

Seamless View Synthesis Through Texture Optimization

Wenxiu Sun, *Student Member, IEEE*, Oscar C. Au, *Fellow, IEEE*, Lingfeng Xu, *Student Member, IEEE*, Yujun Li, and Wei Hu, *Student Member, IEEE*

Abstract—In this paper, we present a novel view synthesis method named Visto, which uses a reference input view to generate synthesized views in nearby viewpoints. We formulate the problem as a joint optimization of inter-view texture and depth map similarity, a framework that is significantly different from other traditional approaches. As such, Visto tends to implicitly inherit the image characteristics from the reference view without the explicit use of image priors or texture modeling. Visto assumes that each patch is available in both the synthesized and reference views and thus can be applied to the common area between the two views but not the out-of-region area at the border of the synthesized view. Visto uses a Gauss–Seidel-like iterative approach to minimize the energy function. Simulation results suggest that Visto can generate seamless virtual views and outperform other state-of-the-art methods.

Index Terms—View synthesis, non-local, Gauss-Seidel-like, texture optimization.

I. INTRODUCTION

IN RECENT years, there have been many researches on 3D related image and video processing. One problem is called view synthesis in which one aims to generate virtual views from one or more captured views. This problem occurs in many applications such as 3D video coding [1], [2], free viewpoint video [3], [4], 2D-to-3D video conversion [5], [6], 3D movie production [7], virtual reality [8], etc. This paper is about view synthesis using one view and one depth.

Given a set of pre-captured images or views of a real scene, view synthesis is to synthesize photo-realistic novel views of the same scene from a virtual camera by processing the real images. This is also called Image-Based Rendering (IBR), especially in early papers. While the term image-based rendering first appeared in the papers [9] and [10], the earlier paper [11] on view interpolation is considered as a seminal work on IBR. IBR methods vary with the 3D representations (*i.e.* how the 3D world is represented in recordable data). Previous works on 3D representation and associated rendering techniques can be classified into three

categories according to how much geometric information is used [12]: rendering without geometry, rendering with implicit geometry, and rendering with explicit geometry.

Typically, early IBR methods belonged to the category of rendering without geometry which is the class of methods that use many aligned images from different view angles in a scene to generate the virtual view using ray-space geometry without requiring any geometric information. They were often used in light field rendering [13], lumigraph [14], and plenoptic modeling [10]. According to the plenoptic sampling theory [15], the minimum view sampling rate (camera spacing) for light field rendering is less than 1 pixel for quality plenoptic modeling. But it is impossible to place cameras that close. Thus, they could only place the cameras as close as possible and apply IBR to generate the missing views using the relatively sparsely sampled views. Compared with rendering with explicit geometry, rendering without geometry has much higher view sampling density with a huge amount of redundant data.

The category of rendering with implicit geometry consists of methods that rely only on implicit geometry without any 3D geometry explicitly available. These implicit geometries are typically expressed in terms of feature correspondence among images. For example, Chen and William's view interpolation method [11] generates novel views by moving pixels in the reference image with interpolated offset vectors between the two or more reference views. The offset vectors of the corresponding pairs are automatically determined by camera transformation and image range data. Similarly, another method called view morphing [16] reconstructs any viewpoint along the camera baseline by using a linear combination of the corresponding pairs in their rectified parallel views.

The category of rendering with explicit geometry contains methods in which explicit 3D geometry is available, often in the form of depth map or 3D coordinates. In general, IBR with explicit geometry offers most flexibility in view synthesis among the three categories, as it allows almost any camera positions and angles to be synthesized but the other two categories allow only limited choices. When explicit 3D geometry information is available for every pixel in one or more images, 3D warping [17] can be used to render views from any nearby camera positions and angles. In 3D warping, an image pixel is projected to their 3D locations, and then re-projected onto the new view. If the original and new views are rectified, 3D warping degenerates to simple horizontal shifting of pixels, with the shifting amount being their disparity values,

Manuscript received August 13, 2012; revised March 14, 2013, June 27, 2013, and October 24, 2013; accepted October 25, 2013. Date of publication November 8, 2013; date of current version November 28, 2013. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Pascal Frossard.

The authors are with the Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong (e-mail: eeshine@ust.hk; eeau@ust.hk; lingfengxu@ust.hk; liyujun@ust.hk; huwei@ust.hk).

Color versions of one or more of the figures in this paper are available online at <http://ieeexplore.ieee.org>.

Digital Object Identifier 10.1109/TIP.2013.2289994

each of which being a function of the corresponding depth value. In this category, when the 3D geometry used is depth, the methods are also called Depth-Image-Based Rendering (DIBR) [2].

In general, most of the existing DIBR view synthesis methods offer “piece-meal” solutions with individual tools to address individual problems, often resulting in unnatural virtual views. In this paper we propose a novel DIBR-based view synthesis method called Visto. In Section II, we give a review of existing DIBR methods. In Section III, we propose the novel integrated DIBR method called Visto. In Section IV, we present simulation results of Visto followed by complexity analysis in Section V. In Section VI, we draw the final conclusion and discuss some future work.

II. RELATED WORKS

In rendering with explicit geometry and in DIBR in particular, there are many challenges associated with 3D warping including inaccurate depth map, occlusions, disocclusions (or holes), ringing artifacts, unnaturally sharp edges, etc. Most of the existing DIBR methods often use some preprocessing steps to detect and refine the inaccurate depth maps, modify 3D warping to handle occlusions, and use some postprocessing steps to handle the holes, to suppress ringing artifacts and to remove unnaturally sharp edges.

DIBR methods assume the availability of a depth map which is often obtained by a procedure called depth estimation. Unfortunately, depth estimation is not perfect and the resulting depth map can be inaccurate, especially at object (*i.e.* depth) boundaries. Most depth estimation algorithms use a procedure called stereo matching in which two corresponding points in two views are matched with a vector specifying their coordinate disparity. While stereo matching works well in texture regions, it is well known to be inaccurate in textureless regions due to the availability of many almost identical candidates, and in occlusion regions due to the loss of correspondence when the object is visible in one view but not visible in the other. Although there are a lot of work to improve stereo matching [18] by using different matching costs, aggregation and global optimization, the estimated depth map still tends to be inaccurate in textureless regions and occlusion regions near depth boundaries. In 3D warping, the inaccurate depth map can cause pixels to be projected to wrong locations in the virtual view. Such wrong projections tend to be acceptable in textureless regions but can cause severe visual artifacts in and around occlusion regions. To minimize the impact of depth map error, Kauff *et al.* [19] detect possible depth map mismatch by using a pair-wise consistency check between each pair of correspondence so that remedial depth map refinement can be performed. However, the checking may not be robust as it is applied on the estimated disparity map only without requiring the object boundaries in the depth map and the texture image to match. Besides, Min *et al.* [20] perform depth denoising based on a joint histogram of texture image and depth map.

Depth map can be ambiguous. Often texture images are captured with edges that are slightly blurred, if not severely

blurred, due to imperfect lens with low-pass characteristics, large aperture with narrow field-of-depth, out-of-focus, object and camera motion, etc. The transition regions of the blurred edges tend to have ambiguous edge locations and depth values, often causing part of the foreground color to appear in the background during 3D warping, and vice versa. This tends to result in some perceptually disturbing ringing artifacts around depth boundaries. To handle the ringing artifacts, one common approach is to perform reliability-based classification and generate the virtual views by blending only the reliable regions from its reference views. Zitnick *et al.* [21] divide a reference view into two regions (or layers), boundary and non-boundary regions, and renders them separately. Boundaries are extracted based on the depth map and are treated as unreliable due to the often ambiguous edge locations. On the other hand, Sun *et al.* [22] label the dilated disocclusion regions in the virtual view as the unreliable regions.

Even when the depth map is accurate, two problems naturally occur during 3D warping: occlusion and disocclusion. Occlusion occurs when a pixel (*e.g.* background) is visible in the reference view but becomes occluded by a pixel of shallower depth (*e.g.* foreground) and thus invisible in the virtual view. In such case, there are at least two candidate values for the pixel in the virtual view during 3D warping: one from the background, one from the foreground. To handle occlusion, Chen and Williams [11] use Z-buffering and selects always the front-most pixel. However, problems can occur when the depth information is inaccurate or even unavailable. McMillan and Bishop [10] use ordered warping based on epipolar geometry and provide an alternative solution that is relatively robust to depth errors. Disocclusion occurs when invisible pixels behind some foreground pixels in the reference view become dis-occluded, *i.e.* they become visible as the foreground pixels are “moved” to a different location in the virtual view. In the virtual view, such dis-occluded pixels would have no candidate values and effectively create hole regions which need to be filled. The width of a hole region (or disocclusion area) tends to be large when the disparity difference (or depth difference) of the corresponding neighboring pixels in the reference frame is large. To reduce the hole width, Zhang and Tam [23] preprocess the depth map using an asymmetric filter to smoothen the sharp changes at depth boundaries. In this way, the width of the hole region tends to decrease. Although this method can reduce the amount of work in the hole-filling process, it can also suffer from geometric distortion in the virtual view due to the distorted depth map. In an innovative way to reduce the size of hole regions, Shade *et al.* [24] use a method called Layered Depth Images (LDI) to store not only what is visible in the input image but also some layers of “hidden” surfaces behind the front surface at some selected depths. Such hidden surfaces help to provide candidate values for some dis-occluded pixels, effectively reducing the hole regions. Chang *et al.* [25] improve LDI by introducing LDI-tree and considers the sampling rate and the LDI density.

To fill the holes, classical image inpainting methods such as PDE-based inpainting by Bertalmio *et al.* [26] or exemplar based inpainting by Criminisi *et al.* [27] can be used.

Starting from the surrounding areas around the holes, they iteratively use existing texture pixels to generate new texture pixels and gradually fill the holes. However, straightforward applications of these techniques tend to perform worse in virtual views than in general images because the starting area of the inpainting is on the depth boundaries with ambiguous edges which can greatly affect the accuracy of inpainting. Another problem is that they use both foreground and background pixels to fill the holes, which is inappropriate. As the holes should correspond to the background, only background pixels should be used to fill the hole, not the foreground. To overcome these problems, Oh *et al.* [28] consider the depth values and manipulate the holes and its surrounding area such that the surrounding areas contain only background pixels, with no foreground pixels. Then regular inpainting is applied. Daribo and Pesquet-Popescu [29], Gautier *et al.* [30] extend traditional exemplar-based inpainting to depth-exemplar-based inpainting [27] by considering both depth and texture information. Ndjiki-Nya *et al.* [31], Schmeing and Jiang [32] fill the disoccluded regions by considering the temporal consistency in video frames. Fitzgibbon *et al.* [33] instead optimize the virtual textures directly by using the local image priors from a stack of reference views, allowing the generation of new views which are locally similar to the input views, but globally consistent with the new viewpoint.

In spite of all the above DIBR methods, the overall look of the synthesized images often have unnaturally sharp boundaries. Hasinoff *et al.* [34] represent each boundary as a 3D curve and apply boundary matting, in which alpha matting is applied to smooth the sharp boundaries. Criminisi and Blake [35] also propose a Split-Patch Search with emphasis on recovering the continuity of object boundaries and faithful synthesis of transparency effects.

In this paper we propose a view synthesis method called Visto which belongs to the category of rendering with explicit geometry. In Visto, we use a dense depth map and thus it is a DIBR method. A preliminary version of Visto is published in [36].

III. PROPOSED VIEW SYNTHESIS WITH TEXTURE OPTIMIZATION (VISTO)

In this section, we describe the proposed novel **View Synthesis with Texture Optimization**, which we call *Visto*. Given a reference image (color or gray-scale) captured by a camera with certain camera parameters (including camera location) together with a corresponding estimated depth map, Visto seeks to estimate a virtual view at another location with different camera parameters. Visto assumes that each patch is available in both the synthesized view and the reference view and focuses on the bottomline performance requirement: the reference view and the synthesized view should have similar texture and depth [33]. Visto seeks to address the problems of inaccurate depth map, occlusion, disocclusion, ringing and unnaturally sharp edges in an integral manner. In Visto, we formulate DIBR view synthesis as an energy optimization problem in which we maximize the inter-view texture similarity while preserving its geometric structure by minimizing the

inter-view depth map error. This approach allows the virtual view to implicitly inherit the image characteristics from the reference view without the explicit use of image priors or texture modeling.

Visto is about view synthesis using one view and one depth. When a synthesized view is to be generated from a reference view, much of the synthesized view is typically inside the camera “cover range” of the reference view, though some portion at the border of the synthesized view is new and is out of the cover range. In this paper, we call the area of the synthesized view inside the cover range the “common” area, and we call the rest of the synthesized view that is out of the cover range the “out-of-range” area. In general, our assumption for Visto holds for the common area, but not necessarily for the out-of-range area as the new contents there can be quite different from the common area. In this paper, we focus on using the proposed Visto to perform view synthesis with visually pleasing appearance for the common area.

A. Visto: Problem Formulation

Let $Z = (Z^t, Z^d)$ be the reference view and $X = (X^t, X^d)$ be the virtual view to be synthesized. In this paper, we use superscript t and d to denote the corresponding *texture image* and *depth map*, respectively. For simplicity, we assume that the texture images are gray-scale images with only the luminance component, though our method can be easily extended to color images. We assume Z^t, Z^d, X^t, X^d are all of the same size, $M \times N$. In practical situations, X^t may be the principle output with X^d optional. Let $p = (i, j)$ be a pixel location. Let $Z_p^t, Z_p^d, X_p^t, X_p^d$ be the corresponding values at p . Let N_p be a rectangular patch around p of size $m \times n$. We represent N_p in Z^t, Z^d, X^t, X^d as row-ordered $mn \times 1$ vectors $\mathbf{z}_p^t, \mathbf{z}_p^d, \mathbf{x}_p^t, \mathbf{x}_p^d$, respectively.

We define a *correspondence map* $C = (C_x, C_y)$, where both C_x and C_y are of size $M \times N$, and are the x- and y-component of the offset of the ‘corresponding’ locations in Z . Let $C_x(p)$ and $C_y(p)$ be the elements of C_x and C_y at p such that $C_p = (C_x(p), C_y(p))$. The C maps the patch N_p at location p in X to the patch at location $p + C_p = (i + C_x(p), j + C_y(p))$ in Z . When both the reference and virtual views are rectified, the C_x of the correspondence map is similar to the regular disparity map (denoted as D) and most of the C_y values are zero. But the correspondence map is different from the disparity map because, for any dis-occluded pixel p (visible in the virtual view but not the reference view) such that the disparity is not defined, we will still force p to be mapped to some pixels in Z in the correspondence map. In other words, we will assign some values for $C_x(p)$ and allow $C_y(p)$ to be non-zero. Actually, Visto has a refining process in which we allow both $C_x(p)$ and $C_y(p)$ to change. With this, it is possible for Visto to generate a non-rectified virtual view with arbitrary camera parameters.

We now define the *normalized patch energy* $E_p^t(\mathbf{x}_p^t, C_p|Z^t)$ and $E_p^d(\mathbf{x}_p^d, C_p|Z^d)$ for texture patch and depth patch, respectively, which is a measure of mismatch between the patch at p in X and its corresponding patch in Z through correspondence

map C .

$$E_p^t(\mathbf{x}_p^t, C_p|Z^t) = \frac{1}{mn} \|\mathbf{x}_p^t - \mathbf{z}_{p+C_p}^t\|^2 \quad (1)$$

$$E_p^d(\mathbf{x}_p^d, C_p|Z^d) = \frac{1}{mn} \|\mathbf{x}_p^d - \mathbf{z}_{p+C_p}^d\|^2 \quad (2)$$

As can be seen, $E_p^t(\mathbf{x}_p^t, C_p|Z^t)$ and $E_p^d(\mathbf{x}_p^d, C_p|Z^d)$ are the normalized (*i.e.* per-pixel) Euclidean distances between the two texture patches, and between the two depth map patches, respectively. The normalized values are used here so that they can be compared meaningfully when m and n are changed. Here Z , but not X and C , is written after a vertical bar because Z would not change in Visto. But X and C are variables and would be iteratively updated. We further define the *normalized total energy* $E^t(X^t, C|Z)$ and $E^d(X^d, C|Z)$ for texture image and depth map:

$$\begin{aligned} E^t(X^t, C|Z^t) &= \frac{1}{K} \sum_{p \in \mathbf{P}} E_p^t(\mathbf{x}_p^t, C_p|Z^t) \\ &= \frac{1}{mnK} \sum_{p \in \mathbf{P}} \|\mathbf{x}_p^t - \mathbf{z}_{p+C_p}^t\|^2 \end{aligned} \quad (3)$$

$$\begin{aligned} E^d(X^d, C|Z^d) &= \frac{1}{K} \sum_{p \in \mathbf{P}} E_p^d(\mathbf{x}_p^d, C_p|Z^d) \\ &= \frac{1}{mnK} \sum_{p \in \mathbf{P}} \|\mathbf{x}_p^d - \mathbf{z}_{p+C_p}^d\|^2 \end{aligned} \quad (4)$$

where $\mathbf{P} = \{p_1, p_2, \dots, p_K\}$ is a set of *selected* locations, and K is the cardinality of \mathbf{P} . Let \mathbf{P}^\dagger be the collection of all the pixel locations in the images. Then $\mathbf{P} \subset \mathbf{P}^\dagger$. Let \mathbf{P}_0 be the collection of hole regions in the common area of the virtual view after 3D warping. In this paper, \mathbf{P} is obtained by performing morphological dilation on \mathbf{P}_0 . \mathbf{P} is called the *untrusted region*, while the rest of the common area is called the *trusted region*. In our algorithm, X^t , X^d and C will be allowed to change only in the untrusted region (*i.e.* \mathbf{P}), because the correct solution of X^t , X^d and C may not be the minimum-energy solution in the trusted region, especially for patches with depth discontinuities.

Intuitively, the view synthesis problem can be formulated as the following constrained optimization problem:

$$\begin{aligned} &\underset{\{X_p^t, X_p^d, C_p; p \in \mathbf{P}\}}{\text{minimize}} && E^t(X^t, C|Z^t) \\ &\text{subject to} && E^d(X^d, C|Z^d) \leq \varepsilon, \end{aligned} \quad (5)$$

where $E^t(X^t, C|Z^t)$ is the objective function, $E^d(X^d, C|Z^d)$ is the depth constraint function, ε is the allowance of the depth constraint term. In this paper, we solve the equivalent unconstrained optimization problem by minimizing $E(X, C|Z)$

$$E(X, C|Z) = E^t(X^t, C|Z^t) + \lambda E^d(X^d, C|Z^d) \quad (6)$$

with λ serving as the Lagrange multiplier.

Starting with large patch size (m, n) , we will perform optimization in a series of steps (which we call *levels*) in which the patch size is gradually decreased from one step to the next. In other words, (m, n) is large in *level 0* and is decreased monotonically in subsequent levels. For every

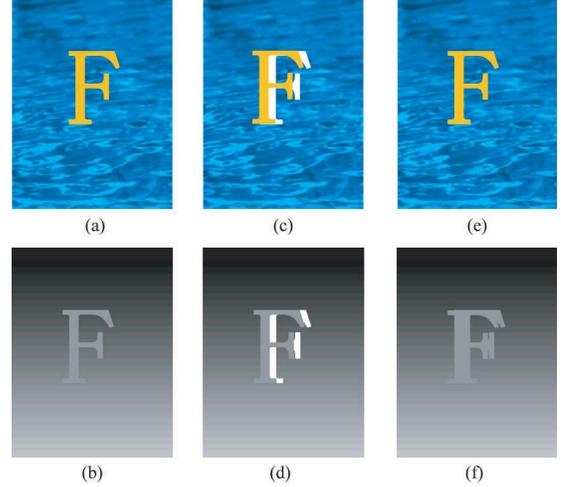


Fig. 1. Comparison of correspondence map and disparity map for rectified views. (a) Reference texture image Z^t . (b) Reference disparity map for Z^t . Brighter intensity means larger disparity. (c) Virtual image X^t after pixel-based 3D warping is applied to (a) and (b). White regions are *disocclusion regions*. (d) Disparity map for (c) during 3D warping. White regions have undefined disparity values. (e) Ground truth for X^t . (f) A possible correspondence map. Every pixel has correspondence though it is not physically meaningful for some.

level, Visto will find iteratively the X and C that minimize $E(X, C|Z)$.

In the rest of the paper, we will use E_p^t , E_p^d and E to mean $E_p^t(\mathbf{x}_p^t, C_p|Z^t)$, $E_p^d(\mathbf{x}_p^d, C_p|Z^d)$ and $E(X, C|Z)$ for the sake of simplicity.

B. Visto: Energy Optimization

In the proposed Visto, we will perform optimization at multiple *levels*. Within each level, several iterative optimizations will be performed. We start with level 0 with an initial choice of the patch size (m, n) . As we progress from level l to level $l+1$, we will allow (m, n) to decrease gradually. In each level, we find the optimal choices of X and C to minimize E at the selected locations \mathbf{P} , using the optimized X and C from the previous level.

1) *Initialization of Correspondence Map C , Texture Image X^t and Depth Map X^d* : Initialization is needed in level 0. Although algorithms should be robust to any initializations, a good initialization can often lower the amount of computation and avoid the result of being trapped in a local minimum. In this section, we propose a simple but effective method to initialize the correspondence map, though other methods are possible. The texture image and depth map in the virtual view are initialized based on the correspondence map. If the reference and virtual images are rectified, each pair of correspondent pixels are in the same horizontal line in the two images. Otherwise, a pre-warp technique [16] could be applied as pre-processing to make the images rectified with known camera parameters, and as post-processing to un-rectify the resulting virtual image. In the rest of the paper, we will assume the images are rectified.

Recall that \mathbf{P} is the untrusted region, and the rest of the common area is the trusted region. The amplitude of C_x in the trusted regions is identical to the warped disparity map D but in the opposite direction, as shown in Fig. 1 (d) and (f),

with zero in the corresponding C_y . For the untrusted region, D is untrustable or undefined, we will initialize both C_x and C_y with some *reasonable* values.

The disparities of pixels in \mathbf{P}_0 are undefined because those pixels are generally newly appeared pixels in the virtual view, and occluded in the reference view. Thus, the pixels in \mathbf{P}_0 cannot be the foreground and we assume they belong to the background. As for pixels outside \mathbf{P}_0 but still in \mathbf{P} , these are untrusted pixels. While the disparities are defined for these pixels, we do not trust them, so we simply group them together with pixels in \mathbf{P}_0 and initialize them in the same way. Typically the pixels on the left and right of any hole regions would contain the foreground on one side and background on the other side. Thus our simple initialization method is to find the side with the background and guess some C_x values so that pixels in the undefined regions correspond to the background. Since the two views are assumed to be rectified, each horizontal line in the virtual view corresponds to the same line in the reference view. Consider one such line pair in Fig. 2. The top line is a line in the reference view with grey pixels being foreground and black pixels being background. The bottom line is in the virtual view in which the grey foreground pixels are shifted to the left by an amount (disparity) larger than the background pixels. With the different shifting, or disparity, two dis-occluded pixels are created and they are in the undefined region \mathbf{P} with undefined disparity. Our initialization is to find some reasonable guess of the C_x for the white pixels, with corresponding $C_y = 0$. Our desire is to fill up the two pixels with two pixels from the background. We note that if we force C_x to take on the background disparity, the white pixels will be essentially copied from the foreground, which is wrong. Instead, if we force C_x to take on the foreground disparity, the white pixels will be essentially copied from the background, which is right. We thus want to find the foreground disparity from the trusted neighboring pixels. We note that the foreground, by definition, would have a larger disparity in its amplitude than the background. Thus we simply examine the disparity of the defined neighboring pixels on the left and on the right of the untrusted pixel and choose the one which is larger. In addition, let $rnd(a, b)$ be a random number between positive a and b . For each pixel in the untrusted region, we add a random number to the chosen disparity, avoiding Visto being trapped in some bad energy local minima. This concept of random element is similar to that of [37]. Typically, we choose $a = 3, b = 10$.

Mathematically,

$$C_x(i, j) = \begin{cases} -D(i, j) & \text{if } (i, j) \notin \mathbf{P} \\ -D(\tilde{i}, j) - \text{sgn}(D) * rnd(a, b) & \text{if } (i, j) \in \mathbf{P} \end{cases} \quad (7)$$

$$\tilde{i} = \arg \max_{i_l, i_r} \{|D(i_l, j)|, |D(i_r, j)|\} \quad (8)$$

$$i_l = \max\{i' | i' < i, (i', j) \notin \mathbf{P}\} \quad (9)$$

$$i_r = \min\{i' | i' > i, (i', j) \notin \mathbf{P}\} \quad (10)$$

$$C_y(i, j) = 0 \quad (11)$$

where $sgn(D)$ is the sign of the warped disparity.

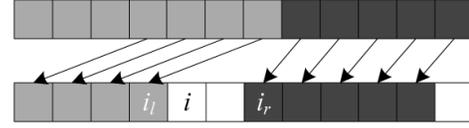


Fig. 2. A line of pixels in the reference view (top line) shift to the virtual view (bottom line) by the amount of their disparity values, with grey pixels being foreground, black pixels being background, white pixels being undefined.

After initializing C , in all image regions $p \in \mathbf{P}^\dagger$, X^t and X^d are easily obtained by applying (13) and (14) respectively, which will be explained later. This process is equivalent to patch-wise backward mapping, where neighboring patches overlap with each other.

2) *Optimization Within One Level:* For a given level with fixed (m, n) , we seek to find three sets of variables, X^t , X^d and C , to minimize E . For any location $p \in \mathbf{P}$, C_p is an indexing term to map the patch \mathbf{x}_p^t and \mathbf{x}_p^d in the virtual view to the patch $\mathbf{z}_{p+C_p}^t$ and $\mathbf{z}_{p+C_p}^d$ in the reference view, respectively. Effectively, the collection of all the patches $\{\mathbf{z}_q^t, \mathbf{z}_q^d | q \in \mathbf{P}^\dagger\}$ in the reference view forms a *patch dictionary*. Each patch in the dictionary carries some local image characteristics of the reference view. In Visto, we do not enforce any image prior in synthesizing the virtual view because the image prior is implicitly inherited from the chosen patches in the patch dictionary. Because C_p is an indexing term, most popular optimization methods will not work for our energy formulation which contains both values and their indexing terms. Instead, motivated by the Gauss-Seidel optimization method, we optimize our energy function E in an iterative manner. We will give a proof that this optimization procedure will converge.

Within one iteration, one set of variables (e.g. C) is derived by minimizing (6) while keeping the other two sets (e.g. X^t , X^d) unchanged, as described in Algorithm 1. Let $X^{t,l,r}$, $X^{d,l,r}$ and $C^{l,r}$ be the corresponding X^t , X^d and C in the r^{th} iteration of level l . When the iterative optimization converges, we use the symbols $X^{t,l,\infty}$, $X^{d,l,\infty}$ and $C^{l,\infty}$ to represent the converged values. The $X^{t,l,0}$, $X^{d,l,0}$ and $C^{l,0}$ are the initial values at the beginning of level l optimization. We choose the converged values from the previous level as the initial values of the current level, i.e. $X^{t,l,0} = X^{t,l-1,\infty}$, $X^{d,l,0} = X^{d,l-1,\infty}$ and $C^{l,0} = C^{l-1,\infty}$.

In iteration r , we fix X^t at $X^{t,l,r-1}$, X^d at $X^{d,l,r-1}$ and perform a search to find a better C within some corresponding search windows around $C^{l,r-1}$. Then we fix C at $C^{l,r}$ and find a better X^t and X^d .

Consider a pixel location $p \in \mathbf{P} \subset \mathbf{P}^\dagger$. Recall that $C_p^{l,r-1}$ is the correspondence vector for the pixel at p after the $(r-1)^{th}$ iteration. In the r^{th} iteration, we allow the correspondence vector to move within a search window of $(\pm w_x, \pm w_y)$ around $C_p^{l,r-1}$ such that

$$C_p^{l,r} = C_p^{l,r-1} + \underset{|\Delta C_x| \leq w_x, |\Delta C_y| \leq w_y}{\operatorname{argmin}} [E_p^t(\mathbf{x}_p^{t,l,r-1}, C_p^{l,r-1} + \Delta C | Z^t) + \lambda E_p^d(\mathbf{x}_p^{d,l,r-1}, C_p^{l,r-1} + \Delta C | Z^d)] \quad (12)$$

Algorithm 1 Energy Minimization at Level l

Initialization: $X^{t,l,0} = X^{t,l-1,\infty}$, $X^{d,l,0} = X^{d,l-1,\infty}$,
 $C^{l,0} = C^{l-1,\infty}$.

for $r = 1 \rightarrow R$ **do**

 Compute $C_p^{l,r}$ for all $p \in \mathbf{P}$ using (12);

 Compute $X^{t,l,r}(p)$ for all $p \in \mathbf{P}$ using (13);

 Compute $X^{d,l,r}(p)$ for all $p \in \mathbf{P}$ using (14);

if (15) is true **then**

break;

end if

end for

where $\Delta C = (\Delta C_x, \Delta C_y)$ is the change in the correspondence vector from one iteration to the next, and $\mathbf{x}_p^{t,l,r-1}$, $\mathbf{x}_p^{d,l,r-1}$ are the column vector \mathbf{x}_p^t , \mathbf{x}_p^d in the $(r-1)^{th}$ iteration of level l , respectively. Typically, we choose $w_x = \alpha m$, $w_y = \alpha n$ for some constant α . Then we fix C at $C^{l,r}$ and solve for a better X^t and X^d by minimizing (6). Since (6) is a quadratic function of X_p^t for any location p , we take the derivative of it with respect to X_p^t and set the result to zero to obtain the following closed-form optimal X^t at p

$$X_p^{t,l,r} = \frac{1}{Q} \sum_{q \in \hat{N}_p} Z_{p+C_q^l}^t \quad (13)$$

where $\hat{N}_p = \{p' | p \in N_{p'}, p' \in \mathbf{P}^\dagger\}$ is a collection of neighboring points in \mathbf{P}^\dagger whose patches contain p , and Q is the cardinality of \hat{N}_p such that $X_p^{t,l,r}$ is effectively the average of the corresponding Z^t values. In a similar way, we obtain X_p^d for any location p :

$$X_p^{d,l,r} = \frac{1}{Q} \sum_{q \in \hat{N}_p} Z_{p+C_q^l}^d \quad (14)$$

The iteration will stop in level l if the decrement percentage of E is less than a threshold T , *i.e.*

$$\frac{E(X^{t,l,r}, X^{d,l,r}, C^{l,r} | Z) - E(X^{t,l,r-1}, X^{d,l,r-1}, C^{l,r-1} | Z)}{\max\{E(X^{t,l,r-1}, X^{d,l,r-1}, C^{l,r-1} | Z), \theta\}} \leq T \quad (15)$$

where θ is a small constant. We also impose a maximum number of iterations R . We now prove that the iteration will converge.

Theorem 1: The energy $E(X^{l,r}, C^{l,r} | Z)$ is a monotonic decreasing function of r ($r \in \mathbb{Z}$), *i.e.*

$$E(X^{t,l,r}, X^{d,l,r}, C^{l,r} | Z) \leq E(X^{t,l,r-1}, X^{d,l,r-1}, C^{l,r-1} | Z) \quad (16)$$

Proof: With $X^{t,l,r-1}$ and $X^{d,l,r-1}$ fixed, $C_p^{l,r}$ is the correspondence map within a search range ($\pm w_x, \pm w_y$) around $C_p^{l,r-1}$ with minimum energy E , for any p . Thus for $C^{l,r}$ with all the $C_p^{l,r}$,

$$E(X^{t,l,r-1}, X^{d,l,r-1}, C^{l,r} | Z) \leq E(X^{t,l,r-1}, X^{d,l,r-1}, C^{l,r-1} | Z). \quad (17)$$

With $C^{l,r}$ fixed, $X_p^{t,l,r}$ and $X_p^{d,l,r}$ are the pixel values at p that minimizes E for any p . Thus for $X^{t,l,r}$ with all the $X_p^{t,l,r}$ and

$X^{d,l,r}$ with all the $X_p^{d,l,r}$,

$$E(X^{t,l,r}, X^{d,l,r}, C^{l,r} | Z) \leq E(X^{t,l,r-1}, X^{d,l,r-1}, C^{l,r} | Z). \quad (18)$$

such that

$$\begin{aligned} E(X^{t,l,r}, X^{d,l,r}, C^{l,r} | Z) &\leq E(X^{t,l,r-1}, X^{d,l,r-1}, C^{l,r} | Z) \\ &\leq E(X^{t,l,r-1}, X^{d,l,r-1}, C^{l,r-1} | Z). \end{aligned} \quad (19)$$

As $E(X^{t,l,r}, X^{d,l,r}, C^{l,r} | Z)$ is lower-bounded by zero and is a monotonic decreasing function, it has to converge. Again, we note that our algorithm is somewhat similar to the Gauss-Seidel optimization.

Intuitively, our algorithm tries to find reasonable texture to “inpaint” the dis-occluded region. In each iteration, the correspondence map (pointing to the best match) is firstly chosen from Z patch-by-patch based on the latest estimate of the texture image. Then, X^t and X^d are solved based on the refined C . Consider a pixel location p . It is in the neighborhood $\hat{N}_{p'}$ of many p' . There are $m \times n$ such p' . Each p' suggests a candidate correspondence map value (*i.e.* $C_{p'}$) for p . Thus, the new X^t and X^d are the average of the candidate Z^t and Z^d values corresponding to these $m \times n$ candidate correspondence map values respectively. The resulting X^t and X^d may be initially blurred, if the candidate Z^t and Z^d values are very different. However, the X^t and X^d will help to give better C which in turn helps to reduce the blurriness in X^t and X^d as the iterations continue.

3) *Visto: Optimization Across Levels:* The algorithm described in the previous subsection allows us to find the optimal X^t and X^d and C for a given level (*i.e.* the patch size of $m \times n$). There is a trade-off between large and small patch size. A larger patch size allows more texture that would help to avoid being trapped in local minima, which is desirable. But a large patch size tends to give relatively blurred texture X^t and depth X^d , which is undesirable. On the other hand, a small patch size tends to give sharper texture and depth, which is desirable, but it can be trapped in local minima which is undesirable. Thus Visto uses a multi-level optimization procedure, as described in Algorithm 2, in which the patch size is large in the initial level but is decreased gradually with some *schedule* in the subsequent levels. Optimization is performed in each level. The patch size reduction schedule helps to approach the global minima, similar to the temperature reduction schedule in Simulated Annealing - a well known multi-level global optimization algorithm.

IV. SIMULATION RESULTS

In this section, we simulate the proposed algorithm, Visto, to study its behavior as the iteration progresses within each level (patch size) and as the patch size decreases. We then test the performance of Visto with different baseline distances. Lastly, we compare its performance with some existing state-of-the-art methods.



Fig. 3. 1st frame of test sequences. From left to right: PoznanStreet, Undo_Dancer, Champagne_tower, Pantomime, Balloons, Newspaper, Mobile. Top row: Texture image; Bottom row: Depth map.

Algorithm 2 Energy Minimization Across Levels

```

for  $l = 0 \rightarrow L$  do
  if  $l == 0$  then
    Initialize  $C^{0,0}$ .
    Compute  $X^{t,0,0}$  from (13) using  $C^{0,0}$ .
    Compute  $X^{d,0,0}$  from (14) using  $C^{0,0}$ .
    Initialize  $(m^0, n^0)$ . Compute  $(w_x^0, w_y^0) = \alpha(m^0, n^0)$ .
  else
     $X^{t,l,0} = X^{t,l-1,\infty}, X^{d,l,0} = X^{d,l-1,\infty}$ ,
     $C^{l,0} = C^{l-1,\infty}$ 
     $(m^l, n^l) = (m^{l-1}, n^{l-1})/\beta$ 
     $(w_x^l, w_y^l) = \alpha(m^l, n^l)$ 
  end if
  Solve  $[X^{l,\infty}, C^{l,\infty}] = \underset{X,C}{\operatorname{argmin}} E(X^{l,0}, C^{l,0})$  using Algorithm 1.
end for

```

A. Experiment Setup

Some selected MPEG test sequences [38], including Poznan_Street, Undo_Dancer, Champagne_tower, Pantomime, Balloons, Newspaper, and Mobile as listed in Table I and shown in Fig. 3 of the texture images and their corresponding depth maps, are used in the experiment. Undo_Dancer and Poznan_Street are Class A test sequences with a resolution of 1920×1088 and a camera spacing of 13.75cm. Champagne_tower and Pantomime are Class B test sequences with a resolution of 1280×960 and a camera spacing of 5cm. Balloons and Newspaper are Class C test sequences with a resolution of 1024×768 and camera spacing of 5cm. Mobile is a Class D test sequence with a resolution of 720×540 and a camera spacing of 5cm. Undo_Dancer and Mobile are synthetic videos with ground truth depth data. The other 5 sequences are natural videos. All views in these test sequences have been rectified. The first frame of each video sequence is used in our simulation. For convenience, these YUV420 texture sequences are converted to YUV444 before view synthesis is applied.

For the test sequences, many texture views are available but the depth maps are only available for a few selected views. Comparing texture images and their corresponding depth maps in Fig. 3, it can be observed that the depth maps are imperfect as some depth boundaries do not align with the corresponding object boundaries (e.g. Poznan_Street,

TABLE I
TEST SEQUENCES

Seq.	Name	Res.	Cam.	Provider
S1	Poznan_Street	1920×1088	4 \rightarrow 3	Poznan
S2	Undo_Dancer	1920×1088	2 \rightarrow 3	Nokia
S3	Champagne	1280×960	39 \rightarrow 40	Nagoya
S4	Pantomime	1280×960	39 \rightarrow 40	Nagoya
S5	Balloons	1024×768	3 \rightarrow 4	Nagoya
S6	Newspaper	1024×768	4 \rightarrow 5	GIST
S7	Mobile	720×540	5 \rightarrow 6	Philips

Balloons, Newspaper). The depth maps of some background regions with visually uniform depth can have highly fluctuating values (e.g. Champagne_tower, Balloons, Newspaper). Sometimes the depth values of the background can be even bigger than the foreground objects (e.g. Champagne_tower, Balloons, Newspaper). The inaccurate depth map is a challenge to all view synthesis methods. When examined closely, the edges in the texture images can also be observed to be blurred in all sequences, to different degrees in different sequences. Such blurred edges help to make the images look natural, but is another challenge to view synthesis methods as they can cause ringing artifacts during 3D warping as explained before. In our experiments, we use one original view (e.g. view 3 of Balloons) and its depth map to synthesize adjacent view(s) (e.g. view 4 of Balloons). For the experiments with fixed baseline distance, the original and synthesized view pairs are shown in Table I. These pairs are chosen according to the MPEG test conditions, with the baseline distance of around 5cm. The baseline distance is approximately the distance between the two human eyes. For the experiments with different baseline distances, we synthesize three virtual views on the left side and three on the right side.

B. Study the Behavior of Visto

Firstly, we do experiments to study the behavior of the proposed Visto. For Visto, we use square patches such that $m = n$. We set $T = 0.95$ and $R = 10$. To test the behavior of Visto, we start with certain initial patch size and allow the patch size to decrease using two schedules. In Schedule 1, $(m^l, n^l) = (m^{l-1}, n^{l-1}) - 1$ and the patch size reduces slowly. In Schedule2, $(m^l, n^l) = (m^{l-1}, n^{l-1})/2$ such that the patch size reduces quickly. The results using the two schedules for Mobile are shown in Fig. 4 in which the

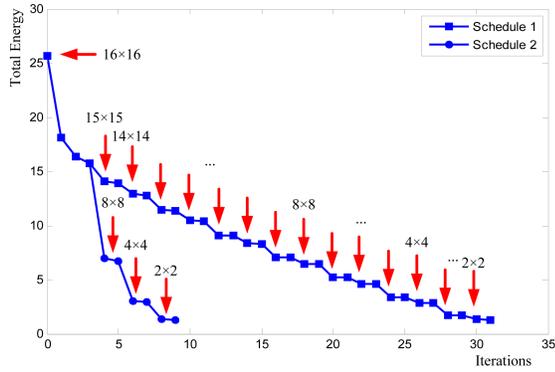


Fig. 4. The decreasing energy in each optimization step by starting at patch size of 16×16 and using two shrink schedules.



Fig. 5. Synthesized texture image and depth map at selected iterations (*i.e.* iteration 0, 3, 17, 27, 31) by starting at patch size of 16×16 and using shrink schedule 1.

initial patch size are 16×16 . In the figure, the vertical axis is the energy $E_Y + E_U + E_V + \lambda E_D$. The λ is set as 2.0 such that the texture energy $E_Y + E_U + E_V$ and the depth energy are of comparable importance. The horizontal axis is the iteration number with iteration 0 being the initial condition. For example, in Fig. 4, there are 3 iterations for patch size 16×16 , and 2 iterations for each of the remaining patch sizes.

As expected, the energy is monotonically decreasing as the iteration progresses using either schedules. We note that the shrinking of patch size is very important, as the converged energy of patch size 16×16 is rather large at 15.75, but can be greatly reduced to 1.28 at iteration 31 after the patch size is gradually reduced to 2×2 under Schedule 1. We note that the two schedules have different number of iterations but manage to reach similar converged energy: 31 iterations for Schedule 1 to reach the converged energy of 1.28, and 9 iterations for Schedule 2 to reach a similar converged energy of 1.30. The results for the other sequences are similar. It appears that Schedule 2, with lower iteration number and complexity and similar converged energy, is better than Schedule 1. The synthesized texture image and depth map using Schedule 1 for Mobile are shown in Fig. 5 for selected iterations: iterations 0, 3, 17, 27, 31 for Fig. 4. Iteration 0 is the initial condition ($X^{t,0,0}$, $X^{d,0,0}$ obtained in the initialization phase) and both the synthesized texture image and depth map contain significant

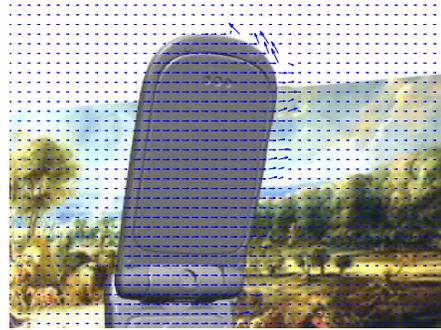


Fig. 6. Final correspondence vector C plotted over the synthesized image. For ease of visualization, the vectors are scaled and only one vector is plotted for each 8×8 block.

ringing artifacts at the hole region (the right side of the mobile phone). During the initialization phase, occlusion is handled by the ordered 3D-warping [39]. Unlike other hole filling methods that tend to generate unnaturally sharp edges, our $X^{t,0,0}$ and $X^{d,0,0}$ tend to be blurred in the untrusted regions due to the averaging effect in (13) and (14). Iterations 3, 17, 27 and 31 are the results upon the convergence at patch size of 16×16 , 8×8 , 4×4 , and 2×2 respectively. We can observe that the ringing artifacts at the hole region in both the texture image and depth map are progressively improved as the patch size reduces. The edges in the untrusted regions are becoming sharp also. The results using Schedule 2 behaves very similarly and thus are not shown here. A typical final correspondence map is shown in Fig. 6. Note that, while the correspondence vectors in the trusted regions are horizontal and pointing to the right as expected, some in the untrusted regions are not horizontal and can be pointed to any directions.

To study the sensitivity to the initial patch size, we simulate Visto using four choices of initial patch size: 16×16 , 8×8 , 4×4 , and 2×2 . For each patch size, we test two shrink schedules: Schedule 1 and Schedule 2 defined above. The SSIM [40] and PSNR values of the untrusted regions (hole region with dilation) of the synthesized view of the seven test sequences are shown in Fig. 7 and Fig. 8 respectively. The regions outside the untrusted regions are not compared as the Visto iterations do not change them and they are identical for all initial patch sizes and shrink schedules. In Fig. 7, the horizontal axis indicates various combination of initial patch size and shrink schedule. The initial patch size is 16×16 for 1 and 2, 8×8 for 3 and 4, 4×4 for 5 and 6 and 2×2 for 7. Schedule 1 (slow) is used for 1, 3, and 5. Schedule 2 (fast) is used for 2, 4, 6. For 7, no shrinking is applied. For each of these cases, five values are shown: initial, final, maximum, minimum and average. According to Fig. 4, there are 31 iterations after iteration 0 for the Mobile sequence, for initial patch size of 16×16 and Schedule 1. For this case (horizontal coordinate = 1), a vertical line with three dots are shown in Fig. 7(g). The lowest dot is the initial SSIM (0.4884) at iteration 0 and the highest dot is the final SSIM (0.5614) at iteration 31. The middle dot is the average SSIM (0.5478). The lower

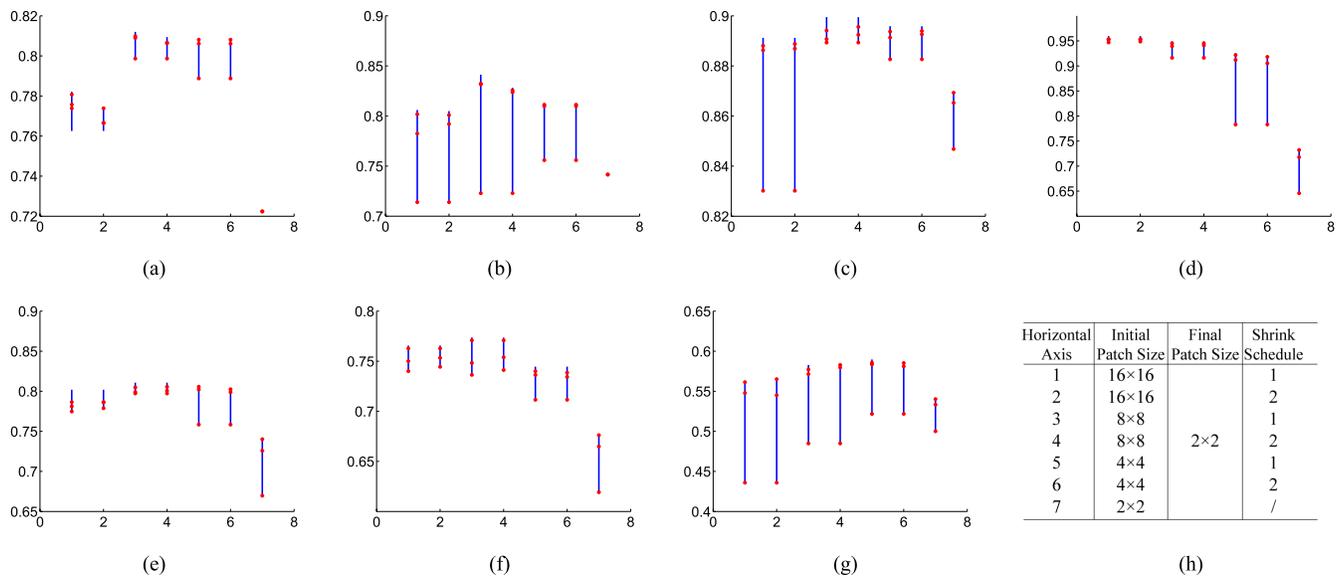


Fig. 7. SSIM values at untrusted regions of tested images by starting with four starting patch sizes of 16×16 , 8×8 , 4×4 , 2×2 and two shrink schedules. The lower end, upper end, lowest dot, highest dot, middle dot in each vertical line are the minimum, maximum, initial, final, and average SSIM values of all iterations respectively. (a) Poznan_Street. (b) Undo_Dancer. (c) Champagne_tower. (d) Pantomime. (e) Balloons. (f) Newspaper. (g) Mobile. (h) Configurations of figure (a-g).

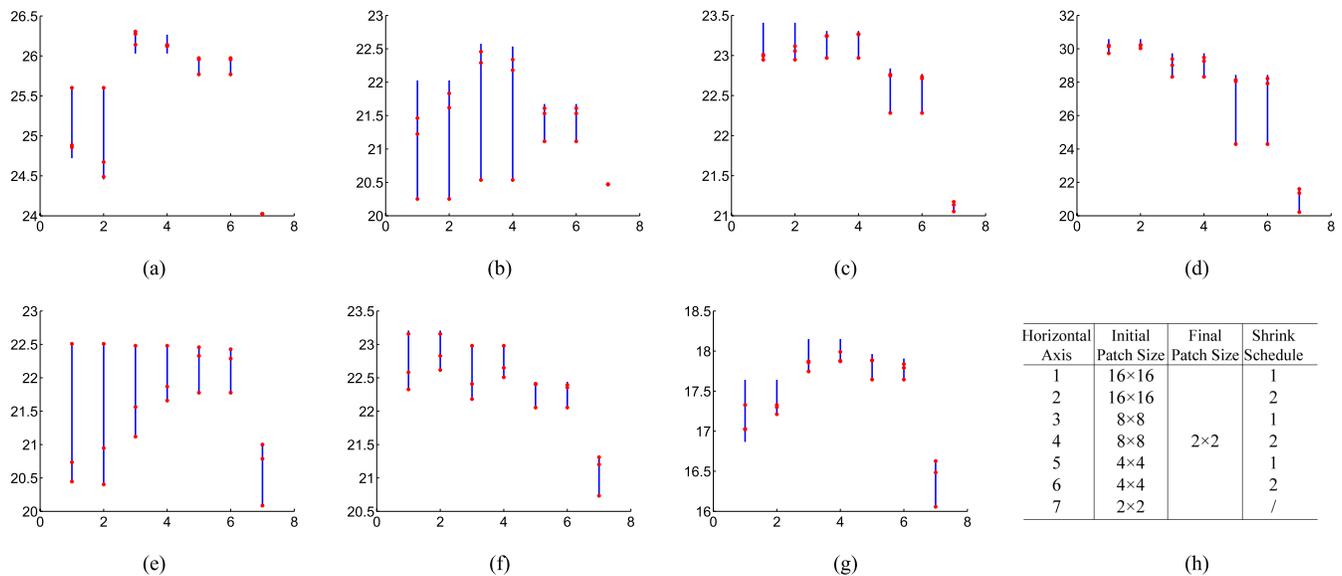


Fig. 8. PSNR values at untrusted regions of tested images by starting with four starting patch sizes of 16×16 , 8×8 , 4×4 , 2×2 and two shrink schedules. The lower end, upper end, lowest dot, highest dot, middle dot in each vertical line are the minimum, maximum, initial, final, and average PSNR values respectively. (a) Poznan_Street. (b) Undo_Dancer. (c) Champagne_tower. (d) Pantomime. (e) Balloons. (f) Newspaper. (g) Mobile. (h) Configurations of figure (a-g).

and upper ends of the vertical line indicate the minimum and maximum SSIM. For Mobile, the initial SSIM is the minimum. The maximum SSIM is 0.5618 which is higher than the final SSIM.

In our experiments, we observe that SSIM tends to reflect our subjective perception of the synthesized views more closely than PSNR. However, for completeness sake, we include the PSNR results in Fig. 8. Similar to Fig. 7, a vertical line with 3 dots are shown for each case in Fig. 8. Again the three dots represent the initial, final and average PSNR and the lower and upper ends of a vertical line represent the minimum

and maximum PSNR respectively. The PSNR tends to exhibit some large fluctuations as the maximum and minimum may be significantly different from the initial and final values. Occasionally, it exhibits some apparently erratic behavior in the large initial patch size cases (16×16 and 8×8), contrary to what we expect, by having maximum PSNR in the initial iteration and minimum PSNR as in the final iteration. Note that, Mobile sequence has relatively lower SSIM and PSNR values, because its dis-occluded region contain something (such as a new cow) that cannot be predicted or guessed using the nearby information.



Fig. 9. Top Row: Typical virtual views of Balloons synthesized from view 3 (from left to right are Views 0,2,4,6). Bottom Row: Corresponding original images.

In Fig. 7 and 8, we observe that the initial patch size of 2×2 always lead to significantly lower initial, final and average SSIM and PSNR, probably because it can be easily trapped in local minima. It appears that small patch size tends to capture and generate small repeatable patterns (with significant high frequency details), but tends not to capture large repeatable patterns. On the other hand, large patch size tend to capture large repeatable patterns, with small details smoothed out. And as the patch size reduces using Schedule 1 or 2, it can capture the texture details (including the small repeatable patterns) in addition. For Poznan_Street, Undo_Dancer, Champagne_tower, Pantomime, and Balloons, the initial patch size of 16×16 and final patch size of 4×4 appear to work slightly better. For Balloons and Mobile, the initial patch size of 8×8 and final patch size of 4×4 seems to be slightly better.

C. Performance of Visto Under Different Baseline Distances

Secondly, we test Visto under different baseline situations for all the test sequences. For each test sequence, we choose the most centered view as the reference view (with view offset = 0) and synthesize 3 virtual views on both the left side (with negative view offset of -3 , -2 and -1) and right side (positive view offset). As both S2 and S7 have fewer than 7 views, we do this experiment only on 5 sequences: S1, S3, S4, S5, S6. For each sequence, we compute the SSIM over the entire common area. In Fig. 9, we show one typical example of the synthesized images and the original images. As pointed by arrows, the musical notes on the background wall of the synthesized images are different from the original ones. However, the overall quality of each synthesized view appears to be visually pleasing especially when the baseline distance is small. Similar trend is observed in the SSIM score (averaged over the 5 sequences) in Fig. 10.

To test the effectiveness of the random initialization in (7), we also test Visto with the random initialization replaced by

$$C_x(i, j) = \begin{cases} -D(i, j) & \text{if } (i, j) \notin \mathbf{P} \\ -D(\tilde{i}, j) & \text{if } (i, j) \in \mathbf{P} \end{cases} \quad (20)$$

which we mark as Visto_RndOff in Fig. 11. Visto is an extension of a preliminary version in [36], which we mark

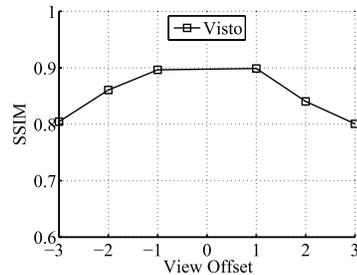


Fig. 10. Average SSIM score over the entire common area of 5 tested sequences for the synthesized views.

as Visto_RndOff_OldP in Fig. 11. Visto_RndOff_OldP is basically Visto_RndOff with the optimization applied to the whole image. Visto_RndOff_OldP also divides the whole image into non-overlapping 2×2 blocks and assumes that all 4 pixels in each block have the same $C_p^{l,r}$. It then computes the $C_p^{l,r}$ only once for each 2×2 block, thus achieving a computation reduction factor of 4. Fig. 11(a) is the Whisker plots of the SSIM (over the entire common area) difference between Visto and Visto_RndOff_OldP for different view offsets. A positive value means Visto is better. Fig. 11(b) is the corresponding Whisker plots of the SSIM (over the entire common area) difference between Visto_RndOff and Visto_RndOff_OldP. A positive value means Visto_RndOff is better. Fig. 11(c) is the corresponding Whisker plots of the SSIM (over the untrusted region only) difference between Visto and Visto_RndOff. A positive value means Visto is better. In the Whisker plots, a box with a top bar and a bottom bar are shown for each synthesized view. In general, it is observed that Visto shows a slightly better performance in terms of the average SSIM than Visto_RndOff and Visto_RndOff_OldP in Fig. 11.

In Fig. 12, we compare the typical differences of their synthesized images in two sequences, *balloons* and *mobile*. The typical differences are marked inside the black circles and white circles in the figure. The two methods, Visto_RndOff_OldP and Visto_RndOff, are showing some ghosting artifacts (inside the black circle) while these artifacts are suppressed in Visto, which demonstrates the effectiveness of the random initialization. In addition, Visto_RndOff_OldP is showing that the integrity of the straight edges of the mobile phone is severely disturbed (inside the white circle) while not in Visto_RndOff and Visto, which suggests that the selected optimization (*i.e.* only optimize the untrusted region) is useful. In general, comparing Visto with Visto_RndOff_OldP and Visto_RndOff, the localized subjective visual quality is improved significantly.

D. Performance Comparison With State-of-the-Art Methods

Thirdly, we compare the proposed Visto with *VSRs_IDfast* used in MPEG 3D experiments [41]. In addition, we also compare with three state-of-the-art image hole filling (inpainting) methods that can be used for view synthesis: *Bertalmio* which

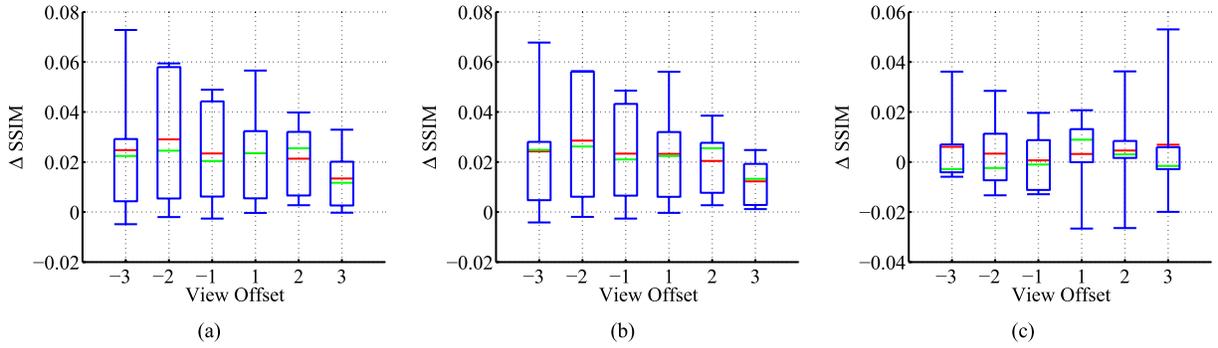


Fig. 11. Whisker plots of SSIM differences among Visto, Visto_RndOff and Visto_RndOff_OldP of 5 tested sequences for the synthesized views. The red and green bars in the box are the mean and median SSIM difference among the 5 sequences; the edges of the box are the 25th and 75th percentile SSIM differences among the 5 sequences; the top bar is the max, and the bottom bar is the min. (a) SSIM difference between Visto and Visto_RndOff_OldP over the entire common area. (b) SSIM difference between Visto_RndOff and Visto_RndOff_OldP over the entire common area. (c) SSIM difference between Visto and Visto_RndOff over the untrusted region.

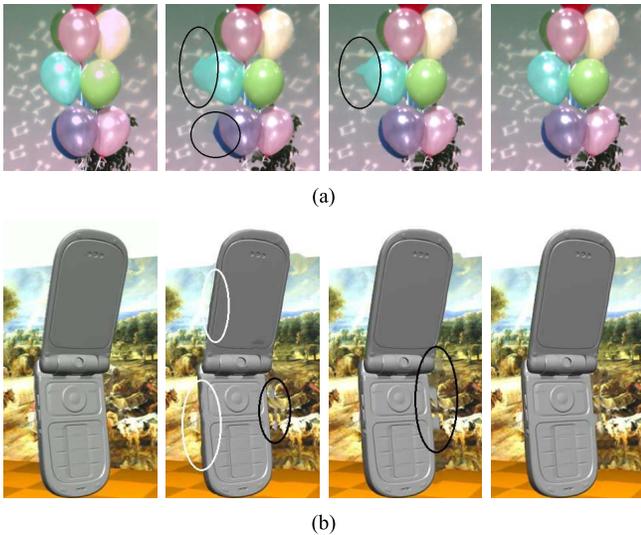


Fig. 12. Typical localized visual differences. 1st column: original image; 2nd column: Visto_RndOff_OldP; 3rd column: Visto_RndOff; 4th column: Visto. (a) Balloons at view 0, synthesized from reference view 3. (b) Mobile at view 5, synthesized from reference view 4.

is PDE-based inpainting [26], *Criminisi* which is exemplar-based inpainting [27], and *Gautier* which is depth-based inpainting [30]. For these hole filling methods, we apply 3D warping which naturally generate holes and then we use these methods to fill the holes. For *Bertalmio*, we use the software publicly available at [42]. For *Criminisi* and *Gautier*, we use the software supplied by the authors. The subjective visual results are shown in Fig. 13. We note that in Fig. 13, Visto produces much more natural results with significantly fewer artifacts, while all other methods have their difficult situations. The objective evaluation over the untrusted region using SSIM is shown in Fig. 14(a). For most of the sequences, Visto has the highest SSIM score which suggests better objective quality. And in terms of average SSIM, Visto is the best among the compared methods. In addition, in Fig. 14(b) we also calculate the mean square error between the gradient magnitudes of the synthesized view X and the original view X^* (G-MSE for

short), which is denoted as

$$\frac{1}{N} \sum_i \left(\sqrt{(\nabla_x X)_i^2 + (\nabla_y X)_i^2} - \sqrt{(\nabla_x X^*)_i^2 + (\nabla_y X^*)_i^2} \right)^2.$$

$\nabla_x X$ and $\nabla_y X$ are the derivatives at the x - and y -dimension that we computed by convolving X with $\nabla_x = [-1, 1]$ and $\nabla_y = [-1, 1]^T$, respectively. For most of the sequences, Visto has the lowest G-MSE score which suggests better objective quality. And in terms of average G-MSE, Visto is significantly better than other compared methods.

V. COMPLEXITY ANALYSIS

When Visto is applied, Algorithm 1 is run for each level (or patch size). For an initial patch size of 16×16 and shrink Schedule 2, Algorithm 1 is run for each of the 4 levels (16×16 , 8×8 , 4×4 and 2×2). In each level, after initialization, Visto goes into a loop for up to R times. In each iteration, there are 3 main steps. In step 1, it applies (12) to each pixel in the untrusted region by performing a brute-force full search within a search window size of $w_x \times w_y$ in the reference image, in order to find the new optimal correspondence map value with minimum energy. For each search point, the cost function (6) is computed for the whole patch. Thus the complexity of step 1 is $O(N_u w_x w_y m n)$, where N_u is the number of pixels in the untrusted region. In step 2, it applies (13) to compute the new texture image pixel in the untrusted region with a complexity of $O(N_u m n)$. Similarly, Visto applies (14) to compute the new depth map value in the untrusted region, with a complexity of $O(N_u m n)$. Therefore, the total complexity is $O(N_u w_x w_y m n)$ for each iteration in the loop in Algorithm 1. Total complexity of Algorithm 1 is $O(R N_u w_x w_y m n)$ for level l .

In our simulation, we choose $m = n$, and $w_x = w_y = \alpha m$. Thus, the complexity is $O(N_u m^4)$. In the worst case, the loop will have R iterations. Thus the worst case complexity is $O(N_u m^4)$. For patch size shrink Schedule 1, the total complexity of Visto is $O(N_u m_0^5)$ because $\sum_{m=2}^{m_0} m^4 \sim O(m_0^5)$, where initial patch size is $m_0 \times m_0$. For Schedule 2, the total complexity of Visto is $O(N_u m_0^4)$ because $\sum_{m=2,4,8,\dots,m_0} m^4 =$

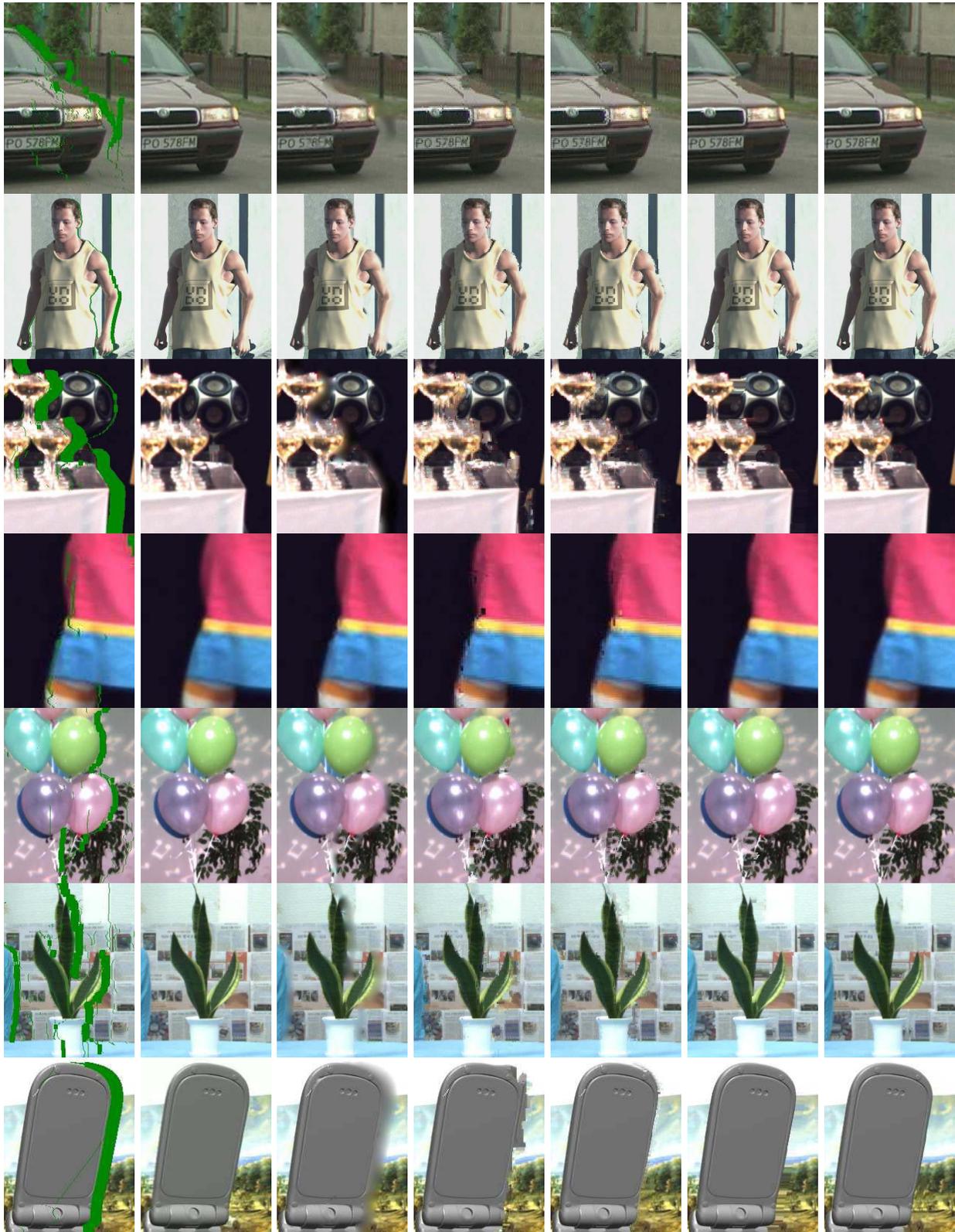


Fig. 13. Synthesized virtual view of tested sequences. 1st column: 3D warped virtual view with holes colored green; 2nd column: Original images in the virtual viewpoint; 3rd column: *Bertalmio*; 4th column: *Criminisi*; 5th column: *Gautier*; 6th column: *VRSID_fast*; 7th column: *Visto*.

$\sum_{i=1}^{\log_2(m_0)} (2^i)^4 \sim O(m_0^4)$. Thus Schedule 2 can be significantly faster than Schedule 1. Using the starting patch sizes mentioned in Section IV, the running time for a single frame of

Visto on our PC (Win64, intel Core i3 CPU at 3.07 GHz, 8GB RAM, code written in C++ without compiler optimization, parallel optimization, or any form of optimization) for both

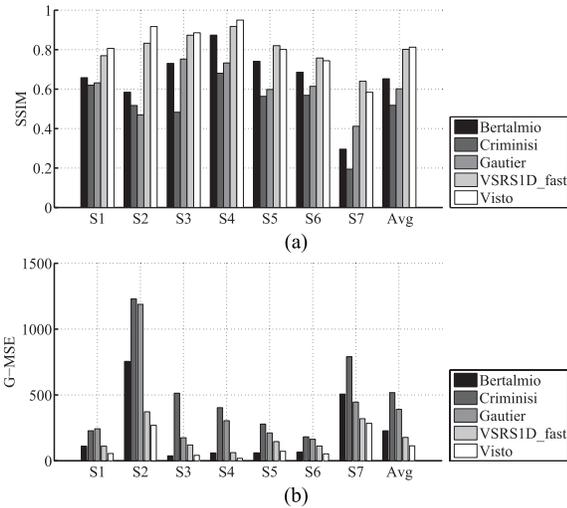


Fig. 14. SSIM (a) and G-MSE (b) of Y component over the untrusted region of the synthesized sequences using different methods. In SSIM (a), the higher score suggests better objective quality. In G-MSE (b), the lower score suggests better objective quality.

TABLE II
RUNNING TIME (SECONDS) OF VISTO WITH DIFFERENT SHRINK
SCHEDULES FOR A SINGLE FRAME

	S1	S2	S3	S4	S5	S6	S7
Schedule 1	245.08	232.70	217.02	104.46	27.61	178.87	12.16
Schedule 2	83.07	64.34	63.15	37.46	13.63	41.46	5.664

Schedules when $\alpha = 2$ for a single frame is shown in Table II. Schedule 2 can be about 3 to 4 times faster than Schedule 1.

VI. CONCLUSION

In this paper, we propose a novel view synthesis method called View Synthesis through Texture Optimization (Visto). By formulating the view synthesis problem in depth-image based rendering (DIBR) as an energy optimization problem, Visto generates iteratively a virtual view (common area only) that maximizes the inter-view texture similarity and minimizes the inter-view depth map error simultaneously. Simulation results suggest that Visto can produce synthesized views that are more natural-looking and perceptually reasonable than some state-of-the-art algorithms.

Visto is designed for 2D (image). Directions of future work include measures to ensure temporal consistency in 3D (video), as independent optimization for each frame can lead to flickering in the synthesized video. Another direction is to ensure view consistency. For two synthesized views at different positions, they can look good individually. However, when viewed in succession, some kind of inconsistency can occur and this needs to be improved in the future. Complexity reduction will also be a good problem to work on.

REFERENCES

- [1] A. Smolic, K. Müller, P. Merkle, C. Fehn, P. Kauff, P. Eisert, *et al.*, "3D video and free viewpoint video-technologies, applications and MPEG standards," in *Proc. IEEE Int. Conf. Multimedia Expo*, Jul. 2006, pp. 2161–2164.
- [2] C. Fehn, "Depth-image-based rendering (DIBR), compression and transmission for a new approach on 3D-TV," *Proc. SPIE*, vol. 5291, pp. 93–104, May 2004.
- [3] A. Smolic, K. Müller, P. Merkle, T. Rein, M. Kautzner, P. Eisert, *et al.*, "Free viewpoint video extraction, representation, coding, and rendering," in *Proc. ICIP*, 2004, pp. 3287–3290.
- [4] A. Smolic, "3D video and free viewpoint video—From capture to display," *Pattern Recognit.*, vol. 44, no. 9, pp. 1958–1968, Sep. 2011.
- [5] H. Murata, Y. Mori, S. Yamashita, A. Maenaka, S. Okada, and K. Ihara, "Device and method for converting two-dimensional video into three-dimensional video," U.S. Patent 1235438, Aug. 28, 2002.
- [6] C. Riechert, F. Zilly, P. Kauff, J. Güther, and R. Schäfer, "Fully automatic stereo-to-multiview conversion in autostereoscopic displays," in *Proc. IBC*, Sep. 2012, pp. 8–14.
- [7] M. Lang, A. Hornung, O. Wang, S. Poulakos, A. Smolic, and M. Gross, "Nonlinear disparity mapping for stereoscopic 3D," *ACM Trans. Graph.*, vol. 29, pp. 1–10, Jul. 2010.
- [8] G. C. Burdea and P. Coiffet, *Virtual Reality Technology*. New York, NY, USA: Wiley, Jun. 2003.
- [9] S. E. Chen, "QuickTime VR—An image-based approach to virtual environment navigation," in *Proc. 22nd Annu. Conf. Comput. Graph. Interact. Tech.*, 1995, pp. 29–38.
- [10] L. Mcmillan and G. Bishop, "Plenoptic modeling: An image-based rendering system," in *Proc. 22nd Annu. Conf. Comput. Graph. Interact. Tech.*, 1995, pp. 39–46.
- [11] S. Chen and L. Williams, "View interpolation for image synthesis," in *Proc. 20th Annu. Conf. Comput. Graph. Interact. Tech.*, 1993, pp. 279–288.
- [12] H.-Y. Shum and S. B. Kang, "A review of image-based rendering techniques," *Proc. SPIE*, vol. 4067, pp. 2–13, May 2000.
- [13] M. Levoy and P. Hanrahan, "Light field rendering," in *Proc. 23rd Annu. Conf. Comput. Graph. Interact. Tech.*, (New York, USA), pp. 31–42, 1996.
- [14] S. Gortler, R. Grzeszczuk, R. Szeliski, and M. Cohen, "The lumigraph," in *Proc. 23rd Annu. Conf. Comput. Graph. Interact. Tech.*, 1996, pp. 43–54.
- [15] J.-X. Chai, X. Tong, S.-C. Chan, and H.-Y. Shum, "Plenoptic sampling," in *Proc. 27th Annu. Conf. Comput. Graph. Interact. Tech.*, Aug. 2000, pp. 307–318.
- [16] S. M. Seitz and C. R. Dyer, "View morphing," in *Proc. 23rd Annu. Conf. Comput. Graph. Interact. Tech.*, 1996, pp. 21–30.
- [17] L. Mcmillan and G. Bishop, "Head-tracked stereoscopic display using image warping," *Proc. SPIE*, vol. 2409, pp. 21–30, Mar. 1995.
- [18] D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," in *Proc. IEEE Workshop SMBV*, Nov. 2001, pp. 1–35.
- [19] P. Kauff, N. Atzpadin, C. Fehn, M. Müller, O. Schreer, A. Smolic, *et al.*, "Depth map creation and image-based rendering for advanced 3DTV services providing interoperability and scalability," *Signal Process., Image Commun.*, vol. 22, no. 2, pp. 217–234, Feb. 2007.
- [20] D. Min, J. Lu, and M. Do, "Depth video enhancement based on weighted mode filtering," *IEEE Trans. Image Process.*, vol. 21, no. 3, pp. 1176–1190, Mar. 2012.
- [21] C. L. 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, Aug. 2004.
- [22] W. Sun, O. C. Au, L. Xu, S. H. Chui, C. W. Kwok, and Y. Li, "Error compensation and reliability based view synthesis," in *Proc. IEEE ICASSP*, May 2011, pp. 1349–1352.
- [23] L. Zhang and W. J. Tam, "Stereoscopic image generation based on depth images for 3D TV," *IEEE Trans. Broadcast.*, vol. 51, no. 2, pp. 191–199, Jun. 2005.
- [24] J. Shade, S. Gortler, L.-W. He, and R. Szeliski, "Layered depth images," in *Proc. 25th Annu. Conf. Comput. Graph. Interact. Tech.*, 1998, pp. 231–242.
- [25] C.-F. Chang, G. Bishop, and A. Lastra, "LDI tree: A hierarchical representation for image-based rendering," in *Proc. 26th Annu. Conf. Comput. Graph. Interact. Tech.*, 1999, pp. 291–298.

- [26] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester, "Image inpainting," in *Proc. SIGGRAPH*, 2000, pp. 417–424.
- [27] A. Criminisi, P. Perez, and K. Toyama, "Region filling and object removal by exemplar-based image inpainting," *IEEE Trans. Image Process.*, vol. 13, no. 9, pp. 1200–1212, Sep. 2004.
- [28] K.-J. Oh, S. Yea, and Y.-S. Ho, "Hole-filling method using depth based in-painting for view synthesis in free viewpoint television (FTV) and 3D video," in *Proc. 27th Conf. PCS*, May 2009, pp. 233–236.
- [29] I. Daribo and B. Pesquet-Popescu, "Depth-aided image inpainting for novel view synthesis," in *Proc. IEEE Int. Workshop MMSP*, Oct. 2010, pp. 167–170.
- [30] J. Gautier, O. Le Meur, and C. Guillemot, "Depth-based image completion for view synthesis," in *Proc. 3DTV-CON*, May 2011, pp. 1–4.
- [31] P. Ndjiki-Nya, M. Köppel, D. Doshkov, H. Lakshman, P. Merkle, K. Müller, *et al.*, "Depth image-based rendering with advanced texture synthesis for 3-D video," *IEEE Trans. Multimedia*, vol. 13, no. 3, pp. 453–465, Jun. 2011.
- [32] M. Schmeing and X. Jiang, "Depth image based rendering: A faithful approach for the disocclusion problem," in *Proc. 3DTV-CON*, Jun. 2010, pp. 1–4.
- [33] A. W. Fitzgibbon, Y. Wexler, and A. Zisserman, "Image-based rendering using image-based priors," *Int. J. Comput. Vis.*, vol. 63, no. 2, pp. 141–151, Feb. 2005.
- [34] S. W. Hasinoff, S. B. Kang, and R. Szeliski, "Boundary matting for view synthesis," *Comput. Vis. Image Understand.*, vol. 103, no. 1, pp. 22–32, Jul. 2006.
- [35] A. Criminisi and A. Blake, "The SPS algorithm: Patching figural continuity and transparency by split-patch search," in *Proc. IEEE CVPR*, Jul. 2004, pp. 342–349.
- [36] W. Sun, O. C. Au, L. Xu, Y. Li, W. Hu, and Z. Yu, "Texture optimization for seamless view synthesis through energy minimization," in *Proc. 20th ACM Int. Conf. Multimedia*, Oct. 2012, pp. 733–736.
- [37] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman, "PatchMatch: A randomized correspondence algorithm for structural image editing," *ACM Trans. Graph.*, vol. 28, no. 3, pp. 24:1–24:11, Jul. 2009.
- [38] "Draft call for proposals on 3D video coding technology," ISO/IEC JTC1/SC29/WG11, N11830, Jan. 2011.
- [39] L. McMillan, "A list-priority rendering algorithm for redisplaying projected surfaces," Univ. North Carolina, Chapel Hill, NC, USA, Tech. Rep. 95-005, 1995.
- [40] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004.
- [41] C. Riechert, F. Zilly, P. Kauff, J. Güther, and R. Schäfer, "Description of 3D video technology proposal by fraunhofer hhi (HEVC compatible configuration B)," ISO/IEC JTC1/SC29/WG11 m22571, Nov. 2011.
- [42] W. Yung and A. J. Shankar. (2013). *Image Inpainting Implementation Software* [Online]. Available: <http://www.bantha.org/~aj/inpainting>



Wenxiu Sun received the B.E. degree in communication engineering from the Department of Electronic Engineering, Nanjing University, Nanjing, China, in 2009. She is currently pursuing the Ph.D. degree with the Department of Electronic and Computer Engineering, Hong Kong University of Science and Technology, Hong Kong.

In 2012, she was an intern student at the National Institute of Informatics, Tokyo, Japan.



Oscar C. Au received the B.A.Sc. degree from the University of Toronto in 1986, and the M.A. and Ph.D. degrees from Princeton University in 1988 and 1991, respectively. After being a post-doctoral at Princeton University, he joined the Hong Kong University of Science and Technology (HKUST) as an Assistant Professor in 1992. He is a Professor of the Department of Electronic and Computer Engineering, the Director of the Multimedia Technology Research Center, and Director of computer engineering in HKUST. His main research contributions are on video/image coding and processing, watermarking/light weight encryption, and speech/audio processing. His research topics include fast motion estimation for H.261/3/4/5, MPEG-1/2/4, and AVS, optimal and fast sub-optimal rate control, mode decision, transcoding, denoising, deinterlacing, post-processing, multi-view coding, view interpolation, depth estimation, 3DTV, scalable video coding, distributed video coding, subpixel rendering, JPEG/JPEG2000, HDR imaging, compressive sensing, halftone image data hiding, GPU-processing, and software-hardware co-design. He has published over 60 technical journal papers, 350 conference papers, and 70 contributions to international standards. His fast motion estimation algorithms were accepted into the ISO/IEC 14496-7 MPEG-4 international video coding standard and the China AVS-M standard. His light-weight encryption and error resilience algorithms are accepted into the China AVS standard. He was a Chair of the Screen Content Coding AdHoc Group in JCTVC for HEVC. He has over 20 granted U.S. patents and is applying for over 70 more on his signal processing techniques. He has performed forensic investigation and stood as an expert witness in Hong Kong courts many times.

He is a fellow of HKIE, and a BoG member of APSIPA. He is an Associate Editor of eight journals: TCSVT, TIP, TCAS1, JVCIR, JSPS, TSIP, JMM, and JFI. He is a Chair of three technical committees: IEEE CAS MSA TC, IEEE SPS MMSP TC, and APSIPA IVM TC. He is a member of six other TCs: IEEE CAS VSPC TC, DSP TC, IEEE SPS IVMSM TC, IFS TC, and IEEE ComSoc MMC TC. He served on two steering committees: IEEE TMM and IEEE ICME. He also served on organizing committee of many conferences including ISCAS 1997, ICASSP 2003, and ISO/IEC 71st MPEG in 2005, ICIP 2010. He was a General Chair of several conferences: PCM 2007, ICME 2010, PV 2010, APSIPA ASC 2015, and ICME 2017. He received the five Best Paper Awards: SiPS 2007, PCM 2007, MMSP 2012, ICIP 2013, and MMSP 2013. He was the IEEE Distinguished Lecturer in 2009 and 2010, APSIPA DL in 2013 and 2014, and has been keynote speaker multiple times.



Lingfeng Xu received the B.E degree in telecommunication engineering from the Department of Electronics and Information Engineering, Huazhong University of Science and Technology, Wuhan, China, in 2009. He is currently pursuing the Ph.D. degree with the Department of Electronic and Computer Engineering, Hong Kong University of Science and Technology, Hong Kong. His current research interests include image/video processing, 3DTV, computer vision, and cloud gaming.



Yujun Li received the B.Sc. degree in computer science and mathematics and the M.Phil. degree in electronic and computer engineering from the Hong Kong University of Science and Technology in 2009 and 2012, respectively. His research interest is stereo matching.



Wei Hu received the B.S. degree from the University of Science and Technology of China, Hefei, China, in 2010. She is currently pursuing the Ph.D. degree from the Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology, Kowloon, Hong Kong.

Her current research interests include image processing on graphs, transform designs, and 3-D video coding and processing. She holds a patent on the compression of piecewise smooth images.