Automatic Generation and Non-Photorealistic Rendering of 2+1D Minkowski Diagrams

Joachim Diepstraten, Daniel Weiskopf, Thomas Ertl

Visualisierung und Interaktive Systeme
Universität Stuttgart
Breitwiesenstr. 20–22, 70565 Stuttgart
Germany
{diepstjm,weiskopf,ertl}@informatik.uni-stuttgart.de

ABSTRACT

In this paper, a system is proposed which can automatically generate 2+1D Minkowski diagrams. We show how these spacetime diagrams reveal more aspects of special relativity than traditional Minkowski diagrams and can therefore further enhance the understanding of special relativity. The interactive system is based on Java3D and can be used as a widely applicable learning and teaching tool. Moreover, we adapt and extend non-photorealistic rendering techniques to improve the perceptibility of the diagrams.

Keywords: non-photorealistic rendering, visualization, Java3D, special relativity

1 INTRODUCTION

The special theory of relativity by Albert Einstein, which was first published in 1905 [Einst05], is still regarded by many as difficult and hardly comprehensible. The reasons are not its basic rules but the power of imagination which is necessary to fully understand the consequences behind the theory. The relationship of space and time is very different to what we see and feel every day. The 3D Euclidean geometry we are used to is replaced by a flat 4D spacetime in special relativity. Scientist were already aware of this problem a few years after Einstein had published his work and introduced spacetime diagrams to convey relativistic properties. These so-called Minkowski diagrams are still widely used today in science books but are regarded to be hardly understandable without prior knowledge because of their high level of abstraction. Therefore, other direct approaches were developed in the past to visualize special relativity by means of computer graphics but spacetime diagrams have so far been mostly ignored. Even if they are more abstract they are still better suited to visualize some of the aspects of special relativity which are hard or even impossible to see with direct approaches.

In this work we would like to revive the traditional Minkowski diagrams and extend them to an enhanced, computer-based visualization tool. Spacetime diagrams in textbooks can only show a static image of a fixed scenario and are very often limited to a 2D reduction of 4D spacetime with only one spatial axis. Our interactive system, however, allows the user to vary parameters and to experience their consequences immediately; she or he can navigate freely through the diagram to see it from different points of view. We are not limited to one spatial axis and can therefore reveal effects of special relativity on extended objects. Moreover, a method is developed to show visibility properties in spacetime diagrams. Finally, both the relativistic view as seen by a fast moving observer and the more abstract Minkowski diagram of the same scenario can be displayed in two neighboring viewports. In this way, two completely different visualization approaches are combined to further enhance the understanding of the theory. The implementation is based on Java3D to ensure platform independency and web-based deployment. It can be used as interactive teaching tool or for illustrations in textbooks.

The paper is organized as follows. In the next section a brief overview of previous work is presented. Section 3 is focused on the spacetime diagrams. Here the necessary background information is summarized. Section 4 describes the generation of 2+1D Minkowski diagrams. In Section 5 we discuss enhanced visual representation to give our diagrams a

more textbook-like touch. In Section 6 we give an overview of our system. Finally Sections 7 and 8 contain results, a short conclusion, and give an outlook on possible future work.

2 PREVIOUS AND RELATED WORK

Around the end of the 80s, works by Hsiung and coworkers [Hsiun89; Hsiun90] introduced special relativistic visualization to the computer graphics community. Their T-buffer technique [Hsiun90] is a polygon rendering method, which is also the basis for our direct relativistic view. In recent work, other relativistic rendering techniques were developed, such as image-based relativistic rendering [Weisk00]. A comprehensive presentation of state-of-the-art relativistic visualization can be found in [Hanso01].

But long before computer machinery had been invented, ways to explore and visualize special relativity on a more abstract level were introduced by the mathematician Hermann Minkowski [Minko08]. He developed and first published spacetime diagrams—the Minkowski diagrams.

Another important aspect of this paper is the appropriate visual representation of complex Minkowski diagrams by means of non-photorealistic rendering techniques (NPR). Much research was conducted in the field of artistic and non-photorealistic rendering [Salis94; Curti97; Marko97]. Our techniques for displaying Minkowski diagrams are closely related to works in the field of automatic technical illustration by Gooch and co-workers [Gooch98; Gooch99]. Last but not least, one part of our system facilitates the WIM metaphor (Worlds in Miniature) introduced by Ware [Ware90]. WIM offers, in addition to a standard first-person perspective, a second dynamic viewport with an outside view onto the virtual environment.

3 THEORETICAL BACKGROUND

In classic Newtonian physics, the time coordinate in a reference frame is completely independent of spatial coordinates (x,y,z); the time transformation between two frames is simply $t = t'$. In relativistic physics the situation is different: space and time are tightly connected to each other. When changing from one frame of reference to another, the time coordinate t depends not only on the temporal but also on the spatial coordinates of the other frame. When the observer is moving with a velocity v along the x axis, the t and x coordinates are related by

$$t' = \frac{t - vx/c^2}{\sqrt{1 - v^2/c^2}} \quad , \tag{1}$$

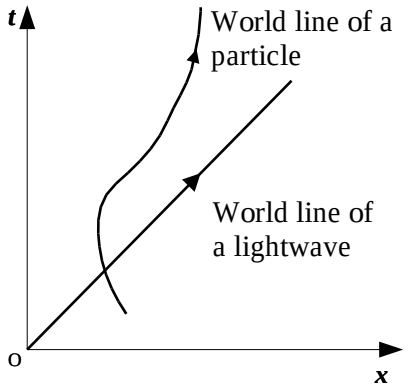

Figure 1: Minkowski diagram showing the world lines of a photon and a particle moving at varying speed.

$$x' = \frac{x - vt/c^2}{\sqrt{1 - v^2/c^2}} \quad , \tag{2}$$

where c is the speed of light in vacuum. Here, the y and z coordinates remain unchanged. The above transformation is called Lorentz transformation. Instead of treating space and time independently from each other, Minkowski connected them tightly in the form of a 4D spacetime and introduced spacetime diagrams as a new kind of visualization tool. Figure 1 shows an example of such a spacetime diagram.

In the traditional form, a Minkowski diagramm is 2D—with one spatial axis and the time axis—i.e., a 1+1D diagram. As special relativity operates in a 4D spacetime, where the time axis is treated differently than the spatial axes, standard Minkowski diagrams employ a reduction of spatial dimensions that, however, does not invalidate their usability. Due to the isotropy of special relativity, many qualitative features of spacetime are still present in 1+1D.

The convention for Minkowski diagrams is that the t axis lies vertically and the x axis lies horizontally for a rest frame. In a 1+1D diagram, an event is determined by its spatial coordinate x and its time t, resulting in a simple point in the diagram. As the diagram formed by the two axes is called "world" by Minkowski, the motion of a point-like particle through space and time is represented by a so-called word line. Note that a particle does not necessarily have to move in space but it automatically moves in time. Physical laws between the interaction of particles can be described by the geometric connection between their world lines. Each point of a world line forms a tangent with the slope dx/dt and an angle with the time axis which is smaller than 45°. This angle θ is given by

$$\tan\theta = \frac{dx}{dt} = \frac{v}{c} \quad .$$

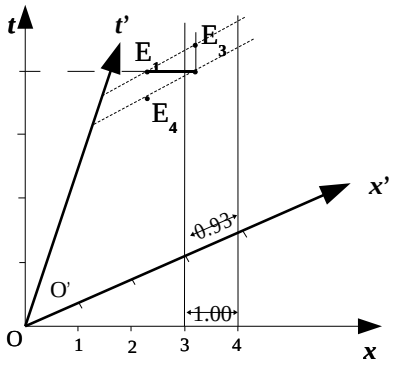

Figure 2: Minkowski diagram with two different inertial frames showing both the effect of concurrence and the contraction of length.

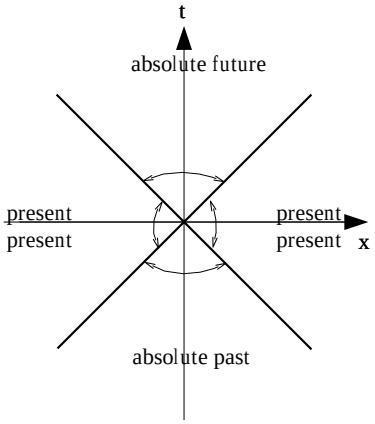

Figure 3: Minkowski diagram showing a light cone dividing the diagram into different causal regions.

For any massive particle, v must be smaller than c. Consequently, the t' and x' axes of a moving frame of reference are rotated and form the angle θ to the axes t and x of the rest frame. Photons—light particles—travel at the speed of light with respect to any frame. Therefore, their word lines always form an angle of 45° to the time axis.

The light cone of a Minkowski diagram consists of all the light rays that are emitted at a single emission event or absorbed at a single absorption event. In a standard 1+1D Minkowski diagram, the "light cone" consist of only two lines. Two different types of light cones can be distinguished—the future light cone and the past light cone. Together they form a double cone and divide the spacetime diagram into different causal regions (see Figure 3). A future light cone of an event is a set of light rays which pass through this event and are emitted into spacetime in all directions, similarly to a circular water wave formed when a stone is thrown into the water, only that the waves are extracted in time.

Spacetime diagrams in their 1+1D form are already well suited to visualize a major variety of effects experienced in special relativity. The most important effects are the following.

First, temporal concurrence can be visualized. For an observer O', two events are concurrent if they have the same time coordinate t'. This means if two events are parallel to the x' axis they are concurrent for O'. In Figure 2, the points E_1 and E_3 are concurrent for O' but not for O.

Second, the contraction of length can be illustrated. Figure 2 shows two inertial frames S and S'. Now let us consider a yardstick represented by the world lines of its two end points. The length of the stick is defined as the spatial distance between these two world lines. In S the length is simply the distance between the intersections of the world lines with the x axis. Accordingly, the length with respect to the moving frame S' can be measured along the x' axis. Note that the units along the x and x' axes differ, yielding a length contraction for the moving stick. In addition, the effect of time dilation can be presented similarly to the contraction of lengths. The only difference is that lengths are not measured along the x and x' axes but along the t and t' axes.

Finally, causal dependency between events can be explored visually. As information cannot travel faster than the speed of light, an event can only have influence on other events if they lie inside its future light cone. Conversely, light emission events that are the source for image generation by the observer's camera lie on the past light cone. Figure 3 shows a diagram divided into several causal regions.

One important goal of our work is to take these 1+1D Minkowski diagrams and extend them with a second spatial coordinate. This results in 2+1D Minkowski diagrams with one temporal and two spatial coordinates, i.e., their spatial part resembles flatland diagrams. 2+1D diagrams are hard to draw by hand but can provide certain new information which cannot be seen in 1+1D:

- Extended objects and their spatial relationships can be shown—instead of just simple point particles. World lines become world tubes in 2+1D.
- The visibility properties between objects can be visualized.

- Objects can move in various directions of motion; in the 1+1D case the direction of motion is fixed along one axis.
- Angles become apparant in two spatial dimensions.
- The relativistic aberration[1] of light can only be seen in more than one spatial dimension.

Therefore, the extension of 1+1D to 2+1D Minkowski diagrams yields a significant increase in information. Conversely, the step to a complete 3+1D representation lacks a comparable qualitative enhancement: The full 3D spatial domain can be sampled by rotating a 2D spatial plane around the direction of motion of the observer; due to the cylindrical symmetry of the Lorentz transformation with respect to this axis of rotation, any orientation of the plane gives the same qualitative picture.

The basic idea is to generate 2+1D Minkowski diagrams automatically by taking any normal 3D scene and reduce 3D object coordinates to 2D object coordinates by intersecting with a plane. This plane can be placed and modified by the user. Finally, the intersecting lines between the objects and the plane are extruded to world tubes. One point of the plane is defined as reference point of the resting frame S. The observer can be placed onto the plane by the user and is used as the reference point of the moving frame S'.

4 GENERATION OF MINKOWSKI DIAGRAMS

The automatic generation of 2+1D Minkowski diagrams consists of the following steps: intersection of 3D scene objects with a plane, extrusion to world tubes, Lorentz transformation of these world tubes, clipping, and creation of the light cone.

In the first step, all the intersection lines between the intersection plane and the objects in the scene are computed. The orientation and position of the plane is represented by a homogeneous transformation matrix M_{plane}. Scene objects are transformed into the coordinate system of the plane by means of the inverted matrix M_{plane}^{-1}. The intersection computation is performed with respect to the plane's coordinate system, i.e., the resulting lines always have a coordinate $z = 0$. Objects are assumed to be represented by triangle meshes. Therefore, the intersection process yields only polygonal lines. A normal vector and material properties—such as object color—are attached to the polylines. The normal vector is obtained from the normal vectors of the source triangle by linear interpolation. Computational costs for this intersection calculation could by reduced by standard acceleration methods such as bounding boxes, octrees, or BSP trees. However, speed measurements show that a brute-force approach is fast enough for typical applications, cf. Section 7.

In the second step, world tubes are extruded from the previously obtained intersection lines. First, the world tubes are generated for the resting frame S. Here, the tubes are just extruded along the t axis. (In the 2+1D Minkowski diagram, the t axis is identified with the z axis.) Theoretically, world tubes are infinitely long. On a computer system, however, they are represented by polygonal meshes, which have finite length. Therefore, we choose corresponding vertex coordinates at $z = 0$ for the lower end of the tube and $z = z_{\text{max}}$ for the upper end, where z_{max} is based on the bounds of the Minkowski diagram. As the observer may move at arbitrary speed, the world tubes have to be transformed into the moving inertial frame S' of the observer. This process is based on the Lorentz transformation of the world tubes' vertices from S to S'. The complete, inhomogeneous Lorentz transformation consists of a so-called Lorentz boost, rotations, and a translation. The Lorentz boost is caused by the different velocities of S and S'. Equations (1) and (2) reflect a Lorentz boost along the x axis; other directions of motions can be implemented by applying rotations before and after the boost. By combining the temporal and spatial coordinates to a 4D vector, general rotations and Lorentz boosts can be represented by a 4×4 matrix. Therefore, several subsequent rotations and boosts can be subsumed in a single matrix by matrix multiplications. The Lorentz transformation changes the world tubes' vertices and might shift them out of the bounding box of the Minkowski diagram. This could cause overlaps or gaps in the diagram. To avoid these overlaps and gaps, the world tubes are either extended to the upper or lower t clipping plane or cut by the upper or lower t clipping plane.

Finally, the past light cone has to be added to the diagram. In contrast to objects of the 3D scene, the observer is described by a full 4D position vector. This means that the light cone does not necessarily have to start at $t = 0$. This past light cone shows which world tubes lie in the observer's view. In any reference frame, the apex angle of the cone always is 45° due to the constancy of the velocity of light, i.e., the radius of the cone is the same as the height of the cone. The height of the cone is determined by the position of the observer in time and the lower t clipping plane.

[1] For a moving observer, the direction of light rays is changed by this relativistic effect. A detailed description of the aberration of light can be found, e.g., in [Rindl91].

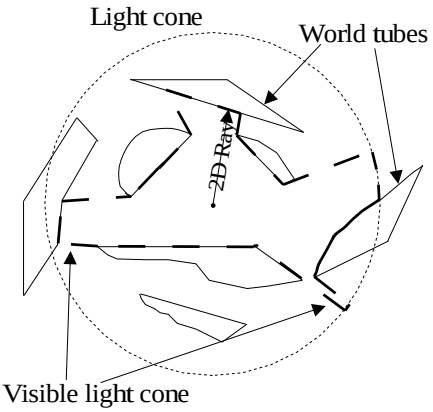

Figure 4: A light cone viewed from above.

Our implementation of the light cone includes two extensions which are usually not considered. The first extension allows to visualize visibility properties of surfaces by intersecting the light cone with the world tubes, i.e., to mark the parts of the objects that are visible for the observer. This could be done by directly calculating intersections between the cone and the triangles of the world tubes. However, by projecting the cone and the world tubes to the *xy* plane, this intersection calculation can be reduced to a 2D problem: recall that the cap of the cone always lies in the *xy* plane of the diagram. Now let us consider a camera located at the top of the *t* clipping plane and looking down the *t* axis. From that point of view, the cone has a circular outline and the world tubes are represented by lines lying inside and outside of that circle (see Figure 4).

Accordingly, these lines need to be intersected with the light rays from the observer to determine the visibility properties. Afterwards, the intersection points are connected to form a new outline which confines the region that is visible from the observer's point of view. The intersection with the lines is calculated by shooting 2D rays from the midpoint of the circle into all directions. Tests show that a sampling rate of $\pi/30$ is enough to obtain satisfactory results. So far, the outline of the visible light cone is known with respect to the rest frame S. Analogously to the Lorentz transformation of the world tubes, this outline has to be transformed to the moving frame S'. The Lorentz transformation matrix for the world tubes is applied to the cone. Here, the time coordinate is set to $t = t_{\text{obs}} - t_{\text{height}}$, where t_{obs} is the time coordinate of the observer and t_{height} is the height of the light cone, i.e., the distance between the cap and the apex. Unfortunately, the Lorentz transformation of cone vertices is equivalent to a transformation of the corresponding light rays. The angles between these rays are changed according to the relativistic aberration of light. Therefore, an originally isotropic sampling of the light cone with respect to frame S yields an anisotropic sampling with respect to the destination frame S'. This effect can be counteracted by considering the aberration of light during the sampling process. Therefore, the actual sampling directions α are given by

$$\alpha = \arccos\left(\frac{v}{1 + v\cos\alpha'}\right) \quad ,$$

which is based on the inversion of the aberration formula,

$$\cos\alpha' = \frac{\cos\alpha - v}{1 - v\cos\alpha} \quad . \tag{3}$$

Here, α' describes the isotropic sampling with respect to S' and v is the current velocity of the observer.

The second extension allows to visualize the aberration of light as a relativistic effect. Here, a second light cone in the form of a wireframe model is drawn on top of the first, standard cone. The visible line structure of the second cone represents the direction of incoming light rays, which is calculated by using Eq. (3). For an observer at rest, these light rays are equally distributed, whereas a moving observers measures a higher "density" of light rays in the direction of motion. Figure 7 (left image) illustrates this kind of visualization of the aberration of light.

5 ENHANCED VISUAL REPRESENTATION

In a typical 2+1D Minkowski diagram, a large number of objects represent the world tubes and the light cone. The depth, orientation, important features, shape, and structure of these objects should be recognizable with ease to fully grasp the geometric structure of a Minkowski diagram and the relationship between its constituents. This goal can be achieved by applying appropriate NPR techniques. In addition, a more traditional textbook-feeling can be attained for illustrating Minkowski diagrams.

In this paper, we focus on work related to technical illustrations, like [Gooch98; Gooch99]. Most of these techniques can operate at interactive frame rates and can be integrated into the normal hardware-based rendering pipeline. Technical illustration are based on fairly algorithmic principles. Although a wide variety of styles and techniques are found in technical illustration, there are some common themes.

- Edge lines are drawn as black curves.
- Matte objects are shaded with colors far away from gray; the warmth or coolness of these colors indicates the direction of the surface normal.

- A single light source provides white highlights.

We restrict ourselves to shading and material property aspects. These can be split into two major categories: line shading and surface shading. For the line shading part, only silhouette and edge lines are considered. A number of techniques were proposed to automatically find silhouettes—both as hardware and software methods [Zhang97; Marko97]. We use a technique described in [Gooch99], which takes advantage of OpenGL's `PolygonOffset` function. This is the only method known to us which can be implemented directly in Java3D and takes advantage of hardware acceleration. For the line weight many conventions exist which the illustrator chooses among, based on the intention of the image. The three most common ones are: a uniform weight used throughout the images, two line weights with heavy lines for outer edges and light lines for interior lines, and varying the line weight along a single line emphasizing the perspective of the drawing by heavy lines in the foreground. An improved depth information can be attained by choosing lines with varying weights. To accomplish different line widths even within a single object—in Java3D a shape object node can only have one appearance which is valid on the complete object—the complete scene is divided into different line width regions.

Surface shading in technical illustrations displays subtle shape attributes. In most technical illustrations, hue changes are used to indicate surface orientation rather than reflectance because shape information is considered more important than precise reflectance information. A hue shift of the shading model reduces the dynamic range for shading in order to ensure that highlights and edge lines remain distinct. This follows Tufte's strategy [Tufte97] of the smallest effective difference: "Make all visuals distinctions as subtle as possible, but still clear and effective". Therefore, Gooch et al. [Gooch98] propose a tone-based cool/warm shading method. We approximate this model by using two directional light sources with direction vectors $\vec{l}$ and -$\vec{l}$. The respective colors are $(k_{\text{warm}} - k_{\text{cool}})/2$ and $(k_{\text{cool}} - k_{\text{warm}})/2$, and the color of the ambient lighting is $(k_{\text{cool}} + k_{\text{warm}})$, where k_{cool} specifies a cool tone (mostly bluish and greenish colors) and k_{warm} a warm tone (mostly reddish, yellowish, and orange). In a second rendering pass, white specular highlights are added.

We extend Gooch et al.'s model by two methods for distance color blending in order to enhance the perception of depth. The first one, which is also used by Ebert et al. [Ebert00], is a simple intensity depth cuing by applying a specific color as cuing color. Artists often use a bluish or another suitable background tone for cuing. This effect can be approximated in a hardware-based rendering pipeline by adding a linear fog to the scene. We propose another method that changes the saturation instead of the intensity of distant objects. In our opinion this results in a better subjective impression of the importance of closer objects. Since this approach does not change the brightness of objects, structures in the background are still recognizable—whereas the first approach blurs these structures. The saturation of a fragment can be described by:

$$s_{\text{d}} = s_{\text{s}} * \left(\frac{z_{\text{fragment}} - z_{\text{min}}}{z_{\text{max}} - z_{\text{min}}} \right)^{k_{\text{s}}} \quad ,$$

where s_{d} is the new saturation, s_{s} the original saturation, and z_{fragment} the z value of this fragment. The maximum z value is z_{max}, the minimum z value is z_{min}, and k_{s} is an additional saturation parameter. However, this method is more difficult to implement because both the frame buffer and the z buffer have to be read in order to adapt the color values. In Java3D, for example, this can only be achieved in intermediate mode rendering [SUN M00] and has some certain drawbacks, such as giving up double buffering.

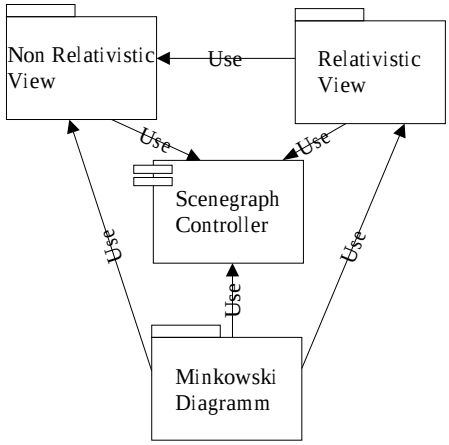

Figure 5: System overview of *MinkRelVis*.

6 SYSTEM OVERVIEW

Our program *MinkRelVis* (Minkowski Relativistic Visualization) makes use of the Java3D scene graph API from SUN and the graphical user interface is based on Java2 Swing/JFC classes. The main reason for using Java3D is its platform independency. Java3D is currently available for Microsoft Windows, Linux, Solaris, HP-Unix, SGI's IRIX, and IBM's AIX. Other reasons are a rapid GUI production in Java, a great variety of freely available 3D file format loaders [Couch01], the possibility to integrate the system into a web environment, and the ongoing strong support for this API. Right now, the future of Java3D

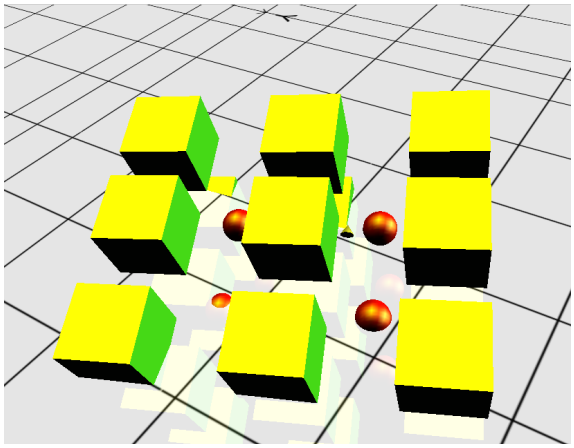

Figure 6: Non-relativistic test scene. The yellow cone represents the observer.

looks bright and many new interesting features will be added in future versions [SUN M01].

MinkRelVis consists of four main parts (see Figure 5):

- The `SceneGraphControler` which handles and stores all scene graph structures of the whole system.
- The `Non-relativistic View` which displays the scene in it's normal "non-relativistic view". Here, the user accomplishes his main tasks like placing the observer and the intersection plane.
- The `Relativistic View` showing a current shot of the scene from the observer's point of view if he would be flying through the scene. The "relativistic view" is based on relativistic polygon rendering.
- The `Minkowski Diagram` module itself.

`Non-relativistic View`, `Relativistic View`, and the `Minkowski Diagram` have all their own render-canvases and scene graphs which are managed by the `SceneGraphControler`.

7 RESULTS

Figure 7 shows two typical snapshots taken from *MinkRelVis*. Both images show a Minkowski diagram generated by using a test scene which can be seen in Figure 6. In Figure 7, the left image uses normal rendering and the right image uses NPR. In both images the observer travels with a velocity of $0.8c$. The contraction of length and time dilatation become apparent. Moreover, the aberration of light is visible in the left image as the points of the second light cone shift into the observer's direction of motion. Visibility tests are also activated; the light cone ends at the intersection points with the world tubes.

Table 1: Performance measurements.

Task	Duration	
	in ms	in percent
intersection calculation	491	20
scene graph traversal	180	7
extrusion of intersection lines	1723	71
others/communication	21	2
total	2415	100

To verify that the system can operate in nearly interactive rates for typical applications, a test with a more complex scene showing a complete Lancia engine block containing 85 objects, 174283 triangles, 67746 normals, and 90108 vertices was conducted. Table 1 shows the results for the different tasks. The values were taken on an Athlon 1200 MHz PC with 512 MBytes RAM, Windows 2000 Service Pack2, Java implementation 1.3.1 from SUN, and Java3D 1.2.1_03 (OpenGL). The values show that the largest amount of time is used for extruding lines into world tubes and not the intersection calculation, which only takes about a quarter of the total time. Overall we are very satisfied with the performance and stability of Java3D, although there were reports on problems concerning dynamic scenes in the past [Kling01]. We also encountered some problems/bugs and inflexibility owing to the intermediate-mode rendering of Java3D. We hope that these issues will be addressed in future implementations. The only real major problem we experienced with Java3D is its memory usage. Even simple scenes already take up several megabytes of physical memory and therefore the Java virtual machine runs out of its standard heap size of 64Mbyte quite fast.

8 CONCLUSION AND FUTURE WORK

In this paper a system for automatically generating 2+1D Minkowski diagrams has been presented. We have proposed some additional properties which do not exist in traditional spacetime diagrams. At the same time our system combines and demonstrates the difference between abstract visualization and direct rendering approaches. Furthermore, NPR techniques have been adapted and extended to improve the perceptibility of the diagrams and allow a textbook-like feeling. We have shown that the system can be implemented successfully in Java3D today and that

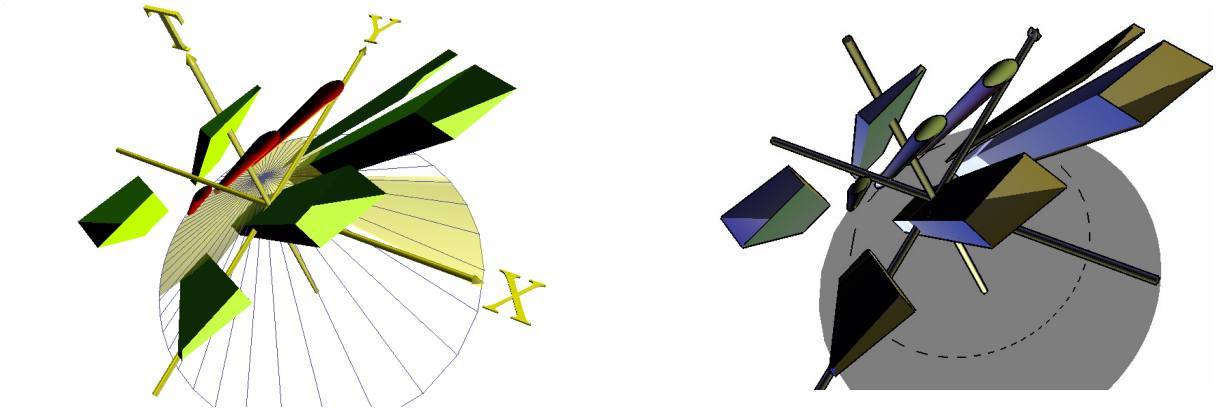

Figure 7: 2+1D Minkowski diagram generated from the test scene in Figure 6, the left image in normal rendering style and the right image in NPR rendering style.

the speed and reliability of Java3D are at a productive level. *MinkRelVis* can be used as an interactive learning tool by students to visually explore the complex nature of special relativity. Another application is the generation of meaningful illustrations of Minkowski diagrams, e.g., as still images in textbooks. *MinkRelVis* is freely available from our web page [Dieps01].

Future work could cover several aspects. First the system could be transferred into a VR environment, the necessary routines are already provided in the Java3D API. Second, not only changes of the observer's velocity but dynamic fly-throughs with changing speed and direction could be added and visualized—both in the Minkowski diagram and in the relativistic viewport. It might be interesting to have more than one observer in the system, so that the user can easily switch from one frame to another. Multiple light cones could also be added to visualize the causal influence between observers. Finally, markers could be added to improve the visibility of the contraction of lengths and time dilatation.

REFERENCES

[Couch01] J. Couch. J3d.org file loader archives. Web Site: http://www.j3d.org/utilities/loaders.html, 2001.

[Curti97] C. Curtis, S. Anderson, J. Seims, and K. Fleischner. Computer-generated watercolor. In *SIGGRAPH 1997 Conference Proccedings*, pages 421–430, August 1997.

[Dieps01] J. Diepstraten and D. Weiskopf. MinkRelVis. Web Site: http://wwwvis.informatik.uni-stuttgart.de/relativity, 2001.

[Ebert00] D. Ebert and P. Rheingans. Volume illustration: non-photorealistic rendering of volume models. In *IEEE Visualization 2000 Proceedings*, pages 195–202, October 2000.

[Einst05] A. Einstein. Zur Elektrodynamik bewegter Körper. *Annalen der Physik*, 17:891–921, 1905. In German.

[Gooch98] A. Gooch, B. Gooch, P. Shirley, and E. Cohen. A non-photorealistic lighting model for automatic technical illustration. In *SIGGRAPH 1998 Proceedings*, pages 101–108, July 1998.

[Gooch99] B. Gooch, P.-P. Sloan, A. Gooch, P. Shirley, and R. Riesenfeld. Interactive technical illustration. In *1999 ACM Symposium on Interactive 3D Graphics*, pages 31–38, April 1999.

[Hanso01] A. J. Hanson and D. Weiskopf. SIGGRAPH 2001 Course 15: Visualizing Relativity, 2001.

[Hsiun89] P.-K. Hsiung and R. H. P. Dunn. Visualizing relativistic effects in spacetime. In *Supercomputing '89 Proceedings*, pages 597–606, 1989.

[Hsiun90] P.-K. Hsiung, R. H. Thibadeau, and M. Wu. T-buffer: Fast visualization of relavistic effects in spacetime. *Computer Graphics*, 24(2):83–88, 1990.

[Kling01] F. Klingener. Simulation clock on the Java3D API. Web Site: http://www.brockeng.com/VMech/Time/Clocks.htm, 2001.

[Marko97] L. Markosian, M. Kowalski, S. Trychin, and J. Hughes. Real-time non-photorealistic rendering. In *SIGGRAPH 1997 Proccedings*, pages 415–420, August 1997.

[Minko08] H. Minkowski. Die Grundgleichung für die elektromagnetischen Vorgänge in bewegten Körpern. *Nachrichten von der Königlichen Gesellschaft der Wissenschaften und der Georg-Augusts-Universität zu Göttingen*, pages 53–111, 1908. In German.

[Rindl91] W. Rindler. *Introduction to Special Relativity.* Clarendon Press, Oxford, second edition, 1991.

[Salis94] M. Salisbury, S. Anderson, R. Barzel, and D. Salesin. Interactive pen and ink illustration. In *SIGGRAPH 1994 Proccedings*, pages 101–108, August 1994.

[SUN M00] SUN Microsystems. The Java3D API specification. Web Site: http://java.sun.com/products/java-media/3D/forDevelopers/J3D_1_2_API/j3dguide/index.html, 2000.

[SUN M01] SUN Microsystems. Proposed Java3D 1.3 API changes. Web Site: http://java.sun.com/products/java-media/3D/1.3.html, 2001.

[Tufte97] E. Tufte. *Visual Explanations.* Graphic Press, 1997.

[Ware90] C. Ware and S. Osborne. Exploration and virtual camera control in virtual three dimensional environments. *Computer Graphics*, 24(2):175–183, 1990.

[Weisk00] D. Weiskopf, D. Kobras, and H.Ruder. Real-world relativity: Image-based special relativistic visualization. In *IEEE Visualization 2000 Proceedings*, pages 445–448, October 2000.

[Zhang97] H. Zhang and K. H. III. Fast backface culling using normal masks. In *Proceedings Symposium on Interactive 3D Graphics 1997*, pages 103–106, April 1997.