AN ABSTRACT OF THE DISSERTATION OF

Jinta Zheng for the degree of Doctor of Philosophy in Computer Science presented on December 15, 2023.

Title: Interactive Design and Visualization of Arbitrary Two-dimensional Non-Euclidean Kaleidoscopic Orbifolds

Abstract approved: ______________________________

Eugene Zhang

Orbifolds are a modern mathematical concept that originates from research in hyperbolic geometry, with broad applications in computer graphics and visualization. Orbifolds contain rich information. Research in orbifolds has traditionally been limited by the construction of non-Euclidean orbifolds and their two-dimensional visual representations. This dissertation proposes an interactive system, with algorithms enabling the construction and interaction with arbitrary two-dimensional kaleidoscopic orbifolds, to address current limitations. To visualize non-Euclidean orbifolds accurately representing mathematical properties, we use rooms with mirrors as a visual metaphor for orbifolds. We first propose an algorithm to mesh the room. The rendering algorithm is modified to account for the geodesics in these spaces, which light rays follow. Moreover, maintaining the efficiency of visualization is crucial. To achieve this goal, we propose analytic mirrors and spatial hierarchies to accelerate the rendering. Our mirror-based orbifold visualization approach has the potential of helping our users gain insight on the orbifold, including its orbifold notation as well as its universal cover, which can also be the spherical space and the hyperbolic space. In addition, our visualization is able to demonstrate the infinity corners in hyperbolic kaleidoscopic orbifolds.

Interactive Design and Visualization of Arbitrary Two-dimensional Non-Euclidean Kaleidoscopic Orbifolds

by

Jinta Zheng

A DISSERTATION

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Doctor of Philosophy

Presented December 15, 2023
Commencement June 2024

ACKNOWLEDGEMENTS

Gratitude to
the pioneers, who bravely navigate unknown territory;
the companions, this dissertation is not possible without their love and support;
the followers, as the application of this dissertation will extend the life of research.

I express my gratitude to my advisor, Prof.Eugene Zhang, for his dedication to training and identifying talent in me. He has diligently guided me to overcome barriers and challenges. Our collaboration on this project, which began in 2018, has inspired valuable insights, shaping me into a researcher. The wisdom learned will undoubtedly benefit my entire life and serve as inspiration for my future career, becoming a professor when opportunities arise.

Gratitude to my co-advisor during my junior year, Prof.Yue Zhang. My achievement at this stage would not have been possible without her support and assistance. She was the guiding force that lifted me from a sense of purposelessness, even sacrificing her family time to support me. She consistently provided thoughtful consideration and encouragement for me to boldly pursue exploration.

Gratitude is expressed to my committee members:

Prof.Yue Zhang, Prof.Ren Guo, Prof.Mike Bailey, and Prof.Amir Nayyeri. Their constructive suggestions contributed to the realization of the final dissertation.

I would like to thank my friends and colleagues: Dr.Botong Qu, Shih-Hsuan Hung, Kyle Hiebel, Dr.Lawrence Roy, Prashant Kumar, Zixuan Feng, Yeajin Kim, Dr. Jun Li, Tieqiao Wang. The assistance and delightful memories shared with you will serve as a reminder of your smiles.

Gratitude is expressed for the funding support received from Oregon Department of Transportation, Electrical Engineering and Computer Science Department at Oregon State University, and National Science Foundation.

I extend my gratitude to Prof.Jing Qin, Prof.Tien-Tsin Wong, Prof.Dong Quan Liu, Dr.Tianjin Zhang, Dr.Yu Guo, and Dr.Zhi Tian for their unwavering support and care.

I would like to thank my family for their love and support, without which I could not have come this far. Special appreciation to my grandfather, my father, and my mother for their sacrifices.

TABLE OF CONTENTS

Page

1 Introduction 1
- 1.1 Recent Developments . . . 1
- 1.2 Research Problems . . . 3
- 1.3 Significance of Research Problems . . . 5

2 Related Work 8
- 2.1 Mathematical Visualization . . . 8
- 2.2 Visualization of Two-dimensional Orbifolds . . . 8
- 2.3 Visualization of Non-Euclidean Spaces . . . 10
- 2.4 Related Software . . . 11

3 Background of Non-Euclidean Orbifolds 12
- 3.1 Non-Euclidean Spaces Based on Stereographic Projections . . . 12
- 3.2 Background of Kaleidoscopic Orbifolds . . . 15

4 Enumeration of Orbifolds 19

5 User Interface Design 22
- 5.1 2D Layout Design . . . 23
- 5.2 Mirror Room Design . . . 26

6 Construction of Polygons for Orbifolds 29
- 6.1 2D Layout Algorithm . . . 29
  - 6.1.1 Spherical Orbifolds . . . 29
  - 6.1.2 Hyperbolic Orbifolds . . . 32
  - 6.1.3 Infinity Corners . . . 37
- 6.2 Meshing of Floor, Ceiling, and Walls . . . 39
  - 6.2.1 Floor and Ceiling Tessellation . . . 39
  - 6.2.2 Wall Tessellation . . . 43
  - 6.2.3 Results of Mirror Rooms . . . 43
- 6.3 Universal Cover Construction . . . 46

TABLE OF CONTENTS (Continued)

Page

7 Interacting with Orbifolds 49

7.1 Object Embedding 49

7.2 Object Movement 49

8 Rendering Orbifolds 52

8.1 Curved Ray Tracing 52

8.1.1 Definition of Curved Ray 53

8.1.2 Curved Ray-Triangle Intersection 56

8.1.3 Optics-based Visualization 57

8.1.4 Visualization of Kaleidoscopic Orbifolds by Mirror Scenes . . . 59

8.2 Analytic Rooms 68

8.2.1 Performance of Rendering by Analytic Rooms 69

8.3 Hierarchical Structures 73

8.3.1 Geodesics-based Space Partition 75

8.3.2 Cylinder-based Bounding Volume Hierarchy 76

8.3.3 Curved Ray Intersects with Nodes of Trees 76

8.3.4 Performance Comparison among Hierarchical Structures 77

9 Conclusion and Future Work 81

9.1 Contributions 81

9.2 Future Work 82

Bibliography 83

Appendices 91

A Proofs for Kaleidoscopic Orbifold Enumeration 92

B Optics-Based Visualization for Orbifold Concept and Properties 95

LIST OF FIGURES

Figure | Page

1.1 The spherical orbifold $*532$ (a), the Euclidean orbifold $*632$ (b) and the hyperbolic orbifold $*533$ (c) represented by 2D textures. The texture in (a) and (c) is made by using a partial pattern of the texture in (b). 2

1.2 Mirror-metaphor visualization of orbifolds: the spherical orbifold $*222$ (a), the Euclidean orbifold $*333$ (b), and the hyperbolic orbifold $*444$ (c). 6

3.1 The spherical space can be modeled as the surface of a unit sphere. The geodesic passing through points p and q in the space is the intersection of the plane passing through the center of the unit sphere with the sphere's surface. 13

3.2 The hyperbolic space can be modeled as the upper-sheet of a double-sheet hyperboloid. The geodesic passing through p and q in the space is the intersection of the hyperboloid with the plane that passes through p and q as well as the vertex of the lower sheet. 14

3.3 Under the stereographic projection, a geodesic in the sphere is mapped to a circle that intersects the boundary of the unit disk at a pair of antipodal points (left). Using the Poincaré disk model, a geodesic in the hyperbolic space is mapped to a circular arc that interests the boundary of the disk at the right angle (right). 14

3.4 An orbifold (right: the first quadrant in the real plane) and its universal cover (left: all four quadrants) are related by a covering map $\tau(x,y) = (|x|,|y|)$. The neighborhood of the origin in the orbifold (right) is a quarter disk. The map τ introduces a symmetry group consisting a horizontal reflection (the letter p to q), a vertical reflection (the letter p to b), and a rotation by π (the letter p to d). The symmetry group is the Dihedral group $\mathbb{D}_2$. 15

3.5 The four Euclidean orbifolds: (a) $*2222$, (b) $*333$, (c) $*244$, and (d) $*236$. 17

5.1 The interface of our design system. The left panel displays the universal cover of a dragon scene, while the right panel shows our design interface for arbitrary two-dimensional kaleidoscopic orbifolds. The interactive design panel can display the Euler characteristic, allowing users to gain insights into the classification of orbifolds. 22

LIST OF FIGURES (Continued)

Figure Page

5.2 This figure shows an example of a 2D orbifold layout designed through a design panel. In step 1, we modify the polygon cardinality of $*$ from 1 to 2, creating $*22$. Moving to step 2, we change the corner number from 2 to 3, resulting in $*33$. Continuing to step 3, we alter the polygon cardinality of $*33$ from 2 to 3, generating $*233$. Step 4 involves changing the corner number from 2 to 3, which results in $*333$. Finally, in step 5, we adjust the polygon cardinality of $*333$ from 3 to 4, producing $*2333$. 24

5.3 Users can modify the free variables in the spin box to create various polygons for $*23456$. The dashed black edge represents the fixed variable, while the dashed blue edge indicates the variable that users alter. . 25

5.4 Users drag the red arrow to position the polygon for $*23456$ in a different location (top) and rotate the red arrow to orient the polygon in a different direction (bottom). 26

5.5 This figure shows the interactive mirror room design process, including creating the underlying polygon for an orbifold (a), constructing the room with walls and a ceiling (b), attaching textures to the floor and ceiling (c), placing mirrors on the walls of the room (d), adding objects and light sources (e), generating an interactive universal cover of the orbifold (f), and rendering the mirror room (g). 28

6.1 In the stereographic plane, the $*$ is realized as the unit disk (top : left). A $*kk$ orbifold can be created by placing its corner points at a pair of antipodal points on the unit disk (top : middle). The $*22k$ type of orbifold is half of the $*kk$ orbifold (top : right). The $*23k$ ($k = 3,4,5$) can be constructed as shown in the bottom row. 30

6.2 Given a point p_i, we simplify the computation of p_{i+1} by performing a Möbius transformation in the stereographic plane for spherical orbifolds (left) and the Poincaré disk for hyperbolic orbifolds (right). In both cases, the unique Möbius transformation M maps the extended geodesic (including the intersections with the unit disk) to the upper semi-circle for spherical orbifolds and line segment between $(-1,0)$ and $(1,0)$ for hyperbolic orbifolds. In addition, the center of extended geodesic is mapped to $(0,1)$ in the spherical case (left) and $(0,0)$ in the hyperbolic case (right). Then we identify $M(p_{i+1})$ from which we can recover p_{i+1} using the inverse of M. 32

LIST OF FIGURES (Continued)

Figure Page

6.3 Two example scenarios of our decomposition algorithm for hyperbolic orbifolds: a quad is divided into two $*22k_3k_4$ type quads (left), and a hexagon is divided into two $*22k_3k_4$ quads and two $*2222k_5$ pentagons (right). 35

6.4 The special quad $*22k_3k_4$ (left) and the special pentagon $*2222k_5$ (right) are the building blocks of our decomposition algorithm. The free variables are colored in red, which correspond to the edges (dashed) introduced during the decomposition. 36

6.5 Our algorithm constructs the infinity corners represented by $*\infty\infty\infty\infty$ (a), $*\infty\infty\infty\infty\infty\infty$ (b), and $*\infty\infty\infty\infty\infty\infty\infty\infty$ (c). 38

6.6 We apply a grid-based tessellation to the spherical orbifold $*222$ (left) and the hyperbolic orbifold $*22222$ (right). The green lines represent the polygons, while the black lines are the tessellations. 40

6.7 The creation of quad-dominant mesh is depicted in this figure. For the spherical polygon (top), the geodesic family: P and Q (left), the family: S and T (center), and the intersections between the geodesic family: PQ, ST (right). For the hyperbolic polygon (bottom), the geodesic family: PQ and $P'Q'$ (left), the family: ST and $S'T'$ (center), and the intersections between the geodesic family: PQ, $P'Q'$, ST, and $S'T'$ (right). . . . 41

6.8 This figure demonstrates the poor results of the geodesics-based tessellations for the spherical orbifold $*$ (left) and the hyperbolic orbifold $*222\,\mathrm{inf}\,\mathrm{inf}$ (right). The red portions highlight the contours that cannot be covered by the quads. 42

6.9 This figure illustrates the steps of contour-based tessellation for the spherical polygon $*222$ (top) and the hyperbolic polygon $*222\,\mathrm{inf}\,\mathrm{inf}$ (bottom). The green dots represent samples on the edges of the polygons (left), while the center image depicts the geodesics connecting the pairs of the red dots and the green dots. The final step involves forming a set of geodesics between the adjacent geodesics to create the quad-dominant tessellation. 44

LIST OF FIGURES (Continued)

Figure | Page

6.10 Mirror rooms are displayed for spherical orbifolds (top), Euclidean orbifolds (middle), and hyperbolic orbifolds (bottom). The floors are depicted in peach color, the walls are represented in light blue, and the ceilings are indicated in dark blue. 45

6.11 Four orbifolds generated using our orbifold layout creation algorithm: (a) the $*235$ orbifold (spherical), (b) the orbifold $*23456789(10)(11)$ (hyperbolic), (c) the orbifold $*\infty\infty\infty\infty$ (hyperbolic), and (d) the orbifold $*\infty\infty\infty\infty\infty\infty\infty\infty$ (hyperbolic). 48

7.1 This figure shows the embedding of a dragon in the spherical orbifold $*222$ (top) and the hyperbolic orbifold $*22222$ (bottom). The red dot indicates the mass center of the dragon, and the last column displays the top view. 50

7.2 The dragon is translated along the real axis in the stereographic plane (top) and the Poincaré disk (bottom). The dragon's mesh is deformed based on its location. 51

8.1 In (a-b), we show the comparison of the rendering of a hyperbolic orbifold with straight rays as in the Euclidean space (a) and the geodesics in the hyperbolic space (b). 53

8.2 The decomposition of a spiral in hyperbolic space. The gray disk is the Poincaré disk in the horizontal plane, and the gray arrow represents the vertical direction. The spiral (a) is decomposed into the geodesic ray (b) along the direction v_H and the vertical ray (c) along the vertical direction v_Y. 54

8.3 The paths and tangents of spirals in hyperbolic space. (a-d) are the paths of the spirals. (a) $|v_H| := 0$, and $|v_Y| \neq 0$; (b) $|v_H| \neq 0$, and $|v_Y| := 0$; (c) v_H is along pO; (d) $|v_H| \neq 0$, and $|v_Y| \neq 0$; (e) shows the tangents of the points on the spiral (d). 54

8.4 This figure shows the comparison between projection in Euclidean space (left) and projection in non-Euclidean spaces (right). The top image represents the spherical orbifold $*235$, and the bottom image represents the hyperbolic orbifold $*2352$. 56

LIST OF FIGURES (Continued)

Figure Page

8.5 A light ray traveling along a curved ray undergoes reflection and refraction on a surface. The dashed line illustrates the curved path of the light ray, and the solid yellow line indicates the directions of reflection and refraction, corresponding to the tangents of the curved rays at the intersection. 58

8.6 By varying the attenuations of the mirrors, our visualization can emphasize the orbifold (a), its translational cover (b), or its universal cover (c). 59

8.7 An avatar walks along the walls of the orbifolds. The universal cover is the sphere for (a-b) and the hyperbolic space for (c-d). 61

8.8 Status scenes for the orbifolds: $*2233$ (left) and $*2323$ (right). Each row displays views as the viewer rotates from left to right. 62

8.9 We render triangular non-orbifolds with corner angles $\frac{2\pi}{3}$-$\frac{\pi}{3}$-$\frac{\pi}{3}$ (left) and corner angles $\frac{2\pi}{3}$-$\frac{\pi}{12}$-$\frac{\pi}{12}$ (right). Notice the dragon now has two heads. . 64

8.10 The figure depicts the Buddha within an orbifold $*kkk$ where k ranges from 2 to 7 (a-f). For $k \geq 4$, the underlying space of the orbifold is hyperbolic. 65

8.11 The figure shows the Buddha within an orbifold $*2_1...2_N$, where N ranges from 2 to 6 (a-f). For $N \geq 5$, the underlying space of the orbifold is hyperbolic. 66

8.12 We render the hyperbolic orbifold with a corner at infinity by adding one mirror at a time for $*\infty\infty\infty\infty$ at (a-c). We increase the number of infinity corners from (d-f). The infinity corner appears dark because the light bounces between mirrors an infinite number of times, causing the energy of the light ray to attenuate to 0. 67

8.13 This figure shows the hierarchical organization of partial mirrors with various colors on a circular wall (right) within a mirrored room (left). . 68

LIST OF FIGURES (Continued)

Figure Page

8.14 The figure illustrates that the analytic room (green line) outperforms the meshed room in terms of rendering time. Specifically, each scene's analytic room corresponds to the first row in Table 8.1, while meshed levels 0, 1, and 2 correspond to the second, third, and fourth rows, respectively. Additionally, it is observed that as the number of triangles increases, the computational overhead also rises. 71

8.15 (a-c) illustrate the decomposition of an AABB into horizontal and vertical surfaces. (b), (d), and (f) depict the procedure for checking intersections between curved rays and horizontal surfaces. (c), (e), and (g) illustrate the procedure for checking intersections between curved rays and vertical surfaces. 74

8.16 These figures illustrate that the hierarchical structure does not hold with translations for AABB. The image (top) illustrates the bounding hierarchy in spherical space, while the image (bottom) represents the bounding hierarchy in hyperbolic space. After translating the object from left to right, the green circles are outside the red boxes. 75

8.17 Geodesics for space partition in spherical (a) and hyperbolic space (b), the examples of bounding an edge (c-d), and the examples of bounding a triangle (e-f). The dark green dots represent vertices of edges, and the green dot represents the projected points on the axes. The dashed lines represent the geodesics for space partition. 79

8.18 These figures demonstrate that the hierarchical structure still holds with left-to-right and top-to-bottom object translations. Thus, updating the center and radius of the bounding cylinders is the only required task, significantly easier than reconstructing the hierarchical tree. The image (top) illustrates the bounding hierarchy in spherical space, and the image (bottom) represents the bounding hierarchy in hyperbolic space. The red bounding volume encloses the white bounding volume, which in turn encompasses the green bounding volume. Within the white bounding volume resides a white dragon, and within the green bounding volume resides a green dragon. 80

LIST OF TABLES

Table Page

4.1 Our enumeration of all two-dimensional kaleidoscopic orbifolds based on the cardinality and the universal cover of the orbifolds. The orbifolds on each row have the same N, which is the cardinality of the underlying polygon. 20

6.1 Calculating the accuracy of infinite corners. The first row shows the formulation for calculations, and the numerical accuracy is 6 digits. *NaN* is not a number. 37

8.1 This table presents the scene settings for rendering, with the last column indicating the rendering time in minutes. The underscored scene represents the room with analytic representation. The first row of the table includes the orbifold used for testing, the type of the orbifold, the size of the XML file used in Mitsuba, the number of triangles for the room in the scene (including the floor, ceiling, and mirrors), the total number of triangles (including the room and the objects), and the time of rendering. All values in the table for measurements are rounded to two decimal places. 72

8.2 Comparing the performance of hierarchical structures over time. The table's first column displays various objects used for testing, while the second column indicates the type of curved ray in the corresponding space. The third column represents the number of intersections. The remaining columns (AABB-based Kd-Tree, Geodesics-based Kd-Tree, and Cylinder-based BVH) present performance metrics, including the depth of the tree, construction time in milliseconds, and travel time to find the intersection in microseconds per ray. The final row displays the average results for all objects. 78

LIST OF ALGORITHMS

Algorithm Page

1 A Curved Ray Intersects with A Room and Triangles . . . 69

LIST OF APPENDIX FIGURES

Figure Page

B.1 A square room with two or four mirrors. The case in the four mirror room (c) corresponds to a Euclidean orbifold $*2222$. 96

B.2 The three triangular Euclidean kaleidoscopic orbifolds. 96

Chapter 1: Introduction

Orbifolds are an important modern mathematical concept. An orbifold is a generalization of a manifold. In simpler terms, an orbifold is a topological space that is locally modeled by a finite group quotient of Euclidean space. The concept of orbifolds was originally proposed by the mathematician William Thurston in the 1970s [67].

Orbifolds are crucial in many dimensions. They have been used to examine the geometric properties of hyperbolic spaces in mathematics. In 2003, they were used to prove the famous Poincaré conjecture for three dimensions, which was considered to be the most difficult one [57]. They have been used in theoretical physics to describe spatial symmetries for string theory [35], and to model the universe in cosmology [79]. In order to explore the symmetrical properties of crystal structures, structural crystallography has also utilized the orbifold concept [38]. Orbifolds are also widely used in tensor field topology [61], texture synthesis [53, 82], injective parameterizations of surface meshes [7, 9, 6], and music [68].

1.1 Recent Developments

Understanding orbifolds is challenging because of the following factors:

1. Abstract concept. Orbifold is an abstract mathematical concept combining ideas from topology, abstract algebra, geometry, and group theory. Learning the concept requires a solid foundation in those areas.

2. Complex definition. Orbifolds are defined by intricate mathematical language and notation, which involves *Hausdorff spaces*, *groups and group actions*, *covering*, *charts*, and *homeomorphism*.

3. Difficulty in visualization. The abstract concepts are difficult to relate to intuition, particularly in non-Euclidean space, where the geometric structure of the majority

of orbifolds is found.

The dissertation concentrates on two-dimensional orbifolds generated through reflections in the Euclidean plane, spherical space, and hyperbolic space.

Current representations of two-dimensional orbifolds frequently use 2D textures that tile the underlying space seamlessly [16]. The user needs to identify corresponding orbits for points or patterns from the universal cover of the orbifold depicted by the texture. The human cognitive system must infer the transformations between locations within the same orbit. A Euclidean orbifold, as seen in Figure 1.1 (b), is represented by the orbifold notation $*632$. In order to determine the orbifold, the user must first identify the points that are placed in the same orbit. Corner points can be identified by examining the relationships between nearby points in separate orbits. The identification of the transformation becomes possible by counting the type and order of symmetry. Besides Euclidean space, it's important to note that a two-dimensional orbifold can exist in spherical or hyperbolic space, as illustrated in Figure 1.1(a) and Figure 1.1(c). These spaces offer crucial details about space curvatures, the parallel postulate, angle-area relationships, and geodesics.

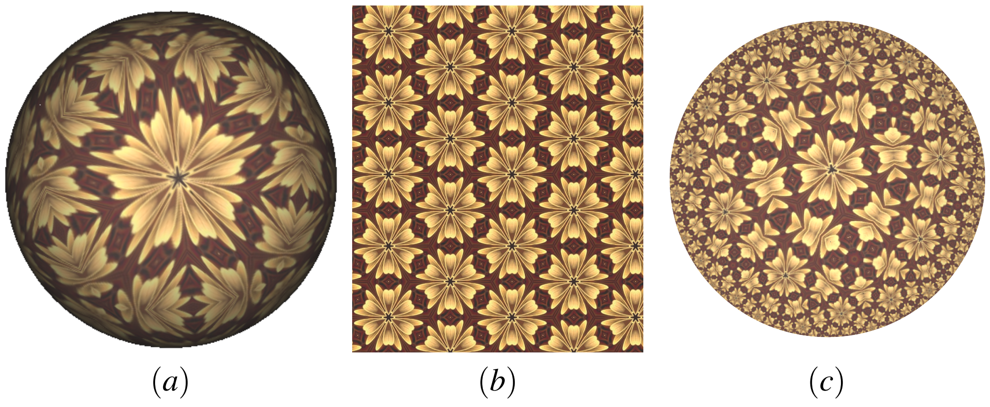


Figure 1.1: The spherical orbifold $*532$ (a), the Euclidean orbifold $*632$ (b) and the hyperbolic orbifold $*533$ (c) represented by 2D textures. The texture in (a) and (c) is made by using a partial pattern of the texture in (b).

Based on the aforementioned analysis, visualizing an orbifold from a 2D texture requires considering multiple factors. These include the position of viewer, the associated points of the orbits, the transformations between those points, and the method employed to project the orbits onto the view plane.

Using 2D textures is a valuable method for visualizing orbifolds and illustrating their universal cover. However, certain crucial properties of orbifolds cannot be adequately visualized using this method:

- 2D textures require additional interactions or annotations to emphasize different aspects of an orbifold: the local symmetries, the cone points, the exact location and orientation of reflective edges, the orbifold itself, and the covering spaces.
- Understanding non-orbifolds is important. Orbifolds and non-orbifolds cannot be intuitively correlated by 2D textures. We are the first to visualize examples of non-orbifolds.
- Orbifolds extend beyond 2D textures; 3D objects can utilize orbifolds by extruding a two-dimensional orbifold into three dimensions. The mirror maze is an iconic real-life example. In contrast to 2D textures, 3D objects allow users to explore the visualization more naturally and from multiple perspectives, enhancing entertainment, information, and intuitiveness. Additionally, 3D objects with orbifolds can be valuable for physical prototyping and modeling.

1.2 Research Problems

While there are only a handful of Euclidean orbifolds, there are infinitely many spherical and hyperbolic orbifolds. In fact, any polygon whose corner angles can each be expressed as $\frac{\pi}{k}$ ($k \in \mathbb{N}^+$) is an orbifold. To the best of our knowledge, there is no algorithm published that allows the realization of arbitrary such polygons when their natural spaces are hyperbolic. Most available tools focus on regular polygons. For arbitrary polygons that represent an orbifold, the lengths of the edges are challenging to determine.

Current educational practices and research indicate an increasing trend in integrating

3D graphics and animations to enrich the learning and visualization experience. [54, 19, 65, 78, 43]. Orbifold visualization with 3D techniques presents several challenges due to the complexities of orbifolds and non-Euclidean geometries. Orbifolds, derived from non-Euclidean spaces, are characterized by curved surfaces and a distinct set of geometric rules compared to Euclidean geometry. Visualization with 3D techniques can offer intuitive insights, and challenges arise when attempting to adapt existing rendering algorithms to conform to the rules of non-Euclidean geometry from a geometric perspective. In non-Euclidean spaces, light rays do not follow straight lines, but rather geodesics, which are the shortest paths between two points. 3D techniques must be adapted to account for these geodesic light paths to create accurate visualizations.

This dissertation aims to address the following research problems:

- While spherical and Euclidean orbifolds have been fully enumerated, there is currently no explicit enumeration available for hyperbolic orbifolds, to the best of our knowledge. The objective of this dissertation is to present a detailed enumeration, including the Euler characteristic, and to deliver this enumeration to users through an interactive user interface.

- This dissertation aims to utilize a 3D interactive technique for designing and visualizing arbitrary two-dimensional kaleidoscopic orbifolds. Specifically, the research focuses on addressing the challenges of creating arbitrary orbifolds, construction of rooms for orbifolds, interacting with them, and using real-time projection based on non-Euclidean geometry for non-Euclidean orbifolds.

- The reflectional symmetries in a kaleidoscope correspond to the behaviors of a particular Euclidean orbifold. Kaleidoscopes or mirror mazes provide an engaging and intuitive approach for visualizing Euclidean orbifolds through mirror metaphor. This dissertation aims to extend the mirror metaphor to non-Euclidean orbifolds and develop a coherent intuition for photo-realistic rendering of mirror scenes, including addressing challenges such as tracing curved rays, intersecting with meshes, speeding up the tracing process.

- The importance of hierarchical structures in non-Euclidean spaces is on par with those in Euclidean space. This dissertation is dedicated to investigating structures that specifically recognize and leverage the geometric features in non-Euclidean spaces.

1.3 Significance of Research Problems

This dissertation has a wide impact on mathematical education and computer graphics:

- Coupling explicit enumeration with a user interface for all two-dimensional kaleidoscopic orbifolds, based on the combination of polygon cardinality and the universal cover, will enhance the understanding and accessibility of the connections among Euler characteristics, underlying spaces, and kaleidoscopic orbifolds.

- The unrestricted creation of two-dimensional kaleidoscopic orbifolds in our interactive design system offers more fundamental units of surface parameterization [9], mesh processing, and texture synthesis [53, 82] in computer graphics. By interacting with orbifolds, users of our interactive design system can learn about them and gain a deeper understanding of mathematical concepts related to Non-Euclidean space and orbifolds.

- The visualization with mirror metaphor enables the exploration of complex kaleidoscopic orbifolds without the requirement of extra duplication of symmetries. Photo-realistic rendering of orbifolds not only provides a coherent intuitive perception of Euclidean orbifolds and non-Euclidean orbifolds but also allows for more attractive creation based on the design of the material of objects. Users of this immersive visualization can learn more about the orbifolds as well as the underlying space. As shown in Figure 1.2, our system produces orbifolds (configuration of the ceiling and the floor) and the symmetry that each orbifold induces. In addition, through the bending of the mirror frames and the unfamiliar deformations of Buddha in (a) and (c), the notions of spherical geometry and hyperbolic geometry are visually delivered, respectively.

- The proposed hierarchical structures, to organize triangles, complement the geometric characteristics of the underlying space. They can be applied to applications with the similar requirements.

In addition, there is increased workload deploying non-Euclidean spaces or mathematical concepts for gaming in the computer graphics community [13, 65, 43]. Our interactive design system and rendering algorithms for orbifolds can create immersive and visually stunning environments for video games and virtual reality applications.

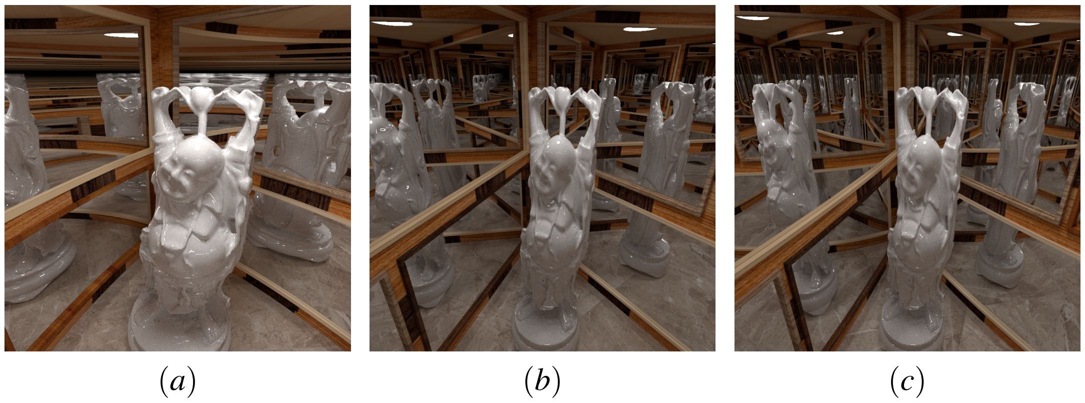


Figure 1.2: Mirror-metaphor visualization of orbifolds: the spherical orbifold $*222$ (a), the Euclidean orbifold $*333$ (b), and the hyperbolic orbifold $*444$ (c).

This dissertation is organized as follows to address the research problems: Chapter 2 provides a detailed overview of existing work on the visualization of orbifolds. Subsequently, we introduce the models used in this dissertation and the fundamental concepts of orbifolds in Chapter 3. Following that, we propose an enumeration of orbifolds based on their cardinality and universal cover (Chapter 4).

Our user interface, which enhances the understanding of orbifolds, is built upon the enumeration described in Chapter 5. We then introduce the algorithms for constructing polygons for arbitrary kaleidoscopic orbifolds and the algorithm enabling interactive creation of the universal cover using shaders in Chapter 6. For interacting with objects within the orbifold, we introduce the algorithm for embedding the orbifold into space and translating within the space (Chapter 7). After gaining the ability to interactively design orbifolds, we present a curved ray tracing algorithm designed to visualize orbifolds with mirror rooms in Chapter 8. We also introduce hierarchical structures designed to enhance rendering speed and complement the features of the underlying space. Lastly, the dissertation concludes by summarizing its contributions and outlining future directions in Chapter 9.

Chapter 2: Related Work

In this chapter, we review the most relevant research in the field of mathematical visualization, visualization of orbifolds, visualization of non-Euclidean spaces, and related software, focusing on the primary techniques and their limitations.

2.1 Mathematical Visualization

This dissertation is inspired by the contemporary advancements in the field of mathematical visualization, such as quaternions [31], knots and links [69], 3D printing for mathematical visualization [63, 45], and branched covering spaces [62]. Our research strives to advance this area of study and promote interest and engagement in mathematical visualization within computer science communities. Specifically, our focus is on orbifolds generated by reflections, incorporating both translational and rotational symmetry. With potential applications in various domains, such as computer graphics and materials science, this research fills a crucial gap in our comprehension of orbifolds and their intuitive aspects. By exploring a broader range of symmetries and providing new insights into their visualization and processing, this research contributes to the ongoing efforts in visualizing orbifolds.

2.2 Visualization of Two-dimensional Orbifolds

M.C. Escher's four ”Circle Limit” woodcuts depict hyperbolic orbifolds [24]. This 2D texture provides a natural visualization of orbifolds. Conway et al. [16] employed artwork with symmetrical textures to illustrate concepts related to orbifolds. However, as mentioned earlier, this artwork primarily emphasizes the universal cover of an orbifold rather than the orbifold itself.

There are essentially two challenges in creating the universal covers of an orbifold: generating the fundamental domain and then producing the universal cover based on the fundamental domain and the related group for orbifolds. Dunham [23, 21, 22] employs a matrix-based hyperbolic pattern generation method by transforming the fundamental domain in the Poincaré disk model. Addressing image resolution limitations, a more effective approach using reverse pixel lookup and GPU solutions is proposed [74, 75]. Beyond pattern generation, creative possibilities include bending hyperbolic patterns [14], establishing hyperbolic symmetry [75], or creating spherical symmetry [48] from the wallpaper pattern. Zeller *et al.* [83] provided a database and program for exploring two-dimensional periodic tilings in E^2, S^2, and H^2. However, their approaches have limitations in creating arbitrary kaleidoscopic orbifolds.

Beyond creating repeated patterns, two-dimensional orbifolds play a role in the Tutte embedding of 3D meshes [8, 9, 5].

To deal with this challenge, we initially propose algorithms for creating polygons for arbitrary orbifolds. We suggest an alternative visual metaphor, likening an orbifold to a mirrored scene. This metaphor allows observers to recognize whether a scene is an orbifold and identify its type based on real-life experiences with mirrors. Additionally, we introduce a system capable of generating a room that corresponds to any given two-dimensional kaleidoscopic orbifold, even if its universal cover is a non-Euclidean space.

We utilize the mirror metaphor to enhance our understanding of orbifolds, providing a complementary approach to texture-based orbifold visualization. The mirror metaphor transforms a two-dimensional orbifold into a three-dimensional room, alternatively conceptualized as a three-dimensional orbifold by combining a two-dimensional orbifold (floor and ceiling) with a line segment (room height). In the computer graphics community, physically based rendering aims to replicate real-world scenes on a computer, producing realistic results for an immersive perception. The simulation of light propagation and the interaction of lights with objects is a specific focus within photo-realistic rendering [54]. Our mirror-based orbifold visualization aims to depict non-Euclidean orbifolds using curved ray tracing in non-Euclidean spaces, representing a significant advancement in the visualization and understanding of orbifolds.

2.3 Visualization of Non-Euclidean Spaces

There has been recent research on rendering three-manifolds. Our idea is inspired by this work, and the distinctions will be outlined in this chapter. An n-manifold is a topological space that locally resembles n-dimensional Euclidean space. Thurston's geometrization conjecture identifies eight Thurston geometries in three-manifolds: E^3, S^3, H^3, $S^2 \times E$, $H^2 \times E$, Nil, $\widetilde{SL}(2,R)$, and Sol. Thurston discussed approaches to visualize three-manifolds with computer aid, influencing the visualization of three-manifolds [66, 56, 19]. There has been some past research on visualizing three-dimensional orbifolds [28, 12, 55], with a focus on the three-dimensional sphere S^3 and three-dimensional hyperbolic space H^3. In these spaces, the geodesics are either a circular arc or a hyperbola. Notably, the geodesics in our product spaces are spirals, distinguishing them from those in S^3 and H^3 and requiring a different approach to the ray-triangle intersection. Additionally, previous research has mainly focused on popular orbifolds in S^3 and H^3, such as the Poincaré sphere and the mirror dodecahedron. Our research aims to expand the scope by providing a method for visualizing any arbitrary two-dimensional kaleidoscopic orbifold, using the example of a room with mirrors.

A game-like approach is more successful and enjoyable for presenting isotropic non-Euclidean spaces. Due to the unintuitive perception of non-Euclidean spaces, 3D gaming engines that have adapted to the rules of non-Euclidean spaces must handle object conversion, transformation, lighting, and physical simulation [64, 50]. For interactive exploration, a real-time rendering pipeline for non-Euclidean spaces was employed. Weeks [77, 80, 46] projected objects onto the tangent space and then applied the rendering pipeline for matrix transformation as E^3. The projection by the exponential map is applicable for isotropic geometries (S^3 and H^3) and product geometries ($S^2 \times E$ and $H^2 \times E$). Additionally, virtual reality and virtual environmental equipment have been utilized to explore the space of S^3, H^3, and $H^2 \times E$, providing an immersive visualization experience [33, 32, 72, 78, 27]. The approaches presented are designed for three-manifolds; however, when these approaches are applied to spheres and hyperboloids, which are two-manifolds, an additional distortion is introduced along the third dimension. In this dissertation, we employ the spiral-triangle intersection to project each vertex on an object in the scene or a virtual duplicate onto the image plane. In this scenario, real-time rendering with an interactive frame rate is achievable.

2.4 Related Software

Several existing software programs are available for visualizing orbifolds or employing non-Euclidean space rendering in gaming, offering opportunities for learning about orbifolds and the corresponding spaces. For instance, KaleidoTile, akin to Conway's approach, can showcase all triangular two-dimensional kaleidoscopic orbifolds as 2D textures, allowing users to interact with the orbifold by changing the texture and translating symmetries [4]. Zeller *et al*. also developed a precomputed database and software for exploring two-dimensional periodic tilings via Delaney-Dress symbols, not limited to triangular orbifolds [83]. However, their approach is constrained by the polygon with an incircle for the creation of a fundamental domain, rendering it incapable of producing all kaleidoscopic orbifolds. Both of these tools generate universal cover by duplicating the fundamental domain. In contrast, this dissertation can generate arbitrary two-dimensional kaleidoscopic orbifolds in real-time for 3D visualization, including the texture-based approach. Moreover, our system facilitates the interactive creation of orbifolds via Möbius transformation and the generation of universal covers by mirrors without the need for extra duplication.

Another category of software focuses on gaming and utilizes the visualization of non-Euclidean spaces with the universal cover of orbifolds for designing games, creating an interactive and engaging environment beneficial for learning non-Euclidean spaces. One such game is HyperRogue, a puzzle roguelike that unfolds in the Poincaré disk, employing orbifolds to craft the map's tiling [2]. Another game, Hyperbolica, enables users to navigate inside underlying space with polished graphics, also generated via universal covers [1]. Both games employ Weeks' approaches for visualizing H^3 and S^3 [77, 80], founded on two-dimensional geometries (sphere or Poincaré disk). Similarly, this dissertation facilitates interaction within non-Euclidean spaces, allowing any two-dimensional kaleidoscopic orbifold to create the universal cover. Additionally, the projections are based on two-dimensional spaces to avoid additional distortion.

Chapter 3: Background of Non-Euclidean Orbifolds

In this chapter, we introduce the basic background of non-Euclidean spaces (Chapter 3.1) and the background of kaleidoscopic orbifolds (Chapter 3.2).

3.1 Non-Euclidean Spaces Based on Stereographic Projections

We first introduce the fundamental spherical and hyperbolic models. By working on the planar models, where we perform stereographic projections on the original models, we can use the advantages of using the same framework to construct polygons, mesh mirror rooms, and trace curved rays. The only difference is to identify the geodesics. For spherical space, the geodesic is identified by antipodal points; however, for hyperbolic space, the geodesic is identified by symmetric points.

The stereographic projection [34] maps the unit sphere (Figure 3.1) to the plane $z = 0$ such that the equator (a unit circle) is mapped to itself and the north pole is mapped to the origin in the plane. In this case, the northern hemisphere is mapped to the inside of the unit disk bounded by the equator while the southern hemisphere is mapped to the outside of the unit disk. This plane can be identified as the complex plane, i.e. the set of complex numbers. The south pole is mapped to ∞. The geodesics are mapped to circles in the plane that intersect the unit circle at a pair of antipodal points (Figure 3.3 (left)).

The hyperbolic plane can be modeled as the upper sheet of the double-sheet hyperboloid $z^2 - x^2 - y^2 = 1$ (Figure 3.2). Like the Euclidean orbifolds, both spherical orbifolds and hyperbolic orbifolds are polygons whose edges follow the geodesics in their universal cover. The geodesics in the unit sphere are the great circles, and the geodesic passing through two mutually distinct points p and q in the hyperbolic space is the intersection of the plane containing p, q and the vertex of the lower-sheet of the hyperboloid (Figure 3.2: the curve passing through p and q).

For a non-Euclidean orbifold, its universal cover is either the sphere or the hyper-

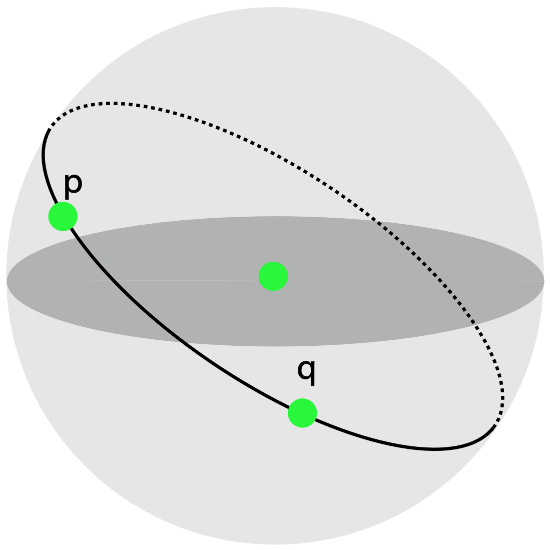


Figure 3.1: The spherical space can be modeled as the surface of a unit sphere. The geodesic passing through points *p* and *q* in the space is the intersection of the plane passing through the center of the unit sphere with the sphere's surface.

bolic space. Constructing a 3D room over the sphere and the hyperboloid would require a second sphere or hyperboloid to hold the ceiling. While it is possible to construct the room this way, we instead choose to express the orbifold using a planar model, i.e. the stereographic projection for the sphere [34] and the Poincaré disk [20] for the hyperbolic space. By using these models, we have a unified framework in which any polygon, regardless of the type of its universal cover, can be constructed in the plane.

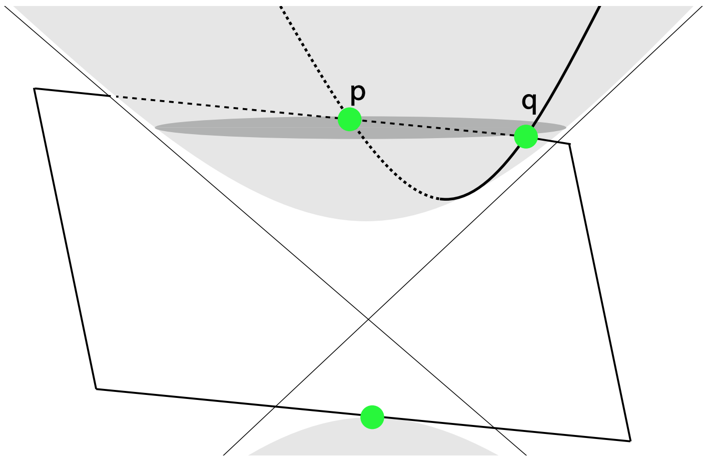


Figure 3.2: The hyperbolic space can be modeled as the upper-sheet of a double-sheet hyperboloid. The geodesic passing through *p* and *q* in the space is the intersection of the hyperboloid with the plane that passes through *p* and *q* as well as the vertex of the lower sheet.

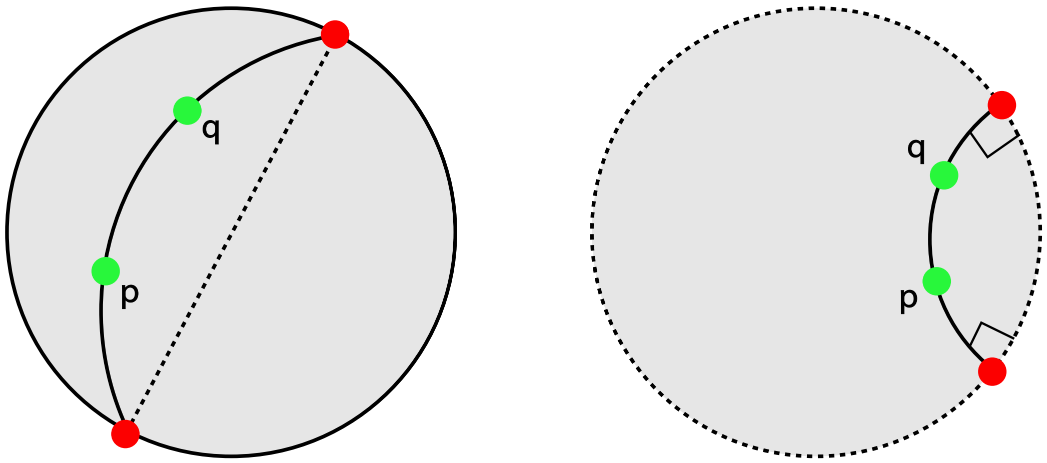


Figure 3.3: Under the stereographic projection, a geodesic in the sphere is mapped to a circle that intersects the boundary of the unit disk at a pair of antipodal points (left). Using the Poincaré disk model, a geodesic in the hyperbolic space is mapped to a circular arc that interests the boundary of the disk at the right angle (right).

3.2 Background of Kaleidoscopic Orbifolds

In this chapter, we review necessary mathematical background on orbifolds used in this paper which include the concepts of *groups* and *group actions* [36], *orbifolds* [51, 18], and *non-Euclidean spaces* [20]. For a rigorous definition of these concepts, we refer our readers to the aforementioned references.

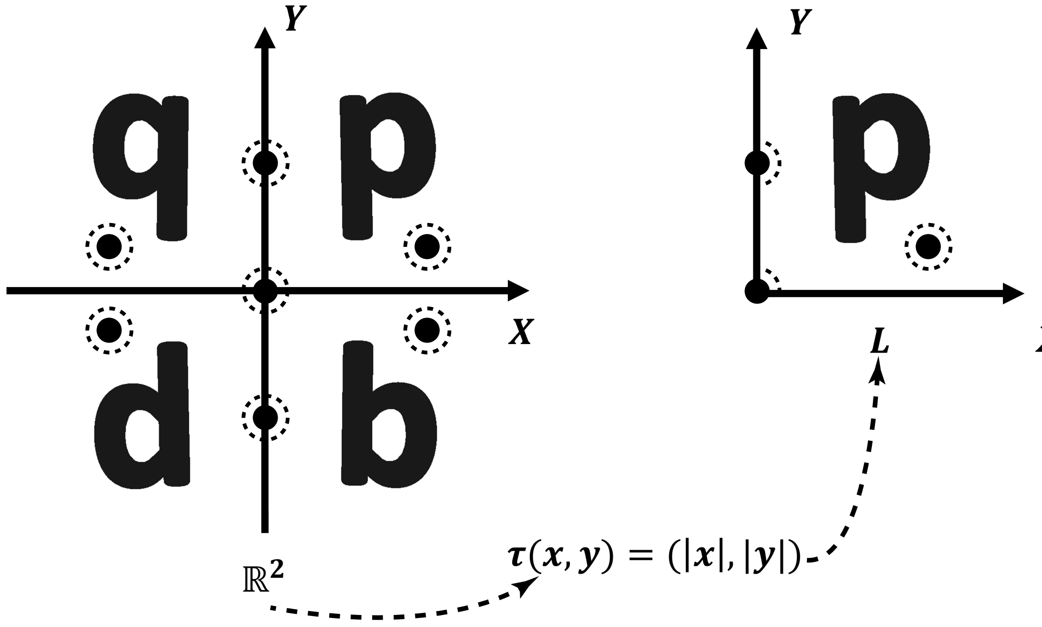


Figure 3.4: An orbifold (right: the first quadrant in the real plane) and its universal cover (left: all four quadrants) are related by a covering map $\tau(x,y) = (|x|,|y|)$. The neighborhood of the origin in the orbifold (right) is a quarter disk. The map τ introduces a symmetry group consisting a horizontal reflection (the letter p to q), a vertical reflection (the letter p to b), and a rotation by π (the letter p to d). The symmetry group is the Dihedral group $\mathbb{D}_2$.

An orbifold O is a topological space X paired with a discrete symmetry group G such that X locally resembles a Euclidean disk under the action of G. To better illustrate this, consider the space $L = \{(x,y)|x,y \geq 0\}$ (Figure 3.4 (right)). For each point in the first quadrant, we can find a small enough disk-shaped neighborhood. However, for a point on the positive Y-axis, there is a neighborhood of the shape of a half-disk which corresponds to a full disk in the Euclidean plane (Figure 3.4 (left)) under the reflection across the Y-axis. In other words, the union of the half disk in the first quadrant and its mirror reflection form a full disk. Finally, the origin has a quarter-disk-shaped neighborhood (Figure 3.4 (right)) which corresponds to a full disk in the Euclidean plane (Figure 3.4 (left)) when being combined with its reflection across the X-axis (in the fourth quadrant), the reflection across the Y-axis (in the second quadrant), and the rotation by π around the origin (in the third quadrant). Thus, L is an orbifold.

Globally, we can see that L is the range of the following function $\tau(x,y) = (|x|,|y|)$, which introduces a map from $\mathbb{R}^2$ to L with the symmetry illustrated as follows. The letter p (Figure 3.4 (right)) corresponds to the letter q in the second quadrant (Figure 3.4 (left)) through the reflection across the Y-axis and the letter b in the fourth quadrant through the reflection across the X-axis. In addition, it corresponds to the letter d in the third quadrant through a rotation of π around the origin, which is a composition of the two aforementioned reflections. Thus, the symmetry induced by the map τ leads to a symmetry group of four elements: the identity, two reflections, and one rotation. The group is the *Dihedral group* of order 2, i.e. $\mathbb{D}_2$, which, when acted on $\mathbb{R}^2$, leads to the orbifold L. It has a *corner point* at the origin and two mirror lines (the positive X-axis and the positive Y-axis).

In general, a two-dimensional *kaleidoscopic* orbifold O is a polygon with a symmetry group induced by reflections across all of its edges. Without causing ambiguity, we also refer to the polygon as O. Each edge of the polygon is thus a mirror line, and every vertex of the polygon is a corner point corresponding to the symmetry of $\mathbb{D}_k$, the dihedral group of order k. Note that $\mathbb{D}_k$ consists of k rotations (including the identity) and k reflections. In Figure 3.5 we show four such orbifolds, whose polygons have a configurations of a square (a), a $60^\circ - 60^\circ - 60^\circ$ triangle (b), a $90^\circ - 45^\circ - 45^\circ$ triangle (c), and a $90^\circ - 60^\circ - 30^\circ$ triangle (d). These orbifolds are given the *orbifold notations* (a) $*2222$, (b) $*333$, (c) $*244$, and (d) $*236$, respectively. A generic kaleidoscopic orbifold corresponding to an N-gon O is given the notation $*k_1...k_N$ where the $*$ indicates the existence of the mirror and k_i implies that the angle of the polygon at the i-th corner is $\frac{\pi}{k_i}$.

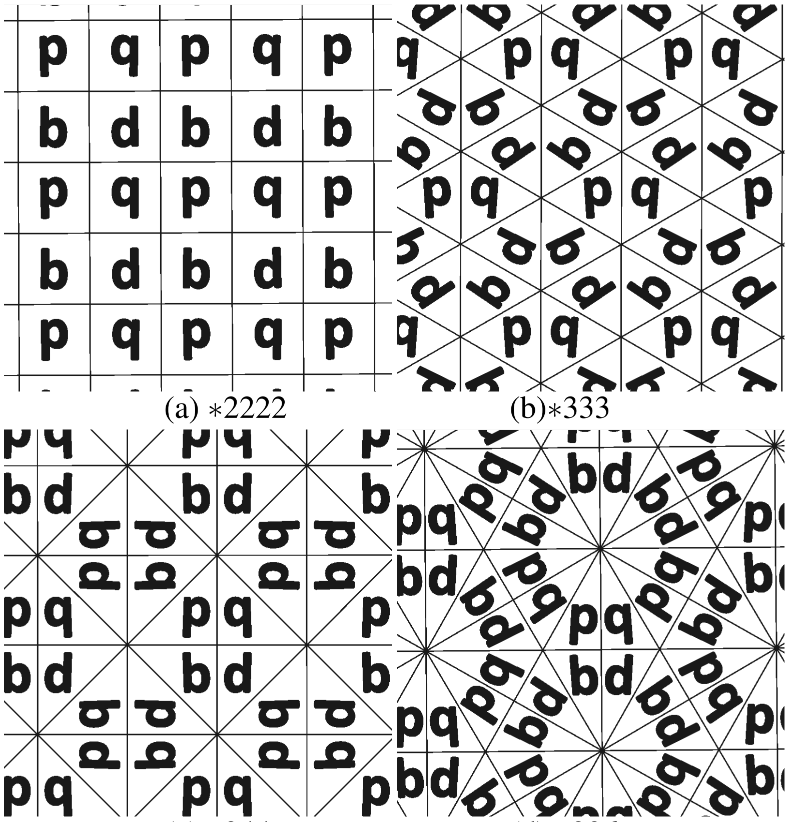

(a) $*2222$ (b)$*333$ (c) $*244$ (d) $*236$

Figure 3.5: The four Euclidean orbifolds: (a) $*2222$, (b) $*333$, (c) $*244$, and (d) $*236$.

An orbifold (the polygon) and all of its virtual copies through its symmetry group can seamlessly tile a space, which is its *universal cover*. The aforementioned orbifolds are kaleidoscopic orbifolds whose universal cover is the Euclidean plane, thus *Euclidean orbifolds*. Each Euclidean orbifold has a *translational cover*, which, along with its translational copies, form the universal cover. The translational cover of $*2222$ consists of the orbifold, two of its reflections, and one rotation by π (Figure 3.5 (a): any 2×2 subgrid with the letters q, p, d, and b). The translations needed to generate the universal cover is the Gaussian integer grid $\mathbb{Z}[i]$ [26]. The translational covers of the other Euclidean orbifolds respectively consist of six copies arranged in a hexagon (Figure 3.5 (b): $*333$), eight copies arranged in an octagon (Figure 3.5(c): $*244$), and twelve copies arranged in a dodecagon (Figure 3.5(d): $*236$). The set of translations for $*244$ is also $\mathbb{Z}[i]$. On the other hand, the set of translations for $*333$ and $*236$ is the Eisenstein integer grid $\mathbb{Z}[\omega]$ [26] where $\omega = \frac{-1+\sqrt{3}i}{2}$.

While it may seem that these are the only kaleidoscopic orbifolds and that all kaleidoscopic orbifolds must be triangular or rectangular, there are many more. In fact, given an arbitrary polygon with at least three sides and whose corner angles divide π individually, there is an orbifold that corresponds to the polygon. Figure 8.7 shows a room with three, four, and five mirrors, respectively. However, these orbifolds cannot tile the Euclidean plane as their universal covers are either the unit sphere (spherical orbifolds) or the hyperbolic plane (hyperbolic orbifolds). In fact, given an orbifold $O = *k_1 \ldots k_N$ where N is the number of walls and $k_i > 1$ $(1 \leq i \leq N)$, its universal cover is decided by the *Euler characteristic* of the orbifold as follows:

$$\chi(O) = \sum_{i=1}^{N} \frac{1}{2k_i} - \frac{N}{2} + 1. \tag{3.1}$$

An orbifold O is spherical, Euclidean, or hyperbolic when $\chi(O) > 0$, $\chi(O) = 0$, $\chi(O) < 0$, respectively.

In the next chapter, we describe our orbifold design system starting with an enumeration of all two-dimensional kaleidoscopic orbifolds.

Chapter 4: Enumeration of Orbifolds

While there has been a complete enumeration of spherical and Euclidean orbifolds, to our best knowledge such an enumeration is not explicitly given for hyperbolic orbifolds. In addition, the enumeration for spherical and Euclidean orbifolds is in the form of an exhaustive list. Our orbifold design system is based on the number of walls (the cardinality of the underlying polygon) in the orbifold. Thus, we strive for an *explicit* enumeration for all two-dimensional kaleidoscopic orbifolds based on the combination of the polygon cardinality and the universal cover.

There are three types of spherical orbifolds:

1. one mirror;
2. two mirrors;
3. three mirrors.

The only one mirror spherical orbifold is $*$, which corresponds to a room that is half of the sphere with its boundary being the mirror. There are no corners. In this case, one can consider the room as a monogon. In the second case, the room has two mirrors that intersect at $\frac{\pi}{k}$ at both ends where $k > 1$. These are diangular orbifolds $*kk$, which are the section of the unit sphere that are between two longitudes that are $\frac{\pi}{k}$ apart. Note that $*11$ is the same as $*$ since the corner angles are π. In fact, every corner with an angle π can be removed from the list of corners. Thus, in the orbifold notation, we require every number to be at least 2 when there are at least two walls.

For the triangular spherical orbifolds, i.e., three mirrors, there are two sub-types. The first sub-type has the form $*22k$ where $k > 1$. Figure 8.7 (a) shows one such orbifold ($*222$). This type of orbifolds can be obtained by taking half of the orbifold $*kk$ in the northern hemisphere and adding a mirror on the equator. The second sub-type has the form $*23k$ where $k = 3,4,5$. Notice that when $k = 6$ we have $*236$, a Euclidean orbifold. From the discussion, we can see there are more spherical orbifolds than Euclidean orbifolds.

There are *bad* orbifolds, namely, $*k$ where $k > 1$ and $*k_1k_2$ where $k_2 > k_1 > 1$. Note that neither type of the bad orbifolds can be realized because it is not physically possible to have one great circle intersecting itself at an angle not equal to π, nor is it possible to have two different great circles that intersect at different angles where they meet. In our system, we do not construct bad orbifolds.

N	Spherical	Euclidean	Hyperbolic
1	$*$		
2	$*22$, $*33$, $*44$, …		
3	$*222$, $*223$, $*224$ …		
	$*233$, $*234$, $*235$	$*236$	$*237$, $*238$, $*239$, …
		$*244$	$*245$, $*246$, $*247$, …
			$*2k_2k_3$ $(k_3 \geq k_2 > 4)$
		$*333$	$*334$, $*335$, $*336$, …
			$*3k_2k_3$ $(k_3 \geq k_2 > 3)$
			$*k_1k_2k_3$ $(k_3 \geq k_2 \geq k_1 > 3)$
4		$*2222$	$*k_1k_2k_3k_4$ $(\max_{1\leq i\leq 4} k_i > 2)$
5			$*k_1k_2k_3k_4k_5$
6			$*k_1k_2k_3k_4k_5k_6$
7			$*k_1k_2k_3k_4k_5k_6k_7$
⋮			⋮

Table 4.1: Our enumeration of all two-dimensional kaleidoscopic orbifolds based on the cardinality and the universal cover of the orbifolds. The orbifolds on each row have the same N, which is the cardinality of the underlying polygon.

The rest of polygonal kaleidoscopic orbifolds are hyperbolic, and there are no bad hyperbolic orbifolds. There are three cases:

1. three mirrors;

2. four mirrors;

3. five or more mirrors.

An orbifold is hyperbolic if its has five or more mirrors (e.g. Figure 8.7 (c)). In addition, all quadrangular orbifolds except $*2222$ are hyperbolic (e.g. Figure 8.7 (b)). Finally, triangular hyperbolic orbifolds include six sub-types:

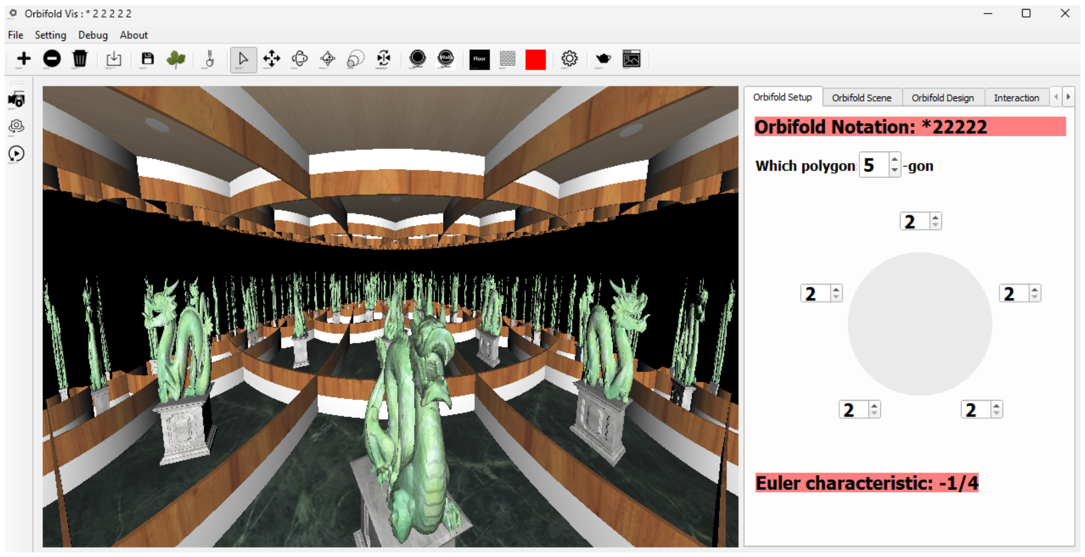
1. $*23k$ where $k > 6$;
2. $*24k$ where $k > 4$;
3. $*2k_2k_3$ where $k_3 \geq k_2 > 4$;
4. $*33k$ where $k > 3$;
5. $*3k_2k_3$ where $k_3 \geq k_2 > 3$;
6. $*k_1k_2k_3$ where $k_3 \geq k_2 \geq k_1 > 3$.

Notice the three cases, each of which corresponds to a Euclidean orbifold that serves as the border between the set of spherical and the set of hyperbolic orbifolds, namely, $*236$ for type (1), $*244$ for type (2), and $*333$ for type (4).

Our enumeration of all two-dimensional kaleidoscopic orbifolds based on the combination of the cardinality of the underlying polygon and the type of its universal cover is shown in Table 4.1. We provide the computations behind our enumeration in Chapter A.

Chapter 5: User Interface Design

Our orbifold visualization system consists of two components: a design panel and the display (Figure 5.1). We employ the Irrlicht game engine [3], which provides an effective balance between interactivity and functionality.



Orbifold Notation: *22222

Which polygon 5 -gon

Euler characteristic: -1/4

Figure 5.1: The interface of our design system. The left panel displays the universal cover of a dragon scene, while the right panel shows our design interface for arbitrary two-dimensional kaleidoscopic orbifolds. The interactive design panel can display the Euler characteristic, allowing users to gain insights into the classification of orbifolds.

In the design panel, the user can specify the type of the scene by entering its orbifold notation in the form of a number N for the number of walls in the scene and a list of N numbers, $k_1, k_2, \ldots, k_N$. Here, k_i indicates that the angle of the i-th corner is $\frac{\pi}{k_i}$.

The default value of N is five, and five evenly spaced nodes are displayed on the disk inside the design panel (Figure 5.1), each of which has a default value of two, i.e. $*22222$. The user can change the value of each node, which can be a non-integer in order to create non-orbifold scenes (Figure 8.1 (c)). The user can also change N, which results in a room with more or fewer walls. The default value for each node in the new setting is again two. Recall that there are two cases that are not physically realizable:

1. a circular room with a single mirror ($N = 1$) that self-intersects at an angle not equal to π ($*k$ where $k > 1$);
2. a room with two mirrors whose two intersection angles are mutually distinct ($*k_1k_2$ where $k_2 > k_1$).

Thus, we disallow these cases from occurring during the design phase. For example, when $N = 1$, the value of the only node is set to one and cannot be changed. Similarly, when $N = 2$, if the user changes the value of one node, the value of the other node is automatically updated to match it.

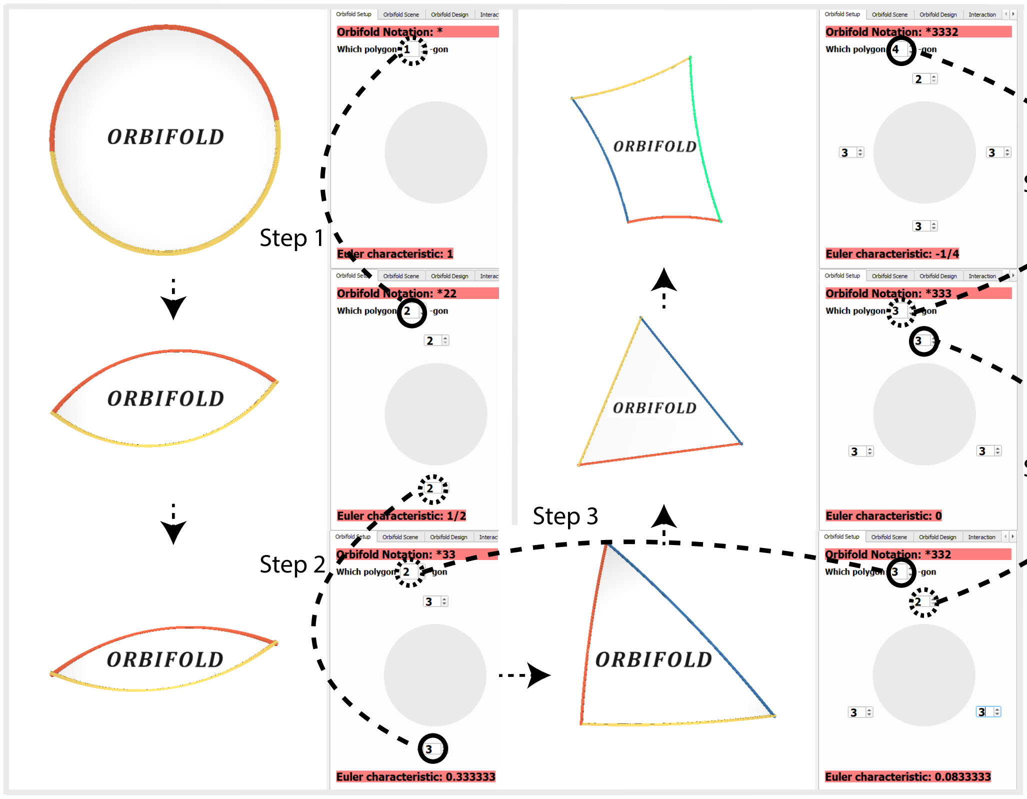
Given the orbifold notation, our system instantaneously generates an empty room (a *right* polygonal prism) whose floor and ceiling are congruent to the orbifold and whose walls are the sides of the prism. In our system, it is possible to have multiple mirrors on a wall as shown in Figure 1.2. The user can also change the height of the room, the color and attenuation of a mirror, and the textures for the ceiling and the floor. Objects can be added to the scene, whose locations, orientations, sizes, base colors, and material properties (e.g., marble, glass) can be modified from their default values as needed. Light sources can also be added to the scene, with control over their locations, shapes, and optical properties. Unwanted objects and light sources can be removed from the scene.

All of the above scene design operations are interactively rendered in order to support the *What-You-See-Is-What-You-Get* (WYSIWYG) paradigm, and all of the examples included in this dissertation were created using our design system.

5.1 2D Layout Design

The purpose of our design panel is to allow users to correlate the differences among orbifolds and follow the orbifold enumeration in Chapter 4. Additionally, we display the Euler characteristic of orbifold, which is related to its area in underlying space.

An example of the procedure for designing 2D layouts of orbifolds is shown in Figure 5.2. The left side of the figure displays the created 2D layout, which includes the orbifold logo, with each edge depicted in a different color. The example, which consists of 5 steps, illustrates the procedure through the 2D layout by the design panel: $* \to *22 \to *33 \to *233 \to *333 \to *2333$. Users can also observe the change in the value of the Euler characteristic from $1 \to \frac{1}{2} \to \frac{1}{3} \to \frac{1}{12} \to 0 \to -\frac{1}{4}$.



Orbifold Notation: *
Which polygon 1 -gon
Euler characteristic: 1

Orbifold Notation: *22
Which polygon 2 -gon
2
2
Euler characteristic: 1/2

Orbifold Notation: *33
Which polygon 2 -gon
3
3
Euler characteristic: 0.333333

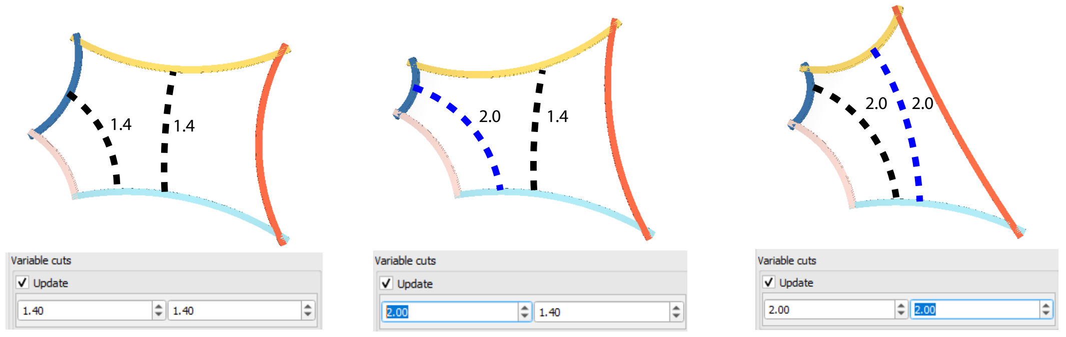
Orbifold Notation: *3332
Which polygon 4 -gon
2
3 3
3
Euler characteristic: -1/4

Orbifold Notation: *333
Which polygon 3 -gon
3
3 3
Euler characteristic: 0

Orbifold Notation: *332
Which polygon 3 -gon
2
3 3
Euler characteristic: 0.0833333

Figure 5.2: This figure shows an example of a 2D orbifold layout designed through a design panel. In step 1, we modify the polygon cardinality of $*$ from 1 to 2, creating $*22$. Moving to step 2, we change the corner number from 2 to 3, resulting in $*33$. Continuing to step 3, we alter the polygon cardinality of $*33$ from 2 to 3, generating $*233$. Step 4 involves changing the corner number from 2 to 3, which results in $*333$. Finally, in step 5, we adjust the polygon cardinality of $*333$ from 3 to 4, producing $*2333$.

As demonstrated in Chapter 6.1, when the cardinality of the underlying polygon $N \geq 4$, every cut edge in the decomposition gives rise to a free variable. Consequently, there are $N-3$ free variables for forming the polygon. Our user interface supports adjusting these free variables to generate different polygons for an orbifold. We illustrate an example of changing the free variable in a 2D layout for $*23456$ (Figure 5.3). Moving from left to right, adjusting the free variables in the spin box at the bottom of the figure automatically updates the polygon. In the left figure, the default setting for the free variable is 1.4. We initially modify the variable to 2.0 for the first cut (dashed blue edge) in the middle figure and subsequently adjust it to 2.0 for the second cut (dashed blue edge) in the right figure.



Variable cuts
✓ Update
1.40 | 1.40

Variable cuts
✓ Update
2.00 | 1.40

Variable cuts
✓ Update
2.00 | 2.00

Figure 5.3: Users can modify the free variables in the spin box to create various polygons for $*23456$. The dashed black edge represents the fixed variable, while the dashed blue edge indicates the variable that users alter.

In addition, users have the option to adjust the position of the 2D layout. As illustrated in Figure 5.4, the construction of the 2D layout for $*23456$ is based on the green geodesic, indicated by the red arrow. Users are able to relocate the polygon by dragging the red arrow to a different position (Figure 5.4, top). Moreover, rotating the red arrow allows users to rotate the polygon while keeping the position of the red arrow fixed (Figure 5.4, bottom).

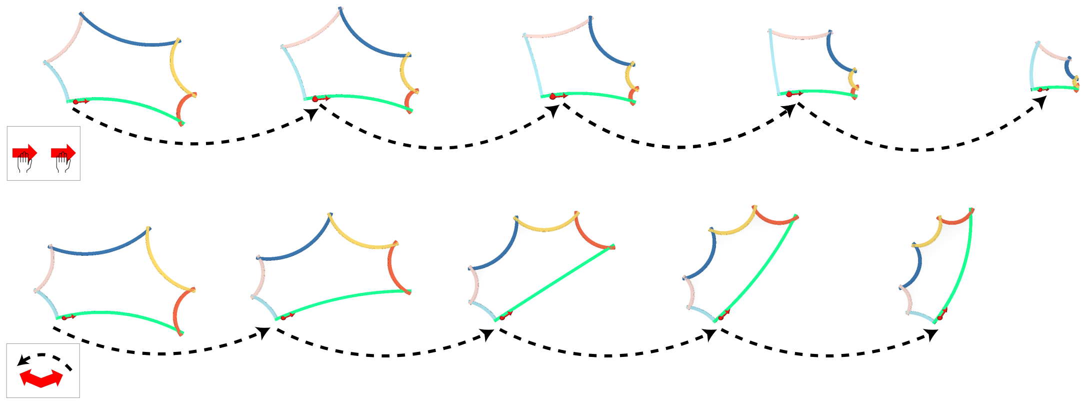
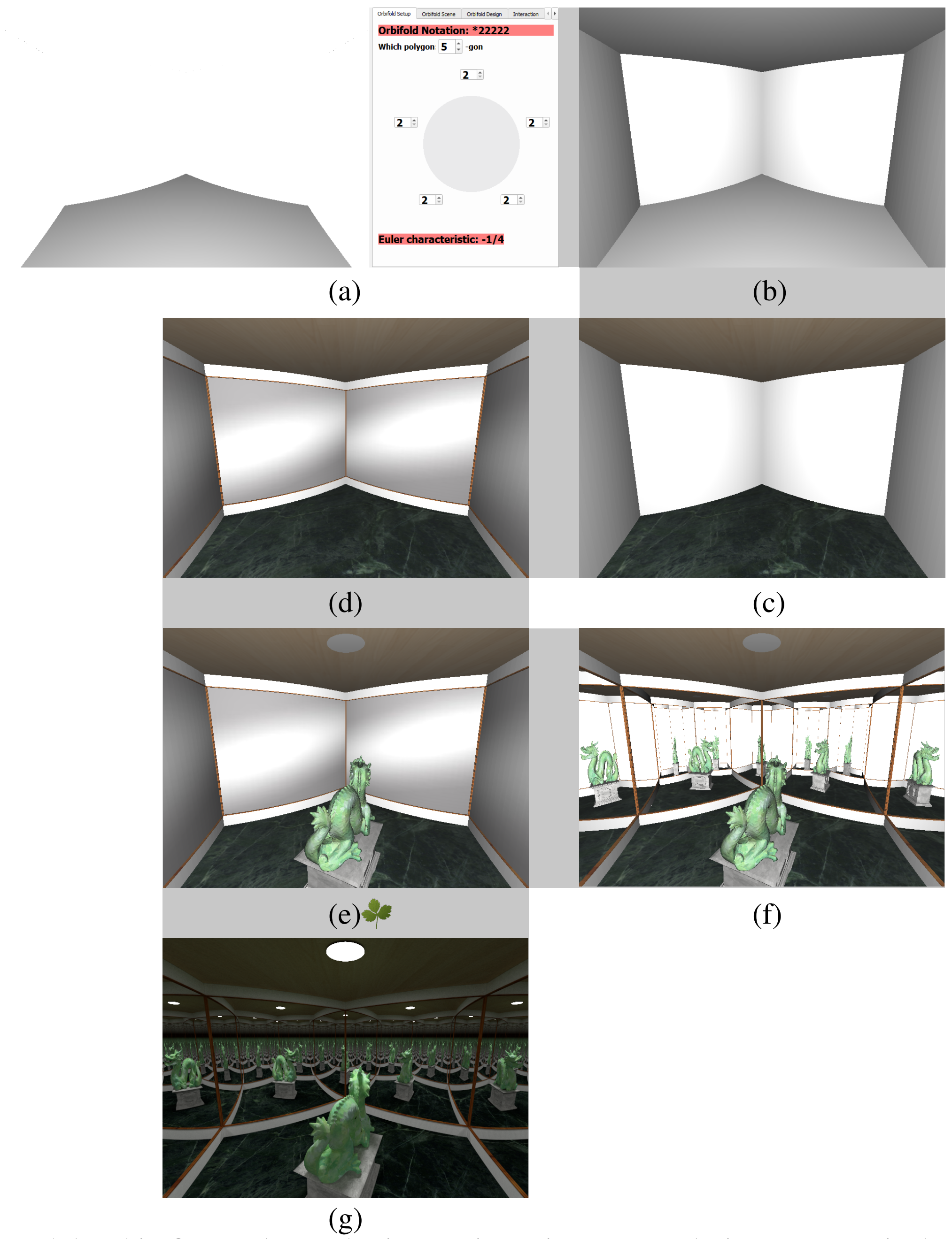


Figure 5.4: Users drag the red arrow to position the polygon for $*23456$ in a different location (top) and rotate the red arrow to orient the polygon in a different direction (bottom).

5.2 Mirror Room Design

After introducing the interface for constructing the polygon for an orbifold, we will generate a room for visualization using the mirror metaphor. In this chapter, a step-by-step example of creating a mirror room (Figure 5.5 (a-g)) is presented. In (a), the design panel creates the underlying polygon for $*22222$. This polygon serves as the floor in (b). The wall and ceiling of the room are automatically generated in (b). Users can specify the height and meshing resolution for the floor, the ceiling, and the walls. In step (c), users can apply texture or solid color to the floor and ceiling and specify the supported materials for rendering. After creating the specific floor and ceiling, users are able to add mirrors to the walls in (d). In this step, the reflectivity of each mirror can be adjusted. The mirrors can be specified to be partial and colorful, and multiple mirrors can be attached to one wall. After creating the mirror room. In (e), users can add objects and light sources, and specify the textures of the objects, the intensities, and the colors of the light sources. Additionally, users have the option to interactively transform the objects. After creating the scene, users can display the universal cover of objects in (f). The number of reflections is controlled in the panel. Additionally, the transformation will be synchronized when users transform the objects. In (g), by choosing to render the scene, the rendering engine will create a photo-realistic result of the mirror scene.

At the core of our system is the ability to create a room given any arbitrary two-dimensional kaleidoscopic orbifold and to correctly deform an object in the scene when it moves. In addition, in our interactive design system, reflections are not explicitly generated. Instead, we emulate the mirror effects by creating copies of the original room which together approximate the universal cover of the orbifold. We will provide details on each of these topics next.



Orbifold Notation: *22222
Which polygon 5 -gon
Euler characteristic: -1/4

Figure 5.5: This figure shows the interactive mirror room design process, including creating the underlying polygon for an orbifold (a), constructing the room with walls and a ceiling (b), attaching textures to the floor and ceiling (c), placing mirrors on the walls of the room (d), adding objects and light sources (e), generating an interactive universal cover of the orbifold (f), and rendering the mirror room (g).

Chapter 6: Construction of Polygons for Orbifolds

In this chapter, we present the proposed algorithms for constructing polygons for orbifolds. This includes the 2D layout algorithm for creating arbitrary non-Euclidean orbifolds (Chapter 6.1), the meshing algorithms for rooms with floor, ceiling, and walls (Chapter 6.2), and the algorithm for creating a universal cover in Shader for real-time rendering pipeline (Chapter 6.3).

6.1 2D Layout Algorithm

We first compute the Euler characteristic of the orbifold O. If $\chi(O) = 0$, i.e. a Euclidean orbifold, it is then $*2222$, $*333$, $*236$, or $*244$. As we know the internal angles and the ratios between the side lengths, we can place the vertices of the underlying triangle or square on the floor. Then, with a user-specified room height, we create the ceiling polygon by duplicating the floor polygon and raising it to match the height. Both the floor and the ceiling are represented by a triangular mesh, so are the rectangular walls in the room.

In the case of a non-Euclidean orbifold, its universal cover is either the sphere or hyperbolic space. We employ the planar model introduced in Chapter 3.1 to create a polygon for a given orbifold on the plane. The only difference in construction is that the geodesic for a spherical orbifold is identified by antipodal points (Chapter 6.1.1); however, for a hyperbolic orbifold, it is identified by symmetric points (Chapter 6.1.2). In the end of the chapter, we also introduce the construction of infinity corners for a hyperbolic orbifold (Chapter 6.1.3).

6.1.1 Spherical Orbifolds

As shown in Table 4.1, the underlying polygon of a spherical orbifold is either a monogon, a diangle, or a triangle. In all of these cases, O can be contained inside a hemisphere. That is, under the stereographic projection, it can be contained inside the unit disk in the complex plane. For the monogon, i.e. $*$, the equator is the mirror. For a diangular orbifold $*kk$ $(k > 1)$, the corner points are on the real axis which are connected by a pair of circle segments that intersect at the corner points at $\frac{\pi}{k}$. Note that the stereographic projection is conformal [17], i.e. angle-preserving, thus our choice of the angles at the intersection points. The triangular orbifold $*22k$ $(k > 1)$ is exactly half of the orbifold $*kk$ for the same k (Figure 6.1 (top)).

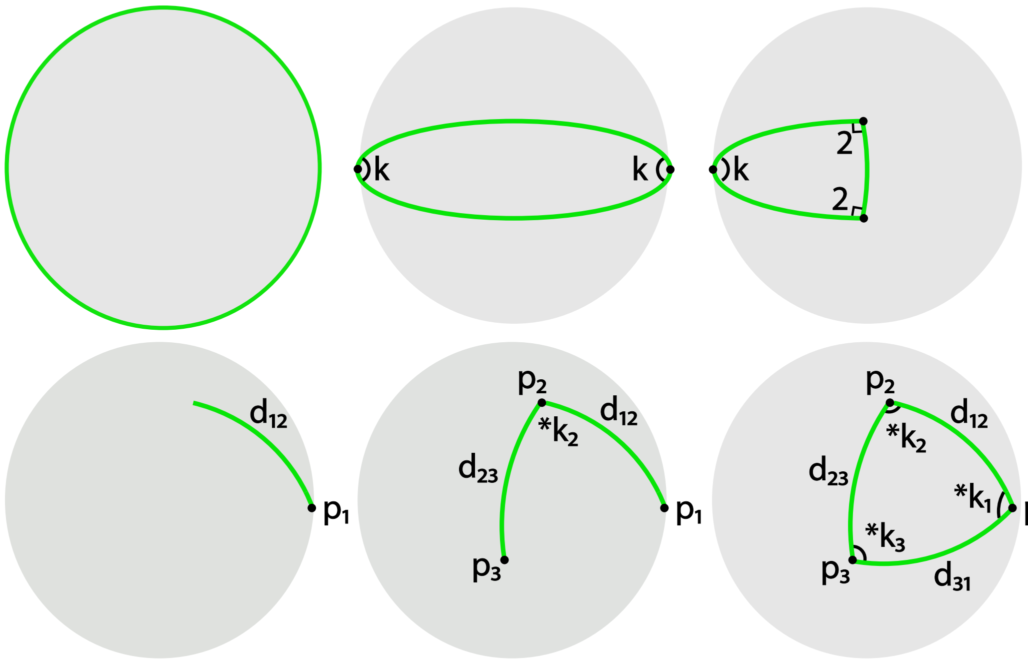


Figure 6.1: In the stereographic plane, the $*$ is realized as the unit disk (top : left). A $*kk$ orbifold can be created by placing its corner points at a pair of antipodal points on the unit disk (top : middle). The $*22k$ type of orbifold is half of the $*kk$ orbifold (top : right). The $*23k$ $(k = 3,4,5)$ can be constructed as shown in the bottom row.

Finally, the orbifolds $*23k$ $(k = 3,4,5)$ can be constructed by placing the corner points (Figure 6.1 (bottom)) in the stereographic plane as follows. Let p_1, p_2, p_3 be the corners of the orbifold corresponding to 2, 3 and k, respectively. The spherical lengths, $d_{i,i+1}$, of the edges $p_i p_{i+1}$ in the polygon are uniquely determined by the angles $\frac{\pi}{k_i}$, $\frac{\pi}{k_{i+1}}$

and $\frac{\pi}{k_{i+2}}$ of the polygon as follows [67]:

$$d_{i,i+1} = \cos^{-1}\left(\frac{\cos\left(\frac{\pi}{k_{i+2}}\right) + \cos\left(\frac{\pi}{k_i}\right)\cos\left(\frac{\pi}{k_{i+1}}\right)}{\sin\left(\frac{\pi}{k_i}\right)\sin\left(\frac{\pi}{k_{i+1}}\right)}\right) \tag{6.1}$$

where $i = 1,2,3$, $d_{3,4} = d_{3,1}$, $k_4 = k_1 = 2$, $k_5 = k_2 = 3$, $k_3 = k$, and $p_4 = p_1$. With this information, we first place the vertex p_1 at $(1,0)$ in the complex plane. Next, we compute a geodesic emanating from p_1, along which we travel for a *spherical* distance of $d_{1,2}$ to find p_2. Since p_1 and p_2 are both represented as complex numbers in the stereographic plane, their spherical distance can be computed from their complex number representations as follows [34]:

$$d(p_1,p_2) = 2\tan^{-1}\left(\left|\frac{p_2 - p_1}{1+\overline{p_1}p_2}\right|\right) \tag{6.2}$$

where $\overline{p}$ is the conjugate of a complex number p. Solving for p_2 in Equation 6.2 can be challenging given any arbitrary geodesic γ. However, on the unit circle in the stereographic plane, one can find p_2 without the need to solve Equation 6.2. This is because the unit circle corresponds to the equator in the sphere under the stereographic projection, thus the spherical distance between the two points is the same as the arc distance between them on the unit circle. However, γ is not always the unit circle. To address this, we identify a translation in the sphere, which, under the stereographic projection, takes γ (Figure 6.2 (left): the arc) to the upper-half of the unit circle. Such a translation can be modelled by Möbius transformations [11] that have the following form:

$$f(z) = e^{i\theta}\frac{z - z_0}{\overline{z_0}z + 1}. \tag{6.3}$$

Here, $\theta \in [0,2\pi)$ and z_0 and z are complex numbers. A Möbius transformation is uniquely determined by three pairs of corresponding points [34]. We first extend γ until it intersects the unit disk and map the intersection points to $(-1,0)$ and $(1,0)$, respectively (Figure 6.2 (left)). We select the third point to be middle point of γ, which is mapped to $(0,1)$. Call this Möbius transformation M. Since Möbius transformations in the stereographic plane correspond to rotating the sphere before the stereographic projection, spherical distance is preserved M. Thus, we can find the point $M(p_2)$ on the unit circle which has an arc distance of $d_{1,2}$ from $M(p_1)$. Finally, performing the inverse of M, which is also a Möbius transformation to $M(p_2)$, we can find $p_2 = M^{-1}(M(p_2))$.

From p_2, we compute a second geodesic which has an angle of $\frac{\pi}{k_2}$ with the first geodesic. Then, travelling along the new geodesic for a *spherical* distance of $d_{2,3}$, we can locate p_3. Thus, we have constructed the $*23k$ type of orbifolds (Figure 6.1: bottom row).

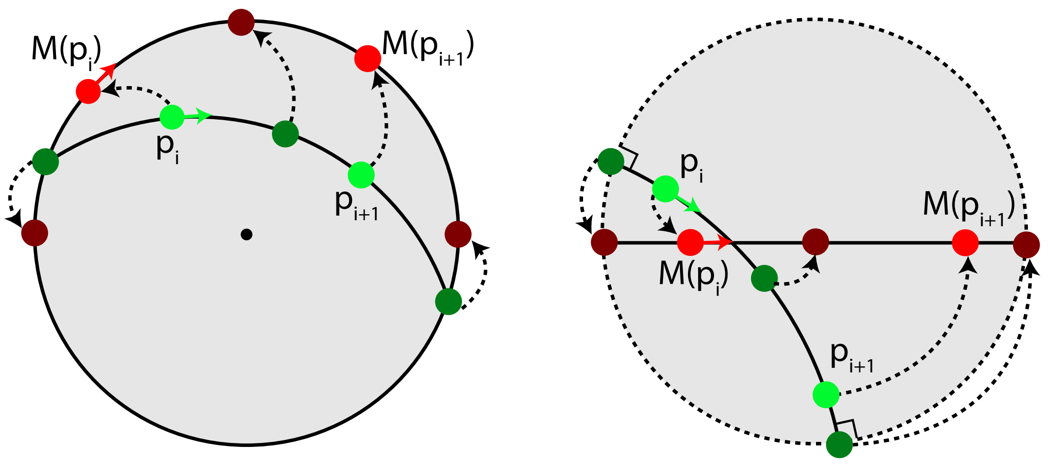


Figure 6.2: Given a point p_i, we simplify the computation of p_{i+1} by performing a Möbius transformation in the stereographic plane for spherical orbifolds (left) and the Poincaré disk for hyperbolic orbifolds (right). In both cases, the unique Möbius transformation M maps the extended geodesic (including the intersections with the unit disk) to the upper semi-circle for spherical orbifolds and line segment between $(-1,0)$ and $(1,0)$ for hyperbolic orbifolds. In addition, the center of extended geodesic is mapped to $(0,1)$ in the spherical case (left) and $(0,0)$ in the hyperbolic case (right). Then we identify $M(p_{i+1})$ from which we can recover p_{i+1} using the inverse of M.

6.1.2 Hyperbolic Orbifolds

We now consider the hyperbolic case, where $\chi(O) < 0$. The hyperbolic space can be modelled by the Poincaré disk [20], which is the interior of the unit disk in the plane.

Under this model, the geodesics of the hyperbolic space are circles that intersect the boundary of unit disk at the right angle (Figure 3.3 (right)). Recall that a hyperbolic orbifold is a polygon with three or more sides (Table 4.1). To create such a polygon, we follow the same approach for spherical orbifolds. That is, we start with the location of p_1 in the Poincaré disk and a geodesic emanating from p_1. We then travel along this geodesic for a prescribed distance to locate p_2. From there, we identify a new geodesic whose angle with the original geodesic is $\frac{\pi}{k_2}$, which we follow to identify p_3. This process terminates once when we have identified p_N. The main difference lies in the fact that the hyperbolic distance between two points in the Poincaré disk is different from the spherical distance of the same two points in the stereographic plane.

Given a triangular hyperbolic orbifold $*k_1k_2k_3$, the length of the edge between p_i and p_{i+1} ($i = 1,2,3$, $d_{3,4} = d_{3,1}$, $k_4 = k_1$, $k_5 = k_2$, and $p_4 = p_1$) is given by [67]

$$d_{i,i+1} = \cosh^{-1}\left(\frac{\cos\left(\frac{\pi}{k_{i+2}}\right) + \cos\left(\frac{\pi}{k_i}\right)\cos\left(\frac{\pi}{k_{i+1}}\right)}{\sin\left(\frac{\pi}{k_i}\right)\sin\left(\frac{\pi}{k_{i+1}}\right)}\right) \tag{6.4}$$

In addition, when represented as complex numbers in the plane containing the Poincaré disk, the hyperbolic distance between p_i and p_{i+1} is given by [34]:

$$d(p_i, p_{i+1}) = \ln\left(\frac{|1-\overline{p_i}p_{i+1}| + |p_{i+1}-p_i|}{|1-\overline{p_i}p_{i+1}| - |p_{i+1}-p_i|}\right) \tag{6.5}$$

Similar to the case of spherical orbifolds, finding p_{i+1} from p_i on an arbitrary geodesic γ in the Poincaré disk requires solving Equation 6.5 for p_{i+1} which can be challenging. To simplify the matter, we identify the hyperbolic translation in the Poincaré disk that takes γ (Figure 6.2 (right): the arc) to the line segment from $(-1,0)$ to $(1,0)$ in the Poicnaré disk (Figure 6.2 (right)). This translation takes the intersection with the unit circle (two dark green points on the unit circle) to $(1,0)$ and $(-1,0)$ (the rightmost and the leftmost burgundy points). It also takes the middle points on γ (the middle dark green point) to $(0,0)$ (the middle burgundy point). Such a translation can be modelled by a Möbius transformation of the following form:

$$f(z) = e^{i\theta} \frac{z - z_0}{1 - \overline{z_0} z} \tag{6.6}$$

where $\theta \in [0, 2\pi)$ and $|z_0| < 1$. Call the translation M. Since translations maintain hyperbolic distances, the hyperbolic distance between p_i and p_{i+1} is the same as the distance between $M(p_i)$ and $M(p_{i+1})$. However, since $M(p_i)$ and $M(p_{i+1})$ are on the real axis, it is easier to solve Equation 6.5. Once we have found $M(p_{i+1})$, we can recover p_{i+1} by applying M^{-1}.

However, deciding the side lengths of a hyperbolic polygon with at least four edges is more challenging as there are no published formulas to the best of our knowledge. To address this, we compute the side lengths based on two facts [51]: (1) any quadrangular hyperbolic polygon can be decomposed into the disjoint union of at most two quads with two right angles ($*22k_3k_4$), and (2) any hyperbolic polygon with at least five sides can be decomposed into the disjoint union of a finite number of quads of the type $*22k_3k_4$ and pentagons of the type $*2222k_5$, i.e. four right angles. Examples of the two facts are shown in Figure 6.3.

The side lengths of the $*22k_3k_4$ polygon (Figure 6.4 (left)) are given by [67, 51]:

$$\begin{aligned}
d_{2,3} &= \sinh^{-1}\left(\frac{\cos\left(\frac{\pi}{k_4}\right) + \cos\left(\frac{\pi}{k_3}\right)\cosh\left(d_{1,2}\right)}{\sin\left(\frac{\pi}{k_3}\right)\sinh\left(d_{1,2}\right)}\right)\\
d_{3,4} &= \cosh^{-1}\left(\frac{\cosh\left(d_{1,2}\right) + \cos\left(\frac{\pi}{k_3}\right)\cos\left(\frac{\pi}{k_4}\right)}{\sin\left(\frac{\pi}{k_3}\right)\sin\left(\frac{\pi}{k_4}\right)}\right)\\
d_{4,1} &= \sinh^{-1}\left(\frac{\cos\left(\frac{\pi}{k_3}\right) + \cos\left(\frac{\pi}{k_4}\right)\cosh\left(d_{1,2}\right)}{\sin\left(\frac{\pi}{k_4}\right)\sinh\left(d_{1,2}\right)}\right)
\end{aligned} \tag{6.7}$$

where $d_{1,2}$, the length of the cut edge in the decomposition, is a free variable. This is similar to the case where the width and length of the $*2222$ orbifold (a rectangle) are free variables.

A generic quadrangular orbifold ($*k_1k_2k_3k_4$) can be decomposed into the disjoint union of two quads ($*22k_1k_2$) and ($*22k_3k_4$) (Figure 6.3, left). This allows us to com-

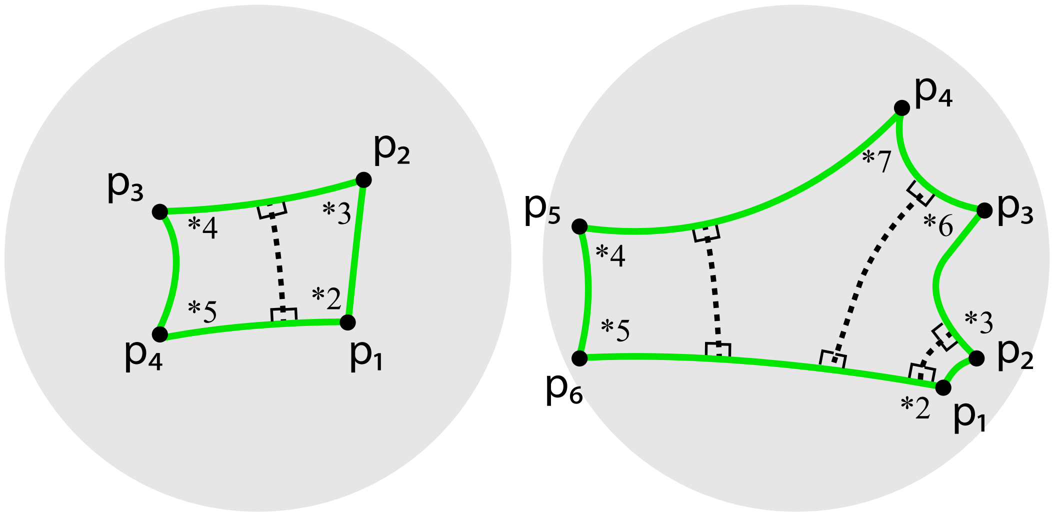


Figure 6.3: Two example scenarios of our decomposition algorithm for hyperbolic orbifolds: a quad is divided into two $*22k_3k_4$ type quads (left), and a hexagon is divided into two $*22k_3k_4$ quads and two $*2222k_5$ pentagons (right).

pute side lengths of the two special quads, which, when combined, give rise to the side lengths of the generic quad.

The side lengths of the $*2222k_5$ polygon (Figure 6.4 (right)) can computed as follows [67, 51]:

$$\begin{aligned} d_{1,2} &= \cosh^{-1}\left(\frac{\cos\left(\frac{\pi}{k_5}\right)+\cosh\left(d_{2,3}\right)\cosh\left(d_{5,1}\right)}{\sinh\left(d_{2,3}\right)\sinh\left(d_{5,1}\right)}\right) \\ d_{4,5} &= \sinh^{-1}\left(\frac{\cosh\left(d_{2,3}\right)+\cos\left(\frac{\pi}{k_5}\right)\cosh\left(d_{5,1}\right)}{\sinh\left(d_{5,1}\right)\sin\left(\frac{\pi}{k_5}\right)}\right) \\ d_{3,4} &= \sinh^{-1}\left(\frac{\cosh\left(d_{5,1}\right)+\cos\left(\frac{\pi}{k_5}\right)\cosh\left(d_{2,3}\right)}{\sinh\left(d_{2,3}\right)\sin\left(\frac{\pi}{k_5}\right)}\right) \end{aligned} \tag{6.8}$$

where $d_{2,3}$ and $d_{5,1}$, the cut edges in the decomposition, are free variables.

Generic pentagons and higher-order N-sided polygons can be decomposed as the union of $N-4$ pentagons of the type $*2222k_5$ with up to two additional quads of the type $*22k_1k_2$. To do so, we first consider the simplest case where there exist $k_i = k_{i+1} = 2$ and $k_j = k_{j+1} = 2$ where $i, i+1, j, j+1$ are mutually distinct. We can find a geodesic that intersects $p_{i-1}p_i$ and $p_{i+1}p_{i+2}$ at the right angle. This geodesic removes from the original polygon a pentagon involving p_i, p_{i+1}, and p_{i+2} which has four right internal angles, thus the type $*2222k_5$. The remaining polygon has one fewer vertex and still has four internal right angles (Figure 6.3). Repeating this process can lead to $N-4$ pentagons of the type $*2222k_5$. On the other hand, a generic polygon with at least five edges can be reduced into the simpler setting by removing up to two quads of the type $*22k_3k_4$. We can then compute the side lengths of each of the sub-polygons, which, when combined, give the side lengths of the original polygon.

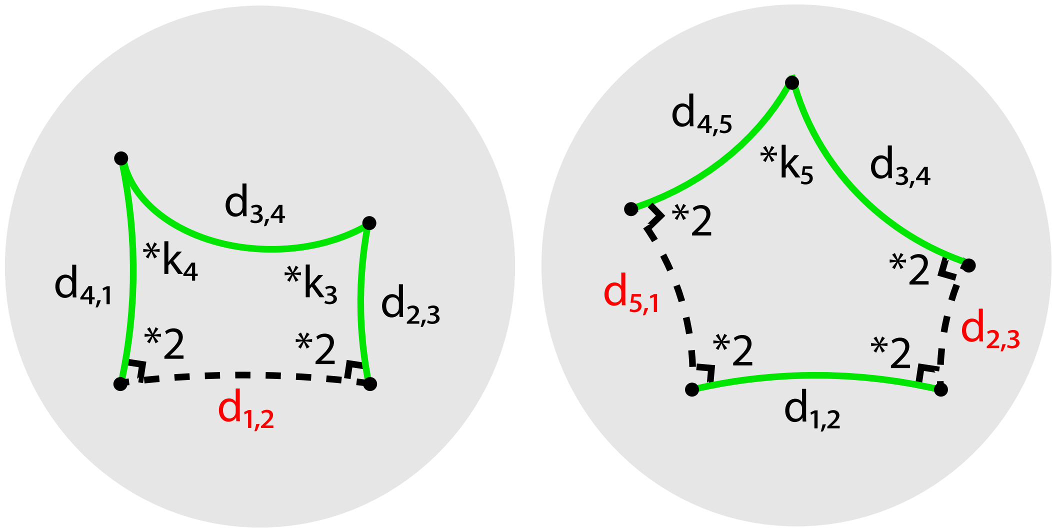


Figure 6.4: The special quad $*22k_3k_4$ (left) and the special pentagon $*2222k_5$ (right) are the building blocks of our decomposition algorithm. The free variables are colored in red, which correspond to the edges (dashed) introduced during the decomposition.

Every cut edge in the decomposition gives rise a free variable, which can be modified by the user. The default value for the free variables is set to 1.4.

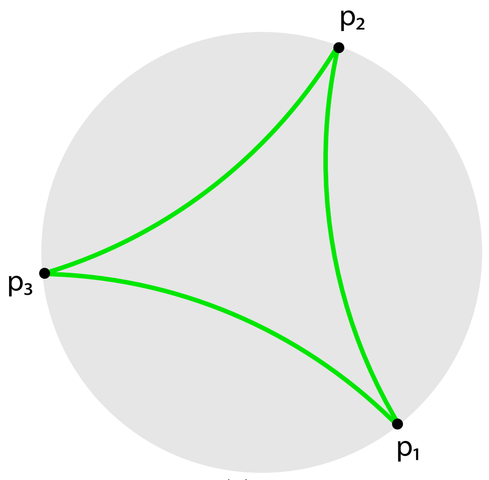
With our algorithm, any spherical and hyperbolic orbifold can be constructed given its orbifold notation. Figure 6.11 show two example orbifolds with their universal covers.

6.1.3 Infinity Corners

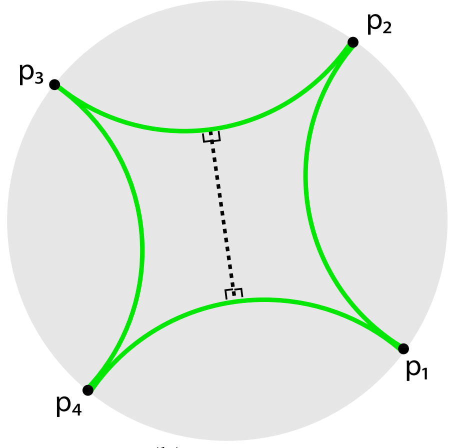
In hyperbolic orbifolds, where k can be ∞, the vertex at the corner is infinitely distant in hyperbolic space. Our algorithm in Chapter 6.1.2 is also adapted to construct orbifolds with infinity corners. However, the following modifications are made: the formulation for calculating the length of edges in Equation 6.4, Equation 6.7, and Equation 6.8 are invalid when $k = \infty$, as shown in the last row of Table 6.1. The last column in Table 6.1 indicates the position to which a geodesic in $(0,0)$ advances with the distance of c_0 in the real axis. From the table, we observe that when $k \geq 10^6$, the geodesic approaches the boundary of the disk with a radius of 1. Therefore, to cope with infinity corners, we use a large value $k = 10^6$ to prevent the calculations from being invalid in the above equations. We present the examples of the construction of infinity corners in Figure 6.5, where each vertex lies on the boundary of the Poincaré disk.

k	$a = \sin \frac{\pi}{k}$	$b = \frac{1}{a}$	$c_0 = \sinh^{-1}(b)$	$\cosh^{-1}(b)$	$d = \frac{e^{c_0}-1}{e^{c_0}+1}$
10^1	0.309017	3.23607	1.89057	1.84273	0.73764
10^2	$3.14108 * 10^{-2}$	$3.18362 * 10^1$	4.154	4.15351	0.969082
10^3	$3.14159 * 10^{-3}$	$3.1831 * 10^2$	6.45618	6.45617	0.996863
10^4	$3.14159 * 10^{-4}$	$3.1831 * 10^3$	8.75876	8.75876	0.999686
10^5	$3.14159 * 10^{-5}$	$3.1831 * 10^4$	11.0613	11.0613	0.999969
10^6	$3.14159 * 10^{-6}$	$3.1831 * 10^5$	13.3639	13.3639	0.999997
10^7	$3.14159 * 10^{-7}$	$3.1831 * 10^6$	15.6665	15.6665	1
10^8	$3.14159 * 10^{-8}$	$3.1831 * 10^7$	17.9691	17.9691	1
10^9	$3.14159 * 10^{-9}$	$3.1831 * 10^8$	20.2717	20.2717	1
10^{10}	$3.14159 * 10^{-10}$	$3.1831 * 10^9$	22.5743	22.5743	1
∞	∞	∞	∞	∞	*NaN*

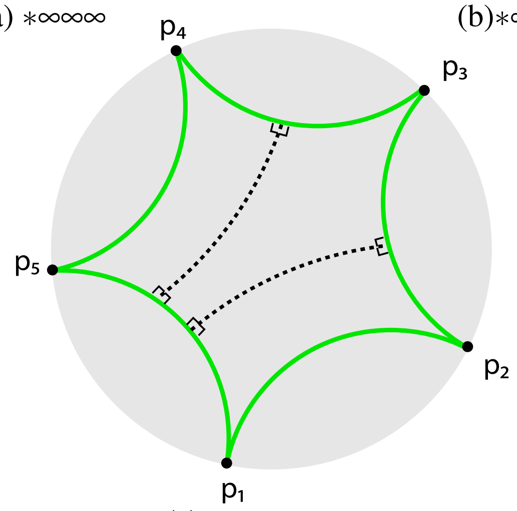
Table 6.1: Calculating the accuracy of infinite corners. The first row shows the formulation for calculations, and the numerical accuracy is 6 digits. *NaN* is not a number.



(a) $*\infty\infty\infty$



(b)$*\infty\infty\infty\infty$



(c) $*\infty\infty\infty\infty\infty$

Figure 6.5: Our algorithm constructs the infinity corners represented by $*\infty\infty\infty$ (a), $*\infty\infty\infty\infty$ (b), and $*\infty\infty\infty\infty\infty$ (c).

6.2 Meshing of Floor, Ceiling, and Walls

While polygons can be constructed for arbitrary orbifolds, the polygons need to be tessellated with sufficient resolution in order to generate high-quality rendering results. In this chapter, we first introduce the triangulation of floors and ceilings in Chapter 6.2.1, after that, we demonstrate the triangulation of walls in Chapter 6.2.2 for producing mirror rooms, and show the meshed results of rooms in Chapter 6.2.3.

6.2.1 Floor and Ceiling Tessellation

A floor and a ceiling are an orbifold, which is modeled by a polygon bounded by geodesics in the stereographic plane or the Poincaré disk. Given the vertices of a polygon defining a kaleidoscopic orbifold in the plane. Numerous traditional triangulation methods exist for polygons in Euclidean space [15, 47, 25]. However, these approaches either fail to account for the geometry in non-Euclidean spaces or are unable to maintain a smooth contour for polygons of non-Euclidean orbifolds. We first describe the fundamental process of constructing a regular grid-based tessellation in the Euclidean plane. Subsequently, we use this tessellation to form regular quadrilaterals (Figure 6.6), followed by the triangulation of these quads. Although the above approach successfully preserves the edges of non-Euclidean polygons, the vertices along the lines do not lie on the geodesics in non-Euclidean spaces, leading to less regular triangulation in such spaces.

We first present a geodesics-based tessellation that achieves a more regular pattern in non-Euclidean spaces, as detailed in Chapter 6.2.1.1. Nonetheless, the geodesics-based tessellation cannot preserve the contour for highly complex polygons. A contour-based tessellation is designed to maintain the contours of complex polygons, as discussed in Chapter 6.2.1.2.

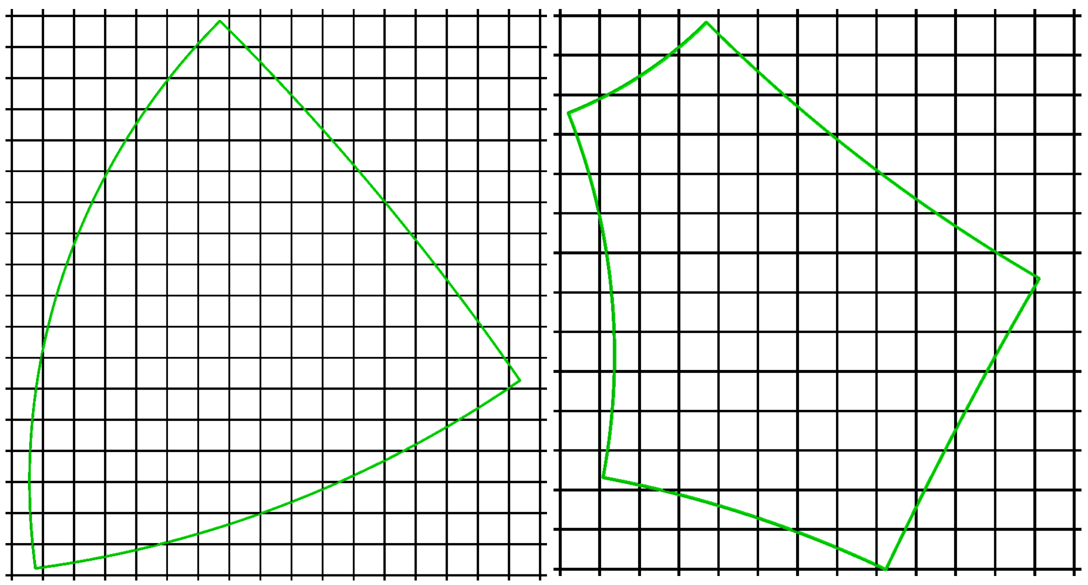


Figure 6.6: We apply a grid-based tessellation to the spherical orbifold $*222$ (left) and the hyperbolic orbifold $*22222$ (right). The green lines represent the polygons, while the black lines are the tessellations.

6.2.1.1 Geodesics-based Tessellation

To build a tessellation, a quad-dominant mesh is first generated which we then triangulate. To generate the mesh, we consider two families of geodesics that are mutually perpendicular. For a spherical polygon, we consider four evenly spaced points on the unit disk in the stereographic plane. The four points form two pairs of antipodal points P, Q and S, T (Figure 6.7 (top)). We then generate a family of geodesics that pass through P and Q and another family of geodesics that pass through S and T. We then extract the intersections of the two families of geodesics and keep those that are inside the orbifold. In addition, the intersection of these geodesics and the boundary of the orbifolds are also extracted. Using these intersection points, we build a quad-dominant mesh which we triangulate. For a hyperbolic orbifold, we also consider two families of geodesics. Again, we choose four evenly-spaced points on the boundary of the unit disk that form two pairs of symmetric points P, Q and S, T (Figure 6.7 (bottom)). For the pair P, Q, we generate a family of geodesics by locating a point P' to the left of P and a point Q' left of

Q such that the arc distances $\overset{\frown}{PP'} = \overset{\frown}{QQ'}$. Similarly, we can select P' and Q' to be on the right side of P and Q, respectively. By generating a family of such P' and Q' with even distance, we have created a family of geodesics that are parallel to the geodesic PQ. We can create the second family of geodesics by repeating the same procedure on S and T. The intersections between the two families of geodesics and between the geodesics and the boundary of the orbifold O provide the set of sample points that can serve as the vertices of the quad-dominant mesh, which we triangulate. In the example shown

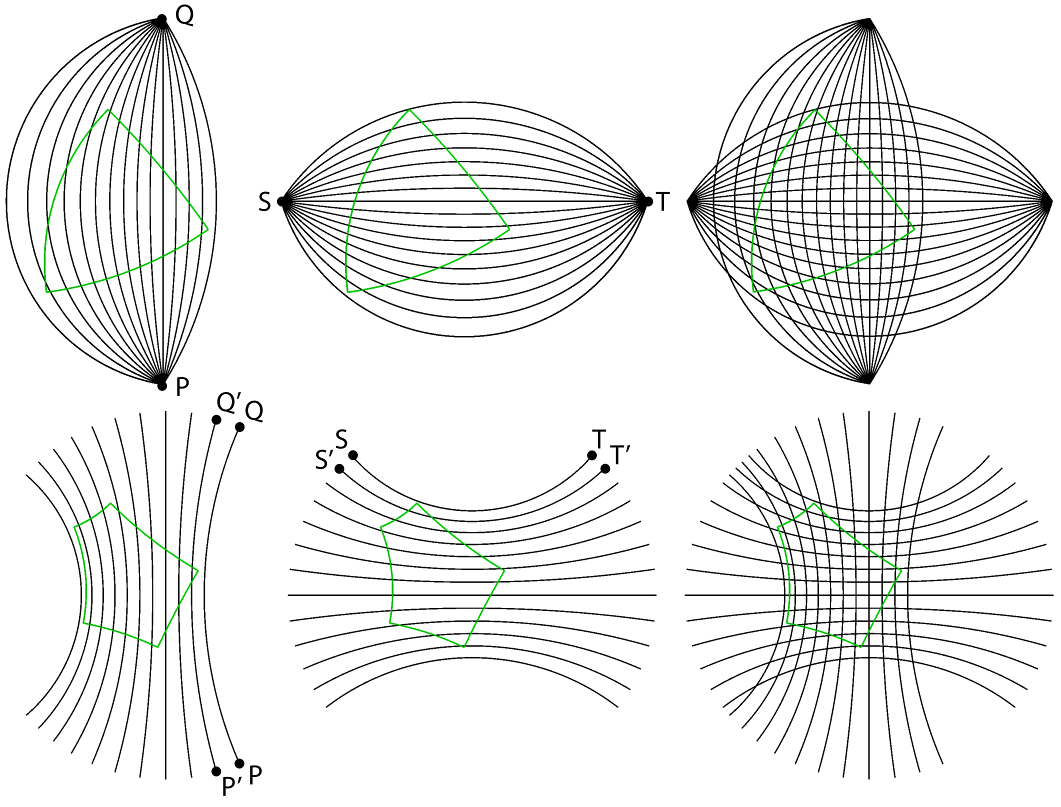


Figure 6.7: The creation of quad-dominant mesh is depicted in this figure. For the spherical polygon (top), the geodesic family: P and Q (left), the family: S and T (center), and the intersections between the geodesic family: PQ, ST (right). For the hyperbolic polygon (bottom), the geodesic family: PQ and $P'Q'$ (left), the family: ST and $S'T'$ (center), and the intersections between the geodesic family: PQ, $P'Q'$, ST, and $S'T'$ (right).

in Figure 6.8, the geodesics-based approach exhibits a loss of resolution in the complex corners, where the tessellations cannot form a quad. Therefore, we introduce a faster contour-based approach for creating tessellations in the next chapter.

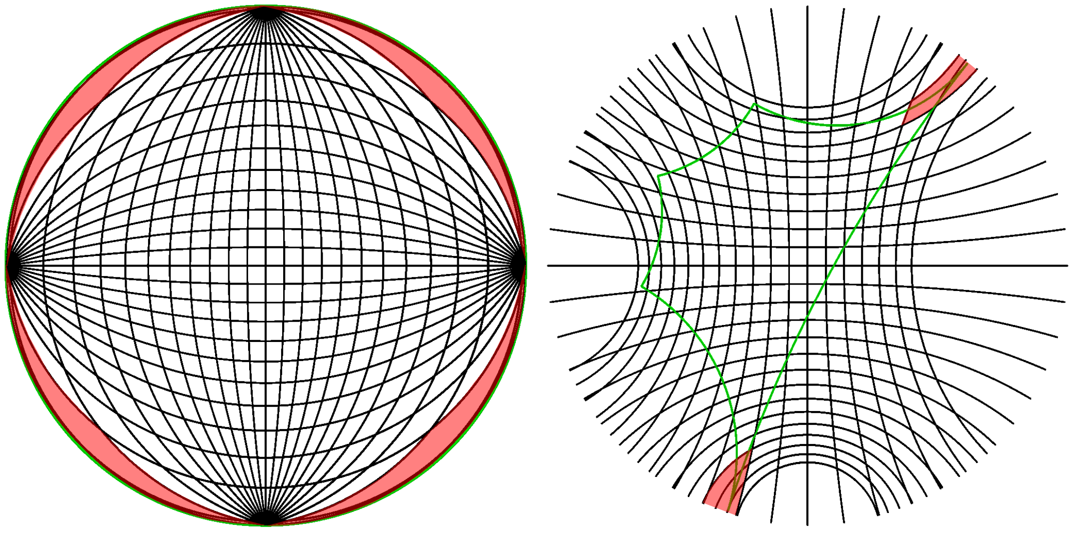


Figure 6.8: This figure demonstrates the poor results of the geodesics-based tessellations for the spherical orbifold $*$ (left) and the hyperbolic orbifold $*222\,\text{inf}\,\text{inf}$ (right). The red portions highlight the contours that cannot be covered by the quads.

6.2.1.2 Contour-based Tessellation

To maintain the contour of a complex polygon, a set of points are first sampled along each edge of the polygon, including the vertices. In such a case, all the samples form a loop (Figure 6.9 (left)). After that, we identify the middle sample, which, when coupled with the first sample, divides all samples into top and bottom sides. Consequently, the geodesic connecting the samples from left to right in the top side and the bottom side is obtained, forming a set of geodesics (Figure 6.9 (center)). In the specific case of $*$, where the polygon is a unit circle, all samples are located on the circumference of the unit circle, and all geodesics coincide with the unit circle. In this special case, we employ straight lines instead of geodesics. By discretizing the geodesics with a fixed length of step, we build a set of quads for triangulation (Figure 6.9 (right)). In comparison to the geodesics-based approach, the contour-based approach is faster (without checking geodesics-based intersections), albeit with some loss of regularity in one direction. Nevertheless, it effectively preserves the contour of polygon.

6.2.2 Wall Tessellation

A wall is essentially the tensor product of a line segment (height) and an arc (geodesic in non-Euclidean spaces). Thus, we tessellate the line segment and the arc separately. A set of evenly spaced sample points are placed on the line segment as well as the arc. However, the distance on the arc is measured not be the distance in the stereographic plane and the Poincaré disk. Instead, we measure the distance between adjacent sample points by the true distances in spherical space (Equation 6.2) and hyperbolic space (Equation 6.5). The tensor product of both tessellated line segment and arc result in a quad mesh, which we triangulate.

6.2.3 Results of Mirror Rooms

Following the triangulation of the floor, ceiling, and walls, we present mirror rooms corresponding to orbifolds in Figure 6.10. In the first row of Figure 6.10, we illustrate

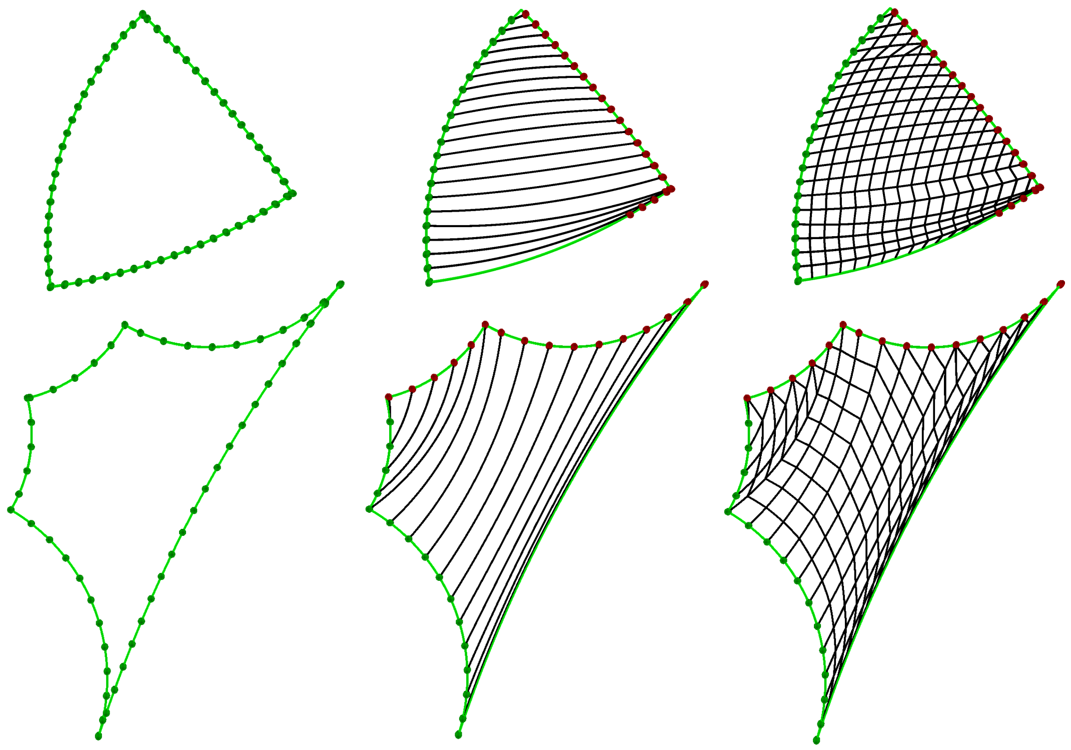


Figure 6.9: This figure illustrates the steps of contour-based tessellation for the spherical polygon $*222$ (top) and the hyperbolic polygon $*222\,\mathrm{inf}\,\mathrm{inf}$ (bottom). The green dots represent samples on the edges of the polygons (left), while the center image depicts the geodesics connecting the pairs of the red dots and the green dots. The final step involves forming a set of geodesics between the adjacent geodesics to create the quad-dominant tessellation.

the rooms corresponding to the spherical orbifolds: $*$, $*22$, and $*222$. The second row of Figure 6.10 shows the rooms corresponding to the Euclidean orbifolds: $*236$, $*333$, and $*2222$. In the third row of Figure 6.10, we present three examples of the hyperbolic orbifolds: $*237$, $*2223$, and $*22222$.

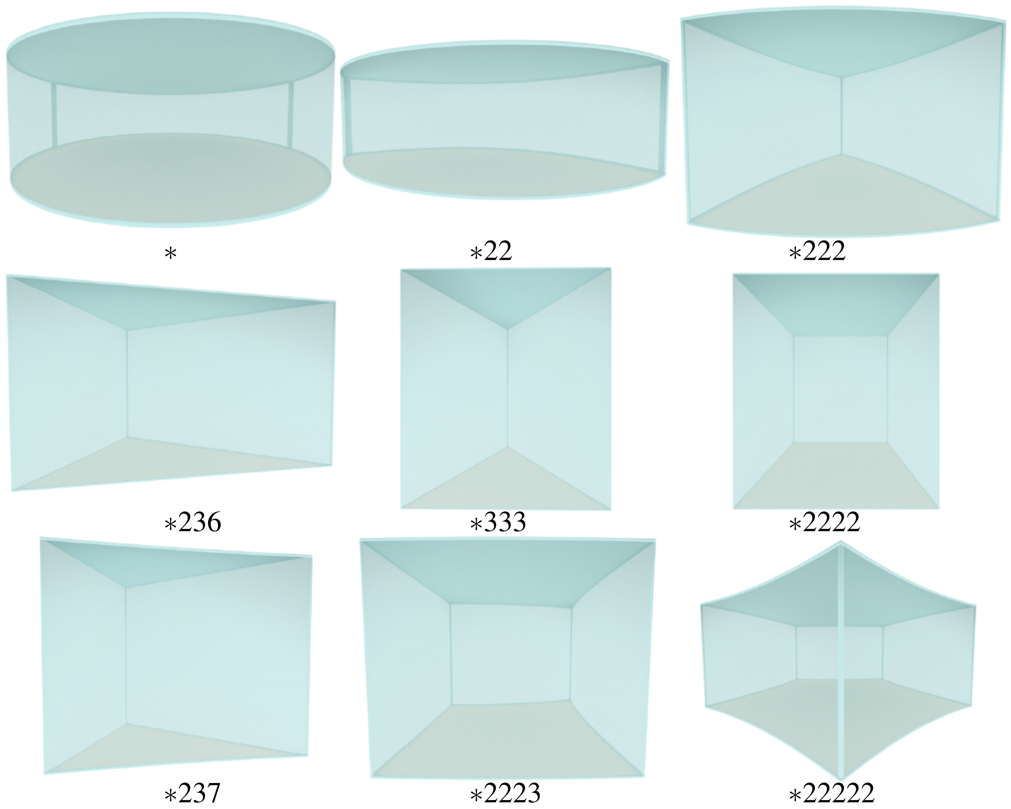


Figure 6.10: Mirror rooms are displayed for spherical orbifolds (top), Euclidean orbifolds (middle), and hyperbolic orbifolds (bottom). The floors are depicted in peach color, the walls are represented in light blue, and the ceilings are indicated in dark blue.

6.3 Universal Cover Construction

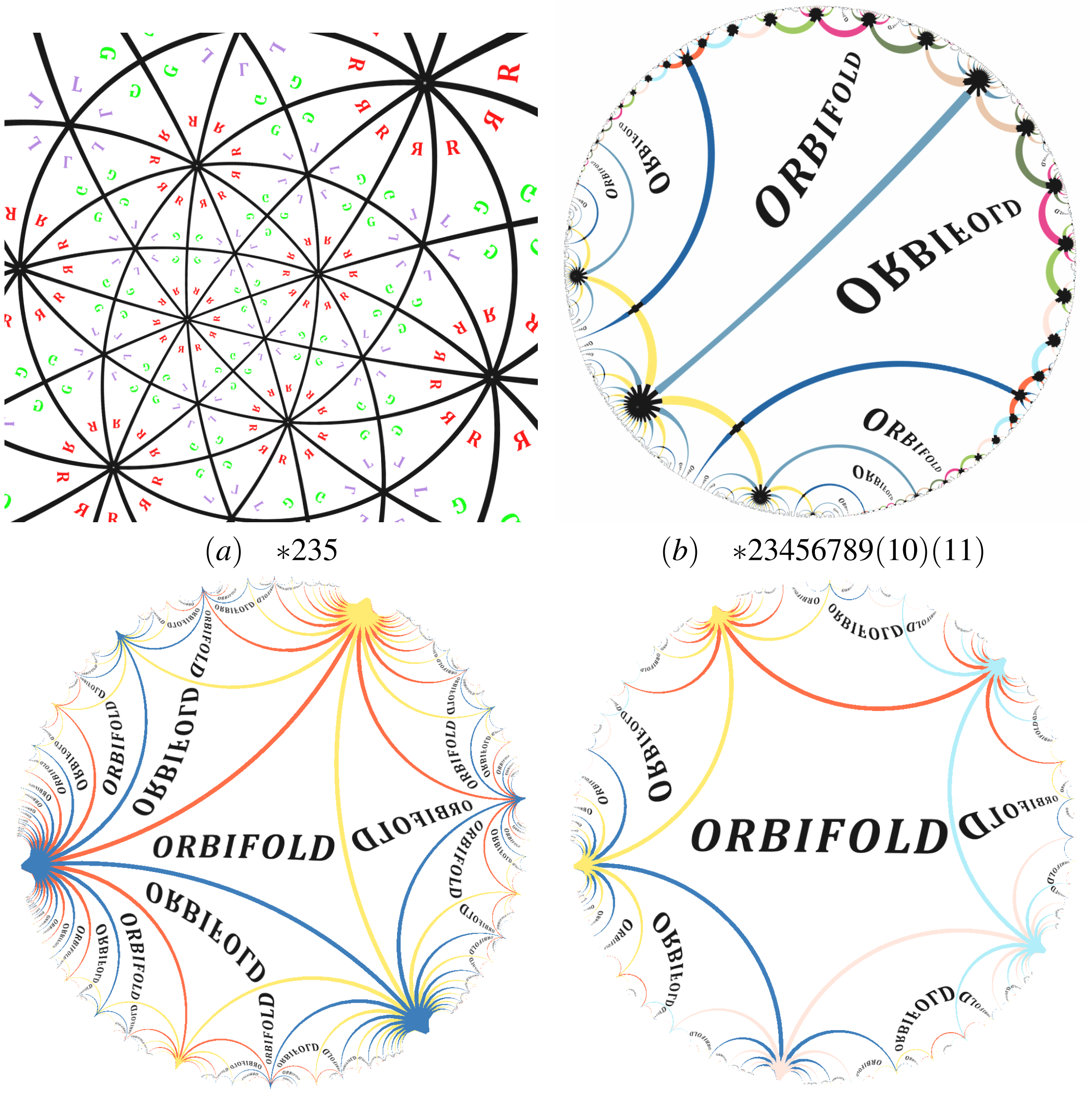
Being able to see the universal cover, i.e. all mirror reflections, can be important for a user while exploring our tool. However, capturing mirror reflections can be computationally expensive with high-quality renderers such as Mitsuba [39]. Williams and Zhang [81] address this for Euclidean kaleidoscopic orbifolds by creating a finite number of copies of the reflections of the original room that approximate the universal cover. This is based on two observations. First, as copies are farther away from the room, their images perceived by the viewer approach the vanishing line and thus do not contribute much to the pixels. Second, the color intensity of faraway copies diminishes as the number of bounces off from the mirrors increases. We employ a similar approach, which focuses on non-Euclidean orbifolds. Four examples of universal covers created by our method are shown in Figure 6.11: (a) the spherical orbifold $*235$, (b) the hyperbolic orbifold $*23456789(10)(11)$, (c) the hyperbolic orbifold $*\infty\infty\infty\infty$, and (d) the hyperbolic orbifold $*\infty\infty\infty\infty\infty\infty\infty\infty$.

In our system, the construction of the universal cover for Euclidean orbifolds closely follows that of Williams and Zhang [81], which computes the translational cover of the orbifold and generates additional copies of the translational cover using either the Gaussian integer lattice $\mathbb{Z}[i]$ for the $*2222$ and $*442$ cases and the Eisenstein integer lattice $\mathbb{Z}[\omega]$ for the $*333$ and $*632$ cases.

For spherical and hyperbolic orbifolds, the notion of translational cover is not well-defined. Consequently, we employ the following process. Starting from the original room, we iteratively add a virtual copy by reflecting the room across one of its mirrors. It is also possible to reflect a virtual room across its mirror. To avoid duplicates, i.e. a virtual room that is discovered through two different paths from the original room, we compare the center of a potentially new room to the centers of already visited room and virtual rooms [83].

To locate the corners of each newly added room, we apply reflections in the spherical or hyperbolic spaces to the already visited room involved in the reflection. Inside the stereographic plane, a reflection in the spherical space can be represented as the compositions of some Möbius transformations (Equation 6.3) and the conjugation function with respect to the real axis $f(z) = \bar{z}$. Similarly, inside the Poincaré disk, a reflection in the hyperbolic space can also be expressed as the composition of some Möbius transformation (Equation 6.6) and the conjugation function with respect to the real axis. The Möbius transformation is stored in the form of a 2×2 matrix whose entries consist of a, b, c, and d [10].

For each copy of the room we save the transformation needed to take the original room to the copy, which is a combination of a Möbius transformation and up to one conjugation function. In addition, for each object in the original room, we also save its Möbius transformation. Then, the position and orientation of an object in a virtual copy of the original scene can be calculated by combining the matrix for the virtual room and the matrix of the same object in the original scene. Also, the computation for both object manipulation and universal cover construction is done using the shader.



(a) $*235$ (b) $*23456789(10)(11)$

(c) $*\infty\infty\infty\infty$ (d) $*\infty\infty\infty\infty\infty\infty\infty\infty$

Figure 6.11: Four orbifolds generated using our orbifold layout creation algorithm: (a) the $*235$ orbifold (spherical), (b) the orbifold $*23456789(10)(11)$ (hyperbolic), (c) the orbifold $*\infty\infty\infty\infty$ (hyperbolic), and (d) the orbifold $*\infty\infty\infty\infty\infty\infty\infty\infty$ (hyperbolic).

Chapter 7: Interacting with Orbifolds

Once the orbifold has been realized as a polygonal prism, the user can add objects to the scene. We present the algorithm for embedding an object in non-Euclidean spaces in Chapter 7.1 and translating an object in Chapter 7.2.

7.1 Object Embedding

When bringing an object, which is created in a presumably Euclidean space, to a non-Euclidean space, a natural question to ask is how to perform the embedding. Due to the difference in their respective distance metrics, it is not always possible to embed the model in such a way that the length of every edge in the mesh is maintained. To address this challenge, when embedding the object into the scene, we first place it so that its center of mass is at the origin of the plane for both the stereographic plane and the Poincaré disk. The coordinates of the object are now considered their corresponding coordinates in the stereographic plane and the Poincaré disk. Then, the embedded mesh is translated to the user-specified initial location with the translation native to the non-Euclidean space.

As shown in Figure 7.2, the mass center of the dragon (red dot) is positioned at the center of the stereographic plane (top) and the Poincaré disk (bottom). From the top view in the third column, we can observe that the mass center is at the center of the spherical orbifold $*222$ and at the center of the hyperbolic orbifold $*22222$.

7.2 Object Movement

Translations in both the spherical and hyperbolic spaces are isometries. In the spherical space, translations can be modelled in the stereographic plane by Möbius transformations according to Equation 6.3. Interestingly, translations in the hyperbolic space using

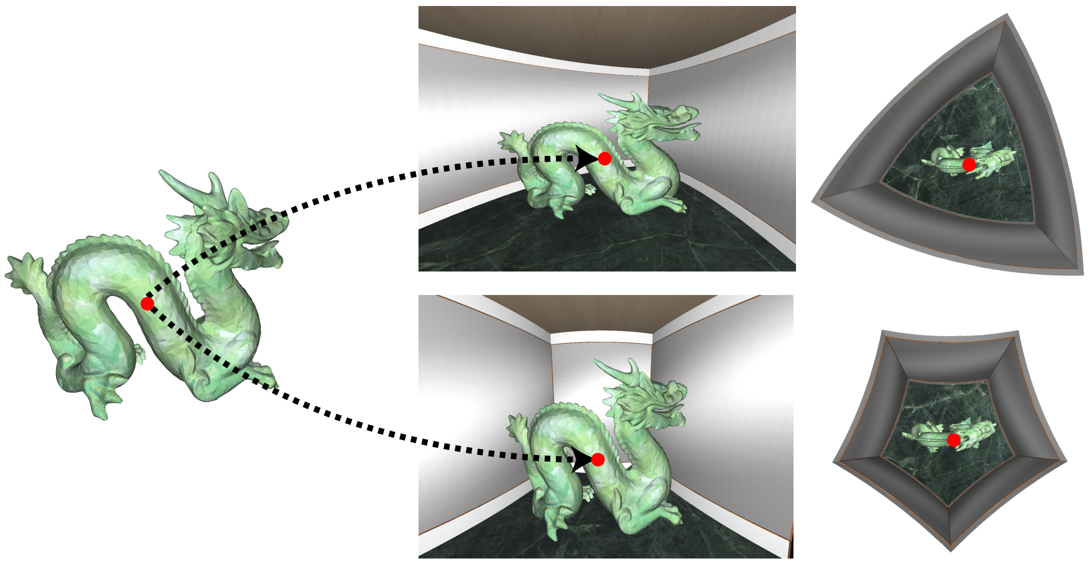


Figure 7.1: This figure shows the embedding of a dragon in the spherical orbifold $*222$ (top) and the hyperbolic orbifold $*22222$ (bottom). The red dot indicates the mass center of the dragon, and the last column displays the top view.

the Poincaré disk can also be modelled by Möbius transformations according to Equation 6.6.

We store the Möbius transformation of each object, and update it when the model is interactively moved inside the room. The Möbius transformation, which corresponds to a translation of the spherical space or the hyperbolic space, is applied to all the vertices in the mesh. Additionally, after updating each vertex of the object, the normal must be recalculated for each triangle of the object.

As shown in Figure 7.2, the dragon undergoes the Möbius transformation along the real axis from $(-1,0)$ to $(1,0)$ in the stereographic plane (top) and the Poincar'e disk (bottom). In the stereographic plane, the size of the dragon is deformed in the sequence: large, middle, small, middle, large. In the Poincaré disk, the size of the dragon is deformed as follows: small, middle, large, middle, small. The deformation of the dragon highlights the features of the underlying spaces.

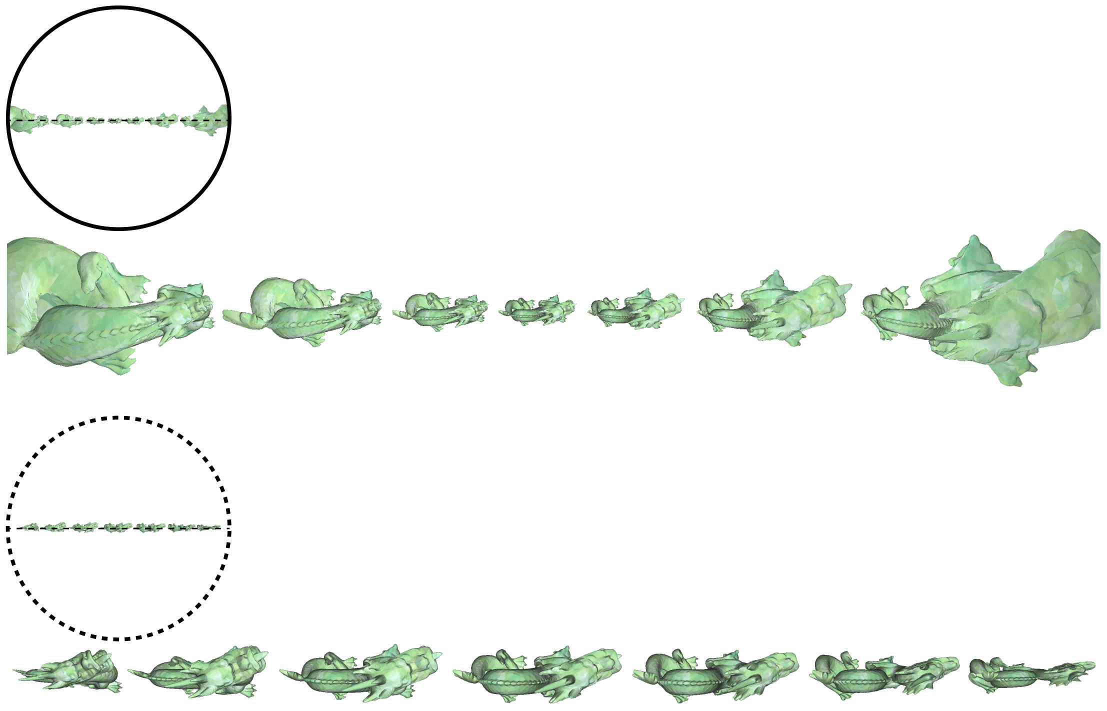


Figure 7.2: The dragon is translated along the real axis in the stereographic plane (top) and the Poincaré disk (bottom). The dragon's mesh is deformed based on its location.

Chapter 8: Rendering Orbifolds

After interactively designing mirror rooms for arbitrary non-Euclidean orbifolds, we can visualize the orbifold using a ray-based rendering algorithm. Due to the stereographic projection in non-Euclidean spaces, the algorithm works for both spherical and hyperbolic spaces without extra modification in the tracing process. In Chapter 8.1, we first introduce the curved ray tracing algorithm to render orbifolds with mirrors as a metaphor. To accelerate the rendering process, we propose handling the tracing with the boundary of the orbifold and the triangles of objects individually in Chapter 8.2. Additionally, we demonstrate the modifications to spatial hierarchies for non-Euclidean spaces in Chapter 8.3.

8.1 Curved Ray Tracing

Once the scene has been constructed, we can either display it interactively during the design stage (Figure 7) or send it to Mitsuba [39] for high-quality offline rendering. Mirror scenes in Euclidean space can be rendered using straight rays in Mitsuba without any modification. However, light rays travel along the geodesics in the spherical and hyperbolic spaces. When rendering a non-Euclidean scene using Euclidean straight lines, incorrect appearances result as shown in Figure 8.1 (a). We modify the rendering algorithms to account for the correct paths for rays (Figure 8.1 (b)). In this chapter, we first define the curved ray used to render non-Euclidean spaces(Chapter 8.1.1). We introduce the formulation for spiral-triangle intersection to render triangular meshes in scenes (Chapter 8.1.2). To achieve photo-realistic rendering of scenes with mirrors, we illustrate the integration of the curved ray into Mitsuba's current rendering framework (Chapter 8.1.3). Finally, we demonstrate our mirror-based visualization across various visualization tasks (Chapter 8.1.4).

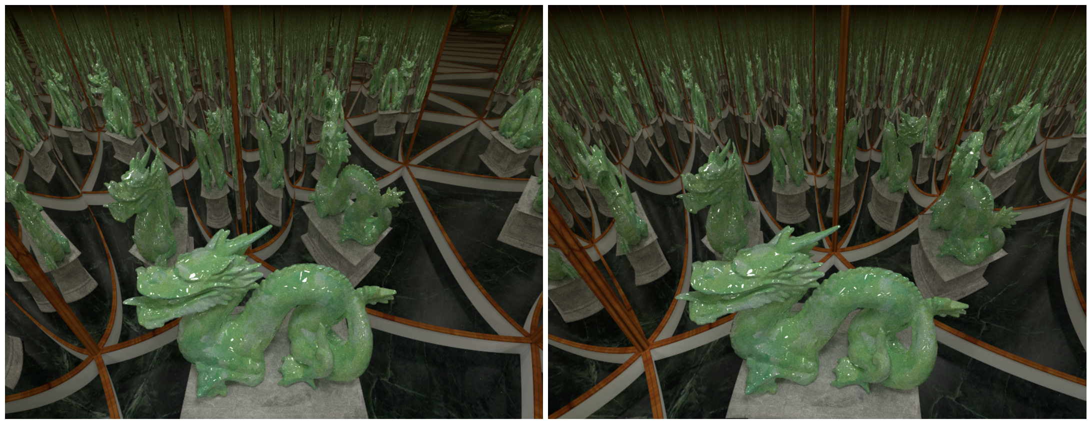


Figure 8.1: In (a-b), we show the comparison of the rendering of a hyperbolic orbifold with straight rays as in the Euclidean space (a) and the geodesics in the hyperbolic space (b).

8.1.1 Definition of Curved Ray

For highest-quality of results, we make use of ray-tracing type of approach with Mitsuba. Recall that our non-Euclidean room is modelled by a subset in $\mathbb{D} \times [0,h]$ where $\mathbb{D}$ is the stereographic plane or Poincaré disk and h is the height of the room. Thus, any geodesic γ in the space satisfies that its horizontal projection is a geodesic in the spherical or hyperbolic space. Consequently, γ takes the form of a 3D spiral. To generate the correct rendering of such scenes, we modify the ray-triangle intersection algorithm in Mitsuba from line-plane intersection to spiral-plane intersection.

A spiral γ in the room can be given a parameterized form based on a point on the spiral p and the tangent direction v at p. We decompose v into its horizontal component v_H and its vertical component v_Y (Figure 8.2).

Therefore, the parametric form of the spiral γ is as follows:

$$p(t) = O_p + r_p \cos\left(|v_w|t\right)\widehat{O_p p} + r_p \sin\left(|v_w|t\right)\widehat{v_H} + tv_Y. \tag{8.1}$$

Here, t is the parameter, O_p is the center of the circle arc and r_p is the radius for the geodesic, and $|v_w| = \frac{|v_H|}{r_p}$. $\widehat{O_p p}$ and $\widehat{v_H}$ are normalized vectors. Figure 8.3 shows

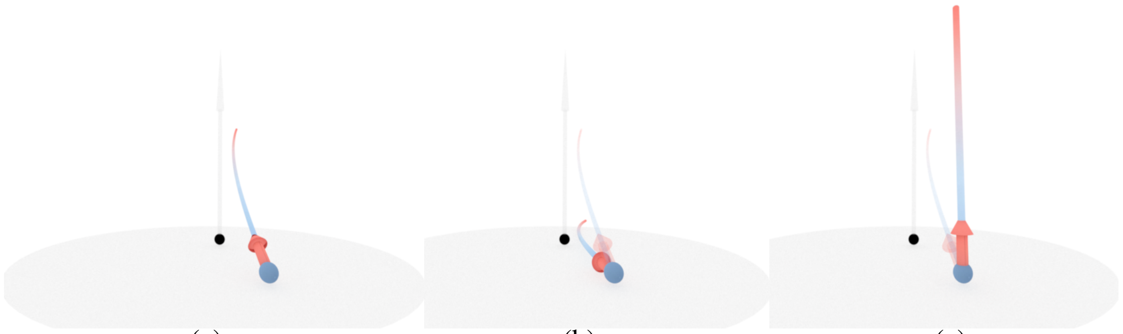

(a) (b) (c)

Figure 8.2: The decomposition of a spiral in hyperbolic space. The gray disk is the Poincaré disk in the horizontal plane, and the gray arrow represents the vertical direction. The spiral (a) is decomposed into the geodesic ray (b) along the direction v_H and the vertical ray (c) along the vertical direction v_Y.

an example of parameterized spiral. We can then use this form to find the intersection of the spiral γ with a given triangle on an object in the scene. When there are multiple intersections with the triangle, we find the one that has the smallest positive t value, which represents the closest intersection from the reference point p. Based on the above

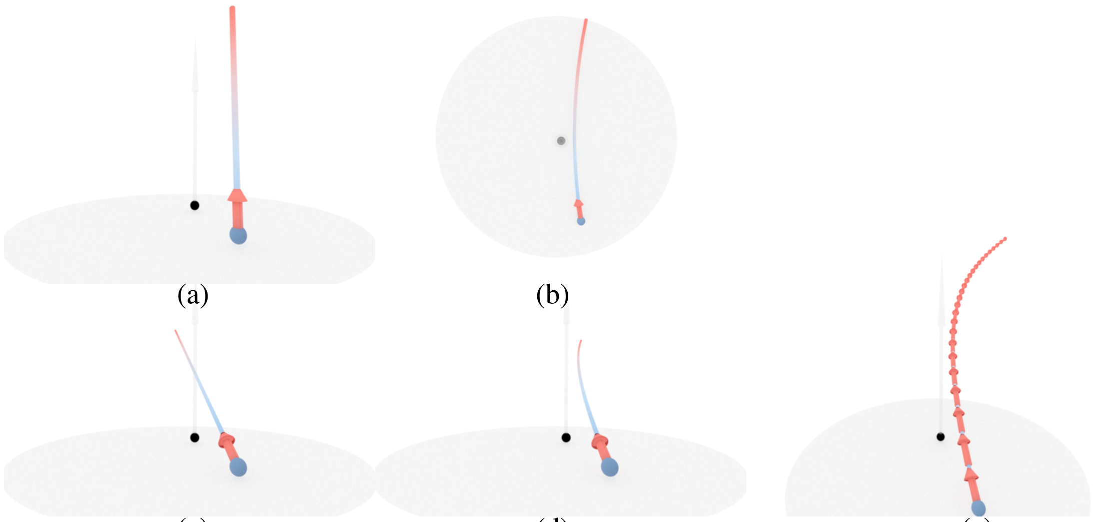

(a) (b) (c) (d) (e)

Figure 8.3: The paths and tangents of spirals in hyperbolic space. (a-d) are the paths of the spirals. (a) $|v_H| := 0$, and $|v_Y| \neq 0$; (b) $|v_H| \neq 0$, and $|v_Y| := 0$; (c) v_H is along pO; (d) $|v_H| \neq 0$, and $|v_Y| \neq 0$; (e) shows the tangents of the points on the spiral (d).

defined curved ray, we can find the precise locations a light rays coming from the viewer, emanating from the light sources, or bouncing off objects including mirrors in the room.

Next, for determining the incident direction of the spiral interacting with a surface, the tangent vector $T(t)$ at parameter t is defined. If the spiral is straight, $T(t) = v$. Otherwise, the tangent is directly obtained using Equation 8.2:

$$T(t) = |v_H| \cos(|v_w| t) \cdot \widehat{v_H} - |v_H| \sin(|v_w| t) \cdot \widehat{O_p p} + v_Y \tag{8.2}$$

The tangents of the points on the spiral are shown in Figure 8.3 (e).

8.1.1.1 Real-time Scan Conversion with Projection

During design stage, we make use of scan conversion for interactive feedback. Recall that in this case, we also render the universal cover since we do not model mirror reflections during design. For each vertex on an object in the scene or a virtual copy, we project it to the image plane by using the same spiral-triangle intersection as in the case of Mitsuba rendering. We then perform barycentric interpolation to find the footprint of any triangle in the image plane by interpolating their vertices' locations. Given enough mesh resolution, the error in this interpolation-based approach is relatively small since all of its vertices are projected correctly using the spiral-triangle intersection. Figure 8.4 demonstrates the result of scan conversion by projecting triangles onto the screen. The left column displays the result without using projection in non-Euclidean spaces, while the right column uses projection by curved rays. In Figure 8.4 (top), showing the spherical orbifold $*235$, the curved wooden frame in the right image appears more straight than it in the left image. Additionally, the view expands from near to far based on the number and size of dragons. In Figure 8.4 (bottom), showing the hyperbolic orbifold $*2352$, the scene on the right side with projection by curved rays demonstrates a different view compared to the left side. The results in the right side indicate exponential shrinkage along curved rays. We deploy a computer featuring an i7-8700K @3.70GHz CPU and an NVIDIA GeForce GTX 2080 GPU to render the scene in Figure 8.4. For the spherical orbifold, with 71 symmetries and approximately 2.62×10^6 triangles, the frame rate is approximately 18 FPS. For the hyperbolic orbifold, with 205 symmetries and around 8.65×10^6 triangles, the frame rate is about 15 FPS.

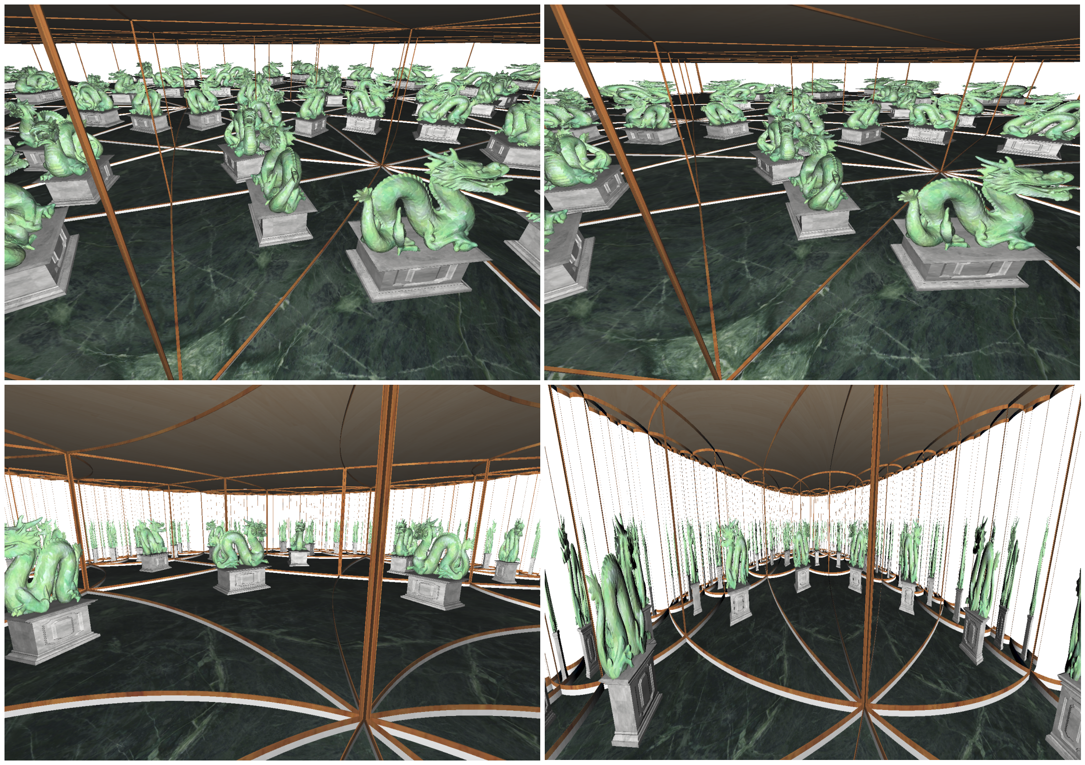


Figure 8.4: This figure shows the comparison between projection in Euclidean space (left) and projection in non-Euclidean spaces (right). The top image represents the spherical orbifold $*235$, and the bottom image represents the hyperbolic orbifold $*2352$.

8.1.2 Curved Ray-Triangle Intersection

Instead of projecting triangles onto the curved ray backward to render the universal cover, we need to find the closest intersection between curved rays and triangles in a forward direction. The basic idea is to first obtain the intersection between the spiral and the plane where the triangle is located. Subsequently, we check whether the intersection is inside the triangle or not. By combining the equation of the spiral with the analytic formulation of a plane, we can numerically solve for the root of the following equation

(Equation 8.3) using Newton's method [52]:

$$f(t) = \sin\left(|v_w|t + \phi\right) - (kt + h) \tag{8.3}$$

where ϕ, k, and h is the parameters from the reformulation (Equation 8.4).

$$\begin{aligned}
k &= \frac{-\left(v_Y \cdot n_t\right)}{r_p\sqrt{a^2+b^2}} \\
h &= \frac{\left(p_0 - O_p\right) \cdot n_t}{r_p\sqrt{a^2+b^2}} \\
\phi &= \arctan\frac{a}{b} \\
\sin\phi &= \frac{a}{\sqrt{a^2+b^2}} \\
\cos\phi &= \frac{b}{\sqrt{a^2+b^2}} \\
a &= \widehat{O_p p} \cdot n_t \\
b &= \widehat{v_H} \cdot n_t
\end{aligned} \tag{8.4}$$

where n_t represents the normal of the triangle, and p_0 denotes a vertex of the triangle. In our setup, the default accuracy for the numerical method is 10^{-6}.

To render mirror scenes with curved rays, we must integrate the curved ray into the current rendering algorithm. We present the details of the modification in Chapter 8.1.3.

8.1.3 Optics-based Visualization

To render Non-Euclidean kaleidoscopic orbifolds, the reflection and refraction of the curved ray are necessary. We assume that light rays travel along the geodesics defined in Equation 8.1. Thus, the incident and outgoing directions when a light ray hits a surface are defined as the tangents of the curved ray at the intersection. The directions of the light ray are represented in Figure 8.5 (the yellow arrows).

To enable rendering the scene based on the physically based rendering framework in Mitsuba, we assume that the bidirectional reflectance distribution function of the mate-

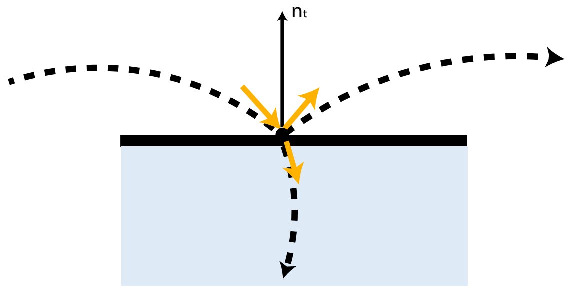


Figure 8.5: A light ray traveling along a curved ray undergoes reflection and refraction on a surface. The dashed line illustrates the curved path of the light ray, and the solid yellow line indicates the directions of reflection and refraction, corresponding to the tangents of the curved rays at the intersection.

rial in the non-Euclidean spaces has the same behavior as in Euclidean space. In order to address the rendering equation [42], we assume that Lambert's law still holds, where the irradiance arriving at the surface varies with the angle of incidence of illumination [65]. As the change in the variable incident direction is continuous, Monte Carlo integration by path tracing can be applied to estimate radiance [42, 70, 71]. However, to reduce variance through multiple importance sampling (MIS), the conversion from a density for area to a density for solid angle no longer holds. This implies that sampling from light sources for MIS is not applicable. The conversion of the area-solid angle density equation is a complex function that depends on the position of the intersection and the incident directions of the area of light sources. To solve the equation, numerically discretizing the area of light sources at each intersection is necessary. We employ a photon mapping algorithm [41, 30, 29, 40] to avoid the conversion and achieve a curved ray tracing, which is a two-stage algorithm involving emitting photons from light sources, recording the photons for each intersection, and then gathering photons on surfaces where the camera ray intersects. Therefore, the conversion of the area-solid angle density equation is numerically estimated. In our implementation, we modify the *SPPM* plugin in Mitsuba to incorporate curved ray tracing, and we present the rendering results in Chapter 8.1.4.

8.1.4 Visualization of Kaleidoscopic Orbifolds by Mirror Scenes

In this section, we show rendering results of visualization for kaleidoscopic orbifold by mirror scenes.

Sensing the Orbifold: An orbifold is a topological space that stands on its own. Using our mirrored scenes, we can produce visualizations that emphasize different aspects of an orbifold. For example, for the scene shown in Figure 8.6, we can adjust the attenuation factors for the mirrors to emphasize the orbifold itself (Figure 8.6 (a)) by setting high attenuation for both mirrors, or the translational cover (Figure 8.6 (b)) by setting high attenuation for one mirror and low attenuation for the other mirror, or the universal cover (Figure 8.6 (c)) by setting low attenuation for both mirrors. Notice that in all of these cases, our mirror-based visualization provides a clear sense of the orbifold, the room.

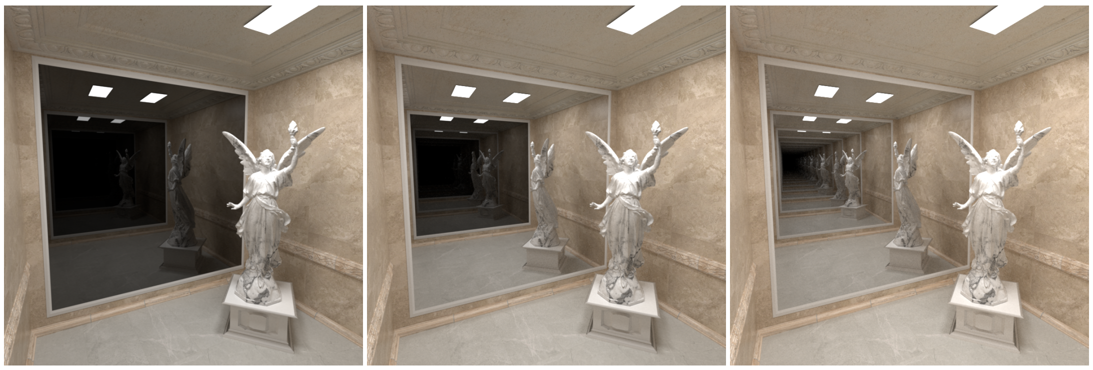
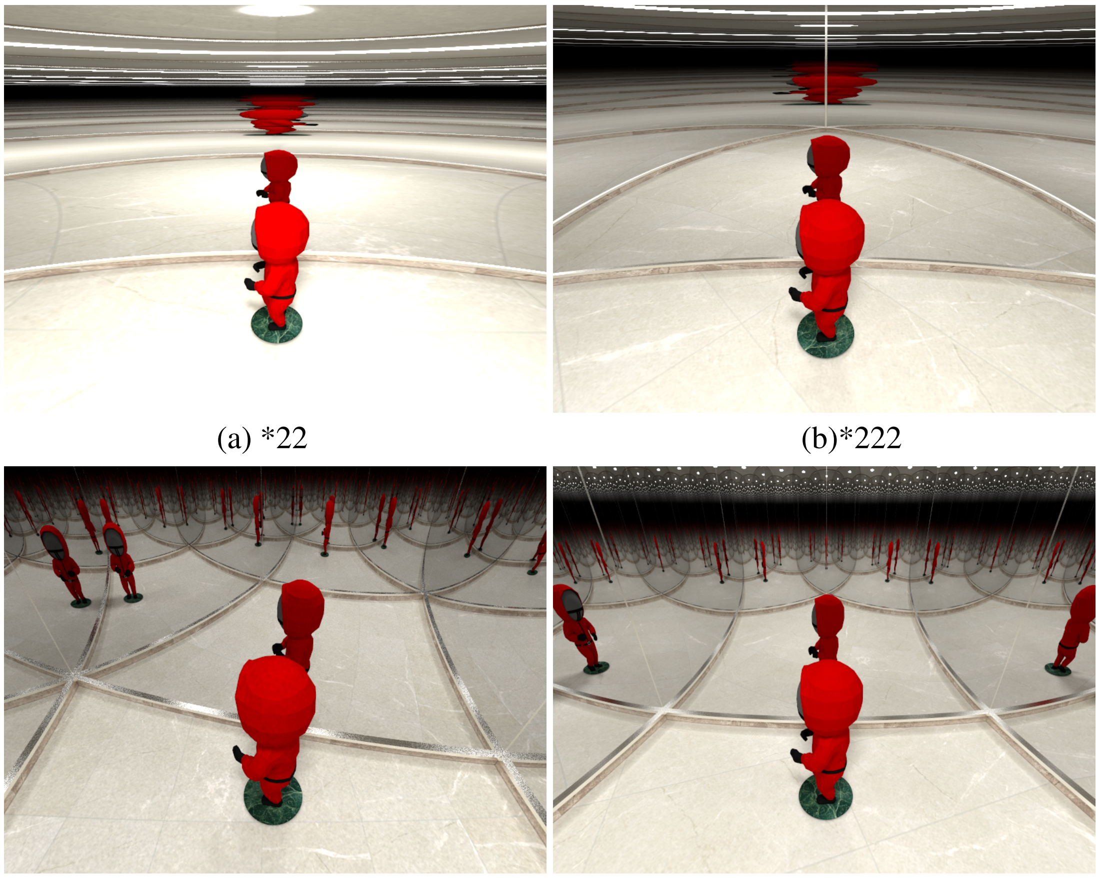

(a) (b) (c)

Figure 8.6: By varying the attenuations of the mirrors, our visualization can emphasize the orbifold (a), its translational cover (b), or its universal cover (c).

Determining the Orbifold Notation: Once an intuitive sense of the orbifold is established and the structure of the reflection is observed, it becomes a relatively straightforward task to determine the type of the orbifold in terms of its orbifold notation. Given the power of 3D graphics and animation, we produce either a panorama of the scene or an animation in which an avatar walks along the walls Figure 8.7. As the avatar approaches the i-th corner, so do its nearest reflections. There appear to be k_i pairs of avatars approaching the corner. Then, k_i pairs of the same avatars leave the corner for the next one. This helps the user identify k_i for the i-th corner. By walking around the room one time, each corner is visited. This can help the user write down the orbifold notation. In kaleidoscopic orbifolds with consistent angles on the corners but different order of angles, the reflections in the universal covers are not the same. As shown in Figure 8.8, the left side represents $*2233$, and the right side represents $*2323$. Each row displays a different view. In the first row, the left scene exhibits two corners with the pattern $*22$, while the right scene features the pattern $*23$. In the second and last row, the left scene shows a single corner $*2$, whereas the right scene displays a single corner $*3$. The configurations inside the orbifolds are identical. Despite both orbifolds having two $*2$s and two $*3$s corners, the differing corner order results in variations in reflections in the universal cover.



(a) *22

(b)*222

(c) *2323

(d) *22222

Figure 8.7: An avatar walks along the walls of the orbifolds. The universal cover is the sphere for (a-b) and the hyperbolic space for (c-d).

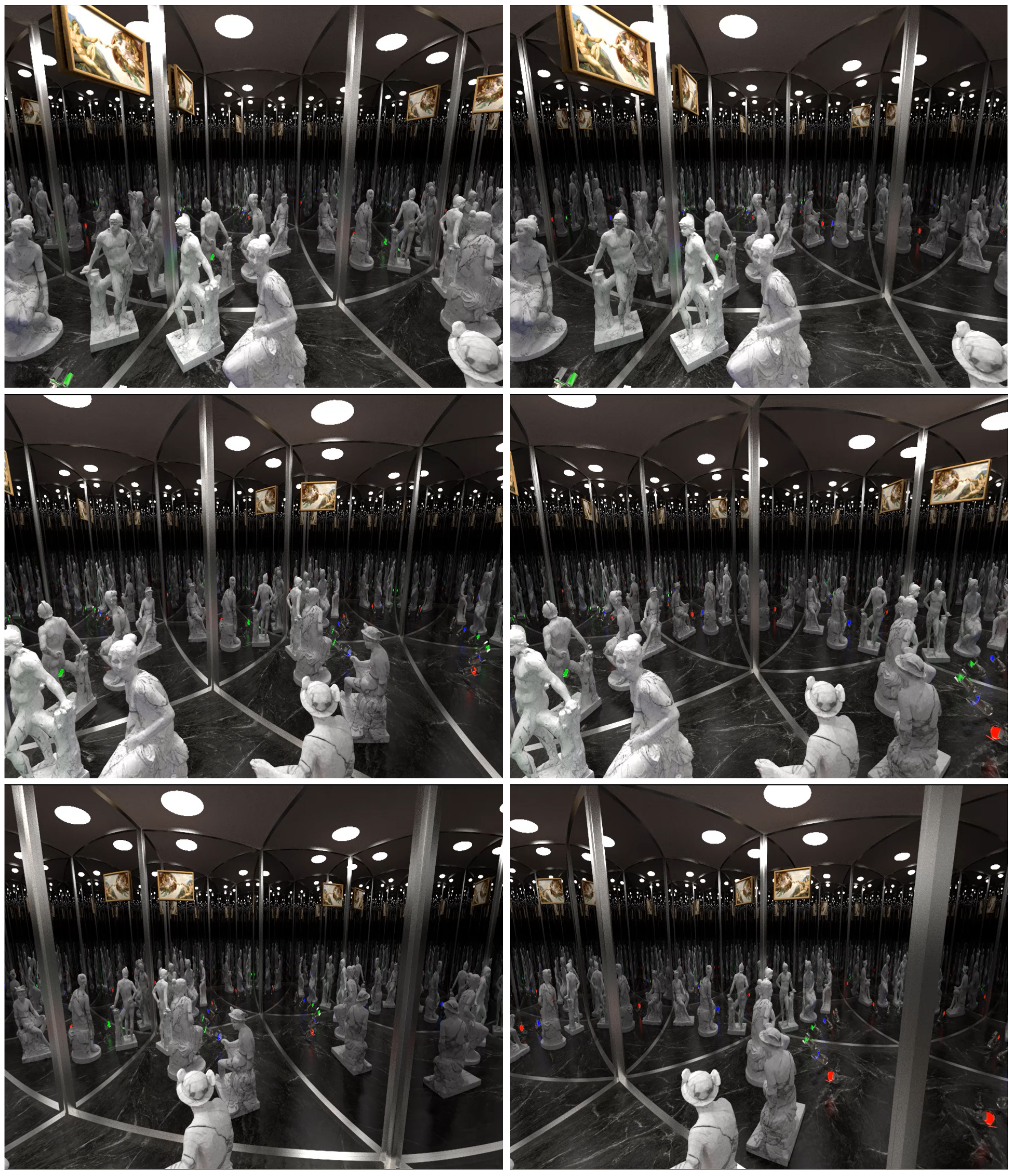


Figure 8.8: Status scenes for the orbifolds: $*2233$ (left) and $*2323$ (right). Each row displays views as the viewer rotates from left to right.

Identifying the Type of the Universal Cover: With the same panorama and walkaround animations, the user can gain insight into the type of the universal cover. Both the spherical space and the hyperbolic space can bring unfamiliar experience to someone new to the concept. For example, inside the spherical space, objects do not always appear smaller when they are further away. Specifically, when the viewer is at the south pole, objects near the north pole appear much wider than the same object at the equator (Figure 8.7 (a-b)). On the other hand, objects farther away always look smaller in the hyperbolic space. However, with the seemingly same distance to the viewers, an object can look much smaller in the hyperbolic space than in the Euclidean space (Figure 8.7 (c-d)). As the objects move around the scene such as the avatar in Figure 8.7, the way the reflections of the objects deform in non-Euclidean spaces is rather different from that in the Euclidean space. For example, a reflection of the avatar may suddenly grow much bigger and then shrink quickly in the spherical space. With a relatively simple scene (the orbifold), the user can gain insight into the entire universal cover through mirror reflections. Moreover, as illustrated in Figure 8.10, the Buddha is placed within a setting with three walls, where each corner has the same angle. The figure displays the universal cover as the corner transitions from $*2$ to $*7$. In Figure 8.11, the Buddha scene is positioned within a setting with a $*2$ corner, while the number of walls increases from 2 to 7. From these two figures, we can observe that higher order of symmetry and more walls at a corner likely lead to hyperbolic orbifolds.

Sensing the Non-orbifold: One of the most fundamental tasks for orbifold visualization is to decide whether a given mirror scene is an orbifold. Non-orbifolds can be difficult to conceptualize, especially when one or more angles at the corners violates $\frac{\pi}{k}$ for $k \in \mathbb{N}^+$. The resulting rendered images and/or videos can deliver a sense of whether the room is a kaleidoscopic orbifold. When the room is a kaleidoscopic orbifold, the viewer should be able to move around the scene and see a consistent larger scene (the universal cover) that is seamlessly tiled by copies of the original room (the orbifold). On the other hand, if the viewer sees conflicting results, such as the double-headed dragon in Figure 8.9, it is an indication of a non-orbifold. In this particular examples, the rooms have a corner with an angle of $\frac{\pi}{k}$ where $k = \frac{2}{3}$, which is not an integer.

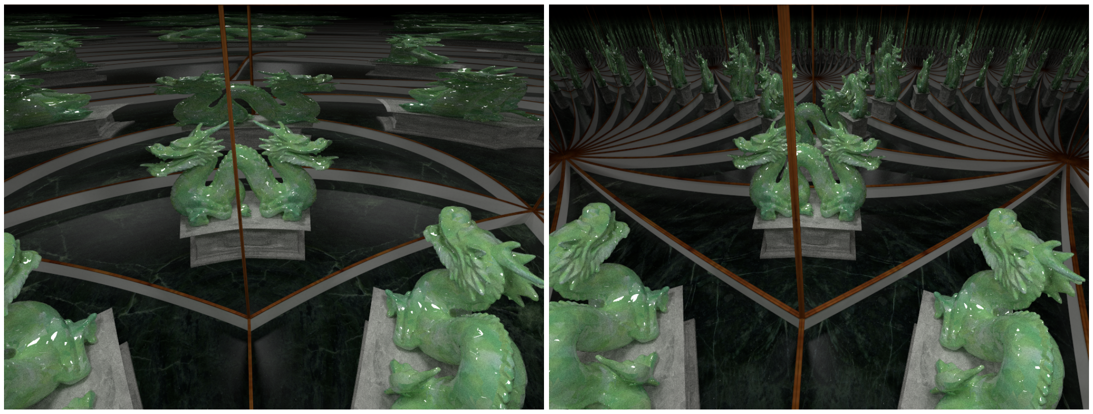
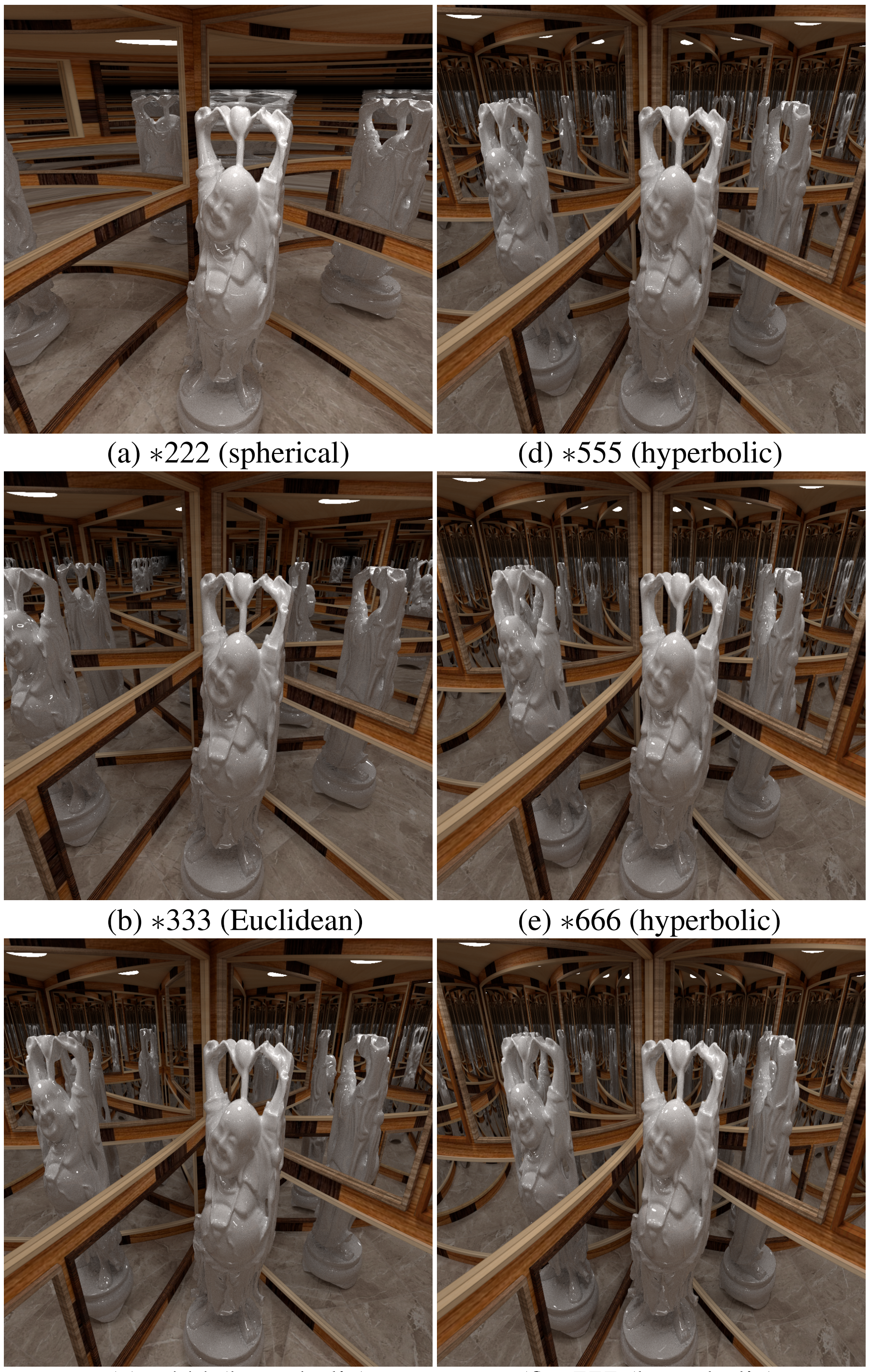


Figure 8.9: We render triangular non-orbifolds with corner angles $\frac{2\pi}{3}$-$\frac{\pi}{3}$-$\frac{\pi}{3}$ (left) and corner angles $\frac{2\pi}{3}$-$\frac{\pi}{12}$-$\frac{\pi}{12}$ (right). Notice the dragon now has two heads.

Sensing the Infinity Corner: In comparison to Euclidean and spherical orbifolds, a special type of orbifold in hyperbolic space has a corner represented as $*\infty$, where the corner point lies at infinity (on the boundary of the Poincaré disk). However, visualizing $*\infty$ is challenging with a 2D texture, as the $*\infty$ point is actually at a finite distance in the disk. Our mirror-based visualization is designed for visualizing the infinity corner. With mirror reflections, the curved ray approaches corner at $*\infty$ will be reflected infinitely, leading to the attenuation of the light energy to 0. Therefore, the $*\infty$ corner will appear dark but still display reflections around the corner. In Figure 8.12 (a-c), we show the gradual addition of mirror walls. Interestingly, when there are zero mirrors (a) or one mirror (b), even though the corner is at infinity, it is difficult to perceive. With two mirrors at the infinity corner, the reflections of Buddha shrink into darkness. In Figure 8.12 (d-f), we depict additional infinity corners for Buddha scenes, revealing that the Buddhas will shrink into darkness with multiple infinity corners. With an increasing number of infinity corners, the Buddhas in the reflections shrink rapidly.

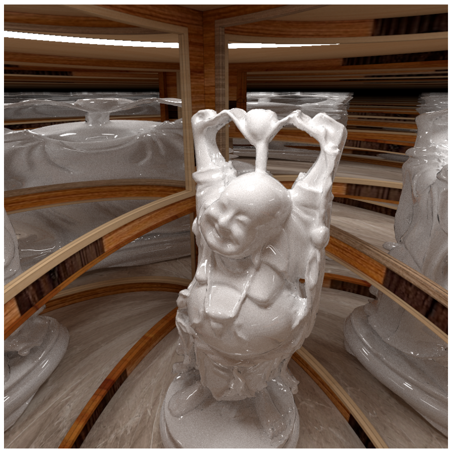


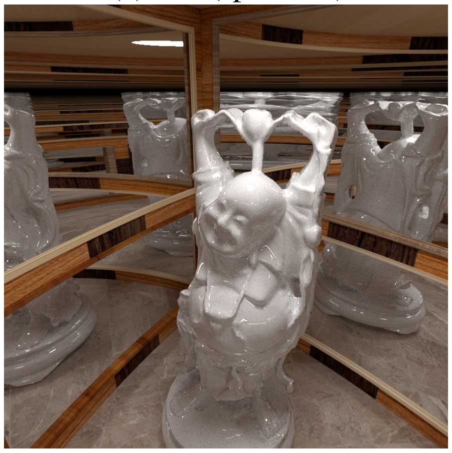
(a) ∗222 (spherical) (d) ∗555 (hyperbolic)

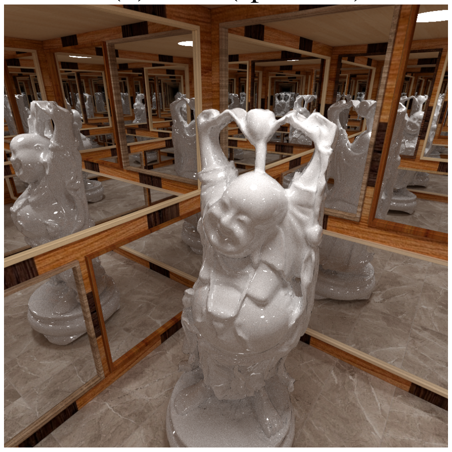
(b) ∗333 (Euclidean) (e) ∗666 (hyperbolic)

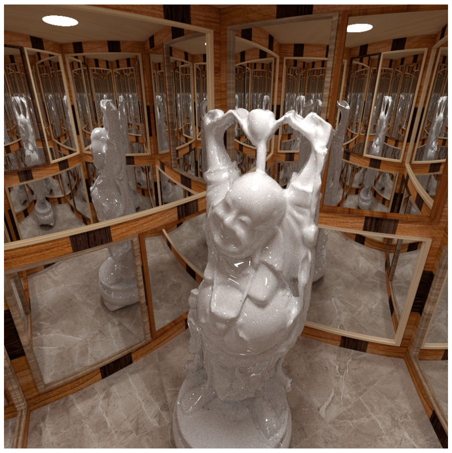
(c) ∗444 (hyperbolic) (f) ∗777 (hyperbolic)

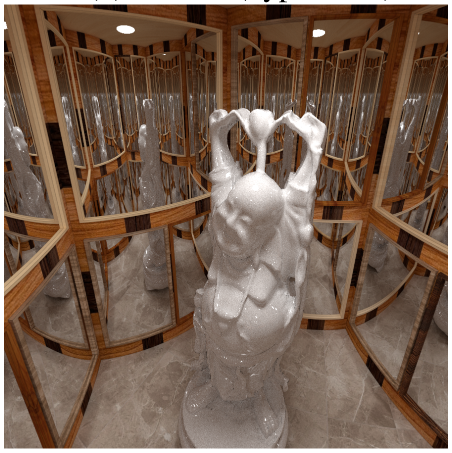
Figure 8.10: The figure depicts the Buddha within an orbifold $*kkk$ where k ranges from 2 to 7 (a-f). For $k \geq 4$, the underlying space of the orbifold is hyperbolic.


(a) $*22$ (spherical)


(b) $*222$ (spherical)


(c) $*2222$ (Euclidean)


(d) $*22222$ (hyperbolic)


(e) $*222222$ (hyperbolic)

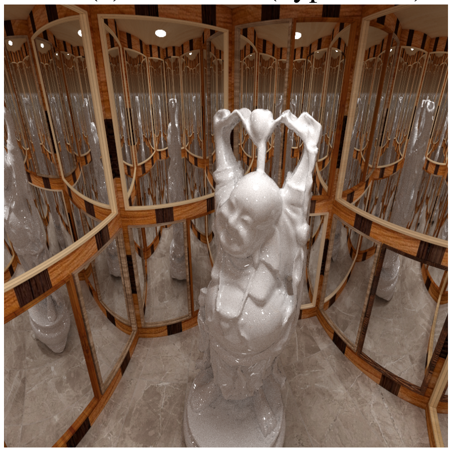
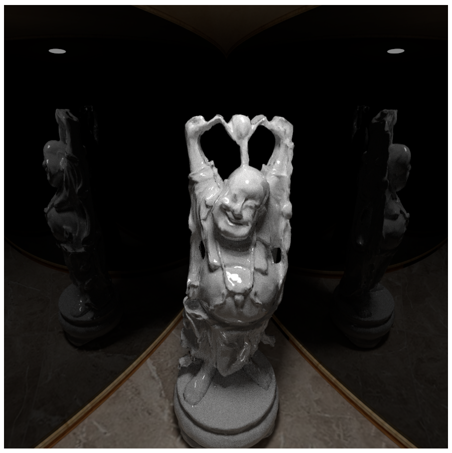
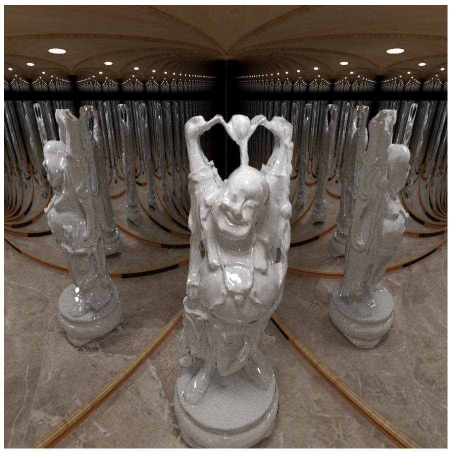
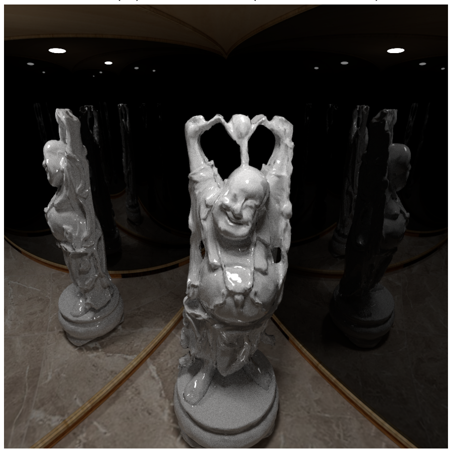
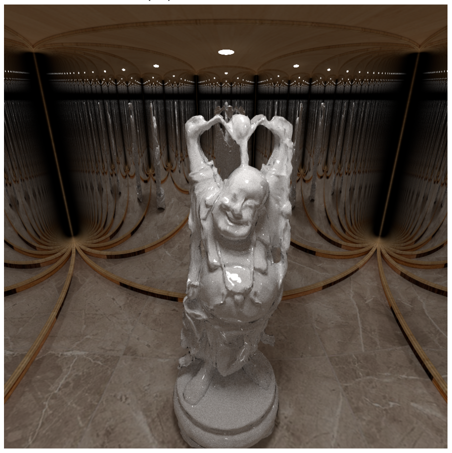

(f) $*2222222$ (hyperbolic)

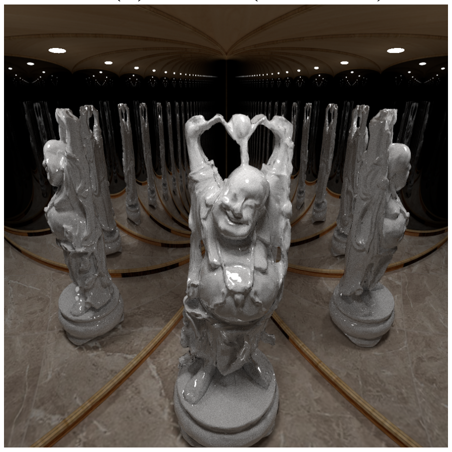
Figure 8.11: The figure shows the Buddha within an orbifold $*2_1 ... 2_N$, where N ranges from 2 to 6 (a-f). For $N \geq 5$, the underlying space of the orbifold is hyperbolic.


(a) $*\infty\infty\infty$ (0 mirrors)


(d) $*\infty\infty\infty$


(b) $*\infty\infty\infty$ (1 mirrors)


(e) $*\infty\infty\infty\infty$


(c) $*\infty\infty\infty$ (2 mirrors)

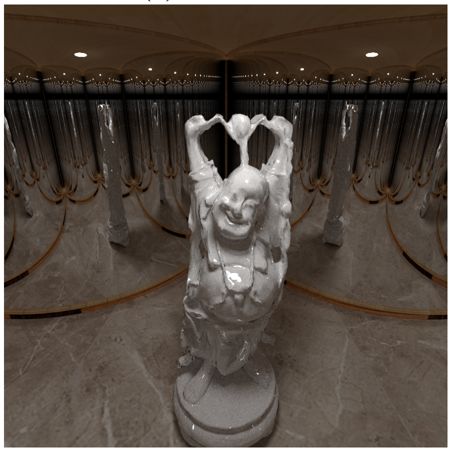

(f) $*\infty\infty\infty\infty\infty$

Figure 8.12: We render the hyperbolic orbifold with a corner at infinity by adding one mirror at a time for $*\infty\infty\infty$ at (a-c). We increase the number of infinity corners from (d-f). The infinity corner appears dark because the light bounces between mirrors an infinite number of times, causing the energy of the light ray to attenuate to 0.

8.2 Analytic Rooms

As described in Chapter 6.2, the mirror walls, floor, and ceiling are triangulated in the mesh. Nevertheless, the mirror walls, responsible for creating reflections of the original room, can be analytically represented. Similarly, the floor and ceiling can be analytically represented. Hence, we can analytically compute the intersection between the room and the curved ray, resulting in substantial computational overhead savings. Based on our observation, there are two types of walls: planes and partial surfaces of cylinders. A plane mirror can be represented by its width and height, while a circular mirror can be represented by its arc and height. The floor and ceiling can be represented as the closure of a set of lines and circles on the plane. The ceiling is defined by a specific height. Our design system can incorporate multiple partial mirrors on a single wall. Each partial mirror has its unique settings for reflectivity and color. As shown in Figure 8.13, we use a tree structure to organize the partial mirrors, facilitating the acceleration of intersection calculations. The approach for curved ray and analytic mirror intersection is the same

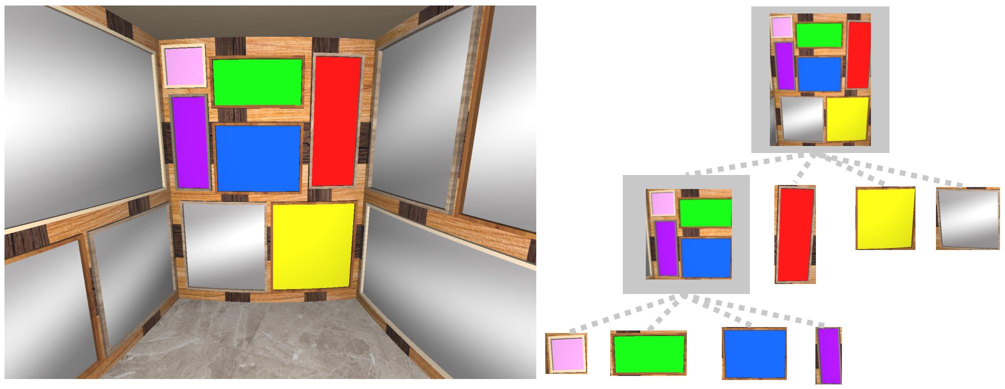


Figure 8.13: This figure shows the hierarchical organization of partial mirrors with various colors on a circular wall (right) within a mirrored room (left).

as the intersection with the bounding of the hierarchical structure Chapter 8.3. We refer to Algorithm 1 to combine curved ray-triangle intersection and curved ray-analytic room intersection.

Algorithm 1 A CURVED RAY INTERSECTS WITH A ROOM AND TRIANGLES

```
Ensure: Check intersection
 1: Initialize boolean variable flg0 as the curved ray intersects with triangles/floor/ceiling at $t_0$
 2: Initialize boolean variable flg1 as the curved ray intersects with the mirrors at $t_1$
 3: if flg0 and flg1 then
 4:     if $t_0 < t_1$ then
 5:         return the intersection at $t_0$
 6:     else
 7:         Generate a new curved ray based on $t_1$
 8:         The curved ray intersects with the room and triangles
 9:     end if
10: else if flg0 and not flg1 then
11:     return the intersection at $t_0$
12: else if not flg0 and flg1 then
13:     Generate a new curved ray based on $t_1$
14:     The curved ray intersects with the room and triangles
15: end if
16: return No intersection
```

8.2.1 Performance of Rendering by Analytic Rooms

In this chapter, we discuss the performance improvement achieved through analytic rooms. We demonstrate the impact of using analytic rooms on both accuracy and rendering time.

Since the meshed mirror is an approximate solution of a circular curve, the analytic mirror is naturally more accurate.

We employ the identical scene depicted in Figure 1.2, using multiple mirrors to assess performance. We use the resolution of the meshed mirror, which produces results approximating analytic mirrors. For off-line rendering, we made modifications to Mitsuba 0.6. The rendering was performed on a Linux Cluster comprised of machines equipped with processors: 2x 20-core 2.50 GHz Intel Xeon Gold 6248 w/ 28160 KB cache, and memory: 768 GB RAM @2933 MT/s. For each scene in Table 8.1, the rendering image had a resolution of 896×896, with the integrator parameters as [39]:

- maxDepth: value=50
- photonCount: value=250000
- rrDepth: value=5
- maxPasses: value=512
- alpha: value=0.7

Additionally, the attenuation of each mirror is set to 0.9. The results of comparison in Table 8.1 demonstrate that analytic rooms (the underscored orbifold in the table) have less memory overhead of meshes (total number of triangles) and reduce up to 23.4% of time overhead over orbifolds. The time saving is only for 512 photon passes; for high-quality rendering, as illustrated in Figure 1.2, utilizing 4096 photon passes, the increase in the number of mirror walls and the refinement of meshed walls will contribute to a greater reduction in overhead of time.

We present a line chart summarizing the time of rendering for each scene in Figure 8.14. The vertical axis represents the time of rendering for the scene in the horizontal axis. The analytic room (green), meshed level 0 (blue), meshed level 1 (orange), and meshed level 2 (red) correspond to the first, second, third, and fourth rows for each scene, respectively. Notably, the analytic room outperforms the meshed rooms, and the time of rendering increases with the number of triangles. Specifically, the $*222$ scene requires more time than others due to the increased number of mirrors and deeper curved ray tracing around the sphere.

In addition to the analytic approach for mirror rooms, we employ hierarchical structures to reduce the computation of intersection between curved rays and triangles.

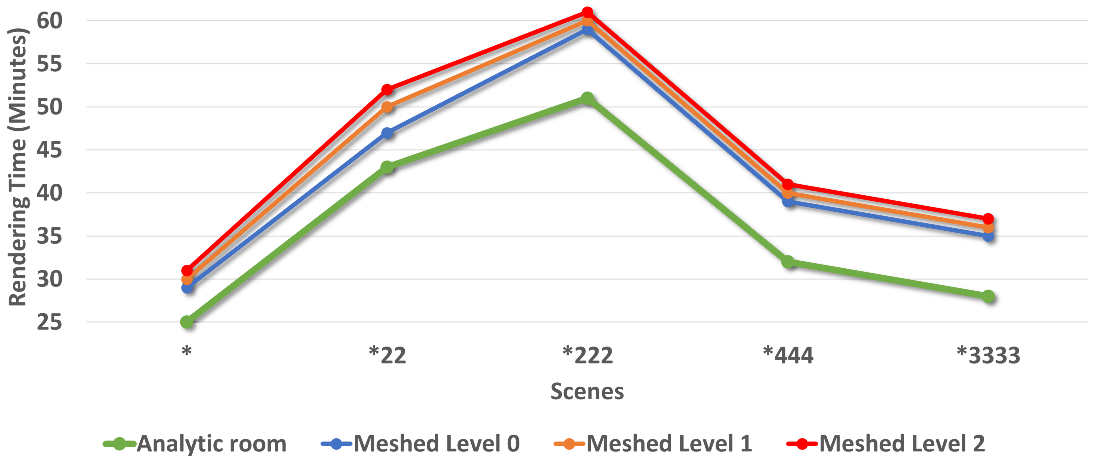


Figure 8.14: The figure illustrates that the analytic room (green line) outperforms the meshed room in terms of rendering time. Specifically, each scene's analytic room corresponds to the first row in Table 8.1, while meshed levels 0, 1, and 2 correspond to the second, third, and fourth rows, respectively. Additionally, it is observed that as the number of triangles increases, the computational overhead also rises.

Orbifold	Type	XML (KB)	# of Tris (Room)	# of Total Tris	Rendering Time
$\underline{\ast}$	S^2	11	4	$\underline{\text{53k}}$	$\underline{\text{25m}}$
$\ast$	S^2	11	1.5k	53k	29m
$\ast$	S^2	11	3.4k	55k	30m
$\ast$	S^2	11	6.2k	59k	31m
$\underline{\ast 22}$	S^2	11	4	$\underline{\text{52k}}$	$\underline{\text{43m}}$
$\ast 22$	S^2	11	0.79k	52k	47m
$\ast 22$	S^2	11	1.7k	54k	50m
$\ast 22$	S^2	11	3.1k	56k	52m
$\underline{\ast 222}$	S^2	15	4	$\underline{\text{53k}}$	$\underline{\text{51m}}$
$\ast 222$	S^2	14	0.9k	54k	59m
$\ast 222$	S^2	14	2.0k	56k	60m
$\ast 222$	S^2	14	3.4k	58k	61m
$\ast 333$	E^2	14	22	51k	3.2m
$\underline{\ast 444}$	H^2	15	4	$\underline{\text{53k}}$	$\underline{\text{32m}}$
$\ast 444$	H^2	14	0.92k	53k	39m
$\ast 444$	H^2	14	1.9k	55k	40m
$\ast 444$	H^2	14	3.3k	58k	41m
$\ast 2222$	E^2	18	28	51k	3.2m
$\underline{\ast 3333}$	H^2	18	4	$\underline{\text{54k}}$	$\underline{\text{28m}}$
$\ast 3333$	H^2	18	2.2k	55k	35m
$\ast 3333$	H^2	18	4.9k	58k	36m
$\ast 3333$	H^2	18	8.6k	63k	37m

Table 8.1: This table presents the scene settings for rendering, with the last column indicating the rendering time in minutes. The underscored scene represents the room with analytic representation. The first row of the table includes the orbifold used for testing, the type of the orbifold, the size of the XML file used in Mitsuba, the number of triangles for the room in the scene (including the floor, ceiling, and mirrors), the total number of triangles (including the room and the objects), and the time of rendering. All values in the table for measurements are rounded to two decimal places.

8.3 Hierarchical Structures

The process of curved ray tracing relies on intersections between curved rays and triangles to render objects in a scene. Therefore, optimizing and accelerating this process is crucial for rendering performance. Two commonly used hierarchical structures are bounding volume hierarchy (BVH) and Kd-Tree [44, 76, 84, 59, 60, 37]. Research suggests that the choice between these structures depends on the scene's complexity [73]. BVH performs better in simpler and moderately complex scenes for ray tracing. However, for more complex scenes, Kd-tree demonstrates higher performance.

We first use the Kd-Tree as the hierarchical structure, akin to Mitsuba's approach, where the Kd-Tree is constructed in Euclidean space for acceleration of ray-triangle intersection checking. The Kd-Tree traversal is modified from straight rays to curved rays. As the following example in Figure 8.15, the intersection of a curved ray with a node of the Kd-Tree formed by an axis-aligned bounding box (AABB) is demonstrated: we first decompose the box into horizontal and vertical surfaces, as shown in Figure 8.15 (a-c). For horizontal surfaces, the surface is projected in the vertical direction, and the intersection parameter t_0 as shown in Figure 8.15 (d) is calculated. Subsequently, we verify whether the horizontal position of the curved ray at t_0 falls within the projected rectangle of the box on the horizontal plane, as illustrated in Figure 8.15 (f). To check the intersection of vertical surfaces, as shown in Figure 8.15 (c). The intersection at t_0 in the horizontal plane of the projected rectangle of the box in Figure 8.15 (e) is determined. Subsequently, the intersection is obtained by examining whether, at t_0, the curved ray resides within the projected range of the box in the vertical direction, as depicted in Figure 8.15 (g). Furthermore, we order the surfaces for intersection checking based on the position of curved ray to prevent redundant checks.

Constructing the Kd-Tree in Euclidean space fails to capture the geometry of non-Euclidean spaces. We propose a construction approach based on the geodesics associated with the underlying space, as discussed in Chapter 8.3.1. However, in the case of both AABB Kd-Tree and geodesics-based Kd-Tree, if the object is translated, the construction of related nodes needs to be recalculated, as the closure no longer holds (Figure 8.16). Leveraging the advantageous property of Möbius transformations pre-

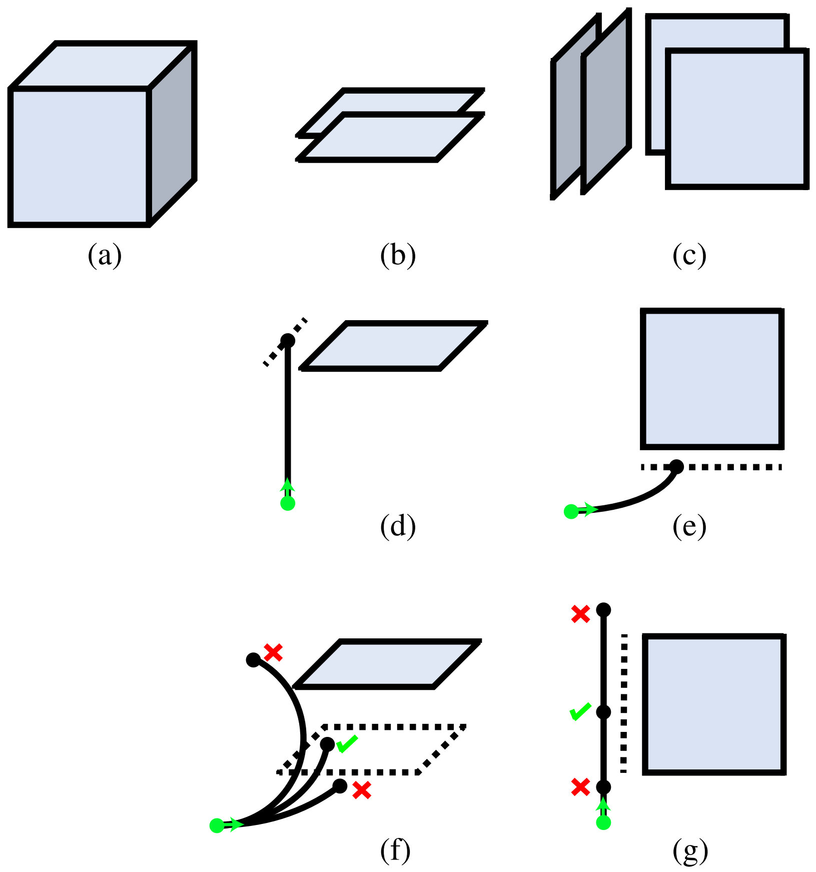


Figure 8.15: (a-c) illustrate the decomposition of an AABB into horizontal and vertical surfaces. (b), (d), and (f) depict the procedure for checking intersections between curved rays and horizontal surfaces. (c), (e), and (g) illustrate the procedure for checking intersections between curved rays and vertical surfaces.

serving circles [34], we suggest employing cylinders for bounding volume hierarchy, requiring only the update of the cylinder's center position and radius. We demonstrate the construction process in Chapter 8.3.2.

In addition, the comparison of the construction time and the speed of intersection checking among three approaches is conducted in Chapter 8.3.4.

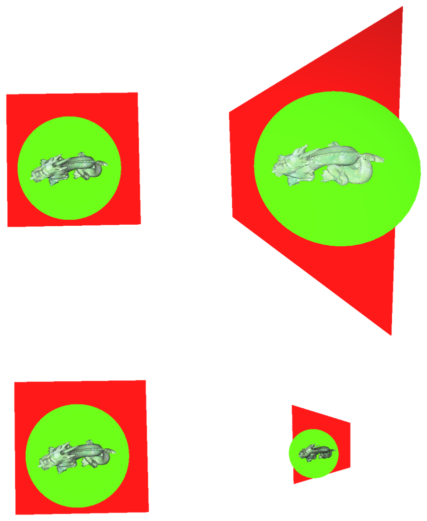


Figure 8.16: These figures illustrate that the hierarchical structure does not hold with translations for AABB. The image (top) illustrates the bounding hierarchy in spherical space, while the image (bottom) represents the bounding hierarchy in hyperbolic space. After translating the object from left to right, the green circles are outside the red boxes.

8.3.1 Geodesics-based Space Partition

We specifically describe the modified portion of the Kd-Tree in non-Euclidean spaces. Chapter 8.3.1.1 shows details on the construction of the Kd-Tree. Refer to Chapter 8.3.3 for details on applying it to calculate intersections while traversing tree nodes.

8.3.1.1 Space Partition Based on Geodesics

In this chapter, we present an algorithm for partitioning triangles in non-Euclidean spaces using geodesics. The geodesics used for partitioning in the vertical direction are the same as those in Euclidean space. For partitioning in the plane, we use geodesics that are perpendicular to the real axis and the imaginary axis in the complex plane (Figure 8.17). To determine the geodesic bounding the triangles, we calculate the geodesic either through the projection of the vertices of triangles on the axis or tangent to the edge (the intersection must be located on the edge), as illustrated in (Figure 8.17 (c-d)). By finding the maximum bounding of all edges for a triangle (Figure 8.17 (e-f)), a geodesics-based partition is created.

When an object in the scene is translated, the geodesics-based space partition requires recalculation of the object's bounding, which can be time-consuming. To address this, we propose a fast algorithm based on cylinder-based bounding volumes.

8.3.2 Cylinder-based Bounding Volume Hierarchy

The bounding volume takes advantage of the property that Möbius transformations preserving circles [34]. Consequently, we only need to update the radius and center of the cylinder-based bounding volumes related to the object. As shown in Figure 8.18. The bounding conditions for objects hold when the object is translated both horizontally and vertically. In addition, the geodesics of curved rays can either form a circle or a line. Thus, we can analytically determine the intersection of the curved ray with the bounding volume. The construction of the bounding volume is similar to the approach presented in [58]. We split triangles based on surface area heuristic [49] and calculate the maximum radius bounding triangles on the plane, as well as the bounding planes in the vertical direction.

8.3.3 Curved Ray Intersects with Nodes of Trees

Both the nodes on the geodesics-based Kd-tree and the cylinder-based BVH are bounded by circles or lines on the horizontal plane. In the former case, it will be on a line and three circles, or two lines and two circles, or four circles. In the latter case, it will be one circle. Therefore, to find the intersection between a curved ray and a node, we perform calculations similar to those in Figure 8.15. The only difference is that we have to perform circle-circle intersection checking.

8.3.4 Performance Comparison among Hierarchical Structures

Comparing the performance of hierarchical structures across different models as depicted in Table 8.2. We evenly distribute locations around the object, with each location having rays evenly distributed along the radial direction. In total, we sample 146 locations, each with 614 rays, resulting in a total of $89,644$ rays. Table 8.2 presents recorded data, including the depth of the tree, the time of construction in milliseconds, and the time of travel to find the intersection in microseconds per ray for each hierarchical structure. Please note that we have rounded the time to the nearest integer. All experiments were conducted on a computer with an i7-8700K @3.70GHz CPU, and the code was executed using a single thread. For all structures, the minimum number of primitives required to halt partitioning is set at 6, and all partitions are based on surface area heuristic [49].

In the comparison, the order of depth for structures is $AABB < Geodesics < Cylinder$, and the associated time of constructions follow the same order: $AABB < Geodesics < Cylinder$. More specifically, the time of construction for the geodesics-based Kd-Tree and the cylinder-based BVH is approximately twice that of the AABB-based Kd-Tree, due to the additional bounding computation. Regarding the time of travel, the order is $AABB < Cylinder < Geodesics$ because the traditional Kd-Tree, with plane bounding, requires at most four line segment-circle intersection checks, which is more efficient than the geodesics-based Kd-Tree with up to four circle-circle intersection checks. The cylinder-based BVH requires one circle-circle intersection check. Additionally, one line segment-circle intersection check is computationally less expensive than one circle-circle intersection check.

Objects	Type	Triangles	Intersections	AABB	Geodesics	Cylinder
Teapot	S^2	4032	7694	14/19/8	15/36/14	16/45/13
Dragon	S^2	34648	3892	18/185/13	20/378/26	22/432/22
Bunny	S^2	69666	8239	18/367/12	20/781/22	24/887/21
Lucy	S^2	100000	3620	19/565/14	21/1157/28	25/1355/24
Buddha	S^2	100000	6081	21/603/16	24/1227/31	26/1418/28
Average	S^2	61669	5905	18/347/12	20/715/24	22/827/21
Teapot	H^2	4032	6224	14/22/7	14/38/14	16/55/12
Dragon	H^2	34648	2959	18/188/12	20/416/26	22/441/21
Bunny	H^2	69666	6560	18/366/12	20/834/23	24/892/20
Lucy	H^2	100000	2622	19/568/13	21/1273/28	25/1390/23
Buddha	H^2	100000	4644	21/604/16	25/1349/33	26/1415/28
Average	H^2	61669	4601	18/349/12	20/782/24	22/838/20

Table 8.2: Comparing the performance of hierarchical structures over time. The table's first column displays various objects used for testing, while the second column indicates the type of curved ray in the corresponding space. The third column represents the number of intersections. The remaining columns (AABB-based Kd-Tree, Geodesics-based Kd-Tree, and Cylinder-based BVH) present performance metrics, including the depth of the tree, construction time in milliseconds, and travel time to find the intersection in microseconds per ray. The final row displays the average results for all objects.

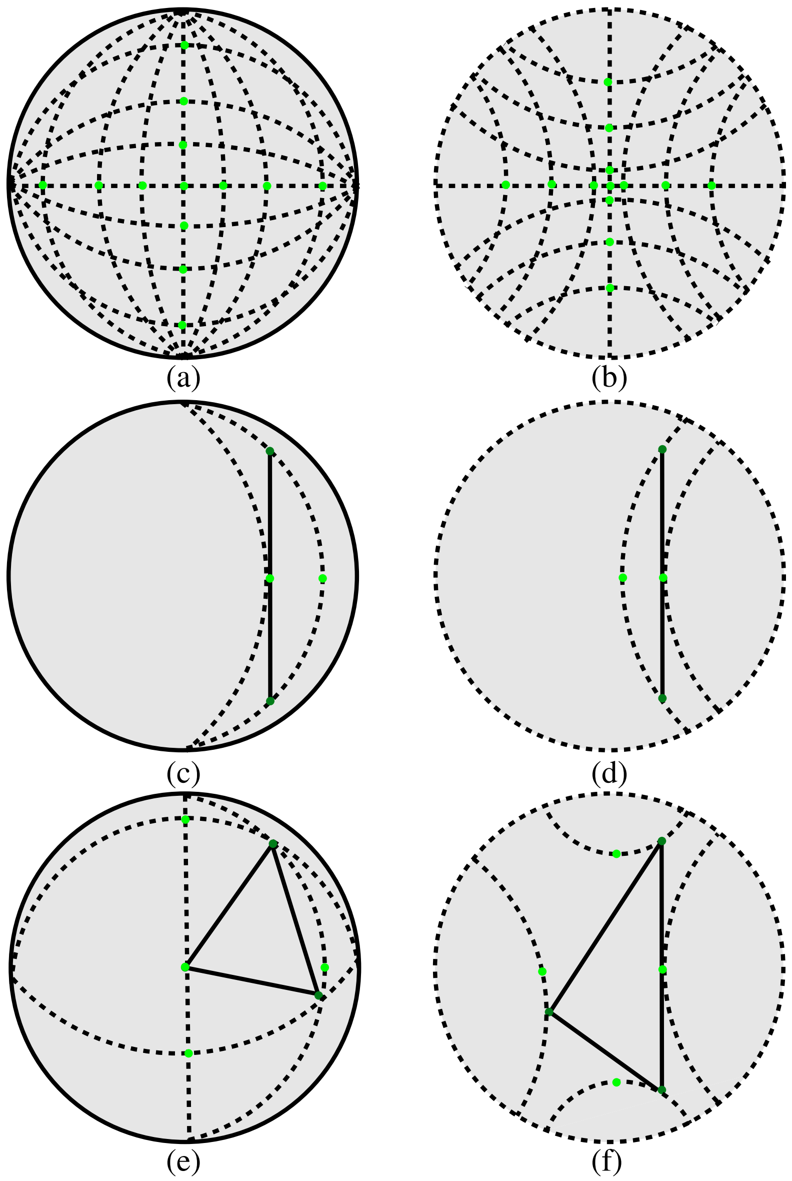


Figure 8.17: Geodesics for space partition in spherical (a) and hyperbolic space (b), the examples of bounding an edge (c-d), and the examples of bounding a triangle (e-f). The dark green dots represent vertices of edges, and the green dot represents the projected points on the axes. The dashed lines represent the geodesics for space partition.

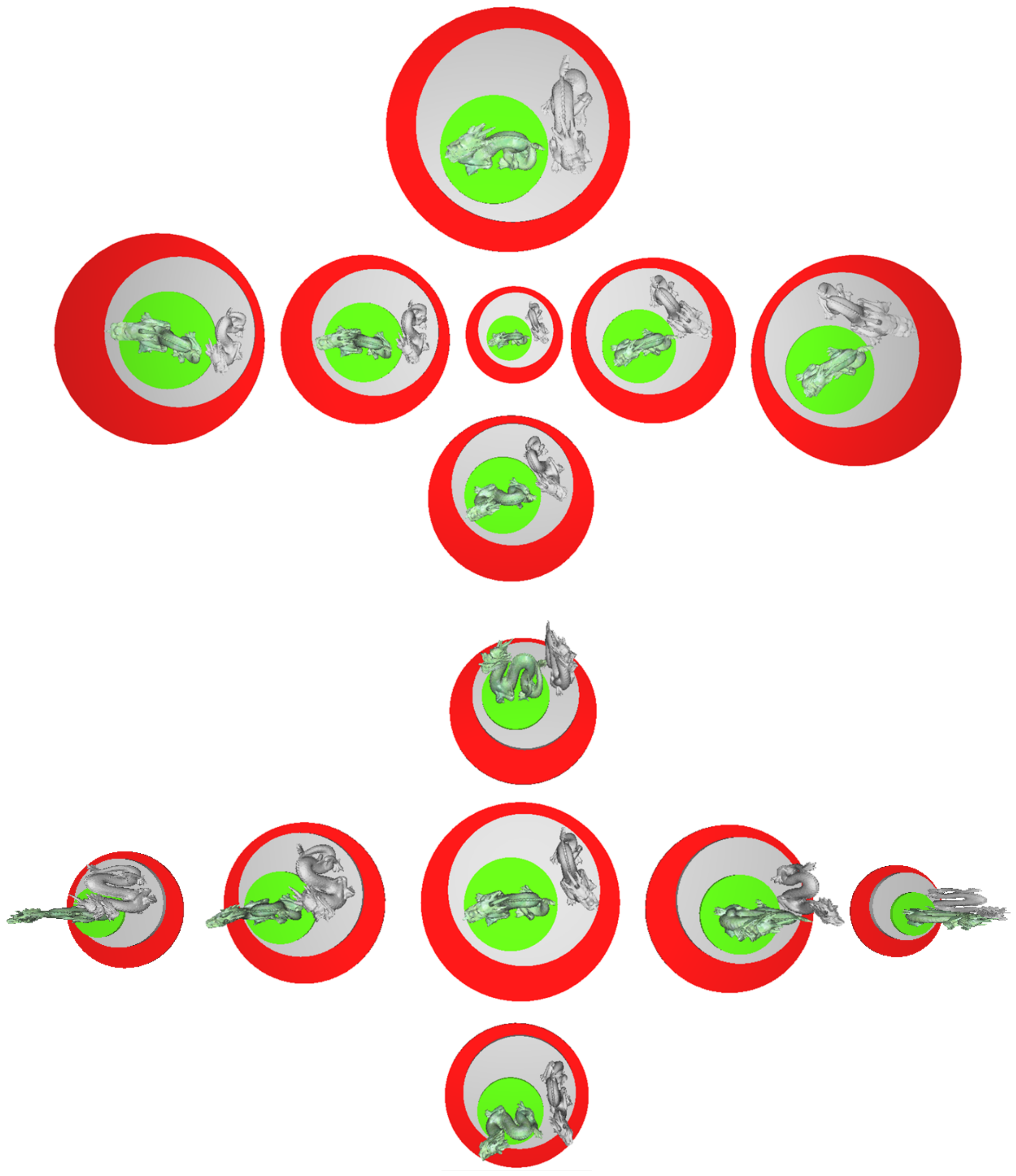


Figure 8.18: These figures demonstrate that the hierarchical structure still holds with left-to-right and top-to-bottom object translations. Thus, updating the center and radius of the bounding cylinders is the only required task, significantly easier than reconstructing the hierarchical tree. The image (top) illustrates the bounding hierarchy in spherical space, and the image (bottom) represents the bounding hierarchy in hyperbolic space. The red bounding volume encloses the white bounding volume, which in turn encompasses the green bounding volume. Within the white bounding volume resides a white dragon, and within the green bounding volume resides a green dragon.

Chapter 9: Conclusion and Future Work

In this chapter, we provide a summary of our contributions to this dissertation. In addition, we discuss the limitations of our work and suggest potential directions for future research.

9.1 Contributions

In this dissertation, we have made the following contributions:

- We provide a system in which users can interactively design any two-dimensional kaleidoscopic orbifolds. Our interactive design system is centered around determining the configuration of a room based on the orbifold notation provided by users. The system provides not only Euclidean orbifolds but also spherical and hyperbolic orbifolds.

- Additionally, we present an enumeration of two-dimensional kaleidoscopic orbifolds, considering the combination of the cardinality of the underlying polygon and the type of universal cover. Our design system facilitates interactive construction of the orbifold's universal cover and allows embedding and movement of objects within the scene.

- We propose the construction of all kaleidoscopic orbifolds in the plane under stereographic projection and provide an algorithm for constructing the underlying polygon of arbitrary kaleidoscopic orbifolds.

- We propose a geodesic-based algorithm to tessellate orbifolds in the underlying space and a contour-based algorithm for fast tessellation with contour preservation.

- We propose using mirrors reflections as a visual metaphor for orbifolds. The user can generate high-quality, photo-realistic renderings of the scene, panorama, and animations with Mitsuba, which we have modified to account for geodesics in spherical and hyperbolic geometry, along which light travels.
- We propose an algorithm to separate the rendering pass between the mirror room and the objects inside the orbifolds, aiming to accelerate the rendering.
- Finally, we propose a geodesic-based space partition algorithm for non-Euclidean spaces and a cylinder-based bounding volume to minimize the reconstruction of hierarchy when translating objects in underlying space.

9.2 Future Work

The dissertation suggests several potential directions for exploration:

- Making rendering more efficient with the off-line rendering by Mitsuba is important, and we plan to investigate more efficient spatial hierarchical data structures for the non-Euclidean spaces. In addition, the quality of the meshes used to represent the floor and the ceiling can impact the rendering speed, and we plan to explore more optimal meshing structures for our purpose.
- Our results show high variance noises when rendering non-Euclidean spaces, particularly in the case of spherical orbifolds. It is challenging for users to select an optimal value for the number of photon passes and the number of photons. We want to propose an algorithm that automatically estimates the number of photons based on irradiance (the radiant flux received by a surface per unit area).
- The distance and size information resulting from the projection and distortion of the non-Euclidean spaces is lost. We wish to investigate how to integrate the perception of distances for orbifolds.
- We wish to expand our rendering system to arbitrary three-dimensional orbifolds. In addition, not all orbifolds are kaleidoscopic, and we would like to incorporate

the visualization for non-kaleidoscopic orbifolds, such as the ones involving gliding reflections. Finally, we wish to explore the visualization of non-orientable orbifolds, whose universal cover is a non-orientable surface such as the Projective plane and the Klein bottle.

- Another way that humans perceive reflection, distance, location, and size is through hearing. We want to investigate how to transfer the geometric information of orbifolds into an aural signal and explore the difference in signals among different orbifolds.

- Dynamic objects can cause ongoing changes in the appearance of objects and can aid in understanding space and orbifolds by providing coherence and discrimination between Euclidean space and non-Euclidean spaces. We wish to extend real-time rigid body and fluid simulation to non-Euclidean spaces and orbifolds. To accomplish this, the difficulty of converting dynamic simulations from Euclidean space to non-Euclidean spaces and reducing the computational overhead of simulation needs to be addressed.

Bibliography

[1] Hyperbolica. https://codeparade.itch.io/hyperbolica. update: 2022-03-14.

[2] Hyperrogue. https://roguetemple.com/z/hyper/. update: 2021-06-03.

[3] Irrlicht engine. http://irrlicht.sourceforge.net/. update: 2016-03-18.

[4] Kaleidotile. https://www.geometrygames.org/KaleidoTile/index.html. update: 2022-05-17.

[5] Noam Aigerman, Shahar Z Kovalsky, and Yaron Lipman. Spherical orbifold tu e embeddings. *ACM Trans. Graph.*, 36(4):90, 2017.

[6] Noam Aigerman, Shahar Z. Kovalsky, and Yaron Lipman. Spherical orbifold tutte embeddings. *ACM Trans. Graph.*, 36(4), jul 2017.

[7] Noam Aigerman and Yaron Lipman. Orbifold tutte embeddings. *ACM Trans. Graph.*, 34(6):190:1–190:12, October 2015.

[8] Noam Aigerman and Yaron Lipman. Orbifold tutte embeddings. *ACM Trans. Graph.*, 34(6):190–1, 2015.

[9] Noam Aigerman and Yaron Lipman. Hyperbolic orbifold tutte embeddings. *ACM Trans. Graph.*, 35(6):217–1, 2016.

[10] James W Anderson. *Hyperbolic geometry*. Springer Science & Business Media, 2006.

[11] Ara Basmajian and Mahmoud Zeinalian. Möbius transformations of the circle form a maximal convergence group, 2006.

[12] P. Berger, Laier A., and L. Velho. An image-space algorithm for immersive views in 3-manifolds and orbifolds. *Vis Comput.*, 31:93–104, 2015.

[13] Marc Ten Bosch. N-dimensional rigid body dynamics. *ACM Transactions on Graphics (TOG)*, 39(4):55–1, 2020.

[14] Vladimir Bulatov. Bending hyperbolic kaleidoscopes, 2011.

[15] Bernard Chazelle. Triangulating a simple polygon in linear time. *Discrete & Computational Geometry*, 6(3):485–524, 1991.

[16] J.H. Conway, H. Burgiel, and C. Goodman-Strauss. *The Symmetries of Things*. Ak Peters Series. Taylor & Francis, 2008.

[17] John H Conway, Peter G Doyle, Jane Gilman, and William P Thurston. Geometry and the imagination in minneapolis. *arXiv preprint arXiv:1804.03055*, 2018.

[18] Daryl Cooper, Craig D. Hodgson, and Steven P. Kerckhoff. *Three-dimensional Orbifolds and Cone-Manifolds*, volume Volume 5 of *MSJ Memoirs*. The Mathematical Society of Japan, Tokyo, Japan, 2000.

[19] Rémi Coulon, Elisabetta A Matsumoto, Henry Segerman, and Steve J Trettel. Ray-marching thurston geometries. *arXiv preprint arXiv:2010.15801*, 2020.

[20] Harold Scott Macdonald Coxeter. *Non-euclidean geometry*. Cambridge University Press, 1998.

[21] Douglas Dunham. Hyperbolic symmetry. In *Symmetry*, pages 139–153. Elsevier, 1986.

[22] Douglas Dunham. An algorithm to generate repeating hyperbolic patterns. *the Proceedings of ISAMA*, pages 111–118, 2007.

[23] Douglas Dunham, John Lindgren, and David Witte. Creating repeating hyperbolic patterns. In *Proceedings of the 8th annual conference on Computer graphics and interactive techniques*, pages 215–223, 1981.

[24] M.C. Escher. *The graphic work of M.C. Escher*. Hawthorn/Ballantine book. Ballantine Books, 1971.

[25] Michael R Garey, David S Johnson, Franco P Preparata, and Robert E Tarjan. Triangulating a simple polygon. *Information Processing Letters*, 7(4):175–179, 1978.

[26] Gary Greaves. Cyclotomic matrices over the eisenstein and gaussian integers. *Journal of Algebra*, 372:560–583, 2012.

[27] Charles Gunn. Advances in metric-neutral visualization. *GraVisMa*, pages 17–26, 2010.

[28] Charlie Gunn. Discrete groups and visualization of three-dimensional manifolds. In *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, pages 255–262, 1993.

[29] Toshiya Hachisuka and Henrik Wann Jensen. Stochastic progressive photon mapping. In *ACM SIGGRAPH Asia 2009 papers*, pages 1–8. 2009.

[30] Toshiya Hachisuka, Shinji Ogaki, and Henrik Wann Jensen. Progressive photon mapping. In *ACM SIGGRAPH Asia 2008 papers*, pages 1–8. 2008.

[31] Andrew J. Hanson. Visualizing quaternions. In *ACM SIGGRAPH 2005 Courses*, SIGGRAPH '05, New York, NY, USA, 2005. ACM.

[32] V Hart, A Hawksley, EA Matsumoto, and H Segerman. Non-euclidean virtual reality i: explorations of h3, 2017, 2017.

[33] Vi Hart, Andrea Hawksley, Elisabetta A Matsumoto, and Henry Segerman. Non-euclidean virtual reality ii: explorations of h2e. In *Bridges 2017 Conference Proceedings*, 2017.

[34] Michael P Hitchman. *Geometry with an introduction to cosmic topology*. Jones & Bartlett Learning, 2009.

[35] Petr Hořava and Edward Witten. Heterotic and type i string dynamics from eleven dimensions. *Nuclear Physics B*, 460(3):506–524, 1996.

[36] John F. Humphreys. *A Course in Group Theory*, volume Volume 5 of *Oxford Science Publications*. Oxford University Press, 1996.

[37] Warren Hunt, William R Mark, and Gordon Stoll. Fast kd-tree construction with an adaptive error-bounded heuristic. In *2006 IEEE Symposium on Interactive Ray Tracing*, pages 81–88. IEEE, 2006.

[38] ST Hyde. Three-dimensional euclidean nets from two-dimensional hyperbolic tilings: kaleidoscopic examples. *Acta Crystallographica Section A: Foundations of Crystallography*, 65(2):81–108, 2009.

[39] Wenzel Jakob. Mitsuba renderer, 2010. http://www.mitsuba-renderer.org.

[40] Wojciech Jarosz, Derek Nowrouzezahrai, Iman Sadeghi, and Henrik Wann Jensen. A comprehensive theory of volumetric radiance estimation using photon points and beams. *ACM transactions on graphics (TOG)*, 30(1):1–19, 2011.

[41] Henrik Wann Jensen. *Realistic image synthesis using photon mapping*. AK Peters/crc Press, 2001.

[42] James T Kajiya. The rendering equation. In *Proceedings of the 13th annual conference on Computer graphics and interactive techniques*, pages 143–150, 1986.

[43] Seung-Wook Kim, Jaehyung Doh, and Junghyun Han. Modeling and rendering non-euclidean spaces approximated with concatenated polytopes. *ACM Transactions on Graphics (TOG)*, 41(4):1–13, 2022.

[44] James T Klosowski, Martin Held, Joseph SB Mitchell, Henry Sowizral, and Karel Zikan. Efficient collision detection using bounding volume hierarchies of k-dops. *IEEE transactions on Visualization and Computer Graphics*, 4(1):21–36, 1998.

[45] Oliver Knill and Elizabeth Slavkovsky. Illustrating mathematics using 3d printers. *arXiv preprint arXiv:1306.5599*, 2013.

[46] E Kopczyński and D Celinska-Kopczyńska. Real-time visualization in non-isotropic geometries (2020), 2002.

[47] Der-Tsai Lee and Bruce J Schachter. Two algorithms for constructing a delaunay triangulation. *International Journal of Computer & Information Sciences*, 9(3):219–242, 1980.

[48] Shihuan Liu, Ming Leng, and Peichang Ouyang. The visualization of spherical patterns with symmetries of the wallpaper group. *Complexity*, 2018, 2018.

[49] J David MacDonald and Kellogg S Booth. Heuristics for ray tracing using space subdivision. *The Visual Computer*, 6:153–166, 1990.

[50] Denis Migranov. A system for visualizing the three-dimensional spherical and elliptic spaces. In *2021 IEEE 22nd International Conference of Young Professionals in Electron Devices and Materials (EDM)*, pages 507–510. IEEE, 2021.

[51] Peter Miley. *A Study of Orbifolds*. PhD thesis, University of British Columbia, 1998.

[52] Jorge J Moré and Danny C Sorensen. Newton's method. Technical report, Argonne National Lab., IL (USA), 1982.

[53] Matthias Nieser, Konstantin Poelke, and Konrad Polthier. Automatic generation of riemann surface meshes. In *Advances in Geometric Modeling and Processing, 6th International Conference, GMP 2010, Castro Urdiales, Spain, June 16-18, 2010. Proceedings*, pages 161–178, 2010.

[54] Tiago Novello, Vincius da Silva, and Luiz Velho. Global illumination of non-euclidean spaces. *Computers & Graphics*, 93:61–70, 2020.

[55] Tiago Novello, Vinícius da Silva, and Luiz Velho. Design and visualization of riemannian metrics. *arXiv preprint arXiv:2005.05386*, 2020.

[56] Tiago Novello, Vinícius da Silva, Luiz Velho, and Mikhail Belolipetsky. How to see the eight thurston geometries. *arXiv preprint arXiv:2005.12772*, 2020.

[57] D. O'Shea. *The Poincare Conjecture: In Search of the Shape of the Universe*. Walker Books, 2007.

[58] Matt Pharr, Wenzel Jakob, and Greg Humphreys. *Physically based rendering: From theory to implementation*. MIT Press, 2023.

[59] Stefan Popov, Johannes Gunther, Hans-Peter Seidel, and Philipp Slusallek. Experiences with streaming construction of sah kd-trees. In *2006 IEEE Symposium on Interactive Ray Tracing*, pages 89–94. IEEE, 2006.

[60] Stefan Popov, Johannes Günther, Hans-Peter Seidel, and Philipp Slusallek. Stackless kd-tree traversal for high performance gpu ray tracing. In *Computer Graphics Forum*, volume 26, pages 415–424. Wiley Online Library, 2007.

[61] Botong Qu, Lawrence Roy, Yue Zhang, and Eugene Zhang. Mode surfaces of symmetric tensor fields: Topological analysis and seamless extraction. *IEEE Transactions on Visualization and Computer Graphics*, 27(2):583–592, 2021.

[62] Lawrence Roy, Prashant Kumar, Sanaz Golbabaei, Yue Zhang, and Eugene Zhang. Interactive design and visualization of branched covering spaces. *IEEE Trans. Vis. Comput. Graph.*, 24(1):843–852, 2018.

[63] Henry Segerman. 3d printing for mathematical visualisation. *The Mathematical Intelligencer*, 34(4):56–62, 2012.

[64] László Szirmay-Kalos and Milán Magdics. Gaming in elliptic geometry. In *Eurographics*, 2021.

[65] László Szirmay-Kalos and Milán Magdics. Adapting game engines to curved spaces. *The Visual Computer*, 38(12):4383–4395, 2022.

[66] William P Thurston. How to see 3-manifolds. *Classical and Quantum Gravity*, 15(9):2545, 1998.

[67] W.P. Thurston and S. Levy. *Three-dimensional Geometry and Topology*. Number v. 1 in Luis A.Caffarelli. Princeton University Press, 1997.

[68] Dmitri Tymoczko. The geometry of musical chords. *Science*, 313(5783):72–74, 2006.

[69] Jarke van Wijk and Arjeh Cohen. Visualization of the genus of knots. page 72, 01 2005.

[70] Eric Veach and Leonidas Guibas. Bidirectional estimators for light transport. In *Photorealistic Rendering Techniques*, pages 145–167. Springer, 1995.

[71] Eric Veach and Leonidas J Guibas. Metropolis light transport. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 65–76, 1997.

[72] Luiz Velho, Vinicius da Silva, and Tiago Novello. Immersive visualization of the classical non-euclidean spaces using real-time ray tracing in vr. 2020.

[73] Marek Vinkler, Vlastimil Havran, and Jiří Bittner. Performance comparison of bounding volume hierarchies and kd-trees for gpu ray tracing. In *Computer Graphics Forum*, volume 35, pages 68–79. Wiley Online Library, 2016.

[74] Freiherr von Gagern et al. *Creating Hyperbolic Ornaments*. PhD thesis, Technische Universität München, 2014.

[75] Martin Von Gagern and Jürgen Richter-Gebert. Hyperbolization of euclidean ornaments. *Electronic Journal of Combinatorics*, 16(2):R12, 2009.

[76] Ingo Wald. On fast construction of sah-based bounding volume hierarchies. In *2007 IEEE Symposium on Interactive Ray Tracing*, pages 33–40. IEEE, 2007.

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

[78] Jeff Weeks. Body coherence in curved-space virtual reality games. *Computers & Graphics*, 97:28–41, 2021.

[79] Jeffrey R Weeks. *The shape of space*. CRC press, 2001.

[80] Jeffrey R Weeks. Real-time animation in hyperbolic, spherical, and product geometries. In *Non-euclidean geometries*, pages 287–305. Springer, 2006.

[81] Francis Williams and Eugene Zhang. Rendering kaleidoscopic scenes using orbifold theory. In *SIGGRAPH ASIA 2016 Technical Briefs*, pages 1–4. 2016.

[82] Katja Wolff and Olga Sorkine-Hornung. Wallpaper pattern alignment along garment seams. *ACM Transactions on Graphics (TOG)*, 38(4):1–12, 2019.

[83] Rüdiger Zeller, Olaf Delgado-Friedrichs, and Daniel H Huson. Tegula–exploring a galaxy of two-dimensional periodic tilings. *Computer Aided Geometric Design*, 90:102027, 2021.

[84] Kun Zhou, Qiming Hou, Rui Wang, and Baining Guo. Real-time kd-tree construction on graphics hardware. *ACM Transactions on Graphics (TOG)*, 27(5):1–11, 2008.

APPENDICES

Appendix A: Proofs for Kaleidoscopic Orbifold Enumeration

In Table 4.1 we provide an enumeration of 2D kaleidoscopic orbifolds based on the cardinality of their underlying polygons and the type of their universal covers. To justify this enumeration, we organize the computation behind this enumeration into a number of theorems in this chapter.

Theorem A.0.1. *Given a kaleidoscopic orbifold $O = *k_1k_2\dots k_N$, O is a spherical orbifold if $N \leq 2$ and a hyperbolic orbifold if $N > 4$. When $N = 4$, O is a hyperbolic orbifold with the only exception of $*2222$, which is a Euclidean orbifold.*

Proof. To show these statements, we only need to compute the Euler characteristics of these orbifolds using Equation 3.1 in the paper which states that Euler characteristic of a kaleidoscopic orbifold $O = *k_1k_2\dots k_N$ is $\chi(O) = \sum_{i=1}^{N} \frac{1}{2k_i} - \frac{N}{2} + 1$.

When $N = 1$, the polygon is a monogon, with one mirror wall that self-intersects at an angle of $\frac{\pi}{k}$. The only good kaleidoscopic orbifold is when $k = 1$, i.e. the wall self-intersects at an angle of π. Note that the orbifold $O = *1$ is usually abbreviated as $*$. Thus, $\chi(O) = \frac{1}{2} - \frac{1}{2} + 1 = 1 > 0$. Consequently, this orbifold is spherical. In fact, this orbifold is a hemisphere with the boundary having the reflectional symmetry.

When $N = 2$, $O = *k_1k_2$ where $1 < k_1 \leq k_2$. Consequently, $\chi(O) = \frac{1}{2k_1} + \frac{1}{2k_2} - \frac{2}{2} + 1 = \frac{1}{2k_1} + \frac{1}{2k_2} > 0$, which means that this type of orbifolds are also spherical.

On the other hand, when $N \geq 4$, we have $k_i \geq 2$ for any $1 \leq i \leq N$. Thus, $\chi(O) = \sum_{i=1}^{N} \frac{1}{2k_i} - \frac{N}{2} + 1 \leq \frac{1}{4}N - \frac{N}{2} + 1 = 1 - \frac{N}{4} \leq 0$. Notice that the equality holds in the above only when $N = 4$ and $k_1 = k_2 = k_3 = k_4 = 2$. Consequently, this type of orbifolds is hyperbolic with the only exception of $*2222$, which is a Euclidean orbifold.

□

Theorem A.0.1 indicates that the more walls there are in the kaleidoscopic orbifold, the more negative its Euler characteristic and the more likely the orbifold being hyperbolic. In contrast, the fewer the walls the more positive its Euler characteristic and more likely the orbifold being spherical. The boundary between the set of spherical orbifolds and the set of hyperbolic orbifolds is drawn when $N = 3$, i.e. triangular orbifolds. The next several theorem inspect this scenario, which consists a number of cases.

Theorem A.0.2. *Given a triangular kaleidoscopic orbifold $O = *k_1k_2k_3$ and without the loss of generality assuming that $3 \leq k_1 \leq k_2 \leq k_3$, O is a hyperbolic orbifold with the only exception of $*333$, which is a Euclidean orbifold.*

Proof. First of all, the assumption that $3 \leq k_1 \leq k_2 \leq k_3$ makes sense since any permutation of k_1, k_2, and k_3 gives rise the same triangular orbifold.

Again, we only need to compute the Euler characteristics of these orbifolds. Here, $\chi(O) = \sum_{i=1}^{3} \frac{1}{2k_i} - \frac{3}{2} + 1 \leq \frac{1}{6}3 - \frac{3}{2} + 1 = 0$. Notice that the equality holds in the above only when $k_1 = k_2 = k_3 = 3$. Consequently, this type of orbifolds is hyperbolic with the only exception of $*333$, which is a Euclidean orbifold.

□

Theorem A.0.2 states that for triangular orbifolds, the higher the minimal order of symmetry at the corners, namely k_1, the more likely the orbifold is hyperbolic. We now consider the case when $k_1 = 2$.

Theorem A.0.3. *Given a triangular kaleidoscopic orbifold $O = *2k_2k_3$ where $2 \leq k_2 \leq k_3$, O is a spherical orbifold if $k_2 = 2$. In contrast, when $k_2 \geq 4$, O is a hyperbolic orbifold with the only exception of $*244$, which is a Euclidean orbifold.*

Proof. Since $N = 3$, when $k_1 = k_2 = 2$ we find the Euler characteristic $\chi(O) = \frac{1}{4} + \frac{1}{4} + \frac{1}{2k_3} - \frac{3}{2} + 1 = \frac{1}{2k_3} > 0$. Thus, in this case the orbifold is always spherical.

On the other hand, when $k_3 \geq k_2 \geq 4$, the Euler characteristics is $\chi(O) = \frac{1}{4} + \frac{1}{2k_2} + \frac{1}{2k_3} - \frac{3}{2} + 1 \leq \frac{1}{4} + \frac{1}{8} + \frac{1}{8} - \frac{3}{2} + 1 = 0$. Notice that the equality holds in the above only when $k_2 = k_3 = 4$. Consequently, this type of orbifolds is hyperbolic with the only exception of $*244$, which is a Euclidean orbifold.

□

The last remaining case is when $O = *23k_3$, which is covered in the next theorem.

Theorem A.0.4. *Given a triangular kaleidoscopic orbifold $O = *23k_3$ where $3 \leq k_3$, O is a spherical orbifold if $k_3 < 6$, a Euclidean orbifold if $k_3 = 6$, and a hyperbolic orbifold if $k_3 > 6$.*

Proof. The Euler characteristic of this type of orbifolds is $\chi(O) = \frac{1}{4} + \frac{1}{6} + \frac{1}{2k_3} - \frac{3}{2} + 1 = \frac{1}{2k_3} - \frac{1}{12} = \frac{6-k_3}{12k_3}$. Thus, $\chi(O)$ is positive when $k_3 < 6$, zero when $k_3 = 6$, and negative when $k_3 > 6$. Consequently, O is a spherical orbifold if $k_3 < 6$, a Euclidean orbifold if $k_3 = 6$, and a hyperbolic orbifold if $k_3 > 6$.

□

Interestingly, each of the above theorems contains a Euclidean orbifold: $*2222$ for Theorem A.0.1, $*333$ for Theorem A.0.2, $*244$ for Theorem A.0.3, and $*236$ for Theorem A.0.4. Not only do these facts confirm that there are only four Euclidean kaleidoscopic orbifolds, but they also show the transition from spherical orbifolds to hyperbolic orbifolds with more walls and higher-order symmetries at the corners.

Appendix B: Optics-Based Visualization for Orbifold Concept and Properties

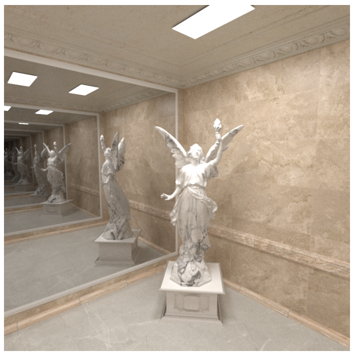
Our system can be used to generate example scenarios to illustrate important concepts and properties of orbifolds such as the following. Given a room with the statue Lucy, we first mount a mirror each on two adjacent walls (Figure B.1 (a)). This leads to an illusion of a space that is four times as large as the room without a mirror. The virtual space is the *universal cover* of the orbifold (the original room).

In addition, the symmetry for the room can be understood by checking the orientations of the statues in the space. While the statue has her left hand up holding the torch in the original room, each mirror generates a virtual statue who raises the torch by her right hand (a reflection). Interestingly, reflecting the statue in the first virtual room with respect to the second mirror leads to the third virtual statue, who switches back to her left hand to raise the torch. However, this virtual statue faces the opposite direction of the statue in the original room, i.e. a rotation by π. One can consider the reflected and rotated virtual copies as the result of the *action* of the *symmetry group* of the underlying orbifold. This group consists of the identity action, two reflections (one per each mirror), and one rotation (the composite of the two mirrors).

By moving one of the mirrors to the wall opposite the other mirror, we obtain a different scene where there are infinitely many copies of the original room (Figure B.1 (b)). In fact, the universal cover of this orbifold can be generated by first grouping the original room with one of the reflections and then translating infinitely many times the two rooms by a distance that is a multiple of twice the room depth. The union of the two rooms (the real room and the virtual room) is thus referred to as a *translational cover*.

When a mirror is mounted on each wall (Figure B.1 (c)), we obtain the orbifold whose translational cover is the same as the universal cover of the room shown in Figure B.1 (a). This translational cover is then translated in two mutually perpendicular directions. Note that this is the first orbifold (in this example) that we have encountered

(a) **two adjacent mirrors**


(b) **two parallel mirrors**

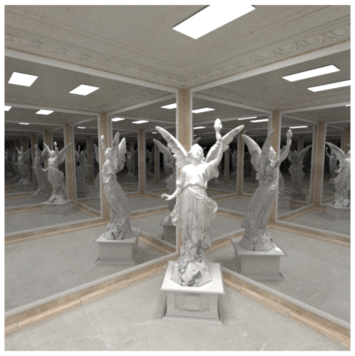
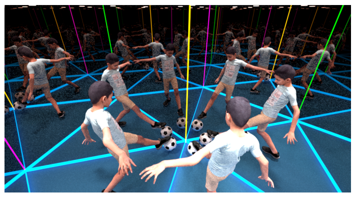
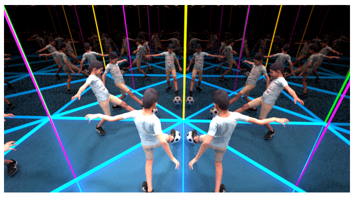

(c) **four mirrors**

Figure B.1: A square room with two or four mirrors. The case in the four mirror room (c) corresponds to a Euclidean orbifold $*2222$.


(a) *333


(b) $*244$

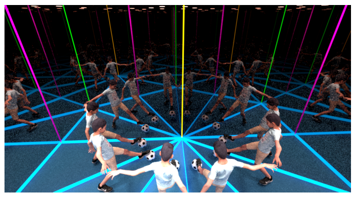

(c) $*236$

Figure B.2: The three triangular Euclidean kaleidoscopic orbifolds.

where all walls have a mirror. This room corresponds to the $*2222$ orbifold. Each of the corner has an angle of $\frac{\pi}{2}$, thus its notation. At such a corner, there are $2k$ copies of the original room forming k pairs. Inside each pair, one of the rooms is a rotational copy of the original room while the other is a reflectional copy. A *kaleidoscopic orbifold* has a *transformation group* that is generated by mirror reflections. The *subgroup* for each corner is thus $\mathbb{D}_k$, the *Dihedral group* of *order* k. In the $*2222$ case, the symmetry group at every corner is the same, i.e. $\mathbb{D}_2$.

$*2222$ is one of the four Euclidean kaleidoscopic orbifolds, i.e. whose universal cover is the Euclidean plane. Figure B.2 shows the other three such orbifolds: (1) $*333$, (2) $*244$, and (3) $*236$. The $*333$ orbifold (Figure B.2 (a)) is obtained by placing three mirror walls in a $\frac{\pi}{3}-\frac{\pi}{3}-\frac{\pi}{3}$ triangular room. Its translational cover consists of six copies of the original room ($\mathbb{D}_3$). Similarly, the $*244$ orbifold (Figure B.2 (b)) is obtained by placing three mirror walls in a $\frac{\pi}{2} - \frac{\pi}{4} - \frac{\pi}{4}$ triangular room. Its translational cover consists of eight copies of the original room ($\mathbb{D}_4$). The $*236$ orbifold (Figure B.2 (c)) is generated by placing three mirror walls in a $\frac{\pi}{2} - \frac{\pi}{3} - \frac{\pi}{6}$ triangular room. Its translational cover consists of 12 copies of the original room ($\mathbb{D}_6$). Notice that the symmetry group can vary from corner to corner. In addition, note that an orbifold does not depend on which corner is referred to as the first corner. Thus, $*236$ and $*362$ represent the same orbifold. Similarly, the orbifold does not change when the corners are numbered in the opposite order. Thus, $*236$ and $*632$ also represent the same orbifold.