Eurographics Symposium on Rendering 2010
Jason Lawrence and Marc Stamminger
(Guest Editors)

Volume 29 (2010), Number 4

BendyLights: Artistic Control of Direct Illumination by Curving Light Rays

William B. Kerr Fabio Pellacini Jonathan D. Denning

Dartmouth College

Abstract

In computer cinematography, artists routinely use non-physical lighting models to achieve desired appearances. This paper presents BendyLights*, a non-physical lighting model where light travels nonlinearly along splines, allowing artists to control light direction and shadow position at different points in the scene independently. Since the light deformation is smoothly defined at all world-space positions, the resulting non-physical lighting effects remain spatially consistent, avoiding the frequent incongruences of many non-physical models.* BendyLights *are controlled simply by reshaping splines, using familiar interfaces, and require very few parameters.* BendyLight *control points can be keyframed to support animated lighting effects. We demonstrate* BendyLights *both in a real-time rendering system for editing and a production renderer for final rendering, where we show that* BendyLights *can also be used with global illumination.*

Categories and Subject Descriptors (according to ACM CCS): Computer Graphics [I.3.7]: Color, shading, shadowing, and texture—

1. Introduction

In computer cinematography, lighting is crucial in supporting storytelling [Cal99]. In order to convey certain moods and appearances, artists require a more flexible control of lighting than possible with physically-based models. In particular, non-physical tools like lights that do not cast shadows or have arbitrary spatial falloffs are often used when controlling the lighting on characters or to mimic indirect lighting effects. Cinematic lighting models have been introduced to provide artists with non-physical controls. For example, [Bar97] controls light shape and falloff with a complex combination of shading parameters. While these models are powerful, artists have found that lighting different objects with different sets of lights, or "light linking", is necessary to achieve many desired effects. These techniques are common in production environments [Cal99], and well-supported by most commercial packages such as Maya [Aut10].

We propose *BendyLights*, a spotlight-based lighting model for artistic control in which light travels along nonlinear paths from a point source. We determine the incident light direction at a point along a nonlinear path by inverting a cubic equation. We determine shadows along a nonlinear path by computing linear visibility on nonlinearly deformed geometry. Fig. 1 shows some basic examples of *BendyLights*.

Compared to traditional cinematic lights and light linking, *BendyLights* have the following advantages:

- lighting effects, including shadows and highlights, can be adjusted independently at different spatial locations using only a single light
- smooth spatial consistency of the non-physical lighting effects is maintained across surfaces
- light paths can be easily controlled with familiar spline editing tools; artists can also drag shadows, highlights and hotspots while our interface appropriately deforms the spline
- *BendyLight* lighting effects can be animated by keyframing the spline control points
- *BendyLights* can be rendered in real-time on GPUs and easily integrated in production renderers; they also work with global illumination

The rest of the paper is organized as follows. We begin by reviewing prior work on cinematic and nonlinear lighting models (Sec. 2), next introducing the *BendyLight* model (Sec. 3). We then show how to integrate *BendyLights* in rendering (Sec. 4) and editing (Sec. 5) systems. We analyze *BendyLights*' performance and compare them with known lighting models in Sec. 6 and finally summarize our contributions, limitations, and future directions (Sec. 7).

Published by Blackwell Publishing, 9600 Garsington Road, Oxford OX4 2DQ, UK and 350 Main Street, Malden, MA 02148, USA.

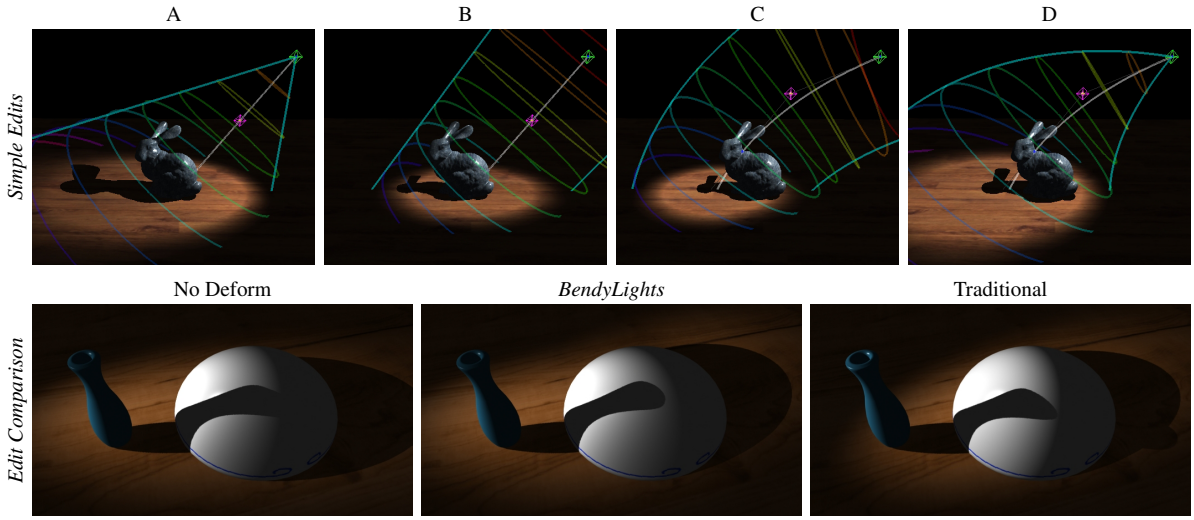

Figure 1: *Row 1: An example of basic* BendyLights *operations. (A) Non-deformed spotlight. (B) Setting a constant radius. (C) Bending light rays from B. (D) Bend from C with the radii of A. Row 2: Example of lighting using a* BendyLight *that is not possible using traditional linear lighting techniques. Shown is a spotlight before deformation, the resulting edit with* BendyLights*, and our best approximation of this edit with traditional linked linear lights.*

2. Related Work

Artistic Lighting Control. Artists often need and use non-physical lighting in their compositions [Cal99]. [Bar97] controls light shape and falloff at the light source with a complex combination of shading parameters. [TiABI07] control diffuse and specular features of lighting on surfaces in a non-photorealistic rendering context. [OKP*08] propose a method for controlling the transport of indirect illumination from surface to surface. [ROTS09] control the directions in which light is reflected off of objects' surfaces. Our method focuses on non-linear deformations for the light directions coming from a point source.

Nonlinear Light. The idea of nonlinear lighting has been explored in the past. [BTL90] use ray marching through discretely placed refraction layers to bend view rays, and [Mus90] replaces refraction with total reflection. [Grö95] introduces a more common nonlinear raytracing technique that renders using linear ray segments, but does so at each step of a numerical solution to differential equations. Similar techniques are used by [GSMA06] to simulate atmospheric phenomena, [Wei00] for gravitational fields, and [Sat03] for generalized nonlinear dynamical systems. [GnAS05] extend this technique for photon mapping. [SLS96] use a slightly different approach with ray marching by perturbing rays according to bounded force fields. Rendering of nonlinear rays can also be accomplished by voxelizing over a vector field, used by [IZT*07] to render refractive objects and [ZHF*07] to render heat shimmering and mirage. [Grö95] also discusses a method for using rays curved according to parametric equations, storing them in hierarchical bounding volumes (with some assumptions on monotonicity of the ray) and computing intersections with linear segments.

[WSE04] propose using GPU hardware to do nonlinear raytracing without shadows, but the speed is still relatively slow. [ZHF*07] updates this algorithm from multiple passes to a faster single pass, but still without shadows. In [IZT*07], caustics and shadows are rendered at interactive rates, but modifying lighting directions requires a more expensive and slow computation, as well as a voxelized representation of the scene.

To our knowledge, we are the first to propose a method for rendering nonlinear light for artistic control. We do so without voxelization, discrete linear steps, or precomputation. Because our light model can be represented linearly after deforming surface positions, we take advantage of GPUs to achieve real-time rendering.

3. Model

Goals. We seek to develop a light model that allows artists to control shadows and incoming light directions independently in different parts of the scene while smoothly interpolating at all scene locations to keep scene appearance consistent. Furthermore, lighting should remain consistent during object animation and the lighting model should be keyframeable to allow for time-varying lighting effects.

BendyLights. To achieve these goals, we propose a spotlight-based lighting model where, conceptually, light travels along nonlinear splines from a point source. We use quadratic Bezier splines [Mor85] to keep render time low,

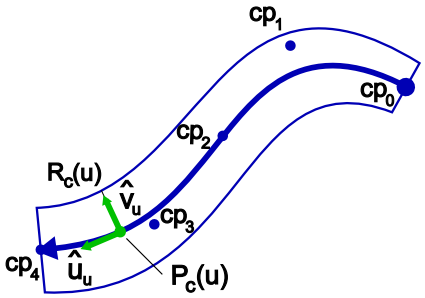

Figure 2: *Parameterization for 3 segments of the quadratic spline.*

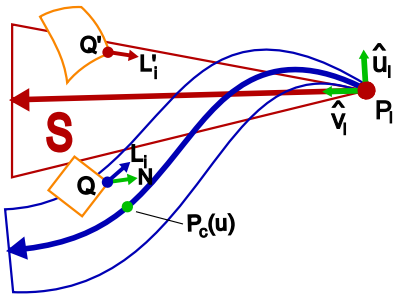

Figure 3: *Deformation of geometry. Red: spotlight S. Blue:* BendyLight. *Orange: scene geometry.*

and represent the light as a radially-symmetric tube surrounding the spline, much like the cone of a spotlight. See row 1 of Fig. 1 for a basic example. Manipulating the shape of *BendyLights* is achieved by moving spline control points in space. To render with *BendyLights*, we need to determine the incident light direction and nonlinear visibility function at all scene points. We define the lighting direction as the tangent of the light tube at that location. We compute visibility by non-linearly transforming geometry according to the inverse of the nonlinear light path. Linear computation of shadows on this deformed geometry is equivalent to nonlinear computation of shadows on the original geometry.

Parametric Representation. We define *BendyLight* starting from an existing spotlight S defined by a coordinate system. The source position of S is the first control point that of the quadratic Bezier spline C defining the center of the light tube (Fig. 2). C can have an arbitrary number of quadratic segments. Given a spline segment, a position on C in world coordinates can be written as follows:

$$P_c(u) = cp_0(1-u)^2 + cp_1 2(1-u)u + cp_2 u^2 \qquad (1)$$

where P_c is a point on C at u and cp_0, cp_1, cp_2 are control points for the spline segment (Fig. 2). Light directions within the tube are defined on the plane orthogonal to the tube spline at every P_c. We define the radius of the tube on these planes by another spline:

$$R_c(u) = cr_0(1-u)^2 + cr_1 2(1-u)u + cr_2 u^2 \qquad (2)$$

where R_c is the scalar radius of the tube at u and cr_0, cr_1, cr_2 are its control points (Fig. 2). We keep a consistent coordinate system along the tube with orthogonal basis vectors $(\hat{u}_u, \hat{v}_u, \hat{w}_u)$. The forward vector $\hat{u}_u$ is the normalized derivative of C at u. $\hat{v}_u$ and $\hat{w}_u$ are oriented by an "up" vector. Using this basis, a world position P_w can be written in the tube's parametric space as

$$P_w(u,v,w) = P_c(u) + R_c(u)(v\hat{v}_u + w\hat{w}_u) \qquad (3)$$

Deformation. Given a surface position Q with normal N in world space, we need to determine its location Q' after deformation into S's light space (Fig. 3). To do so, we want to find the coordinates (u,v,w) of Q with respect to the *BendyLight* tube (see Fig. 3). We find u where Q is orthogonal to the tube by solving the following equation:

$$\Big(Q - P_c(u)\Big) \cdot \hat{u}_u = 0 \qquad (4)$$

See Sec. 4 for our approach to solving equation (4). Once u is found, we can compute

$$v = \Big((Q - P_c(u)) \cdot \hat{v}_u\Big) / R_c(u) \qquad (5)$$

$$w = \Big((Q - P_c(u)) \cdot \hat{w}_u\Big) / R_c(u) \qquad (6)$$

Next we simply perform a change of basis from $(\hat{u}_u, \hat{v}_u, \hat{w}_u)$ of the tube to a set of basis vectors $(\hat{u}_l, \hat{v}_l, \hat{w}_l)$ for S's local light space. The coordinates (u,v,w) from the tube become (u',v',w') in this basis. We can now define

$$Q' = P_l + u'\hat{u}_l + v'\hat{v}_l + w'\hat{w}_l \qquad (7)$$

where P_l is the world-space position of light source S, $u' = u$, $v' = v$, and $w' = w$.

Let L'_i be the non-deformed light direction from S incident at Q', and L_i be the deformed light direction incident at Q. Since S is a spotlight, L'_i is simply the ray from Q' to S's source.

$$L'_i = (P_l - Q') \qquad (8)$$

$$L_i = -P'_c(u) + \frac{Q' - P_c(u)}{|Q' - P_c(u)|} * R'_c(u) * \frac{|Q' - P_c(u)|}{R_c(u)} \qquad (9)$$

where $P'_c(u)$ and $R'_c(u)$ are the first derivatives of C and the radius spline at u respectively. This takes into account both the tangent to the center curve C and the spread of light rays moving outward along the tube radius.

Shading and Shadows. Reflectance and shading can be computed using the deformed light direction, L_i, incident at world-space surface location Q and normal N. Because we have simply replaced the value of L_i, traditional rendering techniques like perturbing normals or displacing surface positions still work. In Sec. 6 we demonstrate that *BendyLights* also work with techniques like motion blur and global illumination. Shadows can be drawn by computing visibility from deformed surface positions Q' along the linear light directions L'_i. The resulting shadow lookups are equivalent to casting bent rays in the scene. Since Q' is non-linearly related to Q, this might require tesselating the geometry to avoid shadow artifacts, the same procedure required by any

Scene	Triangles	OpenGL	RenderMan Pass Shadow	Main	Total
Tabletop Bunny	102k	77.8 fps	8.3 s	4.0 s	12.3 s
Tabletop Vases	15k	137.0 fps	5.7 s	3.5 s	9.2 s
Tabletop Bowl	16k	104.8 fps	6.6 s	3.7 s	10.3 s
GI Bunny	118k	51.9 fps	19.0 s	5.2 s	24.2 s
GI Dancer	69k	49.0 fps	20.1 s	5.4 s	25.5 s
Bedroom	143k	34.3 fps	15.2 s	13.7 s	28.9 s
Outdoor	348k	25.0 fps	14.1 s	9.4 s	23.5 s
Rocks	88k	55.5 fps	12.3 s	2.2 s	14.5 s

Table 1: *For each scene in this paper: number of triangles, render time for our OpenGL renderer (in frames per second), and render time using RenderMan via Pixie (seconds per frame). Times for GI Bunny and GI Dancer for direct illumination from* BendyLight *only.*

non-linear mesh deformation. We find that practical applications require only modest tesselation. Fig. 5 shows an example of tesselated wireframes.

Animation. We can animate *BendyLights* by keyframing its control points for positions or radii along the tube. Additionally, following equations (1) and (2), we can control the global placement of the light tube through spotlight S's local transform. This allows us to animate a *BendyLight* using traditional keyframing toolsets. See Sec. 5 for more information.

Extensions. This deformation model lets us "cheat" any parameter of a standard lighting model in a spatially coherent way, since all deformations occur smoothly along world-space splines. Additional lighting parameters like intensity, radial falloff, or tube cross section shape can be expressed by mapping one parametric space to another (e.g., a nonlinear intensity spline's parametric space $\rightarrow$ the deformation tube's parametric space $\rightarrow$ world space).

4. Implementation

In this section we discuss details on how to render with *BendyLights* in real-time as well as in a production rendering system.

Solving the Parametric Equation. To solve equation (4), we must find the roots of a polynomial over u, the tube spline's parametric variable (Sec. 3). This method must be fast, as it will be applied to every surface position in the scene. It must also be numerically stable, as the spline can bend arbitrarily and imprecise roots would cause strong artifacts. Since we use quadratic splines for C, equation (4) requires inverting a cubic in u. While a closed-form solution exists, we found that its direct use is often numerically unstable as intermediate values vary by several orders of magnitude. We instead use Cardano's method [Tie65] that produces a relatively fast and stable solution to this equation. Assume we have a cubic as in equation (4) written as $x^3 + ax^2 + bx + c = 0$. By substitution we can show that $x = t - a/3$ where t can take the values

$$\begin{aligned} t_1 &= \sqrt[3]{-\tfrac{q}{2} + \lambda} + \sqrt[3]{-\tfrac{q}{2} - \lambda} \\ t_2 &= (-\tfrac{1}{2} + \tfrac{\sqrt{3}}{2}i)\sqrt[3]{-\tfrac{q}{2} + \lambda} + (-\tfrac{1}{2} - \tfrac{\sqrt{3}}{2}i)\sqrt[3]{-\tfrac{q}{2} - \lambda} \\ t_3 &= (-\tfrac{1}{2} - \tfrac{\sqrt{3}}{2}i)\sqrt[3]{-\tfrac{q}{2} + \lambda} + (-\tfrac{1}{2} + \tfrac{\sqrt{3}}{2}i)\sqrt[3]{-\tfrac{q}{2} - \lambda} \end{aligned} \tag{10}$$

with $\lambda = \sqrt{\frac{q^2}{4} + \frac{p^3}{27}}$, $p = b - \frac{a^2}{3}$ and $q = c + \frac{2a^3 - 9ab}{27}$.

To determine which of the roots correctly represent u, we assume that the light tube should not intersect itself, as a single light source should not hit a surface position at more than one direction under direct illumination. For any surface position Q, we iterate over each quadratic segment of tube spline C. If a u exists such that Q is within the tube at u, it is immediately accepted. Otherwise, we choose a default u for which Q will not interfere with shadow lookups. We implement another minor optimization by storing bounding boxes for tube segments so segments can be pruned for some surface positions.

Real-Time on the GPU. We implement *BendyLights* entirely in GLSL shaders. Our system renders in two passes. First, we generate a shadow map using the deformed geometry by solving for the light tube's parametric variables and deforming vertex positions Q to Q' in a vertex shader. Depth tests are then carried out with standard shadow maps. Since the vertex transformation is non-linear, geometry must be tesselated to a degree. Note though, that this is not as problematic for shadow maps since lower precision can often be used, as is shown by the common practice of using approximated shadow meshes. Fig. 5 shows an example of tesselated wireframes. Second, we make a shading pass where vertices are passed through a vertex shader unchanged, but we solve equation (4) to get L_i. The deformed surface position Q' is also calculated in a pixel shader to compute a hotspot with falloff. We do all shading per-pixel to ensure no shading artifacts appear.

Offline Production Renderer. Our GPU implementation shows that *BendyLights* can be implemented well using standard shadow mapping and shaders. This implies that our model can be integrated into production renderers for high-quality cinematic lighting. We use Pixie [Ari10], an open source version of RenderMan, to render all of the figures in this paper that do not show editing gizmos. Our RenderMan implementation uses a light shader to compute the shading from the pixel shader in our GPU implementation. Aside from syntax, the two shaders are virtually identical. To compute the shadow maps, we first make a shadow pass with pre-deformed geometry using the vertex shader code from our GPU implementation. We could also have used raytracing on the deformed geometry for shadow computation, but chose shadowmaps for speed. Performance for each rendering implementation is summarized in Table 1.

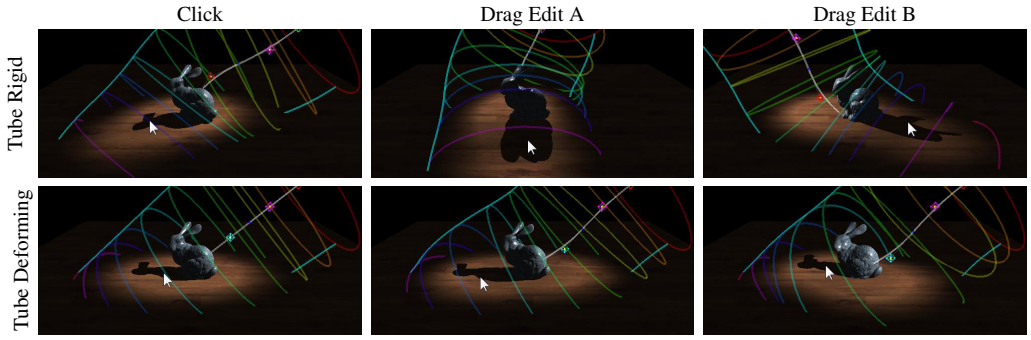

Figure 4: *An example of dragging shadows with a* BendyLight *using an indirect interface.*

5. Control Scheme

In this section we outline several ways in which *BendyLights* can be controlled by a user. Recently, it has been shown that for direct lighting with point lights, direct manipulation and (indirect) feature dragging are desirable user interface paradigms [KP09]. We demonstrate that *BendyLights* can be controlled using both of these paradigms in the supplemental video.

Direct Control. Using a direct control scheme, *BendyLights* can be controlled both rigidly and non-rigidly. Since control points of spline C are defined with respect to a standard spotlight S (Sec. 3), we can move, rotate, and scale the entire tube in exactly the same way we would edit S. To deform the tube non-rigidly, a user drags each of the control points that define spline C. Each control point has an associated radius and RGB value that can be directly edited. See Fig. 1, Row 1 for some basic edits and the supplemental video for a demonstration of all controls.

Indirect Control. We adapt the techniques in [PTG02] to drag shadows, hotspots, and highlights by either transforming the tube rigidly or deforming it non-rigidly, demonstrated in Fig. 4. When we refer to deforming surface positions, we mean the process by which Q is transformed to Q' in Sec. 3. We start by explaining how to accomplish indirect edits that transform the *BendyLight* tube rigidly.

For *shadow dragging*, let M be the surface position under the mouse pointer upon click after deformation. Let P_l be the position of light source S from Sec. 3. Let P be a pivot point, which is the first intersection of the ray from P_l to M with the deformed geometry used for shadowmaps. Let M' be the current surface position under the mouse pointer as the user drags the mouse, after deformation. We move P_l to new position P_l'.

$$P_l' = P + (P - M') \times |P_l - P| / |P - M| \qquad (11)$$

For *hotspot dragging*, M and M' define a rotation for S's coordinate frame, which moves the hotspot with the mouse pointer. We define two vectors

$$v = M - P_l/|M - P_l|, \quad v' = M' - P_l/|M' - P_l| \qquad (12)$$

Using quaternions, we can rotate S's coordinate frame around the axis $v \times v'$ by $cos^{-1}(v \cdot v')$ radians. This moves the hotspot with the mouse pointer no matter which point of the hotspot is clicked.

For *highlight dragging*, let X and N be the surface position and normal currently under the mouse pointer, and V_i be the incoming view direction, none deformed. Let L_i be the incoming light direction as in Sec. 3. We compute two vectors for a rotation as in the case with hotspots.

$$v = L_i, \quad v' = -V_i + 2(N \cdot V_i)N \qquad (13)$$

Here, v' represents the view direction reflected over N. Using quaternions, we can rotate P_l around X along the axis $v \times v'$ by $cos^{-1}(v \cdot v')$ in radians. This aligns L_i with the view reflection angle, assuring a specular highlight at X.

To drag shadows, hotspots, and highlights in a way that deforms the *BendyLight* tube, we have the user select which control points of the tube are of interest. We apply the same rigid edits as before, but instead of changing S, which affects all of C's control points, we modify only the selected control points.

6. Results

In this section, we present the application of *BendyLights*. First we discuss the performance of *BendyLights* in our rendering implementations. Second, we show many examples of how *BendyLights* can be used to create useful artistic effects. Third, we illustrate key differences between *BendyLights* and traditional linear point lights.

6.1. Rendering

Scenes. We test *BendyLights* on a variety of scenes to show robustness. The first set illustrates simple edits and features for clarity. This set includes a bunny on a plane (*Tabletop Bunny*: Fig. 1), two vases on a plane (*Tabletop Vases*: Fig. 5), a vase with a bowl (*Tabletop Bowl*: Fig. 1), a bunny next to colored walls (*GI Bunny*: Fig. 11), and an animated dancer next to colored walls (*GI Dancer*: Fig. 11). The second set

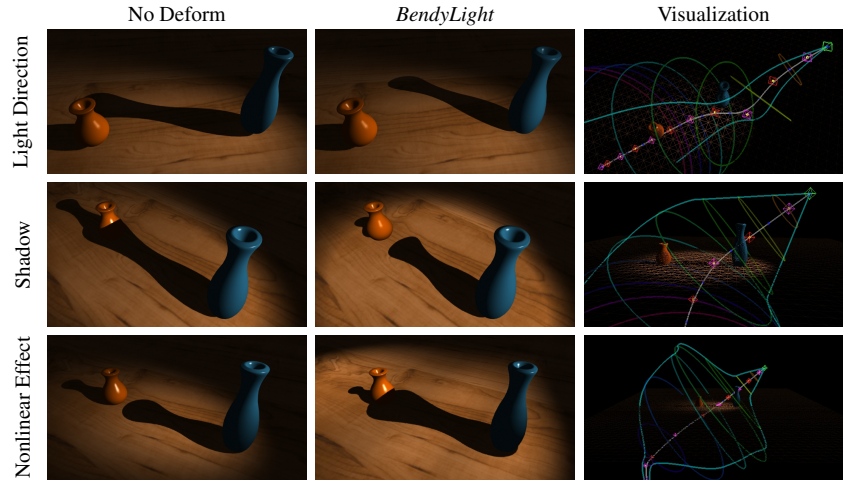

Figure 5: *Row 1: The lighting on the surface of the blue vase is changed while maintaining the lighting on the surface of the orange vase. Row 2: The blue vase's shadow is bent without changing the lighting on its surface. Row 3: The radius of the* BendyLight *tube is modified to fatten the blue vase's shadow in a non-realistic way, and effect not possible with a tranditional linear light.*

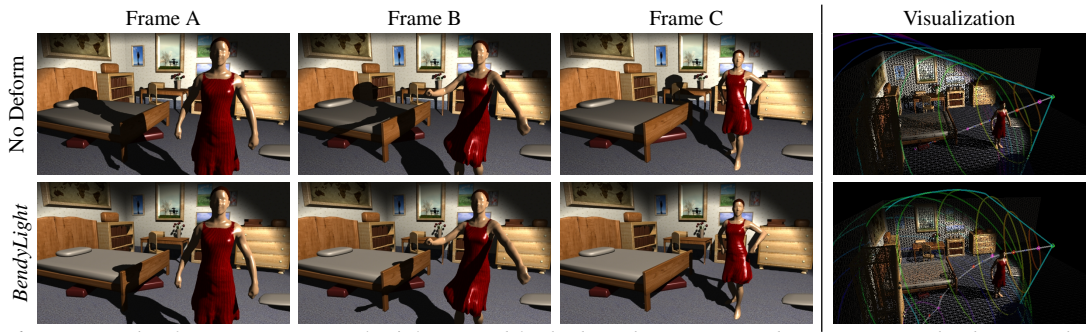

Figure 6: *An example of using a static* BendyLight *to modify shadow placement in a dynamic scene. The dancer's shadow is bent while maintaining the original lighting on the dancer herself. Shadows are spatially consistent across all objects.*

is used to show *BendyLights* applied to scenes that might occur in production animation. This set includes an animated dancer in a bedroom (*Bedroom*: Fig. 7), an outdoor scene with an animated horse (*Outdoor*: Fig. 8), and a rocky hill with vases to show bumpy terrain (*Rocks*: Fig. 10).

Performance. In Table 1, we list each scene and a timing for each of our rendering systems. These timings are for a single *BendyLight* with three segments (same as Figure 7). We render images at 686×382 and shadowmaps at 2048×2048 resolution. The machine is an Intel Core2 Quad 2.83GHz with 4GB of RAM and an NVIDIA GeForce 9800GT GPU. Our editing system works in real-time.

6.2. Example Edits

Light Shaping. Because a *BendyLight* follows a spline, artists can shape the direction light travels in a way that is much like modeling a tube. Fig. 1, Row 1 shows what this looks like.

Shadows. When designing lighting, an artist might want to change the position or shape of shadows. In Fig. 5 we show how a single *BendyLight* can be used to modify the shadow of the blue vase so that it no longer obscures the orange vase, without changing the way the blue vase itself is lit. Similarly, we can dynamically edit the shadows in a scene with animated geometry such as Fig. 6. Notice that the first object along a nonlinear light path occludes that light, casting a shadow on any other object the curved path intersects. The deformation of light directions is defined at all surface locations, meaning all objects added to the scene, including moving objects, will be lit similarly to other objects in close spatial proximity. Sometimes this effect can be accomplished with many linked lights, but other times it cannot (comparison in Section 6.3).

Lighting Direction and Intensity. Artists might also want to modify the direction at which light is hitting some objects. In Fig. 5, a *BendyLight* is used to light the blue vase more from the front without modifying the lighting hitting

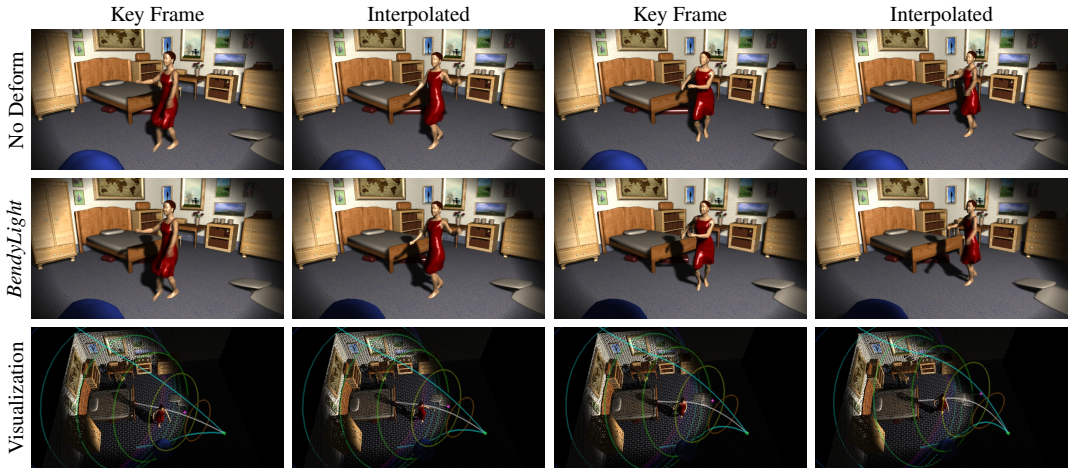

Figure 7: *An example of an animated deformation of light directions in an animated scene with motion blur. We deform the light so that the shadow of the dancer always falls on the bed.* BendyLights *allow spline-based manipulation of light as it travels in world space, modifying incident lighting angles and shadows in a spatially consistent way.*

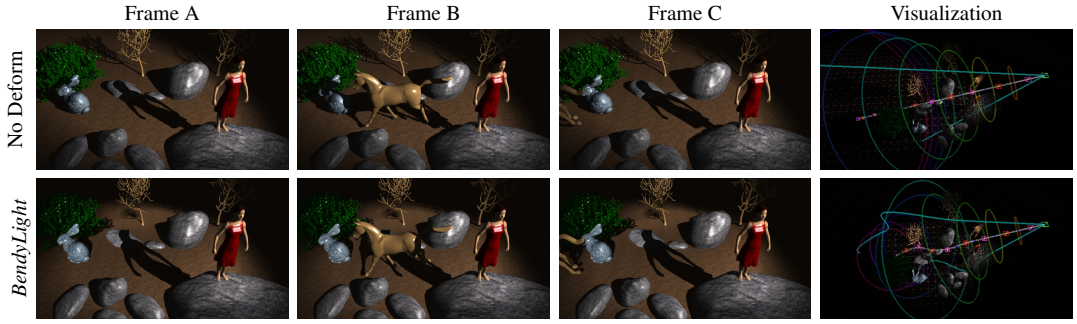

Figure 8: *An example of using a static* BendyLight *to modify the incoming light direction in a dynamic scene. The bunny is lit from a different direction while maintaining the light hitting the woman. As the horse moves across, its shadow remains consistent with those in close proximity.*

the orange vase. Similarly, we change the incident lighting on the bunny in Fig. 8 without changing the incident lighting on the woman standing on the rock. An animated horse runs through this scene to show how the light deformation is defined at every position in space. To the discerning eye, it may not appear that all light is coming from one source in this example, but the deformation gives the artist the power to modify these lighting directions in a way that guarantees that objects with close proximity to one another will have similar lighting. Light intensity can be spatially controlled in a similar way.

Animated Light Deformation. *BendyLight* deformations can be animated smoothly by keyframing and interpolating control point parameters. In Fig. 7 we show an animated *BendyLight* illuminating an animated scene with motion blur. The lighting bends so that as the dancer moves, her shadow always falls on the bed.

Global Illumination. Once the direct illumination is bent with *BendyLights*, other rendering techniques are executed linearly as usual. In Fig. 11 we show two examples of *BendyLight* with single-bounce global illumination computed by a gather loop. For example, we bend the light to get a more silhouette-like shadow shape of the animated dancer on the wall.

6.3. Comparison With Traditional Lighting

We present a comparison of *BendyLights* to standard linear lights and light linking, the process by which a light source can be "linked" to some objects and be active only on them.

Linear Lights Comparison. By exploiting the nonlinearity of *BendyLights*, artists can produce effects that would be impossible with a traditional linear light. Our first example is nonlinear changes in the radius of a *BendyLight*. In Row 3 of Fig. 5, the shadows of the vases are bulged and compressed by bending light rays. A more practical example is shown in Fig. 1, Row 2. With *BendyLights*, the shadow is bent so that it hits the outside of the bowl in a pleasing way. We show that the closest possible linear light in the far right image

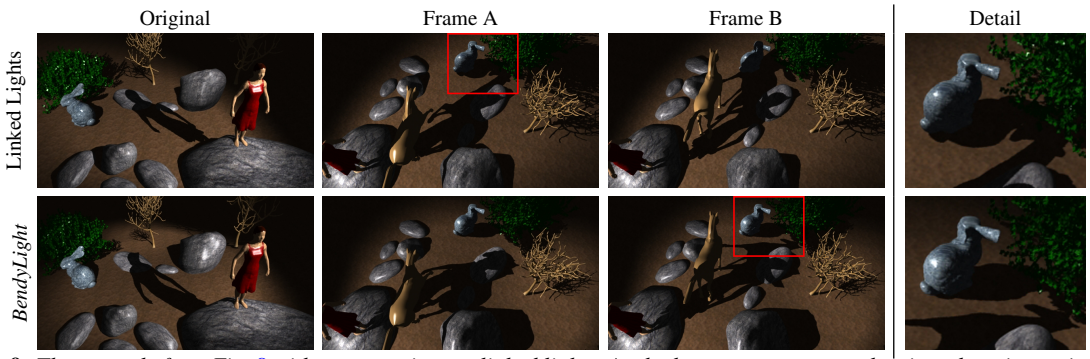

Figure 9: *The example from Fig. 8 with a comparison to linked lights. As the horse moves across the view, there is a point using linked lights where its shadow agrees with the dancer's shadow, but crosses the bunny's shadow. With* BendyLights*, the three shadows appear to agree with each other, even though they are nonphysical. Additional nonlinearities, such as the converging shadow rays from the tree on the right, causing a smaller shadow, are also not possible with traditional linked lights.*

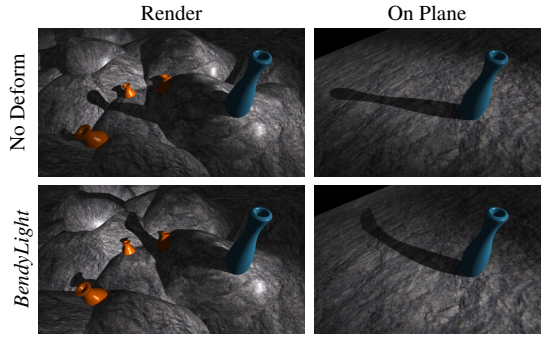

Figure 10: *Using a* BendyLight*, the shadow of the blue vase is bent to look appealing against the rocks and orange vases. Casting this shadow on a plane reveals that the shadow has a nonlinear shape that would not be possible using a traditional linear light. This nonlinearity is masked by the natural terrain.*

still distorts on the bowl. Finally, in Fig. 10 we show another example where lighting and shadows are bent, this time over a surface like natural terrain. We replace the rocky terrain with a planar surface to make the nonlinearity of the shadow more visible. Non-planar surfaces like the bowl and terrain make it easier achieve believeability with such edits. In order to achieve these nonlinearities with linear lights in general, every pixel sample in the image would have to be lit by a different light.

Linked Lights Comparison. When *BendyLight* edits are more subtle, the resulting light features may appear locally linear (e.g., Rows 1 and 2 of Figure 5). A skilled artist might achieve the same with complex linear light linking setups. *BendyLights* offer two primary advantages over light linking. First, *BendyLights* require fewer light sources to achieve similar effects. Second, the lighting directions under *BendyLights* map uniquely to every position in world space, interpolating smoothly. Positions close to each other in space will be lit similarly. A comparison can be seen in Fig. 9. We take the *Outdoor* scene from Fig. 8 and attempt to reproduce the same lighting effects with three linked lights using traditional linear lights. The advantage of using *BendyLights* can be seen when the horse moves to the middle of the shot between the woman and the bunny. At this point, the shadows being cast must simultaneously be consistent with the dancer and the bunny. Using linked lights, if the horse's shadow is consistent with the woman's, it crosses the bunny's shadow in an awkward way. With *BendyLights*, all shadows agree with each other at all times. We render the scene from an alternate camera angle to accentuate this difference (Fig. 9).

7. Conclusions

We have presented *BendyLights*, the first lighting model for artistic control in which, conceptually, light travels along nonlinear spline paths. This allows artists to deform shadows and set light directions independently at different scene locations, while still using only one light, which is consistently defined at all surface locations and times (during animation). *BendyLights* can be controlled using familiar editing interfaces, and work in scenes containing a variety of geometric complexities and with animation. They can be easily deployed both in GPU-based real-time editing systems as well as offline production renderers such as RenderMan, integrating with effects like motion blur and global illumination.

Limitations. *BendyLights* come with a few limitations. First, if the *BendyLight* tube self-intersects, shading artifacts can occur. This can be avoided by simply constraining the tube from self-intersecting during editing. Second, in order for shadows to appear smooth when bent, geometry must be tesselated to an appropriate degree, because *BendyLights* deform geometry for visibility. We find that the required tesselation is reasonable, especially for production scenes where many surfaces are already non-linearly deformed for animation purposes.

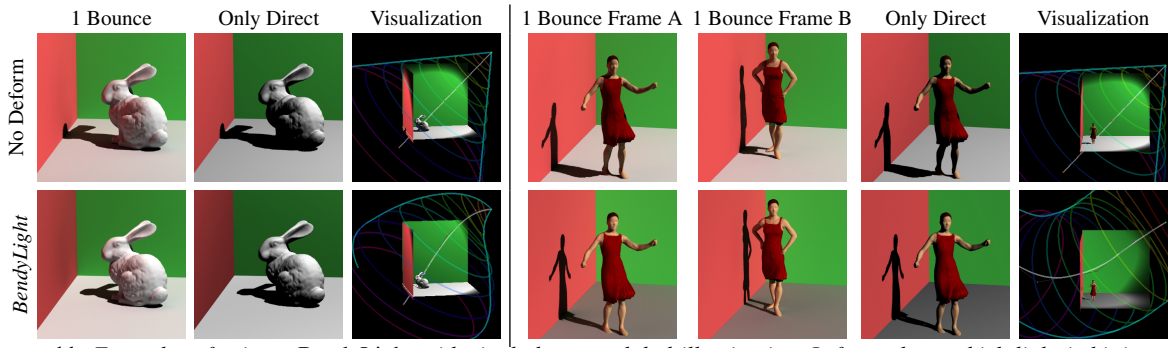

Figure 11: *Examples of using a* BendyLight *with single-bounce global illumination. Left: angle at which light is hitting red wall and bunny changed to decrease redness of bunny. Right: animation of dancing woman, shadows bent to better silhouette her shape.*

Future Work. In the future, we are interested in extending the ability to nonlinearly bend light to other light types, in particular area lights and environmental illumination. We would also like to explore global illumination methods where all bounces are bent.

Acknowledgments

We thank Lori Lorigo for her help in paper preparation. This work was supported by NSF (CNS-070820, CCF-0746117), Intel, and the Sloan Foundation. The horse and dancing woman models are courtesy of [SP04, VBMP08].

References

[Ari10] ARIKAN O.: Pixie open source renderman. http://www.renderpixie.com/. 4

[Aut10] AUTODESK INC: Maya 2010, 2010. 1

[Bar97] BARZEL R.: Lighting controls for computer cinematography. *Journal of Graphics Tools 2*, 1 (1997), 1–20. 1, 2

[BTL90] BERGER M., TROUT T., LEVIT N.: Ray tracing mirages. *IEEE Comput. Graph. Appl. 10*, 3 (1990), 36–41. 2

[Cal99] CALAHAN S.: Storytelling through lighting, a computer graphics perspective. In *Advanced RenderMan: Creating CGI for Motion Pictures*, Apodaca A. A., Gritz L., Barzel R., (Eds.). Morgan Kaufmann, 1999, pp. 223–233. 1, 2

[GnAS05] GUTIERREZ D., NOZ A. M., ANSON O., SERÓN F. J.: Non-linear volume photon mapping. In *Eurographics Symposium on Rendering 2005* (2005), pp. 291–300. 2

[Grö95] GRÖLLER E.: Nonlinear raytracing - visualizing strange worlds. *The Visual Computer 11*, 5 (1995), 263–274. 2

[GSMA06] GUTIERREZ D., SERON F. J., MUNOZ A., ANSON O.: Simulation of atmospheric phenomena. *Computers & Graphics 30*, 6 (2006), 994 – 1010. 2

[IZT*07] IHRKE I., ZIEGLER G., TEVS A., THEOBALT C., MAGNOR M., SEIDEL H.-P.: Eikonal rendering: efficient light transport in refractive objects. In *SIGGRAPH '07: ACM SIGGRAPH 2007 papers* (2007), ACM, p. 59. 2

[KP09] KERR W. B., PELLACINI F.: Toward evaluating lighting design interface paradigms for novice users. In *SIGGRAPH '09: ACM SIGGRAPH 2009 papers* (New York, NY, USA, 2009), ACM, pp. 1–9. 5

[Mor85] MORTENSON M. E.: *Geometric Modeling*. John Wiley and Sons, Inc., New York, NY, USA, 1985. 2

[Mus90] MUSGRAVE F. K.: A note on ray tracing mirages. *IEEE Comput. Graph. Appl. 10*, 6 (1990), 10–12. 2

[OKP*08] OBERT J., KŘIVÁNEK J., PELLACINI F., SÝKORA D., PATTANAIK S.: icheat: A representation for artistic control of indirect cinematic lighting. In *Eurographics Symposium on Rendering* (2008), vol. 27. 2

[PTG02] PELLACINI F., TOLE P., GREENBERG D. P.: A user interface for interactive cinematic shadow design. *ACM Transactions on Graphics 21*, 3 (2002), 563–566. 5

[ROTS09] RITSCHEL T., OKABE M., THORMÄHLEN T., SEIDEL H.-P.: Interactive reflection editing. *ACM Trans. Graph. (Proc. SIGGRAPH Asia 2009) 28*, 5 (2009). 2

[Sat03] SATOH T. R.: Symplectic ray tracing: A new frontier in non-linear ray tracing. In *The 11th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision 2003* (2003). 2

[SLS96] STAM J., LANGUENOU E., SYNTIM P.: Ray tracing in non-constant media. In *Rendering Techniques '96: Proc. 7th Eurographics Workshop on Rendering* (1996), pp. 225–234. 2

[SP04] SUMNER R. W., POPOVIĆ J.: Deformation transfer for triangle meshes. *ACM Transactions on Graphics 23*, 3 (Aug. 2004), 399–405. 9

[TiABI07] TODO H., ICHI ANJYO K., BAXTER W., IGARASHI T.: Locally controllable stylized shading. *ACM Transactions on Graphics 26*, 3 (July 2007), 17:1–17:7. 2

[Tie65] TIETZE H.: *Famous Problems of Mathematics: Solved and Unsolved Mathematical Problems From Antiquity to Modern Times*. Graylock Press, 1965. 4

[VBMP08] VLASIC D., BARAN I., MATUSIK W., POPOVIĆ J.: Articulated mesh animation from multi-view silhouettes. *ACM Transactions on Graphics 27*, 3 (Aug. 2008), 97:1–97:9. 9

[Wei00] WEISKOPF D.: Four-dimensional non-linear ray tracing as a visualization tool for gravitational physics. In *IEEE Visualization 2000 Proceedings* (2000), pp. 445–448. 2

[WSE04] WEISKOPF D., SCHAFHITZEL T., ERTL T.: Gpu-based nonlinear ray tracing. In *Computer Graphics Forum (Eurographics 2004)* (2004), vol. 23, pp. 625–633. 2

[ZHF*07] ZHAO Y., HAN Y., FAN Z., QIU F., KUO Y.-C., KAUFMAN A., MUELLER K.: Visual simulation of heat shimmering and mirage. *IEEE Transactions on Visualization and Computer Graphics 13*, 1 (2007), 179–189. 2