Rendering non-Euclidean worlds

Meelis Perli
Euclidean worlds

- What we are used to
- No curvature
- Euclid’s postulates apply

NB: Real world is not Euclidean
Euclid’s postulates

1. A straight line segment can be drawn joining any two points.
2. Any straight line segment can be extended indefinitely in a straight line.
3. Given any straight line segment, a circle can be drawn having the segment as radius and one endpoint as center.
4. All right angles are congruent.
5. If two lines are drawn which intersect a third in such a way that the sum of the inner angles on one side is less than two right angles, then the two lines inevitably must intersect each other on that side if extended far enough. This postulate is equivalent to what is known as the parallel postulate.
Simplified

1. A straight line is the shortest possible path between two points

\[ D(\text{OB}) \leq D(\text{OA}) + D(\text{AB}) \]

2. Parallel lines are constant distance from each other

3. The inner angles in triangles always add up to 180°
Which is/are Euclidean?
To make a Non-Euclidean world:

Break at least one of the Euclid's postulates

But how?
To make a Non-Euclidean world:

Break at least one of the Euclid’s postulates

But how?
Spheres: Lines and circles

The shortest path on a sphere is always along a great circle.

Great circle - A circle that cuts the sphere into 2 equal hemispheres.
Spherical space: 2D world on 3D sphere

Steps from start:

1. Move straight to the north pole,
2. Turn yourself 90° right,
3. Move straight back to the equator,
4. Turn 90° right again,
5. Move a quarter of world's circumference straight along the equator line,
6. Turn 90° right again,
Spherical space: 2D world on 3D sphere

Summary:

- You moved along 3 straight lines with equal length
- Your position and rotation are the same as you started with.
- So it must be a triangle

But

- You turned 270° right
Spherical space: Cartesian coordinates

- \((x, y, z)\), where \(x^2 + y^2 + z^2 = r^2\)
- \(r\) can just be 1
- Pros: Good for rotations, rendering
- Cons: Bad for position
Spherical space: Spherical coordinates

- \((r, \theta, \phi), \text{ where } r \geq 0, \ 0 < \theta \leq \pi^\circ, \ 0 < \phi < 2\pi^\circ\)
- \(r\) can just be 1
- Pros: Good for position
- Cons: Can’t be used for rotation, rendering

How to convert to Cartesian coordinates \((x, y, z)\)?
Spherical space: Spherical coordinates

- \((r, \theta, \varphi)\), where \(r \geq 0\), \(0 < \theta \leq \pi\), \(0 < \varphi < 2\pi\)
- \(r\) can just be 1
- Pros: Good for position
- Cons: Can’t be used for rotation, rendering

How to convert to Cartesian coordinates \((x, y, z)\)?

\[ z = r \cos(\theta) \]
Spherical space: Spherical coordinates

- \((r, \theta, \phi)\), where \(r \geq 0\), \(0 < \theta \leq \pi^\circ\), \(0 < \phi < 2\pi^\circ\)
- \(r\) can just be 1
- Pros: Good for position
- Cons: Can’t be used for rotation, rendering

How to convert to Cartesian coordinates \((x, y, z)\)?

\[
\begin{align*}
z &= r \cdot \cos(\theta) \\
y &= r \cdot \sin(\theta) \cdot \cos(\phi)
\end{align*}
\]
Spherical space: Spherical coordinates

- \((r, \theta, \phi), \text{ where } r \geq 0, \ 0 < \theta \leq \pi^\circ, \ 0 < \phi < 2\pi^\circ\)
- \(r\) can just be 1
- Pros: Good for position
- Cons: Can’t be used for rotation, rendering

How to convert to Cartesian coordinates \((x, y, z)\)?

- \(z = r \cdot \cos(\theta)\)
- \(y = r \cdot \sin(\theta) \cdot \cos(\phi)\)
- \(x = r \cdot \sin(\theta) \cdot \sin(\phi)\)
Spherical space: What should it look like?

Pics made from video: https://www.youtube.com/watch?v=iiGe2x8t6mA&t=666s
Spherical space: Movement

Problem: Objects seem to move at different speeds
Spherical space: Rotation with spherical coordinates

Changing $\theta$ doesn’t work  

Changing $\varphi$ works
Spherical space: Rotation matrices for movement

\[ v' = R_{yz}(\phi) * R_{xz}(\theta) * R_{xy}(\Psi) * v \]

\[
R_{yz} = \begin{pmatrix}
1 & 0 & 0 & 0 \\
0 & \cos(\phi) & \sin(\phi) & 0 \\
0 & -\sin(\phi) & \cos(\phi) & 0 \\
0 & 0 & 0 & 1
\end{pmatrix}
\]

\[
R_{xz} = \begin{pmatrix}
\cos(\phi) & 0 & \sin(\phi) & 0 \\
0 & 1 & 0 & 0 \\
-\sin(\phi) & 0 & \cos(\phi) & 0 \\
0 & 0 & 0 & 1
\end{pmatrix}
\]

\[
R_{xy} = \begin{pmatrix}
\cos(\phi) & \sin(\phi) & 0 & 0 \\
-\sin(\phi) & \cos(\phi) & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{pmatrix}
\]
Spherical space: Stereographic projection

- Function, that projects a n-Sphere to n dimensional hyperplane.
- Normal sphere is a 2-Sphere, and it is projected to a 2D plane

\[
[x, y, z] \mapsto \left[ \frac{x}{1-z}, \frac{y}{1-z} \right].
\]

- So a 3-Sphere would be projected to a 3D hyperplane

\[
[x_1, x_2, \ldots, x_n] \mapsto \left[ \frac{x_1}{1-x_n}, \frac{x_2}{1-x_n}, \ldots, \frac{x_{n-1}}{1-x_n} \right]
\]
$[x, y, z] \mapsto \left[ \frac{x}{1-z}, \frac{y}{1-z} \right]$.  

Any questions so far?
Spherical space: Let’s make a 4D sphere!
# Spherical space: From 3D to 4D

<table>
<thead>
<tr>
<th></th>
<th>3D</th>
<th>4D</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Spherical coordinates</strong></td>
<td>$(r, \phi_0, \phi_1)$, where $r \geq 0$, $0 &lt; \phi_0 \leq \pi^\circ$, $0 &lt; \phi_1 &lt; 2\pi$</td>
<td>$(r, \phi_0, \phi_1, \phi_2)$, where $r \geq 0$, $0 &lt; \phi_0 \leq \pi^\circ$, $0 &lt; \phi_1 &lt; \pi$, $0 &lt; \phi_2 &lt; 2\pi$</td>
</tr>
<tr>
<td><strong>Cartesian coordinates</strong></td>
<td>$(x_0, x_1, x_2)$, where $x_0^2 + x_1^2 + x_2^2 = r^2$</td>
<td>$(x_0, x_1, x_2, x_3)$, where $x_0^2 + x_1^2 + x_2^2 + x_3^2 = r^2$</td>
</tr>
</tbody>
</table>
| **Spherical to Cartesian coordinates** | $x_0 = r \cdot \cos(\phi_0)$  
$x_1 = r \cdot \sin(\phi_0) \cdot \cos(\phi_1)$  
$x_2 = r \cdot \sin(\phi_0) \cdot \sin(\phi_1)$ | $x_0 = r \cdot \cos(\phi_0)$  
$x_1 = r \cdot \sin(\phi_0) \cdot \cos(\phi_1)$  
$x_2 = r \cdot \sin(\phi_0) \cdot \sin(\phi_1) \cdot \cos(\phi_2)$  
$x_3 = r \cdot \sin(\phi_0) \cdot \sin(\phi_1) \cdot \sin(\phi_2)$ |
Spherical space: Spherical to Cartesian 2

<table>
<thead>
<tr>
<th></th>
<th>2D</th>
<th>3D</th>
<th>4D</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Spherical to</strong></td>
<td><strong>Cartesian coordinates</strong></td>
<td><strong>Cartesian coordinates</strong></td>
<td><strong>Cartesian coordinates</strong></td>
</tr>
<tr>
<td>$x_0 = r \cdot \cos(\varphi_0)$</td>
<td>$x_0 = r \cdot \cos(\varphi_0)$</td>
<td>$x_0 = r \cdot \cos(\varphi_0)$</td>
<td></td>
</tr>
<tr>
<td>$x_1 = r \cdot \sin(\varphi_0)$</td>
<td>$x_1 = r \cdot \sin(\varphi_0) \cdot \cos(\varphi_1)$</td>
<td>$x_1 = r \cdot \sin(\varphi_0) \cdot \cos(\varphi_1)$</td>
<td></td>
</tr>
<tr>
<td>$x_2 = r \cdot \sin(\varphi_0) \cdot \sin(\varphi_1)$</td>
<td>$x_2 = r \cdot \sin(\varphi_0) \cdot \sin(\varphi_1) \cdot \cos(\varphi_2)$</td>
<td>$x_2 = r \cdot \sin(\varphi_0) \cdot \sin(\varphi_1) \cdot \cos(\varphi_2)$</td>
<td></td>
</tr>
<tr>
<td>$x_3 = r \cdot \sin(\varphi_0) \cdot \sin(\varphi_1) \cdot \sin(\varphi_2)$</td>
<td></td>
<td>$x_3 = r \cdot \sin(\varphi_0) \cdot \sin(\varphi_1) \cdot \sin(\varphi_2)$</td>
<td></td>
</tr>
</tbody>
</table>
Spherical space: Rotating a 4D sphere

For moving

\[ R_{yw} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & \cos(\phi) & 0 & \sin(\phi) & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & -\sin(\phi) & 0 & \cos(\phi) & 0 \end{pmatrix} \]

\[ R_{xw} = \begin{pmatrix} \cos(\phi) & 0 & 0 & \sin(\phi) & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ -\sin(\phi) & 0 & 0 & \cos(\phi) & 0 \end{pmatrix} \]

\[ R_{zw} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & \cos(\phi) & \sin(\phi) & 0 \\ 0 & 0 & -\sin(\phi) & \cos(\phi) & 0 \end{pmatrix} \]

For looking around

Up and down

\[ R_{yz} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & \cos(\phi) & \sin(\phi) & 0 & 0 \\ 0 & -\sin(\phi) & \cos(\phi) & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix} \]

Left and right

\[ R_{xz} = \begin{pmatrix} \cos(\phi) & 0 & \sin(\phi) & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ -\sin(\phi) & 0 & \cos(\phi) & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix} \]

<- Forward and back | Roll ->

\[ R_{xy} = \begin{pmatrix} \cos(\phi) & \sin(\phi) & 0 & 0 & 0 \\ -\sin(\phi) & \cos(\phi) & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix} \]
Spherical space: Rotating a 4D sphere

Save rotation matrix $M$ somewhere (for example the camera)

Init $M$ as a 4 by 4 matrix.

After every movement, update $M$

$$M = M \times R_{xy} \times R_{xz} \times R_{xw} \times R_{yz} \times R_{yw} \times R_{zw}$$

The order of multiplying Rs doesn’t matter, but $M$ must be on the left!
Spherical space: Vertex shader pipeline

Vertex pos: \((\varphi_0, \varphi_1, \varphi_2)\)

Rotation matrix: 4 x 4

Spherical to Cartesian

Vertex Transformations with rotation matrices

Stereographic projections

Vertex pos: \((x, y, z)\)

The Standard Graphics Pipeline
The result
Some other spherical spaces

Lens (6, 1)

Binary Tetrahedral

Mirrored Lens 6

Lens (24, 1)

Binary Icosahedral

Mirrored tetrahedron

http://www.geometrygames.org/CurvedSpaces/index.html.en
Hyperbolic space

- Space has negative curvature (Spherical has positive)
- Uses hyperbolic trigonometry
- Objects far away seem compressed

Hyperbolica: https://www.youtube.com/watch?v=yY9GAyJtuJ0
Hyperbolic tile based game

- Object originally on a hyperbolic plane
- Projected onto a Poincaré’s disk
- Weird tessellations possible

HyperRogue: http://www.roguetemple.com/z/hyper/
Morphing space with ray tracing

- Rays are compressed or stretched based on the area

No! Euclid! https://www.youtube.com/watch?v=tI40xidKF-4
What is going on here?

AntiChamber: https://youtu.be/lFEIUcXCEvI?t=534
Tricks: Seamless teleportation

Portals

Reappearing staircases

Portal 1

Antichamber: https://www.youtube.com/watch?v=IFEIUcXEvl&feature=youtu.be&t=417
Tricks: GPU stencil buffer

Extra data buffer with 3 channels
Can be used for:
- Use different channels to render different objects
- Limited area rendering
- Rendering portals

https://youtu.be/lFEIUcXCEvI
Spherical space: Demo and questions

https://github.com/MeelisPerli/BentSpace/releases/tag/0.3.1
Thank you for listening!