Photorealistic Rendering of Knitwear Using The Lumislice

Ying-Qing Xu Yanyun Chen[‡] Stephen Lin Hua Zhong
Enhua Wu[‡] Baining Guo Heung-Yeung Shum

Microsoft Research, China [*] [‡] Institute of Software, Academia Sinica [†]

Abstract

We present a method for efficient synthesis of photorealistic free-form knitwear. Our approach is motivated by the observation that a single cross-section of yarn can serve as the basic primitive for modeling entire articles of knitwear. This primitive, called the *lumislice*, describes radiance from a yarn cross-section based on fine-level interactions – such as occlusion, shadowing, and multiple scattering – among yarn fibers. By representing yarn as a sequence of identical but rotated cross-sections, the lumislice can effectively propagate local microstructure over arbitrary stitch patterns and knitwear shapes. This framework accommodates varying levels of detail and capitalizes on hardware-assisted transparency blending. To further enhance realism, a technique for generating soft shadows from yarn is also introduced.

CR Categories: I.3.3 [Computer Graphics]: Picture/Image Generation. I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism;

Keywords: Knitwear, Image-based Rendering, Transparency Blending, Parametric Surfaces, Photorealistic Rendering.

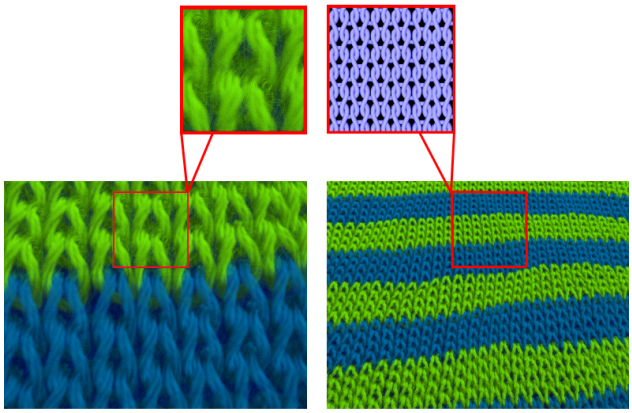

Figure 1: Knitwear synthesized using the lumislice and including soft shadows. Left side: close-up views of knitwear microstructure. Right side: stitch pattern and irregular macroscopic structure.

1 Introduction

Because of the ubiquity of textiles and clothing in everyday life, realistic rendering of such articles has been an active area of computer graphics research for over a decade [29, 25, 3, 4, 5, 7, 9, 2, 33, 15]. In particular, modeling and rendering of knitwear poses a considerable challenge. Unlike woven fabrics which can be well-represented by specialized BRDF models [32, 28, 8], knitwear is characterized by a macroscopic stitch structure that requires an explicit model of the knitting pattern and geometry. Currently there exist no convenient means for comprehensive photorealistic synthesis of knitwear.

One obstacle to realistic knitwear generation is the complicated microstructure of yarn. Close examination of yarn reveals countless thin fibers which give knitwear a fuzzy appearance. Improper rendering of this delicate structure can result in aliasing that causes a scene to look artificial, and given this fine-level complexity, rendering down or fluff with existing techniques such as [1, 30, 26, 16, 10, 21] is cumbersome, if not impossible. Moreover, the appearance of down changes in detail with different viewing distances. Another difficulty presented by knitwear is its variations in shape. For knitwear lying flat, its microstructure has been efficiently displayed using volume-rendering techniques [12]; however, the only known techniques for rendering free-form knitwear with microstructure utilize Gouraud-shaded triangles [33], which do not handle close-up views, and curved ray tracing [13], which is computationally expensive and does not effectively deal with multiple scattering that is especially evident on light-colored yarn.

Photorealistic rendering of knitwear must efficiently handle free-form surfaces, varying levels of detail from different viewing distances, and complex stitch geometry. In this paper, we address these challenges with the introduction of a primitive element called the *lumislice*. The lumislice represents radiance from a yarn cross-section, and arises from the key observation that based on the repetitive structure of yarn, articles of knitwear may be composed entirely of lumislices. This suggests a two-level structural hierarchy that can be formed for yarn to facilitate efficient photorealistic rendering.

- Locally, fine-level interaction among yarn fibers (occlusion, shadowing, multiple scattering) is modelled for a yarn cross-section and is called the lumislice. We will assume that the fluff on all cross-sections follows the same density distribution.
- Globally, because of the repetitive nature of knitted yarn and its known path along a skeleton, knitwear can be represented as a collection of identical lumislices in various positions and orientations.

It is this detailed but flexible quality that gives the lumislice much utility. Local microstructure can be closely represented in the lumislice, yet it can be easily disseminated to all points of a knitwear. The lumislice furthermore does not introduce restrictions on the global structure of the knitted yarn.

[*] Beijing Sigma Center, No. 49, Zhichun Road, Haidian District, Beijing 100080 P.R.China. Email: yqxu@microsoft.com

[†] This work was done when Yanyun Chen was visiting Microsoft Research, China.

ACM SIGGRAPH 2001, 12-17 August 2001, Los Angeles, CA, USA

The reasoning behind this lumislice-based method for knitwear is to elicit the following advantages:

- Fine local interaction can be precomputed and used throughout a piece of knitwear.
- Different levels of detail for varying viewing distances are elegantly handled by multiresolution lumislices, in a manner similar to mip-maps.
- A knitwear skeleton can be created on a free-form surface, which facilitates physically-based animation and other applications.
- The lumislice can be used for rendering arbitrary stitch patterns with better visual effects and lower cost than raytracing. Existing volumetric representation methods have significant difficulties in dealing with complex stitches on a deformable surface [12, 13].
- It can be easily implemented by standard graphics APIs, such as OpenGL.

Additionally, enhanced realism can be achieved with a proposed shadow map for yarn that is integrated into our method for soft shadow generation. Inclusion of such shadows can allow individual yarn fibers to be distinguished, as deep shadow maps have done for hair [19].

This method efficiently produces knitwear rendering results not previously achievable. Examples of lumislice rendering are displayed in Figure 1 for different viewing situations. The close-up views on the left side exhibit the delicate fluff characteristics of knitted yarn, and from a normal viewing distance shown on the right side, the rendering of variable macroscopic structure is demonstrated. The efficient and effective generation of this range of details distinguishes lumislice-based rendering. The lumislice may furthermore be extended to other materials consisting of sub-pixel elements such as hair, carpet or distant foliage. Applications of photorealistic knitwear rendering include web-based display of clothing products and fabric animation on moving characters. Automated generation of knitwear would also be desirable for motion pictures such as Stuart Little, where much fur and clothing structure required significant labor to be synthesized [23].

The rest of this paper is organized as follows. Section 2 provides an overview of related work, and is followed in Section 3, with a general description of our approach. The lumislice, pattern stitiching, and knitwear rendering are presented as components of our technique in Section 4. Subsequently, results are exhibited in Section 5, and conclusions are presented in Section 6.

2 Related Work

An initial approach to rendering of complex repetitive patterns is using volumetric textures [16, 21]. The original volumetric texture model, called the texel, was introduced by Kajiya and Kay in 1989 [16]. The teddy bear rendered by this model remains one of the most successfully synthesized furry objects to date, and its associated ray-tracing algorithm continues to be one of the most general realistic rendering techniques for such scenes.

Other approaches for rendering furry objects include the so-called fake fur rendering that was introduced by Goldman [10] and the real-time fur rendering method introduced by Lengyel [17]. Goldman used a probabilistic lighting model for rendering a thin coat of fur on skin seen from normal distances. Lengyel used an alpha-blending technique to approximately render fur represented by volumetric texture.

Knitwear has unique features different from fur and other fabrics that should be considered when rendered. As previously mentioned, the microstructure of knitted yarn typically consists of a large number of thin fibers, and the size of yarn and stitches exceeds that of thread, thus precluding representation by BRDF models.

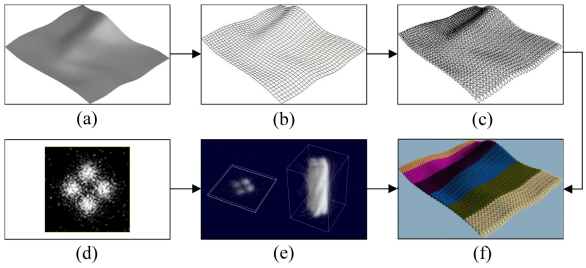

Figure 2: An overview of lumislice-based rendering. Pattern stitching: (a) free-form surface, (b) placement of control points, (c) knitwear skeleton with stitches. Lumislice modeling: (d) density distribution of fluff, (e) lumislice computation and winding over a yarn segment. With (c) and (e), a piece of knitwear can be rendered as (f).

Graphics researchers have successfully applied volume-rendering techniques to efficiently display yarn microstructure for knitwear on a flat surface [12], but for free-form knitwear, few techniques for rendering yarn microstructure exist [13, 33].

The curved ray-tracing method introduced in [13] has three major drawbacks. One is the high cost of rendering. Second, users must build different volumetric textures for different knitting patterns. This makes it difficult to model knitwear with advanced stitching patterns. Third, evident artifacts arise when the knitwear is severely deformed. Zhong *et al.* developed an effective model for rendering yarn microstructure using Gouraud-shaded triangles [33]. Their method works well for situations of normal viewing distances, but it does not address close-up views.

3 General Approach

Because of the detailed structure of knitwear, past methods are burdened with high computational costs and/or large storage requirements for synthesis of photorealistic free-form knitwear over multiple scales. We approach this problem by developing a reflectance primitive for yarn that concisely represents microstructure in a manner that allows for various levels of detail and hardware-assisted rendering. In this section, we highlight the basic idea behind our technique.

To avoid the high cost of raytracing, volumetric objects can be rendered by hardware-aided texture mapping with transparency blending [7, 20, 17]. However, limitations arise from the use of graphics hardware. First of all, it cannot compute the shading or shadowing for each individual texture pixel [20]. Secondly, if 2D alpha texture is employed, the volumetric texture should be split into slices parallel to the viewport and sorted from far to near for accurate blending computation [7].

Splitting and sorting are solvable, but if shading and shadowing cannot be accurately computed according to the lighting and viewing conditions of the volumetric texture, the rendering result would likely appear artificial. A way to achieve realistic shading effects is to compute the results of all possible lighting and viewing conditions offline, and save them in a volumetric texture data structure. However, a moderately complex scene would seemingly require a prohibitive amount of storage and computation resources.

For this, a key observation is made. An examination of knitwear reveals that although patterns of knitted yarn can be complex, they are entirely composed of a single basic element. For a given type of yarn, the distribution of fibers for each cross-section is similar. To make our knitwear rendering goals realizable, we model a yarn strand by winding a fixed 2D fluff distribution along the path of

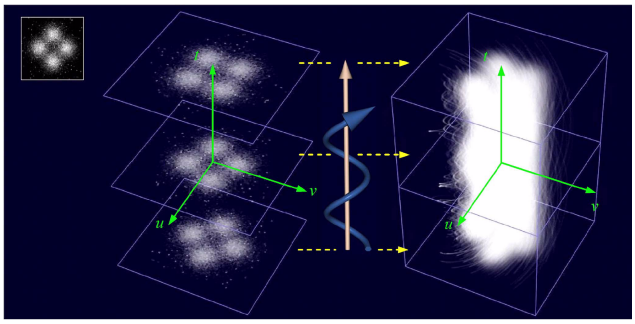

Figure 3: Generation of a volumetric yarn segment. The fluff density distribution in the upper-left corner is rotated along the path of the yarn to yield a yarn segment.

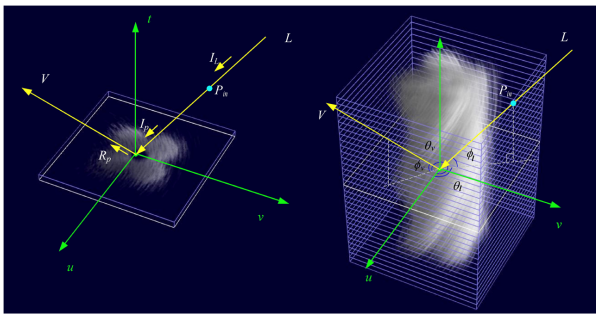

Figure 4: Notation for lumislice calculation. Left: Light of intensity I_L enters the yarn segment at voxel P_{in} and is attenuated to intensity I_p by the yarn upon reaching voxel p; the reflected light intensity upon exiting p towards the viewing direction is R_p. Right: Spherical angles at voxel p of the light direction L and the viewing direction V.

the yarn. Hence, we can completely synthesize knitwear from a single volumetric yarn cross-section. To exhibit microscopic realism, this cross-section incorporates fine visual effects, such as self-occlusion, self-shadowing, and multiple scattering. For different levels of detail, the resolution of this slice can be varied with smooth visual transitions. It is compactly represented in 2D as a single thin volumetric layer and can be computed offline. In addition, this structure is amenable to transparency blending for efficient rendering.

4 Lumislice-based Rendering

Like other realistic synthesis methods, generation of lumislice-based knitwear images consists of modeling and rendering. Specifically for our technique, two components are modelled: the lumislice and the stitch skeleton. Figure 2 illustrates the framework of our method.

From the cross-sectional fluff density distribution, we compute the lumislice to be used as the basic structural primitive for the entire piece of knitwear. We also model the entire shape of the knitwear as a freeform surface defined by control points. These points are additionally used to outline the stitch patterns, thereby delineating the knitwear skeleton.

Finally, we divide the knitwear skeleton into yarn segments that are further divided into volumetric slices. The reflectance characteristics of these yarn cross-sections are each represented by the lumislice, stored in a lookup table of 2D transparency texture. The volumetric slices are then rendered according to lumislice lighting and viewing conditions using hardware-aided transparency-blending APIs.

In the remainder of this section, we present the two primary components of our algorithm, the lumislice and pattern stitching, then describe the rendering routine.

4.1 The Lumislice

The lumislice is the fundamental structure we introduce for knitwear rendering. It is this element which facilitates the rendering of knitwear at different viewing distances while maintaining appropriate microscopic and macroscopic features.

For a given illumination direction, the lumislice is essentially a computed Lumigraph / light field for a yarn cross-section [11, 18]. The lumislice describes a cross-sectional slice of yarn that is divided into voxels. Each of these voxels has an associated four-dimensional array that contains its opacity and its reflectance function. This reflectance function represents the brightness of a voxel viewed from direction $V(\theta_v, \phi_v)$ when illuminated by a unit intensity light from direction $L(\theta_l, \phi_l)$. It differs from the traditional notion of BRDF in that it accounts for the attenuation of incident light passing through the surrounding yarn. Using this *voxel reflectance function* (VRF) allows us to precompute the effects of the surrounding yarn.

In this subsection, we present the method for computing the lumislice for a given type of yarn, and then describe the calculation of viewed radiance from a lumislice voxel.

4.1.1 Lumislice Computation

As mentioned, each lumislice voxel has two associated quantities, the opacity of its yarn fluff and the VRF. The opacity value is derived from the fluff density distribution, illustrated in Figure 3, which is assumed to be uniform for all cross-sections of the given yarn. For yarns with different fluff characteristics, its corresponding density distributions are used instead.

Our formulation of the VRF is presented in terms of R_p, the outgoing radiance from voxel p. Three primary physical elements factor into the reflectance at a voxel p: the fluff density ρ_p, the shading model Ψ of yarn, and the incident light distribution I_{msp} that results from multiple scattering among neighboring voxels N. The emission intensity from a voxel in principle can also affect a VRF, but this quantity for yarn is zero. These factors influence R_p according to the equation

$$R_p = \rho_p \Psi (I_p + I_L \sum_N I_{msp}) \tag{1}$$

where I_L is the light intensity and I_p is its attenuated intensity upon reaching p, as shown in Figure 4. The shading model Ψ of [16] for diffuse reflection is used, and multiple scattering will be later addressed. Based upon the emission-absorption model of volume illumination [6], I_p may be expressed as

$$I_p = I_L e^{-\gamma \sum_{r=p}^{P_{in}} \rho_r} \tag{2}$$

where P_{in} is the point of light entry into the yarn bounding box, γ is a light transmission factor in the emission-absorption model, and voxel dimensions are unit length. Then (2) can be substituted into (1) to produce

$$R_p = \rho_p \Psi I_L (e^{-\gamma \sum_{r=p}^{P_{in}} \rho_r} + \sum_N I_{msp}). \tag{3}$$

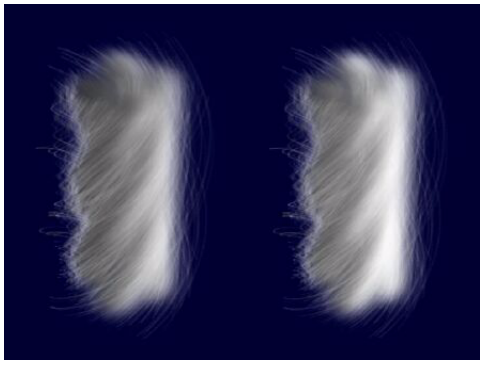

Figure 5: Effects of multiple scattering. Left side: without multiple scattering; right side: with multiple scattering.

R_p can then be restated in terms of the spherical angles in Figure 4:

$$R_p(\theta_l, \phi_l, \theta_v, \phi_v) = I_L C_p(\theta_l, \phi_l, \theta_v, \phi_v) \tag{4}$$

where

$$\begin{aligned} C_p(\theta_l, \phi_l, \theta_v, \phi_v) = \rho_p \Psi(\theta_l, \phi_l, \theta_v, \phi_v)[e^{-\gamma \sum_{r=p}^{P_{in}} \rho_r} \\ + \sum_{i \in N} I_{msp}(\theta_i, \phi_i)]. \end{aligned} \tag{5}$$

Here, C_p is the VRF, which is a 4D function of θ_l, ϕ_l, θ_v, and ϕ_v. It can be approximately represented by a 4D color array after discretization of the four angles, where the discretization increments are determined by the accuracy required. For all lumislices used in this paper, the longitude angle $\theta \in [0, 2\pi]$ and altitude angle $\phi_l \in [-\pi/2, \pi/2]$ are discretized into 32×21 directions, such that the 2D lumislice is computed from a 3D swept rotation of yarn cross-sections. In normal situations, the distance between the knitwear and the light source greatly exceeds the radius of the knitted yarn, so we assume parallel illumination. The spacing of the lumislices along yarn segments is related to the resolution of the lumislice, which is presented in Section 4.3.4 on levels of detail.

For greater realism, multiple scattering between voxels is calculated to simulate light scattering among the fibers. This effect is quantified in the I_{msp} term, which aggregates the reflected intensity from neighboring voxels. For a neighboring voxel N from direction (θ_i, ϕ_i), its contribution to the multiple scattering towards p is computed from (4) as

$$I_{msp}(\theta_i, \phi_i) = \rho_N \Psi(\theta_l, \phi_l, -\theta_i, -\phi_i) e^{-\gamma \sum_{r=N}^{P_{in}} \rho_r}. \tag{6}$$

Although multiple scattering could be computed iteratively from the surrounding voxels, we have found that a first-order approximation with adjacent voxels is adequate. The effect of the multiple scattering component on yarn rendering is illustrated in Figure 5, where the right strand with multiple scattering appears more realistic than the left strand that does not account for this effect.

For additional simplification, the specular component of the VRF is ignored in our implementation, resulting in a view-independent reflectance function. This allows C_p to become a 2D function, and consequently the computational costs and storage requirement are significantly reduced. All lumislices used in this paper consider only the diffuse effect.

The lumislice of a given yarn is precomputed and stored in tables, with the VRF in a 2D RGB color table indexed by (θ_L, ϕ_L) and the opacity in an alpha table.

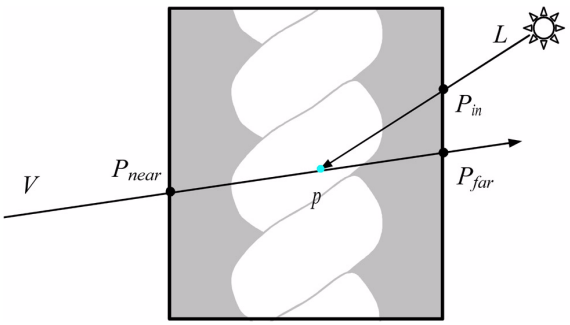

Figure 6: Radiance calculation through a yarn segment. An emission-absorption model is used to compute the contributions to viewed radiance at V from voxels between P_{near} and P_{far}.

4.1.2 Volumetric Yarn and Radiance Calculation

Since the same lumislice represents all cross-sections of a knitted yarn, it is rotated and translated along the yarn path to form the yarn volume, as shown in Figure 3. The radiance R of the knitwear at a given image pixel from viewpoint V in Figure 6 can be computed as

$$R = \sum_{p=P_{near}}^{P_{far}} e^{-\gamma \sum_{s=P_{near}}^{p} \rho_s} \cdot R_p \tag{7}$$

where R_p is calculated in (1).

When the knitwear is rendered, all yarn segments are subdivided into a series of volumetric cross-sections, each represented by the lumislice. When a slice is drawn, its location, its local coordinate system $< t, u, v >$ as shown in Figure 7, the light source position and the viewpoint together determine the reflection angles θ_L, ϕ_L, θ_V, and ϕ_V, as shown in Figure 4. These angles, specifically only θ_L and ϕ_L, are then used to index the 2D color table of the lumislice. This color texture and the opacity table of the slice together form the RGBA transparency texture, which is rendered by the texture mapping and transparency-blending functions of standard graphics APIs [20].

4.2 Pattern Stitching

The path of the yarn, and likewise the arrangement of the lumislices, is given by the process of pattern stitching. In pattern stitching, a knitwear skeleton is built according to a given object surface and stitch pattern. This method is detailed in [33] and is briefly described here.

The knitwear skeleton is a network of interlocking mathematical curves in 3D which interpolate control points that are placed according to the desired stitch pattern. An important feature of control points is that they make it easy to model and edit advanced stitch patterns [14]. An additional advantage of this framework is the modeling of interactions among yarn loops. This free-form structure for knitwear complements our lumislice method for realistic and efficient rendering.

4.3 Rendering

With stitching, the knitwear skeleton is created on a free-form surface. The yarn along the skeleton is divided into a series of short straight segments with a twisting angle ϕ. The Frenet frame $< t, n, b >$ is then defined for determining a local coordinate system $< t, u, v >$ for each slice, as is shown in Figure 7.

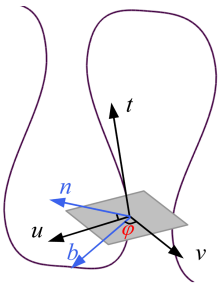

Figure 7: A local coordinate system $t - u - v$ for each volumetric slice is determined from the Frenet frame of yarn segments with twisting angle ϕ.

With these slices, we describe the rendering issues of depth sorting, slice partitioning, shadow casting, and levels of detail. This subsection concludes with a step-by-step description of our rendering algorithm.

4.3.1 Sorting

For the blending computation, the volumetric yarn slices must be drawn from far to near, thus requiring slice sorting before rendering. The number of slices can be extremely large for complex scenes; however, in our implementation it is the yarn segments, not the slices, that are sorted if the yarn segments are short. The view direction and the positioning of a yarn segment determine the orientation in which its slices are rendered.provides the effects shown on the right.

4.3.2 Slice Partitioning

The transparency-blending function supported by graphics hardware performs its blending computation along slices parallel to the viewport. For knitwear, lumislices can have various orientations, which may present difficulties for the blending function.

Serious artifacts may arise in rendering yarn slices whose orientation differs greatly from that of the viewport. As illustrated in Figure 8A, two slices with the same transparency texture can have the same blending result even when they are aligned differently and consequently have different thicknesses with respect to the viewer. This disparity in thickness should in principle be rendered dissimilarly for semi-transparent surfaces.

An approximate solution that we implement is to partition each slice according to its resolution, as illustrated in Figure 8B. Each of the resulting slice units is then rendered as a line, because polygons are taken to have zero thickness by the hardware. This would lead to slices not being rendered when they are perpendicular to the viewport. Because of hardware discretization of the scene with regards to the viewport pixels, the slice would effectively be rendered as shown in Figure 8C, where there may be more than one line projected to each pixel.

4.3.3 Shadows

Shadows are one of the most important factors for realism [27, 24], and there exist many well-known techniques for shadow generation. The shadow map method introduced by Williams [31] is perhaps the most basic. This technique was improved by Reeves *et al.* [22] and was extended to the so-called deep shadow map by Lokovic and Veach [19]. Shadows are particularly important for realistic rendering of knitwear because they not only indicate the spatial relationship between knitwear and other geometric objects, but also

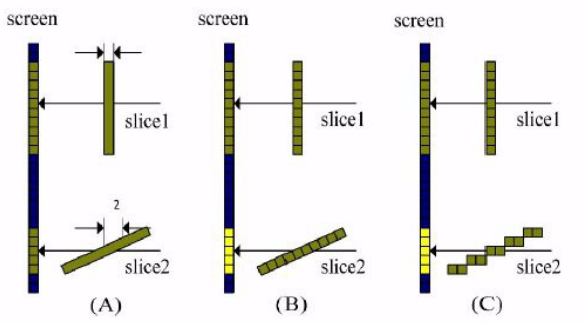

Figure 8: Partitioning of lumislices for transparency blending. (A) For two surfaces with the same transparency texture, differences in orientation do not lead to different blending results if there is no slice partitioning parallel to the screen. (B) When slice2 is partitioned according to resolution, the blending result reflects the difference in thickness between the two surfaces as seen from the screen. (C) Because of discretization, more than one voxel may map to the same screen pixel.

visually reveal the knitting pattern of yarns. Figure 9 exemplifies the significant effects of shadows.

Deep shadow maps can deal with semi-transparent or sub-pixel objects such as hair, fur and smoke because they represent the fractional visibility through a pixel at all possible depths, rather than only a single depth value. However, this technique is incompatible with the lumislice, because part of the light attenuation has been already computed and stored in the lumislice to provide better effects of yarn microstructure. We therefore extend the traditional shadow map to handle yarn, an object with geometry not clearly defined. Experimental evidence suggests that while shadows are important in 3D rendering, they need not be exact [27, 24], so we develop a rough shadowing technique consisting of two steps.

Like traditional shadow map techniques, the first step of our method places a camera at the light source, making sure that all shadow generators and receivers are within the field of view. All objects except for knitwear are then projected to produce a shadow map containing a depth value. For knitwear, only the centerlines of yarns are projected to the shadow map. Thus the shadow map we build will contain depth information of geometric objects and yarn centers.

Next, the effect range R_y of yarn in the shadow map is computed according to the radius of yarn. This parameter is used to determine whether light is partially absorbed by the yarn before reaching more distant positions.

Finally, before a point p is rendered, it is projected onto the shadow map for shadow testing. If the projection result $p_s(x_s, y_s, z_s)$ indicates total occlusion by geometric surfaces, then it is in shadow. Otherwise, the yarn-occlusion test is performed as follows: on the shadow map, if the closest distance d from (x_s, y_s) to the centerline of any yarn is smaller than R_y, light is partially absorbed by that yarn. Given the fluff density distribution ρ as shown in Figure 10, we compute the light transmission for distance d through voxels (d, i) of the lumislice as

$$T_y(d) = 1 - k_0 \frac{1 - \prod_{i=-R_y}^{R_y} e^{-\gamma\rho(d,i)}}{1 - \prod_{i=-R_y}^{R_y} e^{-\gamma\rho(0,i)}} \qquad (8)$$

where we empirically take $k_0 = 0.8$ and the voxel dimensions to be unit length.

The lumislice-based method renders knitwear by a volumetric yarn slice using hardware-aided transparency blending. As

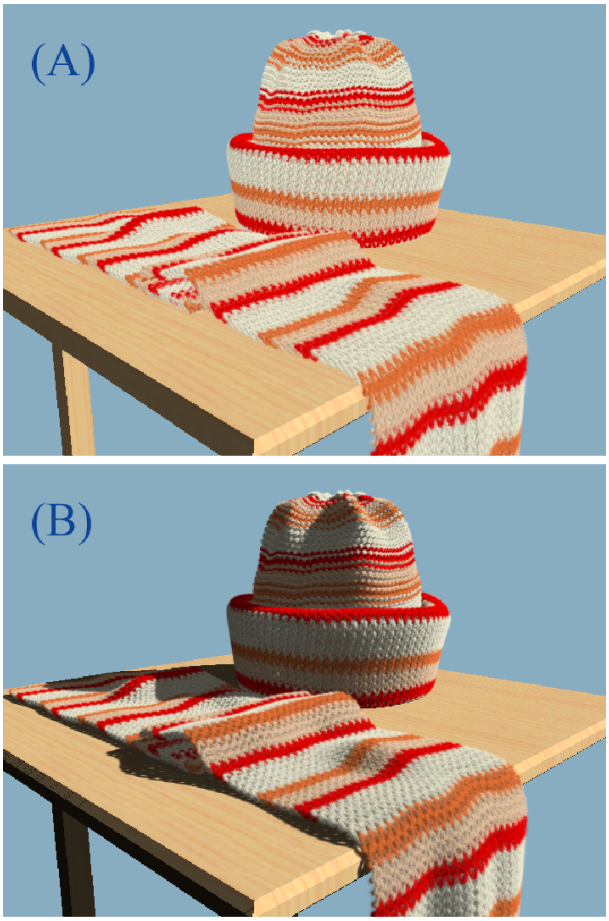

Figure 9: Effects of shadows in rendered knitwear. (A) Synthesized image without consideration of shadows. (B) Enhanced realism resulting from our shadow technique.

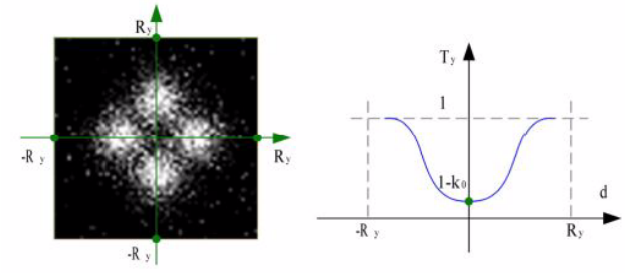

Figure 10: Soft shadows from knitwear. Transmission factor T_y of light passing a distance $d \leq R_y$ from a yarn center.

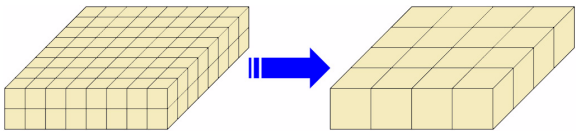

Figure 11: Lumislice levels of detail. To form the lower resolution lumislice on the right, a high resolution lumislice is doubled in thickness, and then groups of eight voxels are averaged according to (9).

described in [20], shadow cannot be calculated for every pixel on a texture if the alpha-texture is employed to render volume-represented objects. Thus the shadow test can be performed only at the slice vertices, and the result is blended to the lumislice by adjusting the texture brightness so that the shadow effect can be seen. If higher accuracy is required, such as for rendering close-up views, the slice could be divided into sub-slices so that shadow computation can be done at more points to improve the result quality.

4.3.4 Level of Detail Representation

To handle a wide range of viewing distances, a view-dependent level of detail representation of the lumislice is utilized to reduce the computational cost, to ensure smooth visual transitions, and to reduce rendering error. When the viewing distance changes, the projection size and visible features of a yarn slice on the screen should change accordingly. A lumislice of fixed resolution will cause error because there is no simple method for determining proper opacity values even though the color values of the lumislice can be obtained by using a mip-map. For this reason, we precompute lumislices of different resolutions and render each slice with the one most appropriate for its current viewing condition.

For decreasing resolution of the lumislice, its unit of thickness should increase by the same factor. For instance, we can form a low-resolution lumislice as the composite of two identical high-resolution lumislices, as illustrated in Figure 11. Locally, a voxel on a low-resolution slice can be computed from corresponding voxels in a high-resolution slice. A reasonable estimate of the low-resolution density ρ' is its average value, and of the color C' as a weighted average:

$$\rho' = \frac{\rho_1 + \rho_2 + \rho_3 + \rho_4}{4}$$

$$C' = \frac{C_1\rho_1 + C_2\rho_2 + C_3\rho_3 + C_4\rho_4}{\rho_1 + \rho_2 + \rho_3 + \rho_4} \qquad (9)$$

where C_1, C_2, C_3 and C_4 are colors of the high-resolution slice, and ρ_1, ρ_2, ρ_3 and ρ_4 are densities of the high-resolution slice. Because of thickness doubling, the estimate of the transparency is $T' = (e^{-\gamma\rho'})^2$. For more accurate results when the resolution is below 8×8, we form the low-resolution lumislice by sampling the high-resolution lumislice.

The resolution for rendering a slice depends upon the projection area of the slice on the screen. For a projection whose dimensions are at most m pixels, a lumislice with resolution $2^n \times 2^n$ is chosen such that n is an integer and $2^n \geq m$, to prevent aliasing. Although our implementation precomputes lumislices at intervals according to 2^n, finer intervals may be used for more accuracy.

The unit length corresponding to the determined resolution also describes the thickness of the lumislice. Lumislice densities along yarn are determined by dividing yarn segments by these computed resolution units. In other words, if we consider lumislices as being two-dimensional, the resolution unit describes the distance between lumislices along yarn segments.

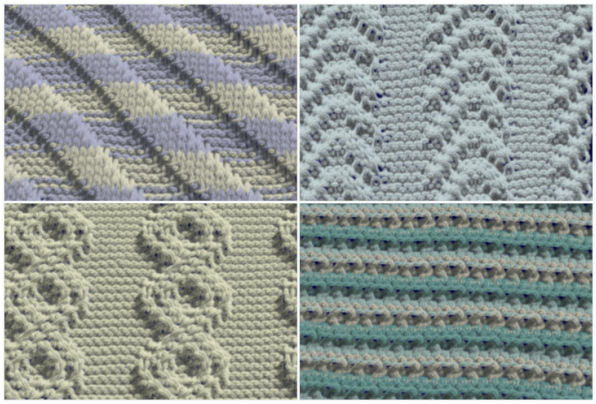

Figure 12: Rendered stitching patterns that depict various knitting styles which can be synthesized by lumislice-based rendering. The desirable effects of soft shadows are evident.

4.3.5 Rendering Algorithm

From the described processes, we outline the steps of the rendering algorithm, given models of the lumislice and the knitwear skeleton. The use of hardware-aided graphics APIs entails the masking or disabling of the z-buffer depth test for accurate transparency-blending computation. Consequently, scene rendering requires more than one step. Our rendering routine follows these steps:

1. Create the shadow map as described in Section 4.3.3.
2. Using the ID color method, draw the color and depth of geometric objects as seen from the viewport into Buffer1.
3. Draw yarn as cylindrical polygons on the result of step 2, then turn the resulting image into a bitmap called Image1. Pixels that are covered by yarn are set to 0, and others are set to 1.
4. Draw the geometric objects with the shadow test under the same projection condition and then save as Image2.
5. Sort all discrete yarn segments by distance from the viewpoint.
6. Disable the depth test, draw volumetric yarn on the result of step 4. Before each slice is rendered, project its vertices into Buffer1. If it is totally occluded by geometric objects, it is not rendered. Render all slices of the yarn segments from far to near. Save the result as Image3.
7. For every pixel in Image3, check its associated value in Image1. If the value is 1, then this pixel should display a geometric object, so the color of this pixel is replaced by the color of the corresponding pixel in Image2. This step is necessary to properly process slices that are partially occluded. The final result is obtained after all the pixels are checked.

5 Results

Several synthesis results of our method are displayed in this section. Figure 12 exhibits a range of complex stitching patterns that can be rendered with our technique. The soft shadows in the knitwear are evident and contribute significantly to the realism of the images.

Another set of renderings is shown in Figure 13. The left column displays real images of a scarf, while renderings from corresponding view distances are shown on the right. Comparison of the two

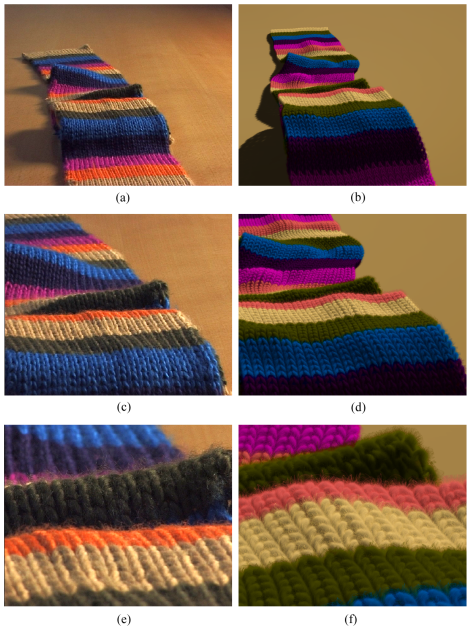

Figure 13: Comparison of lumislice-based rendering to actual images. (a)(c)(e) display real images of a scarf captured from various viewing distances. (b)(d)(f) show corresponding renderings by our method.

columns demonstrates the similarity of our synthesis results to actual knitwear. A third group of images in Figure 14 is captured from an animation and exhibits varying levels of detail and soft shadowing. Lastly, Figure 15 shows a sweater that displays deformations that can easily be made particular to a given person.

These renderings were performed on a 864MHz Pentium III with 256M RAM and a Matrox Millenium G400 DualHead Max display adapter. For the images in Figure 12, the number of rendered slices are, clockwise from the upper left, 211k, 215k, 225k and 216k. The respective rendering times in minutes using our unoptimized implementation are 14.30, 15.07, 15.33 and 15.08. Another example of the rendering time is the sweater in Figure 15, which consists of 368k slices and took 31.93 minutes.

All images presented in this paper utilize a single light source. Multiple light sources can potentially be handled using multi-texturing.

6 Conclusion

In this paper, we presented a reflectance element called the lumislice designed to facilitate the efficient photorealistic rendering of free-form knitwear. The lumislice represents the reflected radiance of light, as influenced by the yarn microstructure. It provides an effective means for propagating local microstructure features over a knitwear while allowing for arbitrary global characteristics, such as the stitch pattern and overall knitwear shape. In conjunction with a proposed shadow technique for yarn, the lumislice can provide convincing visual results.

Although all examples in this paper are knitwear, the lumislice can potentially be applied to other fabric objects consisting of repetitive macroscopic structures, such as carpet and coarse weaves. Additional possibilities for future work include development of a smooth transition between levels of detail at closer viewing distances and BRDFs from farther away.

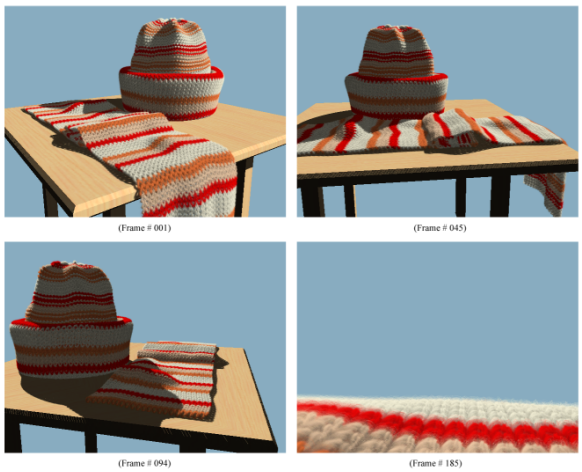

Figure 14: Frames from a fly-by animation.

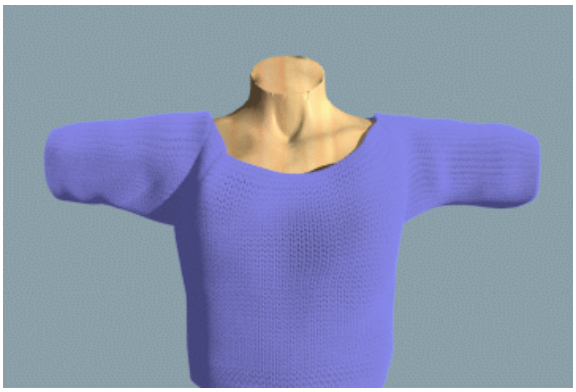

Figure 15: Sweater worn on a person.

Acknowledgement

We thank David Salesin and Xin Tong for useful discussions, and the SIGGRAPH anonymous reviewers for their constructive critiques which have led to significant clarification of the paper. Dr. Enhua Wu is partially supported by grants from the National Natural Science Foundation of China (69873044) and the University of Macau (RG029/00-01S/WEH/FST).

References

[1] K. Anjyo, Y. Usami, and T. Kurihara. A Simple Method for Extracting the Natural Beauty of Hair. *Computer Graphics (Proceedings of SIGGRAPH 92)*, 26(2):111–120, July 1992.

[2] D. Baraff and A. Witkin. Large Steps in Cloth Simulation. *Proceedings of SIGGRAPH 98*, pages 43–54, July 1998.

[3] D. E. Breen, D. H. House, and P. H. Getto. A Physically-Based Particle Model of Woven Cloth. *The Visual Computer*, 8(5-6):264–277, June 1992.

[4] D. E. Breen, D. H. House, and M. J. Wozny. Predicting the Drape of Woven Cloth Using Interacting Particles. *Proceedings of SIGGRAPH 94*, pages 365–372, July 1994.

[5] D. E. Breen, D. H. House, and M. J. Wozny. A particle-based model for simulating the draping behavior of woven cloth. *Textile research joural*, 64(11):663–685, November 1994.

[6] S. Chandrasekar. *Radiative Transfer*. Dover Publications, New York, 1960.

[7] M. Courshesnes, P. Volino, and N. M. Thalmann. Versatile and Efficient Techniques for Simulating Cloth and Other Deformable Objects. *Proceedings of SIGGRAPH 95*, pages 137–144, August 1995.

[8] K. J. Dana, B. van Ginneken, S. K. Nayar, and J. J. Koenderink. Reflectance and Texture of Real-world Surfaces. *ACM Transactions on Graphics*, 18(1):1–34, January 1999.

[9] B. Eberhardt, A. Weber, and W. Straßer. A Fast, Flexible, Particle-System Model for Cloth Draping. *IEEE Computer Graphics & Applications*, 16(5):52–60, September 1996.

[10] D. B. Goldman. Fake Fur Rendering. *Proceedings of SIGGRAPH 97*, pages 127–134, August 1997.

[11] S. J. Gortler, R. Grzeszczuk, R. Szeliski, and M. F. Cohen. The Lumigraph. *Proceedings of SIGGRAPH 96*, pages 43–54, August 1996.

[12] E. Gröller, R. T. Rau, and W. Straßer. Modeling and Visualization of Knitwear. *IEEE Transactions on Visualization and Computer Graphics*, 1(4):302–310, December 1995.

[13] E. Gröller, R. T. Rau, and W. Straßer. Modeling Textiles as Three Dimensional Textures. *Eurographics Rendering Workshop 1996*, pages 205–214, June 1996.

[14] Harmony. *The Harmony Guide to Knitting Techniques*. Collins & Brown, 1992.

[15] D. H. House and D. E. Breen. *Cloth Modeling and Animation*. A K Peters, Natick Massachusetts, 2000.

[16] J. T. Kajiya and T. L. Kay. Rendering Fur with Three Dimensional Textures. *Computer Graphics (Proceedings of SIGGRAPH 89)*, 23(3):271–280, July 1989.

[17] J. Lengyel. Real-Time Fur. *Eurographics Rendering Workshop 2000*, pages 243–256, June 2000.

[18] M. Levoy and P. Hanrahan. Light Field Rendering. *Proceedings of SIGGRAPH 96*, pages 31–42, August 1996.

[19] T. Lokovic and E. Veach. Deep Shadow Map. *Proceedings of SIGGRAPH 2000*, pages 385–392, July 2000.

[20] A. Meyer and F. Neyret. Interactive Volumetric Textures. *Eurographics Rendering Workshop 1998*, pages 157–168, June 1998.

[21] F. Neyret. Modeling, Animating, and Rendering Complex Scenes Using Volumetric Textures. *IEEE Transactions on Visualization and Computer Graphics*, 4(1):55–70, January-March 1998.

[22] W. T. Reeves, D. H. Salesin, and R. L. Cook. Rendering Antialiased Shadows with Depth Maps. *Computer Graphics (Proceedings of SIGGRAPH 87)*, 21(4):283–291, July 1987.

[23] B. Robertson. Building a better mouse. *Computer Graphics World*, 22(12), December 1999.

[24] C. Soler and F. X. Sillion. Fast Calculation of Soft Shadow Textures Using Convolution. *Proceedings of SIGGRAPH 98*, pages 321–332, July 1998.

[25] D. Terzopoulos and K. Fleischer. Deformable models. *The Visual Computer*, 4(6):306–331, December 1988.

[26] N. M. Thalmann, S. Carion, M. Courchesne, P. Volino, and Y. Wu. Virtual Clothes, Hair and Skin for Beautiful Top Models . *Computer Graphics International*, 1996.

[27] L. R. Wanger, J. A. Ferwerda, and D. P. Greenberg. Perceiving Spatial Relationships in Computer-Generated Images. *IEEE Computer Graphics and Application*, 12(3):44–58, May 1992.

[28] S. H. Watson, J. R. Arvo, and K. E. Torrance. Predicting Reflectance Functions From Complex Surfaces. *Computer Graphics (Proceedings of SIGGRAPH 92)*, 26(2):255–264, July 1992.

[29] J. Weil. The Synthesis of Cloth Objects. *Computer Graphics (Proceedings of SIGGRAPH 86)*, 20(4):49–54, August 1986.

[30] S. H. Westin, J. R. Arvo, and K. E. Torrance. Predicting Reflectance Functions From Complex Surfaces. *Computer Graphics (Proceedings of SIGGRAPH 92)*, 26(2):255–264, July 1992.

[31] L. Williams. Casting Curved Shadows on Curved Surfaces. *Computer Graphics (Proceedings of SIGGRAPH 78)*, 21(3):270–274, August 1978.

[32] T. Yasuda, S. Yokoi, and J. Toriwaki. A Shading Model for Cloth Objects. *IEEE Computer Graphics & Applications*, 12(6):15–24, November 1992.

[33] H. Zhong, Y. Xu, B. Guo, and H. Shum. Realistic and Efficient Rendering of Free-Form Knitwear. *Journal of Visualization and Computer Animation, Special Issue on Cloth Simulation*, 2000.