Non-euclidean Virtual Reality I: Explorations of $\mathbb{H}^3$

Vi Hart
eleVR
HARC

Andrea Hawksley
eleVR
HARC

Elisabetta A. Matsumoto
School of Physics
Georgia Institute of Technology

Henry Segerman
Department of Mathematics
Oklahoma State University

Abstract

We describe our initial explorations in simulating non-euclidean geometries in virtual reality. Our simulations of three-dimensional hyperbolic space are available at h3.hypernom.com. The code is available at github.com/hawksley/hypVR.

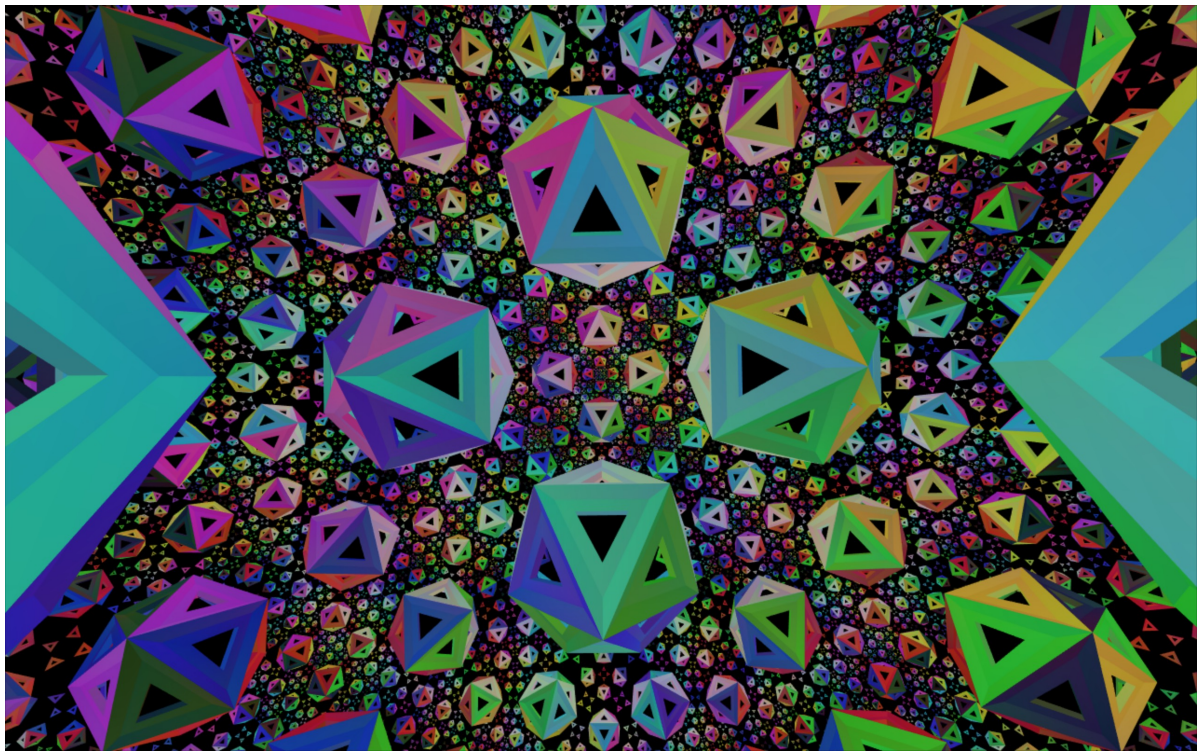
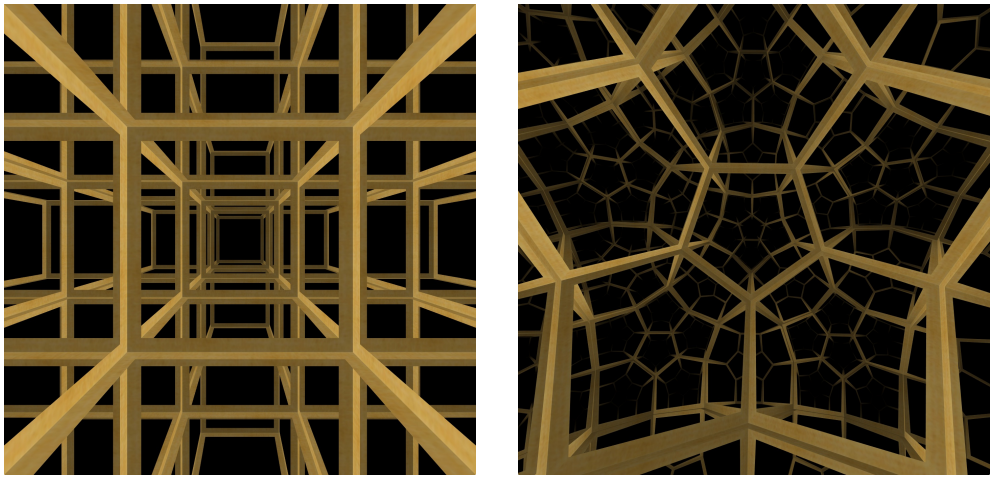


Figure 1: A view from $\mathbb{H}^3$.

The properties of euclidean space seem natural and obvious to us, to the point that it took mathematicians over two thousand years to see an alternative to Euclid's parallel postulate. The eventual discovery of hyperbolic geometry in the 19th century shook our assumptions, revealing just how strongly our native experience of the world blinded us from consistent alternatives, even in a field that many see as purely theoretical. Non-euclidean spaces are still seen as unintuitive and exotic, but we believe that with direct immersive experience we can get a better "feel" for them. The latest wave of virtual reality hardware, in particular the HTC Vive, tracks both the orientation and the position of the headset within a room-sized volume, allowing for such an experience.

Most visualisations of hyperbolic space are seen from the outside, as in Escher's *Circle Limit* series of prints, which use the Poincaré disk model of two-dimensional hyperbolic space, $\mathbb{H}^2$. Three-dimensional hyperbolic space, $\mathbb{H}^3$, can also be visualised in a similar way, via the Poincaré ball model. In virtual reality, we could simulate this ball model floating in the middle of the room. We would then generate graphics on



(a) The three torus, giving the $\{4, 3, 4\}$ honeycomb. (b) The $\{5, 3, 4\}$ honeycomb.

Figure 2: Screenshots from *Curved spaces* by Jeff Weeks.

screen using the standard euclidean graphics pipeline, and motions in real life would translate directly to motions of the virtual camera in the ambient three-dimensional euclidean space, $\mathbb{E}^3$, of the Poincaré ball model. Even though you would be able to put your head inside of this virtual Poincaré ball model, it would give an *extrinsic* experience of $\mathbb{H}^3$ – you would experience the metric of the Poincaré model induced from $\mathbb{E}^3$, and not directly experience the metric of $\mathbb{H}^3$. Such extrinsic visualisations provide a brief, compact snapshot of infinite hyperbolic space, yet the viewer is left to their own imagination to remodel the space to see what life might be like for an inhabitant living inside. Our goal is to make three-dimensional non-euclidean spaces feel more natural by giving people experiences inside those spaces, including the ability to move through those spaces with their bodies. Luckily, computers don't know or care that people live in a mostly euclidean world, a world where cubes fit together four around an edge because they have 90° angles. As long as we program in the correct mathematics, a computer is perfectly happy simulating a hyperbolic space where cubes pack neatly, six around an edge.

We took inspiration from Jeff Weeks' *Curved Spaces* [5] software, a "flight simulator for multiconnected universes", which allows the user to "fly" a spaceship through various three-dimensional manifolds, with spherical, euclidean, and hyperbolic geometries (see Figure 2). The user controls the heading of the spaceship using the mouse, and its speed using keyboard controls. With virtual reality technology however, the user controls the direction in which they are looking by turning their head, and their position by moving their body. Thus, we remove barriers between us and the space – it is easier and more natural to access the experience, particularly for users who are not familiar with moving through space using "computer game" controls, and this extra ease allows a user to discover some not-so-obvious properties of these spaces much more readily.

We are currently developing a virtual reality simulation of $\mathbb{H}^3$, using many of the same ideas as are used in Weeks' work. Weeks explains the implementation in detail in [6]; we give an overview in this paper. Positional tracking in modern virtual reality headsets lets us experience features of hyperbolic space, such as the effects of parallel transport, geodesic deviation and holonomy, in a very direct way.

There are four ingredients that go into our virtual reality simulation of $\mathbb{H}^3$ as outlined in this paper:

1. A way to describe the points of $\mathbb{H}^3$ numerically, i.e. a *model* of $\mathbb{H}^3$
2. A way to convert points in the model into points in $\mathbb{E}^3$ that we can then draw on screen,
3. A way to move around $\mathbb{H}^3$ using the motion inputs from the virtual reality headset, and
4. A set of landmarks in $\mathbb{H}^3$ to draw, to help the viewer navigate the space – we use a tiling of $\mathbb{H}^3$.

1 The Model of $\mathbb{H}^3$

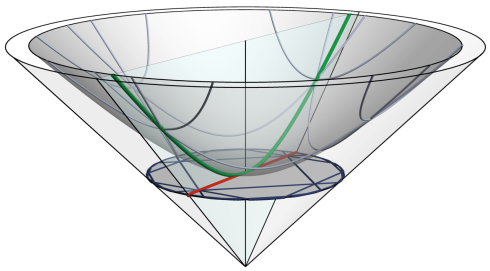


Figure 3: The hyperboloid and Klein models of $\mathbb{H}^2$. Projecting the hyperboloid towards the origin onto the plane $w = 1$ results in the Klein model.

For the first ingredient, there are many different models of hyperbolic space, including the Poincaré disk model, the upper half plane model, the Klein model, and the hyperboloid model. Compared to the other commonly seen models, the hyperboloid model is less easy to use for direct visualisation, but it turns out to be very useful for calculation. The hyperboloid model of $\mathbb{H}^2$ is the set of points $\{(x, y, w) \in \mathbb{E}^{2,1} \mid x^2 + y^2 = w^2 - 1, w > 0\}$, where $\mathbb{E}^{2,1}$ is Minkowski space with two space-like directions, x, y, and one time-like direction, w. Three-dimensional Minkowski space $\mathbb{E}^{2,1}$ has the same cartesian coordinate system as three-dimensional euclidean space, $\mathbb{E}^3$, but comes equipped with a different metric. The *metric* g_{ij} is a function that generalises the method of computing distances and angles (i.e. the dot product in euclidean space).

With the metric induced from the Minkowski space it lives in, the hyperboloid has constant gaussian curvature -1, i.e. it is a model for the hyperbolic plane. For each point (x, y, w) of the hyperboloid, we can divide the coordinates by w to obtain $(x/w, y/w, 1)$. This maps the hyperboloid to the unit radius disk on the $w = 1$ plane. The result is the Klein model of $\mathbb{H}^2$ (see Figure 3). Geodesics in the hyperboloid model of $\mathbb{H}^2$ are intersections of the hyperboloid with planes in $\mathbb{E}^{2,1}$ that pass through $(0, 0, 0)$. These geodesics map to the Klein model of $\mathbb{H}^2$ as straight lines (in the euclidean sense).

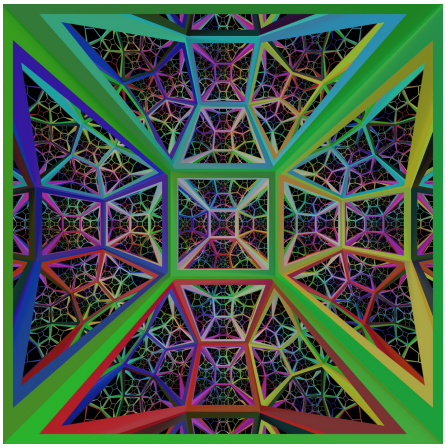
Three-dimensional hyperbolic space $\mathbb{H}^3$, and indeed higher dimensional hyperbolic spaces, can be modeled analogously to $\mathbb{H}^2$, but in higher dimensional ambient spaces. The generalised hyperboloid model for $d-$dimensional hyperbolic space $\mathbb{H}^d$ is the set of points in Minkowski space with d space-like directions, $x_1, ..., x_d$, and one time-like direction w, given by $\{(x_1, x_2, ..., x_d, w) \in \mathbb{E}^{d,1} \mid \sum_{n=1}^{d} x_n^2 = w^2 - 1, w > 0\}$.

2 Drawing Points in $\mathbb{H}^3$ on Screen

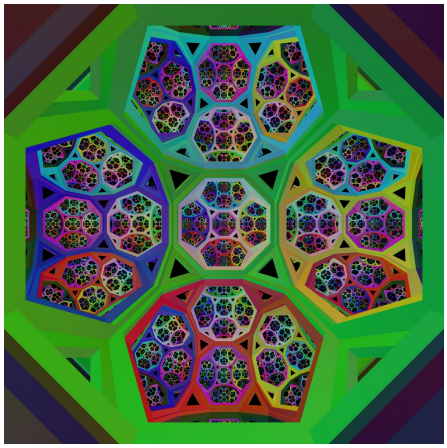
In order to draw a point of $\mathbb{H}^3$ on the screen, we need to understand the relationship between the location of the point on the hyperboloid and us, the viewer, situated at the *origin* of the hyperboloid, $(0, 0, 0, 1) \in \mathbb{E}^{3,1}$. We are not actually viewing points in $\mathbb{H}^3$, but we view their image in the *tangent space* at the origin – a copy of $\mathbb{E}^3$ consisting of the tangent vectors of the hyperboloid at the origin. A point $p_{\mathbb{H}^3} \in \mathbb{H}^3$ is connected to the origin by a parametrised geodesic $\boldsymbol{\gamma}(t)$ that leaves the origin at $t = 0$ and intercepts $p_{\mathbb{H}^3}$ at $t = 1$. Our view of the same point $p_{\mathbb{E}^3} \in \mathbb{E}^3$ should also be connected to us via a geodesic in $\mathbb{E}^3$ (i.e. a straight line). The velocity of the geodesic in $\mathbb{H}^3$ at the origin $\dot{\boldsymbol{\gamma}}(0)$, tells us where to find $p_{\mathbb{E}^3}$ – its direction is the direction in which we must look to find $p_{\mathbb{E}^3}$, and its magnitude indicates the distance between us (situated at the origin of the hyperboloid) and $p_{\mathbb{E}^3}$. The map we have described, taking points on the hyperboloid to points in $\mathbb{R}^3$, is the inverse of the *(riemannian geometry) exponential map*. The exponential map goes in the other direction, sending points in the tangent space of the hyperboloid at our location into the hyperboloid.

The correct thing to do is to use the inverse of the exponential map to draw points on screen, but we don't actually need to compute the absolute distance a point is from us. We merely need to know the relative distance between two points in a given direction, so that nearer objects occlude those further away from us. The Klein model is computationally cheaper to calculate – as it does not involve inverse hyperbolic trigonometric functions – so we (and Weeks [5]) use it. Figure 4 shows a number of views of a honeycomb in (i.e. a tiling of) $\mathbb{H}^3$ drawn using this algorithm[1] [4].

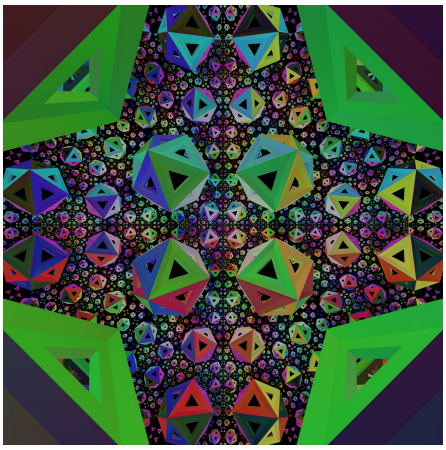
[1]Note, that virtual reality implementations use separate renderings for the left and right eyes to give a three-dimensional view of the environment (see further Section 6 of Ref. [1]).



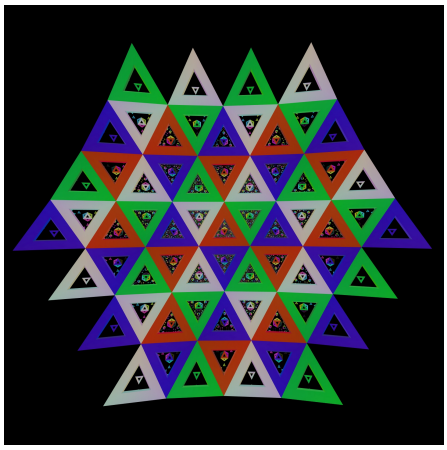
(a) Cubes.



(b) Truncated cubes.



(c) Only the triangular faces of the truncated cubes.



(d) The view from inside the "polyhedron" in the centre lower left of Figure 4c.

Figure 4: Views of the $\{4, 3, 6\}$ honeycomb. We draw the honeycomb out to a depth of six steps from the central cube.

3 Moving Through $\mathbb{H}^3$

At each timestep of our simulation, the virtual reality headset (the HTC Vive) outputs both position and orientation information corresponding to the location of your head in the room and the direction in which you are looking. We can use this information in several ways, depending on what kind of experience we wish to create.[2]

There is freedom in how to map the headset data of the user's motion in the room into the hyperboloid model. One possibility would be to map the position and orientation data onto the Poincaré ball model, and show the in-space view of $\mathbb{H}^3$ from that position and orientation. Although, this might not be the most obvious choice, it would allow us to map an infinite space into the finite confines of a room in $\mathbb{E}^3$. Unfortunately, this sort of mapping violates a property of movement that humans are quite attached to, which is consistency in distance. The user would find that two motions of their head of the same magnitude would correspond to

[2]In the vast majority of virtual reality experiences, the position and orientation of the headset are mapped directly to the position and orientation of a virtual camera in euclidean space. In our geometry simulations, we map orientation directly but treat position differently. In our previous work *Hypernom* [2], there is no position tracking, but we map the headset's orientation to both the orientation and the position of the virtual camera in the three-dimensional sphere, S^3.

vastly different translations within the simulation, depending on where in the room they are standing.

The approach we take here is to look at the relative motion of the headset, and move the virtual camera in a corresponding way. Every timestep, we compare the headset's current position to its previous position, compute the difference, and move the virtual position by that vector. This has the advantage that your head's motions behave the same way no matter where you are. The disadvantage is that depending on your previous movements, a location in real life might map to any location in the virtual space.

Although the trick of implementing graphics using the Klein model as we have described it only works at the origin, it turns out that we can still use it as we move through the space. As in many computer graphics implementations, we leave the viewer at the origin and translate the world around them to simulate the viewer's movement. The appropriate "translations" for us are isometries of $\mathbb{H}^3$. Infinitesimal translations are given by the generators of the Lie group of the space and finite transformations are given by the *(Lie theory) exponential map*[3]. The isometries of $\mathbb{H}^3$ are isometries of $\mathbb{E}^{3,1}$ which preserve the hyperboloid and its metric. These are elements of the group $\mathrm{SO}(3,1)$. The translation by a vector $d\mathbf{r} = (dx, dy, dz)$ in the tangent space is given by the exponential $\exp(\mathbf{M}) = \sum_{n=0}^{\infty} \frac{1}{n!}\mathbf{M}^n$ of the matrix

$$\mathbf{M} = \begin{pmatrix} 0 & 0 & 0 & dx \\ 0 & 0 & 0 & dy \\ 0 & 0 & 0 & dz \\ dx & dy & dz & 0 \end{pmatrix}.$$

Calculation of the series for the matrix exponential $\exp(\mathbf{M})$ can be vastly simplified due to a trick pointed out by Jeff Weeks. Note that $\mathbf{M}^3 = |d\mathbf{r}|^2\mathbf{M}$, and $\mathbf{M}^4 = |d\mathbf{r}|^2\mathbf{M}^2$, where $|d\mathbf{r}| = \sqrt{dx^2 + dy^2 + dz^2}$. Then the matrix exponential can be split into two sums:

$$\sum_{n=1}^{\infty} \frac{|d\mathbf{r}|^{2n-2}}{(2n-1)!}\mathbf{M} = \frac{\sinh(|d\mathbf{r}|)}{|d\mathbf{r}|}\mathbf{M}, \quad \sum_{n=1}^{\infty} \frac{|d\mathbf{r}|^{2n}}{(2n)!}|d\mathbf{r}|^{2n}\mathbf{M} = \frac{\cosh(|d\mathbf{r}|) - 1}{|d\mathbf{r}|^2}\mathbf{M}.$$

Thus, the exponential map is given by

$$\exp \mathbf{M} = \mathbf{Id} + \frac{\sinh(|d\mathbf{r}|)}{|d\mathbf{r}|}\mathbf{M} + \frac{\cosh(|d\mathbf{r}|) - 1}{|d\mathbf{r}|^2}\mathbf{M}^2.$$

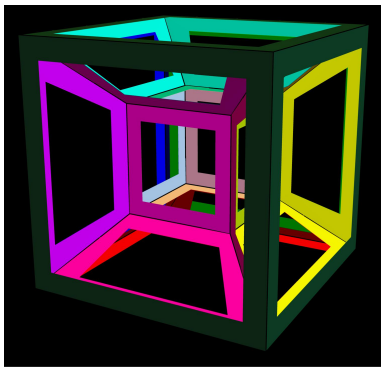
When the user moves their head, the virtual reality headset detects this movement in the three-dimensional euclidean space in which they live. The difference in position between two subsequent frames is some vector, which gives us the translation of the user $-d\mathbf{r}$.[4] We then generate the isometry $\exp(\mathbf{M})$, and apply it to all the points of our simulated world before rendering the next frame. This moves the points of the world in the hyperboloid by isometries, giving the correct sense in which the user moves through the world.

4 Decoration: the $\{4, 3, 6\}$ Honeycomb and its Colouring

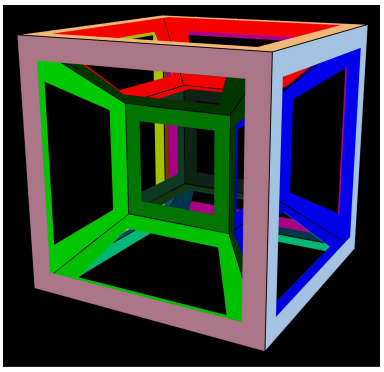
Any three-dimensional manifold can be made by taking a polyhedron and gluing its sides together in some way. Jeff Weeks' *Curved Spaces* shows such a polyhedron for each manifold. For example, Figure 2a shows the view from inside the three-torus, whose geometry is $\mathbb{E}^3$. In this case, the polyhedron is a cube with opposite sides glued. We see a tiling (or honeycomb) of $\mathbb{E}^3$ by cubes – which we get by "unwrapping" the three-torus into space. This tiling has *Schläfli symbol* $\{4, 3, 4\}$, meaning that the faces are squares (with 4

[3]This is similar to the riemannian geometry version of the exponential map, except that instead of converting a tangent vector (an infinitesimal movement in some direction) into a point at the end of a geodesic segment, it converts a more general infinitesimal motion into an isometry.

[4]Note that the sign of $-d\mathbf{r}$ is due to the fact that the sensors detect the displacement of the virtual reality headset as $-d\mathbf{r}$, which corresponds to moving the entire world by a vector with the same magnitude, but in the opposite direction, $d\mathbf{r}$.



(a) Colouring of the hypercube, view 1.



(b) Colouring of the hypercube, view 2.

Figure 5: Any colouring of the hypercube can be mapped onto a colouring of the $\{4,3,6\}$ hyperbolic honeycomb.

sides), the cells are made out of these faces, with 3 around each vertex, and there are 4 cells arranged around each edge. Figure 2b shows a different manifold with a honeycomb in which four dodecahedra meet around each edge. The corresponding Schläfli symbol is $\{5,3,4\}$, meaning that the cells are made out of pentagons (5 sides), with 3 around each vertex, and with 4 cells arranged around each edge.

As our fourth ingredient, we decorate $\mathbb{H}^3$ with another honeycomb of cubes, this time with Schläfli symbol $\{4,3,6\}$ (see Figure 4a). Here we have six cubes around each edge, rather than four. A surprising feature of this honeycomb is that the cubes are no longer of finite size – it turns out that the vertices must be infinitely far away. See [4] for more on this phenomenon.

For the euclidean honeycomb $\{4,3,4\}$ – with four cubes around each edge, if we truncate each of the cubes by cutting off the corners, the revealed triangular faces form an octahedron arranged around each vertex of the original honeycomb. If we do the same thing for our hyperbolic honeycomb, as in Figure 4b, the triangular faces form an infinite tiling – the tiling of the euclidean plane with six triangles around each vertex. In our visualisation, we can experience this directly. This is easiest to see if we remove the edges of the cubes, leaving only the triangular faces, as in Figure 4c. These form strange-looking polyhedra at first sight: one could believe that they are icosahedra, except that the vertex degree is six. If you put your head "into" one of these polyhedra and look back out from the inside, the polyhedron becomes the tiling of the euclidean plane, as we see in Figure 4d. In fact, these polyhedra correspond to *horospheres* in $\mathbb{H}^3$. These are "spheres" centered on points on the boundary of $\mathbb{H}^3$, whose induced metric is the same as that of the euclidean plane – which allows us to draw the regular tiling by equilateral triangles on them seen in Figure 4d.

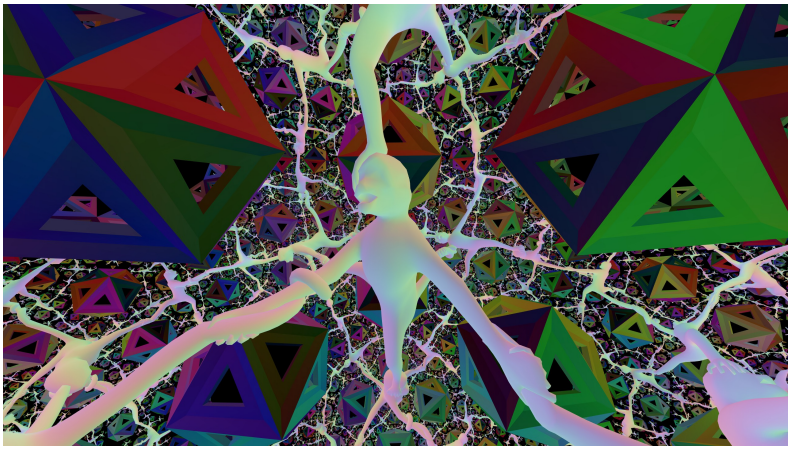


Figure 6: A monkey in each cube of the $\{4,3,6\}$ honeycomb. Note the ring of six monkeys connected together around each edge of the honeycomb.

In Figure 4, we colour the cells using eight colours, in an interesting pattern very special to the $\{4,3,6\}$ honeycomb. This comes from the observation that $\{4,3,6\}$ is a kind of branched cover of the $\{4,3,3\}$ honeycomb, in which three cubes are arranged around each edge. The honeycomb $\{4,3,3\}$ does not tile hyperbolic space; rather it is a honeycomb that tiles spherical space: it is the same as the honeycomb we get by radially projecting the cubical cells of the hypercube onto a circumscribing three-sphere in four-dimensional space. To be more precise, there is a continuous map, F say, from $\{4,3,6\}$ to $\{4,3,3\}$, that maps each cube of $\{4,3,6\}$ to one of the eight cubes of $\{4,3,3\}$. We assign a different colour to each of the eight cubes of $\{4,3,3\}$, as in Figure 5, then colour each cube c of $\{4,3,6\}$ by the colour of $F(c)$. Patterns in the colouring can be seen in Figures 4b and 4c: first that cubes opposite each other around an edge have

the same colour, and second that going in a straight line, from face to opposite face of each cube, we get back to the same colour after four cubes. Any pattern drawn on the hypercube can be lifted to the $\{4, 3, 6\}$ honeycomb. For example, our sculpture, *More fun than a hypercube of monkeys* [3], puts a monkey in each cubical cell of the hypercube. The lift of this sculpture is shown in Figure 6.

5 Virtual Reality and the Consequences of Parallel Transport

The physicality of a virtual reality system with positional tracking gives us a visceral sense of some otherwise abstract phenomena. In a curved space, for example, two neighbouring geodesics that start with parallel velocities (tangent vectors) end up diverging if the space is negatively curved, a phenomenon known as *geodesic deviation*. Suppose that in the simulation, the user is standing on a floor consisting of a geodesic plane in $\mathbb{H}^3$. When they walk forward in real life, in the simulation their head follows a geodesic that starts out with velocity parallel to the floor, which therefore diverges from the floor. This leads to the sensation that the floor is falling out from under your feet (see Figure 7).

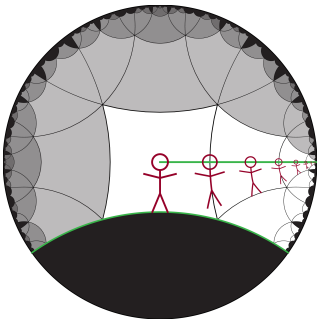
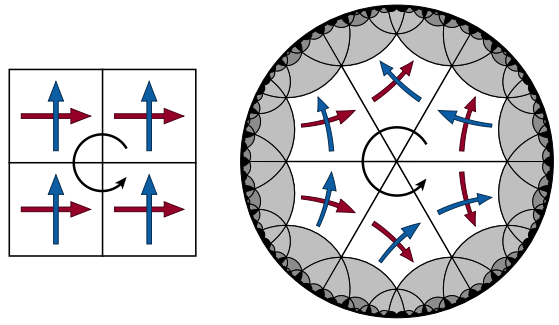


Figure 7: The floor falls out from under your feet as you travel along a geodesic.

When a vector is *parallel transported* along a curve, it moves through space along the curve staying parallel to itself while maintaining a constant magnitude. To move along a manifold following a path in a given direction, we must know how the velocity changes as we move parallel to the path. More formally, this is given by a directional derivative ∇_X along the vector X in the tangent space of the manifold. How might we go about constructing geodesic from this notion? The answer is that a geodesic is a curve that parallel transports its own tangent vector.



(a) Parallel transport in $\mathbb{E}^2$.

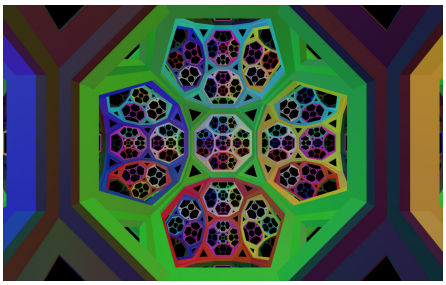
(b) Parallel transport rotates the frame in $\mathbb{H}^2$, shown in the Poincaré disk model.

Figure 8: Walking around an edge.

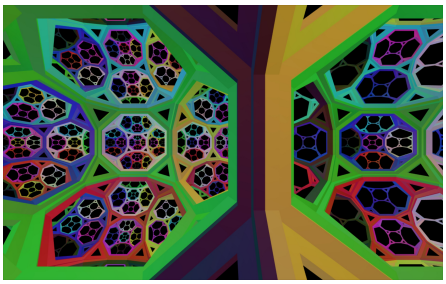
Another, unexpected phenomenon we encounter in the virtual reality experience stems from *holonomy* resulting from the parallel transport of other vectors. When we experience the world, we are aware not only of the vector that points in the direction we are looking but also of the vectors that point up and down, left and right. Moving along a path in the virtual reality $\mathbb{H}^3$ space, these vectors get transported as well. We have a fixed sense of which direction "up" is, but this direction can rotate with respect to the world (see Figures 8 and 9). In particular, this means that certain movements in $\mathbb{E}^3$ produce a rotation of the floor of a room drawn in $\mathbb{H}^3$, so that it no longer appears to coincide with the real-life floor the user is walking on.

These phenomena make $\mathbb{H}^3$ a somewhat confusing place to live in, at least as a visitor from $\mathbb{E}^3$. There may be ways to "hack" the simulation to solve the problems of the virtual floor falling away or rotating away from the real-life floor. To "fix" the angle of the floor changing, we could artificially rotate the virtual view so that the orientation of the virtual camera relative to the virtual floor always agrees with the orientation of the headset relative to the real-life floor. Alternatively, we could avoid both problems by tracking the point directly between the user's feet rather than their head as it moves through space, and for every frame offset the position of the camera up from the feet to the head. These are both somewhat artificial fixes however, and would preclude the user from experiencing the effects of parallel transport, geodesic deviation and holonomy.

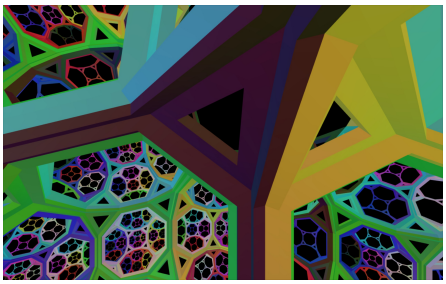
A more natural way to have the floor stay where it should be is to switch from $\mathbb{H}^3$ geometry to the product space $\mathbb{H}^2 \times \mathbb{E}$ – the cartesian product of the hyperbolic plane with the euclidean line. We discuss our simulation of this geometry in our second paper in this volume [1].



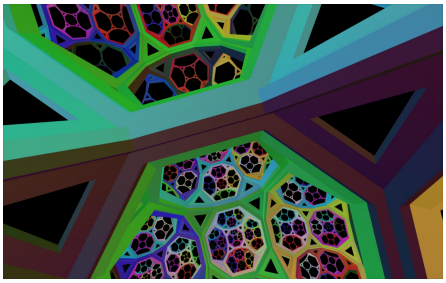
(a) $\mathbb{H}^3$ initial view.



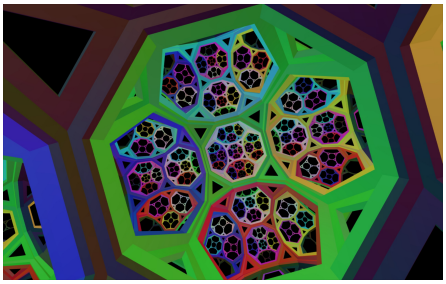
(b) $\mathbb{H}^3$ after moving right 0.5 hyperbolic units.



(c) $\mathbb{H}^3$ after moving up 0.5 hyperbolic units.



(d) $\mathbb{H}^3$ after moving left 0.5 hyperbolic units.



(e) $\mathbb{H}^3$ after moving down 0.5 hyperbolic units.

Figure 9: Holonomy is the rotation of a reference frame after traversing a loop in curved space.

6 Future Directions: Seeing Your Hands

Moving objects in non-euclidean spaces presents several interesting challenges. In addition to the headset, the HTC Vive can also track the position and orientation of hand-held controllers. In most applications that use the controller, a virtual version of the controller is visible in the virtual space. This helps greatly with the user's sense of embodiment in the space, since they can see the positions of their hands. The controller also has various buttons and triggers for other forms of interaction, for example grabbing on to a virtual object near to the virtual position of the controller, allowing the object to be moved in space. We plan to add this kind of interactive element to our simulations.

An obvious way to try to implement a controller would be to track the change in its position from frame to frame, convert that into an isometry, and move the virtual controller by that isometry. However, this would run into problems with geodesic deviation, similar to the floor falling away from the user: as you walk forward, your hand would appear to diverge from your path, sliding off into the distance. Instead, we plan to update the position of the controller each frame as an offset isometry from the position of the headset. With the correct choice of scaling between real life euclidean space and our virtual space, this should mean that the controller appears in a location consistent with the user's sense of proprioception.

References

[1] Vi Hart, Andrea Hawksley, Elisabetta A. Matsumoto, and Henry Segerman. Non-euclidean virtual reality II: explorations of $\mathbb{H}^2 \times \mathbb{E}$. In *Proc. Bridges 2017*. Tessellations Publishing, 2017.

[2] Vi Hart, Andrea Hawksley, Henry Segerman, and Marc ten Bosch. Hypernom: Mapping VR headset orientation to S^3. In *Proc. Bridges 2015*, pages 387–390. Tessellations Publishing, 2015.

[3] Vi Hart and Henry Segerman. The quaternion group as a symmetry group. In *Proc. Bridges 2014*, pages 143–150. Tessellations Publishing, 2014.

[4] Roice Nelson and Henry Segerman. Visualizing hyperbolic honeycombs. *Journal of Mathematics and the Arts*, 11(1):4–39, 2017.

[5] Jeff Weeks. Curved Spaces. a flight simulator for multiconnected universes, available from http://www.geometrygames.org/CurvedSpaces/.

[6] Jeff Weeks. Real-time rendering in curved spaces. *IEEE Computer Graphics and Applications*, 22(6):90–99, 2002.