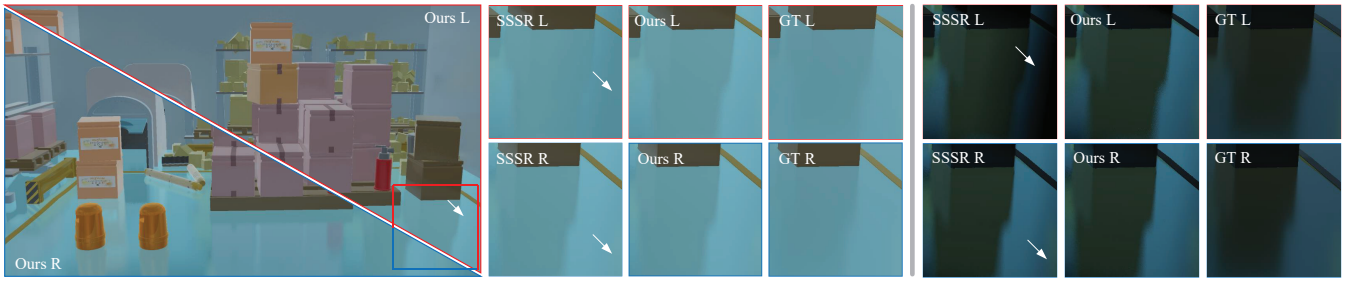
DOI: 10.1111/cgf.15159
Eurographics Symposium on Rendering 2024
E. Garces and E. Haines
(Guest Editors)

COMPUTER GRAPHICS *forum*
*Volume 43 (2024), Number 4*

# Stereo-consistent Screen Space Reflection

X. Wu[1], Y. Xu[1] and L. Wang[†1]

[1]School of Software, Shandong University, China




(a) Ours Global Illumination (b) Global Illumination Comparison (c) Reflection Result Comparison

**Figure 1:** *We propose stereo-consistent screen space reflection to eliminate the inconsistent cues introduced by incomplete geometry, view-dependent fading boundary, and reflection sample. Our method fully reuses information from two views, getting more consistent and reliable results (in PICA Scene). The SSIM is obtained by calculating the similarity between the stereo rendering result and the stereo reference (SSIM values are calculated for the left and right eyes, respectively).*

**Abstract**

*Screen Space Reflection (SSR) can reliably achieve highly efficient reflective effects, significantly enhancing users' sense of realism in real-time applications. However, when directly applied to stereo rendering, popular SSR algorithms lead to inconsistencies due to the differing information between the left and right eyes. This inconsistency, invisible to human vision, results in visual discomfort. This paper analyzes and demonstrates how screen-space geometries, fade boundaries, and reflection samples introduce inconsistent cues. Considering the complementary nature of screen information, we introduce a stereo-aware SSR method to alleviate visual discomfort caused by screen space disparities. By contrasting our stereo-aware SSR with conventional SSR and ray-traced results, we showcase the effectiveness of our approach in mitigating the inconsistencies stemming from screen space differences while introducing affordable performance overhead for real-time rendering.*

**CCS Concepts**

• ***Computing methodologies*** → *Rendering;* • ***Rendering*** → *Real-Time Rendering; Screen space reflection; Stereo consistent;*

## 1. Introduction

Virtual Reality (VR) headsets are becoming increasingly popular in the gaming and video industries due to their capacity to deliver superior immersive experiences. An efficient global illumination (GI) algorithm can improve the user's realistic experience. However, ray tracing for GI is prohibitively expensive for real-time applications, especially for stereo rendering. Precomputed methods reduce runtime costs but increase storage overhead and aren't always suitable for dynamic scenes. Hence, many real-time applications obtain global illumination information by multiplexing screen space information, and screen space reflection (SSR) [Ulu18] is one of them.

SSR takes the screen space geometry as the basis to obtain an approximate indirect illumination result by tracing reflection rays in the screen space. While SSR can get a good reflection effect in monoscopic rendering, it will inevitably lead to inconsistent results in stereo rendering due to the inconsistency of screen space geometries, leading to a potential source of discomfort [KT04].

Different from other global effects, such as ambient occlusion

† Corresponding author: luwang_hcivr@sdu.edu.cn.

(AO) [SBE22], for the same point in three-dimensional space, the visual effect seen by two eyes is not the same for reflection. This is because the reflected directions of the two views are different, which causes inconsistent geometry information, resulting in inconsistent colors/radiance results. These inconsistencies are physically correct and can be solved by binocular vision. However, the inconsistent cues introduced by stereo SSR are physically incorrect. Hence, eliminating inconsistent cues is an important goal when using SSR in stereo rendering to improve visual effects.

In this paper, we reveal three significant sources for inconsistent cues of SSR and propose a stereo-aware SSR method to reduce the inconsistency caused by employing screen-space reflection solutions for stereoscopic views. Though a small amount of computational overhead is introduced, our method still maintains real-time performance. The three inconsistent cues are:

**Inconsistent screen space geometry.** Since the screen space geometries are incomplete for the entire world space, using two different and incomplete screen space geometries to generate reflection results in stereo rendering is bound to produce inconsistent cues, and confusing reflection results.

**Inconsistent fading boundary.** The fading effect fades out SSR near the screen's boundary to avoid reflections popping under different screen borders. However, the two eyes have different screen boundaries, which makes the fading effect of the two eyes inconsistent.

**Inconsistent reflection sample.** Current popular SSR algorithms only allow for low sampling rates for reflection rays to guarantee real-time performance. Excessively divergent and incomplete reflection samples caused by stochastic sampling between two eyes can lead to unsolvable visual inconsistency.

Our main contributions are:

- an efficient stereo-aware SSR traversal scheme to mitigate inconsistency caused by unaligned screen space geometries of two views,
- a heuristic stereo-aware fading method to fade out the artifact near the screen boundary in stereo SSR,
- a stereo-aware and ghosting-aware reprojection filter, alleviating the fault caused by inconsistent reflection samples of two views,
- a user study to evaluate the capability to mitigate inconsistent cues and improve the visual quality for stereo rendering.

## 2. Related Work

**Screen space reflection.** SSR is a family of techniques used in real-time rendering for employing screen space information to calculate reflections in rendered images [Beu20]. Souza et al. [SKS11] introduce the concept of SSR and use it in video games; based on geometry pass and lighting pass, rays are emitted in screen space to simulate the trace process of secondary rays to obtain indirect illumination. The way ray traverse in screen space greatly affects the efficiency of SSR. The linear scheme is a common tracing method. 3D ray marching is one of the earliest forms of SSR used by Souza et al. [SKS11], which works by sampling points along with the ray direction. However, finding an appropriate fixed step size in ray marching is complex, and the result is easily associated with over-sample or under-sample. To avoid over-sampling and under-sampling, McGuire et al. [MM14] and Amanatides et al. [AW*87] take a digital differential analyzer (DDA) algorithm, using the slope of the 2D line segment to obtain the depth of the intersection point by binary search of the step size. Another traverse scheme is the hierarchical scheme. Uludag et al. [Ulu18], and Widmer et al. [WPS*15] construct a hierarchical depth structure (Min-Max Hi-Z) to skip large amounts of empty space. Frostbite engine [Sta15] proposes a stochastic SSR (*SSSR*) algorithm to obtain a more realistic and accurate reflection. In this paper, we use this approach as a baseline for SSR. Although SSR can capture reflections quickly (compared to ray tracing), its view-dependent property makes it challenging to use in stereo rendering.

**Accurate reprojection.** In real-time applications, where computational resources are scarce, existing information is often reused to obtain better results. Walter et al. [WDP99], Nehab et al. [NSL*07], and Yang et al. [YLS20] reuse the temporal information by reprojecting the historical information, expecting to get better results for the current frame, whereas Yang et al. [YLS20] widely used in game applications. Furthermore, image-based rendering utilizes projections of image information [ZK07, HRDB16], which saves computational resources by remapping existing image representations to appropriate locations, projecting the original image information onto the new image. There are studies on accurate motion vectors for more accurate projection information. Mara et al. [MMBJ17] propose a motion vector to compute a specular surface, which utilizes the motion of the virtual image locations to track the movement of glossy reflections. Hirvonen et al. [HSAS19] consider the surface curvature, obtaining the more accurate position of the virtual image. However, both of them are effective when purely specular. Zeng et al. [ZLY*21] sample the reflective positions based on roughness, thus ensuring the accuracy of the motion vectors in case the materials are rougher.

**Stereo rendering.** Mäkitalo et al. [MKKJ19] propose a reuse-based method reprojecting the monoscopic path-space samples to stereoscopic views, reducing the computational complexity in stereo ray tracing. Wißmann et al. [WMFL20] take a hybrid rendering pipeline, utilizing projections and a small amount of ray tracing complements to speed up stereo rendering. Philippi et al. [PFJ23] propose a method of stereo filtering, which reduces errors caused by stereo reprojection by calculating an adaptive parameter while trying to ensure the depth perception is as complete as possible. Stereo-consistent algorithms are also a concern in stereo rendering. Northam et al. [NAK12] ensure stereoscopic consistency in stylized rendering by performing stylization on the merged disparity map, in which the regions correspond in stereoscopic space. He et al. [HWB19] render the stereo contour line information as a concatenated set, ensuring the final result's consistency. Shi et al. [SBE22] improve AO consistency in stereo by using stereo-aware obscure estimation and stereo-aware bilateral filtering, which solves the stereo inconsistency caused by AO prediction and filtering.

**Physiology of the eye.** Watkins et al. [WHP*01] got the binocular vision that can significantly improve the perceptual separation of foreground details and their background. However, research

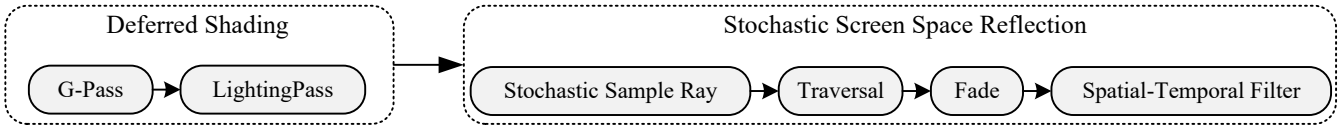



**Figure 2:** *Pipeline of the stochastic screen space reflection (SSSR) algorithm [Sta15], in which the stochastic sample ray process will introduce insufficient reflection sampling, the traversal process will introduce inconsistent screen space geometry, and the fading process will introduce inconsistent fading boundary.*

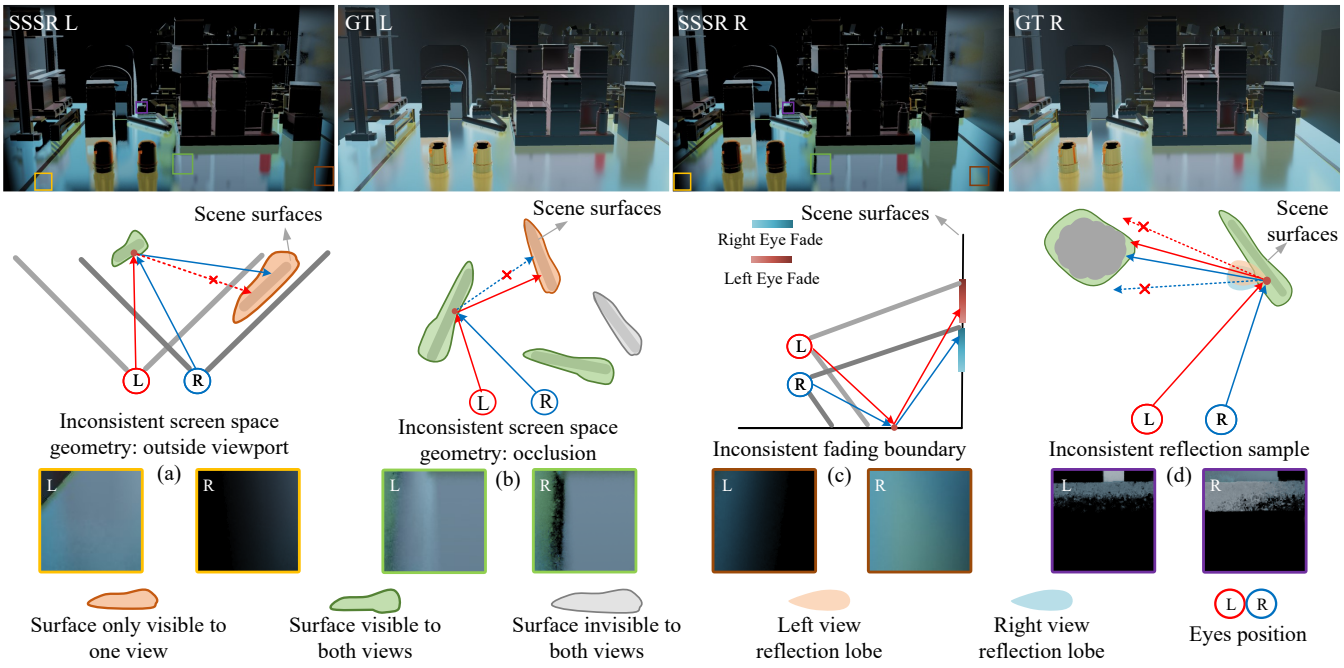



**Figure 3:** *Inconsistent cues. The gray lines identify Eyes' frustum. The red lines are rays from the left eye, and the blue are rays from the right eye. Cropped regions show reflections of the same reflected object in the left and right eyes. (a) Some surfaces are visible to only one eye because the object is beyond the frustum boundary, and reflected rays cannot hit these surfaces. (b) Some surfaces are visible to only one eye because other objects occlude them. (c) The reflected rays emitted by the two eyes fade respectively. (d) Insufficient sampling results in different noises in the reflection rays randomly sampled by the two eyes.*

by Ijsselsteijn et al. [IdRV00], Pfautz et al. [Pfa01], and Siegel et al. [Sie99] highlight serious drawbacks associated with the use of stereo rendering. Human visual perception relies on the combination of many visual cues. The perceptual system receives conflicting information when these cues are inconsistent, leading to visual discomfort.

The inconsistency problem with SSR methods in stereo rendering requires immediate attention to enhance the visual experience. Note that SSR in this paper specifically addresses techniques related to reflection (excluding screen space refraction [Wym05]).

## 3. Motivation

The pipeline of the stochastic screen space reflection (*SSSR*) algorithm proposed by Tatarchuk et al. [Sta15] is shown in Figure 2. After getting the G-Buffer and direct illumination results using deferred shading, the reflected rays from the screen space are randomly sampled for a given pixel. Then, the rays travel through the depth buffer in screen space to find potential intersection points. To ensure smooth reflection results near the screen boundaries, fading is applied to mitigate abrupt changes. Finally, the reflection results are usually filtered using temporal-spatial filters to decrease noise.

However, if *SSSR* is directly applied in stereo rendering, three inconsistent cues will appear in stochastic sample ray, ray traversal, and fading passes, respectively.

- During ray traversal, when the reflected object is only located in one view frustum (Figure 3 (a)) or is occluded by another object (Figure 3 (b)), the reflection effects of two eyes will be different, because the same reflected object is only visible to one view.
- During the fading pass, the fade-out effect is generated near each eye's frustum boundaries, leading to apparent visual inconsistency, as shown in Figure 3 (c).
- Finally, the *SSSR* uses random sampling to obtain reflected rays. A large amount of reflection noise will be generated due to the

ray hitting miss. Even after the spatial-temporal filter, the inconsistent reflection results of two eyes caused by different noises are still noticeable for binocular vision, see Figure 3 (d).

Considering the screen space information of two views can be reused to complement each other, we propose an SSSR-based stereo rendering method, which can handle the three former inconsistent cases and efficiently reuse the complement information of two views.

## 4. Method

Our method follows the general pipeline of *SSSR* [Sta15] and proposes a stereo-aware reflection pass to avoid stereoscopic inconsistencies. In our solution, we take advantage of the complementary property of stereoscopic information to traverse reflection rays in stereo screen space and resolve inconsistent cues generated by invisible geometries. Also, we use a heuristic fading method to mitigate the artifact introduced by inconsistent screen boundaries. We further use a glossy projection vector to assist spatial reprojection, eliminating the inconsistent sample cues due to stochastic sampling.

### 4.1. Stereo-Aware Traversal For Inconsistent Screen Space Geometry

The reflected object must be consistent with both views. We guarantee this using the union of screen space geometries when traversing rays. For stereo rendering, $\mathcal{V}_1$ and $\mathcal{V}_2$ denote the left and right views, respectively.

To take the screen space information from both views into account efficiently, in each ray walking step, we first determine whether the ray's position is occluded or outside the range of viewport of the current view; if so, do extra traversal using the screen-space information of the other view. Otherwise, no extra calculation is needed. In practice, we use different strategies to handle the two situations in Figure 3 (a) and (b).

For situations where a surface is only inside one view's frustum, as shown in Figure 3 (a), if a reflection ray reaches beyond the boundary of $\mathcal{V}_1$'s viewport, we continue to judge whether the ray is inside the boundary of $\mathcal{V}_2$'s viewport, if it is still inside, continue ray walking by using $\mathcal{V}_2$'s information.

For situations where a reflected object is occluded by other objects in the current view, as shown in Figure 3 (b), the reflection ray does not extend the boundary of $\mathcal{V}_1$, but the intersection is occluded in $\mathcal{V}_1$. Before we try to adopt the information of $\mathcal{V}_2$, we first make sure that the current ray's position is not inside any object of view $\mathcal{V}_1$, only under this condition can the ray continue to traverse. As shown in Figure 4, we use dual depth detection to achieve this. We store the depth of all front faces and the depth of all back faces in the deferred shading pass. After that, for each ray traversal step, we test whether the depth of the ray's endpoint is larger than the value stored in the back-face depth buffer of $\mathcal{V}_1$. If so, we consider the ray has reached the back of the object and use $\mathcal{V}_2$'s information to do further tracing.

These stereo-aware traversal strategies ensure that the information about the scene geometry remains consistent in stereo, while fully utilizing the space information from both views. With our dual depth detection, the increased traversal overhead is minimal. Moreover, due to geometry consistency, the results are temporally stable.

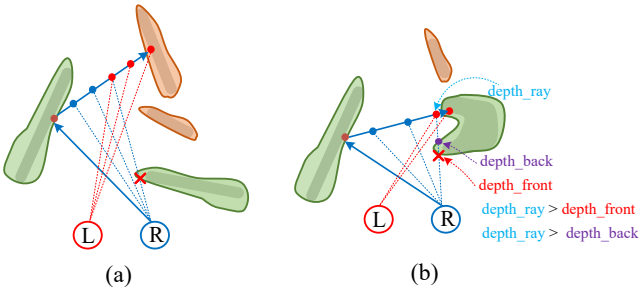



**Figure 4:** *Dual depth detection for ray traversal in the case of occlusion. During ray walking, if the depth of the ray's current position is greater than the front face depth stored in image space, it means that an object in the current view occludes the ray. We further confirm that the depth of the ray is larger than the back face depth, which means the ray is outside the occluder; after that, the ray continues traversal by using the other view's screen space information. Both situations in (a) and (b) can maintain accuracy after applying dual depth detection.*

### 4.2. Stereo-Aware Fade For Inconsistent Fading boundary

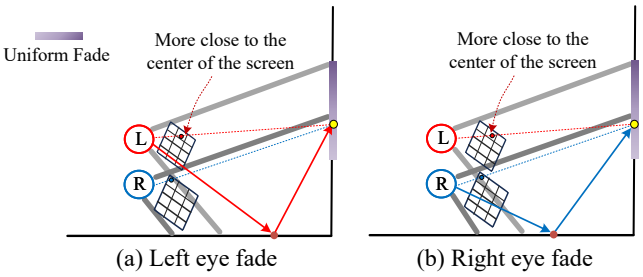



**Figure 5:** *Uniform stereo-aware Fade. The reflected point (yellow point) should be projected to both views. The projection point close to the center of the screen will be used to control the fading parameter.*

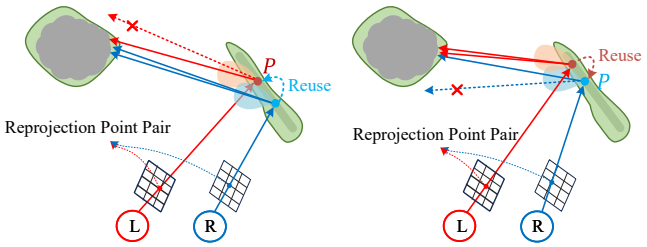
Due to the lack of information available in the screen space, rays should be faded away to remove artifacts in some cases, such as when rays fall close to the screen borders, when reflection directions are close to the inverse view direction, or when rays have traveled a long distance. Only smoothly faded rays that end up close to the screen edges are related to the screen-space geometries, which will cause inconsistency in stereo rendering.

The fade strategy is a compromise solution to face the problem of incomplete screen-spaced information, eliminating visual discomfort through fading effects. For stereo rendering, we prefer to maintain clear reflection results rather than over-blurring. Thus, we use a unified fading parameter for both views to better keep reflection details by considering the boundary information of the two views.

As shown in Figure 5, if an intersection point for a view is needed to perform a fading operation, the point should be reprojected to the other view simultaneously. For those two projected intersection points, we choose the one closer to the center of the screen to compute the fading parameter and control the blurring level. In this way, we can use the lower blurring level of the two eyes, which could maintain consistency between the two eyes while keeping the details of the reflections.

### 4.3. Stereo-Ghosting-Aware Reprojection For Inconsistent Reflection Sample

A stochastic sampling of reflection rays inevitably introduces residual noise, which cannot be removed entirely, even with spatial and temporal filtering. A few samples introduce a lot of such inconsistent stereoscopic cues, inhibiting stereo visual experience. To mitigate the inconsistent cues caused by insufficient samples and improve the reflection results' quality, we adopt the glossy projection vector [ZLY*21] to map the reflection information of two eyes.




(a) Reprojection from R to L (b) Reprojection from L to R

**Figure 6:** *Reprojection reflection samples. Randomly sampled ray hitting miss (dotted line with a cross) get unavailable geometry information (residual noise). Samples from the area around the reflection point P will be reused to get better and more consistent reflection results.*

Inspired by the idea that TAA [YLS20] mitigates artifacts through temporal reuse, our method takes a pure specular approach to compute projection vectors in the stereo-view space, finding the reprojection point pairs between the two eyes. Our approach adds the stereoscopic spatial information reuse, archiving better single-frame results and making the temporal accumulation result more stable and consistent.

As shown in Figure 6, we reproject between point pairs to increase the number of reflection samples, improving the consistency of the geometry information obtained from the samples. In addition, we use $\frac{BRDF}{pdf}$ as the blend weight in reprojection to retain the information of the more important samples. Consequently, our method mitigates the inconsistent signals introduced by the insufficient samples, and obtains better reflection results.

**Find Reprojection Point Pairs.** Similar to the method of Zeng et al. [ZLY*21] and Mara et al. [MMBJ17], we also assume that the virtual image is a real object behind the mirror reflector and calculate the related pixels of two views by using the virtual image instead of the reflector.

During the reflection, the ray is emitted from a point $P_o$ on an object, bouncing at $P_S$ on the reflector surface and finally entering the eye. $P_{o'}$ is the virtual image of $P_o$. Note that, $P_o$, $P_S$ and $P_{o'}$ indicate the points in 3D space, and ${P_s}^{\mathcal{V}}$ is $P_S$'s projection point in a view's screen space.

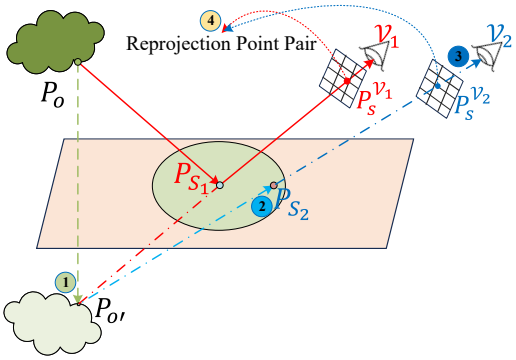


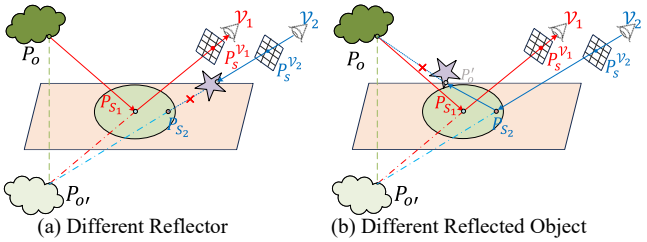

**Figure 7:** *Find Reprojection point pairs. The solid red line is the reflection ray cross ${P_s}^{\mathcal{V}_1}$ on the left eye. (1) Find the virtual image position $P_{o'}$. (2) Find the 3D intersected point $P_{S_2}$ on the virtual plane. (3) Projection $P_{S_2}$ to right eye's screen space get ${P_s}^{\mathcal{V}_2}$. (4) Reprojection point pair $\{{P_s}^{\mathcal{V}_1}, {P_s}^{\mathcal{V}_2}\}$.*

As shown in Figure 7, for a view (such as the left eye), ${P_s}^{\mathcal{V}_1}$ is the point on the reflector (in the left eye's screen space), and $P_{S_1}$ is its related point in 3D space, we reproject it to the right view to find the related reflection point ${P_s}^{\mathcal{V}_2}$. We assume a locally flat virtual plane lies in 3D space around $P_{S_1}$ and find the intersected point $P_{S_2}$ on this plane with the ray-traced from $P_{o'}$ towards the right eye. The intersection point $P_{S_2}$ is then projected into the right eye's screen space, i.e., ${P_s}^{\mathcal{V}_2}$. Then we can reuse the reflection result of ${P_s}^{\mathcal{V}_2}$ for ${P_s}^{\mathcal{V}_1}$.

After we find the mapping in stereoscopic space, there is still a problem to be solved. The surface of an object is usually not purely specular, so the projection vectors we derive are not exactly correctly mapped in the case of high roughness. To obtain the stochastic glossy motion vector, Zeng et al. [ZLY*21] sample on a circle centered at the point $P_S$. However, in the *SSSR* process, the reflection results are usually processed using spatial filtering for each view. Since ${P_s}^{\mathcal{V}}$ represents an area of its geometric neighborhood, we can directly reuse the illumination result of point ${P_s}^{\mathcal{V}_2}$ into ${P_s}^{\mathcal{V}_1}$, even for reflectors with high roughness. Also, note that in our approach, we decompose the spatial filter of the original *SSSR* into reprojection and a spatial filter, ensuring that no additional computational overhead is required. Additionally, in the case of delta reflection, we do not use reprojection.

**Fix Incorrect Reuse.** Thanks to the "point pairs" between ${P_s}^{\mathcal{V}_1}$ and ${P_s}^{\mathcal{V}_2}$, reflection results can be spatially reused between two views (Figure 7), so that the inconsistent noise will be effectively removed. However, artifacts are inevitable due to the lack of accurate computation to find the "point pairs", which may introduce new inconsistency cues. We focus on two major cases of unreliable reprojection when complex occlusions occur.

As shown in Figure 8 (a), $P_{S_2}$ is occluded by another occluder,




(a) Different Reflector (b) Different Reflected Object

**Figure 8:** *Incorrect reuse case. (a) The reflector surface point $P_{S_2}$ is occluded in $\mathcal{V}_2$. (b) The reflected ray of $P_{S_2}$ is occluded by $P_o'$ on another object.*

thus $P_s^{\mathcal{V}_2}$ indicates another point, which reflection result may be significantly different from $P_s^{\mathcal{V}_1}$. In this case, we prohibit the reprojection of those "point pairs" with different reflectors to minimize the wrong mapping.

The other case is shown in Figure 8 (b), $P_{S_1}$ and $P_{S_2}$ are located on the same reflector, but the reflected ray is occluded by another object $P_o'$ before it reaches $P_o$, which is also disallowed "point pairs" for reprojection. To achieve this, we store the depth information of reflected points, i.e., the depth of $P_o'$ is stored on screen for $\mathcal{V}_2$, before we reuse $P_s^{\mathcal{V}_2}$, we test whether the pre-stored reflect depth of $P_o$ is significantly different from the depth of $P_o'$, where the tolerance for the difference depends on roughness. In this way, we can effectively avoid incorrect reuse.

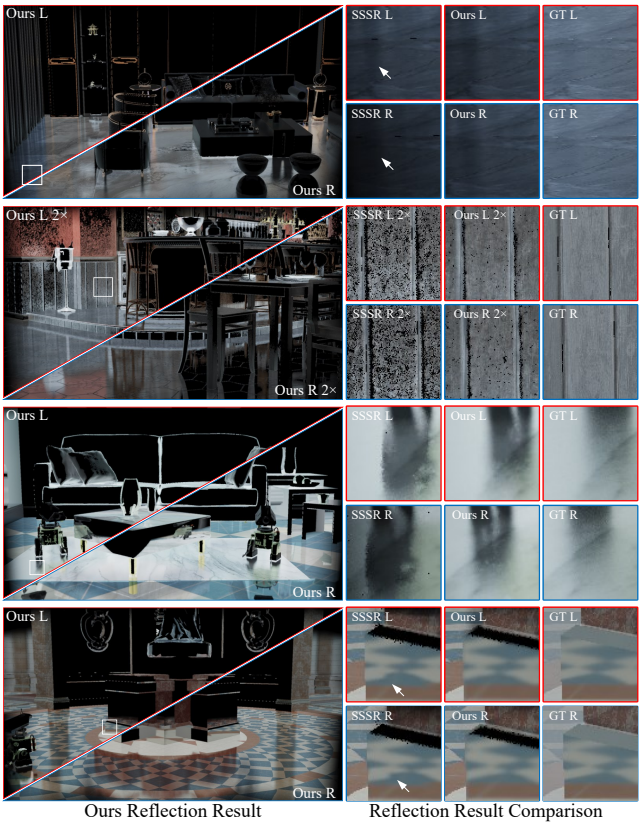
## 5. Result and Analysis

| Scene | $C_{tri}$ | $C_{dynamic}$ | $C_{mat}$ | $\alpha_{min\sim max}$ | $\alpha_{avg}$ |
|---|---|---|---|---|---|
| Pica | 383434 | 0 | 35 | 0.069∼0.600 | 0.421 |
| Room | 3388689 | 0 | 71 | 0.0∼0.634 | 0.390 |
| Wine | 1270923 | 0 | 75 | 0.0∼0.597 | 0.517 |
| Pink | 282260 | 2 | 46 | 0.054∼0.679 | 0.488 |
| Temple | 448935 | 1 | 54 | 0.036∼0.624 | 0.416 |

**Table 1:** *The parameters for test scenes. $C_{tri}$ is the number of triangles, $C_{dynamic}$ is the number of dynamic objects, $C_{mat}$ is the number of reflective materials (roughness $< 0.8$), $\alpha_{min\sim max}$ is the range of roughness, and $\alpha_{avg}$ is the average roughness.*

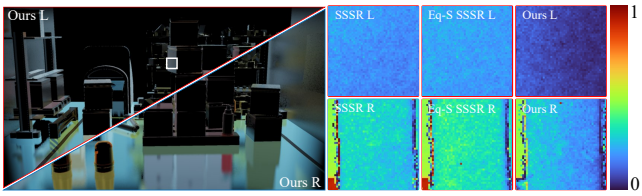
We evaluate our method on an Intel i9-12900K 16-core processor (3.2GHz) with 64GB RAM and a single NVIDIA GeForce RTX 3090 with 24GB video memory. We used Vulkan to implement both the baseline [Sta15] and our method, sampling one ray per pixel, and setting the spatial filter's kernel radius to 2. The ground truth was rendered by hardware ray tracing with a spp of 1024, taking 2-5 seconds; in order to compare with the *SSSR* results, only two bounces were used. In our cropped plots, we ensure that the left and right eyes are cropped to similar areas. We focus on the reflection effect, so the results do not introduce the shadow effect. All the images are rendered with 3840×1080 resolution.

We used five representative scenes to cover common issues in stereo rendering reflections: complex occlusions (*Pica*), blurry reflections and specular elongation (*Room*), reflections of moving objects and complex textures (*Temple* and *Pink*), and complex materials and geometry (*Wine*). The scene parameters are listed in Table 1.




Ours Reflection Result Reflection Result Comparison

**Figure 9:** *Subjective quality comparison on reflection. From top to bottom: Room Scene, Wine Scene, Pink Scene, Temple Scene.*




(a) Ours Reflection Result (b) RSE (L/R) With GT Comparison

**Figure 10:** *Equal sample comparison. We compute SSSR, equal-sample SSSR (Eq-S SSSR), and Ours with the RSE (Relative Squared Error) between GT in the cropped region. Despite increasing the sample number, SSSR cannot resolve inconsistent stereoscopic cues, and our results are more consistent and closer to the ground truth.*

### 5.1. Inconsistency Analysis

We compare our results with *SSSR* [Sta15] directly used in stereo rendering, and we also compare with the ground truth by ray tracing. We use *SSIM*, *LPIPS*, and *DISTS* for quantitative evaluation.

| Scene | | SSIM | | LPIPS ↓ | | DISTS ↓ | |
|---|---|---|---|---|---|---|---|
| | | SSSR | Ours | SSSR | Ours | SSSR | Ours |
| Pica | L | 0.373 | **0.423** | 0.259 | **0.233** | 0.194 | **0.194** |
| | R | 0.393 | **0.447** | 0.267 | **0.242** | **0.201** | 0.202 |
| Room | L | 0.364 | **0.393** | 0.249 | **0.228** | 0.154 | **0.149** |
| | R | 0.353 | **0.382** | 0.251 | **0.229** | 0.158 | **0.150** |
| Wine | L | 0.274 | **0.295** | 0.455 | **0.440** | 0.283 | **0.261** |
| | R | 0.268 | **0.296** | 0.461 | **0.441** | 0.293 | **0.262** |
| Pink | L | 0.455 | **0.477** | 0.412 | **0.409** | 0.233 | **0.231** |
| | R | 0.433 | **0.454** | 0.425 | **0.421** | 0.235 | **0.235** |
| Temple | L | 0.476 | **0.498** | 0.362 | **0.348** | 0.166 | **0.165** |
| | R | 0.481 | **0.505** | 0.356 | **0.345** | 0.171 | **0.167** |

**Table 2:** *Objective quality comparison on reflection. We calculated the perceptual metrics between SSSR/Ours and GT, and marked the best quality in* ***bold****.*

Figure 1 and Figure 9 show five scenes that contain direct illumination under an environment map and reflection. We also show the reflection results in cropped areas to compare with others. *SSSR* methods show significant inconsistency mistakes, while *Ours* obtains rendering results with better consistency, even for complex geometry. Table 2 shows that we can achieve better quality compared to *SSSR*. And results of *Ours* are closer to the ground truth for both views.

We also perform an equal-sample comparison in our results and sample-increased *SSSR*. To ensure the samples used in *SSSR* are equal to our method, we use two samples on the reflection areas for *SSSR*. Figure 10 shows that the inconsistent problem cannot be resolved by directly increasing the number of samples.

### 5.2. Performance Analysis

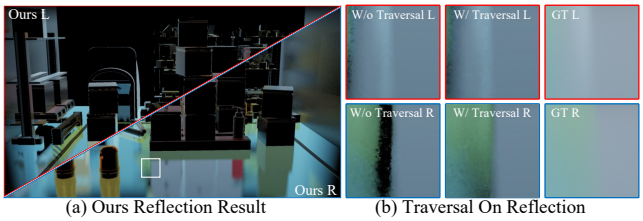
| Scene | PT | Traversal | | Spatial Filter | | Total | |
|---|---|---|---|---|---|---|---|
| | | SSSR | Ours | SSSR | Ours | SSSR | Ours |
| Pica | 3.94 | 2.06 | 2.72 | 0.75 | 1.03 | 2.81 | 3.75 (+33%) |
| Room | 4.92 | 1.92 | 2.65 | 0.94 | 1.04 | 2.86 | 3.69 (+29%) |
| Wine | 8.52 | 3.61 | 5.24 | 0.85 | 1.22 | 4.46 | 6.46 (+44%) |
| Pink | 3.34 | 2.26 | 3.48 | 0.86 | 1.03 | 3.12 | 4.51 (+45%) |
| Temple | 3.65 | 2.09 | 3.16 | 0.89 | 1.09 | 2.98 | 4.25 (+42%) |

**Table 3:** *Render time (in ms) of SSSR [Sta15], Ours, and PT . In the "Spatial Filter" process, SSSR includes the spatial filtering process; Ours includes finding "reprojection point pairs", reprojection, and spatial filtering. Path tracing (PT) is rendered at 2 spp and relies on the RT Core. All methods consider only reflection time.*

We analyze our method by averaging each phase's cost, estimated from 100 frames in three scenes. Table 3 compares the rendering time of our stereo-aware method and *SSSR* [Sta15]. For traversal time, our method increases the performance consumption by $0.7-1.5$ *ms* compared to *SSSR*, but we find more reliable reflection points. Our spatial reuse approach adds roughly $0.2-0.3$ *ms* to the rendering time, primarily due to the computation of reprojection point pairs. Overall, our approach adds only $0.9-2.0$ *ms* (just 2.0 *ms* for highly complex scenes) to the processing time, ensuring the algorithm's suitability for real-time scenarios.

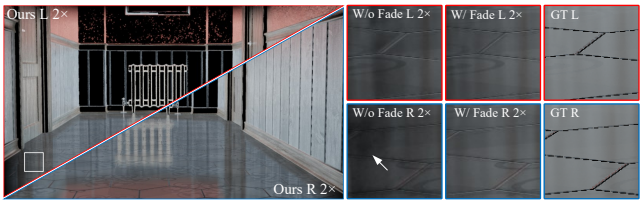
As shown in Table 3, *PT* has less overhead in simple scenes than our method, but as scenes get more complex, its cost exceeds our method. Moreover, *PT* relies on the RT Core, making it difficult to be widely used in stereo devices.

### 5.3. Step-Wise Analysis




(a) Ours Reflection Result (b) Traversal On Reflection

**Figure 11:** *Stereo-Traversal Effect. (a) We cropped out the part of the image with an inconsistency in screen space geometry. (b) At locations where the reflection information is obscured, our method correctly complements the missing information of the original SSSR.*




(a) Ours Reflection Result (b) Fade On Reflection

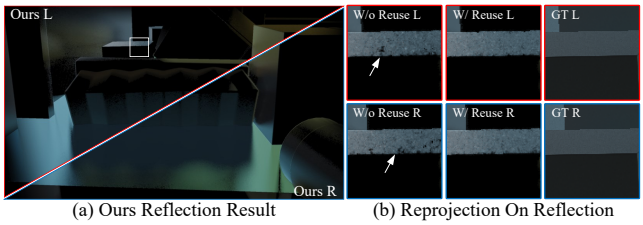
**Figure 12:** *Stereo-Aware Fade Effect. (a) We cropped out the part of the image with an inconsistent fading boundary. (b) At the boundary of the view frustum, our method maintains the consistency of the reflection results.*

We will analyze step-by-step to show where our method mitigates the inconsistent cues of *SSSR* applied to stereo rendering.

**Stereo-Aware Traversal Effect.** We evaluate the traversal effect in *PICA*, which has a more complex occlusion relationship. As shown in Figure 11, our method can complement the reflection information that would otherwise be missing from a single screen space.

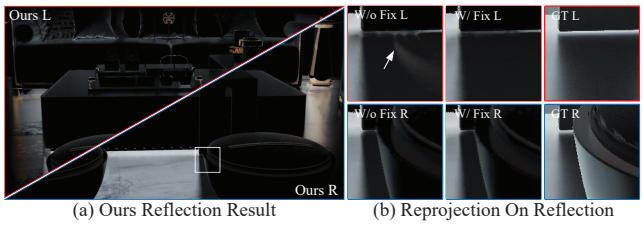
**Stereo-Aware Fade Effect.** We evaluate the fade effect in *Wine*, where the left and right eyes are very different at the frustum boundary. As shown in Figure 12, the *SSSR* method considered the frustum boundary for each eye, respectively, and some reflections incorrectly faded out, leading to visual discomfort. We can see that the reflection result of the left eye is bright, while the right eye is dark. By combining all the boundary information of the stereo, we can obtain a more reasonable fade result.

**Glossy Reprojection Effect.** To verify the effect of reprojection,




(a) Ours Reflection Result (b) Reprojection On Reflection

**Figure 13:** *Stereo-Aware Reprojection Effect. (a) We cropped out the part of the image with inconsistent reflection noise. (b) At locations with a large disparity between the noise levels of the left and right eyes, reprojection eliminates the inconsistent noise cues.*




(a) Ours Reflection Result (b) Reprojection On Reflection

**Figure 14:** *Incorrect Reuse Fix Effect. (a) We cropped out the part of the image that was reused incorrectly. (b) Our "point pairs" detection prevents incorrect reuse at locations with incorrect reprojection artifacts.*

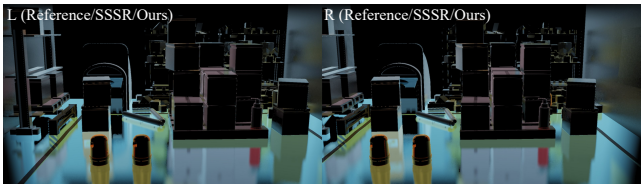



**Figure 16:** *Studies for inconsistent cues and module ablation. The rendered results of the test methods for the left and right eyes are placed side by side, allowing users to perceive their consistency in stereo mode. In each pair, the rendering method is the same for both views.*

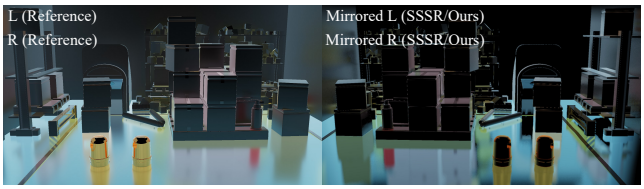
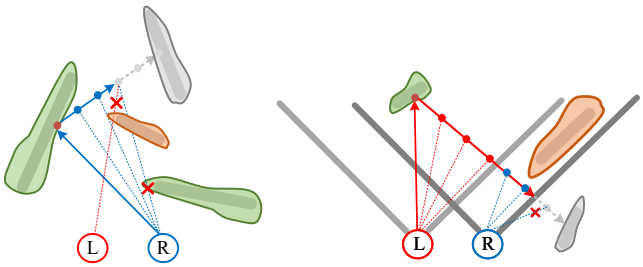



**Figure 17:** *Study for visual quality. The reference image is placed on the left, and the mirrored image (generated by either SSSR or Ours) is placed on the right. Users assess visual quality by switching between Ours and SSSR in 2D mode. In each pair, both sides are rendered by the same viewport but using different methods (with the right image mirrored).*

we turned off the reprojection. We evaluate the reprojection effect in *PICA*. As shown in Figure 13, reprojection mitigates the inconsistent cues brought by noise, and this inconsistency is especially noticeable when geometries are more complex. Our stereo-aware reprojection yields stable and consistent results across consecutive frames (demonstrated in the attached video).

Figure 14 shows the results of our method after fixing incorrect reuse in *Room*; compared to the results without reuse correction, we can better reduce the inconsistent information.

### 5.4. Failure Case




(a) (b)

**Figure 15:** *Failure case. (Left) During ray walking, the depth of the ray's current position is occluded in both eyes. (Right) Reflection ray travels beyond the boundaries of left and right eyes, unable to get available geometry information.*

In common with other screen space algorithms, our method fails when information outside of screen space is required. Figure 15 shows that it will fail if the reflection ray is not visible in the screen space of both views. In other words, our method utilizes the stereo screen space geometric information as much as possible, making them consistent but unable to break screen space limitations.

Our method utilizes the front and back depth of instances. If an instance contains multiple separated objects, the instance depth may not align with the objects, making intersection checks during traversal difficult.

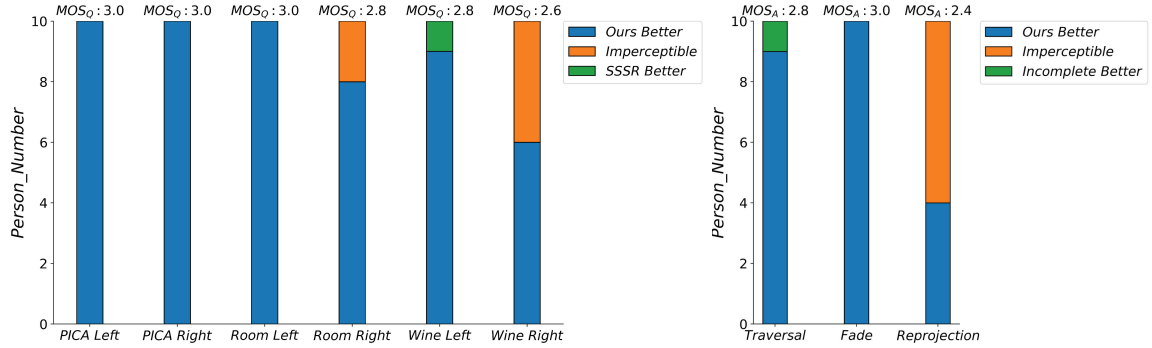
Moreover, we do not deal with the original issues in SSR, such as camera-faced reflection. A multi-view approach can solve this problem by considering missing reflections in SSR, and our method can be easily extended to such approaches.

## 6. User Study

We conduct a user study on a PICO 4 Enterprise virtual reality headset. It offers a high resolution of $2160 \times 2160$ pixels per eye, a refresh rate of 90 Hz, and a field of view (FOV) of 105°. Other parameters are set to the device's default values. The scene is rendered at a resolution of 1920×1080 for each eye, with a constant interpupillary distance (IPD). All study data were generated using the same configuration and hardware environment in Sec. 5.

### 6.1. Study Methods

This user study aims to verify whether our method can mitigate the inconsistent cues introduced by *SSSR* in stereo rendering, thus




(a) Result For Visual Quality Study (b) Result For Module Ablation Study

**Figure 18:** *Study results for visual quality and module ablation. The distribution of user choices and the Mean Opinion Score ($MOS_Q$ and $MOS_A$) for each condition. (a) Users perceive the visual quality of rendering methods Ours and SSSR. (b) Users perceive the visual consistency between Ours and Incomplete (lacking either the Traversal, Fade, or Reprojection module).*

reducing the user's visual discomfort. In addition, we further verify whether our method can obtain better visual quality. Inspired by Mišiak et al. [MFL23] and Kaernbach et al. [Kae01], we set up three studies and rating options.

**First Study (For Inconsistent Cues) Setting.** As shown in Figure 16, we use side-by-side (SBS) stereo images to assess consistency in different rendering methods (*SSSR*, *Ours*, *RayTracing*). Each SBS pair employs the same rendering method for both left and right eyes. The rating options are: (1) very strong inconsistent, (2) strong inconsistent, (3) moderate inconsistent, (4) slight inconsistent, (5) no perceivable inconsistent. Each stereo image is presented in stereo mode for 20 seconds to allow users to perceive consistency. We use a blank frame to reset the user's attention when toggle methods.

**Second Study (For Visual Quality) Setting.** As shown in Figure 17, we provide a pair of reference and mirrored test images with the same viewport in SBS format for users to perceive visual quality. The left side remains the reference image, and the right side shows the mirrored result of *Ours* (or *SSSR*). The rating options are: (1) *SSSR* is better, (2) Imperceptible, (3) *Ours* is better. Since users are unaware of the method applied to the images, they choose the one that resembles the reference more in each comparison. During data processing, these choices are manually mapped (one-to-one mapping) to the rating options. Within 20 seconds, we toggle between image pairs (*Ours* and *SSSR*) in 2D mode four times without a blank frame, helping users perceive which image in each comparison has better quality.

**Third Study (For Module Ablation) Setting.** We conducted ablation experiments in *PICA*, investigating the impact of modules (*Traversal*, *Fade*, and *Reprojection*) on visual perception. Users were asked to compare the consistency between *Ours* and *Incomplete* in stereo mode (Figure 16), where *Incomplete* is based on *Ours* but lacking one of the modules. The rating options are: (1) *Incomplete* is better, (2) Imperceptible, (3) *Ours* is better. Like the second study, we toggle between image pairs (*Ours* and *Incomplete*) within 20 seconds four times without a blank frame and then map users' choices (one-to-one mapping) to the rating options.

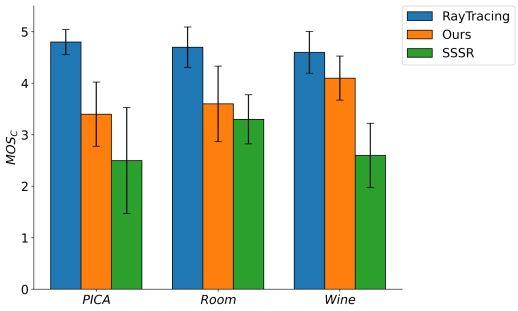



**Figure 19:** *Result of study for inconsistent cues. Mean Opinion Scores with 90% CI (confidence interval) about consistency scores, which were collected for three algorithms: SSSR, Ours, and RayTracing.*

**Conditions.** In the first study, we rendered in *PICA*, *Room*, *Wine*, using *SSSR*, *Ours*, and *RayTracing*, resulting in $3 \times 3 = 9$ conditions. In the second study, we compared visual quality across two eyes in the three scenes, yielding $3 \times 2 = 6$ conditions. In the third study, we conducted ablation experiments on three modules in *PICA*, resulting in $1 \times 3 = 3$ conditions.

**Participants.** Ten individuals (8M, 2F) were recruited to participate. All participants had normal or corrected-to-normal vision, self-reported stereoscopic vision, and were naive to the study's goals. The participants were briefed to look for inconsistent cues and perceive the change in visual quality, but no specifics were given regarding the type of render algorithm they encountered.

### 6.2. Study Results

The collected inconsistent cues, visual quality, and module ablation ratings $(C, Q, A)$ are averaged into a *MOS* (Mean Opinion Score) for each condition. In the first study, we computed the average

scores for three rendering methods across three scenes and provided their corresponding confidence intervals. In the second and third studies, we analyzed the distribution of user choices and presented the *MOS* for each condition.

**Inconsistent Cues.** In the *PICA*, the FOV (field of view) center has significantly inconsistent screen-space geometric information, leading to noticeable visual discomfort for users. In the *Room*, the inconsistent fading boundary becomes the primary problem. Although users show increased tolerance for boundary artifacts, it still diminishes their overall perception. In the *Wine*, due to the complexity of occlusion relationships, SSSR faces inconsistent reflection samples, leading to significant confusion for users in interpreting the scene information. Figure 19 shows our method effectively mitigates visual discomfort from inconsistent cues. The Wilcoxon test ($z = -2.870, p = 0.004 < 0.01$) indicates a significant reduction in visual discomfort.

**Visual Quality.** As shown in Figure 18 (a), our method received better user visual quality ratings ($MOS_Q > 2.0$) in all conditions. We observe slightly lower ratings in the *Wine* compared to the other two scenes. This is due to excessive information loss in complex scenes with screen-space methods, resulting in unresolved artifacts that affect user judgment.

**Module Ablation.** As shown in Figure 18 (b), each module in our method improves user visual consistency ($MOS_A > 2.0$). *Traversal* and *Fade* considered stereo geometry and boundary information, significantly enhancing the user's visual experience. During the study, it was found that users were not particularly sensitive to inconsistent sample cues. However, *Reprojection* utilizes stereoscopic spatial information to produce more stable and consistent stereo results (see the attached video for further details).

## 7. Conclusions And Future Work

In this paper, we focus on alleviating the confusion of non-physical and vision-unacceptable illumination resulting from information inconsistency between the left and right eyes. We analyzed the results with issues, identifying three inconsistent cues: inconsistent screen space geometry, inconsistent fading boundary, and inconsistent reflection noise. To address these inconsistencies, we adopt several stereo-aware methods. Specifically, we fully utilize the geometric information provided by binocular vision, ensuring ray traversal on consistent scene information for both views. Simultaneously, we heuristically optimize the fade method, considering the edges of both eyes' screens to ensure uniform fading. Additionally, we mitigate visual discomfort caused by excessive noise due to low sampling rates by reusing spatial domain information between the left and right eyes. Finally, we conducted user studies on head-mounted devices, confirming that our method enhances users' visual perception in stereo rendering. Our approach requires only stereoscopic G-Buffer and Hi-Z information as additional information, which is easily accessible in game engines such as Unreal Engine (UE) and Unity.

In the future, we would like to consider temporal reuse to improve stereo visions. Additionally, it would also be interesting to apply our method with foveated rendering to reduce the overall computational overhead. Developing other screen-space rendering algorithms into stereo versions could also be a promising direction.

**Acknowledgments.** We thank the reviewers for their valuable comments. This work has been partially supported by the National Key R&D Program of China under grant No. 2022YFB3303200 and the National Natural Science Foundation of China under grant No.62272275.

## References

[AW*87] AMANATIDES J., WOO A., ET AL.: A fast voxel traversal algorithm for ray tracing. In *Eurographics* (1987), vol. 87, Citeseer, pp. 3–10. 2

[Beu20] BEUG A. P.: *Screen Space Reflection Techniques*. PhD thesis, Faculty of Graduate Studies and Research, University of Regina, 2020. 2

[HRDB16] HEDMAN P., RITSCHEL T., DRETTAKIS G., BROSTOW G.: Scalable inside-out image-based rendering. *ACM Transactions on Graphics (TOG) 35*, 6 (2016), 1–11. 2

[HSAS19] HIRVONEN A., SEPPÄLÄ A., AIZENSHTEIN M., SMAL N.: Accurate real-time specular reflections with radiance caching. *Ray Tracing Gems: High-Quality and Real-Time Rendering with DXR and Other APIs* (2019), 571–607. 2

[HWB19] HE D., WANG R., BAO H.: Real-time rendering of stereo-consistent contours. In *2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)* (2019), IEEE, pp. 81–87. 2

[IdRV00] IJSSELSTEIJN W. A., DE RIDDER H., VLIEGEN J.: Effects of stereoscopic filming parameters and display duration on the subjective assessment of eye strain. In *Stereoscopic Displays and Virtual Reality Systems VII* (2000), vol. 3957, SPIE, pp. 12–22. 3

[Kae01] KAERNBACH C.: Adaptive threshold estimation with unforced-choice tasks. *Perception & Psychophysics 63* (2001), 1377–1388. 9

[KT04] KOOI F. L., TOET A.: Visual comfort of binocular and 3d displays. *Displays 25*, 2-3 (2004), 99–108. 1

[MFL23] MIŠIAK M., FUHRMANN A., LATOSCHIK M. E.: The impact of reflection approximations on visual quality in virtual reality. In *ACM Symposium on Applied Perception 2023* (2023), pp. 1–11. 9

[MKKJ19] MÄKITALO. M., KIVI. P., KOSKELA. M., JÄÄSKELÄINEN. P.: Reducing computational complexity of real-time stereoscopic ray tracing with spatiotemporal sample reprojection. In *Proceedings of the 14th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2019) - GRAPP* (2019), INSTICC, SciTePress, pp. 367–374. doi:10.5220/0007692103670374. 2

[MM14] MCGUIRE M., MARA M.: Efficient gpu screen-space ray tracing. *Journal of Computer Graphics Techniques (JCGT) 3*, 4 (2014), 73–85. 2

[MMBJ17] MARA M., MCGUIRE M., BITTERLI B., JAROSZ W.: An efficient denoising algorithm for global illumination. *High Performance Graphics 10* (2017), 3105762–3105774. 2, 5

[NAK12] NORTHAM L., ASENTE P., KAPLAN C. S.: Consistent stylization and painterly rendering of stereoscopic 3d images. In *Proceedings of the Symposium on Non-Photorealistic Animation and Rendering* (2012), Citeseer, pp. 47–56. 2

[NSL*07] NEHAB D., SANDER P. V., LAWRENCE J., TATARCHUK N., ISIDORO J. R.: Accelerating real-time shading with reverse reprojection caching. In *Graphics hardware* (2007), vol. 41, pp. 61–62. 2

[Pfa01] PFAUTZ J. D.: Sampling artifacts in perspective and stereo displays. In *Stereoscopic Displays and Virtual Reality Systems VIII* (2001), vol. 4297, SPIE, pp. 54–62. 3

[PFJ23] PHILIPPI H., FRISVAD J. R., JENSEN H. W.: Practical temporal and stereoscopic filtering for real-time ray tracing. 2

[SBE22] SHI P., BILLETER M., EISEMANN E.: Stereo-consistent screen-space ambient occlusion. *Proceedings of the ACM on Computer Graphics and Interactive Techniques 5*, 1 (2022), 1–12. 2

[Sie99] SIEGEL M.: Just enough reality: A kinder gentler approach to stereo. In *Cockpit Displays VI: Displays for Defense Applications* (1999), vol. 3690, SPIE, pp. 173–179. 3

[SKS11] SOUSA T., KASYAN N., SCHULZ N.: Secrets of cryengine 3 graphics technology. In *ACM SIGGRAPH* (2011), vol. 1. 2

[Sta15] STACHOWIAK T.: Advances in real time rendering, part i. In *ACM SIGGRAPH 2015 Courses* (New York, NY, USA, 2015), SIGGRAPH '15, Association for Computing Machinery. URL: https://doi.org/10.1145/2776880.2787701, doi:10.1145/2776880.2787701. 2, 3, 4, 6, 7

[Ulu18] ULUDAG Y.: Hi-z screen-space cone-traced reflections. In *GPU Pro 360 Guide to Lighting*. AK Peters/CRC Press, 2018, pp. 237–280. 1, 2

[WDP99] WALTER B., DRETTAKIS G., PARKER S.: Interactive rendering using the render cache. In *Rendering Techniques' 99: Proceedings of the Eurographics Workshop in Granada, Spain, June 21–23, 1999 10* (1999), Springer, pp. 19–30. 2

[WHP*01] WATKINS W. R., HEATH G. D., PHILLIPS M. D., VALETON M., TOET A.: Search and target acquisition: single line of sight versus wide baseline stereo. *Optical Engineering 40*, 9 (2001), 1914–1927. 2

[WMFL20] WISSMANN N., MIŠIAK M., FUHRMANN A., LATOSCHIK M. E.: Accelerated stereo rendering with hybrid reprojection-based rasterization and adaptive ray-tracing. In *2020 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)* (2020), IEEE, pp. 828–835. 2

[WPS*15] WIDMER S., PAJĄK D., SCHULZ A., PULLI K., KAUTZ J., GOESELE M., LUEBKE D.: An adaptive acceleration structure for screen-space ray tracing. In *Proceedings of the 7th Conference on High-Performance Graphics* (2015), pp. 67–76. 2

[Wym05] WYMAN C.: Interactive image-space refraction of nearby geometry. In *Proceedings of the 3rd international conference on Computer graphics and interactive techniques in Australasia and South East Asia* (2005), pp. 205–211. 3

[YLS20] YANG L., LIU S., SALVI M.: A survey of temporal antialiasing techniques. In *Computer graphics forum* (2020), vol. 39, Wiley Online Library, pp. 607–621. 2, 5

[ZK07] ZITNICK C. L., KANG S. B.: Stereo for image-based rendering using image over-segmentation. *International Journal of Computer Vision 75*, 1 (2007), 49–65. 2

[ZLY*21] ZENG Z., LIU S., YANG J., WANG L., YAN L.-Q.: Temporally reliable motion vectors for real-time ray tracing. In *Computer Graphics Forum* (2021), vol. 40, Wiley Online Library, pp. 79–90. 2, 5