Joint Virtual Reality Conference of EuroVR - EGVE (2011)
S. Coquillart, A. Steed, and G. Welch (Editors)

Novative Rendering and Physics Engines to Apprehend Special Relativity

T. Doat [†1,3] , E. Parizot [‡2] and J-M. Vézien [§1]

[1]CNRS/LIMSI/VENISE group, Orsay, France
[2]APC – Paris 7 University, Paris, France
[3]Paris-Sud 11 University, Paris, France

Abstract

Relativity, as introduced by Einstein*, is regarded as one of the most important revolutions in the history of physics. Nevertheless, the observation of direct outcomes of this theory on mundane objects is impossible because they can only be witnessed when relative velocities close the speed of light are involved. These effects are so counterintuitive and contradicting with our daily understanding of space and time that physics students find it hard to learn Special Relativity beyond mathematical equations and to understand the deep implications of the theory.*

Although we cannot travel at the speed of light for real, Virtual Reality makes it possible to experiment the effects of relativity in a 3D immersive environment. Our project is a framework designed to merge advanced 3D graphics with Virtual Reality interfaces in order to create an appropriate environment to study and learn relativity as well as to develop some intuition of the relativistic effects and the quadri-dimensional reality of space-time.

In this paper, we focus on designing and implementing an easy-to-use game-like application : a carom billiard. Our implementation includes relativistic effects in an innovative graphical rendering engine and a non-Newtonian physics engine to treat the collisions.

The innovation of our approach lies in the ability i) to render in real-time several relativistic objects, each moving with a different velocity vector (contrary to what was achieved in previous works), ii) to allow for interactions between objects, and iii) to enable the user to interact with the objects and modify the scene.

To achieve this, we implement the 4D nature of space-time directly at the heart of the rendering engine, and develop an algorithm allowing to access non-simultaneous past events that are visible to the observers at their specific locations and at a given instant of their proper time. We explain how to retrieve the collision event between the pucks and the cushions of the billiard game and we show several counterintuitive results for very fast pucks.

The effectiveness of the approach is demonstrated with snapshots of videos where several independent objects travel at velocities close to the speed of light, c.

Categories and Subject Descriptors (according to ACM CCS): I.2.3 [Computer Graphics]: Physically-Based Modeling—I.2.3 [Computer Graphics]: Modeling&Simulation—I.2.4 [Computer Graphics]: Realtime Rendering—I.2.3 [Computer Graphics]: Collision Detection—I.2.4 [Computer Graphics]: Edutainment—

1. Introduction: understanding relativity by simulation

Learning physical sciences often requires imagination and a solid ability to abstraction. This is particularly true in the case of the theory of Relativity, even limited to the case of Special Relativity, which modifies our basic intuition about space and time, and replaces them by notions that are not directly accessible to ordinary human experience.

The theory of Relativity teaches us that space and time are neither absolute, i.e. independent of the observer (or the reference frame associated with the observer), nor independent from one another. Instead, they make up a global geometric structure with 4 dimensions, called space-time, whose "time" and "space" components depend on the

† tony.doat@limsi.fr
‡ parizot@apc.univ-paris7.fr
§ jean-marc.vezien@limsi.fr

DOI: 10.2312/EGVE/JVRC11/009-018

reference frame used to describe physical bodies and events in terms of positions and instants. In particular, the *length* of a given object, as well as the *duration* of a phenomenon (between two well-defined events) will *be* – not only appear – different for two observers moving with respect to one another.

As it turns out, this has some very deep consequences about the nature of space and time, that are in direct contradiction with our intuition. As ordinary human experience is limited to very slow velocities compared to the speed of light, c, it leads to a number of physical effects that are particularly puzzling to the student who first discovers them, and even to experienced physicists. This basic limitation, however, can be challenged and actually overcome by modern numerical simulation systems where universal constants, such as the speed of light, can be adjusted at will. Computer Graphics (CG) techniques can recreate physically realistic phenomena, while Virtual Reality (VR) can immerse users into a virtual world where the speed of light is reduced to a few centimetres per second.

This perspective triggered the creation of an interdisciplinary team, gathering VR specialists, physicists and didacticians, with the aim of merging advanced 3D graphics with VR interfaces in order to create an appropriate environment to study Relativity as well as to develop some intuition of the quadri-dimensional reality of space-time.

In the present work we focus on the building blocks that compose our application, featuring the simulation of a relativistic carom billiard. On the one hand, we describe the rendering part, with an innovative relativistic rendering engine based on the space-time interval. On the other hand, we describe the physical simulation part with an original collision engine based on collision anticipation.

In section 2 we present an overview of previous works on learning and simulating Special Relativity. Section 3 presents the main difficulties faced when designing a relativistic carom billiard application along with the proposed solutions, while section 4 illustrates the results and the performances of our methods. Section 5 summarizes the work and indicates future developments.

2. Virtual relativity: challenges and issues

Several researchers [SSV02, SSV01, DKP10] have studied the problem of how students learn relativity concepts (invariance of the speed of light, frame of reference, simultaneity of space-time events). It was found that, very often, even Master students do not properly grasp the fundamentals and consequences of relativistic physics. Hence there is obvious interest in using modern computer simulations to help them.

Very early, mathematics were used to predict what objects would look like in relativistic motions. The first solutions appeared in 1959 with Penrose [Pen59] and Terrell [Ter59]. Then, researchers focused on various relativistic effects in light interaction (Doppler, search light, aberration of light, acceleration). For real-time rendering, two main techniques were proposed: the ray-tracing technique [HD89] and the polygon rendering technique [HTW90]. Weiskopf extended the basic rendering techniques and presented a solution exploiting modern graphic cards (GPU) for Special [Wei01] and General Relativity [WSE04]. Recently, Savage [SSM06, SSM07] developed a game-like computer simulation where the user can fly at relativistic speeds and see the world according to his reference frame in real-time.

However, all these techniques only handle one moving object (i.e. one reference frame) relative to an otherwise static world. In effect, the simulated world, although 4D in nature, is thus mostly a hybrid 3D+1D simulation. Having two or more moving observers participating to the same scene would allow advanced exploration by seeing the same *4D events* from different points in space and with different reference frames. Also, current works do not allow users to modify the scene. This significantly reduces the power of such tools to explore other features (e.g. kinematics and dynamics) of the theory of Relativity and actually *experience* the 4D spacetime. We believe that promoting the user from the status of observer to the status of actor, which is at the heart of VR, opens a fruitful way to improve students understanding of the theory by making "thought experiments" more concrete.

In this context, our approach consists in creating an application that i) embeds the laws of Special Relativity and 4D spacetime, ii) visually renders the phenomena in a realistic fashion, with no restriction on the relative motion of the observer and/or the objects in the scene and iii) enables the user to alter the simulation by specifying the motion of objects. As a first test case, we present an easy-to-use application that is likely to awake curiosity about a 4D world: a relativistic carom billiard (previously introduced in a poster [DPV11]). The carom billiard is close to the snooker game in essence but has fewer balls and no holes. This French billiard game enables the users to observe and interact with several independent objects. Throught relativistic motions and collisions, we aim to highlight the notion of *event* which is a key concept from the didactic point of view. In this paper we focus on two main aspects of such a simulator: the rendering engine and the physics engine.

3. Simulating a relativistic carom billiard

3.1. Specificity of the relativistic spacetime

Two key features of the theory of Special Relativity have a direct impact on our ability to simulate and render an accurate relativistic spacetime:

- the speed of light c is finite (and invariant), so we don't

see the objects where they are *now*, but where they were when they emitted the photons that we perceive now. The determination of what a given observer effectively sees at a given location at a given time (i.e. at a given point in spacetime), requires a framework in which the whole history containing the past positions of the various objects of the scene is accessible to find the emission event.

- lengths and durations are not invariant and depend on the relative velocity between the objects and the reference frames involved. Thus, there is *a priori* a conflict between the intrinsic definition of the objects in their own reference frame and their actual occurrence in other reference frames, with respect to which they are moving. More precisely, Special Relativity teaches us that, in these other reference frames, the (instantaneous) lengths between two given points of the object are generally *not* the same. For instance, a billiard ball that is intrinsically a sphere, is no longer a sphere when described in the rest frame of the billiard board (see Fig.3), with respect to which it is moving. This calls for a consistent description of the objects in *any* reference frame, i.e. in the 4D spacetime reality itself.

In addition to these key features, the dynamics of the relativistic world are different from the classical dynamics usually simulated in VR. In our case, the main differences appear in the outcome of the collisions (between the balls, see Sec. 3.3.2.2, and between a ball and a cushion, see Sec. 3.3.2.3). A detailed description of the underlying physics will be given in a later article. Here, we simply sketch how we consistently handle the interactions of objects in the simulation, and indicate some interesting features which are relevant to didactic studies of Relativity.

3.2. Implementing relativistic rendering

Both of the above-mentioned effects – the modification of the objects geometry and the photon propagation delay – combine in a non-trivial way so that objects can appear to be significantly distorted compared to what would be expected in their reference frame. Likewise, because of the propagation time of the photons, the objects appearance will vary depending on their *apparent* velocities relative to the observer.

3.2.1. Exploiting the invariance of space-time intervals

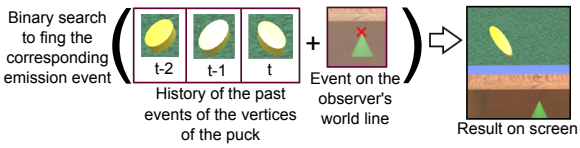


Figure 1: *Rendering a 3D object in Relativity: a combination of the finite speed of light effect and the length contraction(included in the history, see Sec.3.2.3).*

Different objects, and even different points of a given object, may obviously be located at different distances from the eyes of the observer. Therefore, even though the velocity of light is the same in all directions, the photons arriving at the same time where the observer stands – which is referred to as *the observation event*, i.e. the here-and-now of the observer, where the image is calculated –, must have been emitted at different times. This will create visual distorsions. The key challenge is thus to determine, for each vertex of the simulated world, the *emission event* of the photons reaching the eyes of the observer at the observation event.

To calculate physically correct images, our solution (Fig. 1) consists in creating a copy of the mesh of the object and for each frame, modifying the position of each vertex according to what the observer actually sees at a given time under the laws of Relativity. The updated mesh can be rendered by the classic graphics pipeline. In other words, our strategy amounts to inserting a dedicated stage in the simulation pipeline, which takes as an input the scene defined in a particular reference frame (here, the rest frame of the billiard board), and outputs the scene instantaneously seen by the observer, given his position and his velocity.

As it turns out, the determination of these emission events is actually simple and fast, if one uses the space-time interval between two events (more details in our previous work [DV10] and more generally in [Boh10]), defined as:

$$\delta s^2 = c^2 \delta t^2 - \delta l^2 \quad \text{(time-like convention)}, \qquad (1)$$

where δt and δl are respectively the time interval and the (spatial) distance between the events. Both δt and δl depend on the reference frame, but the central feature of Relativity is that δs does not: it is invariant.

Now given that photons propagate along paths with null space-time intervals, the emission event, $(t_e, \vec{r}_e)$, and the observation event, $(t_o, \vec{r}_o)$, are always related by the simplest equation: $\Delta s^2 = 0$. Then, for each vertex of the simulated scene, the emission event corresponding to the current observation event is obtained as *the* event of its timeline (whether the object is moving or not) that satisfies $\Delta s([\text{Emission}], [\text{Observation}]) = 0$. Causality further ensures that this event is unique, and we obtain it by solving this equation, noting that Δs is a monotonic function of the vertex timeline events, for any fixed observation event. Photons emitted by the vertex at instants for which $\Delta s < 0$ have not yet reached the observer, while those emitted at instants for which $\Delta s > 0$ have already struck him in the past.

The use of a dichotomy procedure to solve the above equation for each vertex of the scene proves fast enough, even for quite elaborated scenes. Besides, since the space-time interval is an invariant, the technique can be applied identically in any reference frame, ensuring a generic rendering approach, whatever the number of moving objects and observers.

Let us now explicit how the past positions of the vertices is stored and retrieved during the simulation.

3.2.2. Implementing the Lorentz contraction of lengths

In Computer Graphics, each object is generally discretized with an appropriate number of vertices organized in a mesh structure. This definition of objects, however, may be problematic in a relativistic context, because this intrinsic definition specifies the relative distances between the vertices *in the rest frame of the object*. Now, a consequence of Relativity is that the instantaneous distance between two vertices depends on the reference frame. To follow the evolution of the simulation, we chose the billiard board as a natural reference frame. Then, if an intrinsically spherical ball is moving with respect to the board, it is *not* spherical in the board rest frame, and the position of its vertices must be computed consistently. The so-called Lorentz transform specifies how coordinates transform from one frame to another, and it is easy to show that the net effect of the movement of an object at velocity v is an effective contraction of all distances along the direction of the motion, with a factor, γ, called the *Lorentz factor*, defined as:

$$\gamma = \frac{1}{\sqrt{1 - v^2/c^2}}, \qquad (2)$$

This Lorentz factor essentially measures the amplitude of the relativistic effects. For instance, a spherical ball moving with Lorentz factor γ becomes, in the board frame, an ellipsoid with a smaller axis in the direction of motion, equal to the radius of the sphere divided by γ. Faster moving balls are thus more "compressed". When $v \ll c$, it is essentially equal to 1, which means no difference with respect to the Galilean case. When v reaches sizable fractions of c, γ increases, and so do relativistic effects, becoming divergent as $v \to c$.

Thus, before considering any time delay related to light propagation, each object initially defined in its own rest frame must first be re-defined as an effective mesh, in the natural rest frame of the simulation, according to the velocity and movement direction it exhibits at a given time.

3.2.3. History data structure and real time updating

As indicated above, the rendering of a relativistic scene involves a search in the history of each vertex. Our strategy is the following: each time an image is generated for the user, we store the position of each vertex at the current simulation time, after applying Lorentz contraction to the intrinsic definition of the objects (see Fig. 2). The history table built in this way can then be deep-searched through by dichotomy to find the emission event associated with each vertex, at any later observation event. If additional precision is needed, a linear interpolation of the emission event is made between two successive table cells.

Finally, we record the *proper time* of the vertices as they move around in the scene as an additional parameter in the history data structure. For each vertex, the *proper time* is incremented by $\Delta t' = \frac{\Delta t}{\gamma}$ at each updating step, where Δt is the time increment in the simulation frame and γ is the Lorentz factor given by Eq. (2), which implements *time dilation*.

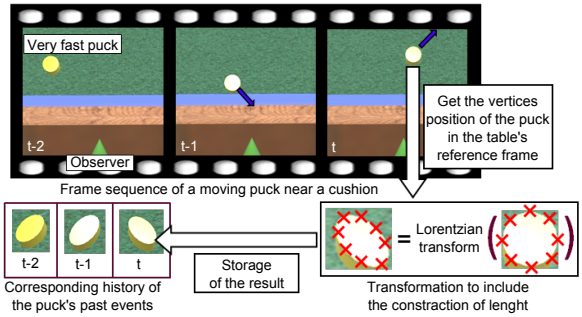


Figure 2: *Example of updating one vertex in the history*

3.3. Implementing relativistic dynamics

Let us now present in more details the physics engine implemented in the application in order to properly simulate collisions between the billiard components. Such physical accuracy is important for users to be able to develop the correct intuition about relativistic dynamics, in addition to the above-mentioned purely kinematic effects. Before describing the main features of our physics engine, let us first stress a fundamental caveat when dealing with solids in Relativity.

3.3.1. Solids and rigid rotation in Relativity

Solids in Physics are but idealizations of actual physical bodies defined as macroscopic entities whose components remain at constant distances from one another, never experiencing any stretching or compression. This is in direct conflict with a key finding of the theory of Relativity: since no information can propagate faster than c, when two solids collide the contact points start changing direction while the other points cannot know anything about the collision yet. The velocities *cannot* change instantly throughout the body, which must thus deform. In other words, a solid cannot exist!

This is a serious problem, which is quite interesting by itself and leads to instructive paradoxes in the context of our application, with great didactic value. The only way out that would be physically satisfactory would be to simulate the balls as a network of interacting particles and follow their motion under the influence of mutual forces transmitted locally at a velocity lower or equal to c. This, however, would be extremely time consuming, and incompatible with real time rendering. We will address this particular problem in a forthcoming paper. For now, we stick to the assumption

that the simulated objects are perfect solids rigidly attached to their center, and set up to determine the motion of these centers according to the laws of Relativity. Doing so, we are prepared to face non physical phenomena appearing *during* the collisions, over time scales corresponding to the "light-width" of the objects involved.

Another important caveat is that the rotation of the balls on the board implies that different points have different velocities, and the ball cohesion requires internal forces that must become extremely large. In addition, a translation of the ball centers at a velocity close to the speed of light is physically impossible if the balls are not gliding (instead of simply rolling) on the board, as rigid rotation would imply the top part to move faster than c. Therefore, in order to avoid internal contradictions and unnecessary complications, we chose to explicitly replace the balls by cylindrical pucks, whose flat bottoms make them simply glide over the board. We thus leave aside the specific problem of rotations, to be addressed in future works.

3.3.2. Handling relativistic collisions

3.3.2.1. General strategy The detection of collisions in a relativistic framework is more complicated than in usual computer simulations, because the shape of the objects and their geometric extension depend on their motion with respect to the reference frame in which they are considered (see Sect. 3.2.2).

In the non relativistic approximation, the pucks are simply cylinders (or circles, as projected on the board) of radius R. At any given time of the simulation, one may check for collision between objects. For puck-puck collision and puck-cushion collision, it occurs when the distance becomes lower respectively than $2R$ and than R.

When the pucks move with relativistic velocities, however, they no longer define circles in the board reference frame $\mathcal{R}$, but ellipses with major axis $2R$ and minor axis $2R/\gamma$. Determining the contact point of such ellipses moving with respect to one another or to a cushion is far more complicated. In the first case, the collision solution involves 4th-order equations.

We overcome this difficulty i) by reformulating the problem in a convenient reference frame and ii) by anticipating collisions so that we do not have to run a collision search algorithm at each time step. Starting from a given state of the billiard game, we compute in advance the time and location of any possible future collision. This can be done analytically, with pucks moving in straight lines between collisions. From the instants of all potential collisions, assuming nothing else occurs in the meantime, we simply determine which one will occur first, and we can then safely let the simulation run freely until that instant. At this time, we determine the outcome of the collision according to relativistic kinematics (see below), and only then do we have to recompute the anticipated times of possible collisions and update the next collision event. This allows to process chained collisions on the same frame. For each frame, if no collision occurred, we simply move the pucks according to their velocity.

3.3.2.2. Puck-puck collision The appropriate reference frame to treat puck-puck collisions is the so-called "center-of-mass frame", in which the total momentum of the two-pucks system is zero. In this frame, the two pucks have exactly opposite velocities and thus the same Lorentz factor. Their elliptic shape is also the same, and we can be certain that the collision event will occur at the middle of their centers, which is at rest in this frame. A more complete description is proposed in Appendix. Sec. 4.2.2 demonstrates that real-time collision detection is achieved.

3.3.2.3. Puck-cushion collision To determine the future collision event between a puck and a cushion (see Fig. 3), the best choice is to place oneself in the rest frame of the puck, where it has its proper circular shape. In this frame, the cushion obviously moves with the velocity opposite to that of the puck in the board rest frame. But although the cushion remains rectilinear, its orientation is tilted, depending on the puck velocity vector. Once this orientation is calculated properly from the laws of Relativity, the first contact point is identified has the intersection between the puck circle and the radius perpendicular to the cushion. This radius also provides the recoil direction of the puck. Finally, a simple inverse Lorentz transform solves the collision problem (event and outcome) in the board rest frame. It can be verified analytically that the resulting reflection angle is identical to the incident angle if (and only if!) the collision is elastic.

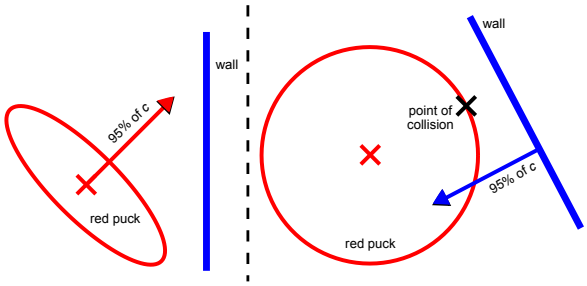


Figure 3: *A moving pucks onto a cushion, respectively, in the board frame and in the puck frame*

4. Experiments

We now present some examples of the results and performances obtained with our real-time billiard simulation. Figure 4 shows the general set up. All pictures below are computed for the same point of view, assuming the observer is standing at rest beside the board, although

he/she could of course move around at any (including relativistic) velocity. We have reduced the speed of light to a velocity such that it takes 2 seconds to cross the board.

Figure 4: *The carom billiard running in the EVE CAVE (CNRS/LIMSI, Orsay, France)*

4.1. Rendering images

As explained in Sect. 3.1, the scene viewed by the observer at a given time gathers earlier positions of the different points of the objects. Figure 5 shows a situation with all pucks in parallel tracks at different velocities. But the observer actually sees pucks with different relative positions and shapes (Fig. 6). In Fig. 7(b), a zoom shows the combined effect of the length contraction and the finite speed of light. Although not visible in these static images, a very striking effect is that the apparent velocity of the pucks with respect to the board also depends on the direction of their motion with respect to the observer. This, as well as other relativistic effects (cf. Sec. 4.3), is not precomputed: it derives naturally from the physical laws embedded in the rendering algorithms of the simulation.

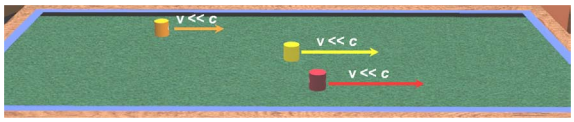


Figure 5: *Actual positions of the pucks, at a given time in the billiard board rest frame.*

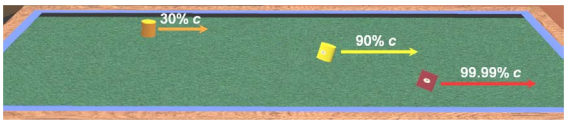
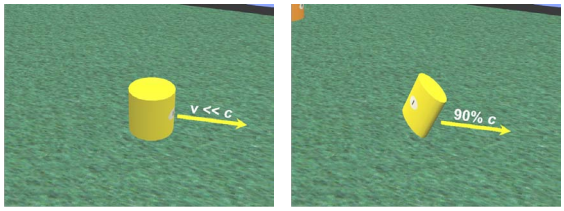


Figure 6: *The scene of Fig. 5, as seen by the observer.*

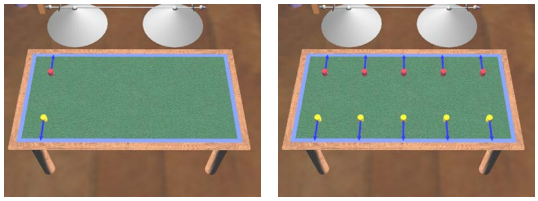
4.2. Performance evaluation

In this section, we evaluate the performances of both our rendering and collision engine within a common test framework. The tests were run on a 64-bits PC architecture, featuring a 3-Ghz processor, a Nvidia QuadroFx 1700 graphics card and 4 Gbytes of RAM. The test scene consists in pair-wise collisions of pucks on the carom billiard table, ranging from 2 to 10 pucks (see Figs. 8(a) and 8(b)). The rendering and collision engine were evaluated separately on short simulation sequences (two back-and-forth travels for each pair of pucks). The scene in Fig. 8 contains $\sim$ 30000 vertices, but only the $\sim$ 4960 vertices of the pucks are actually concerned by the relativistic engines (cf. Sect. 3.2.3).



(a) (b)

Figure 7: *Artificial view of a puck. Here the computed scene is the same as in Figs. 5 and 6 but we (unrealistically) change the point of view to show details.*



(a) The tested scene with couple of pucks (b) The tested scene with ten pucks

Figure 8: *The tested scene with different configurations*

4.2.1. Rendering engine

Figure 9 shows the performance characteristics of the rendering engine depending on the number of relativistic objects. Here the number of memorised events (positions) for each vertex of the scene is set to 1000 in the history array, which is enough given the time frame of the test. From the results, we can infer that the rendering part of the basic three-pucks simulation typically takes about 3.5 ms per frame. Table. 1 shows the memory occupation as a function of the number of pucks and vertices. Both frame rendering time and memory usage vary linearly as complexity increases.

4.2.2. Physics engine

Figure 10 shows the performance characteristics of the physics engine as a function of the number of pucks. The

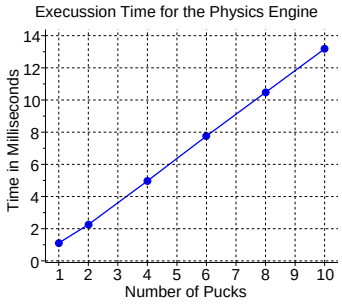


Figure 9: *performance of the rendering engine.*

Pucks	Vertices	Events	Memory Usage
1	496	1000	9,46 Mo
2	992	1000	18,92 Mo
4	1984	1000	37,84 Mo
6	2976	1000	56,76 Mo
8	3968	1000	75,68 Mo
10	4960	1000	94,60 Mo

Table 1: *Table of the memory usage of the rendering engine.*

middle curve is the averaged collision computation time per frame during the complete simulation. The top curve is the maximum time observed, typically on a collision frame (between pucks or with a cushion). The lower curve is the minimum time, in the absence of collisions. These three curves clearly show that the collision anticipation method (see Sect. 3.3.2) provides high performance calculation, the time becoming more important when several collisions occur in the same frame (Table 2). Still, the total run-time dedicated to physics remains well within real-time requirements. In the standard three-pucks simulation, the average calculation time is ~ 0.007 ms.

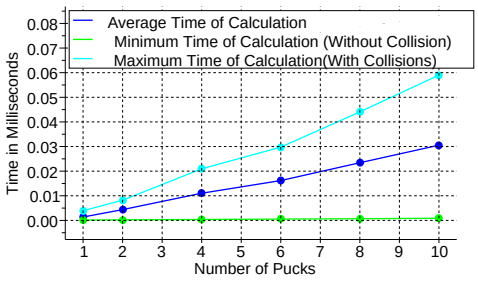


Figure 10: *performance of the physics engine*

4.2.3. Conclusion

Currently, the basic carom billiard application chosen to host special relativity experiments uses three pucks. The complete scene with $\sim$ 30000 vertices (1488 for the relativistic rendering engine) runs on the test PC with a framerate of about $\sim$ 80 fps. In this configuration, the computational time for the combined rendering and physics engines is only $\sim$ 3.5 ms. This time remains small and well within real-time requirements, proving it is possible to render Lorentzian kinematics, the propagation effects due to the finite speed of light as well as non-Newtonian physics. Of course it is now envisaged to use more elaborated simulation scenarios with more complex scenes, in an immersive context (see Fig. 4 where the billiard game is played in a CAVE-like facility).

	2 Pucks	10 Pucks
Frame Withought Collision (ms)	0.0002	0.0009
Frame With Collision (ms)	0.008	0.055
Number of Collision	2	15

Table 2: *Table of the calculation time for frame generation with and withought collisions for two and ten pucks*

4.3. Relativity for didactics: some observations

Interaction with the simulation is made possible by applying impulsions to the pucks, to observe the finite speed of light effect and the Lorentzian length contraction in a carom billiard. Furthermore, our application allows to observe some subtle consequences of the theory of Special Relativity which are particulary important for didactics:

- Apparent simultaneity: our application includes the ability to replay the current simulation. By changing his location, the user can observe events to appear simultaneous in a reference frame (say two pucks encountering a cushion at the same time in the board frame) but non simultaneous in another. The awareness that a scene does not unfold identically in all reference frames is at the heart of the understanding of 4D relativity.
- Apparent acceleration and deceleration of the pucks: during the evolution of the simulation, the apparent velocity of the pucks vary depending on their motion with respect to the observer. This effect is reminiscent of, but distinct from the well-known Doppler effect: if the puck is moving away from the observer, its velocity seems to be smaller than if it is approaching.
- Angle after collision: from the didactic point of view, it is interesting to note that the relativistic dynamics of collisions is significantly different from the classical case. In particular, if a puck collides another puck at rest on the board, the recoil (resp. deflection) angle will be larger if the impact parameter is smaller (resp. larger). With classical laws of physics, it can be shown that after the collision, the redirected pucks moves in a direction that is always perpendicular to the recoil direction of the hit puck. However, this intuitive expectation does not hold any longer in relativistic physics. Resulting trajectories after collisions at velocities close to the speed of light

exhibit an angle that is systematically smaller than 90°, asymptotically approaching zero as the initial velocity approaches c.

We believe that using our application to experience these effects "without thinking" will help to develop intuition on relativistic behaviours while trying to play billiard properly at relativistic velocities. It is expected to help students in their efforts to understand Einstein's theory from a practical point of view. This will be tested by the EVEILS (acronym for Virtual Spaces for Scientific Exploration and Education)group through a dedicated research work in didactics and formal evaluations on physics students.

5. Conclusion and future work

This work introduces an application that simulates a relativistic carom billiard, in which the speed of light may be reduced to such a velocity that unfamiliar or even counterintuitive effects can be directly experienced. This tool significantly improves on previous attempts to simulate our 4D space-time continuum, by extending the observation of static environments by one observer moving at relativistic speed to exploration and interaction with 4D objects of arbitrary velocities and motions, in arbitrary reference frames.

We presented new solutions for relativistic rendering, based on a data structure that keeps track of the spacetime positions of the (vertices in the) different objects, and an algorithm that efficiently retrieves the events that generated the photons seen by an observer at a given time. This algorithm exploits the space-time interval between any two events, and the fundamental invariance of the speed of light in all reference frames. The fact that it relies on the underlying principles laid out by Einstein is a reason for its efficiency, which allows us to render complex relativistic scenes in real time.

We also introduced a physics engine dedicated to the simulation of the collisions of the pucks between each other and with the billiard cushions. We solved the problem of collision detection for relativistic pucks with arbitrary motion, which is much more complicated than in the classical case. This is done thanks to a suitable change of reference frame implemented in an "anticipation algorithm" that computes the collision events and the outcome of the collisions.

This allowed us to emphasize some fundamental differences between the classical and relativistic cases, from the point of view of kinematics as well as dynamics. We believe that the ability for the user to discover these effects by himself, through a direct exploration of and interaction with a relativistic 4D scene that is physically consistent, is of great didactic value. Developing a more intuitive understanding of Relativity through experience should also be valuable for experienced physicists. Initial polls were already conducted on physics students to clearly identify common misconceptions and problems encountered while learning Relativity. As framework, we believe that immersive Virtual Environment offer more intuitive interface than a standard 3D desktop for our application. This point will be studied in details in a futur paper.

We are now in the process of designing VR experiments based on these findings [SMM*10, MSW*08]. Experiments run in the setup presented in Fig. 4, will expose challenging issues such as the change of reference frame or the relativity of simultaneity to students of different levels.

In the Virtual Environment, we adopt an egocentric point of view based on a tracking device. Thus, users can move in the scene and chose different points of view to observe one directly. With the tracking device a new difficulty has become: this device is disturbed by a noise. This is imperceptible for the classical scene but for the relativistic scene it is becoming extremly disturbing. Due to the aberration effect, for a velocity of the user approaching the simulated speed of light, a small variation of this velocity causes a big deformation of the showed scene.

To improve interaction and the understanding of the difficulty to access to relativistic mouvements in term of energy, we plan to include haptic device including a coherent mapping of energy to show that approaching the barrier c require an exponential amount of energy.

In the future, we plan to improve our application by simulating additional rendering artefacts such as aberration, as well as improving the dynamics of the system with the addition of relativistic friction and non-elastic collisions, which can be implemented without changing our computational framework.

On a longer run, we would like to extend the collision engine to more complex objects (i.e. generic polygon meshes) and to investigate relativistic objects in rotation. In terms of rendering, the Doppler and searchlight effects can also be implemented, but the pedagogical value of such essentially visual effects is still unclear.

From the computational point of view, the running time can be reduced by the use of parallelization methods. For example, the back-propagation of photons using the space-time interval can be calculated for each vertex independently, which makes for an easy parallelization.

6. acknowledgements

We thank Cécile De Hosson and Isabelle Kermen for their helpful advice. We would like to thank the French National Research Agency for funding this research via grant $BLAN08-3_314843$.

References

[Boh10] BOHM D.: *La Théorie de la Relativité restreinte*. Le Serpent à plumes, 2010. 3

[DKP10] DE HOSSON C., KERMEN I., PARIZOT E.: Exploring students' understanding of reference frames and time in galilean and special relativity. vol. 31, pp. 1527–1538(12). 2

[DPV11] DOAT T., PARIZOT E., VÉZIEN J.-M.: A carom billiard to understand special relativity. In *Virtual Reality Conference (VR), 2011 IEEE* (2011), pp. 201–202. 2

[DV10] DOAT T., VÉZIEN J.-M.: Virtual reality to experiment out-of-reach physics in a relativistic framework: progress report. 3

[HD89] HSIUNG P.-K., DUNN R. H. P.: Visualizing relativistic effects in spacetime. In *Proceedings of the 1989 ACM/IEEE conference on Supercomputing* (New York, NY, USA, 1989), Supercomputing '89, ACM, pp. 597–606. 2

[HTW90] HSIUNG P.-K., THIBADEAU R. H., WU M.: T-buffer: fast visualization of relativistic effects in space-time. In *Proceedings of the 1990 symposium on Interactive 3D graphics* (New York, NY, USA, 1990), I3D '90, ACM, pp. 83–88. 2

[MSW*08] MCGRATH D., SAVAGE C., WILLIAMSON M., WEGENER M., MCINTYRE T.: Teaching special relativity using virtual reality. In *Proceedings of the Assessment in Science Teaching and Learning* (Oct. 2008), vol. 1263, pp. 67–73. 8

[Pen59] PENROSE R.: The apparent shape of a relativistically moving sphere. In *Mathematical Proceedings of the Cambridge Philosophical Society* (1959), vol. 55, pp. pp 137–139. 2

[SMM*10] SAVAGE C., MCGRATH D., MCINTYRE T., WEGENER M., WILLIAMSON M.: Teaching physics using virtual reality. In *American Institute of Physics Conference Series* (July 2010), B. Paosawatyanyong & P. Wattanakasiwich, (Ed.), vol. 1263 of *American Institute of Physics Conference Series*, pp. 126–129. 8

[SSM06] SAVAGE C. M., SEARLE A. C., MCCALMAN L.: Real time relativity. *ArXiv Physics e-prints* (July 2006). 2

[SSM07] SAVAGE C. M., SEARLE A., MCCALMAN L.: Real time relativity: Exploratory learning of special relativity. 791–798. 2

[SSV01] SCHERR R. E., SHAFFER P. S., VOKOS S.: Student understanding of time in special relativity: Simultaneity and reference frames. S24–S35. 2

[SSV02] SCHERR R. E., SHAFFER P. S., VOKOS S.: The challenge of changing deeply held student beliefs about the relativity of simultaneity. 1238–1248. 2

[Ter59] TERRELL J.: Invisibility of the lorentz contraction. *Phys. Rev. 116*, 4 (Nov 1959), 1041–1045. 2

[Wei01] WEISKOPF D.: *Visualization of Four-Dimensional spacetimes*. PhD thesis, Universit\atsbibliothek, 2001. 2

[WSE04] WEISKOPF D., SCHAFHITZEL T., ERTL T.: Gpu-based nonlinear ray tracing. vol. 23, Blackwell Publishing, Inc, pp. 625–633. 2

Appendix: solving the puck-puck collision problem

In this Section, we briefly describe how we solve the collision problem between two pucks, P_1 and P_2, knowing their position and velocity vectors at a given instant in the billiard board rest frame. The goal is to determine whether the two pucks will collide, and if they do, where and when will the "collision event" take place.

First, let us note that the height of the cylindrical pucks doesn't play any role in the collision problem, and we shall thus consider the pucks as intrinsically 2D circles, with radius R, moving in the plane of the board. The board itself is chosen as the simulation reference frame, noted $\mathcal{R}$, in which the collision problem needs to be solved.

Because of the length contraction effect associated with Lorentzian kinematics, the two moving pucks are not represented by circles with diameter $2R$ in reference frame $\mathcal{R}$, but by ellipses with major axis $2R$ and minor axis $2R/\gamma$, where the Lorentz factor $gamma = (1-v^2/c^2)^{-1/2}$ depends on the velocity of the puck under consideration. The minor axis coincides with the direction of motion, and is also a priori different for each puck. We can identify the position of the center of pucks P_1 and P_2 by the vectors $\vec{r}_1$ and $\vec{r}_2$ in the board rest frame.

As the pucks move around on the board, they may or may not collide, and finding the collision point in the board reference frame $\mathcal{R}$ requires posing and solving a complicated equation of order 4, which is neither practical nor efficient in the context of a real-time simulation. However, the situation is much simpler in the so-called "center-of-mass frame", defined as the reference frame in which the total momentum of the two-pucks system is zero. Let $\mathcal{R}'$ denote this particular reference frame.

A simple analysis of the underlying kinematics shows that this frame moves with respect to the board at a velocity $\vec{v}_0$ given by

$$\vec{v}_0 = \frac{\gamma_1\vec{v}_1 + \gamma_2\vec{v}_2}{\gamma_1 + \gamma_2}, \tag{3}$$

where $\vec{v}_i$ is the velocity vector of puck i, and γ_i is the corresponding Lorentz factor (note that we have assumed all pucks to have the same rest mass).

This velocity (see Fig. 11) fully determines the Lorentz transform to be applied to compute the coordinates of the puck centers in the center-of-mass frame, $\mathcal{R}'$, at any given time in $\mathcal{R}$. Of course, since the two pucks are not located at the same position, the corresponding events in $\mathcal{R}'$ are not simultaneous. We thus need to "retropropagate" the earliest pucks in order to start, in $\mathcal{R}'$, with simultaneous positions of the two pucks centers: $\vec{r}'_1$ and $\vec{r}'_2$.

The velocity $\vec{v}_0$ given by Eq. (3) can also be used to compute the velocities $\vec{v}'_1$ and $\vec{v}'_2$ of the balls in the center-of-mass frame. Applying the relativistic law for the composition of velocities, we obtain:

$$\vec{v}'_i = \frac{\vec{v}_{i,\parallel} - \vec{v}_0 + \frac{\vec{v}_{i,\perp}}{\gamma_i}}{1 - \frac{\vec{v}_i\cdot\vec{v}_0}{c^2}}, \tag{4}$$

where $\vec{v}_{i,\parallel} = \vec{v}_i \cdot \vec{v}_0/v_0^2$ is the component of $\vec{v}_i$ parallel to $\vec{v}_0$, and $\vec{v}_{i,\perp} = \vec{v}_i - \vec{v}_{i,\parallel}$ is the perpendicular component.

Thanks to the above choice for $\vec{v}_0$, the two pucks actually

have exactly opposite velocities in $\mathcal{R}'$: $\vec{v}'_1 = -\vec{v}'_2$(Fig. 12). This implies that they have identical Lorentz factors, $\gamma'_1 = \gamma'_2$, so the effective contractions are the same and both pucks have the same shape in this frame.

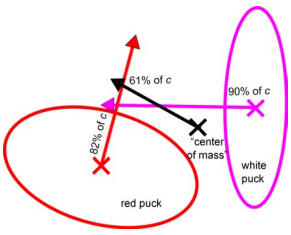


Figure 11: *The moving pucks in the board frame.*

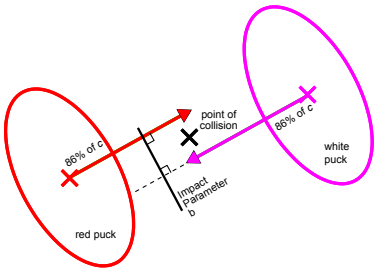


Figure 12: *Same as in Fig. 11, but seen in the center-of-mass frame.*

Thus, we are back to a manageable situation where two identical pucks (with same deformation) move along parallel tracks, deduced from the position of the centers in $\mathcal{R}'$ and their velocities. It is then easy to determine whether a collision will occur or not: if the distance between the tracks is smaller than $2R$, and the balls move *towards* (not away from) each other, the pucks will collide; otherwise, they will not.

Moreover, the symmetry of the problem in $\mathcal{R}'$ makes it obvious that the collision point will be exactly the midpoint of the pucks centers, which turns out to be standing still in the center-of-mass frame.

Finally, simple geometric considerations allow us to compute the instant of the putative collision, knowing the so-called *impact parameter*, b (which is defined as the distance between the above-mentioned parallel tracks), and the initial distance of the ball centers along their relative velocity, $D_{\parallel}$. Both are obtained straightforwardly from the positions and velocities of the pucks in $\mathcal{R}'$.

First, we determine the minimal distance of approach of the balls along their parallel trajectories:

$$d_{\min} = \frac{\sqrt{4R^2 - b^2}}{\gamma'_1}, \tag{5}$$

from which we deduce the time until collision:

$$\Delta t' = \frac{D_{\parallel} - d_{\min}}{2v'_1}. \tag{6}$$

As a final subtlety, we follow our convention to deal with solid bodies in Relativity (see the discussion in Sect. 3.3.1) and add the extra time needed for the information of this contact event to reach the center of the pucks, determining in this way the sought *collision events* (one for each puck, at its own center). A final Lorentz transform with velocity $-\vec{v}_0$ allows us to determine the coordinates of these collision events back in the rest frame of the billiard board, $\mathcal{R}$.

Now, we need to determine the resulting velocities, both in terms of their direction and their norm. This is done in the following way. The collision is first computed in the center-of-mass frame, $\mathcal{R}'$. In the case of an elastic collision, the total kinetic energy is preserved. Furthermore, since the two pucks initially have the same velocity (in norm) in that frame, as well as the same mass, the symmetry of the situation guarantees that they have the same velocity after the collision as well. The total kinetic energy is thus twice that of any of the pucks, both before and after the collision. We thus deduce that an elastic collision leaves the norm of the ball velocities unchanged.

To determine the direction of the motion of the pucks after the collision, our trick is to determine the common tangent to the ellipsoidal pucks at their contact point (when the collision occurs). This is easily done analytically, from the ratio of the great axis and the small axis of the ellipsoid, which is simply γ' (see above). At contact, the reciprocal force between the two pucks acts perpendicularly to that tangent, and is thus unable to modify the component of the balls velocity which is parallel to it. The outgoing velocities are thus fully determined by saying that:

- the component of the velocities parallel to the tangent at the contact point is unchanged
- the component of the velocities perpendicular to the tangent at the contact point is reversed (to ensure an identical norm)

In the case when the collision is *not* elastic, the same procedure applies, the only difference being that the perpendicular component must be reduced so that the kinetic energy of the pucks after the collision is the appropriate fraction of that before the collision, as determined by the elasticity coefficient.

Finally, the outgoing velocities in the billiard board frame are deduced from the above by composing them with the velocity of the board with respect to the center-of-mass frame, i.e. by applying transformation (4) with velocity $-\vec{v}_0$ instead of $\vec{v}_0$.

This algorithm proves to be very efficient, and indeed compatible with real-time rendering (see Sec. 4.2.2).