers* and *Butterfly* sequences. In these sequences, 79 camera views are available. We simulate a wide-baseline scenario by sub-sampling this array of known cameras by skipping 10, 11, and 12 views. The skipped views are then interpolated using both the proposed method and VSRS (4.1) [10]. Since we do have the real views that should be at the skipped positions, we can compute objective quality metrics.

Figure 11 depicts the PSNR computed on the camera views that we chose to synthesize. The "skip" number indicates how many camera positions were skipped while selecting the reference cameras. The horizontal axis depicts the camera index in the array ($6 \rightarrow 84$), and the vertical axis depicts the PSNR value in (dB). The reference camera positions are indicated by the dashed vertical lines. It is clear that the proposed method yields a substantial gain in terms of objective quality. The average gain over all views in a sequence ranges from 0.93 dB to 1.47 dB with peak gains up to 5 dB in some views. Table I gives the average and peak PSNR gains per sequence.

We also compared the proposed method on some older well-known datasets On *Microsoft Ballet* and *Microsoft Breakdancers*. We performed the experiment where camera 4 was selected as the desired virtual viewpoint. We then synthesize it from different baselines by only considering two surrounding reference cameras. We also report the scenario where all reference cameras were available. For VSRS this does not change the results, as VSRS does not support more than two references; this experiment shows that the proposed method successfully exploits the additional information. We also show

TABLE I
PSNR GAINS OF THE PROPOSED METHOD OVER VSRS.

Sequence	average PSNR gain (dB)	max gain (dB)
Flowers (skip 10)	1.14	2.97
Flowers (skip 11)	1.22	2.53
Flowers (skip 12)	1.47	3.99
Butterfly (skip 10)	0.93	5.01
Butterfly (skip 11)	1.28	4.20
Butterfly (skip 12)	1.22	4.77

TABLE II
OBJECTIVE EVALUATION ON OLDER DATASETS.

Sequence (references)	PSNR proposed (dB)	PSNR VSRS (dB)
Ballet (all)	33.05	31.55
Ballet (3 and 5)	32.24	31.55
Ballet (2 and 6)	29.73	27.26
Ballet (1 and 7)	28.49	24.39
Breakdancers (all)	32.9	31.43
Breakdancers (3 and 5)	32.24	31.43
Breakdancers (2 and 6)	31.47	28.92
Breakdancers (1 and 7)	29.53	25.69
Newspaper (2 and 6)	29.18	29.66

results on the well-known *Newspaper* sequence. On this particular sequence, VSRS performs slightly better. The reason for this is that the depth map is of too low quality and the depth-aware label selection in the MRF-based dissoclusion inpainting is misguided. Table II reports the PSNR for various baselines in these datasets.

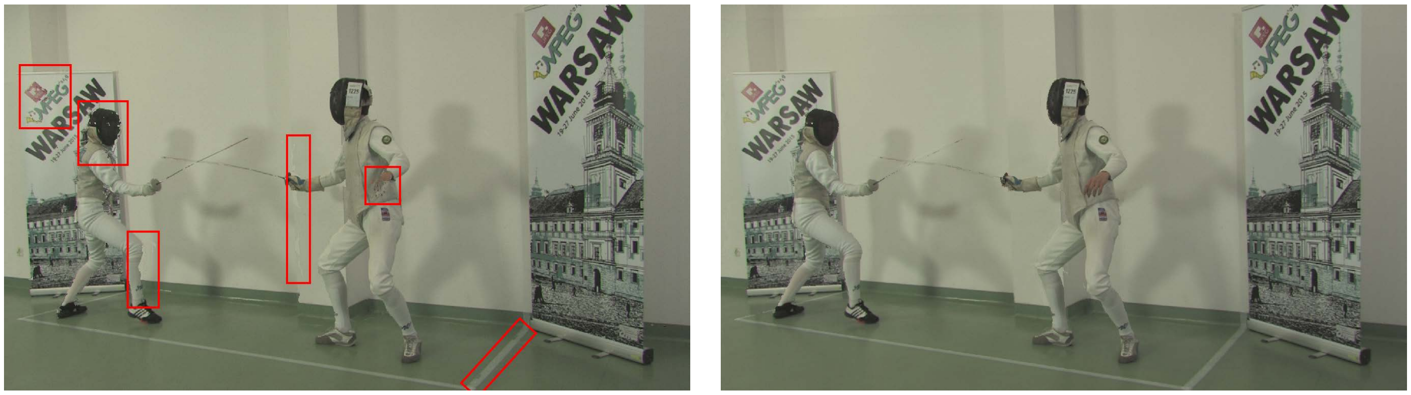
B. Subjective evaluation

In addition to these objective comparisons, we also conducted subjective test sessions. Visual examples of the proposed method compared to VSRS [10] are shown in Figures 12 and 13.

Two of the authors were involved in the subjective evaluation of interpolated viewpoints in the context of a call for evidence of the former MPEG-FTV group [8]. From this experience, we learned that it is not recommended to conduct these experiments by presenting the synthesized views on a regular 2D screen and then sweeping between views. Small rendering artefacts are heavily emphasized when viewed as a temporal change while they are sometimes imperceptible on an autostereoscopic display.

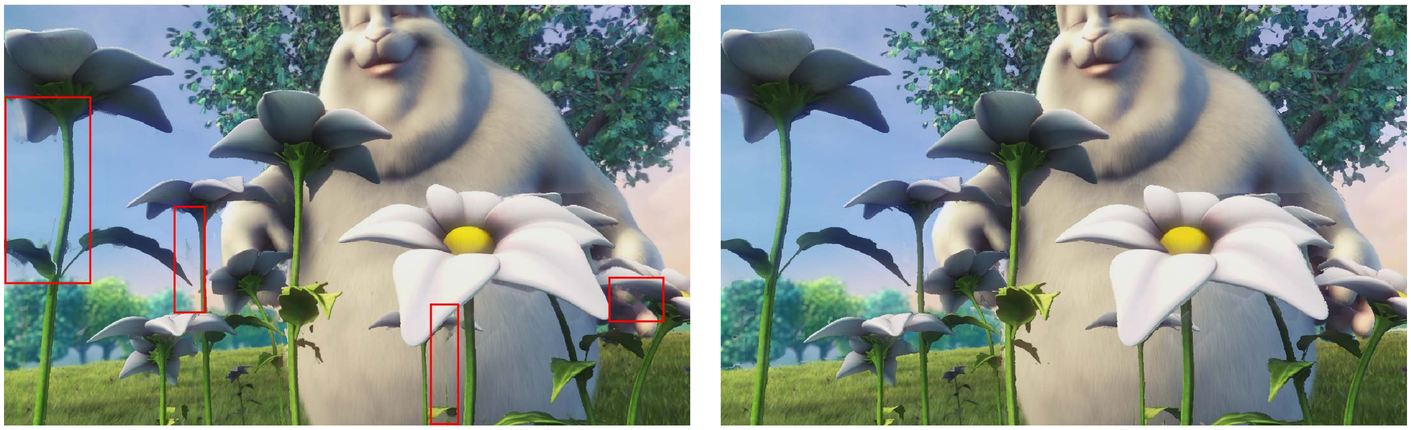
1) Test set-up: A high-end HoloVizio 722RC light-field display from Holografika available at our facilities was used for subjective experiments. The light-field conversion software provided by Holografika is able to process *uncompressed* YUV data, having no impact on the rendered data. The viewing area is equipped with light-blocking curtains and dimmable non-flickering lighting. The lighting conditions and test environment strictly follow the BT.500-13 recommendations [57] for subjective visual testing.

2) Methodology: Currently, there is not yet a standardized method to subjectively assess this type of content. We follow as much as possible the BT.500-13 recommendations [57] and the procedures followed by Dricot *et al.* in their work on SMV compression [58]. We adopt the single stimulus method



(a) VSRS [10]. (b) Proposed method.

Fig. 12. This figure depicts a synthetic view in the *Poznan Fencing2* sequence. As indicated by the overlayed rectangles, the proposed method is able to avoid some artefacts that are common in VSRS renderings.



(a) VSRS [10]. (b) Proposed method.

Fig. 13. This figure depicts a synthetic view in the *BigBuckBunny Flowers* sequence. In this type of sequences, most artefacts are caused by blurry object edges.

TABLE III
ANALYSIS OF SUBJECTIVE TEST RESULTS

Sequence	DMOS	p-value
Breakdancer	0.56	0.0091
Flowers (skip 10)	0.22	0.2574
Flowers (skip 11)	0.69	0.0010
Flowers (skip 12)	0.75	0.00028
Butterfly (skip 10)	1.58	$2 \cdot 10^{-13}$
Butterfly (skip 11)	2.25	$6 \cdot 10^{-21}$
Butterfly (skip 12)	2.02	$5 \cdot 10^{-16}$
Poznan Blocks	-0.28	0.1054
Poznan Fencing	0.58	0.0049

The DMOS value in this table denotes the difference between the Mean Opinion Score of the proposed method and the reference technique. A positive value indicates a performance gain. The p-value is computed on the obtained scores using a two-tailed t-test. A p-value less than 0.05 indicates a statistically significant difference.

instead of the double-stimulus method of [58] which could not be applied to sequences that were not originally captured as dense. Following the recommendation [57], we show only one stimulus and ask the assessor to score it on the classical five-point scale. For the *Flowers* and *Butterfly* sequences, we include the unimpaired reference in the test for which we have ground truth as a computer graphics sequence. We started the experiment with 19 participants. All participants consistently gave these unimpaired references a maximum score with some more modest scores of 4 for the Flowers sequence. One participant scores both of the unpaired references with a 3, but gave a 4 to another impaired sequence so we excluded this participant from the experiment considered as an outlier. The test was organized in two sessions: both sessions contain the same 20 sequences but in a different order. No two sequences of the same content were shown in succession.

3) Analysis: The 18 remaining participants rated each of the 20 sequences twice. We therefore have an array of 720 votes. For each sequence, we compute the average score and the confidence interval (Fig. 14). We conclude that for most sequences, the proposed method is preferred with statistical significance. For the *Flowers* sequence with a skip of 10, the results are comparable and for the *Poznan Blocks* sequence there appears to be a slight preference for the VSRS method.

We screened the observers in order to identify potential outliers using the procedure described in the recommendation [57]

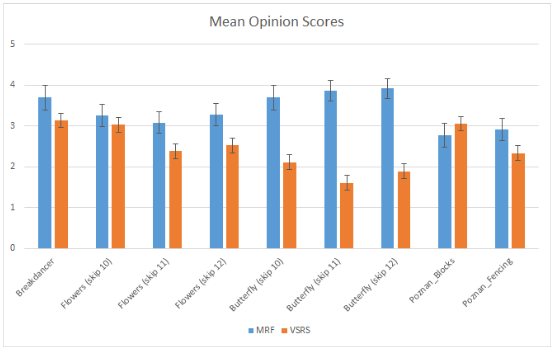


Fig. 14. Subjective evaluation: average of the observer scores on a 5-point scale with error bars.

and an additional check based on the first and third quantiles as described in [58]. Except for the one observer which we had already removed for failing to correctly score the reference sequences, no other observers were identified as outliers. We then performed two-tailed t-tests to check whether the scores for corresponding sequences of the proposed method and the method of [10] are different with statistical significance. The result of the t-tests confirms what is also visible in the bar chart in Figure 14: except for *Flowers (skip 10)* and *Poznan Blocks*, we have to reject the null-hypothesis that the obtained scores have the same mean at a significance level of $p = 0.05$ (see Table III), indicating that for these sequences the measured gains in subjective quality are significant.

C. Parameter configuration

The threshold τ in Eq. 5 is determined empirically by visually assessing the filtered point cloud. A patch size of 15×15 was used for disocclusion inpainting, and the parameters of the priority belief propagation were configured according to Komodakis' original work [13], i.e. $b_{prune} = 2 \cdot b_{conf} = -SSD_0$, where SSD_0 represents a mediocre SSD between two patches $(0.15^2 \cdot patch\_area \cdot 3)$. The number of labels per node in the dynamic label pruning step is set to remain between $L_{min} = 5$ and $L_{max} = 30$. With respect to the λ parameter in Eq. 3, we have analyzed the influence of the rotational term by synthesizing viewpoints for which the ground truth texture is known. In this sense, we have computed the differences in PSNR for various values of λ relative to the case when the rotational term is not accounted for (i.e. $\lambda = 0$). The experiments on both computer graphics material (Butterfly, Flowers) as well as when synthesizing known viewpoints of the *Poznan Blocks*, *Poznan Fencing*, Microsoft Ballet, and Microsoft Breakdancers sequences reveal limited PSNR differences, of no more than 0.1 dB. Hence, we considered $\lambda = 0$ throughout the experiments. However, the rotational term may become important for applications where cameras are arranged in a more arbitrary manner than considered in this work.

V. Limitations and future work

As illustrated in Fig. 1, this paper addresses view synthesis in wide-baseline camera arrays. The proposed approach assumes that there is some structure in the camera setup. That is, when selecting the two closest cameras to synthesize the desired virtual position, the orientation of these cameras should not be completely different from the orientation of the virtual camera. It is also important that the field-of-view of the references used for interpolation have an overlap with the virtual field-of-view; the presented method is not evaluated on datasets where this assumption is violated. The selection of good reference cameras becomes more challenging when the virtual camera path is significantly deviating from a smooth interpolating spline between the input cameras. This is an interesting topic for further research.

VI. Conclusions

This paper presents a novel multiview synthesis framework targeting view interpolation for wide-baseline camera arrays. The framework uses as much as possible of the input color and depth information by relying on more than only the two closest references. Depth maps are carefully preprocessed in order to avoid the propagation of unreliable information throughout synthesized frames, and remaining disocclusions are inpainted using an efficient depth-aware MRF-based inpainting method. The proposed method offers robustness against various types of errors that can be present in the considered camera arrangements. The point cloud-based filter to identify unlikely pixels at object boundaries successfully discards pixels that would have caused ghosting artefacts. Weighted color blending in combination with histogram matching ensures smooth transitions between the color histograms of the reference cameras. Both objective and subjective evaluations on a high-end light-field display demonstrate a significant improvement over the reference method in the literature. Objective performance gains ranging from 0.93 dB to 1.47 dB were observed and statistically significant gains in mean opinion score of 0.56 to 2.25 on a five-point scale.

Acknowledgments

Parts of this research were performed in the scope of the Innoviris 3DLicorneA *2015-R-39c, 2015-R-39d* project. We would also like to thank the reviewers for their constructive feedback and suggestions.

References

[1] B. Lee, "Three-dimensional displays, past and present," *Physics today*, vol. 66, no. 4, pp. 36–41, 2013.
[2] H. Urey, K. V. Chellappan, E. Erden, and P. Surman, "State of the art in stereoscopic and autostereoscopic displays," *Proc. IEEE*, vol. 99, no. 4, pp. 540–555, 2011.
[3] T. Balogh, "The Holovizio system," in *Electronic Imaging*. International Society for Optics and Photonics, 2006, pp. 60 550U–60 550U.
[4] R. T. Held and M. S. Banks, "Misperceptions in stereoscopic displays: a vision science perspective," in *Proc. APGV*. ACM, 2008, pp. 23–32.
[5] M. Domanski, O. Stankiewicz, K. Wegner, M. Kurc, J. Konieczny, J. Siast, J. Stankowski, R. Ratajczak, and T. Grajek, "High efficiency 3D video coding using new tools based on view synthesis," *IEEE Trans. Image Process.*, vol. 22, no. 9, pp. 3517–3527, 2013.
[6] A. Vetro and D. Tian, "Analysis of 3D and multiview extensions of the emerging hevc standard," in *SPIE Optical Engineering + Applications*. International Society for Optics and Photonics, 2012, pp. 84 990Y–84 990Y.

[7] F. Zou, D. Tian, A. Vetro, H. Sun, O. C. Au, and S. Shimizu, "View synthesis prediction in the 3-d video coding extensions of avc and hevc," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 24, no. 10, pp. 1696–1708, 2014.

[8] N15733, "Call for evidence on free-viewpoint television: Super-multiview and free navigation," October 2015.

[9] K. Wegner, O. Stankiewicz, M. Tanimoto, and M. Domanski, "Enhanced depth estimation reference software (DERS) for free-viewpoint television," October 2013, m31518.

[10] ——, "Enhanced view synthesis reference software (VSRS) for free-viewpoint television," October 2013, m31520.

[11] K. He and J. Sun, "Statistics of patch offsets for image completion," in *Proc. ECCV*. Springer, 2012, pp. 16–29.

[12] J.-B. Huang, S. B. Kang, N. Ahuja, and J. Kopf, "Image completion using planar structure guidance," *ACM Trans. Graph.*, vol. 33, no. 4, p. 129, 2014.

[13] N. Komodakis and G. Tziritas, "Image completion using efficient belief propagation via priority scheduling and dynamic pruning," *IEEE Trans. Image Process.*, vol. 16, no. 11, pp. 2649–2661, 2007.

[14] C. Barnes, E. Shechtman, A. Finkelstein, and D. Goldman, "Patchmatch: A randomized correspondence algorithm for structural image editing," *ACM Trans. Graph.*, vol. 28, no. 3, p. 24, 2009.

[15] S. Lu, J. Hanca, A. Munteanu, and P. Schelkens, "Depth-based view synthesis using pixel-level image inpainting," in *Proc. DSP*. IEEE, 2013, pp. 1–6.

[16] J. Habigt and K. Diepold, "Image completion for view synthesis using markov random fields and efficient belief propagation," in *Proc. ICIP*. IEEE, 2013, pp. 2131–2134.

[17] B. Ceulemans, S.-P. Lu, G. Lafruit, and A. Munteanu, "Efficient MRF-based disocclusion inpainting in multiview video," in *Proc. ICME (Oral paper)*, 2016, pp. 1–6.

[18] Y. Mao, G. Cheung, A. Ortega, and Y. Ji, "Expansion hole filling in depth-image-based rendering using graph-based interpolation," in *Proc. ICASSP*, May 2013, pp. 1859–1863.

[19] Y. Mao, G. Cheung, and Y. Ji, "Image interpolation for DIBR view synthesis using graph Fourier transform," in *Proc. 3DTV-CON*, July 2014, pp. 1–4.

[20] T. Tezuka, M. P. Tehrani, K. Suzuki, K. Takahashi, and T. Fujii, "View synthesis using superpixel based inpainting capable of occlusion handling and hole filling," in *Proc. PCS*, May 2015, pp. 124–128.

[21] S.-P. Lu, B. Ceulemans, A. Munteanu, and P. Schelkens, "Spatio-temporally consistent color and structure optimization for multiview video color correction," *IEEE Trans. Multimedia*, vol. 17, no. 5, pp. 577–590, 2015.

[22] S. Ye, S.-P. Lu, and A. Munteanu, "Color correction for large-baseline multiview video," *Signal Process. Image Commun.*, vol. 53, pp. 40–50, 2017.

[23] N. Snavely, S. M. Seitz, and R. Szeliski, "Photo tourism: exploring photo collections in 3D," *ACM Trans. Graph.*, vol. 25, no. 3, pp. 835–846, 2006.

[24] P. Cignoni, M. Callieri, M. Corsini, M. Dellepiane, F. Ganovelli, and G. Ranzuglia, "Meshlab: an open-source mesh processing tool." in *Proc. EICC*, vol. 2008, 2008, pp. 129–136.

[25] A. Dziembowski, A. Grzelka, D. Mieloch, O. Stankiewicz, K. Wegner, and M. Domaski, "Multiview synthesis – improved view synthesis for virtual navigation," in *Proc. PCS*, 2016.

[26] L. C. Zitnick, S. B. Kang, M. Uyttendaele, S. Winder, and R. Szeliski, "High-quality video view interpolation using a layered representation," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 600–608, 2004.

[27] J. Kopf, F. Langguth, D. Scharstein, R. Szeliski, and M. Goesele, "Image-based rendering in the gradient domain," *ACM Trans. Graph.*, vol. 32, no. 6, p. 199, 2013.

[28] P. Pérez, M. Gangnet, and A. Blake, "Poisson image editing," *ACM Trans. Graph.*, vol. 22, no. 3, pp. 313–318, 2003.

[29] P. Ndjiki-Nya, M. Koppel, D. Doshkov, H. Lakshman, P. Merkle, K. Muller, and T. Wiegand, "Depth image-based rendering with advanced texture synthesis for 3-D video," *IEEE Trans. Multimedia*, vol. 13, no. 3, pp. 453–465, 2011.

[30] J. Jin, A. Wang, Y. Zhao, C. Lin, and B. Zeng, "Region-aware 3-D warping for dibr," *IEEE Trans. Multimedia*, vol. 18, no. 6, pp. 953–966, 2016.

[31] N. Plath, S. Knorr, L. Goldmann, and T. Sikora, "Adaptive image warping for hole prevention in 3D view synthesis," *IEEE Trans. Image Process.*, vol. 22, no. 9, pp. 3420–3432, 2013.

[32] D. Gallup, J.-M. Frahm, P. Mordohai, Q. Yang, and M. Pollefeys, "Real-time plane-sweeping stereo with multiple sweeping directions," in *Proc. CVPR*. IEEE, 2007, pp. 1–8.

[33] P. Goorts, C. Ancuti, M. Dumont, and P. Bekaert, "Real-time Video-Based View Interpolation of Soccer Events using Depth-Selective Plane Sweeping," in *Proc. VISAPP*. INSTICC, 2013.

[34] I. Daribo and B. Pesquet-Popescu, "Depth-aided image inpainting for novel view synthesis," in *Proc. MMSP*, 2010, pp. 167–170.

[35] P. Buyssens, M. Daisy, D. Tschumperlé, and O. Lézoray, "Depth-aware patch-based image disocclusion for virtual view synthesis," in *SIGGRAPH Asia 2015 Technical Briefs*. ACM, 2015, pp. 2:1–2:4.

[36] Z. Tauber, Z. Li, and M. Drew, "Review and preview: Disocclusion by inpainting for image-based rendering," *IEEE Trans. Syst., Man, Cybern. C*, vol. 37, no. 4, pp. 527–540, 2007.

[37] V. Kolmogorov and R. Zabih, "Multi-camera scene reconstruction via graph cuts," in *Proc. ECCV*. Springer, 2002, pp. 82–96.

[38] C. Zhu and S. Li, "Depth image based view synthesis: New insights and perspectives on hole generation and filling," *IEEE Trans. Broadcast.*, vol. 62, no. 1, pp. 82–93, 2016.

[39] A. Criminisi, P. Prez, and K. Toyama, "Region filling and object removal by exemplar-based image inpainting," *IEEE Trans. Image Process.*, vol. 13, no. 9, pp. 1200–1212, 2004.

[40] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester, "Image inpainting," in *Proc. SIGGRAPH*. ACM, 2000, pp. 417–424.

[41] J. Gautier, O. L. Meur, and C. Guillemot, "Depth based image completion for view synthesis," in *Proc. 3DTV*, May 2011, pp. 1 –4.

[42] O. Le Meur, J. Gautier, and C. Guillemot, "Examplar-based inpainting based on local geometry," in *Proc. ICIP*, 2011, pp. 3401–3404.

[43] I. Ahn and C. Kim, "A novel depth-based virtual view synthesis method for free viewpoint video," *IEEE Trans. Broadcast.*, vol. 59, no. 4, pp. 614–626, 2013.

[44] S.-P. Lu, B. Ceulemans, A. Munteanu, and P. Schelkens, "Performance optimizations for patchmatch-based pixel-level multiview inpainting," in *Proc. IC3D*, 2013, pp. 1–7.

[45] A. A. Efros and W. T. Freeman, "Image quilting for texture synthesis and transfer," in *Proc. SIGGRAPH*. ACM, 2001, pp. 341–346.

[46] Y. Zhao, C. Zhu, Z. Chen, and L. Yu, "Depth no-synthesis-error model for view synthesis in 3-D video," *IEEE Trans. Image Process.*, vol. 20, no. 8, pp. 2221–2228, 2011.

[47] C.-M. Cheng, S.-J. Lin, and S.-h. Lai, "Spatio-temporally consistent novel view synthesis algorithm from video-plus-depth sequences for autostereoscopic displays," *IEEE Trans. Broadcast.*, vol. 57, no. 2, pp. 523–532, 2011.

[48] T. Emori, M. P. Tehrani, K. Takahashi, and T. Fujii, "Free-viewpoint video synthesis from mixed resolution multi-view images and low resolution depth maps," in *SPIE/IS&T Electronic Imaging*. International Society for Optics and Photonics, 2015, pp. 93 911C–93 911C.

[49] O. Stankiewicz, K. Wegner, and M. Domanski, "Depth estimation based on maximization of a posteriori probability," in *Proc. ICCVG*. Springer, 2016, pp. 253–265.

[50] D. Q. Huynh, "Metrics for 3d rotations: Comparison and analysis," *Journal of Mathematical Imaging and Vision*, vol. 35, no. 2, pp. 155–164, 2009.

[51] U. Fecker, M. Barkowsky, and A. Kaup, "Histogram-based prefiltering for luminance and chrominance compensation of multiview video," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 18, no. 9, pp. 1258–1267, 2008.

[52] A. Schenkel, "Corrections géométriques et colorimétriques automatisées de modèles 3D de grande taille," Ph.D. dissertation, Université Libre de Bruxelles, January 2017.

[53] H. Badino, D. Huber, Y. Park, and T. Kanade, "Fast and accurate computation of surface normals from range images," in *Proc. ICRA*. IEEE, 2011, pp. 3084–3091.

[54] T. Ruzic and A. Pizurica, "Context-aware patch-based image inpainting using markov random field modelling," *IEEE Trans. Image Process.*, 2014.

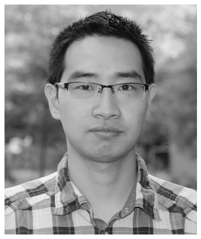
[55] G. Tech, Y. Chen, K. Müller, J.-R. Ohm, A. Vetro, and Y.-K. Wang, "Overview of the multiview and 3D extensions of high efficiency video coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 1, pp. 35–49, 2016.

[56] C.-K. Liang, C.-C. Cheng, Y.-C. Lai, L.-G. Chen, and H. H. Chen, "Hardware-efficient belief propagation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 21, no. 5, pp. 525–537, 2011.

[57] "Recommendation 500-13: Methodology for the subjective assessment of the quality of television pictures," January 2012.

[58] A. Dricot, J. Jung, M. Cagnazzo, B. Pesquet, F. Dufaux, P. T. Kovács, and V. K. Adhikarla, "Subjective evaluation of super multi-view compressed contents on high-end light-field 3D displays," *Signal Process. Image Commun.*, vol. 39, pp. 369–385, 2015.

Beerend Ceulemans is a joint Ph.D candidate in the Department of Electronics and Informatics (ETRO) at the Vrije Universiteit Brussel (VUB) and the Laboratories of Image, Signal processing and Acoustics (LISA) at l'Université Libre de Bruselles (ULB). His research interest are centered on virtual viewpoint synthesis for autostereoscopic 3D screens and free viewpoint video.



Shao-Ping Lu is a senior researcher in the Department of Electronics and Informatics (ETRO) at Vrije Universiteit Brussels (VUB). He received the Ph.D. degree in Computer Science at Tsinghua University, China, in 2012. From Nov. 2012, he has been a postdoctoral researcher at VUB. His primary research areas are image and video processing, with particular interests in multiview video acquisition, representation, compression and rendering.

Gauthier Lafruit is professor at l'Université Libre de Bruxelles (ULB), Brussels, Belgium, in the Laboratory for Image, Signal and Audio processing (LISA). He received his Ph.D. degree in Electrical Engineering from the Vrije Universiteit Brussel, Brussels, Belgium, in 1995. His current research includes Virtual Reality from camera captured content, Light Fields, Computational Imaging and GPU acceleration. He is currently co-chair of the MPEG-i Visual working group.

Adrian Munteanu is professor at Vrije Universiteit Brussel, Belgium. His research interests include image, video and 3D graphics compression, error-resilient coding and multimedia transmission over networks. He is the author of more than 300 journal and conference publications, book chapters and contributions to standards, and received several awards for his work. Adrian Munteanu served as Associate Editor for IEEE Transactions on Multimedia.