Danmarks
Tekniske
Universitet

02561 Computer Graphics Project
Planar Reflector

Authors

Xiaowei Hou - s232255

December 16, 2025

Contents

1 Introduction

Realistic reflections which are significant clue for understanding object and space are crucial element in computer graphics. In rasterization-based pipelines, reflections are not directly available. Therefore, in order to achieve a excellent reflection, geometric or multi-pass rendering techniques are required. This project focuses on *planar reflections*, i.e., reflections on a single flat reflector such as a ground plane. The core idea of the method is to simply render a mirrored copy of an object through reflection transformation with respect to the reflector plane.

Although this approach produce realistic results, it still has many flaws: (1) the reflected object may appear outside the reflector region, (2)the reflection may incorrectly include geometry that lies behind the reflector plane, etc. In this project, these flaws will be addressed step by step, finally leading to a satisfactory solution.

Based on the above, the project is divided into four parts: (1) basic mirrored rendering, (2) blending with a partially transparent reflector, (3) using stencil buffer for clipping (4) ***oblique near-plane clipping*** by aligning the near plane of the view frustum with the reflection plane when drawing reflected objects. [1]

2 Setup and Rendering Pipeline

The scene rendered in the canvas is consists of two components: a jumping teapot and a plane reflector. The camera is a pinhole camera with a vertical field of view of 65°. Its position is fixed at $(0, 0, 1)$, looking towards the point $(0, 0, -3)$ with an up vector $(0, 1, 0)$. There is an single point light source moving along circular trajectory above the object is used for illumination. Phong style shading is applied to the teapot, with adjustable parameters.

In rasterization-based pipelines, planer reflection cannot be generated in a single render pass. Therefore, multi-pipeline pass are employed. Reflection and normal scene are rendered separately. This approach allows each visual effect shadow, reflections and clipping to be implemented in a controlled and modular manner.

3 Reflected object

Planer reflection can be implemented by geometric transformation. In this part, an transformation that mirrors an object with respect to a plane is introduced. Given a vertex $\mathbf{P}$ on the planar reflector's surface and a unit vector $\mathbf{V} = (V_x, V_y, V_z)^T$ perpendicular to the plane, the reflection transformation can be expressed as the following 4×4 matrix[2].

$$\mathbf{R} = \begin{pmatrix} \mathbf{I}_{3\times3} - 2\mathbf{V}\mathbf{V}^T & 2(\mathbf{P}\cdot\mathbf{V})\mathbf{V} \\ \mathbf{0}^T & 1 \end{pmatrix} = \begin{pmatrix} 1-2V_x^2 & -2V_xV_y & -2V_xV_z & 2(\mathbf{P}\cdot\mathbf{V})V_x \\ -2V_xV_y & 1-2V_y^2 & -2V_yV_z & 2(\mathbf{P}\cdot\mathbf{V})V_y \\ -2V_xV_z & -2V_yV_z & 1-2V_z^2 & 2(\mathbf{P}\cdot\mathbf{V})V_z \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{1}$$

In this project, a horizontal plane $y = y_0$ is used as a reflector. Therefore the simplified reflection matrix can be derived:

$$\mathbf{R} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 2y_0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{2}$$

For a horizontal plane, the reflection can be expressed as a combination of translation and non-uniform scale. In the actual implementation, the reflection is a sequence of elementary transformations rather than the matrix in Eq. (2) explicitly. Reflection about the plane $y = y_0$ is implemented as

$$\mathbf{R} = \mathbf{T}(0, y_0, 0)\,\mathbf{S}(1, -1, 1)\,\mathbf{T}(0, -y_0, 0), \tag{3}$$

where $\mathbf{T}$ denotes translation and $\mathbf{S}$ denotes non-uniform scaling.

Applying the reflection, the object is rendered twice. Once normally, and once using reflected model matrix.

Besides transforming the vertex positions, transformation of lighting-related quantities are also required. Normals are implicitly reflected by the model transformation, therefore lighting computations are performed in the reflected space.

The expected visual effect, an mirrored copy of the object under the real object, is achieved. The result is shown in Figure 1. The ground is temporarily removed to clearly observe the reflected geometry.

Figure 1: Result of Part 1

4 Reflector

In this part, the ground is reinserted as planer reflector. After applying transformation described in Eq.3, the reflected teapot appears correctly mirrored below the ground. The ground quad is set to semi-transparent to make visual effect better. The result is shown in Figure 2. The basic illusion of reflection is created correctly.

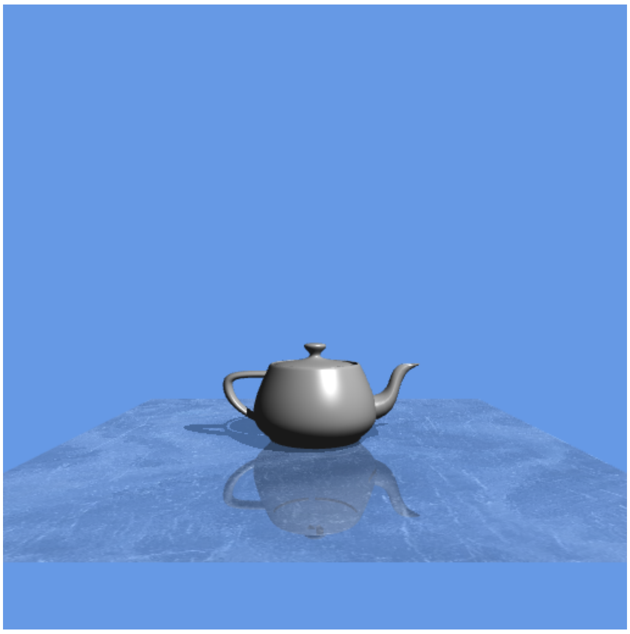

Figure 2: Result of Part 2

However, this approach don't limit the reflected teapot inside the ground. As shown in Figure 3. This flaw break the physical rationality.

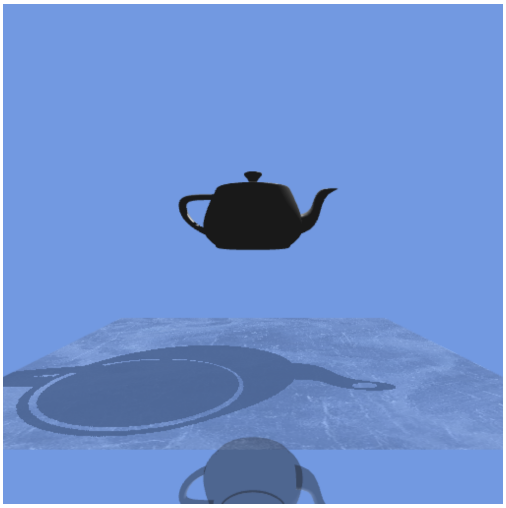

Figure 3: Reflection Outside the Ground

Furthermore, depth testing cannot be used to clip the reflected geometry against the ground plane. These limitations motivate the use of stencil buffer to constrain where reflected geometry may appear. This technique will be illustrated in next section.

5 Stencil Buffer for Clipping

Stencil based clipping is implemented using a two-step rendering procedure.

1. **Stencil marking** The ground is rendered with color writes disabled and depth testing disabled. The stencil operation is configured to replace the stencil value with 1 for all fragments covered by the ground plane.
2. **Reflected object** The reflected teapot is rendered using the reflection transformation described in Eq 3. Only fragments with a stencil value equal to 1 are accepted. This ensures that the reflected object is visible only within the ground region.

The result of this part can be seen in Figure 4.

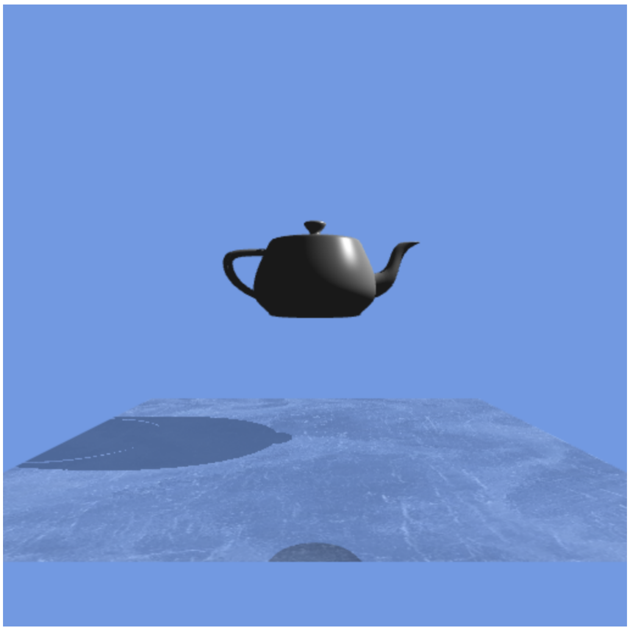

Figure 4: Result of Part 3

6 Oblique Near-Plane Clipping

In previous section, stencil buffer limits the reflected teapot inside the ground successfully. However, it cannot prevent object behind the reflector plane from being reflected. It can be seen in Figure 5

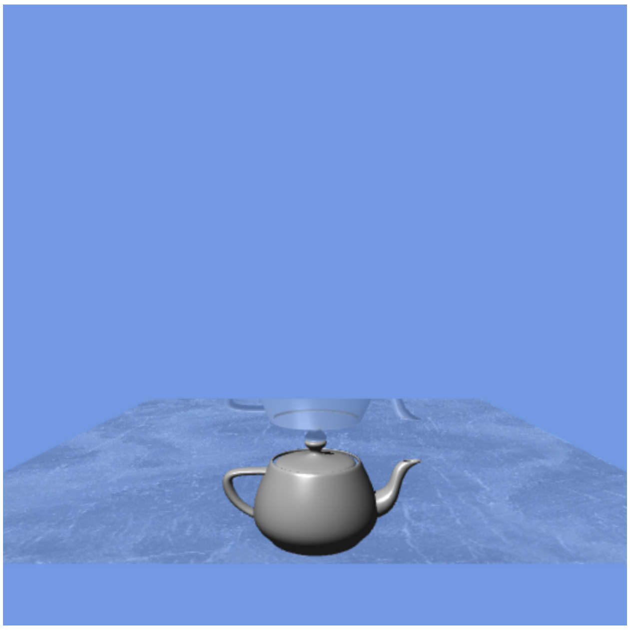

Figure 5: Reflection

The root cause of this issue is the reflect transformation described in Eq 1 mirrors the entire object regardless of its position relative to the plane. Therefore, explicitly clipping is required to clip the geometry shouldn't be reflected.

In this project, *oblique near-plane clipping* as described by Lengyel [1] is employed. The key idea is to modify the view frustum used during the reflection pass such that its near clipping plane coincides with the reflector plane. The reflector plane is defined in world space by the plane equation

$$ax + by + cz + d = 0. \tag{4}$$

Given the reflector plane expressed in eye space, the standard perspective projection matrix can be modified to incorporate the oblique near plane. Following Lengyel's method, a new projection matrix is constructed as

$$\mathbf{P}_{\text{oblique}} = \text{modifyProjectionMatrix}(clipplane, \mathbf{P}), \tag{5}$$

where $\mathbf{P}$ is the original perspective projection matrix. This modified projection matrix replaces the original one only during the reflection pass.

Figure 6 shows the result.

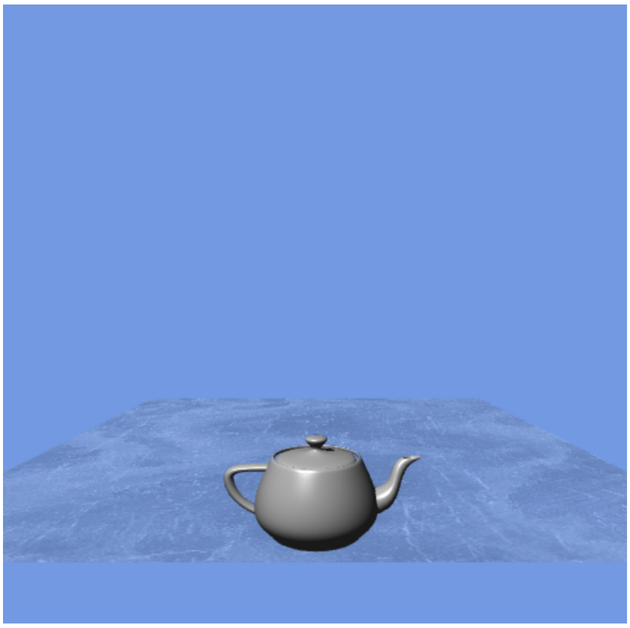

Figure 6: Result of Part 4

7 Conclusion

This project explored the implementation of planar reflections in a rasterization-based rendering pipeline using WebGPU. From an simple mirror transformation, a sequences of techniques are introduced to implement a realistic reflection. These techniques form a foundation for more advanced reflection and shadowing methods in real-time graphics.

The source code and full implementation of this project are available online at **my student homepage**.

References

[1] E. Lengyel, *Mathematics for 3D Game Programming and Computer Graphics*. Boston: Course Technology, third ed., 2012.

[2] R. Goldman, "Matrices and transformations," *Graphics Gems*, pp. 472–475, 1990.