View-Dependent Effects for 360° Virtual Reality Video

Jeremy Hartmann[1], **Stephen DiVerdi**[2], **Cuong Nguyen**[2], **Daniel Vogel**[1]
[1]School of Computer Science, University of Waterloo, [2]Adobe Research
{j3hartma, dvogel}@uwaterloo.ca, {diverdi, cunguyen}@adobe.com

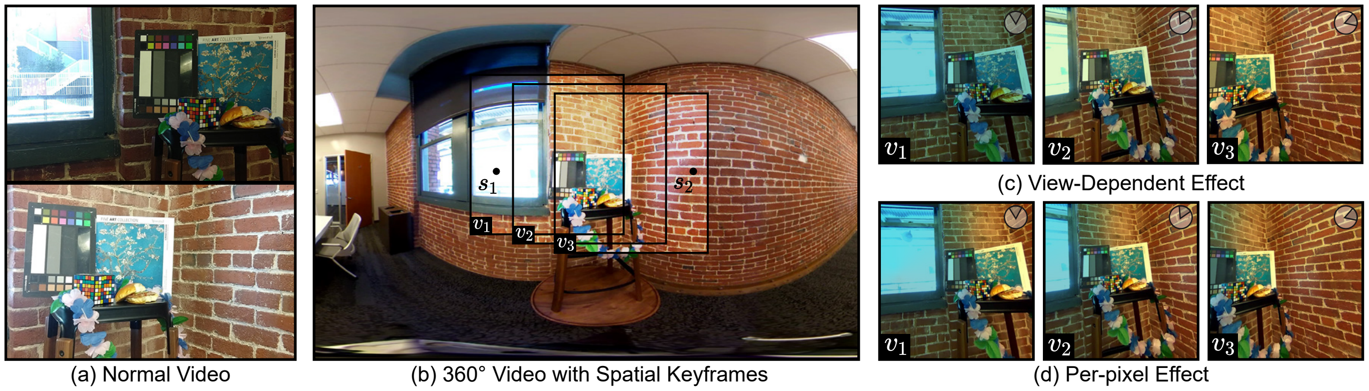

s_1
s_2
v_1
v_2
v_3
(a) Normal Video
(b) 360° Video with Spatial Keyframes
(c) View-Dependent Effect
(d) Per-pixel Effect

Figure 1. View-dependent effect concept: (a) two frames of a normal video sequence shows how camera exposure adaptation and perceived colour constancy affects the appearance of scene elements depending on what is in frame, such as a window (images captured from a cellphone camera); (b) a 360° video of the same scene with two spatial keyframes s_1 and s_2 to apply a blue and yellow tint; during playback, the user turns their head to view the scene as v_1, v_2, and v_3; (c) dynamic view-dependent effects rendered at each view mimic normal video exposure and colour constancy behaviour (video stills captured from our prototype video player); (d) static per-pixel effects result in an obvious gradient across the scene because it is rendered before playback independent of user view.

ABSTRACT
"View-dependent effects" have parameters that change with the user's view and are rendered dynamically at runtime. They can be used to simulate physical phenomena such as exposure adaptation, as well as for dramatic purposes such as vignettes. We present a technique for adding view-dependent effects to 360° video, by interpolating spatial keyframes across an equirectangular video to control effect parameters during playback. An in-headset authoring tool is used to configure effect parameters and set keyframe positions. We evaluate the utility of view-dependent effects with expert 360° filmmakers and the perception of the effects with a general audience. Results show that experts find view-dependent effects desirable for their creative purposes and that these effects can evoke novel experiences in an audience.

Author Keywords
360° video; virtual reality; visual effects rendering; cinematography, view-dependent effects

CCS Concepts
•Computing methodologies → Image-based rendering; Perception; Virtual reality; •Human-centered computing → User studies;

INTRODUCTION
A "view-dependent effect" is when the visual appearance of scene elements change dynamically as the viewpoint of the camera or user changes. Many view-dependent effects occur naturally in the physical world, such as exposure adaptation, colour constancy, and glare, and so reproducing them is an important aspect of achieving realism in video media.

In traditional film, the view is determined by a cinematographer using a physical camera. View-dependent effects such as exposure adaptation (Fig. 1a) and lens flare occur in the camera during capture, and filmmakers often add other effects such as tone mapping and vignettes for creative purposes. However, the "view" is predetermined, allowing filmmakers to render all effects using standard post-production software. This process yields a single static colour for each video pixel, which we call a "per-pixel effect."

In video games and virtual reality (VR), visual effects can depend on the player's view, such as simulating natural phenomena [12,21] or applying artistic stylizations like vignettes, lens dirtying, or glare (Fig. 2). However, in 360° video, there is no authoring or playback component in which to encode or

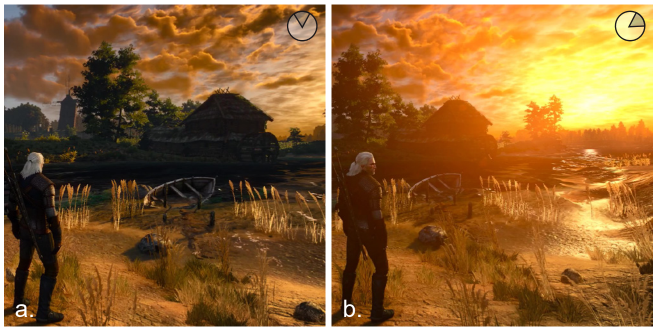


Figure 2. An example of a view-dependent effect from *Witcher 3: Wild Hunt*: when the user changes their view to bring the sun from outside the frame (a) to inside the frame (b), a glare effect changes the appearance of the other scene elements, increasing realism and immersion (images ©CD PROJEKT RED, used with permission).

render these effects, restricting filmmakers to per-pixel effects regardless of the user's view during playback.

We contribute a technique for adding view-dependent effects to 360° video, by interpolating effect parameters stored in *spatial keyframes* throughout the video (Fig 1b). Rendering these view-dependent effects during playback allows us to reproduce the varying appearances of scene elements based on the view (Fig. 1c). This enables the simulation of effects seen in the real world, film, and video games, which is not possible in existing video editing tools that can only encode a single effect appearance per-pixel (Fig. 1d). Since 360° films are optimally edited in-headset for fast feedback and representative colour perception, we developed an in-headset authoring tool using a performance-based interface for placing spatial keyframes across time and space. Our approach was informed by discussions with three leading VR and 360° film studios, we validate our techniques with an expert review study, and an experiment shows general audiences can perceive a difference with view-dependent effects.

RELATED WORK

We consider previous work in three areas: 360° video, dynamic effects in 3D games and VR, and using spatial keyframes for other applications.

360° Video Effects

Previous dynamic effects methods for 360° have not been general purpose and focused on specific non-aesthetic use cases. For example, since gaze guidance is an important part of 360° video production [26], Danieau et al. [6] explore how exposure, saturation, and blur can be used to guide a user's view towards a point of interest. In another special-purpose application to reduce motion sickness in 360° video, Fernandes and Feiner [9] use a simple function of speed and angular velocity to apply a vignette, restricting the user's view. These techniques are applied automatically using heuristics and lack specific control over the final appearance, diminishing the filmmakers' creative direction.

Most related to our work is that of Pouli et al. [31], who proposed colour grading methods for 360° panoramic imagery. Colour grades are applied to a discrete set of views, which are then interpolated across a single image using a spherical weight map. Since the method is applied in post-production, only a single colour effect is applied at each pixel, so the result is not view-dependent and cannot reproduce our target phenomena (Figs. 1a and 2).

Dynamic Effects in 3D Games and VR

Real-time application of "filmic" colour grading is often applied to 3D video games [15] relying on GPU shaders for efficient computation [37]. Game engines such as Unreal support post-processing effects such as adaptation, bloom, depth of field, and lens flare [7]. Other work simulates motion blur [12] and lens effects [21] for interactive experiences. Rather than explicitly authoring these phenomena, they can also be computationally simulated, such as auto exposure and tone mapping [20, 34]. Mantiuk [23] adds gaze-dependence by tracking the users' eyes to provide more accurate tone mapping.

While these dynamic effects are used in video games and interactive VR experiences, they are not available for 360° videos. Our work enables filmmakers to author dynamic effects for 360° videos by controlling them via the user's view during playback.

Spatial Keyframes

Our technical solution uses an expanded version of "spatial keyframes", first introduced by Igarashi et al. [17]. Typical keyframes in video-editing tools are parameterized by one dimension, time, to interpolate one (or more) dimensional effect parameters [40]. Igarashi et al. proposed spatial keyframes to instead parameterize three spatial dimensions to interpolate among model poses determined by 3D user input. Prior to this, Rose et al. [35] performed motion interpolation of poses using radial B-splines, and more recently Noh et al. [27] deformed smooth surfaces by interpolating 3D samples using radial basis functions [32].

A related concept of spatial annotations are also used in mixed reality. Kolbe [19] uses them to annotate outdoor environments for view-stabilized navigation aids, while Sun et al. [41] uses them to place videos and metadata around a museum. Related to annotations, a type of spatial keyframes without interpolation have also been used to mark points of interest in 360° video for gaze guidance [22, 30].

Our keyframe method is distinct from prior works because we parameterize them by both time and two user-controlled spatial dimensions (for the view direction).

VIEW-DEPENDENT EFFECTS

To motivate and ground our work, we interviewed film production experts from three professional 360° film studios. Only one studio used dynamic effects that could be considered view-dependent, because only that studio had the resources for a custom realtime rendering engine to deliver content. However, all interviewees expressed interest in the potential of view-dependent effects. One suggested an application for gaze guidance in horror type videos, another suggested mimicking human vision characteristics, and all commented on how view-dependent effects could expand their creative options.

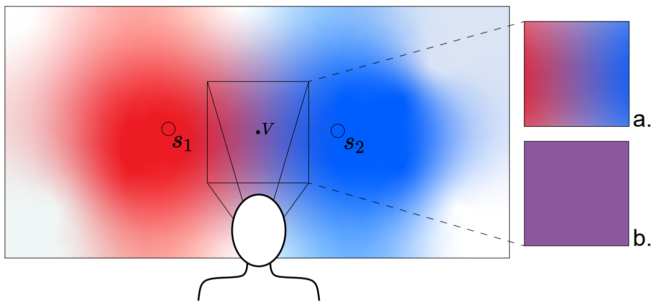


Figure 3. Illustration of a view-dependent effect. *Left:* A 360° equirect-angular frame with two spatial keyframes s_1, red, and s_2, blue. A user views the scene centered at direction v. *Inset a:* A per-pixel effect interpolates s_1 and s_2 at each pixel and applies the resulting colours, yielding a visual gradient across the view. *Inset b:* A view-dependent effect interpolates the colour at v and applies that single colour to the entire view.

With this in mind, we provide an expanded explanation of view dependent effects leading to a formal definition. An illustrative natural phenomenon that can be reproduced by a view-dependent effect is the appearance of scene objects in a mixed lighting environment (Fig. 1a). When the view includes the window (Fig. 1a top), the sunlight darkens and cools the scene objects, but when the camera includes the brick wall (Fig. 1a bottom), the same scene objects are brighter and warmer because of the indoor lighting. The camera sensor mimics the human visual system's adaptation to changing lighting conditions [29]. Video editing software [2] allows editing colour and tone to control this effect, and video game engines [7] provide similar support at runtime.

As noted in our expert interviews, for standard 360° video, no single effect can be applied that will reproduce this experience, as only a single colour can be chosen per pixel of the equirect-angular frame. Interpolating effects across the frame [31] can create unnatural gradients across the visual field (Figs. 1d and 3). We overcome this problem by parameterizing effects based on the users' view direction, and applying the resulting effect parameters to the current view. This allows us to create multiple visual appearances for each scene element and yields experiences currently only possible within traditional movies (Fig. 1a) and video games (Fig. 2).

We define "view-dependent" effects as being (1) parameterized by the user's view direction, typically determined by a VR headset, and (2) rendered dynamically during playback. Conversely, "per-pixel" effects are rendered statically without knowledge of the user's view before playback. In the rest of this section, we detail how we interpolate and render view-dependent effects, and we provide application examples for effects enabled by our method.

Spatial Keyframes

We define a spatial keyframe as an extension of the traditional video editing concept of keyframes, which have a 1D temporal location within the video. Our spatial keyframes have a 3D location: one temporal dimension (the video timecode, t) and two spatial dimensions, the azmuth (ϕ) and altitude (θ) on the view sphere. A 3D coordinate in this space is represented as $[\phi, \theta, t]$, where the spatial and temporal dimensions are unified into a shared space—the temporal dimension is treated as a third spatial dimension. Note that while we re-use the spatial keyframe term from Igarashi et al. [17], they are distinct; Igarashi et al.'s spatial keyframes have three spatial dimensions only, and are not applied to video.

A spatial keyframe in this space associates its coordinate with a set of 1 to N parameters for an associated effect (e.g. a colour filter has one parameter for each colour channel). Each effect is associated with its own set of spatial keyframes for which interpolation happens, allowing each effect to be layered on the video independently.

Spatial Interpolation

Sparse multi-dimensional interpolation has been examined thoroughly [5, 32, 38]. We use radial basis functions (RBF) for implicit surfaces described by Turk and O'Brien [43]. Before choosing this approach, we experimented with a number of different interpolation schemes, barycentric coordinates, uniform grids, separate treatment of spatial and temporal dimensions, and explored the difference between localized and global effects. We found that RBF provides the most predictable results when users made changes (adding or removing keyframes) under the broadest circumstances. For example, a lens-flare fading in and out quickly can be achieved with RBFs by placing two spatial key frames near one another. RBFs also vary smoothly, are fast to compute, and can be tuned easily to our needs. Prior work used RBFs for similar reasons [17, 27, 28].

We consider each spatial keyframe to contain a sample of a real-valued function on the control space. The RBF kernel is

$$\varphi(x) = |x^{\alpha}| \tag{1}$$

where α is a user controlled value that effects how aggressive the gradient is between control points are (default is 1), x is the magnitude of the control point vector, and $\varphi(x)$ is the resulting weight.

The distance between two spatial keyframes is not obvious as it requires mixing spherical and temporal coordinates. The distance between two spherical coordinates, transformed to points on the unit sphere ($\mathbf{n}$), is computed using the geodesic distance with corresponding vectors $\mathbf{n}_v$ (the user's view direction) and $\mathbf{n}_k$ (the location of the spatial keyframe k), as:

$$\Delta\sigma = \arccos(\mathbf{n}_v \cdot \mathbf{n}_k). \tag{2}$$

The normalized temporal distance between two frames, t_v (the frame the user is currently on) and t_k (the frame associated with the spatial keyframe k), is:

$$\Delta t = |t_v - t_k| \frac{24}{t_f} c \tag{3}$$

where $24/t_f$ converts the current frame rate to a standard 24 frames per second (FPS) video rate and c is a user controlled value that has the effect of compressing or expanding the temporal dimension (default is 1). These two distances are combined to produce their magnitude using an ℓ^2-norm:

$$\|x_v - x_k\|_2 = \sqrt{(\Delta\sigma)^2 + (\Delta t)^2} \tag{4}$$

where x_v and x_k are points in our video coordinate space.

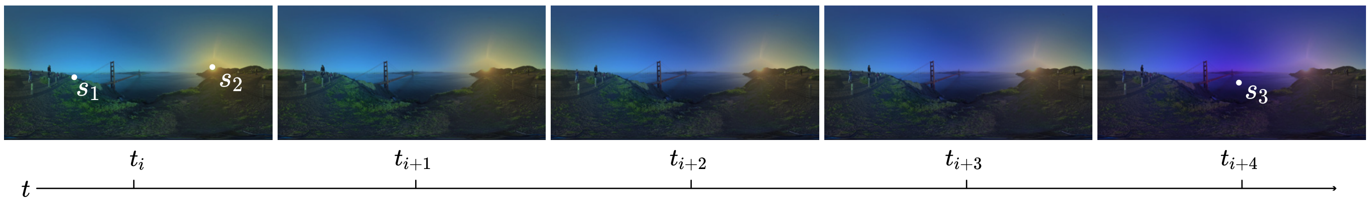


Figure 4. Visualization of parameter interpolation. Spatial keyframes s_1 (cyan) and s_2 (yellow) at time t_i and s_3 (purple) at time t_{i+4}, interpolate both spatially and temporally using radial basis functions. Note this does not represent the in-headset viewing experience because any single pixel will have different parameters applied to it depending on the users' view.

To interpolate a set of spatial keyframes Q, where a spatial keyframe $q \in Q$ contains both spatial and temporal locations in the video, $q^x = [\phi, \theta, t]$, and a set of target effect parameter values, $q^{\mathbb{P}} = [p, p_1, \ldots, p_n]$, for a video coordinate x_v, we write the interpolation function as:

$$f(x_v) = \sum_{k=1}^{N} w_k \varphi(\|x_v - q_k^x\|_2) + P(x_v) \tag{5}$$

Equation 5 is effectively a weighted sum over the calculated keyframe magnitudes produced by the RBF kernel in equation 1, where the weights (w_k) are determined by solving a system of linear equations using the control points q^x and associated parameters $q^{\mathbb{P}}$ with the constraint that $\forall q \in Q : f(q^x) = q^{\mathbb{P}}$. We defer to previous work for details on how to construct and solve the system of linear equations [43]. The interpolated parameter values are then used to render the dynamic effect for the user's current viewing direction (Fig. 4).

Effect Compositing

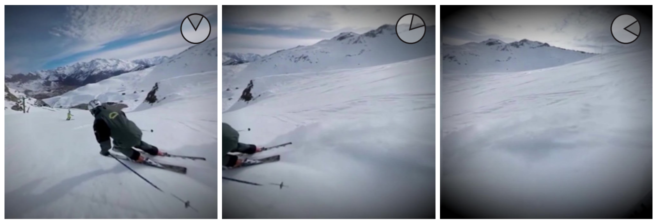
After spatial interpolation, the parameters computed by Eq. 5 are applied to the user's current view in realtime in a post-processing fragment shader. Our framework allows any type of post-processing effect that can be parameterized for viewing direction to be applied to the headset's rendered field of view during playback. For our prototype implementation, we support six effects: blur, tint, exposure, saturation, contrast, and vignette. These are applied in that order in our shader, with the output of each effect becoming input to the next. We use standard filmic effect formulations [13] (details in Appendix).

360° VR Video Player

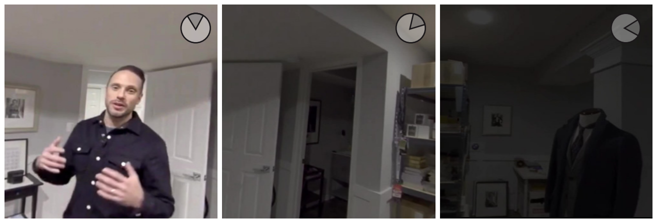
Using the effects detailed above, we built a video player for Windows 10 computers with desktop HMDs (Oculus Rift, HTC Vive, and WinMR) and for iOS devices using Google Cardboard. Our player can process local or online streaming videos in standard formats, and can render view-dependent effects stored as sets of spatial keyframes in a companion sidecar file. Spatial keyframes are interpolated dynamically based on the current view and the effects are composited with the 360° video.

Realtime Rendering Performance

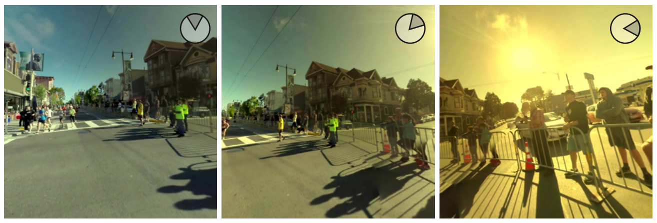
Consumer HMDs refresh their displays at between 72Hz (Oculus Quest) and 120Hz (Valve Index), with the most common being 90Hz. We measure our performance with the SteamVR frame timing monitor on our test machine, an Oculus Rift HMD and an MSI GT73VR Titan laptop with an Intel Core i7-6820HK CPU and NVIDIA GTX 1080 GPU running Windows 10. Our player renders each frame in 3.22ms ($\sigma = 1.63$ms), of which, spatial keyframe interpolation of 1000 keyframes is 2.74ms; this is well within our rendering budget.



(a) View-dependent vignette for motion sickness reduction



(b) View-dependent desaturation for gaze guidance



(c) View-dependent bloom for greater immersion

Figure 5. Example view-dependent effects: (a) a vignette is added to reduce motion sickness in a fast motion video sequence [1]; (b) desaturation and darkening for gaze guidance is added to a video with a point of interest [14]; (c) exposure and colour filters create a bloom effect around the sun. ***All image stills from our prototype 360° VR video player.***

Applications

To demonstrate the utility of view-dependent effects for 360° film, we describe a small selection of significant use cases (though many more are possible):

Motion Sickness — A number of researchers have addressed VR motion sickness caused by 360° video with large, fast translational motions [9, 18, 25]. The common solution is to analyze the HMD view using heuristics like optical flow, and apply a vignette to restrict the field of view. Since vignetting is a screen-space effect that can be parameterized by view direction, our framework enables filmmakers to author their

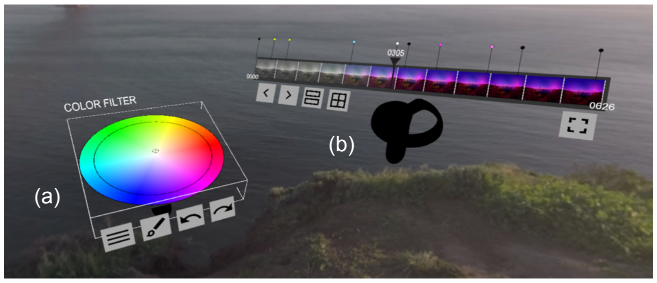


Figure 6. Authoring interface: (a) effect panel with current effect widget, a 2D hue saturation wheel for the tint effect; (b) the timeline panel showing frame thumbnails with spatial keyframe timeline markers rendered just above timeline.

own custom motion sickness compensation based on their directorial purview, without needing to rely on heuristics that have no context of the video's underlying narrative (Fig. 5a).

Gaze Guidance — A common 360° video problem occurs when users lose track of where to look in the video and miss important scene elements. Researchers have experimented with different visual stimuli to direct users towards points of interest [6, 26]. Walt Disney's "Cycles," a CG animated short film that is played in a proprietary real-time rendering engine, uses the desaturation of the entire scene as a way to indicate to the user they should reorient their view to a point of interest [3]. Our framework can author the same visual effects for captured 360° video and with directorial control (Fig. 5b).

Narrative Effects — A growing number of video games and VR experiences use screen-space effects to help portray the environment that the character of the story inhabits. These include rendering dirt or water droplets on the screen as if the "camera" is affected by the environment, creating lens flare effects based on point lights [16], using motion blur [12] to simulate sickness or drunkenness, or using bloom [39] to enhance realism or to create dream-like experiences. Our framework enables these effects for 360° film, bringing greater depth and immersion to the film experience (Fig. 5c).

AUTHORING INTERFACE

We created a VR application for authoring of view-dependent effects using spatial keyframes. It is written in C++ for Windows 10, using OpenFrameworks, OpenGL, OpenVR, and SteamVR, and tested on a MSI GT73VR Titan laptop with an Oculus Rift HMD.

Our interface is entirely in-headset, providing fast editing feedback while avoiding the colour perception issues encounted when desktop editing for VR. Nguyen et al. [25] also developed an in-headset video editor with related features such as spatial markers, adaptive vignetting, and minimap visualization. Nguyen et al. [24] further explored collaborative editing, including spatial and temporal annotations and gaze guidance. Galvane et al. [10] developed a complete production tool with model and camera animation for movie previsualization. These techniques are gaining professional adoption, as seen in the 2019 Lion King, which heavily relied on VR during production [36].

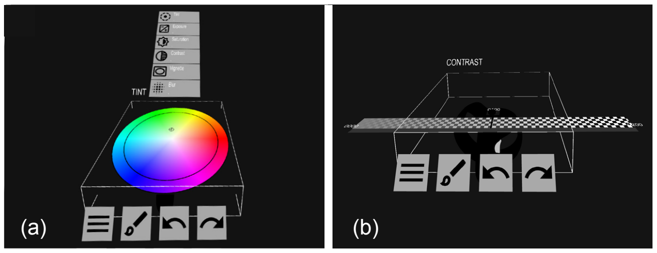


Figure 7. Effect widgets: (a) showing tint effect colour wheel widget and effect selection menu; (b) showing contrast effect horizontal 1D slider.

Traditional video editing tools that support animation (e.g. Adobe Premiere [2]) tend to have complex interfaces with keyframe tracks for each parameter and many small UI widgets, but this is not ideal for VR as the display resolution is low and controller input is imprecise. Performance-based interfaces for animation [4] or painting [8] opt for simpler interfaces requiring less precise control, in favour of easier authoring. This trade-off is well-suited to VR, our interface enables effects to be "painted" onto the video by rapidly dropping spatial keyframes. Instead of a complex UI to edit existing spatial keyframes, equivalent refinement is accomplished via unlimited undo or adding additional spatial keyframes.

The interface is composed of two panels, each attached to a VR controller. The dominant hand controls a timeline panel and the non-dominant hand controls an effect panel (Fig. 6). Interaction is through natural view direction, ray-cast pointing, and controller buttons.

We envision that the user would first cut and finalize their video using traditional video editing tools before adding view-dependent effects, which would be one of the final steps before publishing. The user would load a final draft of the video into our in-headset editor to initiate the process and enable in-headset editing using the tools described next (see the accompanying video for a demonstration of the in-headset user experience).

The Effect Panel

The effect panel is used to manipulate, preview, change, and place spatial keyframes. The panel shows the currently selected effect widget, and buttons to open the system menu, open the effects menu, and invoke undo and redo (Fig. 7).

Six effects can be selected using a menu: tint, exposure, saturation, contrast, vignette, and blur. Tint has two parameters, hue and saturation, adjusted using a 2D colour wheel widget (Fig. 7a). Vignette has two parameters, radius and falloff, adjusted using a 2D input control. Exposure, saturation, contrast, and blur effects have a single parameter adjusted by a widget with a 1D slider (e.g. (Fig. 7b). All effect widgets are manipulated using either the controller "thumbstick" or with a ray cast from the other controller.

Spatial Keyframe Placement

Once an effect and target parameter settings are selected, the user presses the controller trigger to preview the effect when the spatial keyframe is placed at the centre of their current view. They can adjust the position of the keyframe by changing their

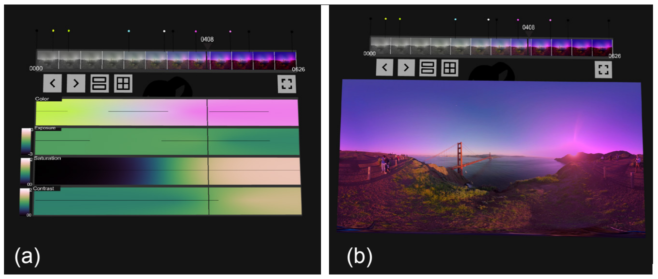


Figure 8. **Timeline effect visualizations: (a) effect tracks show how parameters change over time for the current view direction; (b) effect minimap showing how parameters change over the entire 360° equirect.**

view, then place the keyframe in the video by releasing the trigger.

The Timeline Panel

The timeline panel allows interactions such as play, pause, and scrubbing (Fig. 6b). Buttons under the timeline navigate to previous or next spatial keyframes, toggle different visualizations, and toggle attaching the panel to the controller or the world. The controller's "B button" can be held to view the scene with all effects turned off temporarily, an "unaltered film view."

Effect Visualization

To help users track many effects in time and space, the interface provides three visualizations, all implemented with GPU compute shaders for real time feedback. "Timeline Markers" show what time spatial keyframes are placed in the video near the current view direction. "Effect Tracks" visualize how effects evolve over time in the current view direction (Fig.8a). An "Effect Minimap" visualizes how effects interpolate over the entire 360° sphere using an equirectangular projection of a video frame (Fig.8b).

EXPERT REVIEW

To understand if filmmakers find our implementation of view-dependent effects for 360° video useful and usable, we conduct an expert review. Our design follows related studies [10,24,25] with subjective measures of perceived utility and qualitative observations.

Protocol

We recruited 8 expert users who are all professionals in the film industry and all report familiarity with post-production effect tools (age 26 to 55, $\mu = 39$, 6 male, 2 female). Remuneration was a $50 gift card. Each participant used our authoring interface and system while seated in a swivel chair in an unobstructed space.

After brief on-boarding to introduce view-dependent effects and the authoring interface, the participant explored the system in VR by experimenting on 360° video taken at a Golden Gate bridge scenic lookout (Fig. 9a). Once comfortable with the authoring interface, they viewed a 360° video of a rocket launch (Fig. 9b) on a desktop monitor in equirectangular format. Afterwards, they used our system in VR to add their desired effects to the rocket video using tools for tint, exposure, saturation, and contrast. Restricting tools in this way focused the

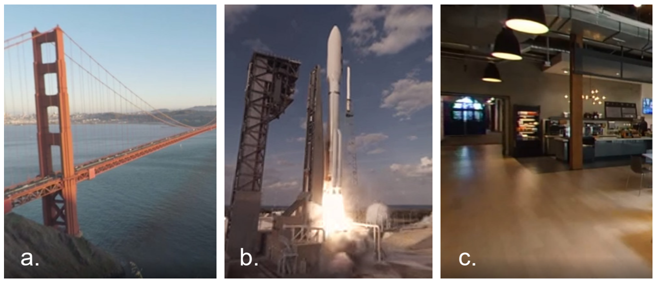


Figure 9. **Study videos: (a) Golden Gate; (b) Rocket Launch; (c) Office Scene.**

task and reduced learning overhead. The participant continued until they were satisfied with their result. After completing the task, the participant answered a series of 5-point Likert questions, followed by open-ended questions.

Results

Responses to Likert questions are shown in Figure 10.

Purpose (Fig. 10a) — All participants responded neutral or positive to statements about view-dependent effects being helpful for *story and narrative*, *artistic vision*, *user enjoyment*, and *user immersion*. Artistic vision received unanimous positive agreement.

Satisfaction (Fig. 10b) — Regarding overall feelings towards view-dependent effects, only one participant was less than neutral when rating *satisfaction with their final video*, *understanding the concept*, or *likelihood of using view-dependent effects in the future*. For the latter two measures, the remaining participants were all above neutral: it is notable that all but one would use view-dependent effects in the future. Participant comments often emphasized creative applications, like *"enhance the atmosphere and emotional feel"* (P7).

Usefulness (Fig. 10c) — Regarding how useful different view-dependent colour effects are: seven responded positively to *saturation* and *contrast*, with no negatives; *tint* and *exposure* are more varied, with one 'Disagree' and one 'Strongly Disagree' respectively, but the rest neutral or positive. While one participant found tint and exposure *"distracting"* (P1), other participants generally had positive comments, like *"I would definitely use this in a 360 video editing workflow"* (P5).

Confidence (Fig. 10d) — Participants were asked if they felt confident when adding spatial keyframes and making changes to the video. When predicting effects, five reported 'Agree' or above. When asked if they felt they had sufficient control to complete their task, four reported 'Agree' or above. When committing spatial keyframes, six participants reported 'Agree' or above. Notably, no participants commented on the interpolation scheme or unintended results.

User Interface (Fig. 10e) — Participants were asked to rate the usefulness of different user interface features. No participants responded negatively, two responded 'Neutral' for 'Timeline Markers' and 'Effect Tracks', all others responded 'Agree' and above. The 'Effect Minimap' visualization received the most positive response, with 7 participants reporting 'Strongly

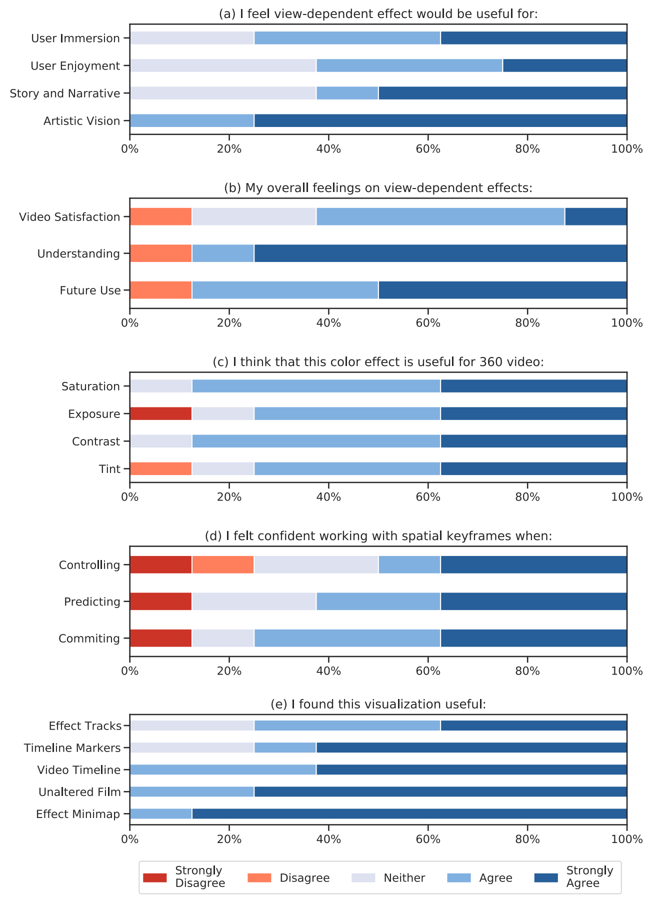


Figure 10. Expert responses to 5-point Likert questions.

Agree' and one reporting 'Agree'. The 'Timeline Panel' and 'Unaltered Film' also received all positive responses. We observed that participants frequently used the 'Unaltered Film' view to check their changes against the original video, giving them a baseline for comparison.

Discussion

Overall, participants were positive about view-dependent effects and our system. There were several comments on the realtime compositing aspect of the system, where one participant liked that they could do local effect changes *"without additional render times"* (P4) and another liked the *"decoupling of [the] filter from the video"* (P1) for realtime feedback. A third participant expressed that they were *"excited about working"* (P6) with these tools.

PERCEPTION STUDY

The goal of our second experiment is to determine if a general audience (i.e. non-experts) can perceive a difference when effects are rendered per-pixel or view-dependent, and whether they evoke different kinds of experiences. Importantly, our goal is not to show view-dependent effects as clearly superior, we include a preference question with a neutral choice as another way to gauge perceived difference. Establishing a universal audience preference for a creative effect may not be as meaningful or useful to filmmakers. Our method is inspired by related studies also gathering participant feedback by toggling between conditions, such as comparing different view-dependent textures [22] or different typefaces [44].

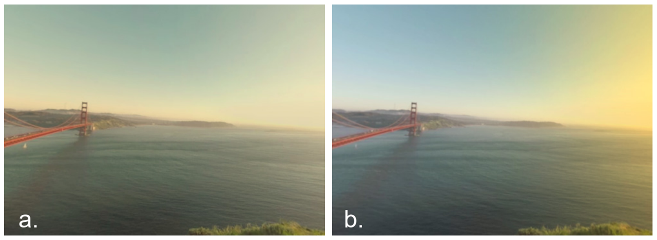


Figure 11. Example of rendering variants for Golden Gate video in perception study: (a) VIEW-DEPENDENT and (b) PER-PIXEL. An apparent colour gradient can be seen across image (b).

Protocol

We recruited 10 participants (age 24 to 35, $\mu = 27.8$, 8 male, 2 female). No remuneration was given, though free food was provided. All reported having tried a VR experience before; only one reported using VR frequently. None of the participants had any professional video or film experience.

The primary independent variable is *rendering variant* with two conditions: PER-PIXEL, the traditional static approach of existing video editing tools, and VIEW-DEPENDENT, our dynamic technique. Our system was used to play videos with both rendering variant conditions. The only difference was how effects were rendered: the same set of spatial keyframes and parameter value control points were used.

Two videos were tested, each with both rendering variant conditions. One video depicts an outdoor scene at a scenic overlook (Fig. 9a), referred to as Golden Gate. The other video depicts an indoor office environment (Fig. 9c), referred to as Office.

Video and rendering variant combinations were counterbalanced using a Latin square. During the session, the participant viewed the video in VR while the experimenter switched between rendering variants (example in Fig. 11). Variants were switched about every 5 to 20s, or at the request of the participant. To avoid bias, each variant was referred to generically as 'A' and 'B'. After viewing each video in this way, the participant rated the degree of perceptual difference on a 5-point Likert scale, and asked which condition they preferred. Each session was approximately 15 minutes.

Results

All participants perceived at least some difference between the variants, all seeing a 'Little Bit' of a difference or more (Fig. 12). A 'Subtle' difference was reported six times and an 'Apparent' difference four times. Two participants were confident that they had seen a large difference between variants for the Office video only. A Chi Squared Goodness of Fit Test found participant responses deviated from a uniform distribution ($\chi^2_{4,N=20} = 11.78,\ p = 0.018$), further suggesting there are perceived differences between the two variants.

Regarding preference, the divergence from the neutral choice of 'Neither' further demonstrates a perceptual difference in

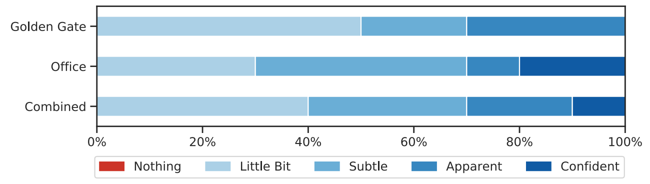


Figure 12. Proportion of perceived difference ratings between variants.

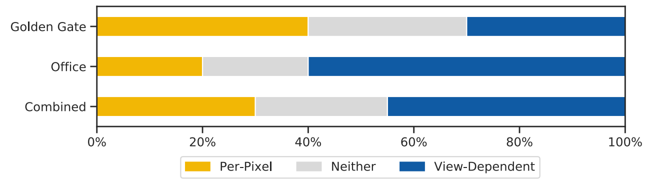


Figure 13. Proportion of rendering variant preference choices.

rendering variants (Fig. 13). Only 3 participants had a neutral experience for at least one video and only one participant had a neutral experience for both. A significant majority deviated from neutral to choose either view-dependent or per-pixel as their preferred variant ($\chi^2_{1,N=20} = 5,\ p = 0.025$). Regarding participants preference across all three choices (view-dependent, per-pixel, or neither), no significant result is reported ($\chi^2_{2,N=20} = 1.3,\ p = 0.52$). This is not unexpected and aligns with previous work examining preferences toward post-processing effects [11].

Discussion

There is compelling qualitative data demonstrating phenomenological variance when participants experience the two rendering variants.

In the Golden Gate video, participants commented that VIEW-DEPENDENT felt "fresh", "more clear", and "brighter" compared to PER-PIXEL. Some felt that there was more contrast between the viewing directions for VIEW-DEPENDENT and that it felt *"like 3D."* In contrast, PER-PIXEL was viewed as "less cozy", and multiple participants pointed out a "noticeable gradient" in the image.

In the Office video, VIEW-DEPENDENT was perceived to have various "light changes" as if a light source in the scene had moved. Others said the scene seemed to be "warmer". One participant suggested that it "felt like a video game" further elaborating to say it felt like a "bloom effect". In contrast, PER-PIXEL was seen as more "neutral" and "washed out."

Importantly, our goal is not to show view-dependent as superior, since finding such a result may not be possible or relevant. For example, related studies found cinematography effect preferences difficult to assess, since they are influenced by aspects like emotional attachment [11] or contextual factors like scene content and camera motion [42]. Consider how a dissolve transition is not better than a wipe transition; each may be used in different contexts and for different creative purposes. Similarly, view-dependent effects are another tool in the filmmaker's narrative and aesthetic toolbox. What is key is how participants stated each variant had a noticeable and perceived effect on their experience, showing view-dependent effects can evoking audience reactions in 360° video.

LIMITATIONS AND FUTURE WORK

The design of both the interpolation scheme and VR editor were refined through iteration and testing, resulting in one possible solution to enable view-dependent effects. In this section, we explore the limitations of our current approach and present areas for future research.

Spatial Interpolation Formulations

In the final implementation of our interpolation scheme, we used a sparse multi-dimensional interpolation method that was inspired by the work of Igarashi et al.[17], but other approaches are also possible. For example, we treat the spatial and temporal dimensions as a single unified space in which interpolation occurs; an alternative approach is to treat the spatial and temporal dimensions separately. Another consideration is how an effect propagates across the video. We implemented a global RBF scheme, which allows a single spatial keyframe to effect the entire video. For example, to apply an effect over a specific duration, spatial keyframes are needed before and after the duration's beginning and end, to set the effect parameter before the duration, within it, and afterwards, effectively localizing it to a particular part of the video. This approach is similar to the behaviour of existing keyframe editing tools found in commercial video editors. Alternatively, the spatial keyframe interpolation could be local instead, only rendering the effect to a particular part of the video. However, this approach has some apparent limitations when the effect needs to be maintained beyond the localized area of the keyframe. For example, in order to maintain an effect, the user would have to continuously apply adjacent keyframes throughout the video, which can be a tedious process.

A limitation of our global interpolation is that it can be difficult to apply effects to moving figures over time, which requires tracking the object. A moving camera can also be difficult. This is similar to editing moving objects and cameras in traditional video, which also requires tracking. Local effects may be able to support this use case more easily. The relative merit of global and local interpolation, and how to effectively combine them in a single system, is an open question.

Preformative and Control Based Interfaces

Our in-headset VR editing interface is designed so the user can layer effects on the video through a simplified interface. This is a similar approach to pixel-based photo editing tools, where the user performs edits by layering corrections on top of existing content. Another approach is a complex interface that enables precise control over individual keyframes, their parameters, and how they behave on the video. This approach is similar to how vector-based editing tools work, where existing content can be tweaked and corrected through direct manipulation. There are obvious trade-offs, where one gives control at the cost of increased UI complexity and the other simplicity at the cost of explicit control. Exploring this dichotomy further is an interesting direction for future work.

Expanding the Effect Vocabulary

We use a set of effects that are based on standard filmic formulations [13]. This provides a complete but minimal basis for evaluating view-dependent effects for 360° video. Other types of view-dependent effects are also possible, including screen-based effects like lens-flares and screen artifacts, or other post-processing effects like chromatic aberration. In actuality, any effect that can be parameterized by view and implemented in a graphics shader could be implemented and applied to the video. Future work could explore other view-dependent effects that are novel for 360° video, or even how view-dependent effects might be applied to traditional video.

CONCLUSION

We presented a technique that makes it possible for 360° filmmakers to add dynamic view-dependent effects for artistic purposes and specific application goals like motion sickness reduction, gaze guidance, and to great greater immersion. The key insight is to use spatial keyframes, which are sparsely interpolated to the user's view direction to compute the applied effect. A VR in-headset interface demonstrates how spatial keyframes and effect parameters can be authored. One user study shows that experts can understand and use our view-dependent effect technique, and they are excited about the possibilities of using it in their projects. A second study suggests audiences perceive stylistic differences in our technique.

In the future, we plan to extend our system with other types of view-dependent effects, and investigate other tools and interface controls for filmmakers, such as fine-grained control of spatial interpolation parameters. Overall, we hope that our work can help evolve 360° video into an even more creative and immersive experience.

REFERENCES

[1] Ábaco Digital Zaragoza. 2015. VIDEO 360: ESQUÍ ALPINO / ALPINE SKIING. (March 2015). `https://youtu.be/3b5H6Wjo_vk` Visited 19-Sep-2019.

[2] Adobe. 2019. Premiere pro. (2019). `http://www.adobe.com/ca/products/premiere.html` Visited 19-Sep-2019.

[3] Animation Magazine. 2018. Jeff Gipson Discusses Disney's Poignant VR Short 'Cycles'. (Nov. 2018). `https://www.animationmagazine.net/events/jeff-gipson-discusses-disneys-hot-vr-short-cycles/` Visited 19-Sep-2019.

[4] Connelly Barnes, David E. Jacobs, Jason Sanders, Dan B Goldman, Szymon Rusinkiewicz, Adam Finkelstein, and Maneesh Agrawala. 2008. Video puppetry. *ACM Transactions on Graphics* 27, 5 (Dec. 2008), 1. DOI:`http://dx.doi.org/10.1145/1409060.1409077`

[5] Jean Paul Berrut and Lloyd N. Trefethen. 2004. Barycentric Lagrange interpolation. *SIAM Rev.* 46, 3 (2004), 501–517. DOI: `http://dx.doi.org/10.1137/S0036144502417715`

[6] F. Danieau, A. Guillo, and R. Doré. 2017. Attention guidance for immersive video content in head-mounted displays. In *2017 IEEE Virtual Reality (VR)*. 205–206. DOI:`http://dx.doi.org/10.1109/VR.2017.7892248`

[7] Epic Games. 2019. Unreal Engine 4 Documentation. (2019). `https://docs.unrealengine.com/en-US/index.html` Visited 19-sept-2019.

[8] Facebook. 2019. Quill. (2019). `http://quill.fb.com/` Visited 19-Sep-2019.

[9] Ajoy S. Fernandes and Steven K. Feiner. 2016. Combating VR sickness through subtle dynamic field-of-view modification. In *2016 IEEE Symposium on 3D User Interfaces (3DUI)*. 201–210. DOI: `http://dx.doi.org/10.1109/3DUI.2016.7460053`

[10] Quentin Galvane, I-Sheng Lin, Fernando Arqelaquet, Tsai-Yen Li, and Marc Christie. 2019. VR as a Content Creation Tool for Movie Previsualisation. In *2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. 303–311. DOI: `http://dx.doi.org/10.1109/VR.2019.8798181`

[11] Michael L. Gleicher and Feng Liu. 2008. Re-Cinematography: Improving the Camerawork of Casual Video. *ACM Trans. Multimedia Comput. Commun. Appl.* 5, 1, Article Article 2 (Oct. 2008), 28 pages. DOI:`http://dx.doi.org/10.1145/1404880.1404882`

[12] Jean Philippe Guertin, Morgan McGuire, and Derek Nowrouzezahrai. 2014. A fast and stable feature-aware motion blur filter. In *High-Performance Graphics 2014, HPG 2014 - Proceedings*. 51–60. DOI: `http://dx.doi.org/10.2312/hpg.20141093`

[13] John Hable. 2017. Minimal Color Grading Tools. Filmic Worlds. (March 2017). `http://filmicworlds.com/blog/minimal-color-grading-tools/` Visited 19-Sep-2019.

[14] He Spoke Style. 2019. 360 Studio Tour! | Go Inside HSS Studios in VR 360. (Jan. 2019). `https://youtu.be/ayOQIIVFCLg` Visited 19-Sep-2019.

[15] Naty Hoffman, Haarm-Pieter Duiker, Joseph Goldstone, Yoshiharu Gotanda, Dominic Glynn, Lou Levinson, Josh Pines, and Jeremy Selan. 2010. Color Enhancement and Rendering in Film and Game Production. SIGGRAPH 2010 Course Notes. (July 2010). `http://renderwonk.com/publications/s2010-color-course/`

[16] Matthias Hullin, Elmar Eisemann, Hans-Peter Seidel, and Sungkil Lee. 2011. Physically-based real-time lens flare rendering. *ACM Transactions on Graphics* 30, 4 (jul 2011), 1. DOI: `http://dx.doi.org/10.1145/2010324.1965003`

[17] Takeo Igarashi, Tomer Moscovich, and John F. Hughes. 2005. Spatial keyframing for performance-driven animation. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation - SCA '05*. ACM Press, New York, New York, USA, 107. DOI: `http://dx.doi.org/10.1145/1073368.1073383`

[18] Dominik P. Käser, Evan Parker, Adam Glazier, Mike Podwal, Matt Seegmiller, Chun-Po Wang, Per Karlsson, Nadav Ashkenazi, Joanna Kim, Andre Le, Matthias Bühlmann, and Joshua Moshier. 2017. The Making of Google Earth VR. In *ACM SIGGRAPH 2017 Talks (SIGGRAPH '17)*. ACM, New York, NY, USA, Article 63, 2 pages. DOI: http://dx.doi.org/10.1145/3084363.3085094

[19] Thomas H. Kolbe. 2004. Augmented videos and panoramas for pedestrian navigation. In *Proceedings of the 2nd Symposium on Location Based Services & TeleCartography*, G. Gartner (Ed.). Vienna.

[20] Grzegorz Krawczyk, Karol Myszkowski, and Hans-Peter Seidel. 2005. Perceptual Effects in Real-time Tone Mapping. In *Proceedings of the 21st Spring Conference on Computer Graphics (SCCG '05)*. ACM, New York, NY, USA, 195–202. DOI: http://dx.doi.org/10.1145/1090122.1090154

[21] Sungkil Lee, Elmar Eisemann, and Hans-Peter Seidel. 2010. Real-time lens blur effects and focus control. *ACM Transactions on Graphics* 29, 4 (jul 2010), 1. DOI: http://dx.doi.org/10.1145/1778765.1778802

[22] Sean J. Liu, Maneesh Agrawala, Stephen DiVerdi, and Aaron Hertzmann. 2019. View-Dependent Video Textures for 360° Video. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology (UIST '19)*. ACM. DOI: http://dx.doi.org/10.1145/3332165.3347887

[23] Radoslaw Mantiuk and Mateusz Markowski. 2013. Gaze-Dependent Tone Mapping. *Lecture Notes in Computer Science* 7950 (May 2013), 426–433. DOI: http://dx.doi.org/10.1007/978-3-642-39094-4_48

[24] Cuong Nguyen, Stephen DiVerdi, Aaron Hertzmann, and Feng Liu. 2017a. CollaVR: Collaborative In-Headset Review for VR Video. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology (UIST '17)*. ACM, New York, NY, USA, 267–277. DOI: http://dx.doi.org/10.1145/3126594.3126659

[25] Cuong Nguyen, Stephen DiVerdi, Aaron Hertzmann, and Feng Liu. 2017b. Vremiere: In-headset virtual reality video editing. In *Conference on Human Factors in Computing Systems - Proceedings*, Vol. 2017-May. 5428–5438. DOI: http://dx.doi.org/10.1145/3025453.3025675

[26] Lasse T. Nielsen, Matias B. Møller, Sune D. Hartmeyer, Troels C. M. Ljung, Niels C. Nilsson, Rolf Nordahl, and Stefania Serafin. 2016. Missing the Point: An Exploration of How to Guide Users' Attention During Cinematic Virtual Reality. In *Proceedings of the 22nd ACM Conference on Virtual Reality Software and Technology (VRST '16)*. ACM, New York, NY, USA, 229–232. DOI: http://dx.doi.org/10.1145/2993369.2993405

[27] Jun-yong Noh, Douglas Fidaleo, and Ulrich Neumann. 2000. Animated Deformations with Radial Basis Functions. In *Proceedings of the ACM Symposium on Virtual Reality Software and Technology (VRST '00)*. ACM, New York, NY, USA, 166–174. DOI: http://dx.doi.org/10.1145/502390.502422

[28] Thomas Oskam, Alexander Hornung, Robert W Sumner, and Markus Gross. 2012. Fast and stable color balancing for images and augmented reality. In *Proceedings - 2nd Joint 3DIM/3DPVT Conference: 3D Imaging, Modeling, Processing, Visualization and Transmission, 3DIMPVT 2012*. IEEE, 49–56. DOI: http://dx.doi.org/10.1109/3DIMPVT.2012.36

[29] Sumanta N Pattanaik, James A Ferwerda, Mark D Fairchild, and Donald P Greenberg. 1998. A multiscale model of adaptation and spatial vision for realistic image display. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques - SIGGRAPH '98*. ACM Press, New York, New York, USA, 287–298. DOI: http://dx.doi.org/10.1145/280814.280922

[30] Amy Pavel, Björn Hartmann, and Maneesh Agrawala. 2017. Shot Orientation Controls for Interactive Cinematography with 360 Video. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology (UIST '17)*. ACM, New York, NY, USA, 289–297. DOI: http://dx.doi.org/10.1145/3126594.3126636

[31] Tania Pouli, Thanh Hang Phung, Technicolor France, Thanh Hang, Phung Technicolor France, and Thanh Hang Phung. 2018. VR Color Grading using Key Views. In *Proceedings of the Virtual Reality International Conference - Laval Virtual on - VRIC '18*. ACM Press, New York, New York, USA, 1–8. DOI: http://dx.doi.org/10.1145/3234253.3234287

[32] Michael J. D. Powell. 1987. Algorithms for Approximation. Clarendon Press, New York, NY, USA, Chapter Radial Basis Functions for Multivariable Interpolation: A Review, 143–167. http://dl.acm.org/citation.cfm?id=48424.48433

[33] Charles Poynton. 2012. Digital Video and HDTV Algorithms and Interfaces. Morgan Kaufmann, San Francisco, CA, USA, Chapter 24, 281–300.

[34] Shaun David Ramsey, J. Thomas Johnson III, and Charles Hansen. 2004. Adaptive temporal tone mapping. In *Proceedings of the 7th IASTED International Conference on Computer Graphics and Imaging*. 124–128.

[35] Charles Rose, Michael F. Cohen, and Bobby Bodenheimer. 1998. Verbs and Adverbs: Multidimensional Motion Interpolation. *IEEE Comput. Graph. Appl.* 18, 5 (Sept. 1998), 32–40. DOI: http://dx.doi.org/10.1109/38.708559

[36] Peter Rubin. 2019. Disney's New Lion King is the VR-Fueled Future of Cinema. Wired Magazine. (July

2019). http://wired.com/story/disney-new-lion-king-vr-fueled-future-cinema/ Visited 19-Sep-2019.

[37] Jeremy Selan. 2005. Using Lookup Tables to Accelerate Color Transformations. In *GPU Gems 2: Programming Techniques for High-Performance Graphics and General-Purpose Computation (Gpu Gems)*, Matt Pharr and Randima Fernando (Eds.). Addison-Wesley Professional, Chapter 24. https://developer.nvidia.com/gpugems/GPUGems2/gpugems2_chapter24.html Visited 19-Sep-2019.

[38] Robin Sibson. 1981. Interpolating Multivariate Data. John Wiley & Sons, New York, Chapter A brief description of natural neighbor interpolation, 21–36.

[39] Greg Spencer, Peter Shirley, Kurt Zimmerman, and Donald P Greenberg. 1995. Physically-based glare effects for digital images. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques - SIGGRAPH '95*. ACM Press, New York, New York, USA, 325–334. DOI: http://dx.doi.org/10.1145/218380.218466

[40] Scott N Steketee and Norman I Badler. 1985. PARAMETRIC KEYFRAME INTERPOLATION INCORPORATING KINETIC ADJUSTMENT AND PHRASING CONTROL. *Computer Graphics (ACM)* 19, 3 (1985), 255–262. DOI: http://dx.doi.org/10.1145/325165.325243

[41] Lingyun Sun, Yunzhan Zhou, Preben Hansen, Weidong Geng, and Xiangdong Li. 2018. Cross-objects user interfaces for video interaction in virtual reality museum context. *Multimedia Tools and Applications* (May 2018). DOI:http://dx.doi.org/10.1007/s11042-018-6091-5

[42] James Tompkin, Min H. Kim, Kwang In Kim, Jan Kautz, and Christian Theobalt. 2013. Preference and Artifact Analysis for Video Transitions of Places. *ACM Trans. Appl. Percept.* 10, 3, Article Article 13 (Aug. 2013), 19 pages. DOI:http://dx.doi.org/10.1145/2501601

[43] Greg Turk and James F. O'Brien. 2002. Modelling with implicit surfaces that interpolate. *ACM Transactions on Graphics* 21, 4 (2002), 855–873. DOI: http://dx.doi.org/10.1145/571647.571650

[44] Shaun Wallace, Rick Treitman, Jeff Huang, Ben D. Sawyer, and Zoya Bylinskii. 2020. Accelerating Adult Readers with Typeface: A Study of Individual Preferences and Effectiveness. In *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems (CHI EA '20)*. Association for Computing Machinery, New York, NY, USA, 1–9. DOI: http://dx.doi.org/10.1145/3334480.3382985

APPENDIX: FILMIC EFFECT FORMULATIONS

Blur removes high frequencies from the image by convolving it with a gaussian kernel $G_\sigma(x,y)$ of size k, controlled by the standard deviation parameter σ:

$$C_{out} = \sum_{i=1}^{k}\sum_{j=1}^{k} G_\sigma(i,j)C_{in}(i,j) \tag{6}$$

Tint emphasizes or de-emphasizes a colour per-pixel, creating colour cast and white balance effects. Tint's parameters are a normalized vector of one gain per colour channel, $C_{tint} = [C_r, C_g, C_b]$, which is applied to a pixel colour C_{in} with the element-wise product $\circ$.

Exposure adjusts the pixel's brightness with a single gain parameter, E. Combined, tint and exposure are applied as:

$$C_{out} = 2^E(C_{in} \circ C_{tint}) \tag{7}$$

Saturation makes a pixel more or less colourful by interpolating with the pixel luminance, $Y_{in} = C_{in} \cdot [0.25, 0.5, 0.25]$, and is controlled by a single parameter S:

$$C_{out} = Y_{in} + S(C_{in} - Y_{in}) \tag{8}$$

Note we do not use perceptual coefficients [33] to compute Y_{in}; ours are chosen by a colour grader for aesthetic reasons [13].

Contrast makes whites and blacks more or less distinct, by interpolating with a constant neutral gray, $b = [0.5, 0.5, 0.5]$, controlled by the parameter X:

$$C_{out} = b + X(C_{in} - b) \tag{9}$$

Vignette darkens and desaturates the image borders, and is controlled by two parameters, radius R and falloff F. For a view pixel $p = [x,y]$ and its distance from the view centre d, we compute the vignette as:

$$f = \begin{cases} 1 & d \leq R \\ 1 - \frac{d-R}{F} & d \leq R+F \\ 0 & d > R+F \end{cases} \tag{10}$$

$$C_{out} = fC_{in} + (1-f)C_{black} \tag{11}$$