OSCAM - Optimized Stereoscopic Camera Control for Interactive 3D

Thomas Oskam [1,2] Alexander Hornung [2] Huw Bowles [3,4] Kenny Mitchell [3,4] Markus Gross [1,2]

[1] ETH Zurich [2] Disney Research Zurich [3] Black Rock Studio [4] Disney Interactive Studios

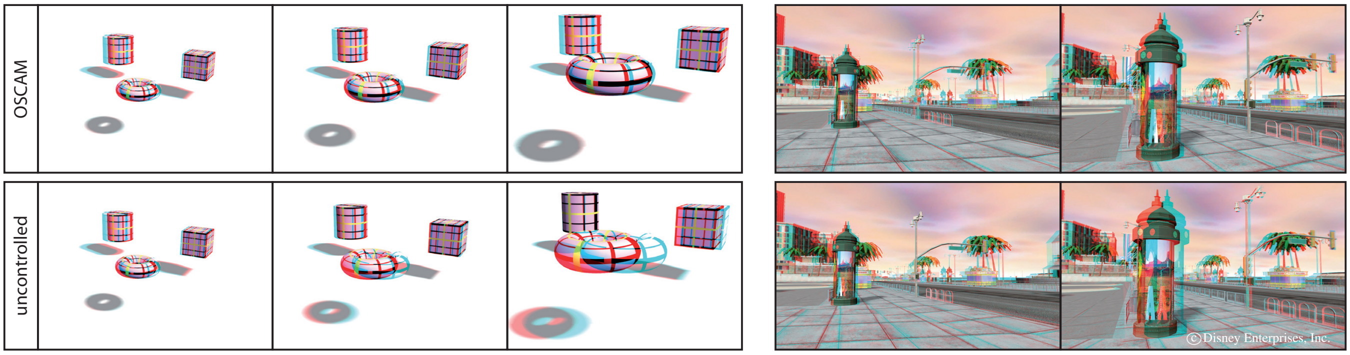

Figure 1: *Two stereoscopic shots of the camera moving towards objects. Our method keeps a constant target depth range when moving close to the objects. Uncontrolled stereoscopy, in contrast, can cause large disparities and destroy stereoscopic perception.*

Abstract

This paper presents a controller for camera convergence and interaxial separation that specifically addresses challenges in *interactive* stereoscopic applications like games. In such applications, unpredictable viewer- or object-motion often compromises stereopsis due to excessive binocular disparities. We derive constraints on the camera separation and convergence that enable our controller to automatically adapt to any given viewing situation and 3D scene, providing an exact mapping of the virtual content into a comfortable depth range around the display. Moreover, we introduce an interpolation function that linearizes the transformation of stereoscopic depth over time, minimizing nonlinear visual distortions. We describe how to implement the complete control mechanism on the GPU to achieve running times below 0.2ms for full HD. This provides a practical solution even for demanding real-time applications. Results of a user study show a significant increase of stereoscopic comfort, without compromising perceived realism. Our controller enables 'fail-safe' stereopsis, provides intuitive control to accommodate to personal preferences, and allows to properly display stereoscopic content on differently sized output devices.

CR Categories: I.3.3 [Computer Graphics]: Picture/Image generation—display algorithms,viewing algorithms;

Keywords: stereoscopic 3D, disparity control, real-time graphics, games, interactive 3D

Links: DL PDF

1 Introduction

Stereoscopic content creation, processing, and display has become a pivotal element in movies and entertainment, yet the industry is still confronted with various difficult challenges. Recent research has made substantial progress in some of these areas [Lang et al. 2010; Koppal et al. 2011; Didyk et al. 2011; Heinzle et al. 2011]. Most of these works focus on the classical production pipeline, where the consumer views ready-made content that has been optimized in (post-) production to ensure a comfortable stereoscopic experience. See Tekalp et al. [2011] for an overview.

In interactive applications that create stereoscopic output in real-time, one faces a number of fundamentally different challenges [Gateau and Neuman 2010]. For example, in a first-person game where the player is in control of the view, a simple collision with a wall or another object will result in excessive disparities that cause visual fatigue or destroy stereopsis (see Figure 1). In order to guarantee proper stereoscopy, one needs a controller that adjusts the range of disparities to the viewer's preferences. An example for such a controller is the work of Lang et al. [2010] which, however, has been designed for *post-capture* disparity range adaptation using complex image-domain warping techniques. In a game environment where the stereoscopic output is *created and displayed* in real-time, it is advisable to optimize the stereoscopic rendering parameters, i.e., camera convergence and interaxial separation, and to avoid computationally expensive solutions.

The problem can be formulated as one of controlling perceived depth. We use the term 'perceived depth' in the geometrical sense, where the distances reconstructed by the viewer are dominated by the observed screen disparities. Even though there are other important cues such as vertical size, focus that influence perceived depth [Backus et al. 1999; Watt et al. 2005], the work of Held and Banks [2008] showed that the geometrical approach is a valid approximation. The range of perceived depth around the screen that can be viewed comfortably is generally referred to as the *comfort zone*, and is defined as the range of positive and negative disparities that can be comfortably watched by each individual viewer [Smolic et al. 2011; Shibata et al. 2011]. Therefore, we are looking for an exact mapping of a specific range of distances in the scene into this depth volume around the screen. In the course of this article, we will refer to this volume as the *target depth range*. While we concentrate on the control of the mapping between the virtual and real space there exists prior work on how to derive a comfortable target depth range [Woods et al. 1993; Shibata et al. 2011]

Contributions. The contribution of this paper is a real-time stereoscopic control mechanism for disparity that is able to guarantee an exact mapping of arbitrary content to a comfortable target depth range. We start with a brief summary of the basic geometry in stereoscopic rendering, based on a viewer-centric model and an inverse scene-centric model. We derive constraints on the camera convergence and interaxial separation from these models that provide full control over resulting disparities and, hence, the mapped target depth range of arbitrary scene content. A second contribution of our work is a controlled temporal interpolation of the camera convergence and interaxial separation, which we use to linearize the perceived change in stereoscopic depth to avoid visual artifacts. Finally we describe how the complete controller is implemented on the GPU with very high performance (approximately 0.2ms per frame at full HD resolution), resulting in minimal added overhead compared to naive stereoscopic rendering.

Applications. Our controller has a variety of benefits for interactive stereoscopic applications. Given the viewing geometry (screen size, viewing distance) we can map any scene content into a specific target depth range. This means that the content created by a producer is guaranteed to create the desired stereoscopic depth effect independent of the actual display device, be it for example a large polarized display or a small Nintendo 3DS. Moreover, an application can be easily adapted to the depth or disparity constraints of a particular device, which helps to reduce ghosting or crosstalk artifacts. Because our method de-couples content production from stereoscopic display and automatically adapts to arbitrary output devices, it has the potential of considerably simplifying production and reducing costs. For the consumer our controller ensures that the stereoscopic footage is always comfortable to watch. The consumer may even adjust the target depth range intuitively to accommodate to personal preferences, such that proper stereopsis without excessive disparities is guaranteed. The results of a user study show that our controller is preferred over naive stereoscopic rendering.

2 Related Work

Production and consumption of stereoscopic 3D has been researched for many years, with applications ranging from cinema [Lipton 1982], scientific visualization [Fröhlich et al. 1999], television broadcasting [Meesters et al. 2004; Broberg 2011] to medical applications [Chan et al. 2005]. A recent survey over the field is provided in Tekalp et al. [2011]. Interestingly, solutions for interactive applications such as games are rare. In the following we discuss related works on stereo geometry, analysis and correction, camera control in real-time environments, and perception.

Stereo geometry: A detailed derivation of the geometry of binocular vision and stereoscopic imaging is given in Woods et al. [1993]. Their main focus is on the description of various image distortions such as keystoning or depth plane curvature, and they show how perceived depth changes under different viewing conditions. Grinberg et al. [1994] also describe the mapping between scene and perceived depth using different frames-of-reference, and propose a framework based on a minimum set of fundamental parameters to describe a 3D-stereoscopic camera and display system. Held and Banks [2008] derive a very complete geometrical model that maps from the scene over the screen to the perceived depth, including the projection to the retina. They not only parameterize the distance from the screen, but also the yaw, pitch, and roll of the viewer's head as well as a relative position to the screen. They use this model to predict distortions perceived by the viewer. Another summary of the stereo geometry is provided by Zilly et al. [2011]. They discuss constraints on the camera separation but do not take the camera convergence into account. Similar to the previous works they are mainly focused on quantifying depth distortions.

In contrast to these works, our paper provides explicit constraints on both camera convergence and interaxial separation in order to control the mapping of virtual scene content into perceived space. Moreover, none of the previous works has proposed a solution to handle nonlinear visual distortion during temporal interpolation of these parameters.

Stereoscopic content analysis and post-processing: Based on the above works, several methods have been developed for stereoscopic video analysis which estimate image disparities in order to predict and correct visual distortions such as cardboarding, the 'puppet theater effect', and other types of distortions [Masaoka et al. 2006; Kim et al. 2008; Pan et al. 2011]. Koppal et al. [2011] describe a framework for viewer-centric stereoscopic editing. They present a sophisticated interface for previewing and post-processing of live action stereoscopic shots, and support measurements in terms of perceived depth. Nonlinear remapping of disparities based on dense depth maps has been discussed by Wang et al. [2008]. Lang et al. [2010] generalize these concepts to more general nonlinear disparity remapping operators and describe an implementation for stereoscopic editing based on image-domain warping. A detailed state-of-the-art report on tools for stereoscopic content production is provided in [Smolic et al. 2011]. This report details challenges in the context of 3D video capturing and briefly discusses the most common problems in practical stereoscopic production such as the comfort zone, lens distortions, etc.

The focus of all these methods is on *post-production* analysis and correction of stereoscopic live-action video. Our work targets real-time stereoscopic rendering, with control over perceived depth for dynamic 3D environments, minimization of nonlinear depth distortions, and high efficiency for demanding real-time applications. Hence, our goals are complementary to these previous works.

Perceptual research on stereoscopy: An excellent overview of various stereoscopic artifacts such as *Keystone Distortion*, *Puppet Theater Effect*, *Crosstalk*, *Cardboarding*, and the *Shear Effect* and their effects on visual comfort is given in the work of Meesters et al. [2004]. The works from Backus et al. [1999] and Watt et al. [2005] show that perceived depth not only depends on the amount of disparities seen by the viewer, but also on monocular cues such as vertical size, focus, or perspective. The previously mentioned work by Woods et al. [1993] provides a user study to what extent different subjects can still fuse various disparity ranges. The results clearly showed that different persons have significantly varying stereoscopic comfort zones, indicating that individual control over stereoscopic depth is desirable. Stelmach et al. [2003] showed in a user study that shift-image convergence changes are generally preferred over toed-in camera setups, and that static, parallel camera setups are often problematic due to excessive disparities for nearby objects. A perceptual model which emphasizes the importance and utility of individual control over disparity and perceived depth, is described by Didyk et al. [2011]. Shibata et al. [2011] thoroughly examine the vergence-accommodation conflict and conduct user studies on the comfort of perceived depth for different viewing distances. Their work also provides a way to define a range of disparities that are comfortable to watch by an average viewer.

Results from perceptual experiments and research on stereoscopy clearly indicate a large variation in the physiological capabilities and preferences of different people. These indications motivate the need for tools that allow for a content-, display-, and user-adaptive control of stereoscopic disparity.

Camera control in interactive environments: Finally there is a large body of work on real-time camera control in virtual 3D environments and games, ranging from intuitive through-the-lens editing interfaces [Gleicher and Witkin 1992] and cinematographic shot composition [He et al. 1996; Bares et al. 1998] to sophisticated camera path planning and minimization of occlusions of points of interest [Oskam et al. 2009]. There exist excellent overviews of

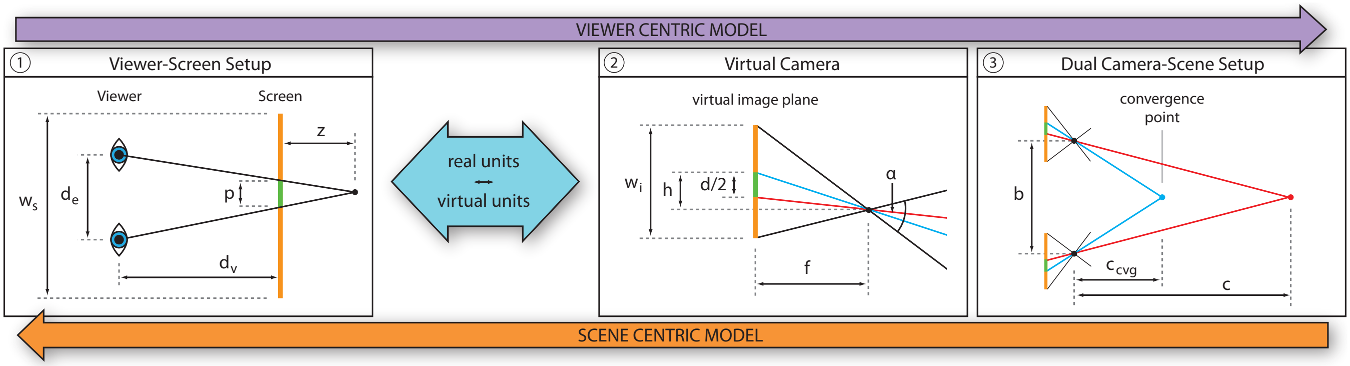

Figure 2: *The geometry pipeline of stereoscopic vision. Stage 1: the viewer with eye separation d_e at distance d_v to a screen of width w_s reconstructs a point at depth z in the target space due to the on-screen parallax p. Stage 2: The on-screen parallax in stage 1 is caused by a disparity d on the two camera image planes. The camera renders the scene with focal length f and an image shift h. Stage 3: Two cameras, with opposite but equidistant image shifts converge at distance c_{cvg} in the scene, and are separated by the interaxial distance b. The image disparity d between both cameras corresponds to a point with distance c from the cameras.*

this field [Christie et al. 2008; Haigh-Hutchinson 2009], which address theory of camera control as well as practical solutions for high-performance interactive applications. The work of Jones et al. [2001] also addresses real-time control of camera parameters to adjust the target depth range. However, they assume a parallel camera setup and solve the problem for still images only.

3 Basic Geometric Models of Stereoscopy

In this section we briefly revisit the basic geometry of stereoscopic vision relevant to our work. Our description is, in general, based on previous models [Woods et al. 1993; Held and Banks 2008; Zilly et al. 2011]. For the stereoscopic camera parameter, i.e. the convergence and interaxial separation, we follow the same definition as the existing models. The interaxial separation b is defined as the distance between the positions of the two cameras. The convergence distance c_{cvg} is defined as the distance between the intersection of the two camera viewing directions and the middle point between the two cameras. Both parameters are schematically shown in figure Figure 2.3. Note that we converge our cameras using image-shift instead of toeing them in. Image-shift convergence produces less artifacts [Woods et al. 1993; Stelmach et al. 2003].

First, we treat the camera convergence and interaxial separation as unknowns. This will enable us later to derive constraints for these parameters to achieve an optimal mapping of depths between scene- and real-world spaces. Second, we define real-world distances in a 3D volume relative to the screen instead of the viewer. This allows for a more intuitive definition and control of the target depth range (see Figure 2.1). Based on these prerequisites we derive the corresponding *viewer-centric model* and, as the inverse case, the *scene-centric model*, both of which are later needed to formulate constraints for the stereoscopic camera controller and the temporal transformation functions. Figure 2 gives an overview of the geometry pipeline of those two models.

3.1 Viewer-Centric Model

The viewer-centric model describes the reconstructed depth z of a point as a function of the scene distance c, the camera interaxial separation b, and the convergence distance c_{cvg}. This corresponds to a left-to-right traversal of the pipeline in Figure 2.

We use the viewer-screen configuration from Figure 2.1, where the viewer is looking from the defined distance d_{v} at the screen of the defined width w_{s}. d_{e} is the human interocular separation, usually considered to be 65*mm*. The viewer reconstructs a point in space relative to the screen due to his eyes converging according to the screen parallax p. This point is perceived at the distance

$$z(b, c_{\mathrm{cvg}}, c) = \frac{d_{\mathrm{v}}\, p(b, c_{\mathrm{cvg}}, c)}{d_{\mathrm{e}} - p(b, c_{\mathrm{cvg}}, c)}. \tag{1}$$

Note that the reconstructed depth z is positive behind the screen, and negative in front of the screen.

The conversion from on-screen parallax p to the image disparity d is simply a matter of scaling according to the screen width w_{s} and the virtual image plane width w_{i} (see Figure 2.1 and 2.2). This extends Eq. (1) to

$$z(b, c_{\mathrm{cvg}}, c) = \frac{d_{\mathrm{v}}\, d(b, c_{\mathrm{cvg}}, c)}{\frac{w_{\mathrm{i}}\, d_{\mathrm{e}}}{w_{\mathrm{s}}} - d(b, c_{\mathrm{cvg}}, c)}. \tag{2}$$

Finally, we can incorporate the scene distance c of the point, that is reconstructed at z, using the camera geometry (Figure 2.2) and the scene setup (Figure 2.3).

The triangle defined through the scene distance c and half the interaxial distance b is similar to the triangle defined by the camera focus f and $h - d/2$, where h is the image shift. Using the intercept theorem and $h = f\,b/(2\,c_{\mathrm{cvg}})$, we can reformulate Eq. (2) to include the convergence distance c_{cvg} and the camera interaxial distance b

$$z(b, c_{\mathrm{cvg}}, c) = \frac{d_{\mathrm{v}} \cdot (c - c_{\mathrm{cvg}})}{\frac{w_{\mathrm{i}}\, d_{\mathrm{e}}\, c\, c_{\mathrm{cvg}}}{w_{\mathrm{s}}\, f\, b} - (c - c_{\mathrm{cvg}})}. \tag{3}$$

3.2 Scene-Centric Model

Inverse to the viewer-centric model, the scene-centric model seeks the scene distance c as a function of the stereoscopic parameters c_{cvg} and b and a defined depth z in real-world space. This corresponds to a right-to-left traversal of the pipeline in Figure 2.

Given the scene setup in Figure 2.3 and the camera geometry in Figure 2.2, the scene distance c is given as

$$c(b, c_{\mathrm{cvg}}, z) = \frac{f\,b}{2h - d(z)} = \frac{f\,b}{\frac{f\,b}{c_{\mathrm{cvg}}} - d(z)}. \tag{4}$$

The image disparity d, which depends on the distance z, can be rescaled back to the on-screen parallax p using the image width w_{i} and the screen width w_{s}

$$c(b, c_{\mathrm{cvg}}, z) = \frac{f\,b}{\frac{f\,b}{c_{\mathrm{cvg}}} - \frac{w_{\mathrm{i}}\, p(z)}{w_{\mathrm{s}}}}. \tag{5}$$

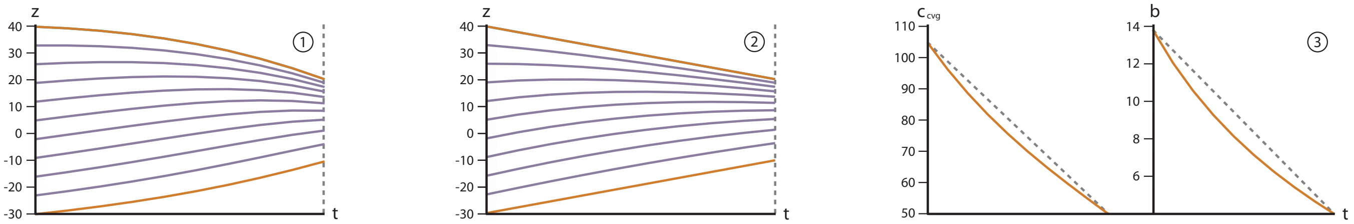

Figure 3: *Temporal interpolation of camera convergence and interaxial separation. The graph on the left (1) shows an example how the target depth range transforms over time when the stereoscopic parameters are linearly interpolated. In the middle (2) the same range is interpolated as in (1), using our linearized transformation. The two functions on the right (3) show the functions of c_{cvg} and b that achieve the linearized range transformation in (2).*

Finally, we can incorporate the viewer-screen geometry (Figure 2.1) to replace the on-screen parallax p by inverting Eq. (1). Inserting the result in Eq. (5) we get

$$c(b, c_{\mathrm{cvg}}, z) = \frac{f\,b}{\frac{f\,b}{c_{\mathrm{cvg}}} - \frac{w_{\mathrm{i}}}{w_{\mathrm{s}}}\frac{d_{\mathrm{e}}\,z}{d_{\mathrm{v}}+z}}. \qquad (6)$$

With Eq. (3) and Eq. (6) we have related perceived depth and scene distances using the camera convergence and interaxial separation. In the next section we use these equations to derive the constraints on these two parameters used in our controller.

4 Dynamic Stereo Control

Our goal is to control the camera convergence and interaxial separation over time such that we can optimally map dynamically changing scene content to a controlled target depth range. We therefore want to find constraints for the stereoscopic parameters that map any scene content to this pre-defined target range. This mapping can be achieved by solving two problems.

First, we derive constraints for the parameters c_{cvg} and b so that a series of points in the scene $[c_1, c_2, \dots c_n]$ are mapped onto a defined series of points $[z_1, z_2, \dots, z_n]$ in the target depth space. We solve for the constraints using a least-squares optimization in Section 4.1. Second, we derive an interpolation function for c_{cvg} and b that minimizes nonlinear distortions over time. We achieve this by guiding the z_i points with control functions as described in Section 4.2.

4.1 Constraints on Convergence and Separation

Given a defined series of depth values $[z_1, z_2, \dots, z_n]$, $z_i < z_j$ for $i < j$, where the screen surface is the reference frame. We want to compute values for the camera convergence c_{cvg} and separation b such that a corresponding series of scene points $[c_1, c_2, \dots c_n]$, with $c_i < c_j$ for $i < j$, is perceived as close as possible in the least squares sense to the z_i. This will allow us to map salient objects or the entire visible scene into defined target depth ranges.

First, we use the relation between perceived depth z and image disparity d in Eq. (2), to simplify the problem. The transformation between these two parameters is independent of c_{cvg} and b and, therefore, we can interchange the depth values z_i with the corresponding disparities d_i. The mapping problem can now be formulated by inserting the disparity values into equation Eq. (4), and setting them equal to the scene points. This gives us the following nonlinear system of equations

$$f\,b\,c_i - f\,b\,c_{\mathrm{cvg}} - c_i\,d_i\,c_{\mathrm{cvg}} = 0 \qquad \text{for} \qquad i = 1 \dots n,$$

which can, for example, be solved in a least-squares sense with a Gauss-Newton solver.

Target range control using two constraints. In general, the special case using two real-world depth values $[z_1, z_2]$ and two scene points $[c_1, c_2]$ is of highest importance, e.g., for mapping a given 3D scene into the pre-defined volume around the screen. In this case the above system has one non-trivial solution, and we can analytically determine the constraints for c_{cvg} and b:

$$c_{\mathrm{cvg}} = \frac{c_1\,c_2 \cdot (d_1 - d_2)}{c_1\,d_1 - c_2\,d_2}, \qquad (7)$$

$$b = \frac{c_1\,c_2 \cdot (d_1 - d_2)}{f \cdot (c_1 - c_2)}. \qquad (8)$$

The constraints in Eq. (7) and Eq. (8) enable an exact mapping of a specific range of distances in the scene into an arbitrary prescribed depth volume around the screen.

Useful application scenarios are, for example, mapping of the complete visible scene or of a particular salient object into a prescribed depth range. The disparities d_i corresponding to the desired output depth volume are defined via Eq. (2), and the values c_i are set to the current minimum and maximum distances in the scene. Another application of these constraints is to adapt to variable z_i boundaries. As the viewer adjusts the desired perceived depth volume, the renderer adjusts the camera convergence and interaxial separation to produce the desired output depth.

4.2 Temporal Constraint Interpolation

In the previous section, we have discussed how the basic stereoscopic parameters can be constrained in order to keep the perceived depth range within a defined limit. The constraints, however, only consider a snap-shot in time.

In an interactive environment, unpredictable object- or viewer-motion can change the scene depth instantly. This causes two problems. Let $(c^t_{\mathrm{cvg}}, b^t)$ denote the set of stereoscopic parameters at time t. On the one hand, if the scene depth changes from time $t-1$ to t and the interaxial separation and convergence distance are kept constant at $(c^{t-1}_{\mathrm{cvg}}, b^{t-1})$, the scene is mapped to a different target range, which can result in excessive disparities and compromise the stereoscopic perception. On the other hand, if the camera convergence and interaxial separation are immediately re-computed as $(c^t_{\mathrm{cvg}}, b^t)$ according to the constraints introduced in the previous section, the perceived depth of scene elements visible at both time-steps will change instantly. These sudden jumps in depth can be irritating to the viewer as well. So in general, we would like to control the stereoscopic parameters over time in order to reduce both types of artifacts.

The straight forward solution would be to interpolate linearly between the two parameter sets $(c^{t-1}_{\mathrm{cvg}}, b^{t-1})$ and $(c^t_{\mathrm{cvg}}, b^t)$. However, a simple linear change of camera convergence and interaxial separation causes the target depth range to transform in a nonlinear fashion, as shown in Figure 3.1. This scaling of the target depth results in nonlinear changes of object shapes and of the scene volume over time. In order to minimize these types of visual artifacts, we derive an interpolation function for our stereoscopic constraints that

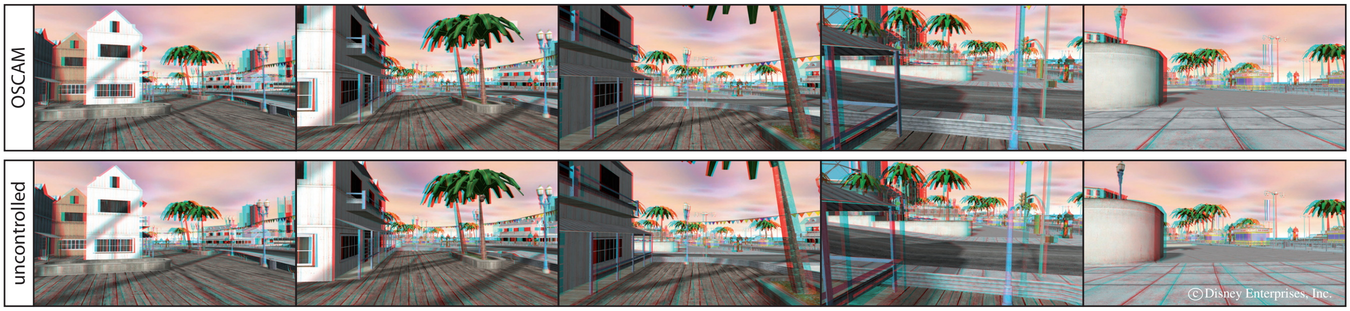

Figure 4: *Comparison between OSCAM and uncontrolled stereoscopy of medium to fast camera motion through complex environments. At the beginning of the shot, the uncontrolled camera has the exact same setup as the OSCAM, so that initially comfortable stereopsis is ensured. The uncontrolled camera fails to preserve a comfortable disparity range, causing excessive disparities and hence inducing eye-strain to the viewer.*

linearizes changes in perceived depth while keeping the perceived depth volume approximately constant.

General depth interpolation. Let z_i^t denote a depth value at time t in target space with respect to the screen (see Section 4.1). In order to keep the perceived depth volume constant over time, we can define an arbitrary (not necessarily linear) interpolation function $I_i(z_i^t, z_i^{t-1}, \alpha)$ for each of the depth values z_i^t. Each function gradually changes the point z_i^t back to its position z_i^{t-1} at a previous time $t-1$. In order to control all interpolation functions for all individual points simultaneously, we define the interpolation variable α as a function that depends on the current time step Δt of the interpolation and a predefined velocity v that is used to control how fast the target depth range transforms

$$\alpha(\Delta t, v) = \min\left(\frac{v\,\Delta t}{\frac{1}{n}\sum_{i=1}^{n}\operatorname{len}(I_i)}, 1\right). \tag{9}$$

The value of α is computed as the ratio between the distance for the time step Δt and the average length of all the control curves I_i. We use the min function in equation Eq. (9) to prevent ‘overshooting’ of the interpolation in case of a large time-step Δt or velocity v.

Now, in order to keep the target depths approximately constant over time, as soon as the scene depth values change from c_i^{t-1} to c_i^t, we first compute the resulting new target depths z_i^t, and then use the individual depth interpolation functions I_i to gradually update the camera convergence and interaxial separation to restore the target depths back to z_i^{t-1}.

Linearized range interpolation. Similar to Section 4.1, the special case of interpolating between two depth ranges defined just by their respective minimum and a maximum depth is of particular interest. Using our above formulations, we can define the interpolation in terms of the z_i^t , which allows us to linearize the change in target depth. Let $[z_1^{t-1}, z_2^{t-1}]$ and $[z_1^t, z_2^t]$ define the two depth ranges. Then the standard linear interpolation functions I_{lin} on the range boundaries achieve the desired linear change in depth

$$I_{\text{lin}}(z_i^t, z_i^{t-1}, \alpha) = (1-\alpha)\, z_i^t + \alpha\, z_i^{t-1}. \tag{10}$$

The graph in Figure 3.2 shows the effect of Eq. (10) on the target depth range over time. The transformation is linear along the boundaries of the target depth range. Compared to a simple linear interpolation of c_{cvg} and b (see Figure 3.1), our linearized transformation introduces significantly less distortions over time.

The resulting functions for c_{cvg} and b over time for the linearized transformation are shown in Figure 3.3. In both graphs, the dashed line shows the linear interpolation of the parameter. It is apparent that our optimized functions differ considerably from linear interpolation of the parameters.

5 Efficient Implementation

The stereoscopic controller algorithm produces a sequence of parameter pairs (c_{cvg}^t, b^t) at each frame t. Pseudo code of the parameter update is provided in Algorithm 1.

Algorithm 1 Stereoscopic Parameter Update

```
1: procedure UPDATESTEREOPARAMETER($c_{\text{cvg}}^{t-1}, b^{t-1}$)
2:     $[c_1^t, c_2^t] \leftarrow$ getSceneDepthRange()
3:     $[z_0^t, z_1^t] \leftarrow$ mapDepth($c_1^t, c_2^t, c_{cvg}^{t-1}, b^{t-1}$)
4:     $\alpha \leftarrow$ computeAlpha($\Delta t, v$)
5:     $[d_0, d_1] \leftarrow$ transform($I_{lin}, z_0^t, z_1^t, \alpha$)
6:     $[b^t, c_{\text{cvg}}^t] \leftarrow$ constrainParameter($c_0^t, c_1^t, d_0, d_1$)
7:     return $[c_{\text{cvg}}^t, b^t]$
8: end procedure
```

The first step is to acquire the new scene depth range $[c_0^t, c_1^t]$ (line 2). Efficient computation of this range is described below. Using this range, the new depth range $[z_0^t, z_1^t]$ for this frame can be computed (line 3) using Eq. (3). Given the target depth range $[z_0^{t_0}, z_1^{t_0}]$, and the velocity v, the interpolation variable α can be determined (line 4) with Eq. (9). Then, the target depth range can be interpolated over time (line 5) using the interpolation function in Eq. (10) converted to the corresponding disparity values $[d_0, d_1]$ (using Eq. (2)). Finally, using the stereoscopic parameter constraints, Eq. (7) and Eq. (8), the new values for c_{cvg}^t and b^t can be computed (line 6).

Efficient depth range acquisition. In real-time applications, budgets are tight and efficiency is a critical factor. To be usable in a production scenario, any real-time algorithm needs to fit into a budget of a few milliseconds (at 60fps, the total frame budget is only 16.7ms). The only stage of our algorithm that cannot be computed in constant cost is the determination of the minimum and maximum depths in the scene. To efficiently obtain these depths, we perform a number of min-max reduction passes of the depth buffer on the GPU [Greß et al. 2006]. Although the number of passes is logarithmic in the screen size, modern GPUs are highly optimized for this workload and these passes are computed very cheaply. Indeed, we timed the passes on a recent Nvidia GPU (the GTX580) and the cost was only 0.09ms at 1280x720 and 0.18ms at 1920x1080.

6 Applications and Experimental Evaluation

In the following, we describe several scenarios and applications of our stereoscopic camera controller. We furthermore conducted a user study to validate the utility of our method. Please see also the accompanying video for the actual renderings of dynamic scenes created with our method.

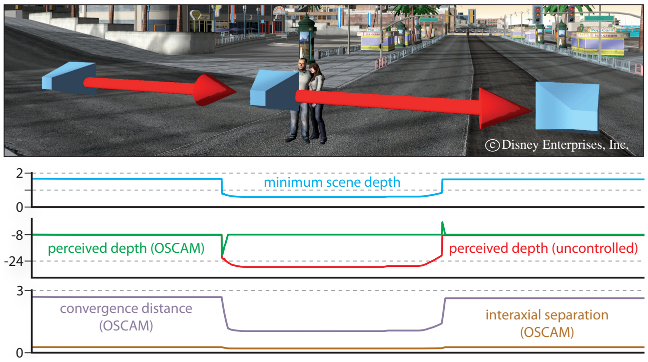

Figure 5: *Comparison between constrained and unconstrained stereoscopic cameras while horizontally moving the camera past a close obstacle. The graphs on the bottom show that the unconstrained stereoscopic parameters cause excessive disparities. Our constrained camera adapts smoothly to the discontinuous depth.*

6.1 Applications

Adaptive stereoscopy and automatic fail-safe. When moving the camera through a scene, the render-depth is constantly changing. Improper camera convergence and interaxial separation can cause large disparity values when an obstacle suddenly comes close to the camera. An example is shown in Figure 4, where a player is moving fast through the environment. While our method is able to adapt to vast depth changes, uncontrolled stereoscopy causes excessive disparities and hence induces eye-strain to the viewer.

Another typical situation encountered in interactive, dynamic environments is the sudden change of scene depth. An example is shown in Figure 5, where the camera is horizontally translated across the street. It passes very closely in front of a couple, creating a sudden discontinuous decrease in the minimum scene depth. If this is not handled properly, the depth perception is immediately destroyed. The graphs in Figure 5 show how our method adapts the stereoscopic parameters to prevent exceeding disparities to appear for too long.

Figure 6 shows an example of our parameter optimization, mapping multiple points in the scene onto multiple target ranges. The the top graph shows the desired mapping of the car and the environment in real space. The bottom images show stereoscopic renderings of the least-squares solution for the stereoscopic parameters and the mappings of the car and environment.

Changing target screen size and viewing conditions. Stereoscopic imagery that is produced for a certain target screen and viewing distance is optimized to create a particular depth impression. If, however, the content is shown under different viewing conditions, the viewer may experience a completely different depth sensation. Figure 7 shows a comparison of two different views that are optimized for a *Television Screen*, a *PC Monitor*, and *Nintendo 3DS*. The viewing conditions and target depth ranges for each device can be found in Table 1. The stereoscopic image created for a Nintendo 3DS shown on a large television screen can cause extremely large disparities. Our method is able to adapt content automatically to different output screens, given a pre-defined comfortable target depth range for an average viewer.

	TV	PC	3DS
TV	[-51.4, 86.5]	[-6.1, 8.2]	[-0.5, 0.6]
PC	[-65.8, 164.7]	[-8.0, 14.0]	[-0.7, 1.0]
3DS	[-837.1, 3532.2]	[-105.4, 240.9]	[-1.0, 1.5]

Table 1: *Content scaling matrix. The entry in the row i and column j shows the target depth range (in cm) of the image that is produced for i and viewed on j. The viewing conditions for each device (in cm): TV: $w_s = 100, d_v = 300$. PC: $w_s = 50, d_v = 65$. Nintendo 3DS: $w_s = 7.6, d_v = 35$.*

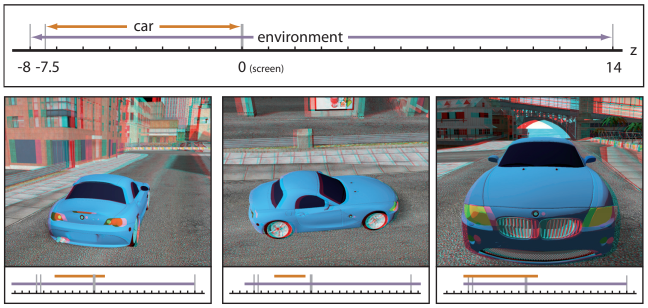

Figure 6: *Example of a scene with multiple points mapped onto multiple target depth values. The top row shows the desired mapping. The car should appear in the target range [-7.5, 0.0] cm while the environment should be mapped into [-8.0, 14.0] cm. The bottom row shows how different views are rendered as close as possible to the desired mapping in the least squares sense using our nonlinear optimization.*

Intuitive control. Our method provides intuitive and exact stereoscopic control by allowing the viewer to adapt the perceived borders of the depth range to the respective personal comfort zone. The user can also define high-level goals for the depth, as shown in Figure 8. The viewer may specify to move or scale the target depth image, without worrying about the exact depth values or the stereoscopic camera parameters.

OSCAM also provides an interesting tool for content creators and artists in production. Our controller can be used to intuitively script perceived depth for different scene parts and hence create artistic effects such as emphasized depth for certain objects, stereoscopic flatlands, etc. Moreover, since our method can map any scene content into a pre-defined target depth range without changing any camera parameters except interaxial separation and convergence, we effectively de-couple classical '2D' camera work from stereoscopic camera work. This allows for a stream-lined production pipeline, where both stereoscopic and classical camera artists can work independently.

6.2 User Study

In order to further evaluate the utility of our method we conducted a user study with 31 subjects. The study was aimed at comparing our OSCAM to standard stereoscopic rendering with fixed camera convergence and interaxial separation. All subjects were tested for proper stereoscopic vision using a random dot stereogram test. The goals of this study were twofold. First, we tested whether the subjects prefer the static, uncontrolled stereoscopy or our OSCAM as more comfortable. Second, we examined if our controller compromises perceived realism due to the involved adaptation of camera convergence and interaxial separation.

To this end we rendered 10 side-by-side comparisons of different scenes using uncontrolled stereoscopic parameters and using the OSCAM controller for pairwise evaluation [David 1963]. The rendered scenes contained typical scenarios encountered in interactive 3D environments, including

- continuous view changes between close-ups and wider scenes
- objects suddenly entering the field of view
- three long sequences where a player explores a complex game environment

We randomized the order of scenes as well as the respective positions on the screen. Each pair was shown three times so that the viewers had sufficient time for comparison. The study was performed on a line-wise polarized 46inch Miracube display. The

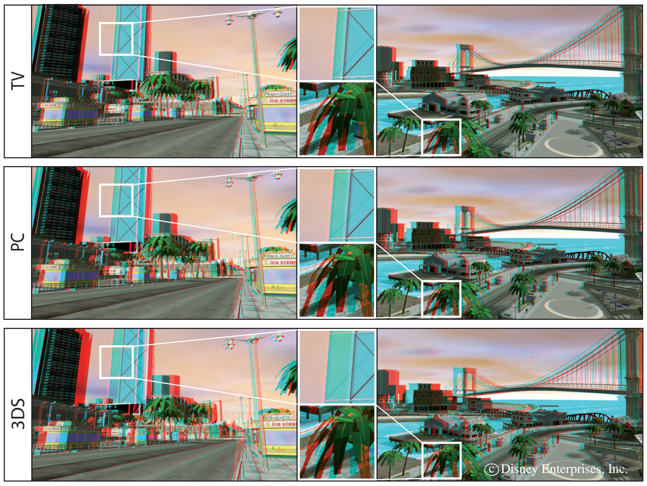

Figure 7: *Content produced for different screen sizes. The viewing conditions and target depth ranges for each device can be found in Table 1. In the middle of each row, magnifications of certain parts of the vistas are shown. The differing viewing conditions demand different disparities for the depth image to be perceived in the desired range.*

viewing distance was 3m, and our controller was configured for a target depth range of [-51.4, 86.5] cm with respect to the display. The static stereoscopic parameters were set such that at the beginning of each scene the resulting disparities were identical to our controller. According to our above mentioned goals, for every comparison the participants had to answer either *left* or *right* for the following two questions:

Q1: Which one is more comfortable to watch?
Q2: Which one looks more realistic to you?

When considering all 10 scenes in the evaluation, we received 310 votes for each of the two questions. Regarding question 1 about comfortable stereo viewing, our controller was preferred in 61.7% (191 of 310) of the examples, while the fixed stereo was preferred in 38.3% of the cases. In terms of realism, the results of our controller were preferred in 60.7% (188 of 310) of the scenes compared to 39.3% for the static stereo settings.

One stereoscopic issue that has not been considered by the stereo controller proposed in this paper is the problem of so-called frame violations: if an object with negative disparity, i.e., in front of the screen, is cropped at the screen borders, the human visual system can get confused. This can be uncomfortable to the viewer. For the results used in this study, our stereo controller mapped the complete scene into a target volume [-51.4, 86.5] cm around the display. This introduced frame violations in some situations. We deliberately did not correct for such frame violations in order to evaluate the effects of depth remapping only. However, such a correction is trivial to add by adding corresponding ‘floating windows’ [Gateau and Neuman 2010; Smolic et al. 2011]. Therefore, if we remove the two sequences from the evaluation where the most obvious frame violations occurred (resulting in 248 answer per question), the preference for our method in terms of comfort raises to 70.9% (176 of 248), and in terms of realism to 69.3% (172 of 248). All these results are statistically significant with a p-value < 0.01.

From these results we can conclude that the stereoscopic imagery optimized by our controller was generally preferred by the subjects and created a more comfortable viewing experience without compromising perceived realism of scene depth. In addition, the results indicate an interesting correlation between comfort and perceived realism that we did not anticipate. In 86.1% of the answers the more comfortable rendering was also selected as the more realistic one. This is interesting since the dynamic adaptation of baseline and convergence and the resulting scaling of perceived depth over time seems to be less compromising in terms of perceived realism than excessive disparities.

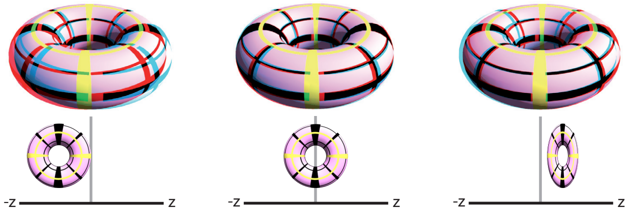

Figure 8: *Examples of exact stereoscopic control using our method. Left: The torus is rendered such that it appears directly in front of the screen plane. Middle: Exactly half of the torus appears in front and the other half behind the screen. Right: The torus appears one seventh of its original target length behind the screen and its perceived length is halved. With our controller such settings can be guaranteed while the viewpoint changes dynamically.*

6.3 Limitations and Future Work

The disparity optimization framework, to this end, only manipulates the two most basic stereoscopic parameters, the camera convergence and interaxial separation. This allows for an analytical solution that is very fast to compute, but it is only a solution in a two-dimensional configuration space. While this is the most practical solution for real-time environments, we would like to investigate techniques for more complex nonlinear disparity remappings. Our experimental study provides encouraging evidence that this might be even more beneficial for the viewer. However, our study is only a first indicator that adaptive stereoscopy can increase viewer comfort. Additional studies need to be conducted to better understand the effects of such a stereoscopic control. Furthermore, our method is designed for interactive environments without control over the camera movement. However, as we only manipulate the camera separation and convergence, nothing would prevent our method from working with real cameras, too. We would like to investigate the possibility to implement our method on a stereoscopic camera rig such as the one by Heinzle et al. [2011]. Finally, the linearized temporal interpolation of the stereoscopic parameters intuitively seems to work well for adjusting stereoscopy on-the-fly. However, it is not clear yet if the linearized interpolation is optimal. On the one hand, we want to further explore the temporal behavior when optimizing for multiple target regions, and evaluate to what extent local minima of the optimization influence the result. On the other hand, we would like to further investigate the effect of our linearized interpolation on the viewer’s perception.

7 Conclusion

In this paper we have described an effective and efficient solution for optimizing stereoscopic camera parameters in interactive, dynamic 3D environments. On the basis of a viewer-centric and a scene-centric model, we have defined the mapping between the scene depth and perceived depth as an optimization problem. We have derived constraints for a stereoscopic camera controller that is capable of rendering any visible scene content optimally into any target depth range for arbitrary devices and viewing configurations. Moreover, we have addressed the problem of blending stereoscopic parameters and the resulting nonlinear distortions in perceived depth. Our method allows for a linearization of such effects, but also for more complex temporal transformations to render desired depth effects in the target space. With running times less than 0.2ms per frame even at full HD resolution, our controller is fast enough even for demanding real-time applications. Our experimental evaluation showed that our controller is preferred over naive stereoscopic rendering.

Acknowledgements

The authors are grateful to Martin Banks, Aljoscha Smolic, Tobias Pfaff, Alex Stuard, and the anonymous reviewers for their helpful comments and suggestions as well as Wojciech Jarosz for providing the car model.

References

BACKUS, B., BANKS, M. S., VAN EE, R., AND CROWELL, J. A. 1999. Horizontal and vertical disparity, eye position, and stereoscopic slant perception. *Vision Research 39*, 6, 1143–1170.

BARES, W. H., GREGOIRE, J. P., AND LESTER, J. C. 1998. Realtime constraint-based cinematography for complex interactive 3D worlds. In *In Tenth National Conference on Innovative Applications of Artificial Intelligence*, 1101–1106.

BROBERG, D. 2011. Infrastructures for home delivery, interfacing, captioning, and viewing of 3-d content. *Proceedings of the IEEE 99*, 4 (april), 684 –693.

CHAN, H. P., GOODSITT, M. M., HELVIE, M. A., HADJIISKI, L. M., LYDICK, J. T., ROUBIDOUX, M. A., BAILEY, J. E., NEES, A., BLANE, C. E., AND SAHINER, B. 2005. Roc study of the effect of stereoscopic imaging on assessment of breast lesions. *Medical Physics 32, 4*, 1001–1009.

CHRISTIE, M., OLIVIER, P., AND NORMAND, J.-M. 2008. Camera control in computer graphics. *Comput. Graph. Forum 27*, 8, 2197–2218.

DAVID, H. A. 1963. *The Method of Paired Comparisons*. Charles Griffin & Company.

DIDYK, P., RITSCHEL, T., EISEMANN, E., MYSZKOWSKI, K., AND SEIDEL, H.-P. 2011. A perceptual model for disparity. *ACM Trans. Graph. 30*, 4, 96.

FRÖHLICH, B., BARRASS, S., ZEHNER, B., PLATE, J., AND GÖBEL, M. 1999. Exploring geo-scientific data in virtual environments. In *IEEE Visualization*, 169–173.

GATEAU, S., AND NEUMAN, R. 2010. Stereoscopy from xy to z. In *SIGGRAPH ASIA Courses*.

GLEICHER, M., AND WITKIN, A. 1992. Through-the-lens camera control. *SIGGRAPH Comput. Graph. 26* (July), 331–340.

GRESS, A., GUTHE, M., AND KLEIN, R. 2006. Gpu-based collision detection for deformable parameterized surfaces. *Comput. Graph. Forum 25*, 3, 497–506.

GRINBERG, V. S., PODNAR, G., AND SIEGEL, M. 1994. Geometry of binocular imaging. In *Stereoscopic Displays and Virtual Reality Systems*, vol. 2177, 56 – 65.

HAIGH-HUTCHINSON, M. 2009. *Real-time cameras. A guide for game designers and developers.* Morgan Kaufmann.

HE, L., COHEN, M. F., AND SALESIN, D. 1996. The virtual cinematographer: A paradigm for automatic real-time camera control and directing. In *SIGGRAPH*, 217–224.

HEINZLE, S., GREISEN, P., GALLUP, D., CHEN, C., SANER, D., SMOLIC, A., BURG, A., MATUSIK, W., AND GROSS, M. H. 2011. Computational stereo camera system with programmable control loop. *ACM Trans. Graph. 30*, 4, 94.

HELD, R. T., AND BANKS, M. S. 2008. Misperceptions in stereoscopic displays: a vision science perspective. In *APGV*, 23–32.

JONES, G., LEE, D., HOLLIMAN, N., AND EZRA, D. 2001. Controlling perceived depth in stereoscopic images. In *Stereoscopic Displays And Virtual Reality Systems VIII*, 200–1.

KIM, H. J., CHOI, J. W., CHAING, A.-J., AND YU, K. Y. 2008. Reconstruction of stereoscopic imagery for visual comfort. In *Stereoscopic Displays and Virtual Reality Systems XIV, SPIE Vol. 6803*.

KOPPAL, S. J., ZITNICK, C. L., COHEN, M. F., KANG, S. B., RESSLER, B., AND COLBURN, A. 2011. A viewer-centric editor for 3D movies. *IEEE Computer Graphics and Applications 31*, 1, 20–35.

LANG, M., HORNUNG, A., WANG, O., POULAKOS, S., SMOLIC, A., AND GROSS, M. H. 2010. Nonlinear disparity mapping for stereoscopic 3D. *ACM Trans. Graph. 29*, 4.

LIPTON, L. 1982. *Foundations of the Stereoscopic Cinema: A Study in Depth.* Van Nostrand Reinhold Inc.,U.S.

MASAOKA, K., HANAZATO, A., EMOTO, M., YAMANOUE, H., NOJIRI, Y., AND OKANO, F. 2006. Spatial distortion prediction system for stereoscopic images. *Electronic Imaging 15*, 1.

MEESTERS, L. M. J., IJSSELSTEIJN, W. A., AND SEUNTIENS, P. J. H. 2004. A survey of perceptual evaluations and requirements of three-dimensional TV. *IEEE Trans. Circuits Syst. Video Techn. 14*, 3, 381–391.

OSKAM, T., SUMNER, R. W., THÜREY, N., AND GROSS, M. H. 2009. Visibility transition planning for dynamic camera control. In *Symposium on Computer Animation*, 55–65.

PAN, H., YUAN, C., AND DALY, S. 2011. 3D video disparity scaling for preference and prevention of discomfort. In *Stereoscopic Displays and Applications XXII, SPIE Vol. 7863*.

SHIBATA, T., KIM, J., HOFFMAN, D. M., AND BANKS, M. S. 2011. The zone of comfort: Predicting visual discomfort with stereo displays. *Journal of Vision 11*, 8.

SMOLIC, A., KAUFF, P., KNORR, S., HORNUNG, A., KUNTER, M., MÜLLER, M., AND LANG, M. 2011. Three-dimensional video postproduction and processing. *Proceedings of the IEEE 99*, 4 (april), 607 –625.

STELMACH, L. B., TAM, W. J., SPERANZA, F., RENAUD, R., AND MARTIN, T. 2003. Improving the visual comfort of stereoscopic images. In *Proc. SPIE 5006, 269*.

TEKALP, A. M., SMOLIC, A., VETRO, A., AND ONURAL, L., Eds. 2011. *Special issue on 3-D Media and Displays*, vol. 99, 4. Proceedings of the IEEE.

WANG, C., AND SAWCHUK, A. A. 2008. Disparity manipulation for stereo images and video. In *Stereoscopic Displays and Applications XIX, SPIE Vol. 6803*.

WATT, S. J., AKELEY, K., ERNST, M. O., AND BANKS, M. S. 2005. Focus cues affect perceived depth. *Journal of Vision 5*, 10.

WOODS, A., DOCHERTY, T., AND KOCH, R. 1993. Image distortions in stereoscopic video systems. In *Stereoscopic Displays and Applications IV, Proceedings of the SPIE*, vol. 1915.

ZILLY, F., KLUGER, J., AND KAUFF, P. 2011. Production rules for stereo acquisition. *Proceedings of the IEEE 99*, 4 (april), 590 –606.