HKUST Institutional Repository

Publication information

Title	Seamless View Synthesis Through Texture Optimization
Author(s)	Sun, Wenxiu; Au, Oscar Chi Lim; Xu, Lingfeng; Li, Yujun; Hu, Wei
Source	IEEE Transactions on Image Processing , v. 23, (1), January 2014, p. 342-355
Version	Pre-Published version
DOI	https://doi.org/10.1109/TIP.2013.2289994
Publisher	IEEE

Copyright information

Copyright © 2014 IEEE. Reprinted from (W. Sun, O. C. Au, L. Xu, Y. Li and W. Hu, "Seamless View Synthesis Through Texture Optimization," in IEEE Transactions on Image Processing, vol. 23, no. 1, pp. 342-355, Jan. 2014.). This material is posted here with permission of the IEEE. Such permission of the IEEE does not in any way imply IEEE endorsement of any of The Hong Kong University of Science and Technology's products or services. Internal or personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution must be obtained from the IEEE by writing to pubs-permissions@ieee.org. By choosing to view this document, you agree to all provisions of the copyright laws protecting it.

Notice

This version is available at HKUST Institutional Repository via http://hdl.handle.net/1783.1/58547
If it is the author's pre-published version, changes introduced as a result of publishing processes such as copy-editing and formatting may not be reflected in this document. For a definitive version of this work, please refer to the published version.

http://repository.ust.hk/ir/

Seamless View Synthesis Through Texture Optimization

Wenxiu Sun*, *Member, IEEE,* Oscar C. Au, *Fellow, IEEE,* Lingfeng Xu, *Member, IEEE* Yujun Li, *Member, IEEE,* Wei Hu, *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 joint optimization of inter-view texture and depth map similarity, a framework that is significantly different from other traditional approaches. The advantage of Visto is that the virtual view tends to implicitly inherit the image characteristics from the reference view without the explicit use of image priors or texture modeling. 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[**?**]. 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 implicity 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, 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, 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

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

results of Visto followed by complexity analysis in Section V. In Section VI, we draw the final conclusion.

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, dis-occlusions (or holes), ringing artifacts, etc. 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 [19] detects 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 [20] performs 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 [21] divides 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 [22] labels the dilated dis-occlusion 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 dis-occlusion. 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 [11] uses Z-buffering and selects always the front-most pixel. However, problems can occur when the depth information is inaccurate or even unavailable. McMillan [10] uses ordered warping based on epipolar geometry and provide an alternative solution that is relatively robust to depth errors. Dis-occlusion 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 dis-occlusion 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 [23] preprocesses 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 [24] uses 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 disoccluded pixels, effectively reducing the hole regions. Chang [25] improves LDI by introducing LDI-tree and considers the sampling rate and the LDI density.

To fill the holes, classical image inpainting methods such as fast inpainting [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, straight-forward 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 et al. [29] and Gautier et

al. [30] extend traditional exemplar-based inpainting to depth-exemplar-based inpainting [27] by considering both depth and texture information. [31][32] fill the dis-occluded regions by considering the temporal consistency in video frames.

In spite of all the above DIBR methods, the overall look of the synthesized images often have unnaturally sharp boundaries. Hasinoff et al. [33] 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 [34] 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.

III. Proposed View synthesis with texture optimization (Visto)

In this section, we describe the proposed novel **Vi**ew **S**ynthesis with **T**exture **O**ptimization, which we call *Visto*. Visto focuses on the bottomline performance requirement: the reference view and the synthesized view should have similar texture and depth. 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.

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. Although the reference view and the virtual view are from different angles or positions, the two views should be similar to each other as they are from the same scene and at the same time. Assuming each local patch in the virtual view could find a correspondence in the reference view, we will define an energy function to describe the similarity between local patches and seek to minimize it iteratively using a Gauss-Seidel like approach. The advantage of this approach is that the virtual view tends to implicitly inherit the image characteristics from the reference view without the explicit use of image priors or texture modeling.

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^t_p, Z^d_p, X^t_p, X^d_p be the corresponding values at p. Let N_p be a rectangular patch around p of size $m \times n$. The local texture of p is captured by N_p. We represent N_p in Z^t, Z^d, X^t, X^d as row-ordered $mn \times 1$ vectors $\mathbf{z}^t_p$, $\mathbf{z}^d_p$, $\mathbf{x}^t_p$, $\mathbf{x}^d_p$, 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.

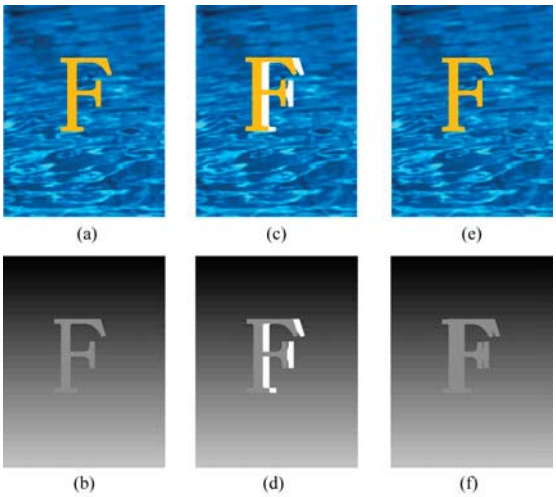


Fig. 1. Comparison of correspondence map and disparity map for rectified views. (a) Reference texture image Z^t. (b) Reference disparity map for Z^t. Brighter intensity means larger disparity. (c) Virtual image X^t after pixel-based 3D warping is applied to (a) and (b). White regions are *dis-occlusion 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.

We now define the *normalized patch energy* $E^t_p(\mathbf{x}^t_p, C_p|Z^t)$ and $E^d_p(\mathbf{x}^d_p, C_p|Z^d)$ 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.

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

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

As can be seen, $E^t_p(\mathbf{x}^t_p, C_p|Z^t)$ and $E^d_p(\mathbf{x}^d_p, C_p|Z^d)$ 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:

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

where $\mathbf{P} = \{p_1, p_2, ..., p_K\}$ 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 virtual view after 3D warping. In this paper, $\mathbf{P}$ is obtained by performing morphological dilation on $\mathbf{P}_0$, and we will call $\mathbf{P}$ the *untrusted region*. In our algorithm, X^t, X^d and C will be allowed to change in $\mathbf{P}$.

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

$$\begin{aligned} \underset{\{X_p^t, X_p^d, C_p : p\in\mathbf{P}\}}{\text{minimize}} \quad & E^t(X^t, C|Z^t) \\ \text{subject to} \quad & E^d(X^d, C|Z^d) \le \varepsilon, \end{aligned} \tag{4}$$

where $E^t(X^t, C|Z^t)$ is the objective function, $E^d(X^d, C|Z^t)$ 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 $E(X, C|Z)$

$$E(X, C|Z) = E^t(X^t, C|Z^t) + \lambda E^d(X^d, C|Z^d) \tag{5}$$

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 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 regions are 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 disoccluded 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.

Let $rnd(a,b)$ be a random number between positive a and b. For each pixel in the untrusted region, we will add a random number to the chosen disparity, avoiding the untrusted regions adhering to its neighboring foreground when viewing in stereo displays. Typically, we choose $a=3, b=10$.

Mathematically,

$$C_x(i,j) = \begin{cases} -D(i,j) & \text{if } (i,j) \notin \mathbf{P} \\ -D(\tilde{i},j) - sgn(D) * rnd(a,b) & \text{if } (i,j) \in \mathbf{P} \end{cases} \quad (6)$$

$$\tilde{i} = \arg\max_{i_l, i_r} \{|D(i_l,j)|, |D(i_r,j)|\} \quad (7)$$

$$i_l = \max\{i' | i' < i, (i',j) \notin \mathbf{P}\} \quad (8)$$

$$i_r = \min\{i' | i' > i, (i',j) \notin \mathbf{P}\} \quad (9)$$

$$C_y(i,j) = 0 \quad (10)$$

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

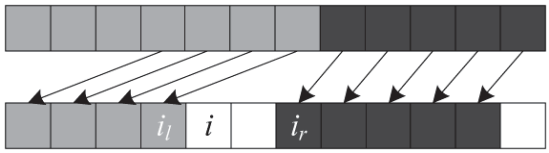


Fig. 2. A line of pixels in the reference view (top line) shift to the virtual view (bottom line) by the amount of their disparity values, with grey pixels being foreground, black pixels being background, white pixels being undefined.

After initializing C, in all image regions $p \in \mathbf{P}^\dagger$, X^t and X^d are easily obtained by applying (12) and (13) 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 (5) 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 $C^{l,0} = C^{l-1,\infty}$.

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 $r = 1 \to R$ **do**
  Compute $C_p^{l,r}$ for all $p \in \mathbf{P}$ using (11);
  Compute $X^{t,l,r}(p)$ for all $p \in \mathbf{P}$ using (12);
  Compute $X^{d,l,r}(p)$ for all $p \in \mathbf{P}$ using (13);
  if (14) is true **then**
    break;
  end if
end for

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

$$\begin{aligned} C_p^{l,r} &= C_p^{l,r-1} \\ &+ \underset{|\triangle C_x| \le w_x, |\triangle C_y| \le w_y}{\operatorname{argmin}} [E_p^t(\mathbf{x}_p^{t,l,r-1}, C_p^{l,r-1} + \triangle C | Z^t) \\ &+ \lambda E_p^d(\mathbf{x}_p^{d,l,r-1}, C_p^{l,r-1} + \triangle C | Z^d)] \end{aligned} \quad (11)$$

where $\triangle C = (\triangle C_x, \triangle C_y)$ 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 (5). Since (5) 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

$$X_p^{t,l,r} = \frac{1}{Q} \sum_{q \in \hat{N}_p} Z_{p+C_q^{l,r}}^t \quad (12)$$

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 $X_p^{l,r}$ is effectively the average of the corresponding Z^t values. In a similar way, we obtain X_p^d for any location p:

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

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

$$\frac{E(X^{t,l,r}, X^{d,l,r}, C^{l,r}|Z) - E(X^{t,l,r-1}, X^{d,l,r-1}, C^{l,r-1}|Z)}{\max\{E(X^{t,l,r-1}, X^{d,l,r-1}, C^{l,r-1}|Z), \theta\}} \le T \quad (14)$$

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.*

$$E(X^{t,l,r}, X^{d,l,r}, C^{l,r}|Z) \leq E(X^{t,l,r-1}, X^{d,l,r-1}, C^{l,r-1}|Z) \tag{15}$$

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 $C_p^{l,r}$,

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

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 $X_p^{d,l,r}$,

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

such that

$$\begin{aligned} E(X^{t,l,r}, X^{d,l,r}, C^{l,r}|Z) &\leq E(X^{t,l,r-1}, X^{d,l,r-1}, C^{l,r}|Z) \\ &\leq E(X^{t,l,r-1}, X^{d,l,r-1}, C^{l,r-1}|Z). \end{aligned} \tag{18}$$

■

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 blurrness 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 to be captured in the patch 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.

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 (12) using $C^{0,0}$.
    Compute $X^{d,0,0}$ from (13) 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

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.

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

Some selected MPEG test sequences ([35]), 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

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.

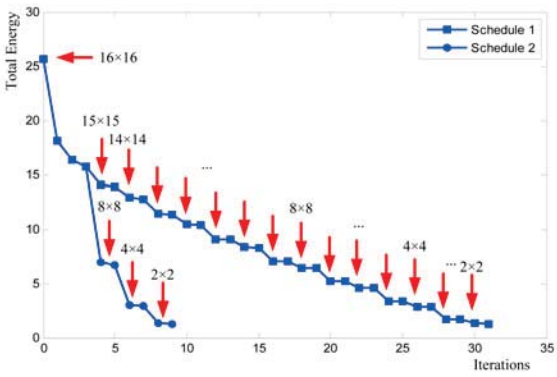


Fig. 4. The decreasing energy in each optimization step by starting at patch size of 16×16 and using two shrink schedules.

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.

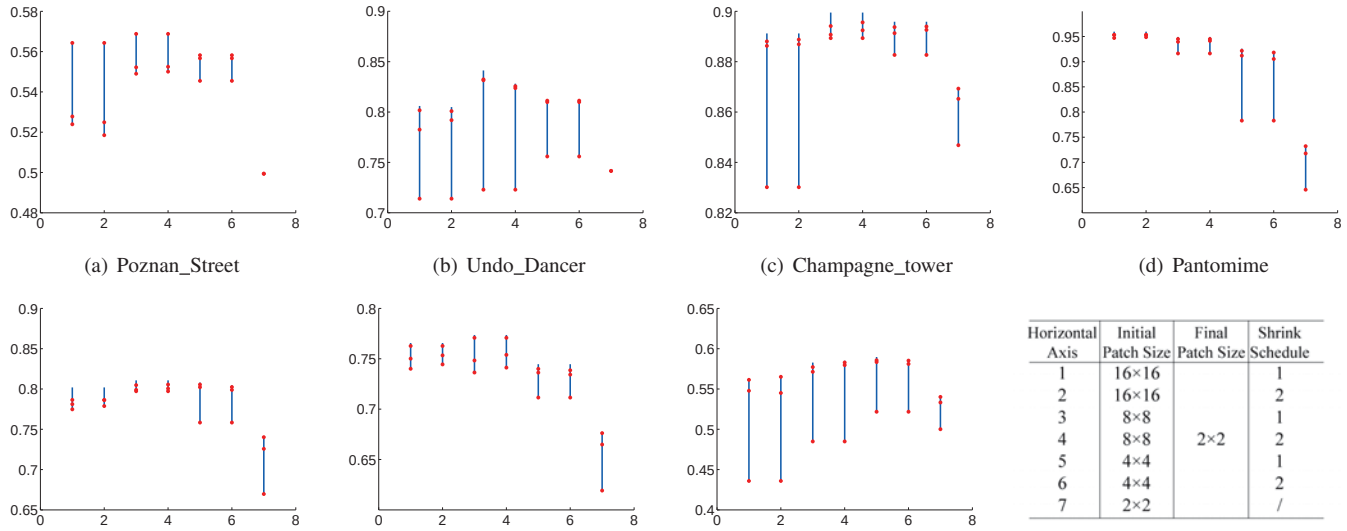
camera space of 5cm. Undo_Dancer and Mobile are synthetic videos with ground truth depth data. The other 5 sequences are natural videos. All views in these test sequences have been rectified. The first frame of each video sequence is used in our simulation. For convenience, these YUV420 texture sequences 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, 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 an adjacent view (e.g. view 4 of Balloons) with a baseline distance of approximately 5cm, corresponding to the distance between the two human eyes. The original and synthesized view pairs shown in Table I are chosen according to the MPEG test conditions.

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 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



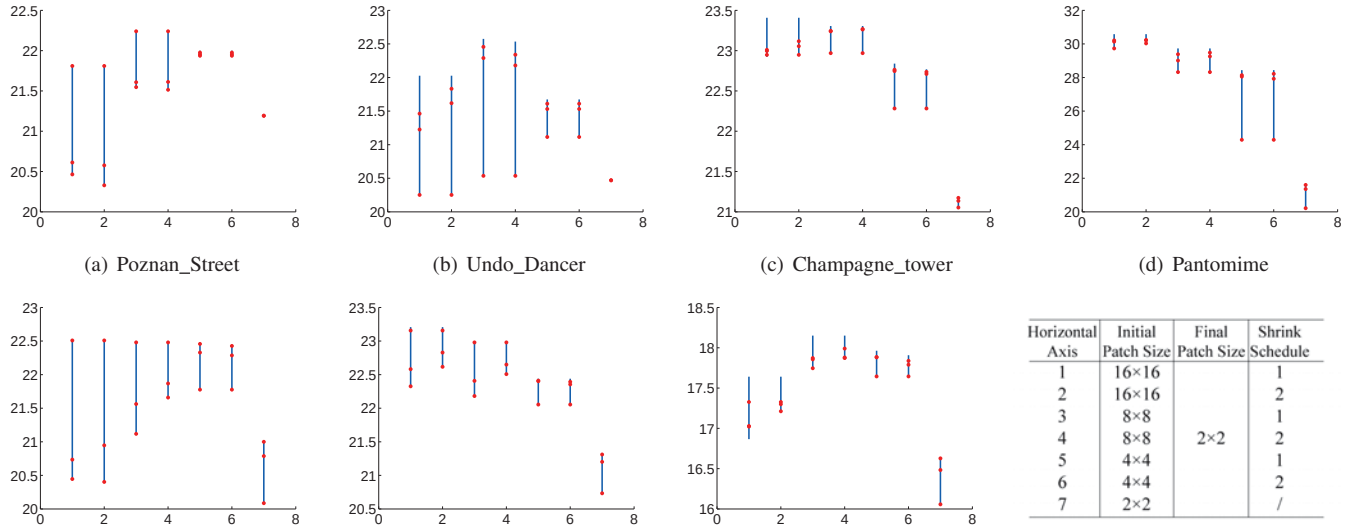
(a) Poznan_Street (b) Undo_Dancer (c) Champagne_tower (d) Pantomime

(e) Balloons (f) Newspaper (g) Mobile

Horizontal Axis	Initial Patch Size	Final Patch Size	Shrink Schedule
1	16×16		1
2	16×16		2
3	8×8		1
4	8×8	2×2	2
5	4×4		1
6	4×4		2
7	2×2		/

(h) Configurations of figure (a-g)

Fig. 6. 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

Horizontal Axis	Initial Patch Size	Final Patch Size	Shrink Schedule
1	16×16		1
2	16×16		2
3	8×8		1
4	8×8	2×2	2
5	4×4		1
6	4×4		2
7	2×2		/

(h) Configurations of figure (a-g)

Fig. 7. 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.

synthesized texture image and depth map using shrink 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 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 [36]. 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 (12) and (13). 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. 8. 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

Fig. 8. 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.

SSIM [37] and PSNR values of the untrusted regions (hole region with dilation) of the synthesized view of the seven test sequences are shown in Fig. 6 and Fig. 7 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. 6, 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. 6(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 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. The final converged synthesized image (at patch size of 2) are shown in Fig. 9 for the four choices of initial patch sizes under Schedule 2. The results for Schedule 1 are similar and thus not shown.

Fig. 9. Synthesized texture image by starting at patch size of 16×16, 8×8, 4×4, 2×2 using shrink schedule 2.

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. 7. Similar to Fig. 6, a vertical line with 3 dots are shown for each case in Fig. 7. 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 disoccluded region contain something (such as a new cow) that cannot be predicted or guessed using the nearby information.

In Fig. 6 and 7, 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.

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 at both the left side (with negative view offset of -3, -2 and -1) and right side (positive view offset). The global SSIM (i.e. SSIM of the entire image) for the test sequences with 7 or more consecutive original views (i.e. S1, S3, S4, S5, S6) are averaged and shown in Fig. 11, and the local SSIM (i.e. SSIM of the untrusted regions) are shown in Fig. 12. Selected untrusted regions of some typical synthesized views (Views 0,2,4,6 for Balloons with View 3 being reference) are compared with the corresponding original views in Fig. 13. To test the effectiveness of the random initialization in Eqn. 6, we also test Visto with the random initialization replaced by

$$C_x(i,j) \quad = \quad \begin{cases} -D(i,j) & \text{if } (i,j) \notin \mathbf{P} \\ -D(\tilde{i},j) & \text{if } (i,j) \in \mathbf{P} \end{cases} \tag{19}$$

which we mark as Visto_RndOff in the figures. Visto is an extension of an early version in [38], which we mark as Visto_RndOff_oldP in Fig. 11. Visto_RndOff_oldP is basically Visto_RndOff with the optimization applied to the whole image. As the computation of $C_p^{l,r}$ is intensive, Visto_RndOff_oldP divides the whole image into non-overlapping 2×2 blocks and assumes that all 4 pixels in each

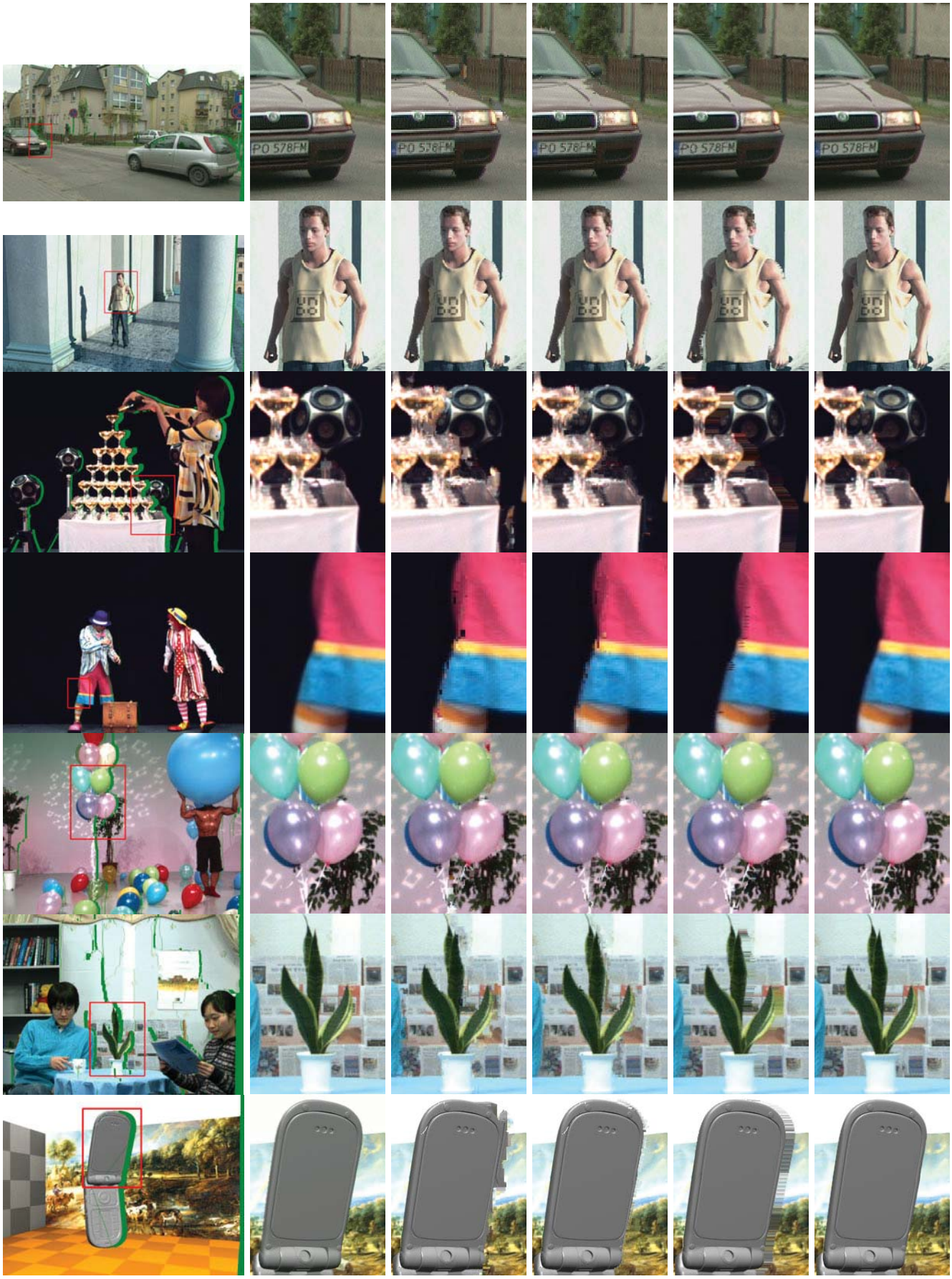


Fig. 10. Synthesized virtual view of tested sequences. 1st column: 3D warped virtual view with holes colored green and red squared regions to be enlarged; 2nd column: Original images in the virtual viewpoint; 3rd column: Criminisi's method (M1); 4th column: Gautier's method (M2); 5th column: VSRS 3.5 (M3); 6th column: Proposed method (Visto).

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.

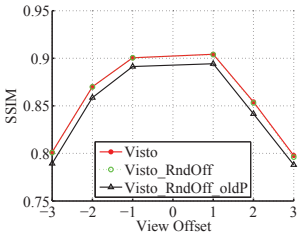


Fig. 11. Average SSIM score of the entire image for selected test sequences at 6 virtual viewpoints.

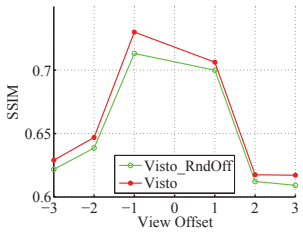


Fig. 12. Average SSIM score at the untrusted regions for selected test sequences at 6 virtual viewpoints.

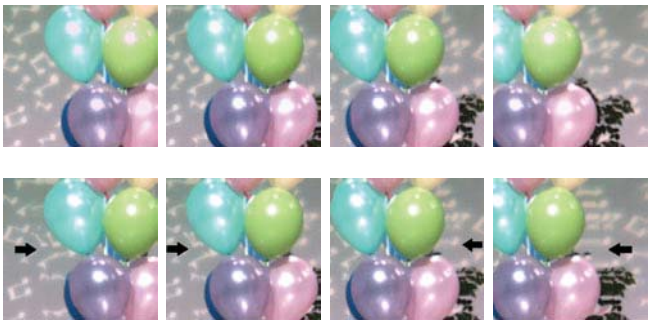


Fig. 13. Bottom row: Typical synthesized virtual views (Views 0,2,4,6 of Balloons synthesized from view 3). Top Row: Corresponding original images.

In general, it is observed that the SSIM of Visto is considerably higher than Visto_RndOff in Fig. 12, which suggests that the random initiation in Eqn. 6 is useful in the untrusted regions. In Fig. 11, Visto and Visto_RndOff are very similar because the trusted region dominates when the entire image is considered, and the two methods are identical in the trusted regions. Fig. 11 suggests that Visto and Visto_RndOff are considerably better than Visto_RndOff_oldP, probably because the texture in the trusted regions is allowed to be changed in Visto_RndOff_oldP which is not reasonable.

When comparing the synthesized images with the original images in Fig. 13, we can observe significant differences in the musical note patterns in the background when the synthesized views are far away (View 0 and View 6). When the synthesized views are near (Views 2 and 4), there are smaller differences in the musical note patterns. These subjective observations agree with the downward trend of the SSIM score in Fig. 11 when the view offset increases in magnitude. But the overall quality of Views 0 and 6 still appear to be reasonable, as expected.

Thirdly, we compare the proposed Visto with the VSRS 3.5 view synthesis software (M3) used in MPEG 3D experiments [39]. The newer VSRS-1Dfast MPEG software [40] is not used because it does not support rendering from a single reference view. In addition, we also compare with two state-of-the-art image hole filling (inpainting) methods that can be used for view synthesis: *Criminisi* (M1) which is based on exemplar-based inpainting [27], and *Gautier* (M2) which is based on depth-based inpainting [30]. For both Criminisi and Gautier hole filling, we apply 3D warping and feed the output to the programs supplied by the authors. The results are shown in Table II and Fig. 10. It is clear that VSRS 3.5, Criminisi and Gautier are not perfect, each with their difficult situations. Visto tends to achieve significantly higher SSIM and PSNR than other methods in Table II, and to produce more natural results with significantly fewer artifacts, as demonstrated in Fig. 10.

TABLE II
SSIM AND PSNR OF Y COMPONENT IN THE UNTRUSTED REGIONS OF SYNTHESIZED SEQUENCES S1-S7 USING DIFFERENT METHODS.

	SSIM				PSNR			
	M1	M2	M3	Visto	M1	M2	M3	Visto
S1	0.5179	0.4683	0.4274	**0.8009**	15.98	15.33	11.92	**21.62**
S2	0.4617	0.4724	0.5124	**0.5501**	19.41	20.18	20.68	**21.51**
S3	0.4850	0.7542	0.7439	**0.8942**	12.69	20.17	21.98	**23.27**
S4	0.6812	0.7287	0.8548	**0.9534**	21.38	22.44	24.88	**30.22**
S5	0.5653	0.6006	**0.8215**	0.8043	17.02	18.91	**22.97**	21.75
S6	0.5735	0.6182	0.7280	**0.7534**	19.1	20.92	21.73	**22.72**
S7	0.1764	0.4105	0.4305	**0.5851**	12.46	16.86	15.19	**17.94**
Avg	0.4944	0.5789	0.6455	**0.7631**	16.86	19.26	19.91	**22.72**

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 (13) 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 (5) is computed for the whole patch. Thus the complexity of step 1 is $O(N_u w_x w_y mn)$, where N_u is the number of pixels in the untrusted region. In step 2, it applies (14) to compute the new texture image pixel in the untrusted region with a complexity of $O(N_u mn)$. Similarly, Visto applies (15) to compute the new depth map value in the untrusted region, with a complexity of $O(N_u mn)$. Therefore, the total complexity is $O(N_u w_x w_y mn)$ for each iteration in the loop in Algorithm 1. Total complexity of Algorithm 1 is $O(RN_u w_x w_y mn)$ 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 $\sum_{m=2,4,8,\ldots,m_0} m^4 = \sum_{i=1}^{log_2(m_0)} (2^i)^4 \sim O(m_0^4)$. Thus Schedule 2 can be significantly faster than Schedule 1. Using the starting patch sizes mentioned in Section IV, the running time of Visto on our PC (Win64, intel Core i3 CPU at 3.07 GHz, 8GB RAM) for both Schedules when $\alpha = 2$ is shown in Table. III. 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).

TABLE III
RUNNING TIME (SECONDS) OF VISTO WITH DIFFERENT SHRINK SCHEDULES

	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

By formulating the view synthesis problem in depth-image based rendering (DIBR) as an energy optimization problem, Visto generates iteratively a virtual view that maximizes the inter-view texture similarity and minimizes the inter-view depth map error simultaneously. Simulation results suggest that Visto can produce seamless (*i.e.* natural-looking and perceptually reasonable) views as compared with other state-of-the-art algorithms.

REFERENCES

[1] A. Smolic, K. Müller, P. Merkle, C. Fehn, P. Kauff, P. Eisert, and T. Wiegand, "3d video and free viewpoint video-technologies, applications and mpeg standards," in *Multimedia and Expo, 2006 IEEE International Conference on*, pp. 2161–2164, 2006.

[2] C. Fehn, "Depth-image-based rendering (dibr), compression and transmission for a new approach on 3d-tv," in *Proceedings of SPIE Stereoscopic Displays and Virtual Reality Systems XI*, pp. 93–104, 2004.

[3] A. Smolic, K. Müller, P. Merkle, T. Rein, M. Kautzner, P. Eisert, and T. Wiegand, "Free viewpoint video extraction, representation, coding, and rendering," pp. 3287–3290, 2004.

[4] A. Smolic, "3d video and free viewpoint video: From capture to display," vol. 44, 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," in *US Patent*, 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*, Sept. 2012.

[7] M. Lang, A. Hornung, O. Wang, S. Poulakos, A. Smolic, and M. Gross, "Nonlinear disparity mapping for stereoscopic 3d," *ACM Transactions on Graphics*, vol. 29, pp. 75:1–75:10, Jul 2010.

[8] G. C. Burdea and P. Coiffet, *Virtual Reality Technology*. Wiley-IEEE Press, June 2003.

[9] S. E. Chen, "Quicktime vr: An image-based approach to virtual environment navigation," in *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques, SIGGRAPH'95*, pp. 29–38, 1995.

[10] L. Mcmillan and G. Bishop, "Plenoptic modeling: An image-based rendering system," in *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques, SIGGRAPH'95*, vol. 95, pp. 39–46, ACM, 1995.

[11] S. Chen and L. Williams, "View interpolation for image synthesis," in *Proceedings of the 20th annual conference on Computer graphics and interactive techniques, SIGGRAPH '93*, pp. 279–288, ACM, 1993.

[12] H.-Y. Shum and S. B. Kang, "A review of image-based rendering techniques," in *Proceedings of SPIE*, pp. 2–13, Spie, 2000.

[13] M. Levoy and P. Hanrahan, "Light field rendering," in *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques - SIGGRAPH '96*, (New York, USA), pp. 31–42, 1996.

[14] S. Gortler, R. Grzeszczuk, R. Szeliski, and M. Cohen, "The lumigraph," in *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques, SIGGRAPH '96*, pp. 43–54, ACM, 1996.

[15] J.-X. Chai, X. Tong, S.-C. Chan, and H.-Y. Shum, "Plenoptic sampling," *Proceedings of the 27th annual conference on Computer graphics and interactive techniques , SIGGRAPH '00*, 2000.

[16] S. M. Seitz and C. R. Dyer, "View morphing," in *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques , SIGGRAPH '96*, (New York, USA), pp. 21–30, 1996.

[17] L. Mcmillan and G. Bishop, "Head-tracked stereoscopic display using image warping," in *Proceedings of SPIE*, vol. 2409, pp. 21–30, 1995.

[18] D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," tech. rep., November 2001.

[19] P. Kauff, N. Atzpadin, C. Fehn, M. Müller, O. Schreer, a. Smolic, and R. Tanger, "Depth map creation and image-based rendering for advanced 3dtv services providing interoperability and scalability," *Signal Processing: Image Communication*, vol. 22, pp. 217–234, Feb. 2007.

[20] D. Min, J. Lu, and M. Do, "Depth video enhancement based on weighted mode filtering," vol. 21, 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, 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 *ICASSP*, pp. 1349–1352, 2011.

[23] L. Zhang and W. J. Tam, "Stereoscopic image generation based on depth images for 3d tv," *Broadcasting, IEEE Transactions on*, vol. 51, no. 2, pp. 191–199, 2005.

[24] J. Shade, S. Gortler, L.-w. He, and R. Szeliski, "Layered depth images," in *Proceedings of the 25th annual conference on Computer graphics and interactive techniques, SIGGRAPH98*, pp. 231–242, ACM, 1998.

[25] C.-F. Chang, G. Bishop, and A. Lastra, "Ldi tree: A hierarchical representation for image-based rendering," in *Proceedings of the 26th annual conference on Computer graphics and interactive techniques, SIGGRAPH '99*, pp. 291–298, 1999.

[26] A. Telea, "An image inpainting technique based on the fast marching method," *Journal of Graphics Tools*, vol. 9, no. 1, pp. 23–34, 2004.

[27] A. Criminisi, P. Perez, and K. Toyama, "Region filling and object removal by exemplar-based image inpainting," vol. 13, 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," *Proc. of the 27th conference on Picture Coding Symposium (PCS'09)*, pp. 233–236, 2009.

[29] I. Daribo and B. Pesquet-Popescu, "Depth-aided image inpainting for novel view synthesis," in *Multimedia Signal Processing (MMSP), IEEE International Workshop on*, pp. 167 –170, Oct. 2010.

[30] J. Gautier, O. Le Meur, and C. Guillemot, "Depth-based image completion for view synthesis," in *3DTV Conference: The True Vision - Capture, Transmission and Display of 3D Video (3DTV-CON)*, pp. 1–4, May 2011.

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

[32] M. Schmeing and X. Jiang in *3DTV-Conference: The True Vision - Capture, Transmission and Display of 3D Video (3DTV-CON)*, June 2010.

[33] S. W. Hasinoff, S. B. Kang, and R. Szeliski, "Boundary matting for view synthesis," *Computer Vision and Image Understanding*, vol. 103, pp. 22–32, Jul 2006.

[34] A. Criminisi and A. Blake, "The sps algorithm: Patching figural continuity and transparency by split-patch search," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR'04*, 2004.

[35] "Draft call for proposals on 3d video coding technology," *ISO/IEC JTC1/SC29/WG11, Doc. N11830*, January 2011.

[36] L. McMillan, "A list-priority rendering algorithm for redisplaying projected surfaces," tech. rep., University of North Carolina at Chapel Hill, Chapel Hill, NC, USA, 1995.

[37] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612.

[38] W. Sun, O. C. Au, L. Xu, Y. Li, W. Hu, and Z. Yu, "Texture optimization for seamless view synthesis through energy minimization," in *Proceedings of the 20th ACM international conference on Multimedia*, (Nara, Japan), pp. 733–736, ACM, Oct. 2012.

[39] M. Tanimoto, T. Fujii, K. Suzuki, N. Fukushima, and Y. Mori, "Reference softwares for depth estimation and view synthesis," *ISO/IEC JTC1/SC29/WG11, Doc. M15377*, April 2008.

[40] 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 Doc.m22571*, Nov. 2011.