EDITOR: Mike Potel, potel@wildcrest.com

DEPARTMENT: APPLICATIONS

Saliency Computation for Virtual Cinematography in 360° Videos

Ruofei Du and Amitabh Varshney, *Department of Computer Science, University of Maryland, College Park, MD, 20742, USA*

Recent advances in virtual reality cameras have contributed to a phenomenal growth of 360° videos. Estimating regions likely to attract user attention is critical for efficiently streaming and rendering 360° videos. In this article, we present a simple, novel, GPU-driven pipeline for saliency computation and virtual cinematography in 360° videos using spherical harmonics (SH). We efficiently compute the 360° video saliency through the spectral residual of the SH coefficients between multiple bands at over 60FPS for 4K resolution videos. Further, our interactive computation of spherical saliency can be used for saliency-guided virtual cinematography in 360° videos.

With recent advances in consumer-level virtual reality (VR) head-mounted displays (HMD) and 360° cameras, omnidirectional videos are becoming ubiquitous. These 360° videos are becoming a crucial medium for news reports, live concerts, remote education, and social media. One of the most significant benefits of 360° videos is immersion: users have a sustained illusion of presence in such scenes. Nevertheless, despite the rich omnidirectional visual information, most of the content is out of field of view (FoV) of the HMD as well as human eyes. The binocular vision system of human eyes can only interpret 114° FoV horizontally, and 135° FoV vertically. As a result, over 75% of the 360° videos are not being perceived. Furthermore, as shown in Table 1, almost 90% of pixels are beyond the FoV of the current generation of consumer-level VR HMDs.

Therefore, predicting where humans will look, i.e., saliency detection, has great potential over a wide range of applications, such as

- efficiently streaming 360° videos under constrained network conditions;[1]
- salient object detection in panoramic images and videos;[2]
- information overlay in panoramic images, videos, and for augmented reality displays;[3]
- directing the user's viewpoint to salient objects or automatic navigation and synopsis of the 360° videos.[4]

Saliency of regular images and videos has been well studied thoroughly since Itti *et al.*[5] However, unlike classic images which are stored in rectilinear or gnomonic projections, most of the panoramic videos are stored in equirectangular projections. Consequently, classic saliency may not work for 360° videos due to the following challenges (as further shown in Figure 6):

- *Horizontal clipping* may slice a salient object into two parts on the left and right edges, which may cause a false negative result.
- *Spherical rotation* may distort the nonsalient objects near the north and south poles, which may cause a false positive result.

Our work addresses three interrelated questions: a) how should we formulate the saliency in $\mathbb{SO}(2)$ space[a] with spherical harmonics (SH), b) how should we speed up the computation by discarding the low-frequency information, and c) how should we automatically and smoothly navigate 360° videos with saliency maps?

0272-1716 © 2021 IEEE
Digital Object Identifier 10.1109/MCG.2021.3080320
Date of current version 12 July 2021.

[a]$\mathbb{SO}(2)$ space represents all 2-D rotations of the image sphere surrounding the observer.

TABLE 1. Approximate binocular FoV of human eyes, as well as the current generation of consumer-level HMD.

Visual Medium	Approximate Field of View (FoV)		Ratio Beyond FoV
	Horizontal	Vertical	
Human Eyes	114°	135°	76.25%
HTC Vive, Oculus Rift	85°	95°	87.53%
Samsung Gear VR	75°	85°	90.16%
Google Cardboard	65°	75°	92.48%

To investigate these questions, we present a novel GPU-driven pipeline for saliency computation and virtual cinematography based on SH, as shown in Figure 1. SH is the spherical analog of a 2-D Fourier transform for planar 2-D images and transforms the 360° images into frequency domain in spherical coordinates.

Our model reveals the multiscale saliency maps in the spherical spectral domain and reduces the computational cost by discarding low bands of SH coefficients. From the experimental results, it outperforms the Itti *et al.* model by over 5× to 13× in timing, and runs in real time at over 60 frames per second for 4K videos on present-day personal computer hardware.

COMPUTING THE SPHERICAL HARMONICS COEFFICIENTS

We begin by preprocessing the SH coefficient for representing the 360° videos. Our pipeline pre-computes a set of the Legendre polynomials and SH functions and stores them in the GPU memory. We adopt the highly parallel prefix sum algorithm to integrate feature maps of the downsampled 360° frames as 15 bands of SH coefficients on the GPU.

Evaluating SH Functions

First, we precompute the SH functions at each spherical coordinate (θ, ϕ) of the input panorama of $N \times M$ pixels. Since the values in the feature maps, which are used to define the intensity and color contrast are positive and real, we compute only the real-valued SH functions $Y(\theta, \phi)$ also known as the tesseral SH, as shown in Figure 2 and detailed by Green[10] and Du.[11]

Evaluating SH Coefficients

Next, we extract the feature maps such as the intensity I, red-green (RG) contrast, and blue-yellow (BY) contrast, inspired by Itti *et al.*'s model[5] and the MATLAB package *SaliencyToolbox* by Walther and Koch[12]

$$RG = \frac{r-g}{\max(r,g,b)}, \quad BY = b - \frac{\min(r,g)}{\max(r,g,b)}. \tag{1}$$

For each feature map, we extract its SH coefficients consisting of L^2 values in L bands. In the equirectangular representation of the 360° videos, we assume that each feature $f_{i,j}$ at the coordinate $(i,j), 0 \le i < N, 0 \le j < M$ represents the mean value $f(\theta_{i+0.5}, \phi_{j+0.5})$ at the solid angle $(\theta_{i+0.5}, \phi_{j+0.5})$, where θ_i and ϕ_j represent equally spaced spherical coordinates. Therefore, for the mth element of a specific band l, we evaluate the SH coefficients of the feature map f as

$$\begin{aligned} c_l^m(\theta,\phi) &= \int_{(\theta,\phi)\in S} f(\theta,\phi)\cdot Y_l^m(\theta,\phi)\sin\theta \, \mathrm{d}\theta \, \mathrm{d}\phi \\ &= \frac{2\pi}{M}\sum_{i=1}^{N}\sum_{j=1}^{M} f_{i,j}\cdot Y_l^m(\theta_{i+0.5},\phi_{j+0.5}) \\ &\quad |\cos\theta_{i+1} - \cos\theta_i|. \end{aligned} \tag{2}$$

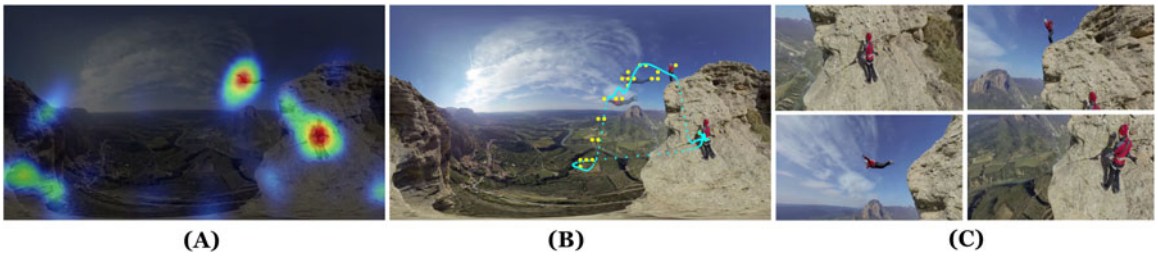

FIGURE 1. (A) shows the saliency map computed by our SSR model in 21.34 ms on a CPU and 10.81 ms on a GPU. (B) shows the optimized camera trajectory in blue. (C) shows four rendering results of virtual cinematography using our shader opensourced at https://shadertoy.com/view/ldBczm.

RELATED WORK

Our work builds upon a rich literature of prior art on saliency detection and spherical harmonics.

Visual Saliency

A region is considered salient if it has perceptual differences from the surrounding areas that are likely to draw visual attention. Prior research has designed bottom-up,[5] top-down,[6] and hybrid models for constructing a saliency map of images (see the review by Zhao *et al.*[7]). The bottom-up models combine low-level image features from multiscale Gaussian pyramids or Fourier spectrum. Top-down models usually use machine learning strategies and take advantage of higher level knowledge such as context or specific tasks for the saliency detection. Recently, hybrid models using convolutional neural networks[8] have emerged to improve the saliency prediction.

One of the most pivotal algorithms for saliency detection remains the Itti *et al.* model.[5] This model computes the center-surround differences of multilevel Gaussian pyramids of the feature maps, which include intensity, color contrast, and orientations, as conspicuity maps. It further combines the conspicuity maps with nonlinear combination methods and a winner-take-all network. Another influential algorithm is the spectral residual approach,[9] which computes the visual saliency by the difference of the original and smoothed log-Fourier spectrum of the image.

However, these and other image saliency approaches assume the input data as rectilinear images, which would not output consistent results for spherical images with horizontal clipping or spherical rotation. Inspired by the Itti *et al.* model[5] and the spectral residual approach,[9] we formulate a SSR model in $\mathbb{SO}(2)$ space. Our model achieves spherical consistency and can be applied to real-time virtual cinematography of 360° videos.

Spherical Harmonics

Spherical harmonics are a complete set of orthogonal functions on the sphere (see Figure 2), and can be used to represent functions defined on the surface of a sphere.[10] In visual computing, spherical harmonics have been widely applied to various domains and applications including, indirect lighting, volume rendering, spatial sound, and 3-D object retrieval. In this article, we use spherical harmonics to efficiently evaluate visual saliency by the difference of the high-frequency and low-frequency spectrum.

Let

$$H_{i,j} = \frac{2\pi}{M} Y_l^m(\theta_{i+0.5}, \phi_{j+0.5}) |\cos\theta_{i+1} - \cos\theta_i| \quad (3)$$

we have

$$c_l^m(\theta, \phi) = \sum_{i=1}^{N} \sum_{j=1}^{M} f_{i,j} \cdot H_{i,j}. \quad (4)$$

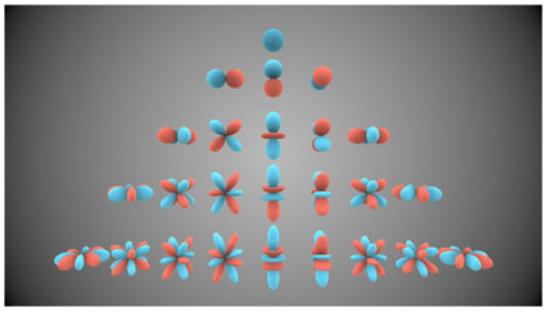

FIGURE 2. The first five bands of SH functions. Blue and red indicate positive and negative values respectively. See https://shadertoy.com/view/4dsyW8 for a live demo built upon Íñigo Quílez's work.

Hence, for a given dimension of the input frames, we can precompute the terms $H(i, j)$ and store them in a lookup table. The integration of the SH coefficients is then reduced to a conventional prefix sum problem.

Implementation Details

On the CPU-driven pipeline, we use *OpenMP* to accelerate the evaluation of SH coefficients with 12 threads. On the GPU-driven pipeline, we take advantage of the *Blelloch Scan* algorithm[13] with *CUDA* to efficiently aggregate the SH coefficients with 2048 kernels on an NVIDIA GTX 1080. The *Blelloch Scan* algorithm computes the cumulative sum in $O(\log N)$ for N numbers. Therefore, our algorithm runs at $O(L^2 \log MN)$ for L^2 coefficients.

Finally, we show the reconstructed image f' with 1–15 frequency bands of SH coefficients with regular RGB color maps in Figure 3.

Note that the low-band SH coefficients capture the background information, such as sky and mountains, while the high-band SH coefficients capture the details, such as parachuters.

FIGURE 3. Reconstructed images using the first 15 frequency bands of SH coefficients extracted from the video frame.

SSR MODEL

Inspired by the spectral residual approach,[9] we define SSR as the accumulation of the SH coefficients between a low-frequency band and a high-frequency band. This model reveals the multiscale saliency maps in the spherical spectral domain and reduces the computational cost by discarding the low bands of SH coefficients.

SSR Approach

As shown in Figure 3, SH frequency bands can be used to compute the contrast directly across multiple scales in the frequency space. We define the SSR as the sum of the frequency bands between P and Q

$$\mathfrak{R}(\theta,\phi) = \sum_{l=P+1}^{Q}\sum_{m=-l}^{l} c_l^m \cdot Y_l^m(\theta,\phi). \quad (5)$$

Here, $Y_l^m(\phi,\theta)$ are precomputed associated Legendre polynomials in the preprocessing stage. The SSR represents the salient part of the scene in the spectral domain and serves as a compressed representation using SH.

For better visual effects, we square the spectral residual to reduce estimation errors and smooth the spherical saliency maps using a Gaussian

$$\mathbf{S}(\theta,\phi) = \mathfrak{G}(\sigma) * [\mathfrak{R}(\theta,\phi)]^2 \quad (6)$$

where $\mathfrak{G}(\sigma)$ is a Gaussian filter with standard deviation σ (we empirically take $\sigma = 5$).

We show the SSR results of the intensity channel with all pairs of the lower band P and the higher band Q in Figure 4. As P increases, the low-frequency information such as the sky and mountains are filtered out. The SSR results within and close to the orange bounding box reveal the salient objects, such as the two people.

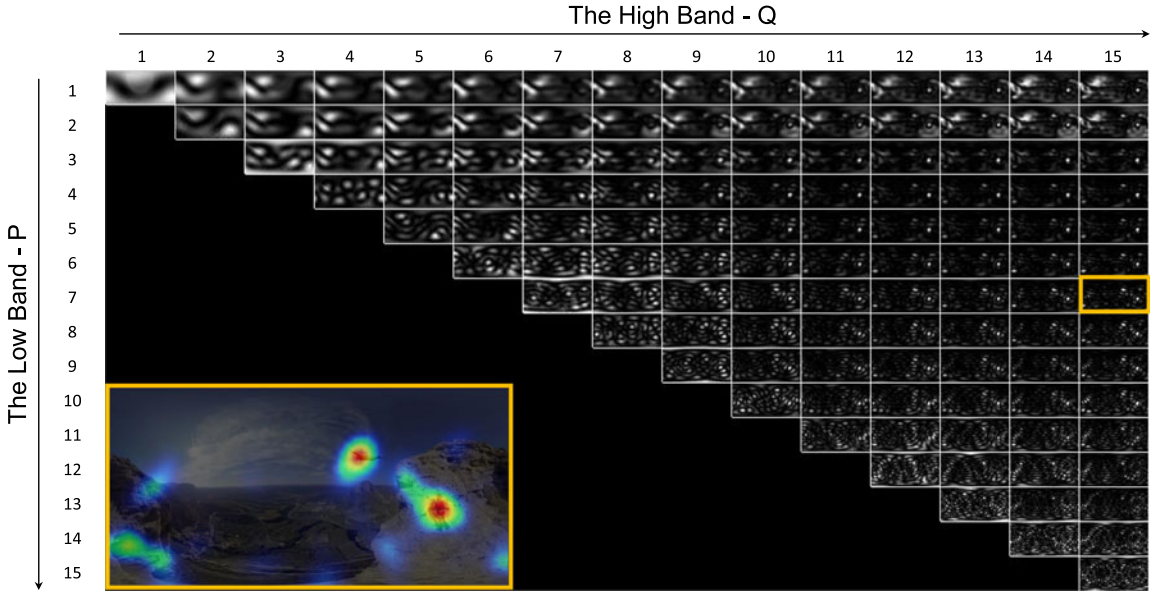

FIGURE 4. Spectral residual maps between different frequency bands of SH. The number along the horizontal axis indicates the high band Q, while the vertical axis indicates the low band P. Note that the saliency maps within or close to the orange box successfully detect the two people in the frame (lower left).

Temporal Saliency

In addition to intensity and color features, we further extract temporal saliency in the SH domain.

For the SH coefficients extracted from the three feature maps, we maintain two sliding temporal windows to estimate temporal contrast. The smaller window w_0 stores the more recent SH coefficients from the feature maps, and the larger window w_1 stores the SH coefficients over a longer term. For each frame, we calculate the estimated SH coefficients $\bar{c}_l^m, \bar{\bar{c}}_l^m$ from both windows, using two probability density functions from the Gaussian distribution ($|w_0| = 5, |w_1| = 25, \sigma = 7.0$). We use a formulation similar to the spatial saliency to measure the SSR between two temporal windows

$$\mathfrak{R}(F_{\text{temporal}}, \theta, \phi) = \left| \sum_{l=P+1}^{Q} \sum_{m=-l}^{l} \left(\bar{c}_l^m(\theta, \phi) - \bar{\bar{c}}_l^m(\theta, \phi)\right) \cdot Y_l^m(\theta, \phi) \right|. \tag{7}$$

We again apply (6) to compute the smoothed temporal saliency maps.

Saliency Maps With Nonlinear Normalization

Following Itti *et al.*,[5] we apply the nonlinear normalization operator $\mathcal{N}(\cdot)$ to all six saliency maps: intensity, red-green, and blue-yellow contrasts, both statically and temporally. This operator globally promotes maps, which contain a few peak responses and suppresses maps with a large number of peaks. After the nonlinear normalization, we linearly combine all saliency maps into the final saliency map

$$\mathcal{S} = \frac{1}{N} \bigoplus_{i=1}^{N} \mathcal{N}(\mathbf{S}(F_i)). \tag{8}$$

Empirically, we choose $Q = 15, P = 7$. The final composed result is shown at the bottom left corner in Figure 4, as well as in the accompanying video: https://youtu.be/jwv5hjlg-MI.

Comparison Between Itti *et al.* and SSR Model

As shown in Figure 6, our SSR model is visually better than the Itti *et al.* model. In addition, our experimental results below compare the classic Itti *et al.* model and our model.

We use six videos from the Insta360[b] and the 360Rize.[c] The video resolutions vary from 1920 × 1080 to 7680 × 3840 pixels.

[b]Insta360: https://www.insta360.com
[c]360 Rize: http://www.360rize.com

TABLE 2. Timing comparison between the Itti *et al.* model and our SSR model.

Resolution	Average Timing Per Frame		
	Itti *et al.* (CPU)	SSR (CPU)	SSR (GPU)
1920×1080	104.46 ms	21.34 ms	10.81 ms
4096×2048	314.94 ms	48.18 ms	13.20 ms
7680×3840	934.26 ms	69.53 ms	26.58 ms

Our results are obtained on a workstation with an NVIDIA GTX 1080 and an Intel Xeon E5-2667 2.90 GHz CPU with 32 GB RAM. Both the Itti *et al.* model and the SSR model are implemented in C++ and OpenCV. The GPU version of the SSR model is developed using CUDA 8.0. We measure the average timing of saliency computation as well as the visual results between the Itti *et al.* model and our SSR model. Note that the timings do not include the uploading time for each frame from system memory to GPU memory. We expect our algorithms would map well to products such as NVIDIA DrivePX[d] in which videos are directly loaded onto the GPU memory.

We measure the average computational cost of the initial 600 frames across three resolutions: 1920 × 1080, 4096 × 2048, and 7680 × 3840, as shown in Table 2. All frames are preloaded into the CPU memory to eliminate the I/O overhead. Both the CPU and GPU versions of our SSR model outperform the classic Itti *et al.* model, with the speedups ranging from 4.8× to 13.4 × , depending on various resolutions. We show example input and the output from both models in Figure 5.

SALIENCY-GUIDED VIRTUAL CINEMATOGRAPHY

We now present a saliency-guided virtual cinematography system for navigating 360° videos. Inspired by prior art on camera path selection and interpolation,[4,14] we formulate a spatiotemporal model to ensure large saliency coverage while reducing the camera movement jitter.

We compute our saliency maps by linearly combining the saliency maps based on intensity, color, and motion, and then performing a nonlinear normalization, as explained in the previous section. However, for 360° videos, the most salient objects may vary from frame to frame, due to the varying

[d]https://NVIDIA.com/en-us/self-driving-cars/drive-px

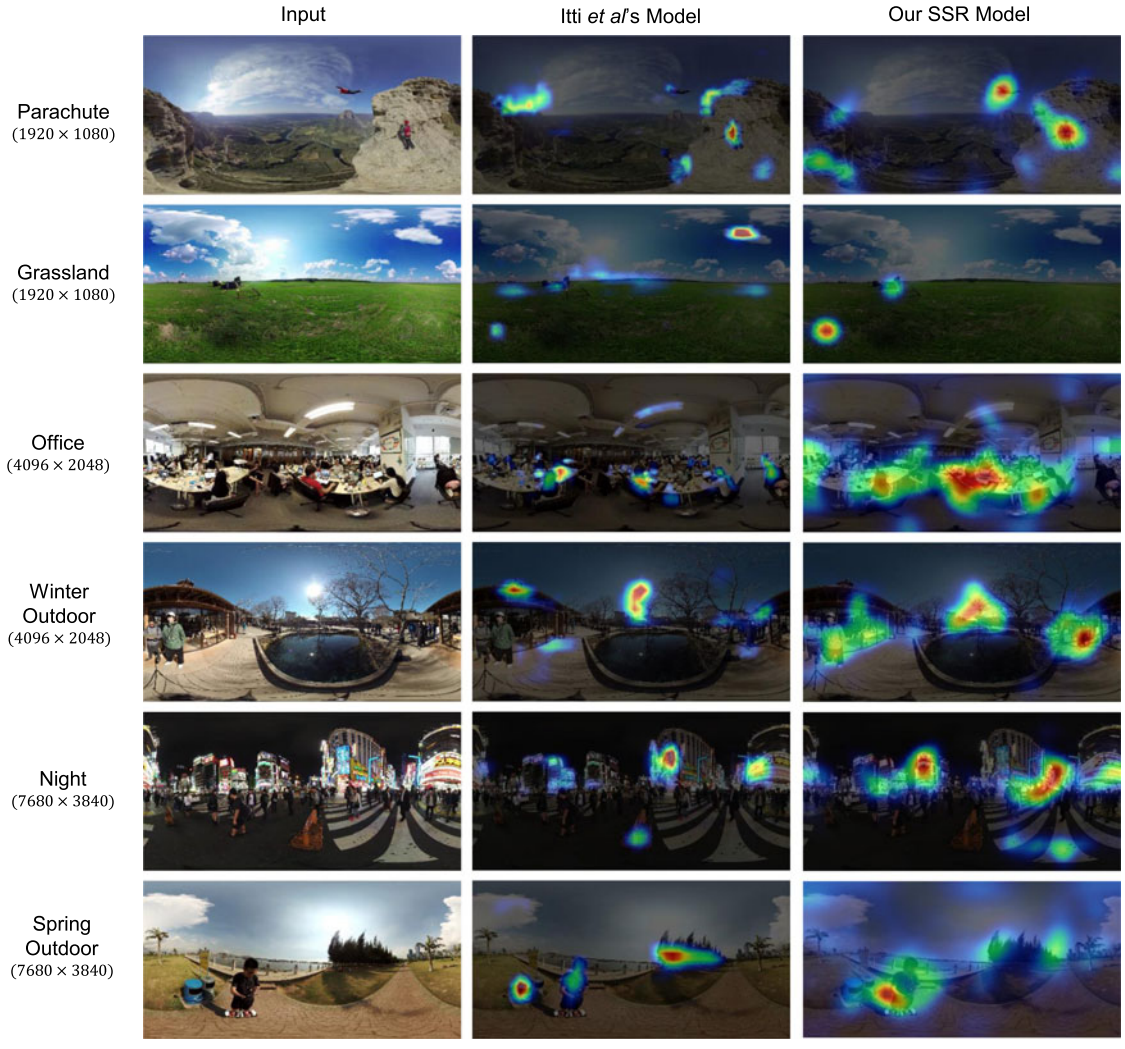

FIGURE 5. Visual comparison between the Itti *et al.* model and our SSR model. Note that while the results are visually similar, our SSR model is 5× to 13× faster than the Itti *et al.* model.

occlusions, colors, and self-movement. As a result, an approach that relies on just tracking the most salient objects may incur rapid motion of the virtual camera, and worse still, may induce motion sickness in virtual reality. Hence, we devise a spatio-temporal optimization model of the virtual camera's discrete control points and further employ a spline interpolation among the control points to achieve smooth camera navigation.

Optimization of the Camera's Control Points

To estimate the virtual camera's control points, we formulate an energy function $\mathbf{E}(C)$ in terms of camera location $C = (\theta, \phi)$. The energy function

$$\mathbf{E}(C) = \lambda \mathbf{E}_{\text{saliency}}(C) + \mathbf{E}_{\text{temporal}}(C) \tag{9}$$

consists of a saliency coverage term $\mathbf{E}_{\text{saliency}}$ and a temporal motion term $\mathbf{E}_{\text{temporal}}$, thus taking both saliency coverage and temporal smoothness into consideration. Empirically, we assign $\lambda = 2$.

Saliency Coverage Term

This spatial term $\mathbf{E}_{\text{saliency}}$ penalizes the coverage of the saliency values beyond the FoV. As for a specific virtual camera location C, this term would be written as

$$\mathbf{E}_{\text{saliency}}(C) = \frac{\sum_{\theta,\phi} \mathbf{S}(\theta,\phi) \cdot \mathbf{O}(C,\theta,\phi)}{\sum_{\theta,\phi} \mathbf{S}(\theta,\phi)} \tag{10}$$

where $\mathbf{O}(C,\phi,\theta)$ indicates whether an arbitrary spherical point (ϕ,θ) is observed by the virtual camera

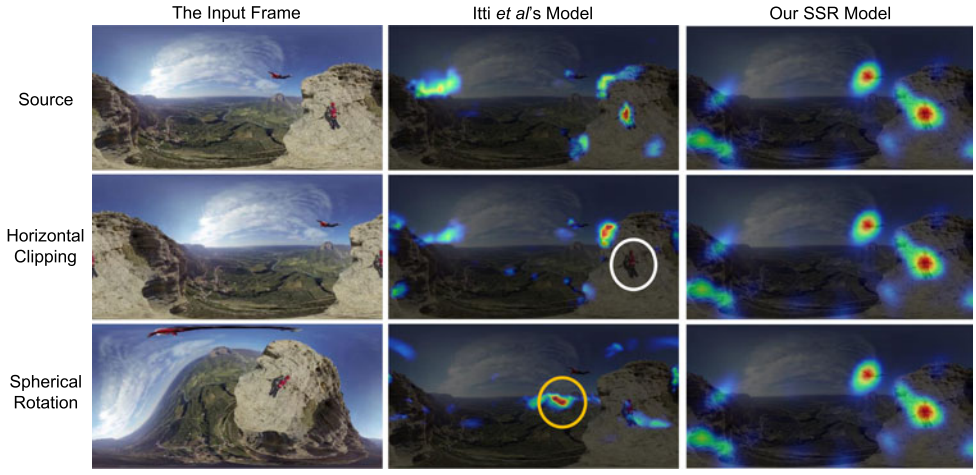

FIGURE 6. Comparison between the Itti *et al.* model and our SSR model with horizontal translation and spherical rotation in the 360° video frame. White circles indicate the false negative result from Saliency Toolbox and orange ones indicate false positive result from Saliency Toolbox. Meanwhile, the results from our SSR model remain consistent, regardless of horizontal clipping and spherical rotation.

centered at the location C_i

$$\mathbf{O}(C,\theta,\phi) = \begin{cases} 1, & (\theta,\phi) \text{ is observed by camera at } C \\ 0, & \text{otherwise.} \end{cases} \tag{11}$$

Thus, $\mathbf{E}_{\text{saliency}}(C)$ measures the coverage of the saliency values beyond the FoV of the virtual camera centered at C. To reduce the computation, we compute the saliency coverage term over 2048 points (θ,ϕ), which are uniformly distributed over the sphere.

Temporal Motion Term

For the ith frame in the sequence of the discrete control points, $\mathbf{E}_{\text{temporal}}(C)$ measures the temporal motion of the virtual camera as follows:

$$\mathbf{E}_{\text{temporal}}(C) = \begin{cases} \|C_{i-1}, C_i\|_2, & i \geq 1 \\ 0 & , i = 0. \end{cases} \tag{12}$$

The Optimization Process

Based on this spatiotemporal model, we evaluate the energy functions over 32×64 pairs of discrete (θ,ϕ). This process is highly parallel, and can be efficiently implemented on the GPU. For each frame, we compute the optimal camera point as follows:

$$C = \underset{C}{\operatorname{argmin}}\, \mathbf{E}(C). \tag{13}$$

In this way, we extract a subsequence of discrete spherical coordinates $Seq = \{C_i | C_i = (\phi_i, \theta_i)\}$ of the optimal camera location in the saliency maps every K frames, $K = 5$ in our examples. Since these locations are discrete and sampled at a lower frame rate, we further perform spline interpolation with C^2 continuity.

Interpolation of Quaternions

To interpolate between 3-D rotations of the surrounding sphere over time through our calculated salient positions, we convert the spherical coordinates to quaternions:

$$Q(\theta,\phi) = (0,\ \sin(\theta)\cos(\phi),\ \sin(\theta)\sin(\phi),\ \cos(\theta)). \tag{14}$$

We then use spherical spline curves with C^2 continuity to compute the smooth trajectory of the camera

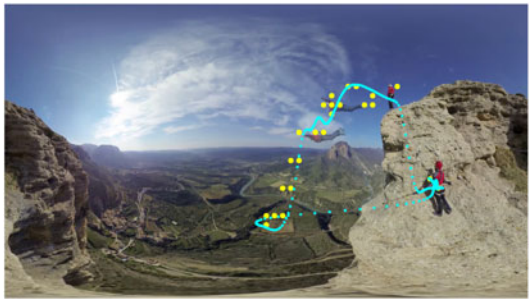

FIGURE 7. Interpolation among the global maximas of the saliency maps in the spherical space. The yellow dots show the discrete optimal locations using the energy function, and the blue dots show the interpolation using the spherical spline curve.

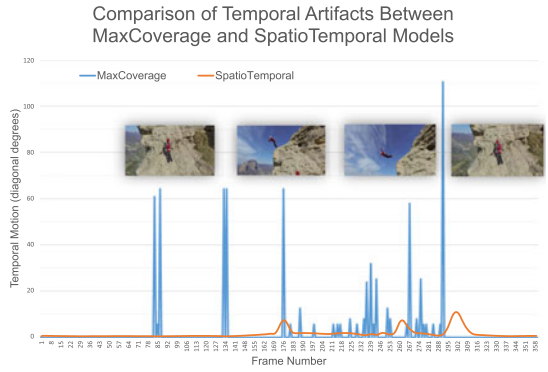

FIGURE 8. Quantitative comparison between the MaxCoverage model and the SpatioTemporal Optimization (STO) model. Compared with the MaxCoverage model, the STO model significantly reduces the temporal jitters.

cruise path over the quaternions. Figure 7 shows the locations of the global maximas, as well as the interpolated spline path over the sphere. An alternative method is to use spherical linear interpolation[e] for the interpolation.

Evaluation of Virtual Cinematography Models

We compare our method, denoted as SpatioTemporal Model (STO) with a MaxCoverage model, which determines the camera position for the maximal coverage of the saliency map. We evaluate the temporal motion terms for the same video sequence and plot the data in Figure 8.

From the quantitative evaluation, as well as the accompanying https://youtu.be/jwv5hjlg-MIvideo, we have validated that the STO model reduces the temporal jittering of the camera motion compared to MaxCoverage model for virtual cinematography in 360° videos.

FUTURE DIRECTIONS

With new datasets of stereoscopic 360° videos and eye-tracking data, one may extend our model and optimize the foveated streaming[1] and automatic camera navigation. We believe our spherical representation of saliency maps will inspire more research to think out of the rectilinear space. We envision our techniques will be widely used for live streaming of events, video surveillance of public areas,[11] as well as templates for directing the camera path for immersive storytelling. Future research may explore how to naturally place 3-D objects with SH irradiance in 360° videos, how to employ SH for foveated rendering in 360° videos, and the potential of compressing and streaming 360° videos with SH.

[e]https://boost.org/doc/libs/1_67_0/libs/qvm/doc/slerp.html

REFERENCES

1. D. Li, R. Du, A. Babu, C. D. Brumar, and A. Varshney, "A log-rectilinear transformation for foveated 360-degree video streaming," *IEEE Trans. Vis. Comput. Graph.*, vol. 27, no. 5, pp. 2638–2647, May 2021.
2. V. Sitzmann *et al.*, "Saliency in VR: How do people explore virtual environments?," *IEEE Trans. Vis. Comput. Graph.*, vol. 24, no. 4, pp. 1633–1642, Apr. 2018.
3. R. Du *et al.*, "Geollery: A. mixed reality social media platform," in *Proc. CHI Conf. Human Factors Comput. Syst.*, no. 685, 2019, pp. 1–13.
4. Y.-C. Su *et al.*, "Pano2Vid: Automatic cinematography for watching 360 videos," in *Proc. Asian Conf. Comput. Vis.*, 2016, pp. 154–171.
5. L. Itti, C. Koch, and E. Niebur, "A model of saliency-based visual attention for rapid scene analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 11, pp. 1254–1259, Nov. 1998.
6. S. Goferman, L. Zelnik-Manor, and A. Tal, "Context-aware saliency detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 10, pp. 1915–1926, Oct. 2012.
7. Q. Zhao and C. Koch, "Learning saliency-based visual attention: A review," *Signal Process.*, vol. 93, no. 6, pp. 1401–1407, 2013.
8. G. Li and Y. Yu, "Visual saliency based on multiscale deep features," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 5455–5463.
9. X. Hou and L. Zhang, "Saliency detection: A spectral residual approach," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2007, pp. 1–8.
10. R. Green, "Spherical harmonic lighting: The gritty details," in *Proc. Archives Game Developers Conf.*, vol. 56, 2003, Art. no. 47.
11. R. Du, "Fusing multimedia data into dynamic virtual environments," Ph.D. dissertation, Univ. Maryland, College Park., MD, USA, Nov. 2018.
12. D. Walther and C. Koch, "Modeling attention to salient proto-objects," *Neural Netw.*, vol. 19, no. 9, pp. 1395–1407, 2006.
13. G. E. Blelloch, "Scans as primitive parallel operations," *IEEE Trans. Comput.*, vol. 38, no. 11, pp. 1526–1538, Nov. 1989.
14. T. Oskam *et al.* "OSCAM-optimized stereoscopic camera control for interactive 3D," *ACM Trans. Graph.*, vol. 30, no. 6, pp. 1–8, 2011.

RUOFEI DU is currently a Senior Research Scientist at Google. He is the corresponding author of this article. Contact him at me@duruofei.com.

AMITABH VARSHNEY is the Dean of the College of Computer, Mathematical and Natural Sciences, and a Professor of Computer Science, University of Maryland at College Park. Contact him at varshney@cs.umd.edu.

Contact department editor Mike Potel at potel@wildcrest.com.