RYAN: Rendering Your Animation Nonlinearly projected

Patrick Coleman* Karan Singh†

University of Toronto

Abstract

Artistic rendering is an important research area in Computer Graphics, yet relatively little attention has been paid to the projective properties of computer generated scenes. Motivated by the surreal storyboard of an animation in production—*Ryan*—this paper describes interactive techniques to control and render scenes using nonlinear projections. The paper makes three contributions. First, we present a novel approach that distorts scene geometry such that when viewed through a standard linear perspective camera, the scene appears nonlinearly projected. Second, we describe a framework for the interactive authoring of nonlinear projections defined as a combination of scene constraints and a number of linear perspective cameras. Finally, we address the impact of nonlinear projection on rendering and explore various illumination effects. These techniques, implemented in *Maya* and used in the production of the animation *Ryan*, demonstrate how geometric and rendering effects resulting from nonlinear projections can be seamlessly introduced into current production pipelines.

CR Categories: I.3.3 [Computer Graphics]: Image Generation—Viewing Algorithms; I.3.5 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation

Keywords: Non-Photorealistic Rendering, Multiprojection, Nonlinear Perspective, Local Illumination

1 Introduction

Artists using traditional media almost always deviate from the confines of a precise linear perspective view. Many digital artists, however, continue to struggle with the standard pinhole camera model used in Computer Graphics to generate expressive 2D images of 3D scenes. The history of the use of linear perspective in art outlined in Figure 2 provides good insight into its benefits and limitations. Even though the earliest documented observation of perspective has been dated to approximately 4000 B.C., renderings of 3D scenes as late as 1400 lack depth and show clear perspective errors, as can be seen on the tower in an illustration from the Kaufmann Haggadah. Artists in the early 1400s, beginning with Brunelleschi, used mirrors, camera obscura, and other optical devices to aid their understanding of perspective. This understanding was reflected in art until the 20th century, when inspired by the theory of relativity, artists such as Picasso broke from the confines of linear perspective to integrate the temporal view of a scene as a nonlinear projection.

Figure 1: A nonlinear projection rendering from *Ryan,* designed with our interactive system

Linear perspective has the primary advantage of being a simple and approximate model of the projections associated with both real cameras and the human visual system. The model also provides simple, consistent, and easily understood depth cues to the spatial relationships in a three-dimensional scene. From a mathematical standpoint, the pinhole camera model is a linear transformation that provides an efficient foundation for current graphics pipelines within which rendering issues such as clipping, shadowing, and illumination are well understood. While a linear perspective view is a robust medium for viewing localized regions of a scene, it can be restrictive for the visualization of complex shapes and environments.

This work has been inspired by two pieces of concept artwork from the animated production *Ryan,* in which deviations from linear perspective are used to convey cinematic mood and a character's state of mind. Given that humans have a strong mental sense of linear perspective, subtle variations in perspective provide an animator with the ability to generate a sense of uneasiness in the audience to reflect the mood within the animated environment. Larger deviations from a linear perspective can be used to affect the sense of space or convey a feeling of lightness in the animation. Figure 3 is a preproduction sketch, and like most artwork created before 1400, shows a mix of projections used to view different parts of a scene. Figure 4 is an artist-created composite image of 3D deformations and multiple projections of the same scene. Evident in these two concept pieces are the qualities of global scene coherence and local distortions of geometry and shading resulting from changes in perspective. These characteristics become the design principles for our approach to representing nonlinear projections as a combination of distortion-inducing linear perspective cameras and constraints that maintain global scene coherence.

In addition to this visual characterization, the authoring approach should be intuitive to an animator experienced with conventional

*e-mail: patrick@dgp.toronto.edu

†e-mail: karan@dgp.toronto.edu

Figure 2: History of perspective in art

methods of animating a single linear perspective camera. While viewing the scene through this primary camera, the animator should thus be able to add or remove scene distorting cameras that turn the primary camera view into a nonlinear projection. As the nonlinear projections need not be applied to all scene elements, the animator needs the ability to interactively specify various scene constraints to preserve overall scene coherence and to create a desired shot composition.

We now present a brief description of how our nonlinear projection system fits these problem characteristics. A conventional animation workflow uses a single perspective camera for setting up and animating a shot: we refer to this as the *boss* camera. *Lackey* cameras are added as needed to represent different target linear views of scene elements. The animator can chain lackey cameras to specify a projective path from the boss camera view to the target view. The parts of a scene that a particular lackey camera influences will be deformed to appear in the boss camera view as though viewed by that lackey camera. The animator can also add constraints on the position, size, and depth of deformed parts of a scene to better control composition. Finally, the animator can control the resulting illumination and shading of the scene as a combination of rendering parameters of the boss and lackey cameras. Illumination with respect to the single view of the boss camera ignores effects of the alternate views. This results in an appearance discrepancy between the local regions of the nonlinearly projected image and the linear perspective projections used by the animator to define the nonlinear projection. We introduce two methods for incorporating the multiple views into illumination calculations, and compare these with the single view illumination model. While both are appropriate, we argue for the use of the model that is both more predictable with respect to controlling multiple linear perspective cameras and has

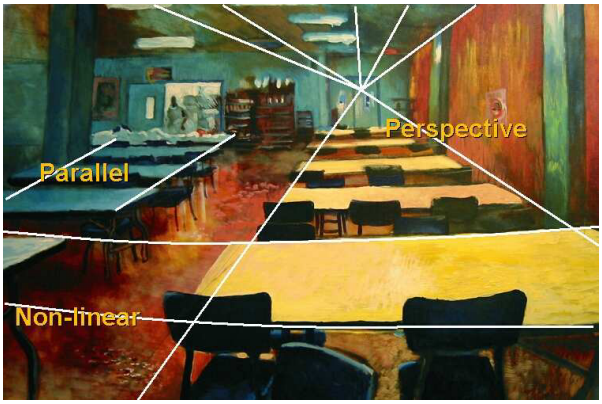


Figure 3: Preproduction artwork for *Ryan* incorporating an artistic combination of projection techniques

stylistically similar effects to those of nonlinear projection (Figure 5).

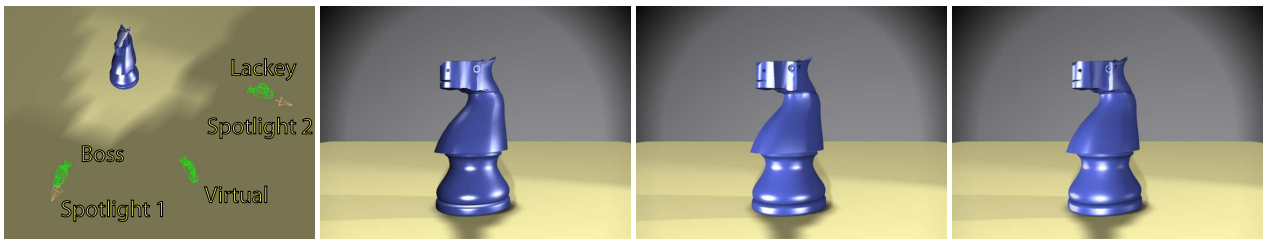
This paper presents the design and implementation of these concepts within the animation system *Maya*, thus making three contributions. First, we present an approach to interactively distorting scene geometry so that when viewed through a standard linear perspective camera, the scene appears nonlinearly projected. Second, we describe a framework that integrates multiple linear perspective views with scene constraints into a single nonlinear projection of the scene. Third, we address the impact of nonlinear projection on rendering and explore various illumination effects.

1.1 Previous Work

Researchers have applied nonlinear projection in computer generated imagery for a variety of purposes. These include image warping, 3D projections, and multi-perspective panoramas. Singh[2002] presents a good survey, but does not describe how these approaches address rendering aspects such as clipping, shadows, and illumination of nonlinearly projected scenes. We give an overview of this here, followed by a discussion of the work of Agrawala et. al.[2000] and Singh[2002], which is of most relevance to this paper.

Image warping techniques[Fu et al. 1999] are inherently two-dimensional approaches with limited ability to explore different viewpoints. View morphing[Seitz and Dyer 1996] addresses the interpolation of a viewpoint in images to provide morphs that have a compelling three-dimensional look. Control over illumination, however, is tied to the given images, resulting in artifacts such as shifting shadows on view interpolation. Approaches that correct for perceived distortions in images[Zorin and Barr 1995], due to curved screens[Max 1983], or resulting from off-axis viewing[Dorsey et al. 1991] modify the geometric projection of pixels by varying their relative size and position, but leave the perceptual view direction and illumination unchanged. View dependent distortions to three-dimensional scene geometry for animation and illustration[Martín et al. 2000; Rademacher and Bishop 1998] are rendered correctly since the intent is to deform geometry and not the viewpoint. Abstract camera models that employ nonlinear ray tracing[Barr 1986; Wyvill and Mcnaughton 1990; Glassner 2000] render scenes correctly, but can be difficult to control by artists and are not well suited to interactive rendering. Multi-perspective panoramas capture three-dimensional camera paths into a single image [Wood et al. 1997; Rademacher 1999; Peleg et al. 2000]. While these approaches render correctly, they provide little control over varying the importance and placement of different objects in a scene and are also not well suited to interactive manipulation. Levene describes a framework for incorporating multiple non-realistic projections defined as radial transformations in eye-space[Levene 1998]. Illumi-

Figure 4: A preproduction composite of multiple nonlinear deformations



(a) Camera setup (b) Boss camera shading (c) Virtual camera shading (d) Blended shading

Figure 5: Illuminating a nonlinear projection

nation calculations are carried out independently of projection, as the projections are defined relative to a single view.

Agrawala et al.[2000] present a multi-projection approach where each object in the scene is assigned to some camera and rendered based on the linear perspective of that camera. The multiple renderings are composited to generate the final image using a visibility ordering of the objects from some master camera view. Position and size constraints for objects in the composite rendering can be specified. Objects are illuminated with respect to their individual perspective and are then composited using a depth buffer. This approach provides good results for multiple discrete projections but does not handle projections continuously varying over objects as seen in Figure 1 or Figure 10d.

The idea of constructing a nonlinear projection as a combination of multiple linear perspectives was presented by Singh[2002]. In the paper, viewports of a number of exploratory linear perspective cameras are laid out on a common canvas onto which the nonlinear projection of the scene is rendered. Each exploratory camera influences different regions in the scene based on local weight values. These are computed as functions of parameters such as distance of the point from the camera's center of interest. The weights define a virtual linear perspective camera for each deformed point, which is computed using a weighted average of the exploratory cameras' parameters. The point is projected using the virtual linear camera and a weight interpolated viewport onto the canvas. All points in the scene are assumed to be influenced by some exploratory camera, and it is unclear how geometry outside the canvas might be culled or clipped. The paper also focuses singularly on geometric projection and does not specify how to illuminate the geometry projected onto the canvas. As presented, Singh's approach does not integrate well into a conventional animation workflow or have ways to control global scene coherence. We, on the other hand, use the notion of exploratory linear perspective cameras but use them in conjunction with scene constraints to induce distortions of geometry and shading such that the view from within a conventional linear perspective camera appears nonlinearly projected. Furthermore, we address both geometric and rendering issues within our nonlinear projection framework.

1.2 Overview

The next section presents our nonlinear projection model, whereby objects are deformed to appear as nonlinearly projected when viewed from a given linear perspective camera. Section 3 then addresses rendering issues in relation to the model proposed in Section 2. Section 4 discusses the implementation of these concepts within the animation system *Maya*, and Section 5 concludes with a discussion of the results obtained.

2 Model for Nonlinear Perspective

In our new approach, we elevate one of Singh[2002]'s exploratory cameras to the status of boss camera; this camera represents the default linear perspective view used in the animation. All other exploratory or lackey cameras, when activated, deform objects such that when viewed from the boss camera, the objects will have some view properties of the lackey cameras.

Let C_b, M_b, V_b represent the eye-space, perspective projection, and viewport transformations, respectively[Foley et al. 1993], for the *boss* camera. Let C_i, M_i, V_i similarly represent the eye-space, perspective, and viewport transforms for *lackey* camera $i \in 1,..,n$. $< x, y, z >= PC_bM_b$ represents the linear projection of a point P into the boss camera's canonical space $x \in [-1,1], y \in [-1,1], z \in [0,1]$. The resulting point in two dimensional screen space $< x_s, y_s >$ is $< x_s, y_s, z_s >= PC_bM_bV_b$. Usually, $z_s = z$ is the depth value of the point P, unchanged by a viewport transform. Here, however, the canonical depth of a point $z \in [0,1]$ is linearly mapped to z_s in an arbitrary user specified range. While the relative depth values are preserved with respect to a single perspective view, users can alter the relative depths of points when transitioning from one perspective view to another by adding a depth offset to the viewport transformations.

Suppose we want to deform a point P in space, such that when viewed through the boss camera, it would appear as if it were being viewed by the ith lackey camera. The deformed point P' is given by:

$$P' = PC_iM_iV_i(C_bM_bV_b)^{-1}. \quad (1)$$

Typically, the ith *lackey* camera only partially influences the point P based on a weight value w_{iP}; the deformed point P is then $P + P(w_{iP}(A_i - I))$, where I is the identity transform and $A_i = C_iM_iV_i(C_bM_bV_b)^{-1}$ (the lackey deformation transform shown in Equation 1). Transforms can be interpolated as described by Alexa[Alexa 2002] or by a linear blend $P' = P + w_{iP}(PA_i - P)$. Singh[2002] advocates the construction of a virtual camera as providing more intuitive results in the general case, but the weighted blend allows for the efficient calculation necessary to handle complex scenes. In practice, it provides good results for all but extreme deformations; such cases can be handled by chained lackey cameras. The results of the multiple lackey cameras' projections are accumulated so that any point P is deformed to:

$$P' = P + \sum_{i=1}^{n} P(w_{iP}(A_i - I)). \quad (2)$$

The following subsections address three important issues relating to the control and usability of our nonlinear projection model: constraints, camera weight computation, and chained lackey cameras.

(a) With constraints

(b) Without constraints

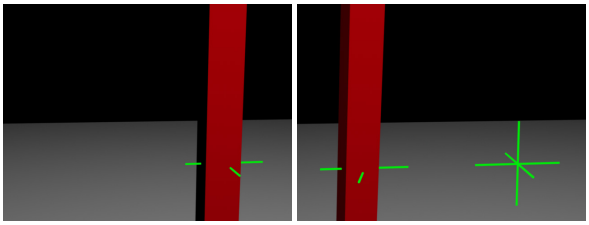
Figure 6: Removal of scene constraints: wall and ceiling collapse into scene

2.1 Constraints

Agrawala et. al.[2002] demonstrate that for multiple linear projections, it can be desirable to constrain objects in space to preserve their relative position and size in a composited scene. They handle these constraints with a translation and scale in screen space after the object has been projected. Singh[2002] allows a user to control the relative position and size of camera projections through viewport transformations within the canvas.

Figure 6 shows the importance of constraints in our system. The removal of scene constraints causes the table on the left to undergo a large vertical translation due to the differing positions of the lackey camera defining its projection and the boss camera. The ceiling and back wall cave into the undeformed portion of the scene for similar reasons. In practice, selective nonlinear projections of complex scenes are easy to mangle without a number of constraints to lay out shot composition in screen space.

We define a spatial constraint matrix Con using two reference frames R_f and R_t, represented as 4x4 matrices. We would like the to see R_f as seen through the ith lackey camera to have the size, position and orientation of R_t when seen through the boss camera.



(a) Pillar, R_t (lackey view)

(b) Constraint deformed pillar, R_t, R_f (boss view)

Figure 7: Constraint setup

The resulting spatial constraint matrix[1] is:

$$Con = (Cartesianize(R_f C_i M_i V_i))^{-1} Cartesianize(R_t C_b M_b V_b) \quad (3)$$

The resulting deformation transform for the lackey camera with a constraint is similar to Equation 1, but with the constraint matrix appropriately inserted:

$$A_i = C_i M_i V_i (Con)(C_b M_b V_b)^{-1}. \quad (4)$$

Con is most often a per object constraint defined for all lackey cameras, but it can also be global for all objects or even defined on a selective basis per object per lackey camera.

Figure 7 demonstrates the use of a position constraint on a pillar seen from an alternate point of view. Figure 7a shows the original pillar geometry from the lackey camera's point of view, as well as a reference frame R_f that indicates a positional constraint on the geometry. Figure 7b shows the column deformed to have the projective appearance of the lackey camera's point of view, but seen from the boss camera, which is located to the right of the lackey camera. Without application of the constraint, the column would appear at the same location in screen space in each view. The additional reference frame R_t indicates the deformed position of the constrained point, and the constraint effects the image space transformation necessary to hold the column in place relative to R_f.

For complex objects it might be necessary to define multiple constraints. Points on the object are constrained to proximal reference frames. Formally stated, a set of constraints $Con_1,..,Con_m$ are defined using frames $R_{f1},..,R_{fm}$ and $R_{t1},..,R_{tm}$. The constraint matrix $Con(P)$ for a point P is defined using frames $R_f(P)$ and $R_t(P)$. $R_f(P)$ and $R_t(P)$ are computed as weighted interpolations of frames $R_{f1},..,R_{fm}$ and $R_{t1},..,R_{tm}$, respectively. The weight for constraint j is inversely proportional[2] to the Euclidean distance from P to the origin of frame R_{fj}. We precompute $Apre_i = C_i M_i V_i$ and $Apost_i = (C_b M_b V_b)^{-1}$ to represent the deformation of a point P, combining Equations 2 and 4 as:

$$P' = P + \sum_{i=1}^{n} P(w_{iP}((Apre_i)(Con(P))(Apost_i) - I)). \quad (5)$$

2.2 Camera Weight Computation

Figure 8 illustrates a number of parameters introduced by Singh[2002], which can be used to calculate the influence weights of cameras. These include camera direction, the center of interest, and user painted weights.

[1] Cartesianize represents the effect of a perspective divide such that the constraint transformation is affine.

[2] To avoid division by zero, the weight for constraint j is $1/(1+d)$, where d is the distance from P to the origin of R_{fj}.

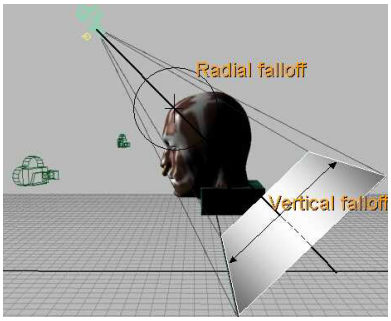


Figure 8: Camera weight computation

In addition, the surface normal, curvature, and other attributes can be as important as surface position in determining a camera's influence on the surface. Figure 9 shows weight computation based on the facing ratio of a point, i.e. the angle its surface normal subtends with the optical axis of a lackey camera. In this image, lackey cameras are used to demonstrate an artificial rim lighting effect from multiple viewpoints without distorting the scene geometry.

2.3 Chained Lackey Cameras

Defining a weight interpolated virtual camera for each point offers the advantage of a potentially better interpolation of the angular parameters of the camera model[Singh 2002]. For complex scenes such as seen in Figure 14, recomputing a different virtual camera transformation for each control point on an object can be expensive, detracting from the system's interactive capabilities. In practice, we find that blending projected points provides good visual results, and the matrix precomputations shown in Equation 5 allows for efficient deformation. Better interpolation than a linear blending of projected points can be computed by either using a better matrix interpolation scheme such as that described by Alexa[Alexa 2002] or by creating a chain of in-between lackey cameras that define the interpolation path from the boss camera to a target lackey camera. Chained lackey cameras also provide an animator with greater control over the illumination blending described in the next section.

3 Rendering the Nonlinear Projection

The previous section presented the methods by which we deform geometry to appear as a nonlinear projection from the boss camera's point of view. To correctly display a nonlinear projection, all aspects of the display pipeline must be addressed, including not only geometric projection, but also culling and clipping, shadowing, and illumination. Furthermore, these steps of the display pipeline need to adhere to a model that maintains meaningful image coherence over time without introducing visual distractions from the story.

3.1 Geometric Culling

We have implemented nonlinear projection of geometry as the linear perspective projection of deformed scene geometry. The linear perspective camera through which the scene is viewed thus handles culling and clipping of objects automatically within the existing graphics architecture. In a dedicated nonlinear projection pipeline, or if our framework were implemented as a projection "shader," care should be taken to consider the final projected position of the geometry in the culling decision process.

In practice, an animator constructs nonlinear projections while considering a short series of frames and the geometry visible within

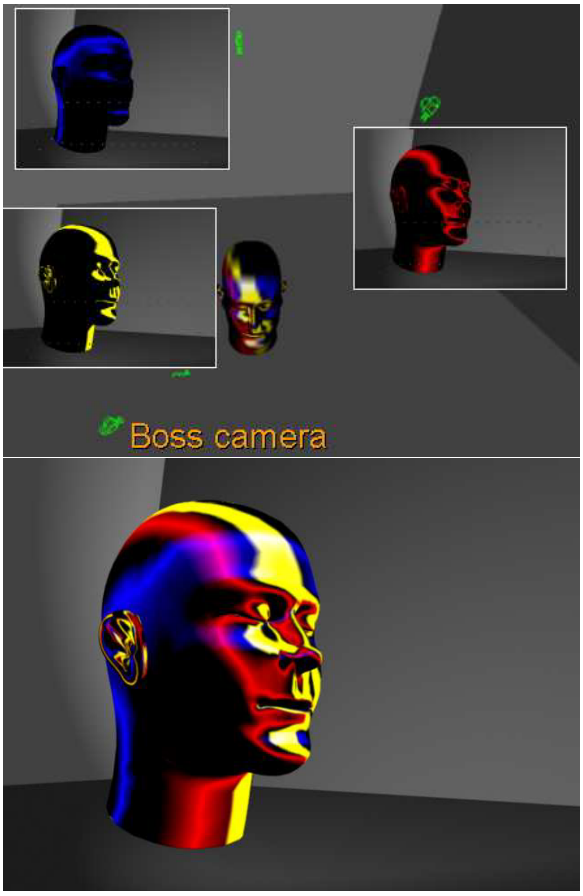


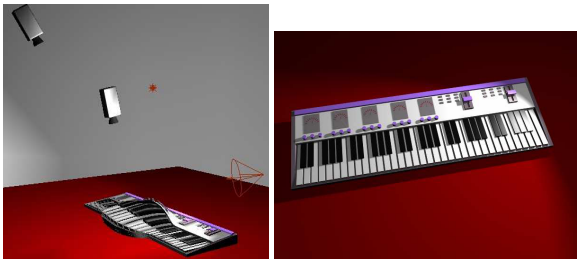
Figure 9: Surface normal based weight computation. Setup (top), Blended illumination (bottom).

the boss and lackey cameras' fields of view. However, geometry affected by a particular lackey camera will remain affected by that camera at frames in the shot temporally distant from where the animator is working. As a result, objects which are deformed to effect a meaningful projection at one point in time might later be subject to unintended large deformation, especially when near the plane through the camera perpendicular to its view vector. This can result in objects deforming to appear in the shot when not intended. To avoid this artifact, deformation is attenuated beyond a threshold distance outside the viewing frustum of the boss camera. This threshold distance may be modified by the animator, as the region of space for which deformation is necessary is dependent on the scale of the deformation. An animator may selectively disable the attenuation when using a projection to intentionally bring a distant object into view.

3.2 Shadows

Nonlinear projection models the perception of a scene, and does change the scene itself. In our projection approach, deformation is only a means of effecting the desired projection; hence, shadows calculated using this deformed geometry are meaningless. In a static image, this results in unwanted shadowing. Figure 10c demonstrates this effect. Even subtle perceptual changes created with a nonlinear projection can result in large geometric deformations. Furthermore, in an animated scene, the shadows cast by deformed geometry will move with distracting speed as the cameras move. To correct this effect, the original geometry should be used

for shadow calculations. Precomputed shadow depth maps used in multi-pass rendering should be generated with nonlinear projection disabled, and the determination as to whether a surface point is in shadow should reference the corresponding point on the undeformed surface. Ray traced shadow calculations in off-line renderers can use the undeformed geometry in all shadow ray calculations. After correcting the shadow calculations, the keyboard appears as in Figure 10d. The distracting self shadowing is no longer present, and only projective effects remain (illumination is also calculated appropriately as described in the next section).



(a) Camera setup

(b) Boss camera view

(c) Nonlinear projection (wrong shadows and shading)

(d) Nonlinear projection (correct shadows and shading)

Figure 10: Shadows

3.3 Illumination

Many global and local illumination calculations are view dependent. A nonlinear projection will affect these computations, as the view vector at a surface point is now dependent on the local projection. As our perspective model is based on a linear combination of linear projections, we define a similar illumination model. Regions of a scene projected from a single viewpoint are simple; the view vector derived from the corresponding lackey camera is used. Regions of interpolated projections are illuminated using one of two approaches:

1. The viewpoint is that of the weight interpolated virtual linear perspective camera through which the given point is projected. The point is illuminated with respect to this viewpoint.

2. There is no single viewpoint. In this case, the point is illuminated with respect to the boss camera and each contributing lackey camera viewpoint. The illumination results are then blended together using a normalized weight vector proportional to the weight contributions of the lackey cameras. As with shadow calculations, all illumination values are calculated with respect to the undeformed geometry.

In practice, we find that animators prefer the latter concept, as they are able to better predict the expected illumination by looking at the projection through the boss and various lackey cameras. Further, chained lackey cameras allow us to combine these two ideas into a single model. In addition, blending illumination calculations allows for a greater variety of surreal effects. Figure 11 shows an example of this flexibility, where no single viewpoint would be capable of creating the dual views of the character seen reflected in the sphere. In this particular example, the illumination model is isolated from the projection model, and projection is disabled.

As a comparative example, Figure 5 shows three variations of illumination for an object viewed with two cameras. The object has been deformed such that when viewed from the boss camera, it appears as an equally weighted combination of projections to the boss camera's view and to the lackey camera's view. Figure 5a shows the layout of the cameras, the undeformed geometry, and two spotlights used for illumination. The virtual camera represents an interpolated viewpoint.

In Figure 5b, the geometry is illuminated with respect to the boss camera viewpoint. Note the two highlights: one directly in front of the viewer reflecting spotlight 1 and one halfway to the region directly illuminated by spotlight 2, with no illumination effects due to the presence of the lackey camera. Figures 5c and 5d show two methods by which the illumination from the lackey camera might be incorporated. In Figure 5c a virtual camera representing an interpolation between the boss and lackey cameras is used as the viewpoint in the illumination calculations for the entire object, resulting in the two modified highlights. In Figure 5d, the object is illuminated with respect to both the boss and lackey cameras, and the results are blended, resulting in four attenuated highlights, although two are close enough to appear as single stretched highlight. Applying this illumination technique to stylized shaders allows for the creation of images such as seen in Figure 9.

4 Implementation

This section describes the implementation of these concepts written as a plug-in to the animation system *Maya*. We first describe the interface to the system, adapted from the approach presented by Singh[2002]. The deformation and rendering system as incorporated into *Maya* is then presented.

4.1 User Interface

In our system, a user animates a scene with a traditional linear perspective camera–the boss camera. Current animation systems such as *Maya* allow users to create, manipulate, and simultaneously view any number of linear perspective cameras. Within our system, the user can at any time label any of these cameras as lackey cameras

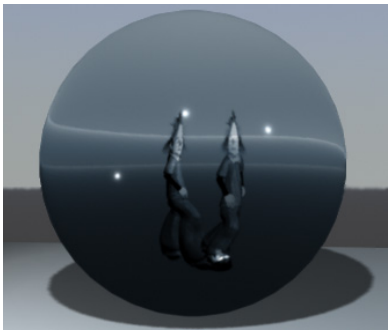


Figure 11: Dual reflections using blended illumination

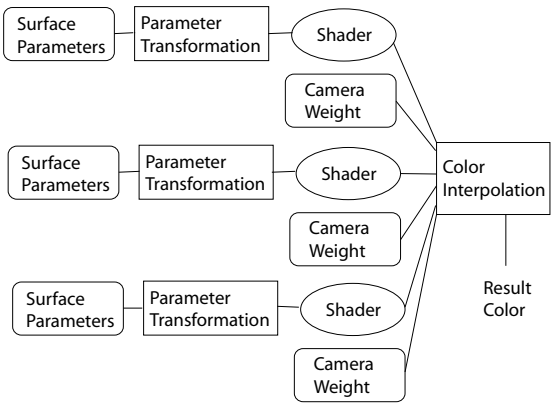


Figure 12: Modifications to the shading pipeline. Rounded blocks represent surface parameters provided by the renderer, and rectangles represent new shading functions.

associated with the boss camera. Adding or manipulating lackey cameras appropriately updates the deformation of scene geometry to result in a nonlinear projection as seen from the boss camera. An attenuation control on the geometric deformation magnitude allows the user to switch or even blend between the nonlinear projection and the boss camera's linear perspective view.

When creating and editing nonlinear projections, users typically require an exploratory bird's eye view of the scene that shows the overall spatial relationship among the scene, boss camera, and lackey cameras. All cameras except the boss camera view the undeformed scene, since the deformation only has visual meaning from the boss camera's point of view. The individual lackey camera views allow the user to visualize the effect of each camera on the nonlinear projection.

To modify relative depths of different views and apply the viewport transformations, the user interactively manipulates a filmbox cube coincident in space with each lackey camera. Affine transformations applied to this filmbox map directly to the viewport transformation matrix used for projection. Translating or scaling the film box along the camera's viewing axis modifies the camera's projection depth relative to other lackey cameras without changing the image space location of projected scene elements.

Users typically create new lackey cameras coincident with the boss camera with no viewport transformation as a starting point from which to manipulate the lackey cameras. Lackey camera manipulation can dramatically alter the composition of scene objects in the image. Unless a single nonlinear projection is applied uniformly across the scene, constraints are necessary to tie objects to locations in the final image. Therefore, our system creates a default constraint at the center of each group of objects and each lackey camera with which they are associated. Manipulating the camera thus alters the projection in the boss camera view while ensuring that the constraint maps to its location in the boss camera's linear perspective.

Constraint reference frames are represented as cross-hair objects (*locators* in *Maya*), visible in Figure 7. Each pair of these objects defines the constraint's reference frames and can be interactively manipulated as with any scene object. New constraints can be created for a group of objects, and they are typically specified uniquely for each lackey camera. The constraints are largely responsible for allowing nonlinear projections to be easy to control, allowing the overall image composition to remain coherent and predictable. Constraints also provide a direct method for altering the size, position, orientation, or depth of a local region of the scene. The viewport transform does not allow such local changes, as it acts globally for a particular camera. As such, we allow constraints to be defined at any point during the workflow.

4.2 Rendering

The nonlinear projection is rendered by interactively deforming the scene geometry and viewing it through the underlying linear perspective of the boss camera. The illumination and shading calculations are implemented assuming a typical shading pipeline in which local surface parameters are provided by the renderer. An arbitrary shader calculates the shading and illumination from these parameters and any number of additional parameters that are constant across the surface. The Maya rendering architecture also supports interpolation of user-defined parameters across surfaces, which is necessary for our particular implementation, although the parameters we interpolate could also be calculated directly by the shader. We have constructed two steps in the shading process (analogous to functions within a conventional shading language) that support our illumination model. As shown in Figure 12, the renderer provides surface parameters, which we partially replace with the surface point and normal of the undeformed geometry using a reference to the original surface. This takes place for each camera, and the parameter modification also replaces the view vector with that of the corresponding camera. These new values override the renderer provided parameters to an arbitrary surface shader, thus providing a surface color appropriate for each camera. Another step interpolates among these colors using the weights calculated for projection interpolation, and the resulting color is used to shade the surface. The camera weights are stored as parameters that the renderer interpolates, which are then accessible to the shader. This approach is particularly useful in that any existing surface shader can be used with our model, rather than having to construct specialized versions of each shader.

Ghosting effects as seen in Figure 4 are created by duplicating the deformed geometry as nonlinear projection parameters are varied and then rendering the multiple instances of geometry in the scene with varying opacity.

5 Results and Conclusion

This paper presents a comprehensive system for constructing and rendering nonlinear projections appropriate for use in a production environment. The system has been in use in the production of the animation *Ryan,* demonstrating its artistic and practical usefulness. Figures 14 and 15 show stills from animation tests that employ nonlinear projections to distort the scene. Figure 1 shows a production quality rendering incorporating nonlinear projection. In Figure 13, three blocks have been independently projected to reveal features otherwise not visible. The upper portion of the block on the left is seen from a viewpoint to the left, the center of the middle block is seen from a viewpoint looking in from the right side, and the top of the right block is projected according to an elevated point of view looking down. Projections such as this can be exceptionally useful in creating renderings of complex objects that reveal features not visible from any single linear perspective, while maintaining a coherent visualization. The blended illumination model can also be used independently to craft interesting effects, as demonstrated in Figures 9 and 11.

In our approach, animators work almost exclusively with the boss camera for shot composition, but switch among lackey camera views to collect ideas for constructing the nonlinear projection. The ability to gradually apply the existing nonlinear projection to the underlying linear perspective has also proven valuable in both understanding an existing projection during authoring and as a means of subtly introducing nonlinear perspective effects as a shot

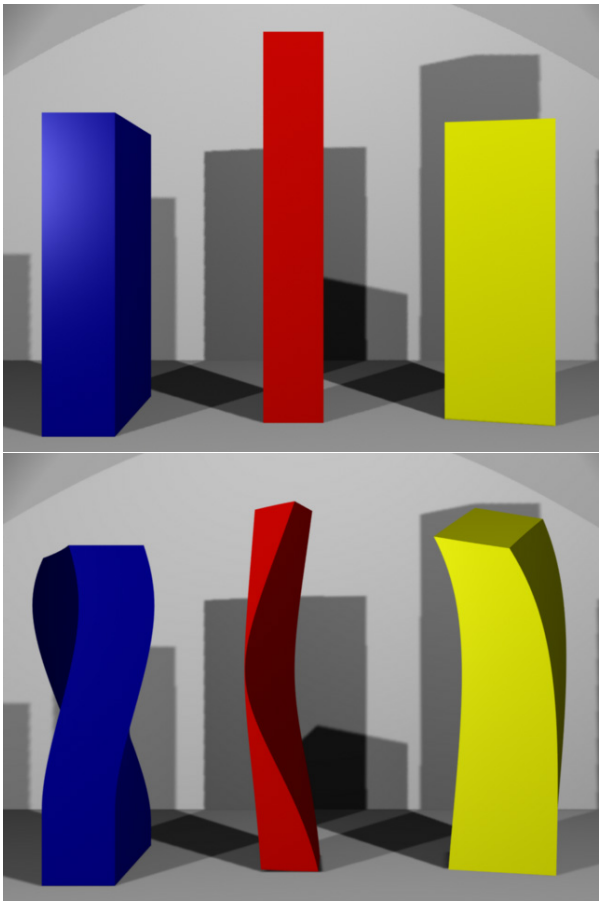


Figure 13: Bringing occluded regions into view with nonlinear perspective

progresses. We recognize, however, that manipulating many cameras can be a complicated task. The development of higher level techniques for manipulating multiple cameras is a subject of future work.

In summary, this paper presents a new formulation for interactive nonlinear projections that addresses spatial scene coherence, shadows, and illumination, as well as their integration into current production pipelines. Practical methods of constructing various nonlinear projection effects are shown. Our results showcase the use of our technique in the commercial animation production *Ryan*.

6 Acknowledgments

Chris Landreth and the *Ryan* crew provided the sets and models for many of the images in this paper. Aaron Hertzmann and Michael McGuffin provided feedback on an early draft. Alias provided the *Maya* animation software for both research and production.

References

AGRAWALA, M., ZORIN, D., AND MUNZNER, T. 2000. Artistic multiprojection rendering. In *Proceedings of Eurographics Rendering Workshop 2000*, Eurographics, 125–136.

ALEXA, M. 2002. Linear combination of transformations. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, ACM Press, ACM, 380–387.

BARR, A. H. 1986. Ray tracing deformed surfaces. In *Proceedings of the 13th annual conference on Computer graphics and interactive techniques*, ACM Press, ACM, 287–296.

DORSEY, J. O., SILLION, F. X., AND GREENBERG, D. P. 1991. Design and simulation of opera lighting and projection effects. In *Proceedings of the 18th annual conference on Computer graphics and interactive techniques*, ACM Press, ACM, 41–50.

FOLEY, J., VAN DAM, A., FEINER, S., AND HUGHES, J. 1993. *Computer Graphics: Principles and Practice*. Addison Wesley.

FU, C.-W., WONG, T.-T., AND HENG, P.-A. 1999. Computing visibility for triangulated panoramas. In *Proceedings of Eurographics Rendering Workshop 1999*, Eurographics, 169–182.

GLASSNER, A., 2000. Cubism and cameras: Free-form optics for computer graphics. Microsoft Research Technical Report MSR-TR-2000-05, January.

LEVENE, J. 1998. *A Framework for Non-Realistic Projections*. Master's thesis, Massachusetts Institute of Technology.

MARTÍN, D., GARCÍA, S., AND TORRES, J. C. 2000. Observer dependent deformations in illustration. In *Proceedings of the first international symposium on Non-photorealistic animation and rendering*, ACM Press, ACM, 75–82.

MAX, N. L. 1983. Computer graphics distortion for imax and omnimax projection. In *Nicograph '83 Proceedings*, Nicograph Association, 137–159.

PELEG, S., ROUSSO, B., RAV-ACHA, A., AND ZOMET, A. 2000. Mosaicing on adaptive manifolds. *IEEE Transactions on Pattern Analysis and Machine Learning 22*, 10, 1144–1154.

RADEMACHER, P., AND BISHOP, G. 1998. Multiple-center-of-projection images. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, ACM Press, ACM, 199–206.

RADEMACHER, P. 1999. View-dependent geometry. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., ACM, 439–446.

SEITZ, S. M., AND DYER, C. R. 1996. View morphing. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, ACM Press, ACM, 21–30.

SINGH, K. 2002. A fresh perspective. In *Proceedings of Graphics Interface 2002*, 17–24.

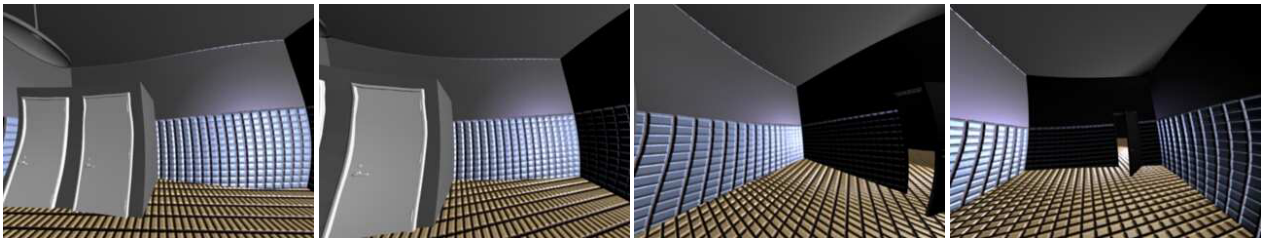
WOOD, D. N., FINKELSTEIN, A., HUGHES, J. F., THAYER, C. E., AND SALESIN, D. H. 1997. Multiperspective panoramas for cel animation. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., ACM, 243–250.

WYVILL, G., AND MCNAUGHTON, C. 1990. Optical models. In *Proceedings of CGI 1990*.

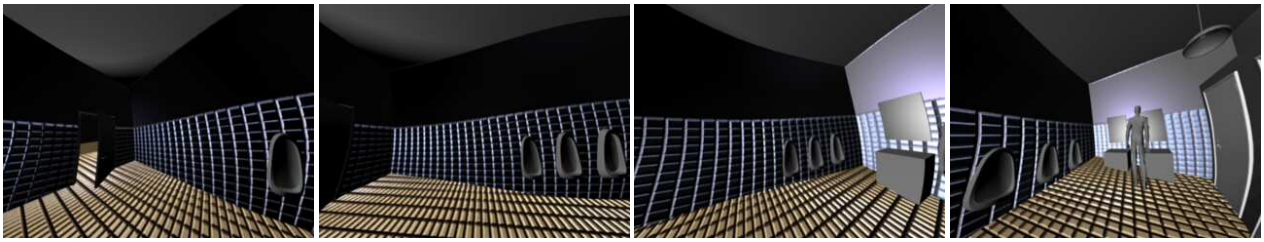
ZORIN, D., AND BARR, A. H. 1995. Correction of geometric perceptual distortions in pictures. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, ACM Press, ACM, 257–264.

(a) (b) (c) (d)

Figure 14: *Ryan* Cafeteria Set



(a) (b) (c) (d)



(e) (f) (g) (h)

Figure 15: *Ryan* Bathroom Set