Visualization of relativistic phenomena

Zoltán Simon

Abstract

Special relativity describes many phenomena, not observable in everyday life. This article presents an educational application, which helps to understand these effects. It models vision of objects travelling close to the speed of light. e. g. it visualizes length contraction, time dilatation, relativistic Doppler effect. It considers the path of light between the object and our eye, but it also presents an opportunity to visualize events happening simultaneously as well. It allows to switch between Lorentz and Galilean transformation, so we can compare Einstein's and Newton's model. It serves with a three-dimensional space-time diagram, on which we have a chance to further analyse the movement of objects.

1. Introduction

Theory of special relativity [1] is a key part of modern physics. There are many use cases of this theory. Despite of it's importance there is very little chance for people to experience relativistic phenomena in their everyday life, thus differences between Isaac Newton's classical and Albert Einstein's modern model remain unexplored. The application described in this article offers a way to visualize well-known phenomena associated with special relativity. I took inspiration from the work of Ute Kraus and Corvin Zahn [2]. On their website *Space Time Travel* they present many computer generated images and short animations about relativistic effects. The goal of my work was the implementation of a software that can render images similar to the ones on *Space Time Travel*, enables inspection of world lines using space-time diagrams and allows the user to freely change the parameters of the simulation and see the changes take effect. In this article I showcase the algorithmic challenges surrounding this program and introduce solutions for these problems. To present it's capabilities, I included images rendered by this simulator.

2. Theoretical background

The rules of simulation are based on Einstein's postulates [1]. These suggest that

- an observer can not differentiate between different inertial frames of reference by taking measurements in each frame of reference.
- the velocity of propagation of any physical effect can not

exceed the speed of light in vacuum: $c = 299792458 \frac{m}{s} \sim 3 \cdot 10^8 \frac{m}{s}$.

These two constraints do not provide sufficient tools for the programmer on their own. However, all the required formulas can be derived from these two postulates.

2.1. Usage of the term 'absolute frame'

To simplify representation of world lines every event and velocity is described in the same frame of reference. In code this frame of reference is called 'absolute frame'. This helps circumvent confusion when transforming between different observer's proper frames. Each value is stored as it is measured in 'absolute frame'. When visualizing a given value, then it gets transformed to a particular observers proper frame.

2.2. Representation of physical bodies

To model the journey of physical objects through space-time, it is convenient to assign a world line to each object. To be more precise, this is going to be the world line of the object's origin point. For practical purposes this point is selected to be the same as the origin point in model space. Such world line can be stored as a set of events. We assume geodetic world line between two neighbouring events. At this point it is necessary to note that we also assume flat space-time. This method allows us to represent world lines consisting of few geodetic sections easily. World lines of accelerating objects appear curved in space-time diagrams. We can approximate these lines by a set of short geodetic sections. This approximation is useful, because along the small sections

we can use the simple formulas of Lorentz transformation. The downside of this method is that many calculations require iteration over the small sections. E. g. determining the age of an object without a priory information requires summing of ageing of the object along the geodetic sections. The other obvious downside of such approximation is that the resulting world line is going to have relatively low resolution. This is particularly noticeable, when we –as a viewer– place ourselves in the proper frame of such accelerating object. In this case the violent jumping nature of the movement of surroundings is much more noticeable.

Length values are stored in metres. To get space-time diagrams, where the angle between world line of light and coordinate axes is $45°$, we also measure time in metres. This way the velocity is dimensionless quantity.

3. View modes

The application supports multiple view modes. The two main modes are the *real time* three dimensional view and the *diagram view*. In the *real time* mode the scene is shown as regular three dimensional scene, changing over time. We can look around with the virtual camera to observe the moving objects. The shape and colour of objects gets distorted in accordance with the relativistic effects. The simulation speed is slowed down greatly compared to the real physical world, to make comprehension of fast movements possible.

In the *diagram view* we can observe the world lines of objects present in the scene. This provides additional insight into the behaviour of objects. The user can switch to orthographic projection from the standard perspective projection. This improves the readability of the space-time diagram, and it's also useful in the *real time* view.

In order to help the user spot differences between classic and relativistic physics, there is a way to switch between Galilean [4] and Lorentz transformation. A viewer can see the incoming light from an object, thus we can only see images showing the past. This is due to the finiteness of the speed of light. The application allows to toggle between visualization of present event as described in the selected observer's frame, and past events in accordance with the incoming light beams.

3.1. Switching between observers

To compare the observations in different inertial frames of reference, we can toggle between multiple observers. In this implementation observers are independent from physical objects, however it's possible to make an observer follow an object. Every observer has it's own world line. These lines can also be visualized in *diagram view*.

When switching between frames of reference the origin of the frame also changes. The $\vec{O} = (0,0,0)$ in currently selected proper frame gets assigned to the space coordinate

vector of an event along the world line of the observer. The question is, how do we select the time component. In other words, which event will be selected along the new world line. This implementation chooses the event from the new observers world line, which happened simultaneously with the event of changing observers according to the previous observer. This method introduces a somewhat more realistic approach, however in real life still no object can travel from one event's location to an other's if these two events appear to be simultaneous. This would require velocity greater than the speed of light. In defence of this approach in this computer simulation neither the user's perspective nor an observer is necessary bound to a physical object. At least this way the false sense, that the 'absolute frame' in the program would have special characteristics in real life, is weakened. There is a downside to this approach. When we change the frame of reference once, and then change back to the initial frame of reference, we potentially arrive to a different event.

3.2. Visualization of simultaneous events

First let's take a look at the less realistic approach of visualization. We render the simultaneously happening events. Due to relativity of simultaneity it is necessary to determine simultaneity in the currently selected observers proper frame. Luckily present events form a hyperplane in the four dimensional space-time. Present events along object's world lines can be obtained by calculating intersections between the world lines and this simultaneous hyperplane. In this simulation the above-stated hyperplane is described by coordinates assigned in the 'absolute frame'. A hyperplane can be defined by an event on the hyperplane and a normal four-vector. The event is going to be the one along the world line associated with the observer, of which time component is t_{abs} time variable of the simulation. The normal vector is obtained from the $\vec{v} = (v_x, v_y, v_z, v_t)$ velocity of the observer's frame relative to the 'absolute frame'. This is described in form 1.

$$\vec{n}_{pl} = (-v_x, -v_y, -v_z, v_t) \quad (1)$$

Equation 2 for calculating the intersection between the line and plane is analogous to the one used in three dimensional space, where $\vec{n}_{pl}$ is the normal of the hyperplane, $\vec{p}_{pl}$ is the event on the plane, $\vec{s}_{wl}$ is a starting event of the geodetic world line and $\vec{v}_{wl}$ is the velocity four-vector of the this world line. Note that the resulting t parameter is not the time measured by the object travelling along the world line nor the time measured in the 'absolute frame'. This is simply a ray parameter, although can be easily transformed to the time measured in 'absolute frame' by multiplying by the time component of $\vec{v}_{wl}$ four-velocity.

$$t = \frac{\vec{n}_{pl} \cdot (\vec{p}_{pl} - \vec{s}_{wl})}{\vec{v}_{wl} \cdot \vec{n}_{pl}} \quad (2)$$

The previously described equation can handle geodetic

world lines. To use this with segmented world lines we need to iterate over the short geodetic segments.

3.3. Visualization of events considering the propagation of light

A more realistic approach for finding events to render uses the event horizon as the intersecting hypersurface. From greater distances older events are seen. This characteristic makes the event horizon analogous with a cone in three dimensional space. The event at the viewers position is enough to describe this hypercone. The 'slope' of this hypercone is always the same, because the speed of light is constant. The quadratic equation 7 provides the tool for calculating the intersection.

$$a = v_x^2 + v_y^2 + v_z^2 - v_t^2 \tag{3}$$

$$\vec{m} = (s_x, s_y, s_z) \cdot (v_x, v_y, v_z) - (v_x, v_y, v_z) \cdot (c_x, c_y, c_z) \tag{4}$$

$$b = 2((m_x + m_y + m_z) - (s_t \cdot v_t - v_t \cdot c_t)) \tag{5}$$

$$c = ((c_x, c_y, c_z) - (s_x, s_y, s_z)) \cdot ((c_x, c_y, c_z) - (s_x, s_y, s_z)) \tag{6}$$

$$0 = ax^2 + bx + c \tag{7}$$

Because no world line can exceed the speed of light, the roots of the equation are always real values. We select the root associated with the past event.

When using Galilean transformation, the hypercone could have different shapes in different frames of reference. This is because the speed of light is not measured to be equal in all frames of reference. This implementation tackles this problem by considering the 'absolute frame' the one, where speed of light equals to c in any random direction. This is convenient from computational standpoint, because the program works with values represented in this frame of reference anyway.

4. Rendering pipeline in real time mode

Let's take a closer look at the path taken by the data describing an object until it appears on screen. We focus on the *real time* mode, because in *diagram view* the world line of an object explicitly gets converted to a line in three dimensional space by removing one space coordinate and using the time component instead. However in *real time* mode more calculation is required.

We need to calculate separate world line for each vertex of a geometry. Because of the spacial differences between different points of an object, they tend to behave significantly different. Consequently the shape of an object gets distorted. For this to behave somewhat realistically, high resolution meshes are required. Even a cube hast to be composited from many polygons. The program was tested with cube geometry consisting of 12300 polygons, 36900 vertices. This makes possible, that the edges of a cube appear curved when transformed.

4.1. Generating world lines for vertices with relativistic procedure

First step in generating the world line for a given vertex is determining the s starting event of the world line. This is done by taking the spacial coordinates of the vertex modified by the length contraction formula 8 and appending these with a 0 as the time component. This is interpreted as an event that is at the intersection of the simultaneous hyperplane of the 'absolute frame' at $t = 0m$ and this world line. The length contraction is applied in the direction of the velocity of the object relative to the 'absolute frame'.

$$l' = l\sqrt{1 - \frac{v^2}{c^2}} \tag{8}$$

By taking the initial $\vec{v}_0$ four-velocity of the object's origin's world line measured in the 'absolute frame' and our new $\vec{s}$ starting event, we get the first geodetic segment of the new world line. The next step is to calculate the ending event of this section. In this implementation we assume, that all vertices of the object get boosted simultaneously with the origin of the object. Simultaneity is determined in the proper frame of the object prior to the event of the boost. Thus, the ending event is the intersection between the mentioned simultaneous hyperplane and the geodetic world line segment. This method is able to recreate *Wigner rotation* as described in section 5. The ending event of the current segment will be the starting event of the following geodetic section. Please note, that this method introduces an accumulated error to the iteration caused by the computer's finite precision number representation. The velocity is always equal to the origin point's velocity in the corresponding segment. Before the first starting event and after the last ending event the algorithm assumes, that the first and last segment is a continuation and is continued by the same geodetic world line respectively.

During the previously described iteration intersection between the hyperplane or hypercone and the segments can be obtained as described in sections 3.2 and 3.3. As an optimization the iteration can be quit after this intersection is found.

The finishing steps of the rendering pipeline include transforming the found intersection event from the 'absolute frame' to the proper frame of the current observer using the *Lorentz transformation* and calculating the *Doppler shift* of colour. When calculating the *Doppler shift* the relative velocity between the selected observer and the object's proper frame also must be transformed with the *Lorentz transformation* for velocity.

4.2. Generating world lines for vertices with classic procedure

When using the *Galilean transformation*, generating world lines for the vertices is similar to the one described in sub-

section 4.1. The key differences are described in this subsection. When creating the *s* starting event for the world line, no length contraction is applied. The ending event of segments can be obtained by translating the events of the object's origin's world line by the spacial coordinates of the vertex. This way we also eliminate the accumulated error introduced by the method in subsection 4.1. This simplification is available, because in Newton's model simultaneity is absolute, thus events of boosting are simultaneous in every frame of reference.

After obtaining the intersection event between the hyperplane or hypercone and the world line of a vertex, *Galilean transformation* is applied, to obtain the coordinates of the event in the selected observers proper frame and the relative velocity between the object's and observer's proper frame.

5. Wigner rotation and Thomas precession

Let's consider a stationary object in frame *S*. Now let's boost the object to $\vec{v}_1$ velocity. We assume that, the effect of the boost is instantaneous, and all points of the object get boosted simultaneously in the proper frame of the object. Also no interaction takes place between points of the object. With the change of velocity the proper frame of the object also changes. Let's name the new proper frame of the object S'. In the frame S' we boost the object once again to $\vec{v}_2$ velocity. This boost is perpendicular to the first one. After the two boosts in frame *S* the distortion of the object is measured as if it had been boosted diagonally by (v_x, v_y) and rotated by angle α. This rotation is called *Wigner rotation*. The values can be calculated in the form of the following equations.

$$v_x = v_1 \tag{9}$$

$$v_y = \frac{v_2}{\gamma_1} \tag{10}$$

$$\sin \alpha = \frac{\gamma_1}{\gamma_2} \cdot \frac{v_1 v_2 (1 - \gamma_1 \gamma_2)}{\gamma_1^2 v_1^2 + v_2^2} \tag{11}$$

$$\cos \alpha = \frac{\gamma_1}{\gamma_2} \cdot \frac{\gamma_2 v_1^2 + \gamma_2 v_2^2}{\gamma_1^2 v_1^2 + v_2^2} \tag{12}$$

Now that we have introduced the concept of Wigner rotation, let's discuss Thomas precession. Let's say that we have a fast spinning particle. Let's move the particle along a circular path, so that only the center of mass of the particle is being accelerated. After the particle arrives back to it's starting position, the axis will point in different direction than at the start. This phenomena is called Thomas precession. It can be explained by the Wigner rotation. The particles velocity is tangential to the circular path. The centripetal acceleration boosting the particle is perpendicular to the velocity. Consequently a series of infinitesimal Wigner rotations are occurring. Equation 13 defines the angular velocity of the described precession, where *a* is the centripetal acceleration, *v* is the tangential velocity of the particle and γ is the Lorentz

factor calculated as $\gamma = \frac{1}{\sqrt{1 - \frac{v^2}{c^2}}}$.

$$\omega_T = \frac{\gamma^2}{\gamma + 1} \frac{av}{c^2} \tag{13}$$

As stated earlier, the Thomas precession is a series of Wigner rotations. This is convenient, because we only need to make sure that the boost of each individual world line assigned to the vertices of an object happen simultaneously. This is the part of the relativistic render pipeline described is subsection 4.1.

Although previously Thomas precession was described using the spin axis of a particle, extended and not spinning objects also suffer such precession, when travelling in circles. However it's harder to precisely describe the nature of the deformations suffered by such objects. Figure 1 shows the effect of Thomas precession on a circulating cross shaped object. The circular path has a radius of $r = 20m$ and the tangential velocity is $v = 0.6$. The The object travels counterclockwise. For comparison the object at the center of the image is stationary in the observer frame. The changes in length and the rotations add up to a complex deformation. Although it is complex, the amount of general Thomas precession is in accordance with equation 13. Given the radius *r* and the tangential velocity *v*, the time it takes the object to travel one full circle is described in equation 14.

$$t_{rot} = \frac{2\pi r}{v} \tag{14}$$

After substitution to equation 13 and 14 we get $\omega_T = \frac{1.25^2}{1.25 + 1} \cdot 0.018 \cdot 0.6 = 0.0075 \frac{1}{m}$ and $t_{rot} \doteq 209.44 m$. Consequently $\alpha_T \approx \frac{\pi}{2}$. This precession can be observed in figure 1. Figure 2 shows the same scene after two full rotations. The lines on the images plot the path of the center (white line) and the initially left and right (blue and red) endpoints of the red beam of the cross.
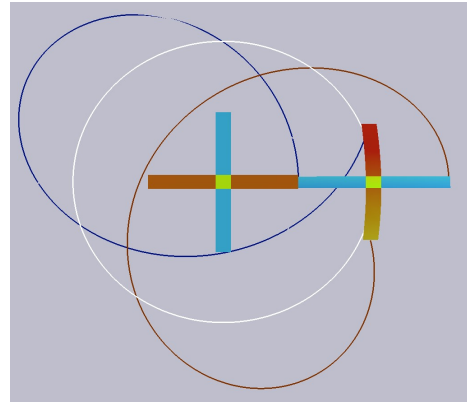

Figure 1: *Thomas Precession after one full rotation*

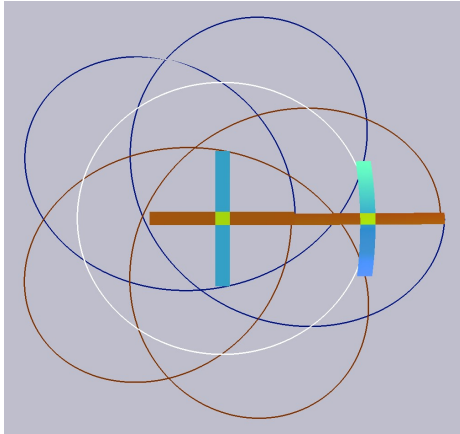In Figure 3 we can take a look at the world line of the

Figure 2: *Thomas Precession after two full rotations*

stationary and rotating object. The additional world lines of the end points of the object are also visible. The lines appear, to be wound up around the time axis.
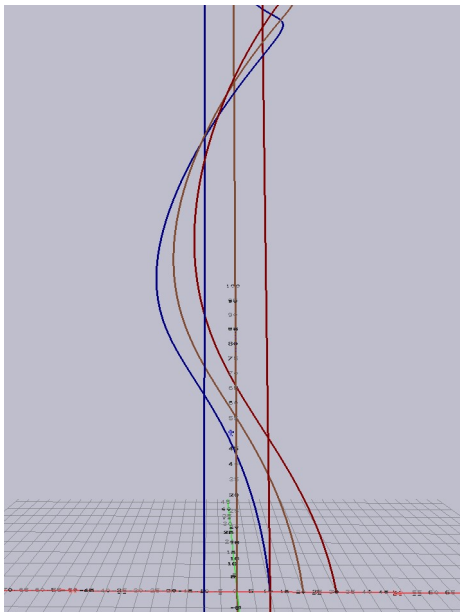

Figure 3: *Circular world line*

6. Results

This section showcases further images rendered by the application. These images focus on a selection of popular relativistic phenomena.

When an object passes by in front of the viewer, the object appears to be rotated. The rotation axis is perpendicular

to the movement direction and the viewer's direction from the object. The effect is the consequence of light beams travelling different distances from different points of the object. This can also be described using the classical model. The difference is that when we use the relativistic model, the length contraction cancels out the visual lengthening of the object. This way it creates the perfect illusion of rotation. Results of Lorentz and Galilean transformation on a moving dice are shown in Figure 4 and 5 respectively. Top dice is moving to the left in parallel with the viewing plane with velocity $v = 0.9$.
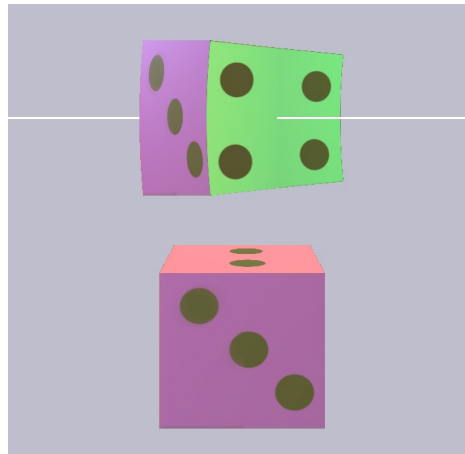

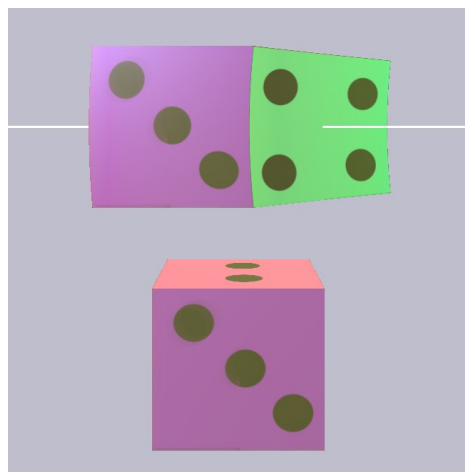**Figure 4:** *Rotated view of fast moving dice using Lorentz transformation*

Figure 5: *Rotated view of fast moving dice using Galilean transformation*

We can ignore the path travelled by photons towards the eye and render simultaneous events. This way the length

contraction is observable explicitly as shown in Figure 6. The dice on the top of the image moves to the left with velocity $v = 0.9$.
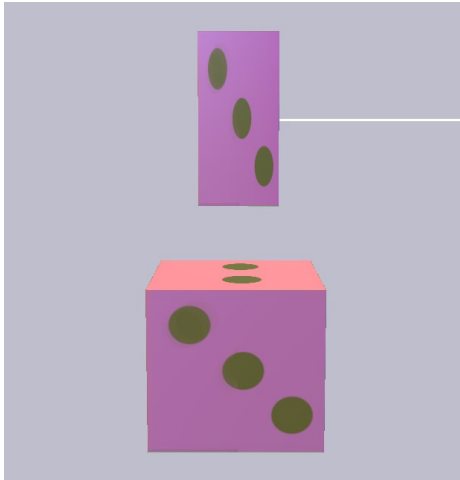

Figure 6: *Length contraction*

Relativistic Doppler effect is the shift of the electromagnetic spectrum of incoming light depending on the relative velocity of the light source compared to the viewer. Figure 7 and 8 show a row of globes first without any Doppler shift and then with realistic amount of shift.
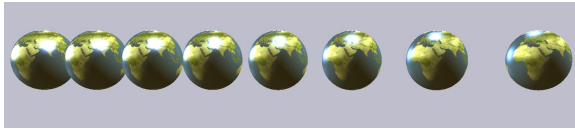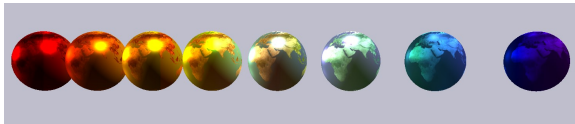

Figure 7: *Globes without Doppler effect*

Figure 8: *Globes with Doppler effect*

6.1. Performance

The application implementing algorithms described in this article had been tested with different amount of detailed objects. The results of performance tests are listed in table 1. The tests where performed on a computer running *Windows 11* operating system, with *AMD Radeon(TM) Vega 8 Graphics* and 8 *AMD Ryzen 5 3500U* core processor. The geometry of the object used for the tests consists of 36900 vertices.

Table 1: *Performance of the simulation*

Number of objects	FPS (short iter.)	FPS (long iter.)
1	198	76
2	127	43
3	99	31
4	85	25
5	71	20
6	60	17
7	54	15
8	48	13
9	45	12
10	41	11
20	23	5
30	16	4

Without any objects to display the programs frame rate was around 400 frames per second. Two different settings where used during testing. In both cases the time variable of the simulation was set to a fixed value. In each test both the relativistic and classical algorithms where performed simultaneously to consider the case, where the simulator must interpolate to provide visually pleasing transitions between the results of the two methods. The first series of tests used the fixed $t_{abs} = 0m$ time setting as the proper time of the observer. The world lines where geodetic. This way the least possible amount of world line sections must be checked to determine the intersection between the hypersurface and the world lines. In the second series of tests spiral world lines where used. This type of world line uses relatively large number of geodetic sections. Each spiral line described a circular path with $r = 20m$ radius and $v = 0.5$ tangential velocity. The time variable was set to $t_{abs} = 251m$. This way the displayed objects have travelled approximately one full circle around their center of rotation.

7. Conclusions

The methods described in this article are capable of visualizing the most common relativistic phenomena. The performance details show that the program is able to run with interactive frame rates while displaying detailed objects, although there is room for improvement when it comes to displaying objects with complex world lines. The ability to change the parameters of the simulation enables users to experiment freely with objects moving with velocities close to

the speed of light. The combination of *real time* and *diagram* view alongside the interactivity unlocks the ability to further deepen one's understanding of the theory of special relativity.

Acknowledgements

References

1. Albert Einstein. On the Electrodynamics of Moving Bodies. In *Annalen der Physik*, **17**:891, June 1905.

2. Kraus U., Zahn C. Relativity visualized *Space Time Travel*. url: *https://www.spacetimetravel.org/*, Last access 12:30 May 27. 2022.

3. Albert Einstein. *Relativity: The Special and General Theory*. Henry Holt and company, 1920.

4. Knudsen J.M., Hjorth P.G. The Galilei Transformation. In *Elements of Newtonian Mechanics*. Advanced Texts in Physics. Springer, Berlin, Heidelberg, 2000.

5. Buenker, Robert J. The Global Positioning System and the Lorentz Transformation. In *Apeiron: Studies in Infinite Nature*, 15.3, 2008.

6. Utkarsh Bajaj. Visual Appearance of Extended objects in Special Relativity. *arXiv*, eprint: 2104.03908, April 2021.

7. Sherin Z. W., Cheu R., Tan P. Visualizing relativity: The OpenRelativity project. In *American Journal of Physics*, **84**:369, doi: 10.1119/1.4938057, 2016.

8. Taylor E. F., Wheeler J. A. *Spacetime Physics*. New York, W. H. Freeman and Co., 1992. isbn: 0-7167-2327-1.