Compression of Plenoptic Point Clouds

Gustavo Sandri, *Member, IEEE*, Ricardo L. de Queiroz, *Fellow, IEEE*, and Philip A. Chou, *Fellow, IEEE*

***Abstract*— Point clouds have been recently used in applications involving real-time capture and rendering of 3D objects. In a point cloud, for practical reasons, each point or voxel is usually associated with one single color along with other attributes. The region-adaptive hierarchical transform (RAHT) coder has been proposed for single-color point clouds. The cloud is usually captured by many cameras and the colors are averaged in some fashion to yield the point color. This approach may not be very realistic since, in real world objects, the reflected light may significantly change with the viewing angle, especially if specular surfaces are present. For that, we are interested in a more complete representation, the plenoptic point cloud, wherein every point has associated colors in all directions. Here, we propose a compression method for such a representation. Instead of encoding a continuous function, since there is only a finite number of cameras, it makes sense to compress as many colors per voxel as cameras, and to leave any intermediary color rendering interpolation to the decoder. Hence, each voxel is associated with a vector of color values, for each color component. We have here developed and evaluated four methods to expand the RAHT coder to encompass the multiple colors case. Experiments with synthetic data helped us to correlate specularity with the compression, since object specularity, at a given point in space, directly affects color disparity among the cameras, impacting the coder performance. Simulations were carried out using natural (captured) data and results are presented as rate-distortion curves that show that a combination of Kahunen–Loève transform and RAHT achieves the best performance.**

***Index Terms*— Point cloud, plenoptic, compression, image compression.**

I. Introduction

THE plenoptic function is the 5-dimensional function representing the intensity or chromaticity of light observed from every position and direction in a 3-dimensional (3D) space [1]:

$$P(x, y, z, \theta, \phi), \tag{1}$$

where (x, y, z) is a point in space, θ the azimuth and ϕ the elevation angle. Fixing (x, y, z) we get the set of rays passing through a given point, which is referred to as a pencil. There are several ways to capture a plenoptic function sample. If a conventional camera, for example, represented by a pinhole model, is placed at a given position, it captures the pencil at that position and produces an image. One can add more degrees of freedom by adding more conventional cameras or by using a light-field camera [2]–[4] that also registers the angle at which a ray enters the camera.

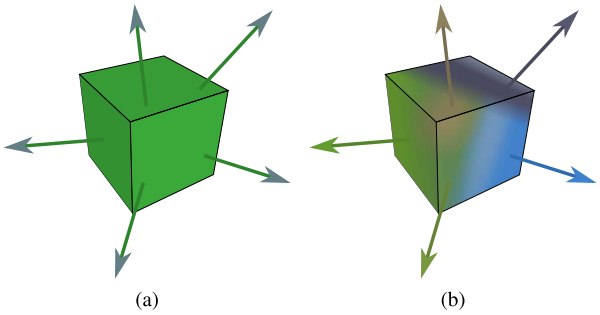

Fig. 1. A non-plenoptic voxel has no directional color information. We define a plenoptic voxel as one wherein the directional color information is present and can be used to represent a scene in a more realistic way. (a) Non-plenoptic voxel. (b) Plenoptic voxel.

We are interested in voxelized point clouds, i.e., the capture space of dimensions $W \times W \times W$ is divided into M^3 volume elements (voxels), each voxel width being W/M. An L-level voxel space is one with $M = 2^L$. Hence, instead of referring to point attributes we refer to voxel attributes.

By processing the information captured by an array of cameras combined with depth maps [5]–[6], or from light-field cameras [7], one can produce a point cloud representing the scene. Point clouds can be used to provide a better representation of the plenoptic function. In such, we define plenoptic voxels where each point (voxel) is seen as a source emitting light in all directions (see Fig. 1).

The light emitted by a voxel in the cloud is determined by its appearance attribute. Traditionally, the appearance contains only a single RGB color triplet (or YUV) as if the voxel emitted the same color in all directions. A few compression methods were devised for point clouds [8]–[15]. While pioneering works [8], [10] proposed unusual compression methods, octrees [9] have since then been prevalent for geometry coding, i.e., coding the position of the occupied voxels. Octrees were also combined with other techniques to encode dynamic (moving) point clouds [11]. Efficient compression of the color attributes (not the geometry) was proposed using Graph Transforms [12], which was later extended to compress dynamic point clouds [13]. Graph-transform-based coding [12] yields top performance. However, it demands computing eigenvectors of many large matrices, which impacts

Manuscript received February 17, 2018; revised July 3, 2018, September 18, 2018, and October 2, 2018; accepted October 3, 2018. Date of publication October 22, 2018; date of current version November 21, 2018. This work was supported by the Conselho Nacional de Pesquisa Científica (CNPq) of Brazil under Grant 308150/2014-7. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Xin Li. *(Corresponding author: Ricardo L. De Queiroz.)*

G. Sandri and R. L. de Queiroz are with the Department of Electrical Engineering, Universidade de Brasilia, Brasília 70910, Brazil (e-mail: gustavo.sandri@ieee.org; queiroz@ieee.org).

P. A. Chou was with 8i Labs, Bellevue, WA 98004 USA. He is now with Google, Seattle, WA 98103 USA (e-mail: pachou@ieee.org).

Digital Object Identifier 10.1109/TIP.2018.2877486

Fig. 2. Three viewpoints of the same scene to highlight the color variation according to viewing direction. The colors at same point in the wet floor or the car surface vary when the viewing point changes.

overall complexity. Such an approach was only surpassed with another similar approach but with a different model, which is based on Gaussian Process Transforms [15]. A more practical lower-complexity approach for color attribute compression was later developed [14] which is based on a region-adaptive hierarchical transform (RAHT) and rivals the more complex methods [12], [15]. All these methods, unlike the present case, assume a single color per voxel.

The concept of a voxel emitting the same color in all directions might not be efficient in many cases, especially with specular surfaces where the color of a given point significantly varies according to the view direction (See Fig. 2). The extreme case is a mirror, for which the emitted color is the color of the reflected environment.

In this work, we associate to each voxel the colors seen by all cameras. Since cameras are placed at different positions, if we know the camera and voxel positions, one can calculate samples of the plenoptic function for each voxel. We refer to this representation as the plenoptic point cloud. For other directions beyond those sampled (captured) we can synthesize the color by interpolation. Nevertheless, view synthesis is out of the scope of this work.

II. Plenoptic Point Clouds

We define a voxelized plenoptic point clouds as a collection of occupied voxels for which we attribute not only geometry, but also color attributes for each voxel as a continuous function in all directions, as depicted in Fig. 1. Such a continuous function can be represented in different ways. We, here, opted for a sampled representation, for which the renderer at the receiver side should interpolate the colors for desired positions in between samples. We also opted to place the samples, not uniformly over a sphere around the voxel, but at the direction of the existing cameras. Since many cameras may be occluded at each voxel, such a representation may lead to a variable number of samples per voxel. In order to deal with that complication, we further opted to interpolate the missing cameras at the capture point, i.e., the number of samples (cameras) per voxel is fixed across all voxels in the point cloud. These three choices are part of the contribution in this paper and we will show that we can make this representation quite efficient.

Our work on compressing plenoptic point clouds is closely related to work on light field compression. Light field compression is an active area of research [16] as well as the subject of ongoing standardization activities in both JPEG and MPEG [17], [18]. In both research and standards activities, the dominant approach to date is compression of the 4D plenoptic function,

$$P(u, v, s, t), \tag{2}$$

which is parametrized by the location (u, v) where a ray enters a bounding volume through a surface and the location (s, t) of where it leaves the bounding volume. While such a parameterization suffices for so-called "windowed 6 degrees of freedom" (6-DOF), or "omni directional 6-DOF" it does not suffice for "full 6-DOF" otherwise known as free-viewpoint navigation, for which the 5D plenoptic function, such as treated in our paper, is required.

In not-yet-published upcoming work [19]–[21] Zhang *et al.* present a follow-on to our work from the point of view of using a point cloud codec to compress surface light fields, which are equivalent to our 5D plenoptic function (2). Their approach differs from ours in that, while we directly code N_c transform coefficients of the N_c colors for each point, they instead fit a continuous interpolating function to the N_c colors in the θh plane (see next section), and code the coefficients of the basis functions that represent the interpolant in some continuous basis. The difference is primarily in where the responsibility for novel view synthesis lies. In our work, as noted in the Introduction, novel view synthesis lies outside our scope. In contrast, Zhang et al. assume through their choice of basis how to perform view synthesis, which is built into their encoding. Which architecture is better depends on the system requirements including whether an encoding independent of the view synthesis technique is required, complexity considerations at the encoder and decoder, etc. Another upcoming work also presents preliminary variations of the present work [22].

There has also been previous work in compressing surface light fields [23], [24] using transform coefficients to represent and compress the color vector at each point n; however they do not compress the coefficient vectors spatially across the surface. In contrast, we use RAHT to remove the redundancy between spatially adjacent vectors of coefficients, before coding. We believe that we are the first to propose removing spatial redundancy in this way. In summary, our contribution is based on mentioned options to process a fixed number of samples per voxel at exactly the camera positions, and to transform-code the data in both the spatial and color (camera) domains.

III. Color Vector Transforms

As mentioned, a plenoptic point cloud comprises a list of voxels $\{v_i\}$, each being described by its geometry (location in space) and color (in RGB or YUV color spaces) seen by the N_c cameras, *i.e*,

$$v_i = \left[x_i, y_i, z_i, R_i^1, G_i^1, B_i^1, \cdots, R_i^{N_c}, G_i^{N_c}, B_i^{N_c}\right]. \tag{3}$$

The color vector $\left[R^1, G^1, B^1, \cdots, R^{N_c}, G^{N_c}, B^{N_c}\right]$, is referred as the plenoptic appearance attribute of a voxel. See Fig 1. It usually contains a high level of redundancy that can be exploited for compression. It can be compared to 2D images where neighboring pixels are usually highly redundant, as well. Most compression algorithms for

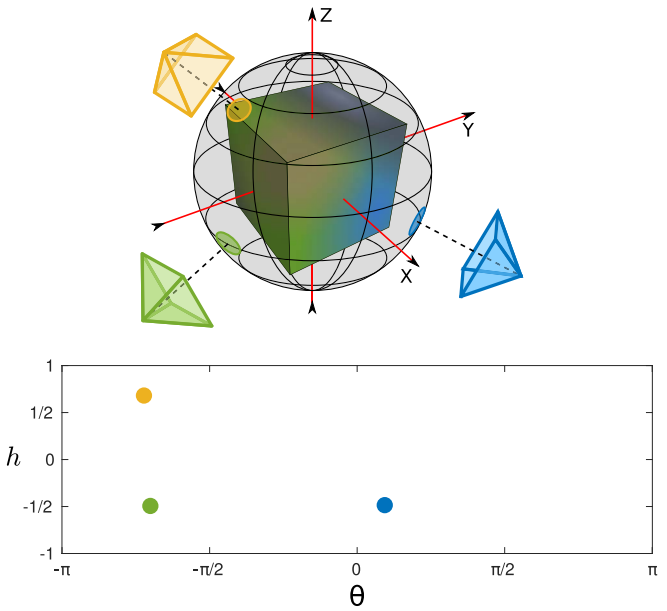

Fig. 3. Cameras viewing the point (voxel) from different directions are mapped onto the θh plane.

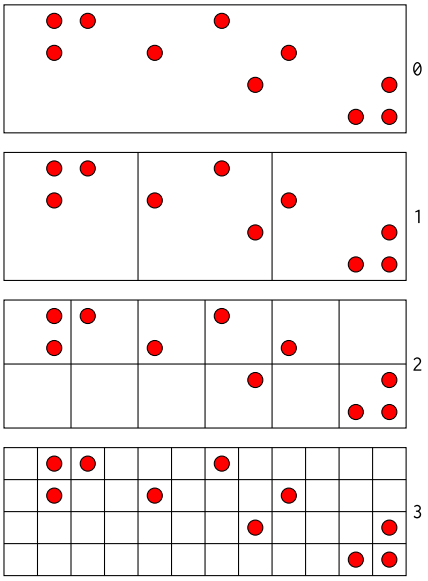

Fig. 4. The color captured by each camera is projected onto the θh plane according to the direction where they see the voxel surface. The plane is then divided in sub-regions. The number of divisions defines the precision in which the camera's position is represented. The final subdivision is associated with a color or it is empty, if there are no cameras in that position.

2D images apply a transform to provide energy compaction in order to enable compression. Hence, we first apply a transform to the plenoptic appearance color vector (one transform for each color component, independently). For each component, it generates $N_c - 1$ high-pass coefficients and one DC coefficient that represents the average color of a voxel. We will describe instantiations of this transformation in the following subsections.

A. θh Plane

Let us assume a very small sphere, surrounding a voxel, viewed by several cameras, as illustrated in Fig. 3. Each camera may be observing different colors as the light that the sphere emits may vary with the direction. These cameras are capturing a sample of the plenoptic function at this particular point in space.

The direction of the camera, i.e., a point over the sphere, can be described in spherical coordinates by the angles $-\pi \leq \theta \leq \pi$ (azimuth or longitude) and $\pi/2 \leq \varphi \leq \pi/2$ (polar or latitude). It is often more convenient to represent the sphere surface using a cylindrical equal-area projection, replacing φ with $h = \sin(\varphi)$. We refer to this as the θh plane, which is illustrated in Fig. 3. It should be appreciated that a better sampling of the plenoptic function may be obtained if the camera views are uniformly spread over the θh plane.

B. RAHT Over the θh Plane

We generate a two-dimensional map over the θh plane with a projection of the colors as seen in every direction captured by the cameras (Fig. 3). One may divide the camera directions (θh) plane into sub-regions of equal area as depicted in Fig. 4. Each area is a quantized description of the camera direction. We may further divide each sub-region several times until attaining the desired precision. The smaller the sub-region, the more precise the camera position is represented. The number of divisions should be chosen in order to make the position information for two or more cameras not to fall within the same sub-region. In Fig. 4, after dividing the plane, several sub-regions remain unoccupied. This representation is similar to voxelized point clouds in the 3D space. Therefore, we apply RAHT [14] to the colors associated with each camera (sub-region), through a 2D quad-tree decomposition rather than the 3D octree.

The hierarchical transform generates $N_c - 1$ high-pass coefficients and one DC coefficient, ordered according to the depth of the quad-tree where they were generated. Deeper coefficients are associated to higher frequencies.

We combine the RAHT over the θh plane with the RAHT over the voxel spatial coordinates (xyz) in two ways:

RAHT-1 – In RAHT, we start from the leaves and transform attributes all the way to the root (DC value). Once the DC value of the cameras is obtained, we propagate it into the voxel space (xyz coordinates), i.e., the DC value becomes the color of the voxel, and we follow the rest of RAHT until reaching an overall DC value for all voxels and cameras. The coefficients to be encoded, using the same entropy coding as in [14], are the AC values of the cameras (θh space), the spatial AC values (xyz space), and the DC value.

RAHT-2 – When transforming the color samples in the θh plane we end up with N_c coefficients for each voxel: one DC value and $N_c - 1$ AC ones. Unlike the previously described approach, we assume all coefficients to be attributes and all are subject to further processing for encoding. One can view it as if there were N_c point clouds, where voxel colors are the coefficient values, all sharing the same geometry. Each cloud is then transformed and encoded using the RAHT coder.

C. Transforming the Color Vector

Let the colors of the n-th voxel be $R_i(n)$, $G_i(n)$ and $B_i(n)$, for $1 \leq i \leq N_c$. The colors are transformed into YUV space and let C represent one of the color components such that

$$\mathbf{c}(n) = [C_1(n), C_2(n), \ldots, C_{N_c}(n)]^T \tag{4}$$

represents that color component for a given voxel. If there is a linear transform $\mathbf{H}$ such that

$$\mathbf{s}(n) = \left[S_1(n), S_2(n), \ldots, S_{N_c}(n)\right]^T = \mathbf{H}\ \mathbf{c}(n), \tag{5}$$

then each transformed signal $S(n)$ can be viewed as an attribute of a point cloud to be transformed and encoded. For a trichromatic color space such as YUV, there would be $3N_c$ data sets (point clouds) to undergo the RAHT transforming and encoding procedures, all sharing the same geometry.

In this approach, our preferred transform is the Kahunen-Loève transform (KLT). Briefly, we first compute the mean of the C samples, per camera, per color component, i.e.,

$$\mu_C^i = \frac{1}{N}\sum_{n=1}^{N} C_i(n) \tag{6}$$

We, then, remove the mean and compute the $N_c \times N_c$ covariance matrix $\Gamma = \{\Gamma(i, j), 1 \leq i, j \leq N_c\}$ among the camera signals $\{C_\ell(n)\}$. Hence,

$$\Gamma(i, j) = \frac{1}{N-1}\sum_{n=1}^{N}\left(C_i(n) - \mu_C^i\right)\left(C_j(n) - \mu_C^j\right). \tag{7}$$

We compute eigenvectors and eigenvalues of Γ through the singular value decomposition as $\Gamma = \mathbf{H}\Lambda\mathbf{H}^T$, where $\mathbf{H}$ contains the eigenvectors of Γ as its columns and Λ is a diagonal matrix with the eigenvalues of Γ. $\mathbf{H}$ is the KLT of the camera vector signal and is used to transform each $\mathbf{c}(n)$.

As a side information, it is necessary to send the covariance matrix Γ to the decoder, so that it can properly calculate its inverse. Such information is sent in the file header using 32 bits floating point numbers. As the covariance matrix is symmetrical, it is only necessary to send $N_c(N_c + 1)/2$ elements for each color component, instead of N_c^2. The overhead for a level-10 point cloud with 1.5 million occupied voxels operating at 0.1 bits/voxel/camera, for 12 cameras, takes less than 0.15% of the total bit-rate and is not a burden for the compression. We refer to this method as RAHT-KLT.

RAHT-KLT has a few disadvantages. Firstly, it would only work if we interpolate the cameras to force the number of color samples N_c to be constant, which is clearly wasteful. Secondly, we need to compute statistics and perform matrix inversions. We were unable to model a set of covariance coefficients that would work across many point clouds as a reference to replace the true KLT. This is caused by a few factors, such as uneven camera displacement, reduced availability of view-dependent point cloud data sets, statistical disparities among specular and diffuse regions. A discrete cosine transform (DCT) has been successfully used to replace the KLT in regular 2D image compression. However, in 3D there is no predefined concept of sequencing among the cameras and there are many possible permutations of the cameras in building vector $\mathbf{c}$. It is desirable that the amplitudes may not abruptly vary from one sample to its neighbor. This may be a problem if the cameras are randomly ordered. Therefore, for every voxel, its plenoptic appearance attributes are ordered according to the direction of the camera relative to the voxel, in a spiral manner, starting from the bottom, as depicted in Fig. 5, which amounts to left-right-top-down scanning of the θh plane. Hence, we can select $\mathbf{H}$ as the $N_c \times N_c$ DCT and we refer to this method as RAHT-DCT.

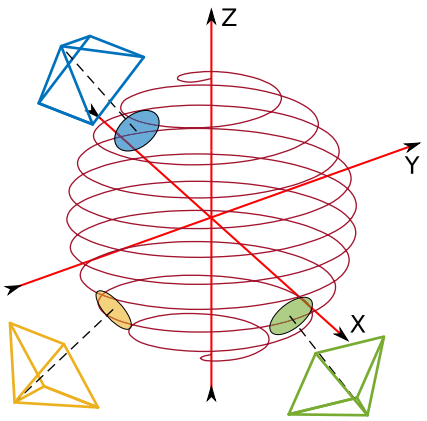

Fig. 5. Camera ordering for the RAHT-DCT approach.

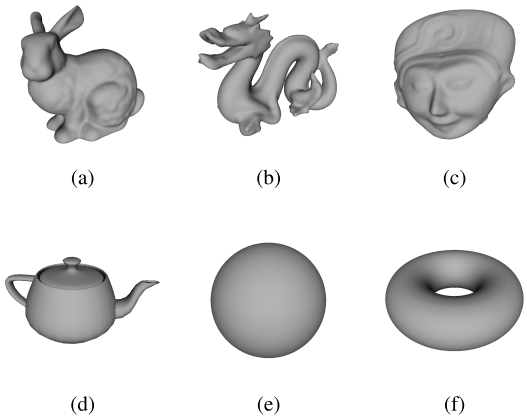

Fig. 6. 3D models. (a) Bunny. (b) Dragon. (c) Egyptian_mask. (d) Teapot. (e) Sphere. (f) Torus.

IV. Specularity and Transform Performance

To test the effectiveness of our algorithm for different object reflectivity, we carried tests with synthetic point clouds where we could control the specularity. The synthetic point clouds were created using widely known 3D models (as in Fig. 6) and adjusted reflection characteristics.

We took 10 frames from 10 omnidirectional videos available at Youtube®. See Table I. The 3D models were voxelized using a depth of 9 and, thus, are contained in a box of size $512 \times 512 \times 512$ voxels. The θh plane was voxelized with a depth of 6 for the quad-tree. The omnidirectional images are projected within a sphere, centered around the object, with a radius of 5×512. 20 cameras are uniformly distributed around

TABLE I
SPHERICAL IMAGES

#	Video
1	Wild Dolphins
2	Maldives
3	The Eye Of The Tiger
4	Underwater National Park
5	Fifty Shades Darker
6	NASA Encapsulation Launch of OSIRIS REx
7	Peru Presenta Cusco en Realidad Virtual
8	Clash of Clans Hog Rider
9	Angel Falls Venezuela
10	Hawaii The Pace of Formation

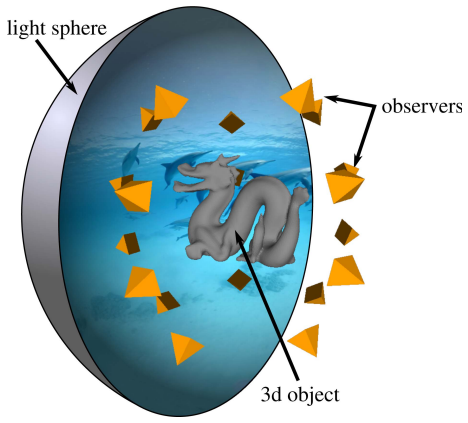

Fig. 7. Synthetic data. A 3D object model is placed inside a sphere that acts as a light source. 20 observers (cameras) are uniformly placed around the object.

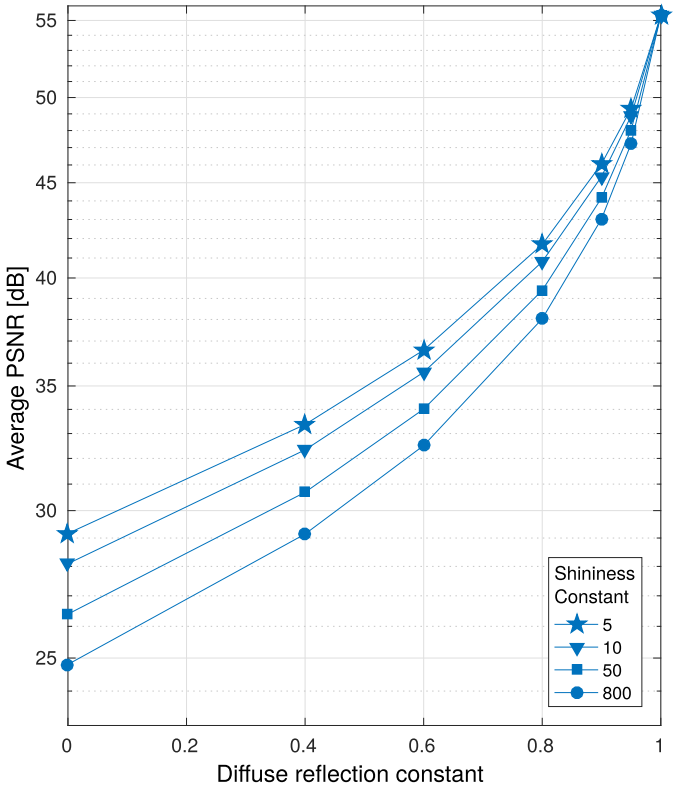

Fig. 8. Distortion curves to evaluate the effect of diffuse reflection in the compression of the synthetic data. Bit-rate was set at 0.2 bits/voxel/camera, using the RAHT-KLT method.

TABLE II
DATABASE

Image	Number of Points	Voxels	Cameras
boxer	3496011	2056256	13
longdress	3100469	1860104	12
loot	3021497	1858707	13
redandblack	2776067	1467981	12
soldier	4007891	2365732	13
thaidancer	3130215	1719408	13

the object at a radius of 2×512. Fig. 7 is an out of scale representation of the setup used to synthesize the data.

We adopted Phong's shading model [25] to compute the light reflected by each voxel in the directions observed by each of the 20 cameras. It has four parameters: the diffuse reflection constant k_d (ratio of light reflected in all directions); the specular reflection constant k_s (ratio of light emitted in the direction that a perfectly reflected ray would take) that we chose to be equal to $1 - k_d$; the ambient reflection constant that we chose to be zero; and the shininess constant, which is larger for surfaces that are smoother and more specular.

To evaluate the performance we fixed the rate at 0.2 bits/voxel/camera (in this work, we always mean bits per occupied voxel to describe bit-rates) and we computed the average peak signal-to-noise ratio (PSNR) between original and reconstructed attributes (colors), for a fixed set of reflection constants while varying the 3D model and spherical image. The diffuse reflection constant was varied between 0 and 1 while the shininess constant was assumed values of 5, 10, 50 and 800.

In objects that are more specular than diffuse, the light reflected from a voxel can highly vary according to the viewing angle, thus reducing correlation and the overall compression. This can be seen in Fig. 8 where the PSNR of the decompressed image is higher for higher values of the diffuse reflection constant. The shininess constant also influences the performance as low values are associated with rough surfaces, while high values are associated with smooth ones. Rough surfaces tend to randomly deflect the direction of the reflected rays, producing more correlated appearance attributes. As shown in Fig. 8, the higher the shininess constant, the lower the PSNR of the decompressed image for the same bit rate.

V. SIMULATION RESULTS

Fig. 9 compares the performance among the methods for a fixed bit-rate (0.2 bits/voxel/camera), but varying the appearance constants. In order to facilitate the comparison, the PSNR numbers are given relative to the performance of the RAHT-KLT method and the PSNR was computed by computing the errors over the RGB values. When the diffuse reflection constant is below 0.6, RAHT-2 has a performance close to RAHT-KLT presenting a distortion at most 0.5 dB below that of RAHT-KLT. However, the performance of RAHT-2 quickly worsens compared with RAHT-KLT as k_d approaches 1.

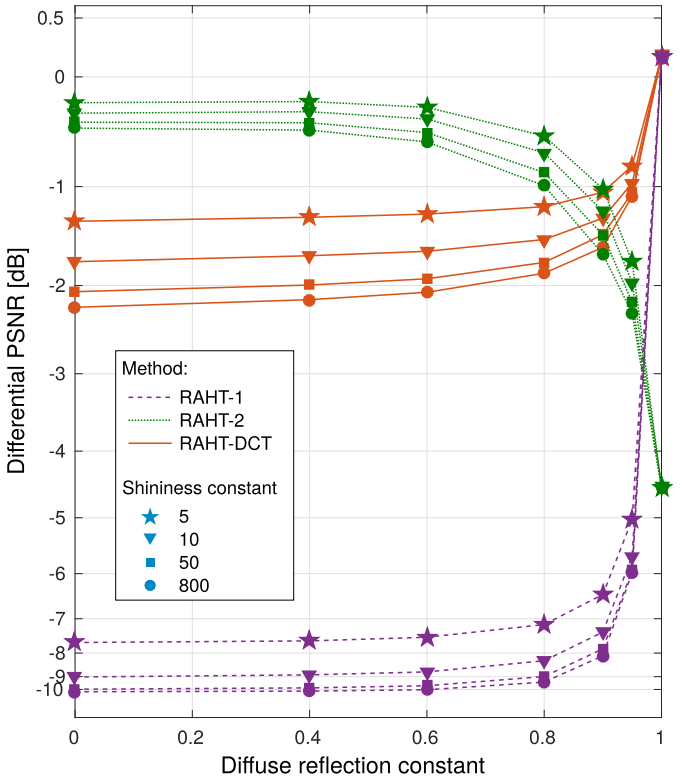

Fig. 9. Performance comparison among the methods for the synthetic clouds in our test set, varying parameters. Data was compressed at 0.2 bits/voxel/camera and PSNR was averaged over all 3D models and presented in differential form relative to the numbers of the RAHT-KLT method.

(a) Boxer (b) Longdress (c) Loot

(d) Redandblack (e) Soldier (f) Thaidancer

Fig. 10. Rendered Images of our captured view-dependent point clouds. (a) Boxer. (b) Longdress. (c) Loot. (d) Redandblack. (e) Soldier. (f) Thaidancer.

The synthetic dataset is very helpful to study the behavior of view-dependent and plenoptic representation, but it is not representative for our target of realistic scenes. For that, we also carried tests on 6 realistic real-time-captured images. They

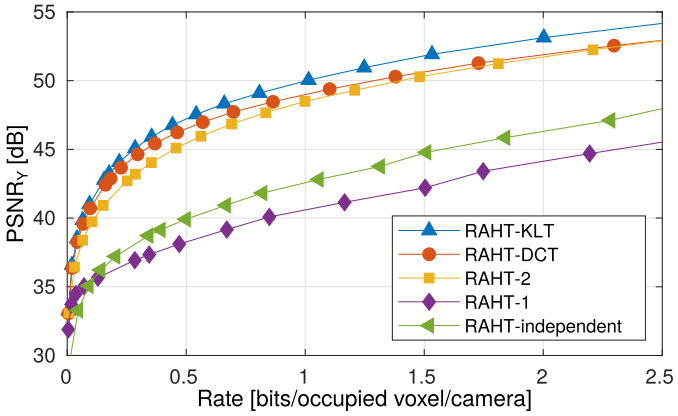

Fig. 11. Rate-distortion curve for *Boxer*.

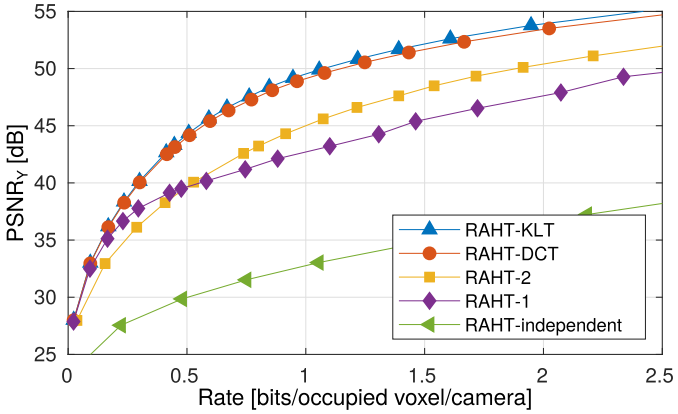

Fig. 12. Rate-distortion curve for *Longdress*.

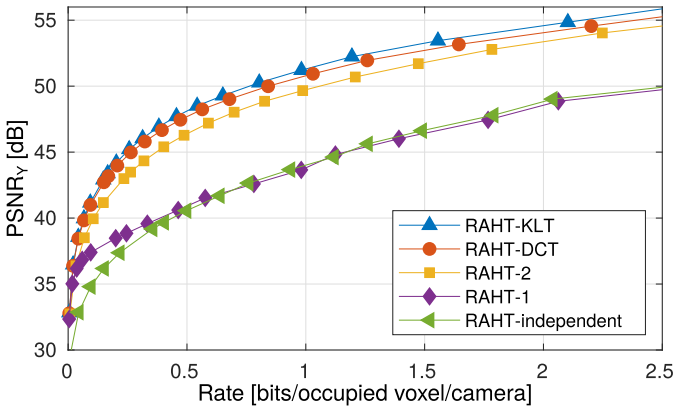

Fig. 13. Rate-distortion curve for *Loot*.

were captured with 12 or 13 cameras and around 3 million points. See Table II and Fig. 10. These images were voxelized using 11 bits of spatial resolution (octree with a depth level $L = 11$), resulting in around 2 million voxels. The θh plane was created with a depth of 6 for the quad-tree. The colors are represented in the RGB space and transformed into YUV space for compression. In our experiment, the quantization step was exponentially varied from 1.3 to 300 and we used the RAHT-based coder with the modified entropy coding described in [26]. The results in terms of rate-distortion (RD) curves are shown in Figs. 11–16. The PSNR (PSNR$_Y$) is evaluated for the Y channel. Average gains in PSNR and

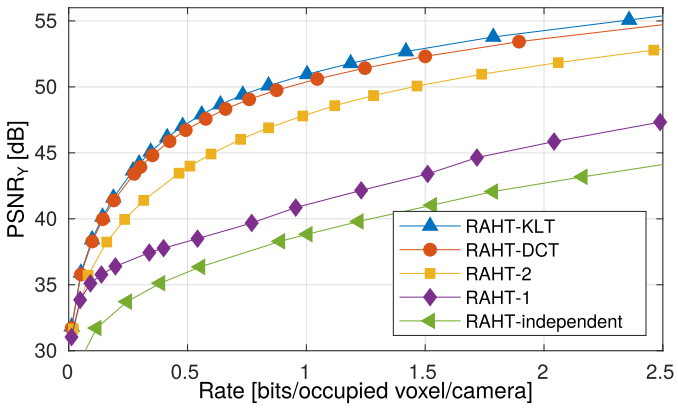

Fig. 14. Rate-distortion curve for *Redandblack*.

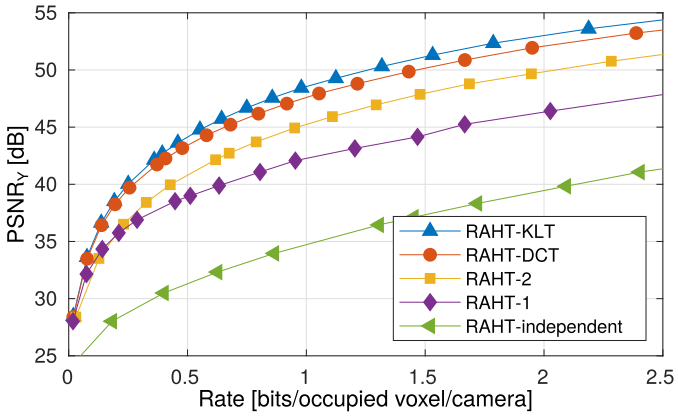

Fig. 15. Rate-distortion curve for *Thaidancer*.

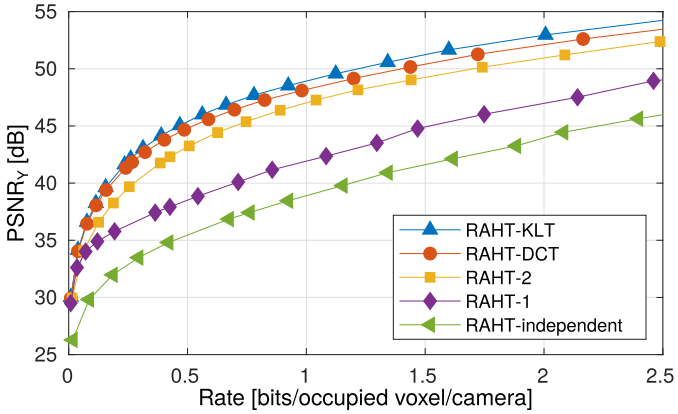

Fig. 16. Rate-distortion curve for *Soldier*.

savings in bit-rate are given in Tables III and IV for all point clouds, color channels and methods. The reference is the independent RAHT compression of each camera view and we used the popular average-difference metric used in video coding standards [27]. The measurements (differences in rate or PSNR) for the YUV color channels were taken as averages over all camera viewpoints and most bit-rates.

The RAHT-KLT has been shown to provide the best performance among all methods. This is expected since its transform

TABLE III

AVERAGE PSNR DIFFERENCE COMPARED TO INDEPENDENT USE OF RAHT

Point Cloud		RAHT-KLT	RAHT-DCT	RAHT-2	RAHT-1
Boxer	Y	**7.11**	6.15	5.64	-2.14
	U	**3.88**	3.18	2.82	-0.70
	V	**3.78**	3.08	2.71	-0.75
Longdress	Y	**16.98**	16.53	12.99	10.49
	U	**15.46**	15.19	11.49	12.12
	V	**15.72**	15.44	11.74	12.42
Loot	Y	**6.67**	6.37	5.18	-0.19
	U	**3.99**	3.62	2.88	-0.43
	V	**4.06**	3.65	2.91	-0.31
Redandblack	Y	**11.90**	11.29	8.90	2.51
	U	**10.07**	9.78	7.61	4.72
	V	**12.74**	12.34	10.03	7.2953
Soldier	Y	**9.58**	8.73	7.66	3.07
	U	**5.82**	5.36	4.28	1.75
	V	**5.50**	5.09	4.03	1.93
Thaidancer	Y	**13.58**	12.69	10.37	7.15
	U	**12.31**	11.77	9.53	7.10
	V	**12.08**	11.49	9.25	6.78

TABLE IV

AVERAGE BITRATE SAVINGS (NEGATIVE INCREASE) COMPARED TO INDEPENDENT USE OF RAHT

Point Cloud		RAHT-KLT	RAHT-DCT	RAHT-2	RAHT-1
Boxer	Y	**-75.58%**	-70.91%	-66.74%	50.40%
	U	**-74.59%**	-70.20%	-63.67%	36.88%
	V	**-75.75%**	-71.41%	-64.81%	43.60%
Longdress	Y	**-89.92%**	-89.66%	-83.45%	-83.79%
	U	**-91.73%**	-91.60%	-84.44%	-89.98%
	V	**-92.14%**	-91.96%	-84.54%	-90.55%
Loot	Y	**-68.31%**	-65.10%	-57.96%	2.40%
	U	**-75.19%**	-72.59%	-64.29%	20.52%
	V	**-75.59%**	-72.91%	-64.68%	14.73%
Redandblack	Y	**-86.29%**	-85.28%	-78.24%	-34.67%
	U	**-89.28%**	-88.54%	-81.43%	-75.30%
	V	**-89.89%**	-89.40%	-82.03%	-79.53%
Soldier	Y	**-76.53%**	-73.67%	-68.79%	-35.38%
	U	**-82.67%**	-81.11%	-74.36%	-45.46%
	V	**-83.15%**	-81.71%	-74.91%	-52.22%
Thaidancer	Y	**-86.18%**	-84.57%	-77.53%	-68.13%
	U	**-89.77%**	-88.84%	-82.12%	-82.33%
	V	**-89.76%**	-88.86%	-81.91%	-81.57%

is adapted to the data statistics, encompassing all degrees of specularity and diffuseness found for each point cloud.

The results using the real point clouds show that the methods can be ordered, from best to worst, as: RAHT-KLT, RAHT-DCT, RAHT-2 and RAHT-1. With synthetic data this is observed when $0.9 < k_d < 0.95$, which indicates that the objects in the real scenes are more diffuse than specular, but not Lambertian.

VI. CONCLUSION

Objects with higher specularity may be less accurately represented by traditional single-color point clouds that are often used in real-time 3D capture and rendering. We extended the

representation to encompass multiple-color voxels, which are samples of the plenoptic function for each voxel, thus referring to them as plenoptic point clouds. We have developed and tested four methods to extend the RAHT coder to compress plenoptic point clouds. RAHT-based coder is an excellent candidate for compression because of its simplicity of implementation allied with competitive rate-distortion performance. Two of the proposed approaches extend the RAHT over the sphere representing the plenoptic function (RAHT-1 and RAHT-2), while two other approaches involve the combination of RAHT with 1D transforms over the cameras color vector (RAHT-KLT and RAHT-DCT). Among all four proposed methods, RAHT-KLT presents the best overall performance, even when the objects in the scene varies from completely specular to plainly diffuse.

As future work, much is still to be done for improving the coder, specially with regards to entropy coding and in trying to identify regions of higher specularity.

ACKNOWLEDGMENTS

The authors would like to thank 8i Labs Inc. for contributing the captured view-dependent point clouds used in this work.

REFERENCES

[1] E. H. Adelson and J. R. Bergen, "The plenoptic function and the elements of early vision," in *Computational Models of Visual Processing*, M. Landy and J. A. Movshon, Eds. Cambridge, MA, USA: MIT Press, 1991, pp. 3–20.

[2] T. Georgiev, *et al.*, "Spatio-angular resolution tradeoffs in integral photography," in *Proc. 17th Eurograph. Conf. Rendering Techn. (EGSR)*. Aire-la-Ville, Switzerland: Eurographics Association, 2006, pp. 263–272.

[3] T. Georgiev, S. Tambe, A. Lumsdaine, J. Gille, and A. Veeraraghavan, "The radon image as plenoptic function," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Oct. 2014, pp. 1922–1926.

[4] N. Sabater, M. Seifi, V. Drazic, G. Sandri, and P. Pérez, "Accurate disparity estimation for plenoptic images," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2015, pp. 548–560.

[5] S. Orts-Escolano *et al.*, "Holoportation: Virtual 3D teleportation in real-time," in *Proc. 29th Annu. Symp. Interface Softw. Technol. (UIST)*, New York, NY, USA, 2016, pp. 741–754.

[6] A. Pujol-Miro, J. Ruiz-Hidalgo, and J. R. Casas, "Registration of images to unorganized 3D point clouds using contour cues," in *Proc. 25th Eur. Signal Process. Conf. (EUSIPCO)*, Aug./Sep. 2017, pp. 81–85.

[7] C. Perra, F. Murgia, and D. Giusto, "An analysis of 3D point cloud reconstruction from light field images," in *Proc. 6th Int. Conf. Image Process. Theory, Tools Appl. (IPTA)*, Dec. 2016, pp. 1–6.

[8] T. Ochotta and D. Saupe, "Compression of point-based 3D models by shape-adaptive wavelet coding of multi-height fields," in *Proc. 1st Eurograph. Conf. Point-Based Graph.* Aire-la-Ville, Switzerland: Eurographics Association, 2004, pp. 103–112.

[9] R. Schnabel and R. Klein, "Octree-based point-cloud compression," in *Proc. 3rd Eurographics/IEEE VGTC Conf. Point-Based Graph.* Aire-la-Ville, Switzerland: Eurographics Association, 2006, pp. 111–121.

[10] Y. Huang, J. Peng, C. C. J. Kuo, and M. Gopi, "A generic scheme for progressive point cloud coding," *IEEE Trans. Vis. Comput. Graph.*, vol. 14, no. 2, pp. 440–453, Mar/Apr. 2008.

[11] J. Kammerl, N. Blodow, R. B. Rusu, S. Gedikli, M. Beetz, and E. Steinbach, "Real-time compression of point cloud streams," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, Saint Paul, MN, USA, May 2012, pp. 778–785.

[12] C. Zhang, D. Florêncio, and C. Loop, "Point cloud attribute compression with graph transform," in *Proc. IEEE Int. Conf. Image Process.*, Oct. 2014, pp. 2066–2070.

[13] D. Thanou, P. A. Chou, and P. Frossard, "Graph-based motion estimation and compensation for dynamic 3D point cloud compression," in *Proc. IEEE Int. Conf. Image Process.*, Sep. 2015, pp. 3235–3239.

[14] R. L. De Queiroz and P. A. Chou, "Compression of 3D point clouds using a region-adaptive hierarchical transform," *IEEE Trans. Image Process.*, vol. 25, no. 8, pp. 3947–3956, Aug. 2016.

[15] R. L. de Queiroz and P. A. Chou, "Transform coding for point clouds using a Gaussian process model," *IEEE Trans. Image Process.*, vol. 26, no. 7, pp. 3507–3517, Jul. 2017.

[16] F. Pereira, E. A. B. Da Silva, and G. Lafruit, "Plenoptic imaging: Representation and processing," in *Academic Press Library in Signal Processing*, vol. 6. 2018, doi: 10.1016/B978-0-12-811889-4.00002-6.

[17] G. Lafruit, S. Quackenbush, S. Foessel, and A. Hinds, *Technical Report of the Joint Ad Hoc Group for Digital Representations of Light/Sound Fields for Immersive Media Applications*, document ISO/IEC JTC1/SC29/WG1N72033, Geneva, Switzerland, Jun. 2016.

[18] *JPEG Pleno Call for Proposals on Light Field Coding*, Standard ISO/IEC JTC 1/SC29/WG1 N73013, Chengdu, China, Oct. 2016.

[19] X. Zhang *et al.*, "A framework for surface light field compression," in *Proc. 25th IEEE Int. Conf. Image Process. (ICIP)*, Athens, Greece, Oct. 2018, pp. 2595–2599.

[20] X. Zhang, P. A. Chou, M.-T. Sun, M. Tang, S. Wang, and S. Ma, *Light Field Compression Uing a Point Cloud Codec (PCC)*, document m42366, ISO/IEC JTC1/SC29/WG11, San Diego, CA, USA, Apr. 2018.

[21] X. Zhang *et al.* (2018). "Surface light field compression using a point cloud codec." [Online]. Available: https://arxiv.org/abs/1805.11203

[22] G. Sandri, R. L. De Queiroz, and P. A. Chou, "Compression of plenoptic point clouds using the region-adaptive hierarchical transform," in *Proc. 25th IEEE Int. Conf. Image Process. (ICIP)*, Athens, Greece, Oct. 2018, pp. 1153–1157.

[23] D. N. Wood *et al.*, "Surface light fields for 3D photography," in *Proc. 27th Annu. Conf. Comput. Graph. Interact. Techn.* New York, NY, USA: ACM Press, 2000, pp. 287–296.

[24] W.-C. Chen, J.-Y. Bouguet, M. H. Chu, and R. Grzeszczuk, "Light field mapping: Efficient representation and hardware rendering of surface light fields," *ACM Trans. Graph.*, vol. 21, no. 3, pp. 447–456, Jul. 2002.

[25] B. T. Phong, "Illumination for computer generated pictures," *Commun. ACM*, vol. 18, no. 6, pp. 311–317, Jun. 1975.

[26] G. Sandri, R. L. De Queiroz, and P. A. Chou. (May 2018). "Comments on 'compression of 3D point clouds using a region-adaptive hierarchical transform." [Online]. Available: https://arxiv.org/abs/1805.09146

[27] G. Bjontegaard, *Calculation of Average PSNR Differences Between RD-Curves*, document VCEG-M33, ITU-T SG16/Q6, Austin, TX, USA, 2001.

Gustavo Sandri (M'14) received the Engineer degree from the Universidade de Brasilia, Brazil, in 2013, the master's degree in signal and image processing by a partnership between the Université de Bordeaux and ENSEIRB-MATMECA, Bordeaux, France, in 2013, and the M.Sc. degree in signal processing from the Universidade de Brasilia in 2016, where he is currently pursuing the Ph.D. degree. Since 2018, he has been a Lecturer of electronics with the Instituto Federal de Brasilia.

Ricardo L. de Queiroz (M'88–SM'99–F'17) received the Engineer degree from the Universidade de Brasilia, Brazil, in 1987, the M.Sc. degree from Universidade Estadual de Campinas, Brazil, in 1990, and the Ph.D. degree from The University of Texas at Arlington, in 1994, all in electrical engineering.

From 1990 to 1991, he was with the DSP Research Group, Universidade de Brasilia, as a Research Associate. He joined Xerox Corporation in 1994, where he was a member of the research staff until 2002. From 2000 to 2001, he was also an Adjunct Faculty with the Rochester Institute of Technology. He joined the Electrical Engineering Department, Universidade de Brasilia, in 2003. In 2010, he became a Full (Titular) Professor with the Computer Science Department, Universidade de Brasilia. Since 2015, he has been a Visiting Professor with the University of Washington, Seattle.

Dr. de Queiroz has published extensively in Journals and conferences and contributed chapters to books as well. He also holds 46 issued patents. He is a past Elected Member of the IEEE Signal Processing Society's Multimedia Signal Processing (MMSP) and the Image, Video and Multidimensional Signal Processing Technical Committees. He is a an Editor for the IEEE TRANSACTIONS ON IMAGE PROCESSING and a past Editor for the *EURASIP Journal on Image and Video Processing*, the IEEE SIGNAL PROCESSING LETTERS, and the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY. He has been appointed an IEEE Signal Processing Society Distinguished Lecturer from 2011 to 2012.

His research interests include image and video compression, point cloud compression, multirate signal processing, and color imaging. He is a member of the Brazilian Telecommunications Society. He has been actively involved with IEEE Signal Processing Society chapters in Brazil and in U.S. He was the General Chair of ISCAS'2011, MMSP'2009, and SBrT'2012. He was also part of the organizing committee of many SPS flagship conferences.

Philip A. Chou (M'81–SM'00–F'03) received the B.S.E. degree in electrical engineering and computer science from Princeton University, Princeton, NJ, USA, and the M.S. degree in electrical engineering and computer science from the University of California at Berkeley, Berkeley, in 1980 and 1983, respectively, and the Ph.D. degree in electrical engineering from Stanford University in 1988. From 1988 to 1990, he was a member of Technical Staff at AT & T Bell Laboratories, Murray Hill, NJ, USA. From 1990 to 1996, he was a member of Research Staff, Xerox Palo Alto Research Center, Palo Alto, CA, USA. In 1997, he was a Manager of the Compression Group, VXtreme, Mountain View, CA, USA, an Internet video startup before it was acquired by Microsoft. From 1998 to 2016, he was a Principal Researcher with Microsoft Research, Redmond, WA, USA, where he was managing the Communication and Collaboration Systems Research Group from 2004 to 2011. He served as a Consulting Associate Professor with Stanford University from 1994 to 1995, an Affiliate Associate Professor with the University of Washington from 1998 to 2009, and has been an Adjunct Professor with the Chinese University of Hong Kong since 2006. He was with a startup, 8i.com, where he led the effort to compress and communicate volumetric media, popularly known as holograms, for virtual and augmented reality. In 2018, he joined the Google Daydream Group as a Research Scientist.