Fast Visualization of Special Relativistic Effects on Geometry and Illumination

Daniel Weiskopf

Institute for Astronomy and Astrophysics
University of Tübingen
Auf der Morgenstelle 10, D-72076 Tübingen, Germany
weiskopf@tat.physik.uni-tuebingen.de

Abstract. This paper describes a novel rendering technique for the special relativistic visualization of the geometry and illumination of fast moving objects. The physical basis consists of the relativistic aberration of light, the Doppler effect, and the searchlight effect. They account for changes of apparent geometry, color, and brightness of the objects. The rendering technique makes use of modern computer graphics hardware, in particular texture mapping and advanced per-pixel operations, and allows the visualization of these important special relativistic effects at interactive frame rates.

1 Introduction

Einstein's Special Theory of Relativity is widely regarded as a difficult and hardly comprehensible theory. One important reason for this is that the properties of space, time, and light in relativistic physics are totally different from those in classical, Newtonian physics. In many respects, they are contrary to human experience and everyday perception because mankind is limited to very low velocities compared to the speed of light. Therefore, computer simulations are the only means of visually exploring the realm of special relativity and can thus help the intuition of physicists and other people interested in the theory.

Hsiung and Dunn[5] were the first to use advanced visualization techniques for image shading of fast moving objects. They proposed an extension of normal three-dimensional ray tracing. Hsiung et al.[6] described a polygon rendering approach which is based on the apparent shapes of objects as seen by a relativistic observer.

In our previous work[13], we introduced texture-based relativistic rendering as a new means of visualizing the apparent geometry of fast moving objects. In this paper, the texture mapping approach is extended to the visualization of the special relativistic effects on both geometry and illumination. The physical basis consists of the relativistic aberration of light, the Doppler effect, and the searchlight effect. The novel rendering method makes extensive use of the functionality available on high-end graphics workstations[9], such as pixel textures and 3D textures. These features are exploited through OpenGL and its extensions.

Unlike the relativistic polygon rendering approach, texture-based relativistic rendering does not transform the coordinates or the color information of the vertices, but

transforms the images which are rendered in the normal non-relativistic way. Therefore, the standard rendering pipeline is not changed and only an additional step is added at the end of the rendering process. The relativistic transformation is performed on the image plane by texture mapping. This transformation is split in two phases. The first phase determines the geometrical effects by using standard 2D texture mapping. The second phase implements the Doppler and searchlight effects by using pixel textures and 3D texture mapping. In this way, interactive frame rates are achieved on modern computer graphics hardware.

2 Lorentz Transformation

This section describes only those terms and equations of special relativity which are relevant for rendering. A detailed presentation of special relativity can be found in [7, 8, 11]. The effects used from relativistic physics are the relativistic aberration of light, the Doppler effect, and the searchlight effect.

The relativistic aberration of light causes a rotation of the direction of light when one is changing from one inertial frame of reference to another. The aberration of light is sufficient to completely describe the apparent geometry seen by a fast moving camera.

The Doppler effect accounts for the transformation of wavelength from one inertial frame of reference to another and causes a change in color.

The searchlight effect is based on the transformation of wavelength dependent radiance from one inertial frame of reference to another. The transformation of radiance increases the brightness of objects ahead when the observer is approaching these objects at high velocity. The definition of wavelength dependent radiance can be found, e.g., in [3, Chapt. 13][14].

Let us consider two inertial frames of reference, S and S', with S' moving with velocity v along the z axis of S. The usual Lorentz transformation along the z axis connects frames S and S'.

In reference frame S, consider a photon with the wavelength λ and the direction determined by spherical coordinates (θ, ϕ). In frame S', the photon is described by the wavelength λ' and the direction (θ', ϕ'). These two representations are connected by the expressions for the relativistic aberration of light, cf. [8],

$$\cos\theta' = \frac{\cos\theta - \beta}{1 - \beta\cos\theta}, \qquad \phi' = \phi, \tag{1}$$

and for the Doppler effect,

$$\lambda' = \lambda\, D. \tag{2}$$

The Doppler factor D is defined as

$$D = \frac{1}{\gamma\,(1 - \beta\cos\theta)} = \gamma\,(1 + \beta\cos\theta'),$$

where $\gamma = 1/\sqrt{1-\beta^2}$, $\beta = v/c$, and c is the speed of light.

Wavelength dependent radiance is transformed from one frame of reference to another according to

$$L'_\lambda(\lambda', \theta', \phi') = D^{-5} L_\lambda(\lambda, \theta, \phi). \tag{3}$$

A derivation of this relation can be found in [14].

3 Relativistic Rendering

The physical basis for texture-based relativistic rendering is the Lorentz transformation of properties of light. Suppose that the photon field is known at one point in spacetime, i.e. at a point in three-dimensional space and at an arbitrary but fixed time. This photon field is measured in one frame of reference which is denoted S_{obj}.

Now consider an observer, i.e. a camera, at this point in spacetime. In the following, the generation of a snapshot taken by this camera is investigated. The observer is not at rest relative to S_{obj}, but is moving at arbitrary speed relative to S_{obj}. However, the observer is at rest in another frame of reference, S_{observer}. The photon field can then be calculated with respect to S_{observer} by the Lorentz transformation from S_{obj} to S_{observer}. Finally, the transformed photon field is used to generate the picture taken by the observer's camera.

Now we restrict ourselves to static scenes, i.e. all scene objects and light sources are at rest relative to each other and relative to S_{obj}. Here, all relevant information about the photon field—namely direction and wavelength dependent radiance of the incoming light—can be determined by standard computer graphics algorithms, since the finite speed of light can be neglected in all calculations for the static situation. With this information and with the use of the equations for the relativistic aberration of light and for the Doppler and searchlight effects, the picture seen by the relativistic observer can be generated. In this way, the relativistic effects on geometry, color, and brightness are taken into account.

3.1 Representation of the Photon Field

The relevant information about the photon field is the direction of the light and the spectral power distribution in the form of the wavelength dependent radiance. This is the information about the full plenoptic function[1] $P(\theta, \phi, x, y, z, t, \lambda)$ at a single point (x, y, z, t) in spacetime, effectively leaving a three-parameter function $\tilde{P}(\theta, \phi, \lambda)$. The polarization of light is neglected.

The function $\tilde{P}$ can be sampled and stored in the form of an image projected onto the unit sphere, with the camera being at the midpoint of the sphere. This image is called *radiance map*. The direction of an incoming light ray is determined by the spherical coordinates (θ, ϕ) of the corresponding point on the sphere.

The wavelength dependent radiance can be described by a vector with respect to a fixed set of n_L basis functions. Therefore, each point of the radiance map holds n_L components for the wavelength dependent radiance. Peercy[10] describes in detail how a finite set of orthonormal basis functions can be used to represent radiance. In this

paper, no projection onto the basis functions is needed. Therefore, the orthonormality condition can be omitted and an arbitrary set of basis functions can be used. In this representation the wavelength dependent radiance is

$$L_\lambda(\lambda) = \sum_{j=1}^{n_L} \epsilon_j E_j(\lambda), \tag{4}$$

with the set of basis functions, $\{E_j(\lambda)|j \in \mathbb{N}, 1 \leq j \leq n_L\}$, and the coefficients of the vector representation, ϵ_j.

In the literature, various sets of basis functions are described, such as box functions [4], Fourier functions[2], Gaussian functions[12], and delta functions[2]. As alternative basis functions, we propose the Planck spectral distributions at various temperatures with the wavelength dependent radiance

$$L_{\lambda,\text{Planck}}(\lambda, T) = \frac{2hc^2}{\lambda^5} \frac{1}{\exp\left(\frac{hc}{k\lambda T}\right) - 1},$$

where T is the temperature in Kelvin, h the Planck constant, and k the Boltzmann constant. The Planck distribution describes the blackbody radiation and thus has a direct physical meaning and a widespread occurance in nature.

For the display on the screen, three tristimulus values such as RGB have to be calculated from the wavelength dependent radiance. For example, the RGB values (c_R, c_G, c_B) can be obtained by

$$c_i = \int L_\lambda(\lambda) \bar{f}_i(\lambda)\, d\lambda, \qquad i = R, G, B, \tag{5}$$

where $\bar{f}_i(\lambda)$ are the respective color matching functions for RGB, cf. [15].

How can the radiance map be generated for the non-relativistic situation, i.e. a camera at rest in the frame S_{obj} of the objects? Since perspective projection is restricted to a maximal field of view of $180°$, the complete radiance map cannot be created in a single step by using standard computer graphics hardware. Therefore, the covering of the whole sphere is accomplished by projecting several images which are taken with differing orientations of the camera. A similar method is used for reflection and environment mapping.

Usually, images do not provide an arbitrary number of channels in order to store the n_L components for the photon field. However, several radiance maps each of which contain three standard RGB channels can be combined to represent the complete photon field.

3.2 Apparent Geometry

Let us assume that the photon field is known for an observer at rest in the frame S_{obj}. Now consider the same situation with the moving observer. The photon field has to be changed by the Lorentz transformation from S_{obj} to S_{observer}. This transformation is split in two parts, in the relativistic aberration of light and in the transformation of

wavelength and radiance. Accordingly, the relativistic part of the rendering algorithm consists of two phases which determine apparent geometry and relativistic illumination.

The relativistic aberration of light yields a transformation of (θ, ϕ) to (θ', ϕ') according to (1). The resulting distortion of the image mapped onto the sphere is illustrated in Fig. 1.

Fig. 1. Effect of the relativistic aberration of light on the texture mapped onto the unit sphere. The left sphere shows the mapping without distortion, the right sphere illustrates the distortion for 90 percent of the speed of light. The direction of motion shows towards the north pole of the sphere.

Usually, the direction of motion is not identical to the z axis. Therefore, additional rotations of the coordinate system have to be considered. The complete mapping of the coordinates of a point (u, v) in the original image to the spherical coordinates (θ', ϕ') of the corresponding point seen by the moving observer is

$$T_{\text{mapping}} = T_{\text{rot, b}} \circ T_{\text{aberration}} \circ T_{\text{rot, a}} \circ T_{\text{proj}} \quad : \quad [0,1] \times [0,1] \longrightarrow S^2.$$

S^2 is the unit sphere. The map T_{proj} projects the rendered image onto the sphere and determines the corresponding coordinates on the sphere. The coordinates of the pixels are in the interval $[0, 1]$. The rotation $T_{\text{rot, a}}$ takes into account that the direction of motion differs from the z axis of the sphere. The actual relativistic transformation is accomplished by $T_{\text{aberration}}$ with the use of (1). Finally, $T_{\text{rot, b}}$ describes the rotation of the observer's coordinate system relative to the direction of motion.

The relativistic rendering process has to generate the texture coordinates for the mapping of the non-relativistic images onto the unit sphere which surrounds the moving observer. With the inverse map T^{-1}_{mapping}, the texture coordinates can be calculated from the spherical coordinates (θ', ϕ') in the coordinate system of the observer.

3.3 Relativistic Illumination

The Doppler and searchlight effects account for the relativistic effects on illumination. According to (2) and (3), both effects depend on the Doppler factor D. Therefore, in addition to the photon field, the information about the Doppler factors has to be known.

Here, we introduce the term *Doppler factor map*. Analogous to the radiance map, the Doppler factor map holds the Doppler factors for various directions. The Doppler factor map is a one-parameter map because, for a given velocity, the Doppler factor depends only on the angle θ'.

With (4) and (5), the tristimulus values (c'_R, c'_G, c'_B) for each pixel in the frame S_{observer} can be calculated by

$$c'_i = \int \bar{f}_i(\lambda') L'_\lambda(\lambda')\, d\lambda' = \int \bar{f}_i(\lambda') \sum_{j=1}^{n_L} \epsilon_j E'_j(\lambda')\, d\lambda' = \sum_{j=1}^{n_L} X_{i,j}(\epsilon_j, D), \qquad (6)$$

with

$$X_{i,j}(\epsilon_j, D) = \int \bar{f}_i(\lambda') \epsilon_j E'_j(\lambda')\, d\lambda' \quad , \qquad i = R, G, B, \quad j = 1 \ldots n_L.$$

The transformed $E'_j(\lambda')$ are computed from the original $E_j(\lambda)$ according to (2) and (3). The c'_i and $X_{i,j}$ can be combined to form the three-component vectors $\boldsymbol{c'}$ and $\boldsymbol{X_j}$.

The function $\boldsymbol{X_j}(\epsilon_j, D)$ can be represented by a three-component look-up table (LUT) depending on the two variables ϵ_j and D. This LUT can efficiently be implemented by using pixel textures. Pixel textures assign texture coordinates on a per-pixel basis instead of a per-vertex basis. Pixel textures are specified in the same way as normal 3D textures. With a pixel texture being activated, all pixels which are drawn from main memory into the frame buffer are interpreted as texture coordinates, i.e. each RGB color triple is mapped into the texture and then the interpolated texture values are actually drawn.

The LUT for each function $\boldsymbol{X_j}(\epsilon_j, D)$ is stored in a 3D RGB texture. A 2D texture would be sufficient for a two-parameter function. OpenGL, however, does not support 2D pixel textures. Therefore, a third, dummy dimension which is set to one is included. The LUTs do not change for a fixed set of basis functions E_j. Therefore, the respective pixel textures can be built in a preprocessing step, thus not impairing rendering performance.

Finally, the relativistic transformation of wavelength and radiance is implemented as follows. Another sphere, now textured by the Doppler factor map, is rendered in addition to the sphere resulting from Sect. 3.2. The Doppler factors are stored in a previously unused, separate channel, such as the α channel. The final RGB values are evaluated according to (6) by iterating over the n_L channels which contain the vectors ϵ_j, by applying the corresponding pixel textures, and by adding up the results.

If n_L is greater than three, more than one radiance map is used to hold the n_L channels, and the whole process above has to be performed several times depending on the number of different radiance maps.

3.4 Rendering Process

Fig. 2 shows the structure of the relativistic rendering process. In addition to normal non-relativistic rendering, two phases for the relativistic transformations are appended to the rendering pipeline.

The geometric effects are taken into account in the first phase which consists of the white and gray boxes in the diagram. This phase resembles the implementation of reflection or environment mapping onto a sphere. The main difference is localized in the generation of the texture coordinates. The operations in the white boxes work on

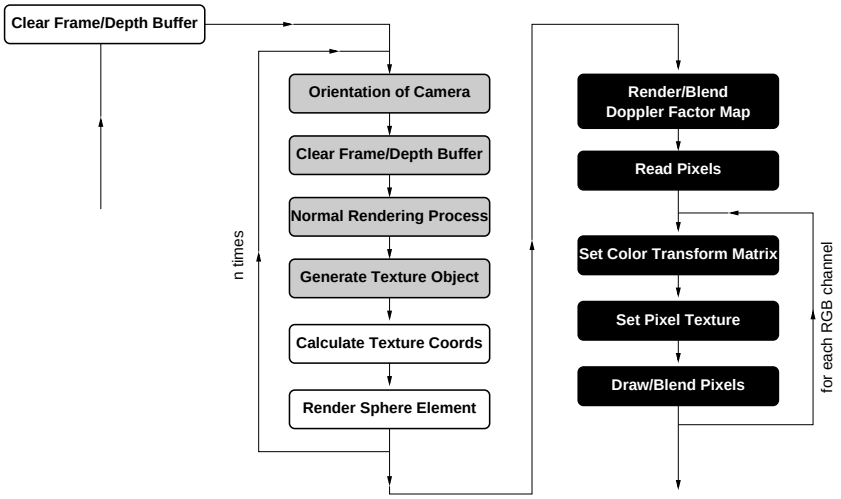


Fig. 2. Structure of the relativistic rendering process.

the back buffer of the normal frame buffer. The operations in the gray boxes work on an additional, off-screen frame buffer. In our implementation, pbuffers (`SGIX_pbuffer` extension) are used.

In the gray part, the textures for a spherical mapping are generated. Here, the standard non-relativistic rendering process is performed n times; n is the number of textures mapped onto the sphere and depends on the viewing angle and the orientation that are used for the rendering of the texture images. The OpenGL command `glCopyTexImage2D` transfers the results of non-relativistic rendering from the frame buffer to texture memory. Texture objects (`glBindTexture`) allow fast access to the stored textures.

In the white part, the texture coordinates are calculated with the use of T^{-1}_{mapping}. Then the textured sphere is actually drawn. The relativistic transformation is absorbed into the calculation of the texture coordinates. For the generation of the intermediate image which contains the visualization of apparent geometry, a picture is taken from inside the sphere. Note that the viewpoint has to be at the midpoint of the sphere, whereas the orientation is not restricted and allows for viewing into arbitrary directions.

The relativistic effects on illumination are taken into account in the second phase which consists of the black boxes. This phase works on the back buffer of the standard frame buffer. Another sphere, now textured by the Doppler factor map, is rendered into the α channel and blended with the result of the previous rendering steps. Then the complete RGBα buffer is read into main memory.

The last part is iterated three times, for the respective three color channels which hold the corresponding values ϵ_j. A color matrix (`SGI_color_matrix` extension) is set, which shifts the current color channel to the red channel and the α channel to the green channel. Now, the current values of ϵ_j are stored in the red channel and the corresponding Doppler factors in the green channel. Then the pixel texture which transforms

(ϵ_j, D) to displayable RGB values is applied. Finally, the operations in the rasterization stage are performed by drawing the image back into the frame buffer. These results are added up by a blending operation in order to obtain the final image.

So far, the rendering process supports up to three different basis functions E_j. A larger number of n_L can be implemented by iterating the whole rendering process several times with varying sets of basis functions and by adding up the results.

4 Implementation and Results

The relativistic rendering algorithm is implemented in C++ and runs on top of a standard OpenGL renderer. OpenGL version 1.1 with pbuffer (`SGIX_pbuffer`), pixel texture (`glPixelTexGenSGIX`), and color matrix (`SGI_color_matrix`) extensions is used. The implementation runs on an SGI Octane with Maximum Impact graphics and 250MHz R10000 processor.

An example of relativistic rendering can be found in Fig. 3. It illustrates the aberration of light, the Doppler effect, and the searchlight effect. The latter is very prominent and causes an extreme change of brightness.

In the current implementation, a constant number of six textures is used to cover the whole sphere. The six textures originate from the projection of the six sides of a cube onto the sphere. The texture coordinates are only recalculated if the velocity has changed. Three arbitrary basis functions E_j are supported. The following performance measurements are based on the test scene from Fig. 3 and a window size of 800*600 pixels. A frame rate of 7.6 fps is achieved for the relativistic visualization of apparent geometry and of 4 fps for the visualization of geometry and illumination. Approximately 30% of the total rendering time is used for the generation of the non-relativistic images, 10% for transferring these images from the frame buffer to texture memory, 40% for the pixel operations, and 20% for the remaining tasks.

5 Discussion

In this section, texture-based relativistic rendering is conferred to the well-known relativistic polygon rendering method.

Relativistic polygon rendering extends the normal rendering pipeline by a relativistic transformation of the vertex coordinates for the triangle meshes representing the scene objects. Since this transformation is non-linear, artifacts are introduced by the linear connection between the transformed vertices through straight edges. The error depends on the angular span under which each single triangle is viewed and might become very large for objects passing by closely. The artifacts can be reduced by a fine remeshing of the original objects in a preprocessing step or by an adaptive subdivision scheme during runtime. This is not needed for the texture approach, since the relativistic transformation is rather performed at the end of the rendering pipeline and affects every pixel in the image plane. Therefore, texture-based relativistic rendering does not need any modifications of the scene or the core rendering method. It does not increase the number of triangles of the scene objects to be rendered, it has no extra computational costs per vertex, and it does not introduce the geometric artifacts described above.

In the texture mapping approach, the sphere surrounding the observer is represented by a triangle mesh. The texture coordinates which are computed for the pixels inside each triangle by the usual perspective correction scheme differ from the true values. However, these errors do not impair the quality of the relativistic image as long as the angular span under which each single triangle is viewed is not too wide. The errors are independent of the geometry of the scene objects and can be controlled by choosing an appropriate size for the triangles representing the sphere, which is an important advantage over the relativistic polygon rendering approach. In the example depicted in Fig. 3, the whole sphere is tessellated by 5120 triangles, guaranteeing a good image quality for velocities as high as $\beta = 0.99$.

A very important issue is the physically correct handling of illumination. The reflection on a surface is usually described in terms of the angle between the light vector and the surface normal and the angle between the viewing vector and the surface normal. Relativistic polygon rendering transforms the coordinates of all vertices. Therefore, both angles are changed, which, in general, results in a wrong calculation of illumination. Texture-based relativistic rendering, however, completely transforms all relevant physical properties of the non-relativistic image to the frame of the fast moving camera, which brings about a physically correct visualization.

Rendering performance depends on different factors for relativistic polygon rendering and for texture-based relativistic rendering, cf. [13] for details. However, comparable frame rates are achieved by both rendering techniques on the used hardware.

One problem with texture-based relativistic rendering arises because of the properties of the aberration of light. The aberration equation does not conserve the element of solid angle. Therefore, the relativistic mapping does not conserve the area of an element on the sphere. The image is scaled down in the direction of motion, whereas the image gets magnified in the opposite direction. This magnification can reveal an inappropriate resolution of the texture. This issue could be solved by adapting the texture size to the relativistic distortion which depends on the observer's velocity and direction of motion.

Another problem could be the limited resolution of the RGBα channels, which might cause color aliasing effects whose extent depends on the chosen basis functions E_j and the interval of the used Doppler factors. These color aliasing effects are usually not very prominent for a depth of eight bits per channel, which is available on current hardware. They should completely disappear on future hardware supporting ten or twelve bits per channel.

6 Conclusion and Future Work

In this paper a texture-based approach to special relativistic rendering has been introduced. The physical basis consists of the relativistic aberration of light and the Doppler and searchlight effects. The algorithm uses texture mapping in order to transform the non-relativistic image into the frame of reference of a fast moving observer. Texture-based relativistic rendering allows for the fast visualization of special relativistic effects on both geometry and illumination.

There are several advantages compared to relativistic polygon rendering. There are no artifacts due to straight lines between vertices. There is no need for additional poly-

gons and for the computation of the transformation of vertices. Most importantly, a physically correct model for the calculation of illumination is implemented.

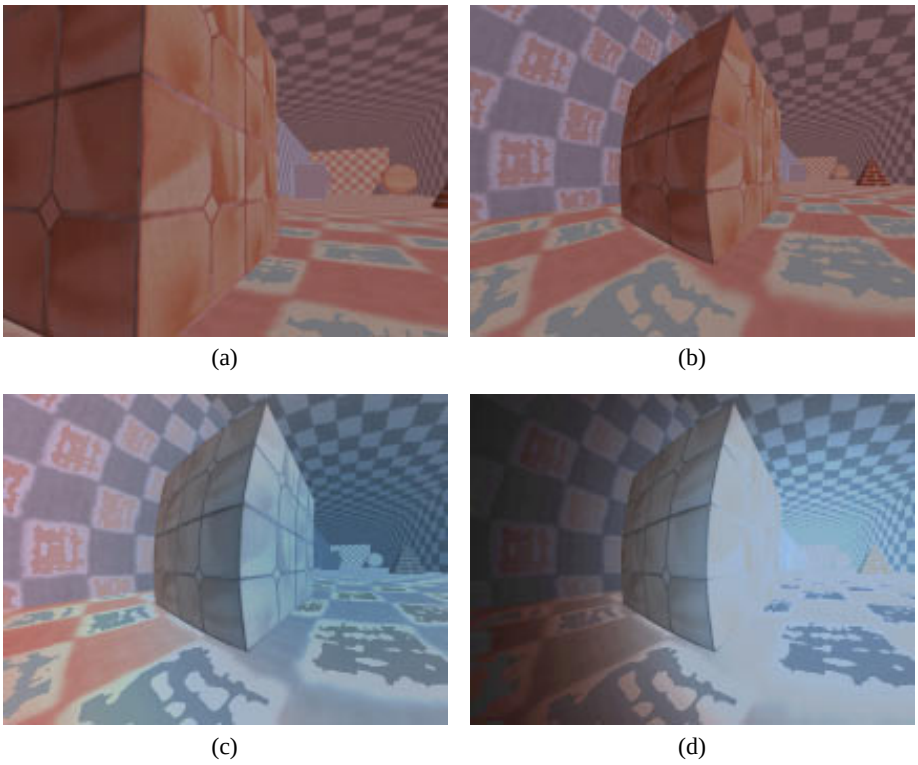
In future work, the issue of rendering performance and image quality will be addressed. A limiting part in the rendering process is the pixel fill rate for the generation of the radiance map. The number of textures can be reduced if only that part of the sphere which is actually viewed by the relativistic observer is covered by textures. In addition, the resolution of the textures could be adapted to the magnification by the aberration of light. This will increase rendering speed and enhance image quality.

Acknowledgments

This work was supported by the Deutsche Forschungsgemeinschaft (DFG) and is part of the project D4 within the Sonderforschungsbereich 382. Thanks to B. Salzer and J. Schulze-Döbold for proof-reading, and to the anonymous reviewers for helpful comments.

References

1. E. H. Adelson and J. R. Bergen. The plenoptic function and the elements of early vision. In M. Landy and J. A. Movshon, editors, *Computational Models of Visual Processing*, pages 3–20, Cambridge, 1991. MIT Press.
2. A. S. Glassner. How to derive a spectrum from an RGB triplet. *IEEE Computer Graphics and Applications*, 9(4):95–99, July 1989.
3. A. S. Glassner. *Principles of Digital Image Synthesis*. Morgan Kaufmann, San Francisco, 1995.
4. R. Hall and D. P. Greenberg. A testbed for realistic image synthesis. *IEEE Computer Graphics and Applications*, 3(8):10–20, Nov. 1983.
5. P.-K. Hsiung and R. H. P. Dunn. Visualizing relativistic effects in spacetime. In *Proceedings of Supercomputing '89 Conference*, pages 597–606, 1989.
6. P.-K. Hsiung, R. H. Thibadeau, and M. Wu. T-buffer: Fast visualization of relativistic effects in spacetime. *Computer Graphics*, 24(2):83–88, Mar. 1990.
7. C. W. Misner, K. S. Thorne, and J. A. Wheeler. *Gravitation*. Freeman, New York, 1973.
8. C. Møller. *The Theory of Relativity*. Clarendon Press, Oxford, second edition, 1972.
9. J. S. Montrym, D. R. Baum, D. L. Dignam, and C. J. Migdal. InfiniteReality: A real-time graphics system. In *SIGGRAPH 97 Conference Proceedings*, pages 293–302, Aug. 1997.
10. M. S. Peercy. Linear color representations for full spectral sampling. In *SIGGRAPH 93 Conference Proceedings*, pages 191–198, Aug. 1993.
11. W. Rindler. *Introduction to Special Relativity*. Clarendon Press, Oxford, second edition, 1991.
12. Y. Sun, F. D. Fracchia, T. W. Calvert, and M. S. Drew. Deriving spectra from colors and rendering light interference. *IEEE Computer Graphics and Applications*, 19(4):61–67, July/Aug. 1999.
13. D. Weiskopf. A texture mapping approach for the visualization of special relativity. In *IEEE Visualization 1999 Late Breaking Hot Topics Proceedings*, pages 41–44, 1999.
14. D. Weiskopf, U. Kraus, and H. Ruder. Searchlight and doppler effects in the visualization of special relativity: A corrected derivation of the transformation of radiance. *ACM Transactions on Graphics*, 18(3), July 1999.
15. G. Wyszecki and W. S. Stiles. *Color Science*. John Wiley & Sons, New York, second edition, 1982.



(a) (b)

(c) (d)

Fig. 3. Example of special relativistic rendering. The image (a) shows a non-relativistic view of the tunnel-like test scene. The scene emits blackbody radiation at 3500K, 5900K, and 10000K, producing the red, white, and blue colors in the non-relativistic image. In (b)–(d), the observer is moving towards the end of the tunnel with 60 percent of the speed of light. Picture (b) shows the visualization of apparent geometry, (c) adds the Doppler effect, and (d) illustrates the complete transformation of illumination. In (d), the overall intensity is reduced to 10 percent of that in (a)–(c) in order to keep the intensities in the displayable range.