Simulating, Visualizing and Playing with de Sitter and anti de Sitter spacetime

Eryk Kopczyński[1][0000−0001−5588−1181]

Institue of Informatics, Institute of Warsaw, Poland erykk@mimuw.edu.pl

Abstract. In this paper we discuss computer simulations of de Sitter and anti de Sitter spacetimes, which are maximally symmetric, relativistic analogs of non-Euclidean geometries. We present prototype games played in these spacetimes; such games and visualizations can help the players gain intuition about these spacetimes. We discuss the technical challenges in creating such simulations, and discuss the geometric and relativistic effects that can be witnessed by the players.

Keywords: de Sitter spacetime · relativity · science game.

1 Introduction

Science-based games are games based on a real scientific phenomenon. For example, there are games based on special relativity [14,8], quantum mechanics [20], orbital physics [22], non-Euclidean geometry [24,11,4,25]. Compared to typical educational games, where the concept explained and the gameplay are not related, science games take the scientific phenomenon and use it to create interesting gameplay [20]. Other than providing entertainment, science games provide intuitive unterstanding of difficult science concepts; many of them let the player perform their own experiments, by including sandbox elements, level editors, or being open source. Such a better understanding helps young players consider a scientific career, and also gives ideas for new scientific developments to mature researchers [9]. Science games require a specialized engine for efficient modelling and visualization of the given scientific concept; such engines may be later used for other applications than just games [2].

In this paper, we describe a game (or rather, a collection of two games) taking place in de Sitter and anti de Sitter spacetimes. These spacetimes could be seen as relativistic analogs of spherical and hyperbolic geometry. De Sitter spacetime is of interest as an asymptotic approximation of our universe [16], and anti-de Sitter spacetime is of interest for its correspondence with conformal field theory [15]. While many games exist based on special relativity and non-Euclidean geometry, the unintuitive properties of spacetimes combining these two concepts seems to be still unexplored.

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63749-0_10

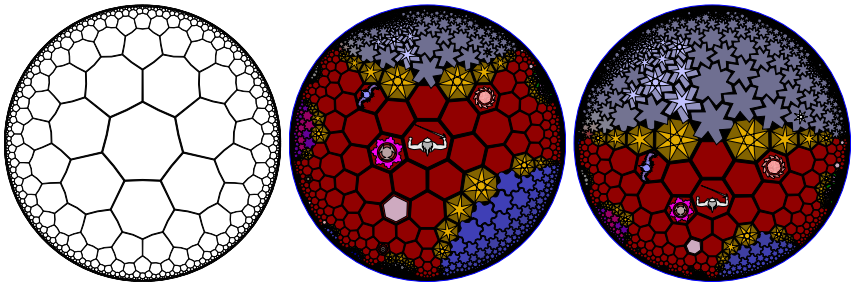

Fig. 1. On the left, the {7,3} hyperbolic tessellation in Poincaré disk model. The Poincaré disk model is conformal: it does not distort small shapes, so all heptagons look close to regular; however, it distorts scale: all the heptagons shown are of the same size. On the right, the same scene in *HyperRogue* viewed from two points. In the Poincaré model, straight lines are projected as circular arcs orthogonal to the disk boundary; moving the center shows the player that the walls (orange) are indeed straight lines.

Players can try our prototype game, named *Relative Hell*, by downloading the Microsoft Windows binary[1]. The source code, based on the non-Euclidean engine RogueViz [12], is also available.

2 Background

Our prototype helps non-experts understand scientific concepts intuitively. In this section, we provide an intuitive introduction to non-Euclidean geometry and relativity theory, and explain how games can provide such understanding.

The history of non-Euclidean geometry starts with Euclid's *Elements*. This book has changed teaching geometry by giving it structure: starting with very basic *postulates* and *axioms*, such as the space never ending and being the same everywhere and in every direction, and continuing with all the more complex geometric facts known to Euclid, which followed from the postulates and axioms [5]. However, some geometric facts related to *parallel lines* did not actually seem to follow from such basic postulates. For example, if we have two points A_1B_1 in distance d on a straight line l_1 and two points A_2B_2 in distance d on a parallel straight line l_2, the distance between A_1 and A_2 should be always the same as between B_1 and B_2. Euclid has solved this by declaring (an equivalent formulation of) this statement as his *fifth postulate* (or *parallel postulate*).

Mathematicians have been trying to prove the *parallel postulate* from the other postulates, until Lobachevsky and Bolyai have discovered *hyperbolic geometry* $\mathbb{H}^d$, which satisfied all Euclid's postulates except his *fifth postulate* [5]. (In $\mathbb{H}^d$, d is the number of dimensions.) To explain how the parallel postulate

[1] `https://zenorogue.itch.io/relative-hell`, source code: `https://github.com/zenorogue/hyperrogue/blob/master/rogueviz/ads/ads-game.cpp`. Last accessed April 9, 2024.

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63749-0_10

could not be true, imagine drawing great circles on a sphere; such great circles are an analog of Euclidean lines, however, if we look at two meridians, we discover that the parallel postulate does not hold. In our three-dimensional world, great circles are obviously curved; however, we can imagine that our Universe is actually a hypersphere in four dimensions, and what we perceive as straight lines is actually great circles on this hypersphere. This is the *spherical geometry* $\mathbb{S}^d$; hyperbolic geometry is the opposite of it (while, in spherical geometry, "parallel" lines converge, in hyperbolic geometry they diverge). While many everyday objects are spherical, hyperbolic geometry is more difficult to understand; a good way to obtain intuitive understanding is to play games, aiming to simulate the experience of a inhabitant of a non-Euclidean world. Games in $\mathbb{H}^2$ [24,11,17] typically display the view in the *Poincaré disk model* (Figure 1, which is a projection of hyperbolic plane $\mathbb{H}^2$ to a Euclidean disk, centered at the player character's position; this lets the player see that straight lines are indeed straight and acting strangely. There are also immersive visualizations of $\mathbb{H}^3$ [18,23,10,11,4].

While the world was originally assumed to be Euclidean, further experiments have shown this to not to be the case. The Morley-Michelson experiment has shown that the speed of the light measured by a moving observer is always the same (independent by the observer's velocity), which was in conflict with theories at that time. This was resolved in the theory of special relativity: Euclidean space and Newton's notion of time were replaced by *Minkowski spacetime*; there was no absolute time – for two observers O_1 and O_2 meeting at position 0 in time 0, an event at position x and time t according to O_1 would be at position x' and t' according to O_2, with both space and time being changed by Lorentz transformations ($x \neq x'$ and $t \neq t'$), a bit similar to how spatial rotations change both x and y coordinates, and causing effects such as contraction of lengths and dilation of time. Special relativity effects normally require large speed to observe in the real world, but again, they can be experimented with in games [14,8]. Special relativity could not explain gravity; for this, we need general relativity: the notion of curved spacetime, similar to the curved space of non-Euclidean geometry. Furthermore, as explained in Section 3 below, Minkowski spacetime geometry can be used to provide an elegant model of $\mathbb{H}^d$.

Most existing games and simulations, including non-Euclidean ones, do not take relativity into account. At every point, they represent the current state of the simulation (at time t) in the computer memory for all objects using the chosen internal model of the geometry, and to compute the further states of the simulation (at times $t' > t$), a chosen model of physics is used, usually some simplification and adaptation of Newtonian physics. Interestingly, while such a non-relativistic model of Euclidean spacetime obeys the Galilean principle of relativity (a moving object having no access to external reference point cannot determine that it is moving), this is no longer the case for non-Euclidean geometries. For example, in $\mathbb{H}^d$, lines diverge, so a large object moving experiences apparent centrifugal force. This is the reason why, for example, the flock of boids in the flocking simulations in RogueViz [12] cannot keep its shape, contrary to a Euclidean flocking simulation [19]. For a similar reason, HyperRogue [11] could

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63749-0_10

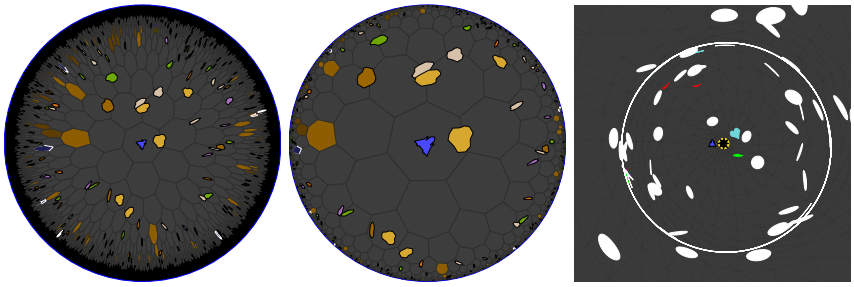

Fig. 2. Relative Hell. $\widetilde{ad\mathbb{S}^2}$ is displayed in the Poincaré disk model on the left, and the Beltrami-Klein disk model in the center. $d\mathbb{S}^2$ on the right, in stereographic projection.

not feature large, freely moving objects – the boundary of such an object would have to move significantly faster than the center, in a curved line, which would be unintuitive; so while the game does feature some somewhat large creatures (such as snakes and krakens), they are narrow enough to avoid this problem.

This can be solved by using *de Sitter* ($d\mathbb{S}^d$) and *anti-de Sitter* ($\widetilde{ad\mathbb{S}^d}$) spacetimes, which are relativistic analogs of $\mathbb{S}^d$ and $\mathbb{H}^d$, respectively. They are symmetric under the Lorentz transformations used to simulate the change of velocity, and therefore, they obey the Galilean principle of relativity. Intuitively, while $\mathbb{H}^d$ and $\mathbb{S}^d$ stretch the distances in space, $d\mathbb{S}^d$ and $\widetilde{ad\mathbb{S}^d}$ also stretch the time. The game *Relative Hell* (Figure 2) lets the player fly a two-dimensional spaceship in these spacetimes. Due to the nature of these spacetimes, the gameplay is different: in $d\mathbb{S}^2$, objects are naturally pulled apart, so the goal is to keep close to the *main star* while avoiding bullets, as in the *bullet hell* genre; while in $\widetilde{ad\mathbb{S}^2}$, objects are naturally pulled together, so the whole space is rotating in order to generate centrifugal force to balance that, and the goal is to shoot, similar to classic omnidirectional shooters such as *Asteroids*. The player can also gain insight into the usual non-Euclidean geometric and relativistic (time dilation, space contraction) phenomena, such as the exponential expansion of $\mathbb{H}^2$, and time dilation and Lorentz contraction.

3 Preliminaries

We briefly recall the definitions of basic spaces and spacetimes. For more details, see, e.g., [1] for hyperbolic geometry, and [7] for $d\mathbb{S}^d$ and $\widetilde{ad\mathbb{S}^d}$.

The *Euclidean space* of dimension d, $\mathbb{E}^d$, is $\mathbb{R}^d$ equipped with the Euclidean inner product $g_E(x,y) = \sum_i x_i y_i$. For a vector $x \in \mathbb{R}^d$, we use the notation $x[a/i]$ for the vector x with i-th coordinate replaced by a. In particular, $0[1/i]$ is the point whose i-th coordinate is 1 and the other coordinates are 0. Our Euclidean inner product defines the distance between points: $d_E(x,y) = \sqrt{g_E(x-y,x-y)}$. The length of a differentiable curve $\gamma : [a,b] \to \mathbb{E}^d$ can

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63749-0_10

be computed as $\int_a^b \sqrt{g_E(\dot\gamma(t),\dot\gamma(t))}dt$; note that $d_E(x,y)$ is also the length of the shortest curve from x to y. Orientation-preserving isometries of the Euclidean space are generated by translations $T_i^a(x) = x[x_i + a/i]$ and rotations $R_{i,j}^\alpha(x) = x[x_i \cos\alpha + x_j \sin\alpha/i][x_j \cos\alpha - x_i \sin\alpha/j]$ for $i \neq j$. Isometries preserve g_E, and thus distances and curve lengths. The basic translations and rotations can be composed to obtain other isometries. Isometries of $\mathbb{E}^d$ correspond to looking at $\mathbb{E}^d$ from another frame of reference.

In video games and computer graphics, it is convenient to use the *homogeneous coordinates*, that is, $\mathbb{R}^{d+1}$, where the extra $(d+1)$-th coordinate is always equal to 1. This method lets us represent translations and rotations as matrices.

The *spherical space (sphere)* of dimension d, $\mathbb{S}^d$, is $\mathbb{E}^{d+1}$ restricted to $\{x : g_E(x,x) = 1\}$. (For convenience we consider only spheres of radius 1 here.) The distance $d_S(x,y)$ between two points $x, y \in \mathbb{S}^d$ is the length of the shortest curve (geodesic) connecting them on the sphere, $c : [a,b] \to \mathbb{S}^d$; we have $d_S(x,y) = \arccos(g_E(x,y))$. Orientation-preserving isometries of the sphere are generated by $R_{i,j}^\alpha$ (while in the Euclidean space we needed the extra coordinate for translations, in $\mathbb{S}^d$ we can instead just use the one extra coordinate we already have). A sphere is maximally symmetric: any isometry of the underlying $\mathbb{E}^{d+1}$ that maps 0 to 0 maps the sphere to itself.

A *signature* is a $\sigma \in \{-1,+1\}^d$; we will drop the 1 and just write the signs, for example $(+,+,+,-)$; if a sign repeats, we use an exponent, for example $(+^3,-)$. The *pseudo-Euclidean spacetime* of signature σ, $\mathbb{E}^\sigma$, is $\mathbb{R}^d$ equipped with the inner product $g_\sigma(x,y) = \sum_i \sigma_i x_i y_i$. In this paper, coordinates x_i with $\sigma_i = 1$ correspond to space dimensions, and coordinates with $\sigma_i = -1$ correspond to time dimensions. Positive values of $g_\sigma(v,v)$ correspond to space-like intervals, and negative values correspond to time-like intervals. A curve is *space-like* if the integral in equation (3) is well-defined (that is, the value under the square root is always non-negative), and *time-like* if this value is always non-positive. The proper time of a time-like curve γ is defined as $\int_a^b \sqrt{-g_E(\dot\gamma(t),\dot\gamma(t))}dt$.

In special relativity, our spacetime is modelled as a pseudo-Euclidean space of signature $(+,+,+,-)$; the fourth coordinate corresponds to time, and the units of time and distance are chosen so that Einstein's constant c equals 1. Isometries of pseudo-Euclidean spacetime are similar, but for $\sigma(i) \neq \sigma(j)$ we replace rotations $R_{i,j}^\alpha$ by Lorentz boosts $L_{i,j}^\alpha(x) = x[x_i \cosh\alpha + x_j \sinh\alpha/i][x_j \cosh\alpha + x_i \sinh\alpha/j]$. The parameter α is called *rapidity*; in spacetimes with 1 time coordinate, Lorentz boosts correspond to changing the velocity of our frame of reference. Objects generally move along time-like curves; Proper time is interpreted as the time measured by a clock, or intuitively felt by a sentient creature, moving along such a curve. Time-like geodesics have the longest possible proper time. While the relative speed of moving objects $\tanh\alpha$ is bounded by $c = 1$, the rapidity is not bounded. The interaction of space and time coordinates causes well-known relativistic effects such as Lorentz contraction and dilation of time. The set of points connected to v with time-like geodesics γ such that the time coordinate of $\gamma(t)$ is increasing on is called the *future light cone* of v, and the set of points connected with time-descreasing time-like geodesics is the *past light cone*.

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63749-0_10

These light cones are preserved when we apply the isometries of $\mathbb{E}^\sigma$ that keep v. The other points are neither in the future or the past, but rather elsewhere – their time coordinate may be greater or smaller than the time coordinate of v, depending on the chosen isometry. What happens at v causally depends only on the spacetime events in the past light cone of v, and may affect only the spacetime events in the future light cone of v.

The *hyperbolic space* of dimension d, $\mathbb{H}^d$, is $\mathbb{E}^\sigma$ for $\sigma = (+^d, -)$ restricted to $\{x : g_\sigma(x,x) = -1, x_{d+1} > 0\}$. This is the Minkowski hyperboloid model of hyperbolic geometry. This is a space, not a spacetime, that is, all the curves on $\mathbb{H}^d$ are space-like. While hyperbolic geometry is usually taught in the Poincaré disk model, the Minkowski hyperboloid model is generally easier to understand (assuming familiarity with Minkowski geometry) and work with computationally because of its similarity to the natural model of the sphere. In particular, orientation-preserving isometries are generated by rotations $R^\alpha_{i,j}$ and Lorentz boosts $L^\alpha_{i,d+1}$ (corresponding to translations of $\mathbb{H}^d$), and the distance between x and y is $\operatorname{arcosh}(g_\sigma(x,y))$. While in special relativity the time coordinate is usually indexed as x_0, in this paper we prefer to make it the last coordinate, for consistency with the usual indexing of homogeneous coordinates of the Euclidean space in computer graphics.

Taking $z \in \mathbb{R}$, we can project $x \in \mathbb{H}^d$ to $x' \in \mathbb{R}^d$ by $x'_i = x_i/(x_{d+1} + z)$. This is called the *general perspective projection*. For $z = 1$ we get the *Poincaré ball model* (also called the *Poincaré disk model* in 2 dimensions). It is the hyperbolic analog of the stereographic projection of the sphere, which uses the same formula. It is the most popular method of visualizing hyperbolic geometry; just like the stereographic projection, it is conformal, meaning that the angles and small shapes are mapped faithfully. Another one is the *Beltrami-Klein ball (disk) model*, obtained for $z = 0$. Beltrami-Klein disk and Poincaré disk are called models because they are alternative mathematical representations of hyperbolic geometry; in this paper, we use only the Minkowski hyperboloid model for mathematical representation, and the disk models are used as projections for visual representation. More possible projections exist which are not special cases of the general perspective projection, for example the azimuthal equidistant projection, which renders the distances and angles from the chosen central point correctly [17]. Dozens of spherical projections are used in cartography [21]; many of them have hyperbolic analogs, available in the HyperRogue engine [11].

The *de Sitter spacetime* with d space dimensions and 1 time dimension, $d\mathbb{S}^d$, is $\mathbb{E}^\sigma$ for $\sigma = (+^{d+1}, -)$ restricted to $\{x : g_\sigma(x,x) = 1\}$.

The *wrapped anti-de Sitter spacetime* with d space dimensions and 1 time dimension, $ad\mathbb{S}^d$, is $\mathbb{E}^\sigma$ for $\sigma = (+^d, -^2)$ restricted to $\{x : g_\sigma(x,x) = -1\}$. Note that $ad\mathbb{S}^d$ has closed time-like loops, for example, $\gamma(t) = 0[\sin t/d + 1][\cos t/d + 2]$. Such closed-time loops are obtained by going around the axis $A = \{x \in \mathbb{E}^\sigma : \forall i \leq d\ x_i = 0\}$. Due to the closed time-like loops, $ad\mathbb{S}^d$ has no causal structure (everything is simultaneously in the past and future of everything else and itself). This problem is resolved by taking the universal cover of $ad\mathbb{S}^d$, i.e., the point reached by going $n \neq 0$ times around the axis A is considered a different

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63749-0_10

point in the spacetime. We will call this universal cover the *unwrapped anti-de Sitter spacetime* $\widetilde{ad\mathbb{S}^d}$, or just *anti-de Sitter spacetime* for short.

Spacetimes $d\mathbb{S}^d$ and $\widetilde{ad\mathbb{S}^d}$ will be discussed in detail in the following sections. Here, we will only remark that time-like and space-like curves and their lengths and proper times are defined as above, and that both $d\mathbb{S}^d$ and $\widetilde{ad\mathbb{S}^d}$ are maximally symmetric spacetimes, with their isometries generated from $R^\alpha_{i,j}$ and $L^\alpha_{i,j}$.

4 Simulation of the Anti-de Sitter spacetime

For simplicity, we will start with $ad\mathbb{S}^d$. When necessary, we fix $d = 2$.

The point $O = 0[1/d + 2]$ is considered the origin of the $ad\mathbb{S}^d$. Every object b in the simulation, at a specific point of its proper time t, is represented by an isometry $T_{b,t}$ of $ad\mathbb{S}^d$, which maps the coordinates relative to (b, t) to the world coordinates. Usually, (b, t)-relative coordinates of b itself at time t is O, so the world coordinates of b at time t are $T_b(O)$. The player controls a ship s, which is one of the objects in the game. To display an object at world coordinates x on the screen at time t, we need to map x into ship-relative coordinates, $x' = T^{-1}_{s,t}x$. The state of the game universe at the current time is a slice of the spacetime. This slice is $S = \{x \in ad\mathbb{S}^d : x_{d+1} = 0, x_{d+2} > 0\}$, which is isometric to $\mathbb{H}^d$, and can be rendered e.g. in the Poincaré disk/ball model.

The ship s is controlled in real time by a player, so for every time moment t displayed in an animation frame, the game has to compute the next frame transform $T_{s,t+\epsilon}$ depending on both $T_{s,t}$ and player's decisions. Objects in spacetime move along timelike geodesics if no force is acting on them. Timelike geodesics in $ad\mathbb{S}^d$ are generally of form $TR^t_{d+1,d+2}O$. Thus, if the player does not accelerate from time t_1 to t_2, we have $T_{s,t_2} = T_{s,t_1}R^{t_2-t_1}_{d+1,d+2}$. Changing the camera speed in dimension $i \in \{1, \dots, d\}$ corresponds to changing the frame of reference by multiplying it by $L^\alpha_{i,d+2}$. Thus, if the player accelerates, we also need to multiply the formula for T_{s,t_2} by $L^\alpha_{i,d+2}$ on the right.

One possible objection to displaying the slice S is that the player should not see the state of the universe at the current time – the ship at time t only knows the past light cone of $T_{s,t}O$. In the current game prototype, we assume that all the objects other than s behave deterministically, so this is not an issue—the ship could compute the current state of other objects depending on what it knows. While displaying S is less immersive, it is useful for understanding how the spacetime works. (One can change the options to get the actual view.)

The simplest deterministic movement is geodesic movement. Let $b \neq s$. The object b at any given time is not a point, but it is rather a subset $X(b) \subseteq S$. The formula $T_{b,t} = T_{s,0}R^t_{d+1,d+2}$ turns out not to be satisfactory – while the object b's origin O moves geodesically, other points in $X(b)$ do not. In $d = 2$, this issue can be fixed by using the formula $T_{b,t} = T_{b,0}R^t_{3,4}R^t_{1,2}$, which enforces geodesic movement of every point of b. For short, we denote the isometry $R^t_{3,4}R^t_{1,2}$ with M^t. (It is possible to change the options to also use M^t for the ship, but with such a setting, the game becomes more confusing to the player.) Every object

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63749-0_10

$b \neq s$ starts its lifetime at some time t_1 and ends at time t_2; the time t_1 may be $-\infty$ if the object did exist forever, and t_2 may depend on player's actions, and be ∞ if it is not yet known to be ever destroyed. We can compute $T_{b,t}$ at every time t knowing $T_{b,0}$. So, to display the point $x \in X(b)$ of object b at the ship's proper time t, we need to find t_b such that $x_b := T_{s,t}^{-1} T_{b,0} M^{t_b} x \in S$. If $t_b \in [t_1, t_2]$, we apply the chosen projection to x_b. We display the point x there.

So far, we have been assuming $ad\mathbb{S}^d$ for simplicity, which does not really work due to the time-like loops – technically, all of spacetime is in the past, so the player would be able to perceive the results of the actions they have not performed yet (this is a problem with all games featuring a powerful enough form of player control and time travel). Another issue is numerical precision: hyperbolic space (S in our case) is characterized by its exponential growth, which causes numerical errors to accumulate quickly, and using world coordinates relative to some fixed origin does not really work when we travel far enough from that origin. Our solution of these issues is based on the existing implementation of $\widetilde{G}$, the universal cover of Lie group G of orientation-preserving isometries of $\mathbb{H}^2$, also called $\widetilde{SL(2,\mathbb{R})}$ or the twisted product of $\mathbb{H}^2$ and $\mathbb{R}$. This implementation is described in paper [13]. The space of isometries of $\mathbb{H}^2$ is a three-dimensional space; the three dimensions correspond to the two dimensions of $\mathbb{H}^2$ itself (translations), plus one extra dimension which corresponds to rotation by some angle α. In [13], this space of rotations is represented using unit split quaternions (this is the hyperbolic analog of the fact that isometries of $\mathbb{S}^2$ are represented using quaternions, which is well known in computer graphics). The set of unit split quaternions corresponds exactly to $ad\mathbb{S}^d$ – the only difference is that in [13] the dimension corresponding to rotation is considered spacelike, while in $ad\mathbb{S}^d$ it is timelike; specifically, rotation by angle α corresponds to M^α.

In the universal cover, we consider the isometries whose rotation components are described by different angles $\alpha, \beta \in \mathbb{R}$ to be different, even if they are actually the same rotation (that is, $\alpha - \beta$ is a multiple of 2π). We have a natural projection of the universal cover to the underlying space $\pi : \widetilde{ad\mathbb{S}^d} \to ad\mathbb{S}^d$. The space $\widetilde{ad\mathbb{S}^d}$ has its origin O', $\pi(O') = O$. We pick a lift $j : ad\mathbb{S}^d \to \widetilde{ad\mathbb{S}^d}$ such that $\pi \circ j$ is identity; the point $j(x)$ is chosen among the points x' such that $\pi(x') = x$ in a natural way: the path from x' to O' does not cross $\pi^{-1}\{x \in ad\mathbb{S}^(d) : x_1 < 0, x_2 = 0\}$. We reuse the same notation M^α, $R^\alpha_{i,j}$ and $L^\alpha_{i,j}$ for the isometries of $\widetilde{ad\mathbb{S}^d}$. Note that the pass-of-time isometry M^α (as well as the underlying $R^\alpha_{2,3}$) will now be different for every $\alpha \in \mathbb{R}$, while in $ad\mathbb{S}^d$ we had $M^{2\pi} = M^0 = Id$. If T is an isometry of $\widetilde{ad\mathbb{S}^d}$, we likewise have uniquely defined $\pi(T)$, which is an isometry of $ad\mathbb{S}^d$ such that $\pi(Tx) = \pi(T)(\pi(x))$. If T is an isometry of $ad\mathbb{S}^d$, let $j(T)$ be the isometry of $\widetilde{ad\mathbb{S}^d}$ which takes O to $j(T(O))$ and such that $\pi(j(T)) = T$.

In our simulation engine, the points of $\widetilde{G}$, and equivalently $\widetilde{ad\mathbb{S}^d}$ are represented as *shift points*. A shift point (x, h) consists of a point $x \in ad\mathbb{S}^d$ and a shift $h \in \mathbb{R}$, and represents $x' = M^h j(x)$. Note that this representation is not unique; the *canonical* representation of x' is the one in which $x_3 = 0$ and $x_4 > 0$. Similarly, a isometry T' of $\widetilde{ad\mathbb{S}^d}$ is represented as *shift matrices*: (T, h), where

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63749-0_10

T is an isometry matrix and h is a shift, represents $M^h j(T)$. The canonical representation of T' is the one in which (TO, h) is canonical.

To avoid the numerical precision issues, we tessellate $\widetilde{ad\mathbb{S}^d}$. That is, $\widetilde{ad\mathbb{S}^d}$ is subdivided into a number of tiles τ, and every object is described not by a shift matrix relative to some origin of the whole space, but by a tile τ it lives in, and a shift matrix (x, d) relative to the center of the tile τ. As previously mentioned, $\widetilde{ad\mathbb{S}^2}$ corresponds to the Lie group of isometries of $\mathbb{H}^2$; the centers of our tiles correspond to the isometries which map the order-3 heptagonal tessellation (Figure 1) to itself. Knowing the shift matrices transforming the coordinates relative to tile τ_1 into the coordinates relative to adjacent tile τ_2, it is straightforward to compute the coordinates of all the object nearby to the player, relative to the player. The tessellations themselves can be computed using discrete methods such as automata theory [6,3], thus avoiding numerical precision issues arising otherwise when the player makes their ship travel a large distance from the start.

Due to the nature of time-like geodesics in $\widetilde{ad\mathbb{S}^d}$, objects appear to move in circles. This suggests a game design somewhat reminiscent of the classic arcade multidirectional shooter game *Asteroids* (which took place in a space with torus topology, so the objects also did not escape the playing area). The player has to shoot rocks for resources, such as gold (increasing score), health (used up when hit by an asteroid), ammo (used up when shot), fuel (used up when accelerating), and oxygen (used up proportionally to proper time elapsed). These are standard resources well-known to gamers. One interesting consequence of relativity is that the player may use up the fuel resource to save a bit of the oxygen resource, since acceleration can be used to reduce the proper time necessary to reach another point in the spacetime (similar to the twin paradox). Shooting the missile creates a new object m (missile); we compute if the worldline of the missile m intersects the world line of some rock r, and if so, the life of both m and r end at this time, and we create a collectible resource at the same spacetime event.

The implementation of such a simulation requires us to implement the necessary operations for shift points and shift matrices: compose isometries (multiply shift matrices), apply isometry to a point (multiply shift matrix by a shift point), find t_b and x_b for rendering objects. While in $ad\mathbb{S}^d$ we would just use the well-known matrix and vector multiplications, in $\widetilde{ad\mathbb{S}^d}$ these operations are somewhat more involved due to the necessity of computing shifts correctly (we need to be careful to obtain the correct shift value h instead of, e.g., $h \pm 2\pi$). To keep the paper short, we do not include the full formulas in this paper; they can be found in the source code of our simulation (file `math.cpp`).

5 Simulation of the de Sitter spacetime

The general ideas of our de Sitter simulation are similar to the anti-de Sitter case described in the previous section, so we only list the differences.

The origin is now $O = 0[1/d + 1]$. The slice corresponding to current time is $S = \{x \in d\mathbb{S}^d : x_{d+2} = 0\}$. It is isometric to $\mathbb{S}^d$, so it can be rendered e.g. in

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63749-0_10

the stereographic model. The pass of time is now represented using isometry $M^\alpha = L^\alpha_{d+1,d+2}$ (we use the same isometry for the pass of time for s and the other objects). If the player accelerates, we multiply T_{s,t_2} by $L^\alpha_{i,d+2}$.

The equivalence of S to $\mathbb{S}^d$ might suggest *Asteroids*-like design again: rocks flying around the sphere. However, the spacetime $d\mathbb{S}^d$ works very differently: the space appears to be expanding as time passes, and objects which fly too far away from s are impossible to reach anymore. In particular, the other side of the sphere is unreachable. If the game started with a number of randomly pre-placed objects at time 0, after some time all of them would depressingly fly away from each other, with at most one of them remaining in the part of universe reachable to the player. So the world of our de Sitter game is constructed differently. There is a main star (black-and-yellow in Figure 2), and the goal is to remain close to the main star as long as possible. Other objects in the universe make this task harder. Every object (bullet) b gets close to the main star at some time t_b; avoiding being hit is the main challenge of the game, thus making the game an example of a game in the *bullet hell* genre. As usual in bullet hell games, the bullets arrive in waves (sharing similar value of t_b) arranged in various patterns.

Numerical precision issues caused by the space expanding exponentially also arise in $d\mathbb{S}^d$. This time, the solution we use in our simulation is not generally applicable, but rather tailored to the design of our game, that is, based on the assumption that the player will be always forced to remain close to the main star. We again use the concept of shift matrices: the shift matrix describing an object b will be of form (T, t_b), which is equivalent to matrix $M^{t_b}T$ relative to the main star at time 0, or equivalently, T relative to the main star at time t_b. Objects need not be rendered if their t_b is not close enough to the proper time of the main star currently visible to the player (if we tried to render them, we would run into precision issues due to the exponential growth of sinh and cosh).

In the anti-de Sitter case we assumed that every point of the object b, $x \in X(b) \subseteq S$, moves geodesically. This does not work in $d\mathbb{S}^d$ – if every point moved geodesically, the objects would expand. Instead, we only assume that the center O of the object moves geodesically. Instead, after computing t_b^O and x_b^O for the center O ($x_b = T_{s,t}^{-1} T_{b,0} M^{t_b} O \in S$), for every other point $x \in X(b)$ we find a geodesic γ'_x which is correct at time t_b using the formula $\gamma'_x(t_b + t) = T_{b,0} M_{t_b} L^{x_1}_{1,4} L^{x_2}_{2,4}$ where (x_1, x_2) are the coordinates of point x, and find x'_b and t'_b such that $x'_b = T^{-1} s, t\gamma'_x(t'_b)$. The point is now rendered at x'_b. We compute x'_b for every vertex of the polygonal model describing the object b, and render b as a polygon with vertices obtained by mapping x'_b using the chosen projection.

6 Visualizations and Insights

Science-based games should allow the player to not only experience the challenge provided by the gameplay, but also play with the simulation. One aspect of this is that the player can change the parameters of the game (amount of resources, rocks, scale, speed, etc.) to explore more diverse scenarios without being bothered with the challenge, making the game accessible to players who are not skilled at

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63749-0_10

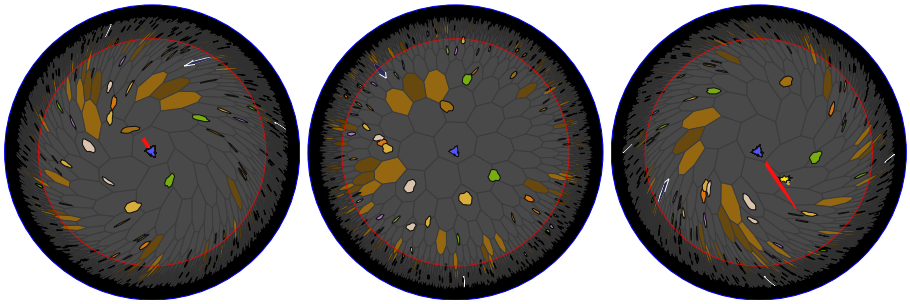

Fig. 3. Anti-de Sitter spacetime: past light cone view (left), present (middle), and future light cone view (right). Note the stretched missile in the future light cone view. Taken from a replay; the red circle is the boundary of the light cone relative to a future position of the ship. Poincaré disk model.

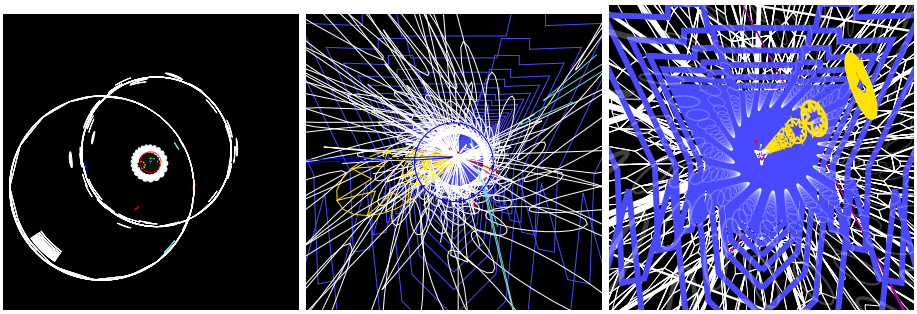

Fig. 4. De Sitter spacetime: present in stereographic projection (left), spacetime view (center), and $\mathbb{H}^3$ view (right). Unfortunately, the spacetime and $\mathbb{H}^3$ views are difficult to capture in a still image (and also in video due to the compression).

the given game genre. Another aspect is that we should keep the history of every object in the game. Such history keeping is necessary by the basic construction of a relativistic game – for example, even if the game knows that some object b disappears at proper time t (for example, a rock is hit by player's missile), and the player has seen that, the player might accelerate, causing them to see the object b still existing, due to how Lorentz transformations work. While it would be safe to remove b from memory if the life of b ended in the past light cone, for the discovery purposes it is better to just remember everything. Since the whole history is kept, the player can replay their game, possibly from other frame of reference. While viewing such a replay (or possibly even during the actual game), the player can change aspects of how the in-game universe is visualized, such as:

– In $\widetilde{ad\mathbb{S}^d}$, let the time α pass according to $R^{\alpha}_{d+1,d+2}$ or M^{α}. The first option is generally more easier to understand (the camera is not rotating), but

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63749-0_10

both views are interesting. In the second view, the world appears to stop spinning around the player, but the movement starts looking like wrapping the spacetime, which is interesting but unintuitive.
- The player can choose to use the current S as explained so far, or they can take S to be the boundary of the past light cone (corresponding to the events the ship would be actually seeing at that precise moment), or the boundary of the future light cone (Figure 3).
- Proper time of every object could be displayed, to let the player see whether the relativistic effects (time dilation) work as they expect.
- There is also an option to view the 2+1-dimensional spacetime using the perspective projection. In this visualization, it is assumed that light travels along geodesics of all kinds. We see a circle of radius 1, the interior of that circle corresponds to time-like geodesics (space-time events the ship could reach), and the exterior corresponds to events happening elsewhere. The circle is essentially a hyperbolic plane in the Beltrami-Klein model; changing the ship's velocity transforms the interior of the circle according to the isometries of this hyperbolic plane. See Figure 4 for an example.
- Interestingly, the isometries of $d\mathbb{S}^d$ are also the isometries of $\mathbb{H}^3$. Both of them live in $\mathbb{E}^\sigma$ for $\sigma = (+^3, -)$, but $\mathbb{H}^3$ is the set of points x where $g_\sigma(x, x) = -1$, while $d\mathbb{S}^2$ is the set of points x where $g_\sigma(x, x) = 1$. This observation leads to another visualization: take the isometry describing the ship's view, and use the same isometry to view $\mathbb{H}^3$ using Klein-Beltrami ball model, or equivalently, perspective. Such a dual view has interesting properties (pass of time corresponds to moving the dual camera forward; accelerating corresponds to moving the dual camera sidewise; points of $d\mathbb{S}^d$ are represented in this perspective as points outside of the Klein-Beltrami ball, in particular points of S being represented as points in infinity; etc.). See Figure 4 for an example.

The following insights into de Sitter and anti-de Sitter spaces are gained.

- The space in the anti-de Sitter game appears to be rotating. This is because the fixed parts of the map (the walls based on the tessellation) move along the geodesics $M^{t_1}v$, while the progress of time is modelled by multiplying the spacetime coordinates $T^t_{3,4}$. So, if $v \in S$ and the player moves by t units, the obbject is displayed at (taking $t = t_1$) $T^{-t}_{3,4}M^t v = R^{1,2}(t)$. An object cannot simply stay in a fixed place v on screen, because that would not be geodesic movement: the time interval between $v \in S$ and $T^t_{3,4}v$ is greater and greater when we increase the distance from v to O, and thus timelike geodesics are pulled towards the center. So in general objects appear to be orbiting around the ship in circular or elliptical orbits.
- In the de Sitter game, if we imagine S as a sphere with the main star at the north pole N, objects moving along geodesics will get close to N at some point, then escape towards the equator. Similarly, if we reversed the time, we discover they started towards the equator too. If we view the spacetime relative to the main star, the equator is well visible due to all the objects close to it. Normally we view the spacetime relative to the ship, which is a

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63749-0_10

different frame of reference, so the two equators are distinct. Figure 2 shows one of them, while Figure 4 shows the equator circle made by bullets we have avoided in the past, the equator circle made by bullets we will have to avoid in the future, and also a small circle of bullets we are avoiding right at the moment when the screenshot was taken.

- Another observation is the Lorentz contraction. All objects appear compressed in the direction of the movement. For example, while all the tessellation tiles are of regular heptagonal shape and the Poincaré model is conformal, but they appear more and more narrow as we move further away from the center (Figure 2). Due to the relationship between split quaternions and the underlying hyperbolic plane, the distance between tiles in $\widetilde{ad\mathbb{S}^2}$ are half the distances between the respective tiles in $\mathbb{H}^2$; after switching the view to Beltrami-Klein model, the heptagons look regular again.
- Dilation of time is best observed in the de Sitter game, by comparing the proper time of the ship with the score, which is the proper time of the main star we are close to. Generally the score will be smaller, due to the geodesic movement of the main star, and non-geodesic movement of the ship. Interestingly, when the player loses the de Sitter game by getting too far away from the main star, their score no longer increases – the main star is escaping so fast that its proper time seen by us does not change.

7 Further work

The engine described in this paper can be extended to other games and experiments in $\widetilde{ad\mathbb{S}^2}$ and $d\mathbb{S}^2$. To conclude the paper, we describe some extensions.

- An active enemy that attempts to predict where the player's ship is going to go (for example, assuming that the ship is simply going along a time-like geodesic), and shoot there. Such enemies are popular in video games. This concept would be naturally more interesting in a relativistic game, since such an enemy would have to predict the player's ship position based on the past (that is, if the enemy shoots at time t, it only sees what the player did at some time in the past, due to the limited speed of light). This is still deterministic. Non-deterministic enemies are of course also a possibility. However, interestingly, making the game multi-player introduces an unexpected challenge, at least if we want the proper time of both players proportional to the real time elapsed. One interesting way to avoid this issue would be to make the game end in the case when causality is lost, i.e., one of the players falls inside the past light cone of another player.
- A map editor to let the players construct their own scenarios and puzzles, for example, to explore the twin paradox (using fuel to conserve oxygen).
- We have chosen the game to have two spatial dimensions. Two-dimensional games generally showcase the experimental gameplay better (multi-directional shooters and bullet hell games work better in two dimensions), and also allow our three-dimensional visualizations of the space-time. However, immersive

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63749-0_10

three-dimensional variants of our games would also be worthwhile. Our implementation of $\widetilde{ad\mathbb{S}^2}$ crucially depends on two-dimensionality, in particular, the trick of using M^α to pass time while keeping all the world stable, due to all dimensions being rotated, works only if the number of space dimensions is even. In three dimensions, one of the dimensions would not be rotated, and thus not stable. A simple way to add another dimension, keeping our tessellations of $\mathbb{H}^2$ and interpretation of M^α, would be to make our universe still roughly two-dimensional (a bit like real-world galaxies), just adding one extra dimension of our space that extrudes the tessellation and is not rotated.

Acknowledgments. This work has been supported by the National Science Centre, Poland, grant UMO-2019/35/B/ST6/04456.

Disclosure of Interests. Authors have no conflict of interest to declare.

References

1. Cannon, J.W., Floyd, W.J., Kenyon, R., Walter, Parry, R.: Hyperbolic geometry. In: In Flavors of geometry. pp. 59–115. University Press (1997), available online at `http://www.msri.org/communications/books/Book31/files/cannon.pdf`
2. Celińska, D., Kopczyński, E.: Programming languages in github: A visualization in hyperbolic plane. In: *Proceedings of the Eleventh International Conference on Web and Social Media,* ICWSM, Montréal, *Québec,* Canada, May 15-18, 2017. pp. 727–728. The AAAI Press, Palo Alto, California (2017), `https://aaai.org/ocs/index.php/ICWSM/ICWSM17/paper/view/15583`
3. Celińska-Kopczyńska, D., Kopczyński, E.: Generating tree structures for hyperbolic tessellations (2021)
4. CodeParade: Hyperbolica (2022)
5. COXETER, H.S.M.: Non-Euclidean Geometry. Mathematical Association of America, 1 edn. (1998), `http://www.jstor.org/stable/10.4169/j.ctt13x0n7c`
6. Epstein, D.B.A., Paterson, M.S., Cannon, J.W., Holt, D.F., Levy, S.V., Thurston, W.P.: Word Processing in Groups. A. K. Peters, Ltd., USA (1992)
7. Griffiths, J.B., Podolský, J.: Exact Space-Times in Einstein's General Relativity. Cambridge Monographs on Mathematical Physics, Cambridge University Press (2009)
8. Hall, A.: Velocity raptor (2011), `https://testtubegames.com/velocityraptor.html` (accessed April 9, 2024)
9. Hamilton, L., Moitra, A.: A no-go theorem for robust acceleration in the hyperbolic plane. In: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual. pp. 3914–3924 (2021), `https://proceedings.neurips.cc/paper/2021/hash/201d546992726352471cfea6b0df0a48-Abstract.html`
10. Hart, V., Hawksley, A., Matsumoto, E.A., Segerman, H.: Non-euclidean virtual reality I: explorations of $\mathbb{H}^3$. In: Proceedings of Bridges: Mathematics, Music, Art, Architecture, Culture. pp. 33–40. Tessellations Publishing, Phoenix, Arizona (2017)
11. Kopczyński, E., Celińska, D., Čtrnáct, M.: HyperRogue: Playing with hyperbolic geometry. In: Proceedings of Bridges *: Mathematics, Art, Music, Architecture, Education, Culture*. pp. 9–16. Tessellations Publishing, Phoenix, Arizona (2017)

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63749-0_10

12. Kopczyński, E., Celińska-Kopczyńska, D.: RogueViz: non-Euclidean geometry engine for visualizations, games, math art, and research (Oct 2023), https://github.com/zenorogue/hyperrogue/, https://github.com/zenorogue/hyperrogue/
13. Kopczyński, E., Celińska-Kopczyńska, D.: Real-time visualization in anisotropic geometries. Experimental Mathematics **0**(0), 1–20 (2022). https://doi.org/10.1080/10586458.2022.2050324, https://doi.org/10.1080/10586458.2022.2050324
14. Kortemeyer, G., Tan, P., Schirra, S.: A slower speed of light: Developing intuition about special relativity with games. In: International Conference on Foundations of Digital Games (2013)
15. Maldacena, J.: The large-n limit of superconformal field theories and supergravity. International Journal of Theoretical Physics **38**(4), 1113–1133 (Apr 1999). https://doi.org/10.1023/A:1026654312961, https://doi.org/10.1023/A:1026654312961
16. Medved, A.J.M.: How not to construct an asymptotically de sitter universe. Classical and Quantum Gravity **19**(17), 4511 (aug 2002). https://doi.org/10.1088/0264-9381/19/17/303, https://dx.doi.org/10.1088/0264-9381/19/17/303
17. Osudin, D., Child, C., He, Y.H.: Rendering non-euclidean space in real-time using spherical and hyperbolic trigonometry. In: Rodrigues, J.M.F., Cardoso, P.J.S., Monteiro, J., Lam, R., Krzhizhanovskaya, V.V., Lees, M.H., Dongarra, J.J., Sloot, P.M. (eds.) Computational Science – ICCS 2019. pp. 543–550. Springer International Publishing, Cham (2019)
18. Phillips, M., Gunn, C.: Visualizing hyperbolic space: Unusual uses of 4x4 matrices. In: Proc. I3D. pp. 209–214. Association for Computing Machinery, New York, NY, USA (1992). https://doi.org/10.1145/147156.147206, https://doi.org/10.1145/147156.147206
19. Reynolds, C.W.: Flocks, herds and schools: A distributed behavioral model. In: Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques. p. 25–34. SIGGRAPH '87, Association for Computing Machinery, New York, NY, USA (1987). https://doi.org/10.1145/37401.37406, https://doi.org/10.1145/37401.37406
20. Seskir, Z.C., Migdał, P., Weidner, C., Anupam, A., Case, N., Davis, N., Decaroli, C., İlke Ercan, Foti, C., Gora, P., Jankiewicz, K., Cour, B.R.L., Malo, J.Y., Maniscalco, S., Naeemi, A., Nita, L., Parvin, N., Scafirimuto, F., Sherson, J.F., Surer, E., Wootton, J.R., Yeh, L., Zabello, O., Chiofalo, M.: Quantum games and interactive tools for quantum technologies outreach and education. Optical Engineering **61**(8), 081809 (2022). https://doi.org/10.1117/1.OE.61.8.081809, https://doi.org/10.1117/1.OE.61.8.081809
21. Snyder, J.: Flattening the Earth: Two Thousand Years of Map Projections. University of Chicago Press (1997), https://books.google.pl/books?id=0UzjTJ4w9yEC
22. Squad: Kerbal space program – create and manage your own space program (2022), https://www.kerbalspaceprogram.com/ (accessed April 9, 2024)
23. Weeks, J.: Real-time rendering in curved spaces. IEEE Computer Graphics and Applications **22**(6), 90–99 (Nov 2002). https://doi.org/10.1109/MCG.2002.1046633, https://doi.org/10.1109/MCG.2002.1046633
24. Weeks, J.: Geometry games (2009-2021), https://www.geometrygames.org/HyperbolicGames/ (accessed April 9, 2024)
25. Weeks, J.: Non-euclidean billiards in vr. In: Yackel, C., Bosch, R., Torrence, E., Fenyvesi, K. (eds.) Proceedings of Bridges 2020: Mathematics, Art, Music, Architecture, Education, Culture. pp. 1–8. Tessellations Publishing, Phoenix, Arizona (2020)

ICCS Camera Ready Version 2024
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-63749-0_10