University of Toronto , pp. 1–9
(Editors)

RYAN: Rendering Your Animation Non-linearly projected

Patrick Coleman and Karan Singh

Department of Computer Science
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 surrealistic storyboard of an animation project "Ryan", this paper describes interactive techniques to control and render scenes using non-linear projections for use in current production pipelines. The paper builds upon an existing idea of constructing non-linear projections as a combination of multiple linear perspective views. First, we adapt and extend the approach to be usable on complex 3D scenes within a conventional production workflow. We then explore shadowing and illumination for the interactive non-linear projection of scenes, which has not been explicitly addressed by previous work. We show how animators can control both the geometry and shading of scenes using a combination of multiple linear perspective cameras.

1. Introduction

Artists using traditional media almost always deviate from the confines of a precise linear perspective view. Many digital artists on the other hand continue to struggle with the standard pin-hole 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 1 provides good insight into the benefits and limitations of linear perspective. Even though the earliest documented observation of perspective is dated around 4000BC, renderings of 3D scenes as late as 1400, were either flat and lacking depth or showed clear perspective errors as seen on the tower in an illustration from the Kaufmann Haggadah. Artists from the early 1400's starting with Brunelleschi used mirrors, camera obscura and other optical devices to aid their understanding of perspective, strongly reflected in art until the 20th century. Inspired by the theory of relativity in the 20th century, artists like Picasso integrated the viewing of a scene over time with linear perspective into non-linear projections combining space and time.

The advantage of linear perspective is that it is a simple but good approximation to human vision. It is also the simplest form of projection that provides consistently understandable depth cues to parts of a 3D scene. From a mathematical standpoint the pin-hole camera model is a linear transformation that fits right into 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.

This paper is inspired by two concept renderings from an animation production titled "Ryan", where deviations from a linear perspective are used to convey moods in the animation. Given that humans have a strong sense of linear perspective, controlling subtle variations in perspective allows an animator to generate a sense of uneasiness in the audience to reflect the mood within the animation. Similarly larger deviations from a linear perspective can be used to convey a sense of space or lightness in the animation. Figure 2 is a storyboard sketch showing a mix of projections used to view different parts of a scene. Figure 3 is rendering of multiple projections of parts of a scene

composited in a single view. The problem statement thus posed by the requirements of "Ryan" is to begin with a conventional single camera workflow. An animator should then be able to add or remove cameras to induce non-linear projections viewed through the conventional single camera. These non-linear projections manifest themselves as geometric and rendering distortions to the scene to appear as non-linear projections in the given camera. The animator further, needs the ability to specify various scene constraints to have precise control over the non-linear distortions of parts of the scene.

Figure 1: *History of use of perspective in art.*

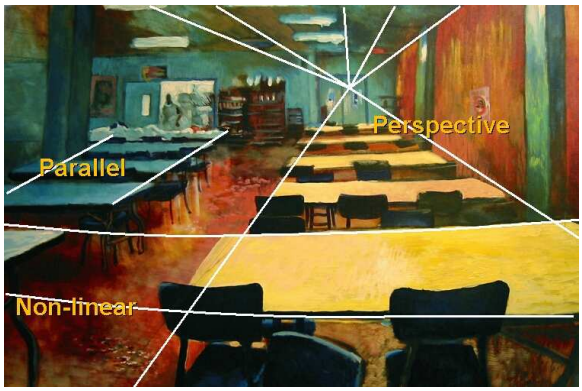


Figure 2: *Artistic mix of projections in Ryan*

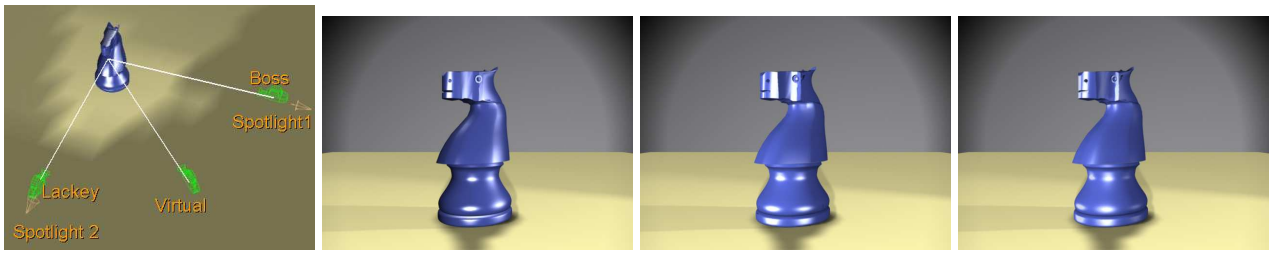
We now give a brief overview of how the non-linear projection model presented later in the paper fits our problem statement. A conventional animation workflow uses a single perspective camera that is used for setting up and animating a shot. We refer to this as the *boss* camera. *Lackey* cameras are added as needed to represent different target linear views for parts of a scene. Lackey cameras may be chained to specify a camera path from the boss camera to a target view. The parts of a scene affected by a lackey camera, and the degree of influence is based on a combination of user selection and automatic camera parameter based

Figure 3: *Compositing multiple non-linear projections*

functions. Parts of a scene influenced by a lackey camera will be deformed to appear in the boss camera as though viewed by the lackey camera. Constraints on the position, size and depth of parts of the scene may be added to control the overall projection of geometry in the scene. Of particular importance to this paper is the efficient calculation of illumination and shading parameters using appropriate combinations of boss and lackey cameras.

Figure 4 shows 3 variations of illumination for an object viewed with two cameras as shown. The object itself has been deformed, so that when viewed through the boss camera, it appears as if it is viewed halfway from the lackey camera. Figure 4a shows the layout of cameras and spotlights. In Figure 4b the geometry is illuminated with respect to the boss camera viewpoint (note the two highlights: on the side from spotlight1 and one along the virtual camera optical axis from spotlight2) ignoring the illumination effects of the lackey camera. Figure 4c,d shows two ways by which the illumination from the lackey camera may be incorporated. In Figure 4c a virtual camera, that is an interpolation between the boss and lackey, is used as the viewpoint in the illumination calculations for the entire object, resulting in the two highlights seen. In Figure 4d the object is illuminated with respect to both the boss and lackey cameras and the results are blended resulting as expected in four distinct but attenuated highlights.

This paper presents the design and implementation of these concepts within the animation system *Maya*. The contributions of this paper are twofold. First, we extend and adapt ideas presented by Singh[12] and Agrawala et. al.[1] into a system that seamlessly fits existing animation production pipelines. Second, we address the illumination of scenes interactively rendered using non-linear projections.



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

Figure 4: *Illuminating a non-linear projection*

1.1. Previous work

Non-linear projections have been applied in computer generated imagery for a variety of purposes such as image warping, 3D projections, and multi-perspective panoramas, a good survey of which is presented by Singh[12].

Of particular relevance to this paper is how this prior work addresses rendering aspects such as clipping, shadows and illumination of non-linearly projected scenes. Image warping techniques[6] are inherently 2D approaches with limited ability to explore different viewpoints. View morphing[13] addresses the interpolation of a viewpoint in images to provide morphs that have a compelling 3D look. Control over illumination, however, is tied to the given images, resulting in artifacts such as shifting shadows on view interpolation. Approaches[8, 4, 16] that correct for perceived distortions resulting from curved screens or off-axis viewing, correct the geometric projection of pixels by varying their relative size and position, without addressing changes in their illumination that may have occurred due to the change in viewpoint or viewing model. View dependent distortions to 3D scene geometry for animation and illustration[7, 11] are rendered correctly since the intent is to deform geometry and not the viewpoint. Abstract camera models that employ non-linear raytracing[3, 15] render scenes correctly but are difficult to control by artists and are not well suited to interactive rendering. Multi-perspective panoramas capture 3D camera paths into a single image [14, 10, 9]. 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.

Agrawala et al.[1] 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. The visibility ordering is done with respect to a given master camera view. Conflicts in visibility ordering of objects are resolved at the pixel level by depth comparison. Position and size constraints for objects in the composite rendering can be specified. Objects are illuminated correctly with respect to their individual perspective. This approach provides good results for multiple discrete projections but does not handle projections continuously varying over objects as seen in Figure 9d.

This paper builds upon the work of Singh[12], who presents a way to combine a number of exploratory linear perspective views to construct a single non-linear projection. The exploratory cameras have viewports that are laid out on a common canvas onto which the non-linear projection takes place. Each exploratory camera influences different points in the scene based on weight values, computed as functions such as distance of the point from the camera's center of interest. The weights define a virtual linear camera for the point, as a weighted average of the exploratory cameras. 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 may be culled or clipped. The paper is also singularly focused on geometric projection and does not specify how the points projected on the canvas should be illuminated. As presented, Singh's approach does not integrate easily into a scene being animated and viewed through a single linear perspective camera. Turning non-linear projection on would require rendering to switch from the single camera to the non-linear projection canvas that is both an interruption of animator workflow and likely to cause rendering discontinuities as a result of the switch. This paper allows the conventional perspective camera being used in the animation

to be used for non-linear projection and goes further to address scene constraints and rendering issues within this framework.

1.2. Overview

The next section presents our non-linear projection model, where objects are deformed, to appear as non-linearly 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 concludes with a discussion of the results obtained.

2. Model for non-linear perspective

In this paper we elevate one of Singh's exploratory cameras to the status of boss camera, which represents the default linear perspective view being used in the animation. All other exploratory or lackey cameras, when turned on, deform objects so 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[5], 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 cannonical 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 (as illustrated by Singh[12]).

Now 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 would only partially influence the point P based on a weight value w_{iP}, making the deformed point P' be $P' = P + P(w_{iP}(A_i - I))$, 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). The interpolation between transforms can be done as described by Alexa[2], or by a linear blend $P' = P + w_{iP}(PA_i - P)$ when the relative difference between boss and lackey camera is small. The results of multiple lackey cameras 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 non-linear projection model, namely, constraints, camera weight computation and chained lackey cameras.

2.1. Constraints

Agrawala et al.[1] show that for multiple linear projections, it is desirable to constrain objects in space to preserve their relative position and size in a composited scene. They handle such constraints with a translation and scale in screen space after the object has been projected. Singh[12] allows a user to control the relative position and size of camera projections through viewport transformations within the canvas.

Figure 5shows importance of constraints in our system. The removal of scene constraints causes the table to fly off and the ceiling and back wall to cave into the rest of the undeformed scene. In practice non-linear projections of complex scenes are easy to mangle without a number of constraints to lay it out in screen space.

(a) With constraints (b) Without constraints

Figure 5: *Removal of scene constraints: wall and ceiling collapse onto scene*

We define a spatial constraint matrix Con using two reference frames R_f, 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. The resulting spatial constraint matrix Con is:

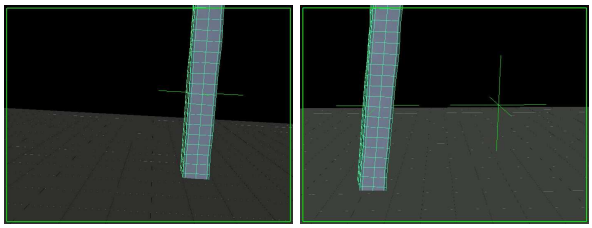
$$Con = (Cartesianize(R_fC_iM_iV_i))^{-1}Cartesianize(R_tC_bM_bV_b)$$

†.
The resulting deformation transform for the lackey camera with a constraint is just like in Equation 1 with the constraint matrix appropriately inserted:

$$A_i = C_i M_i V_i (Con)(C_b M_b V_b)^{-1}. \quad (3)$$

Typically, Con is a per object constraint globally defined for all lackey cameras but can clearly be global for all objects or even selectively defined on a per object per lackey camera basis.

Figure 6 shows the operation of a position constraint on a pillar. Figure 6a shows the original pillar and a reference frame R_f viewed through a lackey camera. Figure 6b shows the same pillar constrained and deformed. It has the projective appearance of the lackey camera view but is moved over to adhere to the constraining reference frame R_t. The reference frame R_f is also shown where the unconstrained pillar would have appeared.



(a) Pillar, R_f (lackey view)

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

Figure 6: *Constraint setup*

For highly complex objects we sometimes need to define multiple constraints for the same object. Points on the object are constrained to the reference frames that are proximal to the point. 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 the jth constraint is inversely proportional to the Euclidean distance from P to the origin of frame R_{fj}. We precompute $Apre_i = C_i M_i V_i, Apost_i = (C_b M_b V_b)^{-1}$ to represent the deformation of a point P (combining Equation 2,3)as:

$$P' = P + \sum_{i=1}^{n} P(w_{iP}((Apre_i)(Con(P))(Apost_i) - I)). \quad (4)$$

2.2. Camera weight computation

Singh[12] introduced a number of parameters based on which the influence weights of cameras could be calculated, including camera direction, center of interest and user-painted weights as shown in Figure 7.

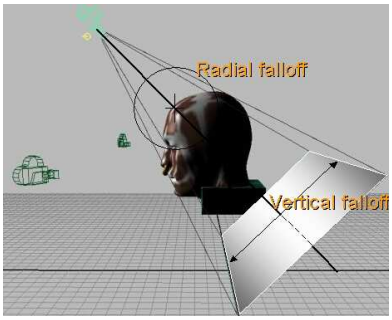


Figure 7: *Camera weight computation*

Normal, curvature and other surface attributes can be as important as surface position in determining a camera's influence on the surface. Figure 8 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 Figure 8 lackey cameras are used to show a rim lighting effect from multiple viewpoints without actually distorting the scene geometry.

2.3. Chained lackey cameras

The advantage behind defining a weight interpolated virtual camera for each point is that the angular parameters of the camera model can be interpolated better[12]. For highly complex scenes such as Figure 11, recomputing a different virtual camera transformation for each control point on an object can be expensive, slowing down the interaction with the system. In practice we find that blending projected points provides good visual results and the matrix precomputations showin in Equation 4 make the overall deformation process highly efficient. Better interpolation than a linear blending of projected points can be obtained either by using a better matrix interpolation scheme such as that described by Alexa[2] or by creating a chain of in-between lackey cameras that define the interpolation path from the boss camera to any given lackey

† Cartesianizing the matrices performs a perspective divide so that the resulting constraint matrix is affine

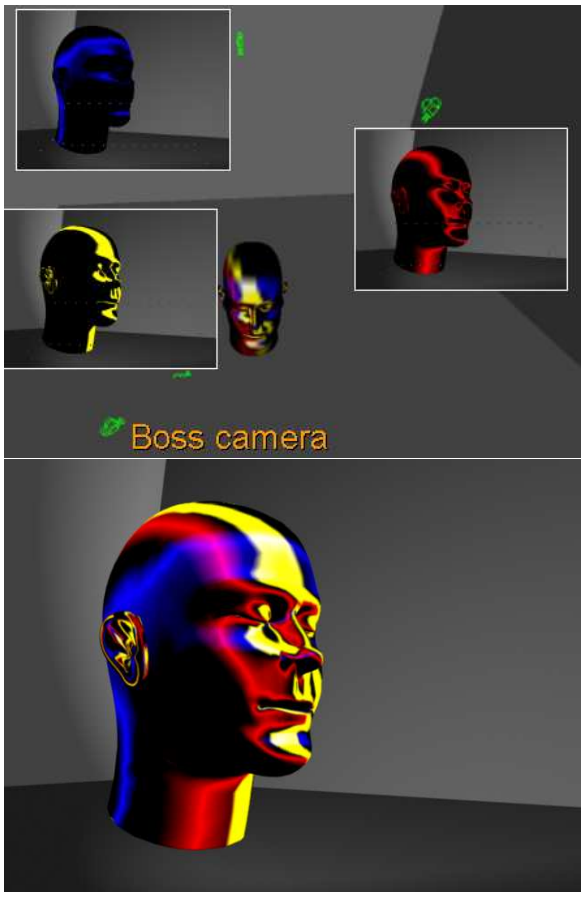


Figure 8: *Surface normal based weight computation. Setup (top), Blended illumination (bottom)*

camera. Chained lackey cameras also provide an animator with control over illumination blending as will be seen in the next section.

3. Rendering the Non-linear projection

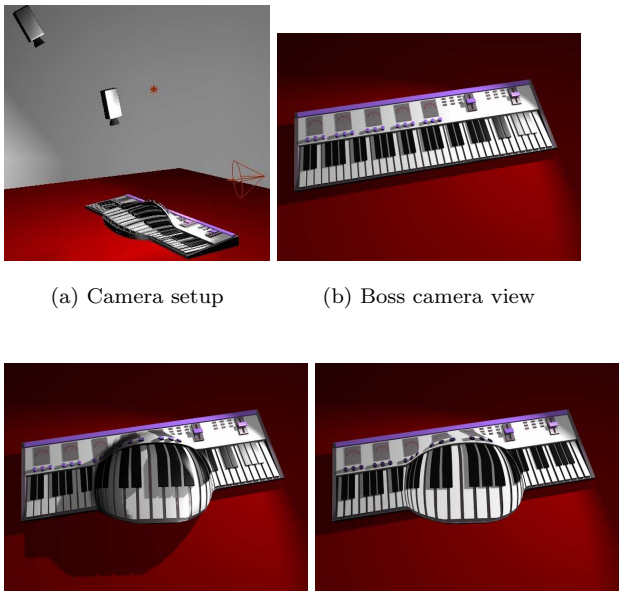
The previous section dealt with deforming geometry so that it appears as a non-linear projection in the boss camera view. To correctly display a non-linear projection, one must address other aspects of the display pipeline, which not only includes projecting the geometry but clipping it to a given view and then illuminating it in a consistent fashion.

3.1. Geometry culling

With our given non-linear projection model, clipping of geometry automatically takes place to the boss camera's viewing frustum. Objects deformed to account for influences of lackey cameras are treated seamlessly while others viewed in strict linear perspective through the boss camera, which is the default behavior for objects outside the influence of any lackey camera.

3.2. Shadows

Shadowing in a scene is view independent and should be computed using the scene with objects in their undeformed state. Figure 9 shows problems resulting from using the deformed objects to compute shadows. Often the objects need to be grossly distorted to appear as projected using a lackey camera that is quite different from the boss camera. These distortions cause irregularly shaped and moving shadows (see Figure 9b) as the cameras are animated in the scene. Figure 9c shows the same scene correctly rendered with shadows being cast by the undeformed objects.



(a) Camera setup (b) Boss camera view

(c) Non-linear proj. (wrong shadows and shading) (d) Non-linear proj. (correct shadows and shading)

Figure 9: *Shadows*

3.3. Illumination

Many global and local illumination calculations are view dependent. These computations are clearly affected by a non-linear projection technique that is based on the composition of multiple linear perspective cameras, from various viewpoints. Conceptually there are two ways of perceiving the viewpoint for any point rendered as a non-linear projection.

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.

In practice we find that animators prefer the latter concept, as they are able to better predict the expected illumination by looking at the illumination through the boss and various lackey cameras. Further, chained lackey cameras allow us to combine these two ideas into a single model. In between lackey cameras, typically reduce the separation of viewpoints being blended for any given weight value. Figure 10 shows an example of the flexibility of blended illumination, where no single viewpoint would be capable of creating the dual views of the character seen reflected in the sphere.

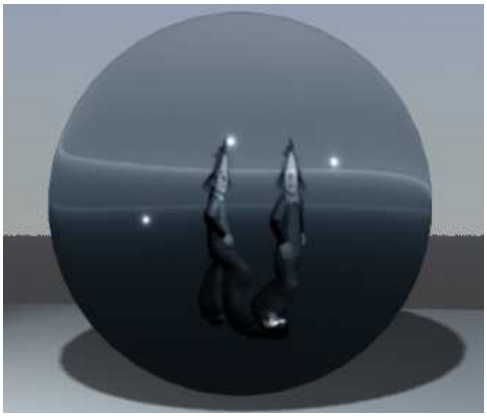


Figure 10: *Dual reflections using blended illumination*

3.4. Implementation

This section describes an implementation written as a plug-in to the animation system *Maya 4.5*.

The basic user interface framework is as described in[12] and can be seen in Figure 9, where an animator has the freedom of looking at the scene through the boss or lackey cameras or a global view that shows the overall spatial relationship between the scene, boss, and lackey cameras. Constraints can be freely added to groups of objects and cameras and their reference frames are edited by interacting with the cross hairs shown in Figure 6. Unlike the composite canvas used in Singh, viewport transforms are represented as an interactive filmbox in front of its respective lackey camera.

Shadowing and illumination is implemented as a shading network that illuminates the original scene with respect to each lackey and boss camera and blends the results onto the corresponding deformed scene geometry. Ghosting of the kind seen in Figure 3 is easily created by duplicating the deformed geometry as non-linear projection parameters are varied and then rendering the multiple instances of geometry in the scene with varying opacity.

4. Results and Conclusion

As described in the introduction this paper presents a comprehensive system for constructing and rendering non-linear projections, that is currently being used in an animation titled "Ryan". Figure 11 and Figure 12 show stills from animation sequences that employ non-linear projections to distort the scene. Figure 8 and Figure 10 show that the non-linear camera model can be used not only to generate scene distortions but interesting illumination effects as well. The formulation of the model in Section 2 is different from that in Singh[12] (only weight vector computation remains the same). The model is easy to understand for animators and is usually added to a conventionally animated scene after most of the animation has been specified. Animators almost exclusively work with the boss camera and frequently switch from non-linear projection to the underlying linear perspective.

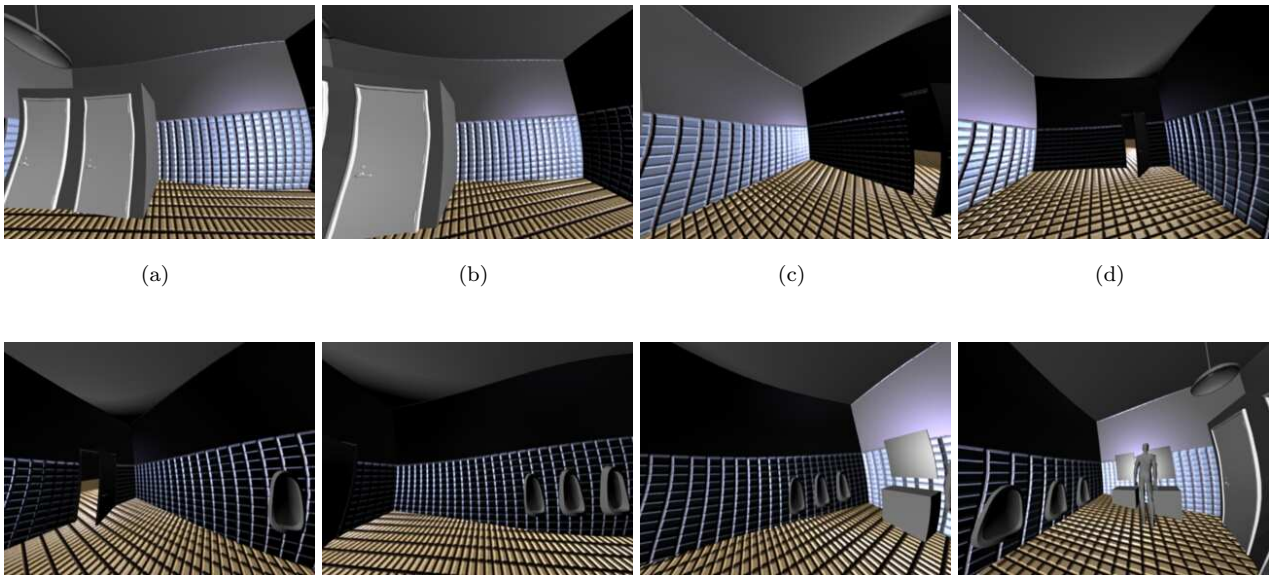
Summarizing this paper presents a new formulation for interactive non-linear projections that addresses scene constraints, shadowing and illumination and its integration into current production pipelines. The concept of using multiple linear perspectives to construct a non-linear projection and the concept of computing camera influence using various functions are existing ideas[12] that we build upon. Our results showcase the current use of our technique in the commercial animation production "Ryan".

References

1. M. Agrawala, D. Zorin and T. Munzner. Artistic Multiprojection Rendering. *Eurographics Rendering Workshop*, 125–136, 2000.
2. M. Alexa. Linear Combination of Transformations *SIGGRAPH*, 380–387, 2002.
3. A. Barr. Ray Tracing Deformed Surfaces. *Computer Graphics*, 20(4):287–296, 1986.
4. J. Dorsey, F. Sillion and D. Greenberg. Design and Simulation of Opera Lighting and Projection Effects. *Computer Graphics*, 25(4):41–50, 1991.

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

Figure 11: *Ryan Cafeteria Scene*



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

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

Figure 12: *Ryan Bathroom Scene*

5. J. Foley, A. van Dam, S. Feiner, J. Hughes. Computer Graphics, Principles and practice. Addison-Wesley, Chapter 6:229–284 ,1990.
6. C. Fu, T. Wong, P. Heng. Warping Panorama Correctly with Triangles. Eurographics Rendering Workshop,1999.
7. D. Martin, S. Garcia and J. C. Torres. Observer dependent deformations in illustration. *Non-Photorealistic Animation and Rendering 2000* , Annecy, France, June 5-7, 2000.
8. N. Max. Computer Graphics Distortion for IMAX and OMNIMAX Projection. *Nicograph '83 Proceedings* 137–159.
9. S. Peleg, B. Rousso, A. Rav-Acha and A. Zomet. Mosaicing on Adaptive Manifolds. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(10):1144–1154, 2000.
10. P. Rademacher and G. Bishop. Multiple-Center-of-Projection Images. *Computer Graphics*, 199–206, 1998.
11. P. Rademacher. View-Dependent Geometry. *Computer Graphics*, 439–446, 1999.
12. K. Singh. A Fresh Perspective. *Graphics Interface*, 17–24, 2002.
13. S. Seitz and C. Dyer. View Morphing: Synthesizing 3D Metamorphoses Using Image Transforms. *Computer Graphics*, 21–30, 1996.
14. D. Wood, A. Finkelstein, J. Hughes, C. Thayer and D. Salesin. Multiperspective Panoramas for

Cel Animation. *Computer Graphics*, 243–250, 1997.

15. G. Wyvill and C. McNaughton. Optical models. *Proceedings CGI*, 1990.

16. D. Zorin and A. Barr. Correction of Geometric Perceptual Distortion in Pictures. *Computer Graphics*, 257–264, 1995.