Watermarking of Free-view Video

Alper Koz, Cevahir Cigla, and A. Aydın Alatan, *Member, IEEE*

***Abstract*—With the advances in image based rendering (IBR) in recent years, generation of a realistic arbitrary view of a scene from a number of original views has become cheaper and faster. One of the main applications of this progress has emerged as *free-view TV*(FTV), where TV-viewers select freely the viewing position and angle via IBR on the transmitted multiview video. Noting that the TV-viewer might record a *personal* video for this arbitrarily selected view and misuse this content, it is apparent that copyright and copy protection problems also exist and should be solved for FTV. In this paper, we focus on this newly emerged problem by proposing a watermarking method for free-view video. The watermark is embedded into every frame of multiple views by exploiting the spatial masking properties of the human visual system. Assuming that the position and rotation of the virtual camera is known, the proposed method extracts the watermark successfully from an arbitrarily generated virtual image. In order to extend the method for the case of an unknown virtual camera position and rotation, the transformations on the watermark pattern due to image based rendering operations are analyzed. Based upon this analysis, camera position and homography estimation methods are proposed for the virtual camera. The encouraging simulation results promise not only a novel method, but also a new direction for watermarking research.**

***Index Terms*—Free-view television, image based rendering, light field rendering, multiview video, watermarking, 3-D watermarking.**

I. Introduction

In the last decade, the utilization of 3-D information for modelling and representation of a real world scene has become widespread in many applications, such as animation films, video games and stereoscopic displays. Parallel to these applications, the research on more sophisticated technologies based upon rendering of 3-D scenes, such as free-view televisions (FTVs) and 3-D holographic televisions, has already reached to an acceptable maturity [1]. In addition, standardization on multiview coding has been delivered by ISO and ITU bodies, and FTV is expected to be the next goal for standardization [2]–[4]. In this aspect, it can be concluded that spread of these new technologies in the near future will bring a serious necessity for the protection of the 3-D information for the content owners.

Similar to its image and video watermarking counterparts, 3-D watermarking is proposed as a promising solution for the copyright problem of 3-D information by means of embedding the watermark into the main representation of a 3-D content and extracting the embedded watermark after the content is used in an application. Previous work on 3-D watermarking [5] can be classified into three groups as *3-D-3-D*, *3-D-2-D* and *2-D-2-D watermarking*. In the first group [6]–[8], the watermark is embedded to 3-D geometric structure of an object residing in a scene and tried to be extracted from the 3-D geometry after any attacks on the geometry. The protection of 3-D models in computer graphics applications forms the major objective of this category. The second group [9], [10], *3-D-2-D watermarking*, aims to extract the watermark that is originally hidden into the 3-D object from the resulting images or videos, which are obtained after projection of 3-D object onto 2-D image planes. Such requirement is certainly applicable for computer animation, computer gaming, and virtual and augmented realities. The watermark can be both embedded into the geometry [9] or the texture [10] of the object in this category.

In this taxonomy, the final category, *2-D-2-D watermarking* [11], [12], aims to protect the image-based representation of a 3-D scene. While the first two groups try to protect the intellectual property rights for the two important components of a traditional representation of a 3-D scene, geometry and texture, the third group approaches to this problem by watermarking sequences of images, which represent the 2-D projections of the same 3-D scene, and by extracting the watermark from any 2-D rendered image, generated for an arbitrary angle of the scene via these sequences.

While the previous two scenarios have been studied in a good extent in the literature [5]–[10], this last category of 3-D watermarking has recently emerged after the progresses in image based rendering (IBR) techniques [13], which render novel views directly from input images, mostly with no utilization of any explicit information related to the 3-D scene structure. The advances in IBR technology have resulted with much easier and faster generation of a realistic arbitrary view of a scene from a number of original views. Consequently, this progress has yielded a new technology, namely *free-viewpoint television* [14], [15], in which TV-viewers select freely the viewing position by the application of IBR algorithms on the transmitted multiview video.

A major application of *2-D-2-D watermarking* is being expected in the copyright protection of multiview video content in this new technology. Noting that the TV-viewer might also record a *personal* video for the arbitrarily selected view and misuse this content, it is apparent that copyright and copy

Manuscript received April 28, 2009; revised January 09, 2010. First published March 08, 2010; current version published June 16, 2010.This work was supported by the EC 6th Framework IST 3DTV NoE Project. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Min Wu.

A. Koz is with the Multimedia Signal Processing Group, Faculty of Electrical Engineering, Mathematics and Computer Science, Delft University of Technology, 2628 Delft, The Netherlands (e-mail: A.Koz@tudelft.nl).

C. Cigla and A. A. Alatan are with the Department of Electrical and Electronics Engineering, METU, Balgat, 06531, Ankara, Turkey (e-mail: cevahir@eee.metu.edu.tr; alatan@eee.metu.edu.tr).

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.2010.2045024

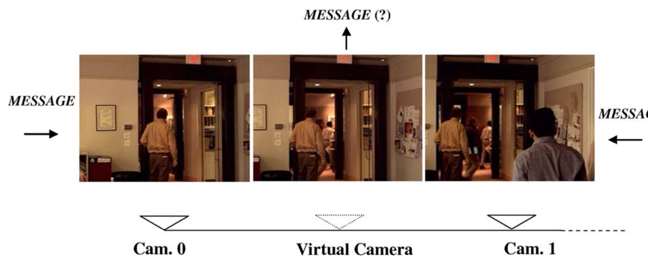


Fig. 1. Watermarking problem for FTV.

protection problems should also exist for FTV. In a possible scenario, the owner of the multiview content should prove his/her ownership, not only on the original views of the multiview video, but also on any *virtual* view, which is generated by the viewer using IBR from the original views.

As in previous copyright problems for image and video data, the copyright problem for multiview video can also be treated by means of watermarking. However, there are more challenging requirements, compared to well-studied mono-view video watermarking. In this paper, we first introduce the general framework and essential differences in this new watermarking problem, and then propose a watermarking method for multiview video.

A. General Framework and Basic Requirements

First of all, beside the robustness to common video processing and multiview video processing operations, the main characteristic challenge for multiview watermarking is to detect the embedded signal from a virtual video sequence, generated for an arbitrary (user selected) view (Fig. 1). Based upon the availability of the original and/or watermarked multiview content, the utilized data during watermark detection can be as follows:

1) a single watermarked virtual view generated from watermarked multiview video;
2) a single watermarked virtual view, and the watermarked multiview video from which the virtual view is generated;
3) a single watermarked virtual view generated from watermarked multiview video, and original (unwatermarked) multiview video.

The first case, watermark detection from a single virtual view, is the most difficult scenario. Due to the rendering operations, the watermark signal residing inside a virtual view is a combination of watermark components coming from different views of watermarked multiview video. Therefore, it is uncertain for the detector to determine the transformations that has occurred on the original watermark pattern by analyzing only a single virtual view, without the knowledge of watermarked multiview video. This case is not considered further in this manuscript.

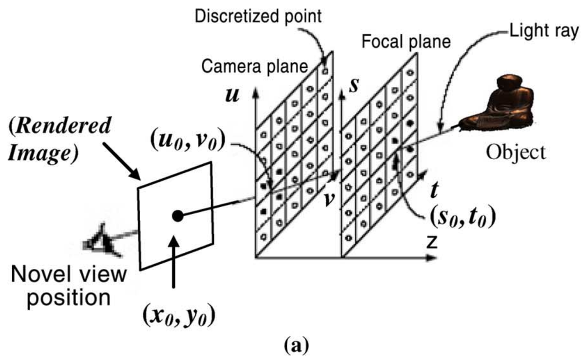
The second and third cases are equivalent to the blind and nonblind scenarios in video watermarking. This paper focuses on more challenging blind scenario where only the watermarked multiview video is available during the watermark detection from a single virtual view. In this case, the watermark detection scheme should also involve an estimation procedure for the virtual camera position and orientation, for which the rendered view is generated. In addition, the watermark should also survive from image-based rendering operations, such as frame interpolation between neighbor cameras and pixel interpolation inside each camera frame. The weights of the neighbor cameras and the pixels in these interpolations are strictly dependent upon the position and rotation of the virtual camera. This makes the problem different than the 1-D interpolations for monoview video.

The second challenge for multiview video watermarking is collusion type of attacks which is more difficult to cope with. In video watermarking, a number of watermarked frames can be used to generate a new frame that the watermark may not be detectable, if different watermarks are embedded to similar frames [16]. Similarly, in multiview case, a number of views containing different watermarks can be used to generate a view that the watermark may not be detectable. Furthermore, embedding different watermarks to different views of the same scene makes also tracking of watermark components in the rendered view quite difficult during watermark detection.

Finally, multiview watermarking extends the imperceptibility requirement for the video case. A watermark for single-view video should be spatially invisible in each frame and temporally invisible in the temporal dimension of the video. Moreover, in multiview video watermarking, the watermark should be embedded to the adjacent original camera frames such that it does not yield any temporal distortions in the resulting virtual video during rendering.

In the literature, there are mainly two rendering approaches for virtual view generation, namely *depth-based rendering* and *image-based rendering*. Depth-based rendering techniques [17] also utilize the geometry information about the scene during rendering, whereas image-based rendering [13] only uses the images taken from different views of the scene with very slight information on the geometry during view generation. From beginning to the end, IBR techniques has evolved according to the scene structure and characteristics. In its initial stage, static scenes that involve only one depth layer or one object has formed the main focus of the research [18]. Then, it is tailored for the scenes consisting of multiple objects, and dynamic scenes [15], [19]. These latter cases are in general achieved by means of proper segmentation of the scene into depth layers with respect to the object positions and then, by individually applying the rendering methods for each object with respect to its depth layer. Therefore, the rendering methods for one object forms the most fundamental stage in free-view video based upon IBR.

In this research on free-view watermarking, we have focused on this fundamental case, i.e., static scenes consisting of one object. The extension of the problem to the scenes with multiple objects requires proper segmentation of the objects in a scene before watermarking operations, which is beyond the scope of this paper. However, the requirements and general framework in the paper are valid for both of the rendering approaches and different scene structures. The proposed watermark method is specially tailored to protect the virtual views of a single object which are generated by *light field rendering* (LFR) [18], which is one of the competing IBR technology for FTV systems [15] due to its low production costs and simple hardware implementation.



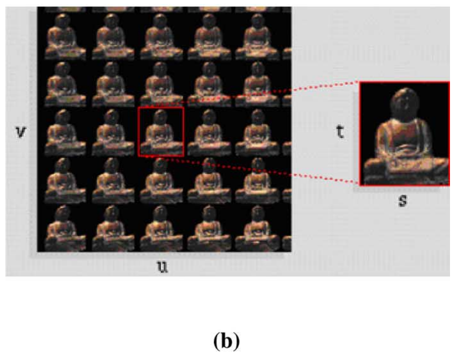


Fig. 2. (a) General configuration for LFR [13] and (b) sample light field image array: *Buddha* light field [18].

The rest of this paper is organized as follows. Section II gives a brief summary of *light field rendering* [15]. The main interpolation methods in LFR, namely, *nearest neighborhood interpolation* and *bilinear interpolation*, are also summarized in this section. Next, the details of the proposed watermarking method are given in Section III. The robustness results for watermark detection from an arbitrarily generated view are presented in Section IV. The detection scheme assumes that the position and orientation of the virtual camera is given *a priori* in this approach. In order to extend the method for the case of an unknown virtual camera position and orientation, the transformations on the watermark pattern due to *nearest neighborhood interpolation* and *bilinear interpolation* in LFR are examined in Sections V and VI, respectively. Based upon this analysis, the camera localization and homography estimation methods are proposed and the robustness results for the unknown virtual camera position and orientation are presented. Finally, the paper is concluded in the last section.

II. Light Field Rendering (LFR)

LFR is a simple method for generating novel views from arbitrary camera angles by combining and resampling the available images without utilizing any depth information about the scene [18]. The basic idea behind the technique is a representation of the *light field* as the radiance at a point in a given direction in free space which is free of occluders. This representation characterizes the flow of light through free space in a static scene with fixed illumination [18].

A practical way to create (capture) light fields is to assemble a collection of images by a number of cameras where the intensity of the pixels of the camera views corresponds to the radiance of the light rays passing through the pixel locations and camera centers. A general configuration for LFR is given in Fig. 2(a) with two parallel planes, namely *camera (uv) plane* and *focal (st) plane*.

Camera plane can be interpreted as the plane where the cameras are located. *Focal plane* is parallel to the camera plane, which is utilized for parameterization of the light field. A light ray is parameterized with four variables, (u_o, v_o, s_o, t_o), where (u_o, v_o) and (s_o, t_o) are the intersections of the light ray with *camera* and *focal* planes, respectively. These planes are usually discretized, so that a finite number of light rays can be recorded [Fig. 2(a)]. If all the discretized points from the focal plane are connected to a single point on the camera plane, an image (2-D array of light fields) is resulted [13]. The only issue in this process is the proper alignment between st samples of the resulted image and pixel coordinates of the camera located at that point on the camera plane. This can be easily achieved by performing a sheared perspective projection [13]. If the same procedure is considered for all the points on the camera plane, a 2-D image array is obtained, as it is shown in Fig. 2(b). Hence, 4–D representation (i.e., (u, v, s, t)) of the light field can also be interpreted as a 2-D image array.

While generating the virtual view of the object for an arbitrary viewing position, the light ray for each pixel of the rendered image is intersected with *camera* and *focal* planes [Fig. 2(a)]. Then, intensity of the pixel is calculated by linearly interpolating existing neighbor light rays in the image array. For example, the intensity of a pixel (x_o, y_o), corresponding to the light ray (u_o, v_o, s_o, t_o) in Fig. 2(a), is interpolated from the intensity of the light rays connecting the solid discrete points on the two planes [13].

Based upon the utilized neighbor light rays during interpolation, there are two major interpolation methods in LFR: *nearest neighborhood interpolation* and *bilinear interpolation* [18]. In the nearest neighborhood interpolation-based LFR, the intensity value of each pixel for the virtual view is generated from the pixel intensities, belonging only to the nearest camera. On the other hand, in bilinear interpolation-based approach, the intensity value of each rendered pixel is obtained, as a weighted sum of pixels belonging to a number of neighboring cameras (see Fig. 3). While the nearest interpolation is simpler to implement, it usually yields more distortion (discontinuities) in the rendered video, especially in the regions whose pixels are rendered by different (adjacent) cameras. On the other hand, bilinear interpolation gives more natural and subjectively pleasant outputs as a result of relatively complex rendering operation [18].

III. Proposed Watermarking Method

In the upcoming FTV scenario, the broadcast stations is expected to deliver their multiview content to the set-top boxes of their viewers and the viewers might interact with their viewing angles by implicitly using the rendering algorithms that process the transmitted multiviews in their set-top boxes. Hence, a watermarking algorithm should be capable of detecting a watermark, which might be transmitted within the multiview content, from any rendered view for proving ownership or copy protection, in case of illegal usage. Unfortunately, it is not possible to

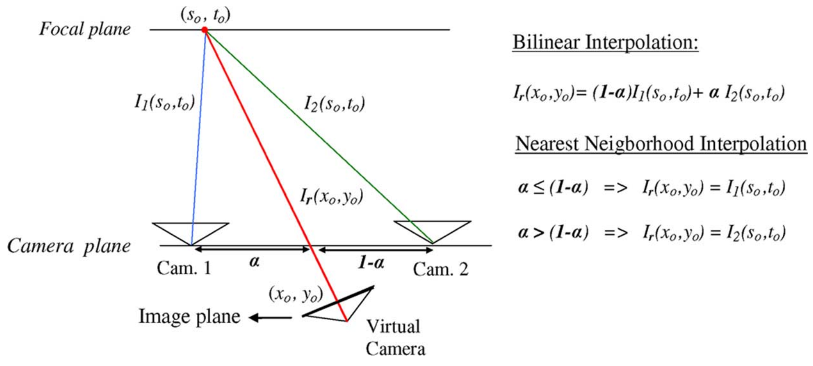

Focal plane
(s_o, t_o)
$I_1(s_o,t_o)$
$I_2(s_o,t_o)$
$I_r(x_o,y_o)$
Camera plane
Cam. 1
α
$1-\alpha$
Cam. 2
(x_o, y_o)
Image plane
Virtual Camera
Bilinear Interpolation:
$I_r(x_o,y_o)= (1-\alpha)I_1(s_o,t_o)+ \alpha\, I_2(s_o,t_o)$
Nearest Neigborhood Interpolation
$\alpha \leq (1-\alpha)$ => $I_r(x_o,y_o) = I_1(s_o,t_o)$
$\alpha > (1-\alpha)$ => $I_r(x_o,y_o) = I_2(s_o,t_o)$

Fig. 3. Interpolation methods in LFR: nearest neighbor versus bilinear interpolation.

detect the trace of a watermark in a rendered image by using the conventional watermarking algorithms in the literature, hence, a novel approach is required.

A. Watermark Embedding

The proposed approach inserts the watermark into each image of the light field image array [Fig. 2(b)] by exploiting spatial sensitivity of human visual system (HVS) [20]. For that purpose, the watermark is modulated with the resulting output image which is obtained after filtering each light field image by a high-pass filter, and spatially added onto the light field image. Such a process decreases the watermark strength over the flat regions of the image, in which HVS is more sensitive, whereas increases the embedded watermark energy in the textured regions, where HVS is relatively insensitive [20].

Two important points should be noted at the embedding stage. First of all, the watermark is embedded to the transmitted light-field images, which are the *sheared perspective projection* of the original camera-captured frames [18]. These frames can be obtained easily by camera calibration information [18]. Secondly, the watermark component, added to the intensity of each image pixel, is determined according to the intersection of the light ray corresponding to that pixel with the focal plane. The same watermark is added to the pixels of different camera views, whose corresponding light rays are intersected at the same point in the focal plane, as illustrated in Fig. 4. The rationale behind such a procedure is to avoid facing with the superposition of the different watermark samples from different camera frames in the interpolation step during the rendering. Otherwise, such a superposition severely degrades the performance of the correlation operation during the watermark detection stage. Moreover, such an approach should also avoid the flickering type of distortion during rendering, since the watermark component in any rendered view will be identical to the watermark components in the adjacent original views.

The embedding method is shown with the relation

$$I^*_{uv}(s,t) = I_{uv}(s,t) + \alpha.H_{uv}(s,t).W(s,t) \tag{1}$$

to each light field image where I_{uv} is the light field image corresponding to the camera at the (u,v) position on the camera plane, H_{uv} is the output image after high pass filtering, α is the global scaling factor to adjust the watermark strength, W is the watermark sequence generated from a Gaussian distribution

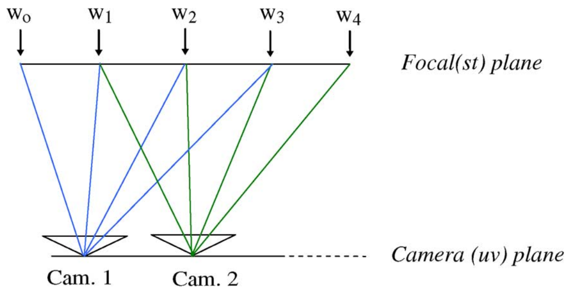


Fig. 4. Watermark embedding methodology.

with zero mean and unit variance, and finally, I^*_{uv} is the watermarked light field image. It should be noted that (s,t) domain indicates the focal plane at which the watermark is embedded.

B. Watermark Detection

The transmitted watermarked camera views, which are utilized during the generation of arbitrary view in FTV, are available during detection. The detector should also have the original watermark pattern embedded to the original camera views. The well-known correlation-based detection scheme is utilized during watermark extraction. Assuming that the position and orientation of the virtual view are known a *priori* (not available in practice and a novel algorithm is proposed to solve this problem in Section V), the first step is applying the same rendering operations during the generation of an arbitrary view to the watermark pattern, W, in order to generate a "rendered watermark", denoted as W_{ren}. After the arbitrarily selected view, (I_{ren}), is filtered by a high-pass filter, the normalized correlation between the resulting image $(\hat{I}_{ren})$ and rendered watermark is determined as

$$\frac{\langle \hat{I}_{ren}, W_{ren}\rangle}{|\hat{I}_{ren}||W_{ren}|}. \tag{2}$$

In the next step, the normalized correlation is compared to a threshold for the detection of the watermark. The overall structure of the watermark detection is shown in Fig. 5.

IV. Experimental Results for Given Camera Positions

Two common light fields, *Buddha* [21] and *Teapot* [22], are used during the simulations. These light fields obtained from 3-D graphical objects provide an easy implementation of the

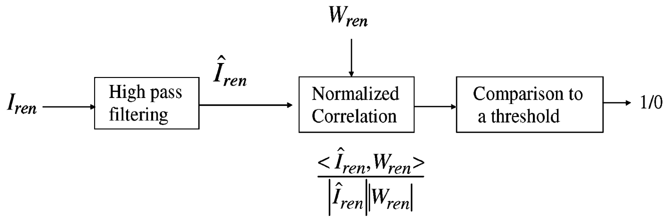

W_{ren}
$\hat{I}_{ren}$
I_{ren}
High pass filtering
Normalized Correlation
Comparison to a threshold
1/0
$\frac{<\hat{I}_{ren}, W_{ren}>}{|\hat{I}_{ren}||W_{ren}|}$

Fig. 5. Overall structure of watermark detection process.

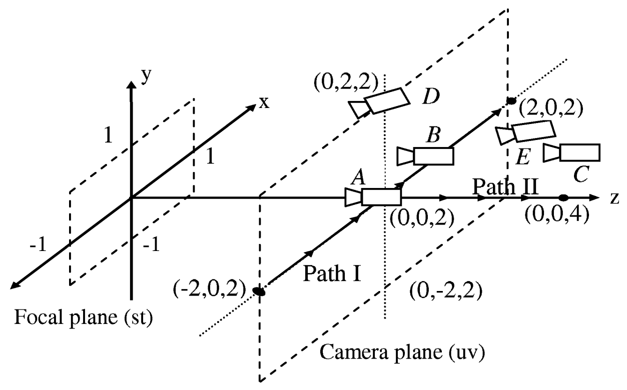


Fig. 6. Location of camera & focal plane for *Buddha* light field [21].

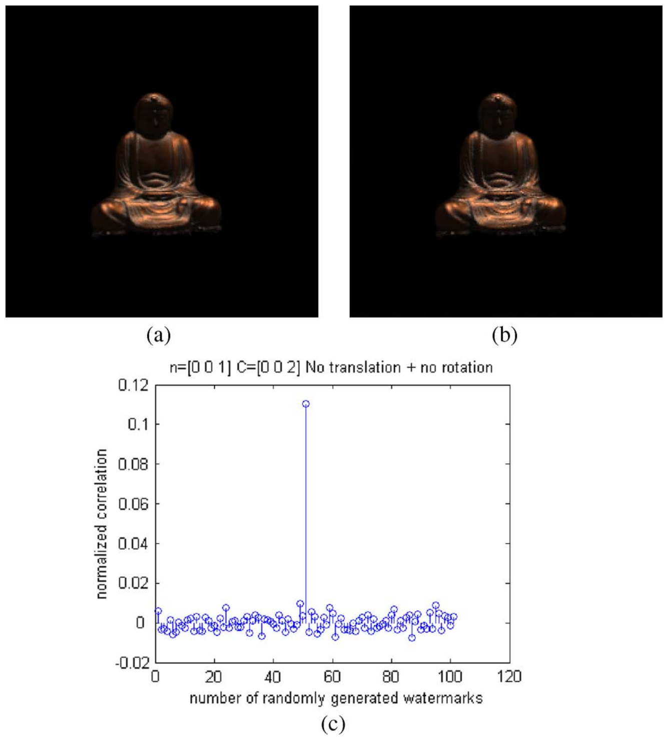


Fig. 7. (a) Rendered view, (b) watermarked view, and (c) normalized correlation result.

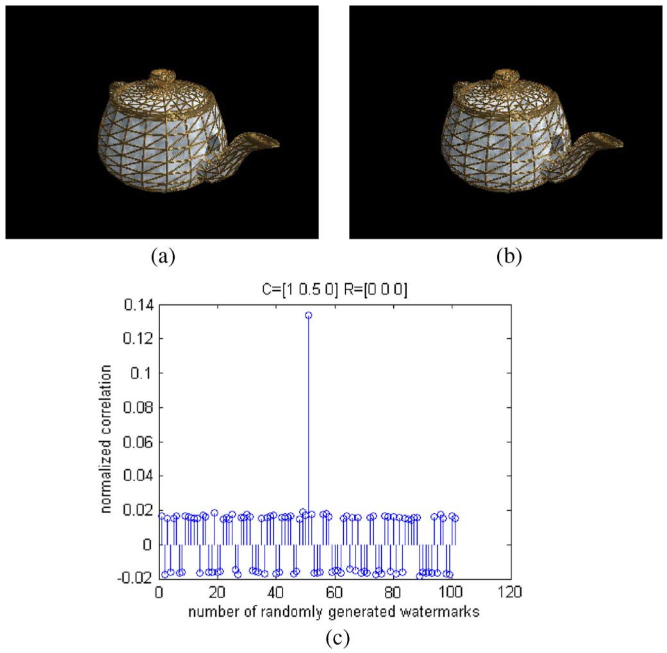


Fig. 8. (a) Rendered view, (b) watermarked view, and (c) correlation result.

rendering schemes to check the performance of watermarking scheme for the virtual views generated on arbitrary positions.[1] A *Laplacian* high pass filter of size 3×3 is used during filtering operations. The global scale factor, α, is adjusted to 0.5. The parameterization of the focal and camera plane for *Buddha* light field [21] is shown in Fig. 6. (The calibration parameters for *Teapot* light field is given in [22].)

A. Imperceptibility Tests

In the imperceptibility tests, the visual equivalence of the original and watermarked rendered image is checked at the first step. Typical rendered views for the original and watermarked *Buddha* light field are presented in Fig. 7. Virtual camera is located at $[0, 0, 2]$ with the normal direction of $[0, 0, -1]$. Another example is given in Fig. 8 for *Teapot* light field. Based upon subjective assessment, there is no visible difference between the original and watermarked rendered views.

In the second step of the imperceptibility tests, the visual equivalence of the original and watermarked rendered video is also tested. The video is obtained by sequencing the light field images according to the trajectory (from $[-2, 0, 2]$ to [2, 0, 2]) on the camera plane in Fig. 6. Since the watermark patterns embedded to adjacent frames can yield visible distortions during rendering, these tests should also be included in the experiments. The rendered video sequences from the original and watermarked Buddha light field for the illustrated trajectory in Fig. 6 are given in http://www.eee.metu.edu.tr/~alatan/PaperData/. Based upon subjective assessment, there is no visible difference between the watermarked and original rendered video sequences.

[1]It should be noted that the important point for a watermarking scheme for free-view video is to achieve robustness against the rendering operations during the generation of a virtual view, rather than type of the input object (i.e., real or synthetic) and the capturing process of light fields.

In order to show the positive effect of embedding the same watermark component into the pixels of different camera frames whose corresponding light rays are intersected at the same point in the focal plane (see Fig. 4), two more video sequences are also generated along the same trajectory in Fig. 6. The first video is generated from the watermarked light field by the proposed method where the *same* watermark is embedded to each light field image. However, for the second video, *different* watermark

TABLE I
NORMALIZED CORRELATION FOR THE EMBEDDED WATERMARK AND MAX. OF THE NORMALIZED CORRELATIONS FOR 100 OTHER WATERMARKS IN *BUDDHA* LIGHT FIELD

Normalized Correlation Values (Eq. 2)	**Case II**	**Case III**	**Case IV**	**Case V**
Embedded Watermark	0.125	0.085	0.075	0.11
Max. Other Watermark	0.010	0.009	0.010	0.009

TABLE II
NORMALIZED CORRELATION RESULTS FOR *TEAPOT* LIGHT FIELD. (THE COMPONENTS OF THE VECTORS FOR THE CAMERA ROTATION ARE THE ANGLES BETWEEN THE IMAGE PLANE NORMAL AND X, Y, Z CARTESIAN COORDINATES, RESPECTIVELY)

Camera Position	**[2.7, 2.7, 0]**	**[1, 0.5, 0]**	**[1.3, 0, 0.5]**	**[1, 0.5, 0.5]**
Camera Rotation	**[0, 0, 0]**	**[0, 0, 0]**	**[0, 0, π/8]**	**[0, 0, 0]**
Embedded Watermark Corr.	0.085	0.134	0.056	0.035
2nd Best Watermark Corr.	0.017	0.019	0.018	0.019

components are added into the pixels of different camera frames whose corresponding light rays are intersected at the same point in the focal plane. Although, there is no flickering type of distortion in the first video, a flicker is observable in the second video, which shows the validity of the proposed embedding strategy. It should be noted that the watermark strength is increased from 0.5 to 0.9 for both video sequences in order to easily view the flickering in the second video (The video sequences are available in the aforementioned URL link).

B. Robustness Tests for Rendering

During the robustness tests, the detection scheme is applied for different imagery views based upon the virtual camera position and orientation. In *Buddha* light field, the camera position of [0, 0, 2] and normal direction of $[0, 0, -1]$ (Position-A in Fig. 6) are taken as reference, in order to describe translation and rotation in the results. The following cases are considered during the simulations:

Case I: No translation in uv-plane and z-axis. No rotation. Camera position $= [0, 0, 2]$; Image plane normal $= [0, 0, -1]$ (Position-A in Fig. 6); focal length $= 2$ for all cases (default).

Case II: No translation in z-axis. No rotation. Only translation in uv-plane. Camera position $= [0.5, 0, 2]$; Image plane normal $= [0, 0, -1]$ (Position-B in Fig. 6).

Case III: Translation in uv plane and z-axis. No rotation. Camera position $= [0.5\ 03]$; Image plane normal $= [0, 0, -1]$ (Position-C).

Case IV: Translation on uv-plane + Rotation. No translation in the z-axis. Camera position $= [0, 2, 2]$; Image plane normal $= [0, -1, -1]$ (Position-D).

Case V: Translation on uv-plane and z-axis + Rotation. Camera position $= [1.5, 0, 2.5]$; Image plane normal $= [0, -1, -1]$ (Position-E).

The normalized correlation values in Case I for the embedded watermark and other 100 randomly generated watermarks from a Gaussian distribution of zero mean and unit variance are shown in Fig. 7(c). The embedded watermark is placed at the 50th (center) position. Its higher value with respect to the other correlation values shows that the watermark is detected successfully. The correlation results for the other cases are shown in Table I. The results for *Teapot* light field, for different virtual camera positions and orientations are given in Fig. 8 and Table II.

C. Robustness Tests Against Other Attacks

Although the main characteristic attack for free-view video is rendering operations during virtual view generation, we also evaluate the robustness of the method for typical processing that could occur in the transmission chain of FTV. These attacks might include additive Gaussian noise (due to A/D and D/A conversions), compression and occlusion, in particular, for the scenes consisting of multiple objects. For the compression attack, all the light field images are passed from JPEG compression, which are expected to give similar results with the upcoming multiview coding standards based upon DCT transformation [3], [4]. For occlusion, rather than focusing on the segmentation and reconstruction of a scene with multiple objects [23], we handle the consequence of occlusion, which turns out as cropping in 2-D rendered views, assuming that the proper segmentation of the objects in the scene are achieved.

For these tests, we consider two paths (I and II) shown in Fig. 6. In the first path, the virtual camera moves on the camera plane, while it is always directed toward the origin ([0 0 0]). This case covers the translation and rotation of the virtual camera, whereas in the second case, the camera moves through the z-axes (path II in Fig. 6). This case corresponds to scaling type of processing for the rendered views.

Fig. 9 gives the normalized correlations for the embedded watermark and the maximum of normalized correlations among 100 other watermarks, in different positions on the first and second path, respectively, against the aforementioned attacks. The proposed method detects the watermark up to additive noise of 29 dB, JPEG compression of 40%, and cropping of 80% for the first path. For the same attack levels, the watermark is successfully detected until $z \cong 4$ in the second path, which corresponds to a scaling of about 20% compared to the camera plane $(z = 2)$. We also indicate the extreme levels of the attacks that

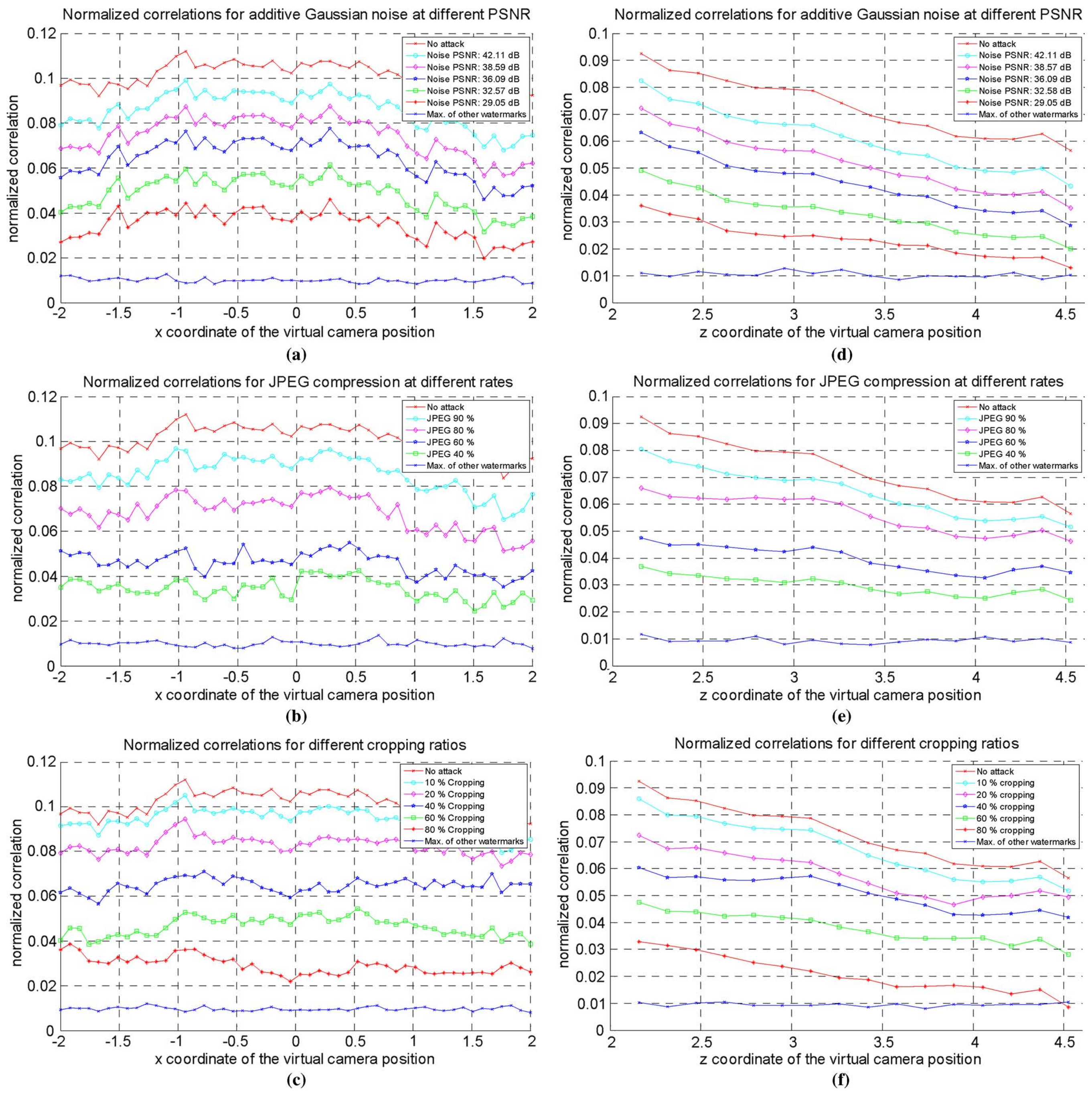


Fig. 9. Normalized correlations at different levels of attacks on *Buddha*. Additive Gaussian noise for (a) path I and (d) path II. JPEG compression for (b) path I and (e) path II. Cropping for (c) path I and (f) path II.

TABLE III
NORMALIZED CORRELATION RESULTS FOR THE EXTREME CASES THAT THE WATERMARK IS STILL DETECTABLE FOR *TEAPOT* LIGHT FIELD. VIRTUAL CAMERA IS ON THE CAMERA PLANE (AT [1 0.5 0] WITH A ROTATION [0 0 0])

Normalized Correlation	Noise PSNR 29.07 dB	JPEG 20 %	CROP 80 %
Embedded watermark	0.118	0.060	0.067
Max. of other 100 watermarks	0.012	0.010	0.013

the watermark can still survive for another sequence, *Teapot*, in Tables III and IV. The robustness results indicate that the proposed method is quite successful against typical processing in a possible FTV application.

TABLE IV
NORMALIZED CORRELATION RESULTS FOR THE EXTREME FOR *TEAPOT* LIGHT FIELD. VIRTUAL CAMERA IS MOVED FROM CAMERA PLANE (AT [1 0.5 3] WITH ROTATION [0 0 0]. CORRESPONDS TO ~50% SCALING WITH RESPECT TO CAMERA PLANE)

Normalized Correlation	**Noise PSNR 29.07 dB**	**JPEG 20 %**	**CROP 80 %**
Embedded watermark	0.019	0.020	0.016
Max. of other 100 watermarks	0.008	0.008	0.011

V. Analysis of LFR on Watermark Pattern for an Unknown Camera

In the previous tests, the position and orientation of the virtual camera is assumed to be known during the detection stage. However, this information is not available in practice. Hence, the detection algorithm should also include a procedure to determine the position and orientation of the virtual camera.

In order to handle this problem, the relation between the embedded watermark and the rendered watermark should be analyzed separately for this two interpolation methods in LFR, namely *nearest neighbor* and *bilinear interpolation*. In this part, the analysis for nearest neighbor interpolation and its robustness results are given; later, the proposed solution for bilinear case will also be presented.

Considering the virtual camera position and orientation, the problem for the nearest neighbor interpolation can be handled in three cases:

Case 1) The virtual camera is located in the camera plane and its rotation matrix is a unit matrix. This configuration is shown in Fig. 10. The same watermark, W, is embedded to the light field images for Camera-1 and 2. From the analysis of the operations in LFR, the watermark corresponding to the imagery camera, W_{ren}, is simply a *shifted version* of W.

Case 2) The virtual camera is again located in the camera plane and its rotation is not equal to unity (see Fig. 10). For this case, the relation between W_{ren} and W is a *planar projective transformation* (homography) [24] (see [22] for a simple verification). Planar projective transformation gives the relation between the pixel coordinates of two cameras viewing a planar surface as a matrix multiplication in homogeneous coordinates as illustrated in Fig. 11.

Case 3) The virtual camera is in an arbitrary position and orientation (Fig. 10), as the most general case. The relation between W_{ren} and W is still a planar projective transformation. However, the projective transformation is different for various regions of the virtual view based upon the original images used during the generation of the virtual view. Hence, this case is the most difficult one for watermark detection.

A. Proposed Solution for Case 1

In order to solve the problem for the first case, the correlation is computed for the possible shifts of the watermark pattern in the detector. The computation of this operation is decreased by utilizing a fast Fourier transform (FFT) and its

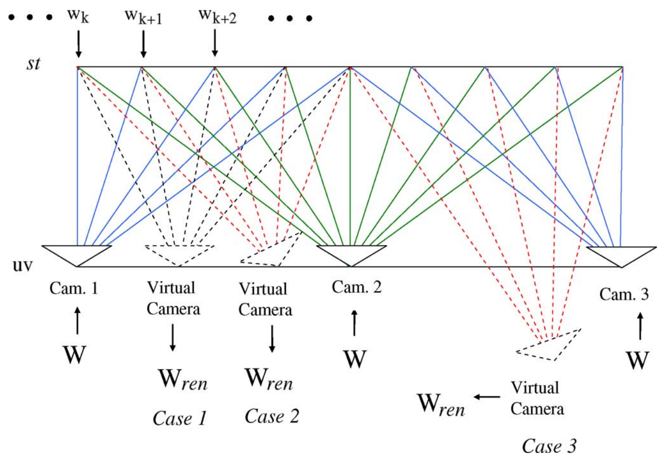


Fig. 10. Configurations for the imagery camera position and rotation for Case 1, 2, and 3.

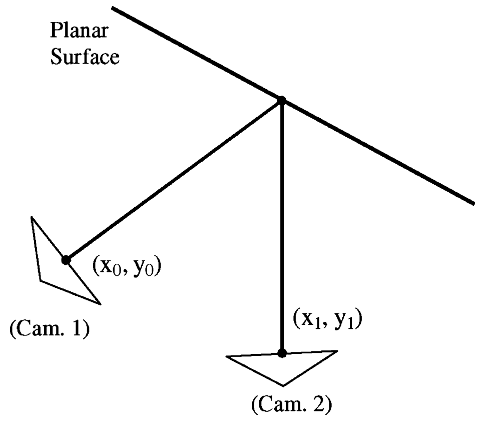


Fig. 11. Planar projective transformation between the (homogeneous) camera coordinates is characterized with a 3 × 3 matrix, H, where $[x_1\ y_1\ 1]^T = H[x_0\ y_0\ 1]^T$ [24].

inverse (IDFT). Specifically, *symmetrical phase only matched filtering* (SPOMF) is used for correlation computations [26]. The position of the virtual camera is changed from the actual case and the normalized correlation is computed for this shifted camera position during these tests. The robustness results in Fig. 12 show the correlations for the normal and SPOMF type of detection, when the camera position is slightly different than the actual case. While the SPOMF still detects the watermark, normal detection algorithm fails.

B. Proposed Solution for Case 2

In Case 2, the rendered image is generated from the image corresponding to the nearest neighbor camera. The relation between the rendered and the nearest neighbor original image can be approximated as a planar projective transformation [22].

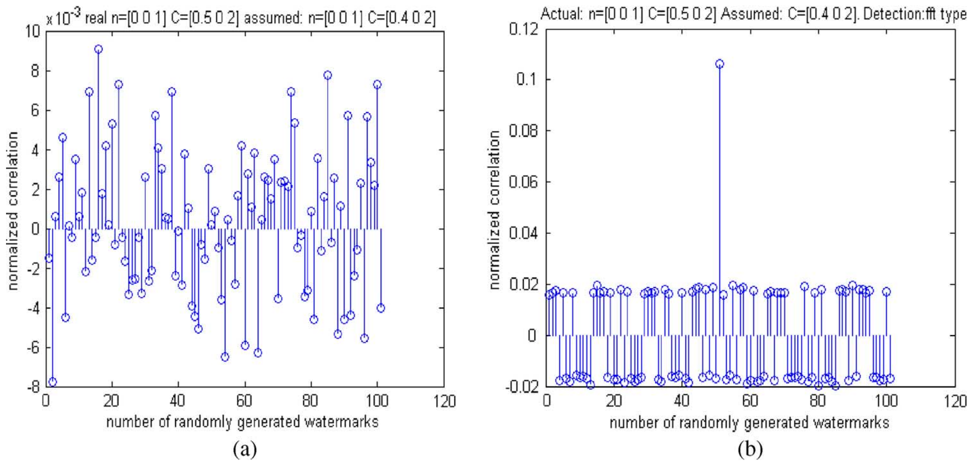


Fig. 12. Normalized correlation graphs for the (a) normal and (b) SPOMF detection. During the test, actual camera position $= [0.5, 0.0, 2.0]$ and assumed camera position $= [0.4, 0.0, 2.0]$.

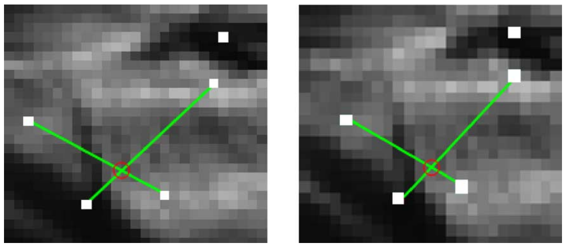


Fig. 13. Illustration of cross-line intersection of four matched feature points on the rendered image and one of original light field image.

TABLE V
AVERAGE MSE BETWEEN THE RENDERED IMAGE AND THE ORIGINAL IMAGES IN *BUDDHA* LIGHT FIELD. (VIRTUAL CAMERA POSITION: [0.5, 0.0, 3.0]; IMAGE PLANE NORMAL: $[0, 0, -1]$)

Image No.	**Average *MSE***	**Image No.**	**Average *MSE***	**Image No.**	**Average *MSE***
22.15	*23.20*	22.16	*22.45*	22.17	*30.61*
23.15	*23.08*	**23.16**	***0.09***	23.17	*36.41*
24.15	*33.89*	24.16	*19.71*	24.17	*40.91*

In this case, there are mainly two stages in the solution. First, the nearest neighbor camera to the virtual camera should be determined among all other cameras. Second, the planar projective transformation between the rendered and the nearest neighbor original image should be estimated. Finally, the planar projective transformation could be applied to the original watermark to obtain the rendered watermark.

The proposed novel solution for finding the nearest neighbor image to the rendered image utilizes a fundamental property originating from planar projective mapping. If four points are matched between two images, one of which is a planar projective transformation of other, then the intersections of cross lines of these four matches is another correspondence between two images [24]. As the relation between the coordinates of rendered and nearest neighbor original image is such a transformation [24], the intersections of cross lines of corresponding four points on these images should be approximately at the same location (see Fig. 13). However, they should be at different locations for the other images. This property can be used to determine the original image from which the rendered image is generated.

The procedure for obtaining the nearest neighbor light field image is as follows:

1) Find feature points that belong to the same scene point in the rendered image and original images. Match some feature points in the rendered image and one of the original images [27].
2) Randomly select four matched points and find intersection point of cross lines (the crosses in Fig. 13) in both of the images.
3) Find the mean square error (MSE) between the surrounding regions of intersection points (3×3 neighborhood of the intersection points are utilized for MSE computation).
4) Repeat step 2 and 3 for N times and obtain best *percentage-p* results. Record the average of MSE.
5) Repeat the procedure for each of the original images. The minimum MSE average corresponds to the nearest neighbor image to the rendered image.

During simulations, Harris corner detector [28] is utilized for feature extraction and N is taken as 2000, while p is selected as 60%. Table V gives the average of MSE at the intersection points between the rendered image and the neighbor original images. The average MSE is lowest for the *buddha.23.16* [22], from which the rendered image is generated, as expected. ("*buddha.u.v*" corresponds to the light field image at row u and column v on the camera plane. Please see Fig. 2).

After determining the nearest neighbor original image to the rendered image, the planar projective transformation is estimated between the two images by using the well-known RANSAC algorithm [29]. The resulting homography is applied to the embedded watermark pattern to obtain the rendered watermark, W_{ren}. Then, the SPOMF type of detection is utilized

TABLE VI
NORMALIZED CORRELATIONS FOR THE EMBEDDED WATERMARK AND MAX. OF THE NORMALIZED CORRELATIONS FOR 100 OTHER WATERMARKS

Camera Center	**[1, 0, 2]**	**[0, -1, 2]**	**[0, 0, 2]**
Image plane normal	**[-0.5, 0, -1]**	**[0, 0.5, -1]**	**[0, 0.5, -2]**
Embedded Watermark	0.078	0.088	0.121
Max. for the others	0.023	0.020	0.020

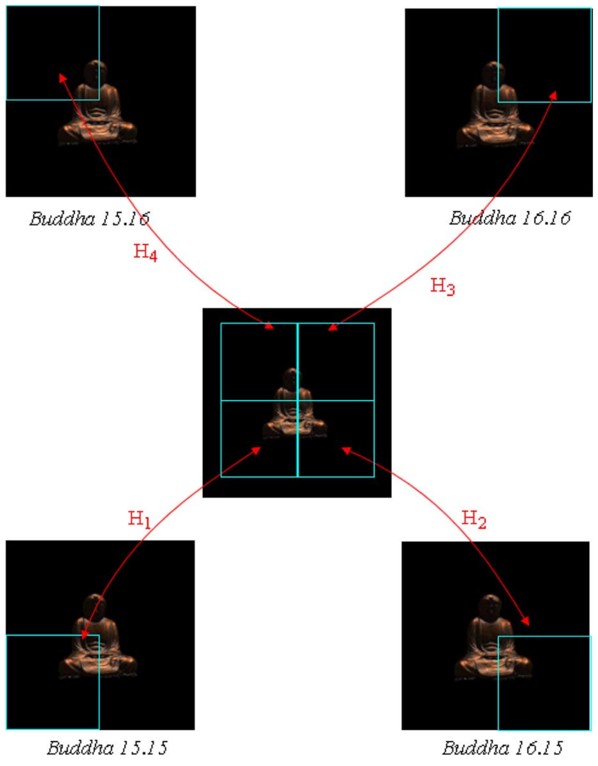


Fig. 14. Relation between the regions of the rendered image and the original light field images. H_1, H_2, H_3, and H_4 correspond to the homographies between the illustrated regions.

to extract the watermark from the rendered image. The normalized correlation values are given in Table VI for different virtual camera positions and rotations. The normalized correlation for the embedded watermark is clearly higher than the maximum of the other correlation values which shows that the method clearly detects the watermark for the mentioned cases.

C. *Proposed Solution for Case 3*

In this case, the rendered image is composed of different regions, where the light fields for each region are generated from a different light field image corresponding to a different camera. As an example, the rendered image, which is generated for a virtual camera located at [0, 0, 2.4] with [0, 0, 1] orientation, is shown in Fig. 14 for *Buddha* light field. This rendered image is composed of four main regions each of which is generated from the light field images, *buddha.15.15*, *buddha.16.15*, *buddha.15.16* and *buddha.16.16*. The relation between each region of rendered image and the corresponding regions in the light field images is a different planar projective transformation (Fig. 14). This relation can be used for the watermark detection in this scaled case. Since the same watermark is embedded to

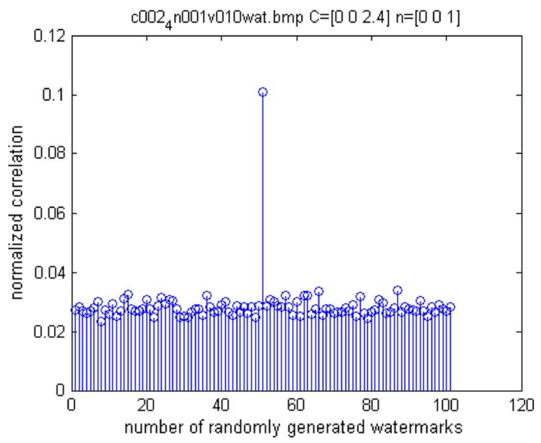


Fig. 15. Normalized correlations results [Eq. (2)]. Camera position $=[0, 0, 2.4]$. Image normal $=[0, 0, 1]$.

each light field image, the relation between the illustrated regions (in Fig. 14) of the rendered watermark, W_{ren}, and the regions of the original watermark, W, is also a homography.

In the detection process, first of all, the rendered image should be partitioned into the regions according to the original light field images from which the pixels in each region are generated. Similar to Case 2, the intensity values at intersections of four feature points should be quite similar to each other at the corresponding regions in the rendered image and its originating light field image. Based upon this observation, a partition algorithm is proposed in [25], which finds the original light field image correctly for more than 90% of pixels in the rendered image.

After such a partition of the rendered image, the homographies between the regions of the rendered image and corresponding regions of the light field images are estimated by means of utilizing the scale invariant feature points [30]. Then, the estimated homography relations are applied to the corresponding regions in the original watermark, W, to generate the rendered watermark, W_{ren}. Finally, the normalized correlation between the rendered watermark and the rendered image that has passed from high-pass filtering is computed.

The normalized correlations for the original watermark and randomly generated 100 watermark patterns are shown in Fig. 15 for the rendered image shown in Fig. 14. The rendered image corresponds to approximately 70–80% scaled version of the original light field images. For this case, the watermark is successfully detected. Another example is shown in Fig. 16. In this case, some amount of rotation is also included at the virtual camera orientation, while the watermark is still detected.

VI. PROPOSED SOLUTION FOR BILINEAR INTERPOLATION IN LFR

The other interpolation method widely used during LFR is bilinear interpolation. In this case, intensity of each pixel of the rendered image is calculated by the weighted sum of light rays corresponding to the four nearest neighbor original images. The weights of the cameras are determined according to the distance of the camera centers to the intersection of the corresponding light ray of the rendered image with the camera plane (Fig. 3). Since each pixel in the rendered image is equal to the weighted

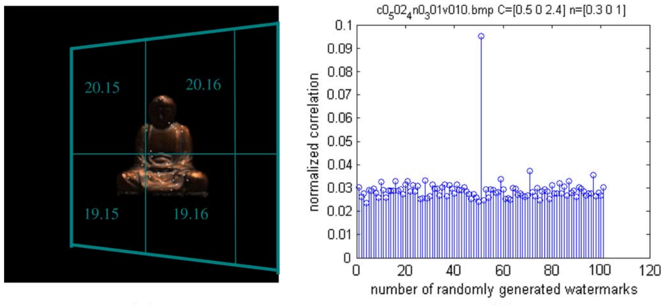


Fig. 16. (a) Rendered image for the virtual camera at location [0.5, 0, 2.4] with orientation [0.3, 0, 1]. The numbers indicate the original image from which corresponding region is generated. (b) Normalized correlations results for the rendered image in (a).

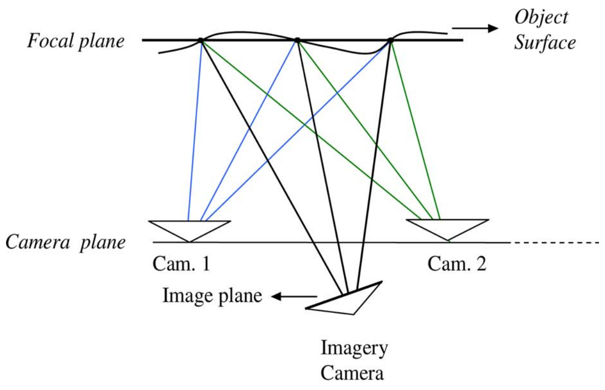


Fig. 17. Illustration for the points of the 3-D object surface located on the focal plane.

sum of the pixels from different cameras, it is not possible to define the relation between the rendered image and the original light field images as a projective planar transformation. Therefore, a different strategy should be followed for this case.

The proposed solution is based upon the fact that the relation between the watermarks in the rendered image and the original images is same as the projective planar transformation between the focal plane and the image plane of the imagery camera. This observation can be justified by the following relation (see Fig. 3):

$$\begin{aligned} I'_r(x_o,y_o) &= (1-\alpha)\left(I_1(s_o,t_o)+W(s_o,t_o)\right) \\ &\quad + \alpha\left(I_2(s_o,t_o)+W(s_o,t_o)\right) \\ \Rightarrow I'_r(x_o,y_o) &= (1-\alpha)I_1(s_o,t_o)+\alpha I_2(s_o,t_o) \\ &\quad + W(s_o,t_o) \\ \text{where}\quad I'_r(x_o,y_o) &= I_r(x_o,y_o)+W(x_o,y_o). \end{aligned} \tag{3}$$

In (3), $W(s_o,t_o)$ refers to the watermark component embedded to each light ray passing through (s_o,t_o), whereas the other terms are illustrated on Fig. 3. It should be noted that the corresponding weights for the watermark component during bilinear interpolation sum up to one. Hence, the relation between watermarks in (3) is only related with (s_o,t_o) to (x_o,y_o) transformation, which corresponds to planar projective transformation between focal plane and image plane of virtual camera (Fig. 3).

A. Determining Projective Planar Transformation Between Focal Plane and Image Plane

If the corresponding projective planar transformation between the focal plane and the image plane of the virtual camera is estimated, the watermark on the rendered image can also be obtained by applying same transformation on the original watermark pattern. For this reason, 3-D points of the object, which are located on (intersecting with) the focal plane, should be detected and utilized (Fig. 17). In order to determine these points, two properties resulted from light field parameterization can be used. First of all, all light rays of different light field images passing through these points should have similar intensity information. In addition, the coordinates of the pixels in each light field image corresponding to the light ray passing through these points should be close (ideally equal) to each other.

Considering the aforementioned properties, the algorithm for determining the crossing points is proposed as follows:

1) Find matched feature points between the rendered and each of the light field images [30].
2) Determine the light field images that give the first N highest number of feature point matches
3) Form a set of matched points for each of these light field images, $\{S_1 \ldots S_N\}$
4) Determine common feature points on the rendered image which are elements of each set, S_i.
5) From the common feature points, choose the points whose matches in each corresponding light field image have the same pixel location and similar intensity values (black dots in Fig. 17, as a typical example).

During the experiments, N is selected as 4. In Fig. 18, results of the algorithm for the rendered image for a virtual camera (located at [0 0 2.4] with [0 0 1] orientation), is illustrated for *Buddha* light field. The four original light field images which give the maximum number of matched points with the rendered image are *buddha.15.15*, *buddha.16.15*, *buddha.15.16* and *buddha.16.16*.

After determination of these points, the homography between the focal plane and the image plane (Fig. 3) is estimated by a model fit algorithm [24]. In the proposed approach, we utilize a search by randomly selecting 4 correspondences and estimating a homography between the selected pairs. Then, the total reprojection error [24] is determined by applying the estimated homography to all of the pairs. This operation is iterated for a number of times and the model, which results in the minimum error, is chosen, as the homography between the *focal* plane and the *image* plane.

B. Robustness Results

The estimated projective transformation between the focal plane and image plane is applied to the original watermark, W, to compute the rendered watermark W_{ren}. Then, the normalized correlation between the rendered watermark and the high-pass filtered rendered image is computed. Since there can be some small shifts in the calculation of W_{ren} due to the homography

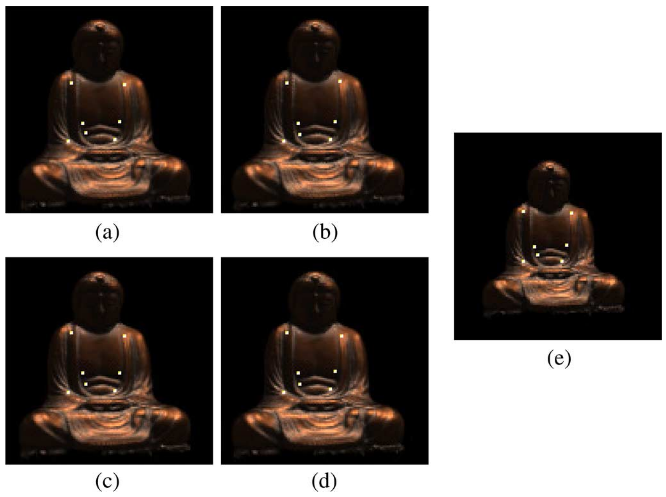


Fig. 18. Results of the algorithm to find the matched points located on the *focal* plane. (a) *Buddha 15.16*, (b) *Buddha 16.16*, (c) *Buddha 15.15*, (d) *Buddha 16.15*, and (e) *rendered image*.

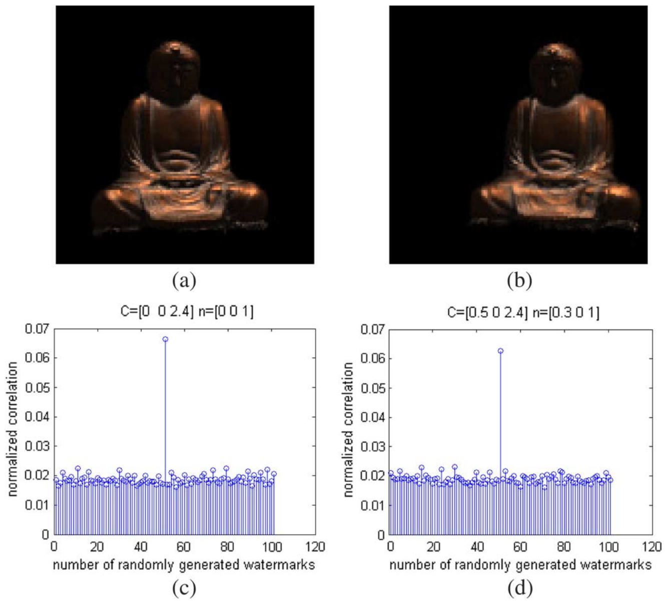


Fig. 19. Rendered image: (a) Camera position $= [0\ 0\ 2.4]$. Image normal $=$ $[0\ 0\ 1]$ (b) Camera position $= [0.5\ 0\ 2.4]$. Image normal $= [0.3\ 0\ 1]$. (c), (d) Normalized correlation results for the images in (a) and (b).

estimation, the normalized correlation is computed by using the magnitude coefficients of 2-D *Fourier* transformation of the images, which are invariant to the translations in the pixel locations [26]. The results for the rendered images for different imagery camera positions and orientations are presented in Fig. 19.

VII. Conclusion

The emerging FTV systems have produced the copy-right problems for multiview video content. In this paper, we have pointed out the essential differences in multiview video watermarking as a solution to this copyright problem compared to the well-studied single-view video watermarking. The main challenge has emerged as the watermark detection from virtual views rendered for arbitrary camera positions. Our proposed method has achieved to detect the watermark from the rendered views both for the known and unknown camera positions by means of analyzing and estimating the variations on the watermark patterns due to the rendering operations. In this pioneer work, we have handled the static scenes consisting of only one object and the main distortions in the transmission chain of multiview video as the attacks on the rendered object. The future work will focus on the extensions of the method for the scenes with multiple objects as well as more sophisticated attacks such as maliciously distorting the utilized feature points in the proposed method or colliding the multiple views for watermark removal.

References

[1] *Three-Dimensional Television: Capture, Transmission, and Display*, H. M. Ozaktas and L. Onural, Eds.. Heidelberg, Germany: Springer, 2007.

[2] ISO/IEC JTC1/SC29/WG11, Requirements on Multi-View Video Coding v.2 Doc. N7282. Poznan, Poland, Jul. 2005.

[3] A. Smolic, H. Kimata, and A. Vetro, "Development of MPEG standards for 3D and free-viewpoint video," in *SPIE Conf. Optics East: Communications, Multimedia & Display Technologies*, Nov. 2005, vol. 6014, pp. 262–273.

[4] A. Smolic, K. Müller, N. Stefanoski, J. Ostermann, A. Gotchev, G. B. Akar, G. A. Triantafyllidis, and A. Koz, "Coding algorithms for 3DTV—A survey," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, no. 11, pp. 1606–1621, Nov. 2007.

[5] A. Koz, G. Triantafyllidis, and A. A. Alatan, *3D Watermarking: Techniques and Directions, in Three-Dimensional Television: Capture, Transmission, and Display*, H. M. Ozaktas and L. Onural, Eds. Heidelberg, Germany: Springer, 2007.

[6] O. Benedens, "Geometry-based watermarking of 3d models," *IEEE Comput. Graph. Appl.*, vol. 19, no. 1, pp. 46–55, Jan. 1999.

[7] R. Ohbuchi, H. Masuda, and M. Aono, "Watermarking 3D polygonal models," in *ACM Multimedia*, Nov. 1997, pp. 261–272.

[8] A. G. Bors, "Watermarking mesh-based representations of 3D objects using local moments," *IEEE Trans. Image Process.*, vol. 15, no. 3, pp. 687–701, Mar. 2006.

[9] E. Garcia and J. L. Dugelay, "Texture-based watermarking of 3D video objects," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 8, pp. 853–866, Aug. 2003.

[10] J. Bennour and J. L. Dugelay, "Protection of 3D object through silhoutte watermarking," in *Proc. 31st IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, May 2006, vol. 2, pp. 221–224.

[11] A. Koz, C. Çığla, and A. A. Alatan, "Free-view watermarking for free-view television," in *Proc. IEEE Int. Conf. Image Processing*, 2006, pp. 1405–1408.

[12] A. Koz, C. Çığla, and A. A. Alatan, "Watermarking for image based rendering via homography-based virtual camera location estimation," in *Proc. IEEE Int. Conf. Image Processing*, 2008, pp. 1284–1287.

[13] C. Zhang and T. Chen, "A survey on image based rendering-representation, sampling and compression," *EURASIP Signal Process.: Image Commun.*, vol. 19, no. 1, pp. 1–28, Jan. 2004.

[14] C. Fehn, P. Kauff, O. Schreer, and R. Schafer, "Interactive virtual view video for immersive TV applications," in *Proc. Int. Broadcast Conf.*, Sep. 2001, vol. 2, pp. 53–62.

[15] M. Tanimoto, "FTV (Free-viewpoint Television) creating ray-based image engineering," in *Proc. IEEE Int. Conf. Image Processing*, 2005, vol. 2, pp. 25–28.

[16] M. Wu, H. Yu, and B. Liu, "Data hiding in image and video: Part II-designs and applications," *IEEE Trans. Image Process.*, vol. 12, no. 6, pp. 696–705, Jun. 2003.

[17] W. R. Mark, L. McMillan, and G. Bishop, "Post-rendering 3D warping," in *Proc. Symp. Interactive 3D Graphics*, Apr. 1997, pp. 7–16.

[18] M. Levoy and P. Hanrahan, "Light field rendering," in *Proc. ACM Siggraph*, Aug. 1996, pp. 31–42.

[19] K. Takehashi, A. Kubota, and T. Naemura, "All in-focus view synthesis from under-sampled light fields," in *Proc. ICAT* , Tokyo, Japan, Dec. 3–5, 2003, vol. 13, pp. 249–256.

[20] A. B. Watson, "DCT quantization matrices visually optimized for individual images," in *Proc. SPIE on Human Vision, Visual Processing, and Digital Display IV*, Feb. 1993, vol. 1993, pp. 202–216.

[21] The Stanford Light Field Archive [Online]. Available: http://graphics.stanford.edu/software/lightpack/lifs.html (Last Access Date: 21 September 2009)

[22] Light Field Archieve of Advanced Multimedia Processing Lab of CMU [Online]. Available: http://amp.ece.cmu.edu/projects/MobileCamArray/ (Last Access Date: 21 September 2009)

[23] M. Grum and A. G. Bors, "Multiple image disparity correction for 3-D scene representation," in *Proc. IEEE Int. Conf. Image Processing*, 2008, pp. 209–212.

[24] R. Hartley and A. Zisserman*, Multiple View Geometry in Computer Vision*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

[25] A. Koz, "Watermarking for 3D Representations," Ph.D. dissertation, Electr. Electron. Eng., Middle East Tech. Univ., Ankara, Turkey, Aug. 2007.

[26] M. Maes, T. Kalker, J.-P. Linnartz, J. Talstra, G. Depovere, and J. Haitsma, "Digital watermarking for DVD video copy protection," *IEEE Signal Process. Mag.*, vol. 17, no. 5, pp. 47–57, Sep. 2000.

[27] Z. Zhang, R. Deriche, O. Faugeras, and Q.-T. Luong, A Robust Technique for Matching Two Uncalibrated Images Through the Recovery of the Unknown Epipolar Geometry INRIA, Report No. 2273, May 1994.

[28] C. Harris and M. Stephens, "A combined corner and edge detector," in *Proc. 4th Alvey Vision Conf.*, 1988, pp. 147–151.

[29] M. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, Jun. 1981.

[30] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, 2004.

[31] J.-X. Chai, X. Tong, S.-C. Chan, and H.-Y. Shum, "Plenoptic sampling," in *Proc. Int. Conf. Computer Graphics and Interactive Techniques*, 2000, pp. 307–318.

Alper Koz was born in Kahramanmaras, Turkey, in 1978. He received the B.S., M.S., and Ph.D. degrees in electrical and electronics engineering from Middle East Technical University, Ankara, Turkey, in 2000, 2002, and 2007, respectively.

In 2008, he joined the Multimedia Signal Processing Laboratory of Delft University of Technology, Delft, The Netherlands, as a postdoctoral researcher. His research interests include multimedia watermarking and fingerprinting in the context of free-view and 3-D televisions, and peer-to-peer networks.

Cevahir Cigla received the M.Sc. degree from the Department of Electrical and Electronics Engineering of Middle East Technical University, Ankara, Turkey, and is currently pursuing the Ph.D. degree at the same university.

His research interests are mainly stereo and multiview image processing, image segmentation and watermarking, and 3-D enhancement.

A. Aydın Alatan (S'91–M'07) was born in Ankara, Turkey in 1968. He received the B.S. degree from Middle East Technical University, Ankara, Turkey, in 1990, the M.S. and DIC degrees from Imperial College of Science, Medicine and Technology, London, U.K., in 1992, and the Ph.D. degree from Bilkent University, Ankara, Turkey, in 1997, all in electrical engineering.

He was a post-doctoral research associate at Center for Image Processing Research at Rensselaer Polytechnic Institute, Troy, NY, between 1997 and 1998 and at the New Jersey Center for Multimedia Research at New Jersey Institute of Technology, Newark, between 1998 and 2000. In August 2000, he joined faculty of Electrical and Electronics Engineering Department at Middle East Technical University.