Foveated Instant Radiosity

Category: Research

Paper Type: algorithm/technique

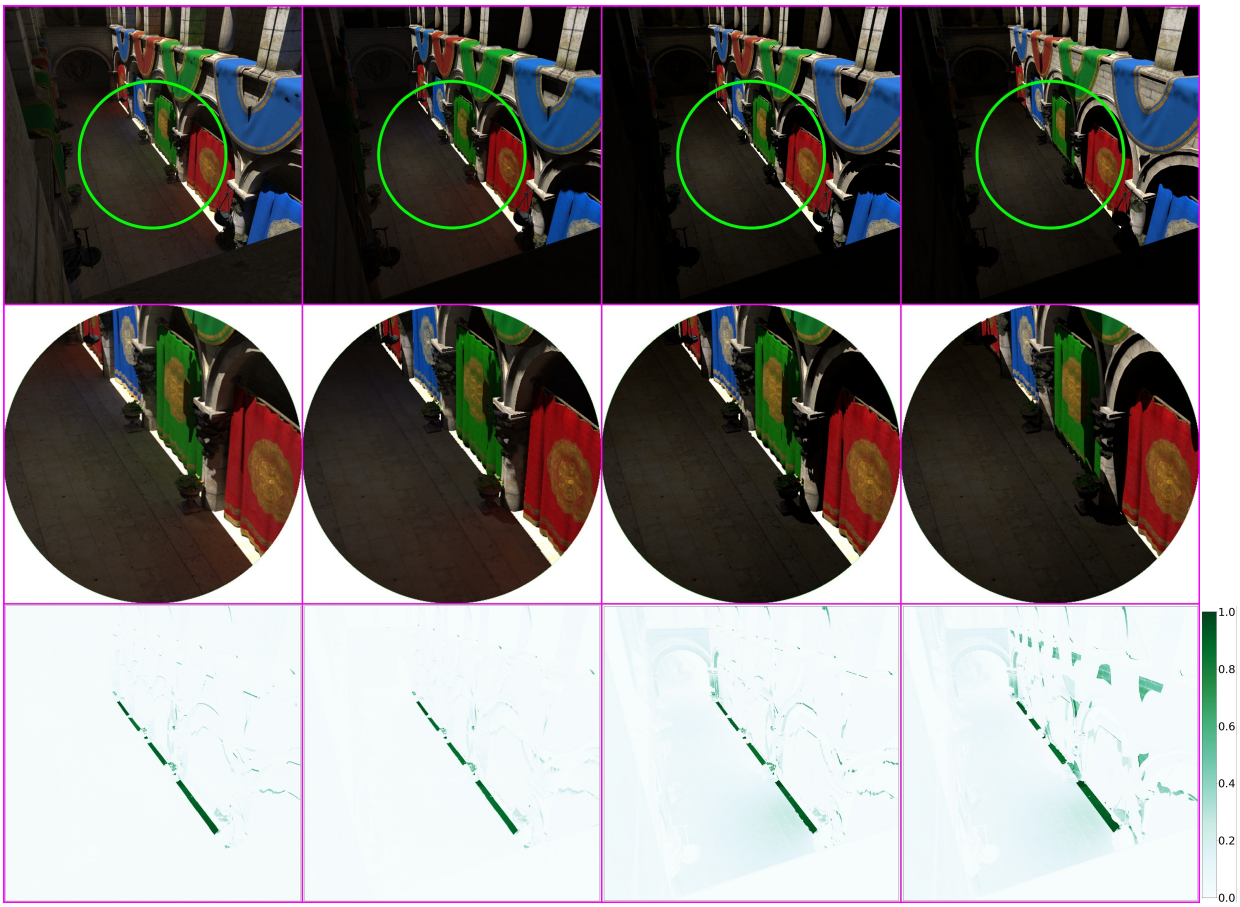

Fig. 1: Row 1: Global illumination rendered with path tracing (column 1), with our method (column 2), instant radiosity with Virtual Point Light (VPL) shadow maps of 1024×1024 (column 3), and instant radiosity with VPL shadow maps of 128×128 (column 4) for Sponza. Row 2: Magnified (2×) foveated regions are shown. Row 3: Temporal errors between adjacent frames with a moving light source.

Abstract— Visual acuity based foveated rendering is widely used to accelerate 3D graphics rendering. The traditional forward projection rendering and ray tracing methods can be adapted into the foveated rendering framework in a quite straightforward way for multiple spatial resolutions. While radiosity rendering is an object space method, it can not be adapted into the foveated rendering pipeline directly. In this paper, we propose a foveated rendering method for instant radiosity based on the idea of multiple resolutions of illumination. We introduce the foveated importance of virtual point lights (VPLs) to measure the illumination contribution for the foveated regions. The distribution of VPLs is generated and optimized based on the foveated importance. In order to maintain the stability between adjacent frames, we select the valid VPLs of the last frame based on the changes of their foveated importance, and reuse them to avoid the flickers between the consecutive frames. At last, we use interleaved rendering method to render indirect illumination with all valid VPLs. Our method supports dynamic scenes. We tested our method with several scenes, demonstrating a comparable quality of the foveated region to path tracing, better quality to instant radiosity, a 4 orders of magnitude performance gain compared to the path tracing method, and a 50× speedup as compared to traditional instant radiosity method.

Index Terms— foveated rendering, global illumination, instant radiosity, temporal coherence

1 Introduction

Foveated rendering [10] provides different rendering qualities for users in foveated regions and peripheral regions according to the human visual system model [35], aiming at improving the time performance of rendering. The traditional forward projection and ray tracing methods can be adapted into foveated rendering directly because they can render scenes with multiple spatial resolution easily. For example, Guenter et al. [10] uses foveated graphics to generate 3 layers of different resolutions image using traditional forward projection, and composite them to generate output images. For ray tracing, a gaze-contingent method [8] was designed to shoot different numbers of rays for pixels

in different regions.

However, when it comes to global illumination, the traditional forward projection methods become inefficient. In computer graphics, radiosity [40] is an important method to render global illumination for scenes with diffuse surfaces, and the render results are natural and noiseless for diffuse environments. But radiosity is only used to render static scenes due to heavy computation of form factors. Instant radiosity [18] (IR) is a global illumination method for dynamic scenes. It uses Virtual Point Lights (VPLs) to simulate light transport efficiently. It works well for diffuse scenes, but the computation is in object space. Moreover, it remains unknown how to apply IR to multiple spatial resolutions in the context of foveated rendering.

In this paper, we propose a foveated instant radiosity method to render global illumination for scenes with diffuse surfaces. Inspired by Yee's method [46], which accelerated the global illumination computation by taking advantage of the human visual system, our method aims to provide different rendering qualities at different regions according to the human visual system, which is based on the idea of multiple resolution of illumination, i.e. our method computes the high-quality illumination in the foveated region with high performance at the cost of low-quality illumination in the peripheral region. Our method supports two-bounce indirect illumination at interactive frame rates with the current hardware and is capable to handle scenes with dynamic objects.

Our foveated instant radiosity method introduces a new multiple illumination rendering pipeline to compute indirect illumination. There are 4 steps in our method. First, we generate VPL candidates on the surfaces that are directly visible from the foveated regions. Second, we compute the foveated importance of VPLs to measure the illumination contribution of each VPL for the foveated region. Third, we generate the valid VPLs for the current frame by selecting from the new VPL candidates and the old valid VPLs of the last frame according to their contributions. At last, we compute radiosity for the valid VPLs and render the final image with an interleaved sampling method. Our method is based on scene voxelization. Since dynamic objects can be integrated into the voxelization easily, our method can support rendering indirect illumination of scenes containing the objects with rigid body movement and the deformable objects interactively.

Our method produces quality illumination at interactive rates(Figure 1). Compare to path tracing(column 1), the average pixel error of the foveated region (marked by the green circle) of our method is 0.036(column 2), the error of instant radiosity with VPL shadow maps of 1024×1024 pixels is 0.055(column 3), and the error of instant radiosity with VPL shadow maps of 128×128 pixels is 0.081(column 4). In the magnified images in row 2, with our method, the color bleeding on the floor caused by the red and green curtains of Sponza is more obvious than that of instant radiosity, which is more similar to the results of path tracing. Our method also achieves similar average temporal error as path tracing and better temporal stability than instant radiosity. The performance of our method is 56ms, which corresponds to a speedup of 38,923$\times$ versus path tracing and 184$\times$ versus instant radiosity with VPL shadow maps of 1024×1024 pixels and 166$\times$ versus instant radiosity with VPL shadow maps of 128×128 pixels. Images in row 3 visualize the pixel temporal errors between two consecutive frames with slight light movement.

In summary, the contributions of our method are as follows:

1. A foveated instant radiosity method for dynamic scenes, which computes high-resolution illumination in the foveated region and low-resolution illumination in the peripheral region;

2. A foveated importance computation method to evaluate how much illumination a VPL candidate contributes to the foveated region;

3. A high performance VPL management algorithm to optimize the VPL distribution and select the old valid VPLs of the last frames based on the foveated importance, which can help us generate high-quality illumination in the foveated region and keep the temporal stability for adjacent frames.

2 Related Work

A considerable amount of previous research has been focusing on foveated rendering. In this section, we first discuss the early work related to the foveated rendering, which is usually referenced as gaze-contingent rendering. Then we introduce the prior work for 3D foveated rendering in recent years. After this, we discuss the previous work for radiosity rendering upon which our foveated instant radiosity method is based. Finally, we briefly overview some temporal coherence based rendering methods, which are related to the temporal stability of our method.

2.1 Gaze-contingent Rendering

Previous research on gaze-contingent rendering (foveated rendering) attempted to balance the amount of information displayed against the visual information processing capacity of the observer through real-time eye movement sensing. First, gaze-contingent rendering was proposed to render volume data [22]. Then, gaze-directed adaptive rendering was introduced to render polygon geometry scenes with gaze detection [31]. An imperceptible geometry simplification method was proposed to accelerate rendering for polygon geometry scenes based on a hierarchy of local simplification operations [24]. Murphy et al. [29] proposed a non-isotropic model-based method to generate and evaluate the level of detail rendering. We suggest readers to read several excellent surveys that provide a comprehensive and penetrating review on this research direction [35] [6] [5].

Another research direction related to gaze-contingent rendering is gaze-dependent Depth-of-field (DOF) rendering. The regular pipeline was to render all in focus with full resolution, and then convolved with a filter, such Gaussian, circle, Bokeh or Possion filter, to generate low-resolution image for the regions that were not in the focus range. Mipmapping, multi-texture, and fragment programming were often used to improve the performance of rendering. Hillair et al. [15] proposed a method to render the depth-of-field blur effect based on the focus point detected. Based on this method, Mantiuk et al. [25] used a circle of confusion (CoC) as a blur factor to control the blurriness for the geometries at different depth in rendering. Gupta et al. [11] proposed a gaze-contingent depth of field display pipeline for light field data. A user study [26] was designed to evaluate how effectively people could use gaze-contingent depth of field rendering to perceive depth. Duchowski et al. [7] tested gaze-contingent DOF for the reduction of visual discomfort when viewing stereoscopic displays. Gaze-contingent DOF [44] on stereoscopic displays was compared to the same effect on non-stereoscopic 3D in terms of depth perception, image quality, and viewing comfort.

2.2 Foveated 3D Rendering

Foveated 3D Rendering was proposed for the applications that have interactions in real-time in [10], as HMDs are widely used in Virtual Reality(VR) in recent years, which requires a high-quality stereoscopic display with no latency. The basic idea of foveated 3D rendering is to render a high-quality image in the foveated region, and render a low-quality image in the peripheral region to save time.

One challenge of 3D foveated rendering is to process the geometry based on the attributes of the human visual system. The idea of the level of detail technique is used to keep the high-quality geometry in the sensitive region of human visual perception and simplify the objects or part of objects in the insensitive area. Luebke et al. used perceptual metrics derived from the contrast sensitivity function (CSF) to guide local simplification operations [24], and the result from the simplified model is indistinguishable from the original one. Murphy et al. [29] calculated the visual angle between the gaze direction and the coarsest geometry of the object, and used an acuity-based resolution degradation function to select triangles for further subdivision. Parkhurst et al. [32] designed a user study to evaluate gaze-contingent level of detail rendering of virtual environment when a user was required to complete a visual search task. gaze-contingent DOF tessellation was proposed to combine tessellation and gaze-contingent DOF [23]. In this method,

the pop artifacts in the transition of the level of detail can be hidden in the transition from blurry to sharp of gaze-contingent DOF rendering.

The most important and difficult problem of foveated rendering is to generate images with different quality for different regions. Multiple spatial resolution idea was used in foveated rendering. Patney et al. [33] proposed a contrast enhancement method to recover the image details and a multi-resolution and saccade-aware temporal antialiasing algorithm to address aliasing in the peripheral region. Adaptive Image-Space Sampling was introduced in [42]. Besides acuity, more visual cues in the perception model, such as eye motion, texture contrast, silhouettes, and highlight, were considered to generate the sampling pattern for sparse shading. A general rendering pipeline was introduced to shade the pixels of multi-rate in one output image adaptively [13], which could be used to render foveated image with forward projection more efficiently. Swafford et al. [43] designed a user study and used an HDR-VDP2 based metric to evaluate the foveated rendering methods implemented with four techniques: quality degradation of resolution, screen-space ambient occlusion (SSAO), tessellation, and ray-casting steps with visual eccentricity. Ray tracing methods can be adapted into the foveated rendering framework directly. A gaze-dependent sampling technique was proposed to control the number of rays traced to the different regions of the output image [25]. In order to further accelerate foveated rendering, a precomputed Poisson-disc sampling sets were introduced to guide ray tracer to shoot rays, and a scattered interpolation method was proposed to fill the blank spaces of sparse shading [28]. A kernel log-polar mapping algorithm was designed for 3D graphics [27], which provided a framework with controlled trade-off of visual quality and time cost for foveated rendering.

Besides the multiple spatial resolution idea, multi-resolution temporary, color and illumination ideas can also be used to accelerate the foveated rendering. Duchowski et al. analyzed the characters of color vision in the fovea and periphery and proposed a level of color details framework [4], which constructed color degradation mapping for gaze-contingent display. Yee et al. introduced an aleph map in [46], which represented spatiotemporal error tolerance for dynamic scenes. The same idea can be adapted to the foveated rendering easily.

Our method is a foveated rendering method based multi-resolution illumination idea. More VPLs are used to compute the high-quality indirect lighting in the foveated region, and less VPLs are used to illuminate the periphery region.

2.3 Instant Radiosity

The radiosity method can achieve high-quality illumination effect for scenes with diffuse surfaces. But it is very slow for complex scenes due to long computation time of form factors. Instant radiosity method is an interactive method for efficiently estimating global illumination for 3D scenes with diffuse objects. It was first introduced in [18]. In instant radiosity method, it's important to distribute VPLs into the visually important regions of the scene.

Jensen et al. [17] distributed VPLs in scenes with the random walk procedure that was used to distribute photons in photon mapping. The idea of generating VPLs by tracing paths from the camera appeared in [38] under the name bidirectional instant radiosity. The metropolis instant radiosity algorithm proposed by Segovia et al., replaced the standard VPL tracing algorithm based on independent random walks by a Metropolis-Hastings sampler for generating VPLsin [39]. Davidovic et al. [3] referred to the VPLs generated from the camera as local VPLs, as opposed to global VPLs, which were generated by tracing paths from the light sources. Georgiev et al. [9] proposed a simple approach to improve VPL distribution by rejecting VPLs that do not significantly contribute to the image. For sampling VPLs, instant radiosity emitted photons from the light sources and placed the VPL on the surfaces where the photons locate. This may cause problems for large scenes, as it doesn't concentrate VPLs in the region where the VPLs would produce direct lighting onto the output image. Segovia et al. [38] solved this problem by resampling the VPLs from a larger set of candidates, and they also placed VPLs using a variant of metropolis light transport [39]. Simon et al. [41] sampled VPLs from the product of two complementary photon maps. Ignoring the visibility between pixel samples and VPLs, Ritschel et al. [36] presented a method to fast resample VPLs to approximately match the amount of light they bring to the image.

Different from the forward projection methods and ray tracing methods, radiosity methods can not be integrated into the foveated rendering framework easily. Our method adapts instant radiosity method with foveated rendering technique to render high-quality global illumination for dynamic scenes in the foveated region.

2.4 Temporal Coherence Based Rendering

Temporal Coherence based rendering can reduce the flickers between the frames and accelerate for interactive rendering. Wald et al. [45] enforced temporal coherence by fixing the random number sequence used to generate the VPLs. In the context of VPL-based rendering, Aine et al. [21] achieved temporally stable indirect illumination by only moving a few VPLs each frame. Light clustering allows a smart selection of VPLs for more temporally stable indirect illumination [19]. Hăsan et al. [12] grouped point lights into clusters and reused the shaded results from the clusters over multiple frames in an animated video. Multiresolution splatting reduces overdraw by rendering illumination at varying frequencies [30]. Knecht et al. [20] improved the stability of the indirect illumination with temporal reprojection filtering. Prutkin et al. [34] and Bařak et al. [1] devised the methods to improve the temporal stability of the VPL sampling.

Our foveated instant radiosity method also uses temporal coherence in VPL management step, which reduces temporal errors between the adjacent frames and the number of VPLs' shadow maps that need to update.

3 Foveated Instant Radiosity

The core of our approach is a foveated based rendering for instant radiosity. The VPLs' distribution is based on the foveated region of the user. In order to maintain a temporally coherent VPL distribution from frame to frame, our idea is to replace VPLs gradually according to their contribution to the output image is used in our method. Given a scene and a viewpoint, the output image with illumination is rendered using Algorithm 1.

Algorithm 1 Foveated Instant Radiosity Method

Input: 3D scene S, the output viewpoint vp, foveated center $foveatedC$, foveated radius $foveatedR$
Output: $Framebuf$
▷ $Initialization$
1: $V_s \leftarrow VoxelizeStaticScene(S)$
2: $VPLs_p, Imp_p \leftarrow InitializeVPLs()$
3: **for** *each frame* **do**
4: $V_d \leftarrow VoxelizeDynamicObjects()$
▷ VPL candidateds generation
5: $VPLs_c \leftarrow GenerateVPLCandidates(V_s, V_d)$
▷ Foveated Importance Computation
6: $FoveatedWeight \leftarrow computeFoveatedWeight(V_s, V_d, foveatedC, foveatedR, vp)$
7: $Imp_c \leftarrow computeFoveatedImportance(VPLs_c, FoveatedWeight)$
▷ VPLs Management
8: $VPLs_e, Imp_e \leftarrow ManageVPL(VPLs_c, Imp_c, VPLs_p, Imp_p, FoveatedWeight)$
▷ Radiosity Rendering
9: $RenderScene(VPLs_e, V_s, V_d, S, vp)$
10: $VPLs_p \leftarrow VPLs_e$
11: $Imp_p \leftarrow Imp_e$
12: **end for**

There are five main steps of our algorithm. 1. **Initialization**. In this step, the static part of the scene is voxelized [37] (line 1) and the valid VPL list of the last frame ($VPLs_p$) is initialized as empty in the preprocess (line 2). Then, the dynamic part of the scene is voxelized for each frame (line 4). 2. **VPL candidates generation**. Once we have the voxelized representation of the scene, VPL candidates ($VPLs_c$)

are then generated based on the location of the foveated region (line 5). 3. **Foveated importance computation**. We calculate the foveated importance of the VPL candidates based on the foveated weight for each visible voxel of the scene (line 6-7). 4.**VPL management**. After this, the VPL management is used to select the valid VPLs from the current frame ($VPLs_e$). In this step, the geometry attributes of the valid VPLs of the last frame are updated according to the dynamic part of the scene. Then $VPLs_e$ is generated by selecting from the VPLs in $VPLs_p$ and $VPLs_c$ according to the foveated importance and density (line 8). 5. **Radiosity rendering**. At last, $VPLs_e$ is obtained and the radiosity of the scene is rendering with $VPLs_e$ (line 9). For each frame, we also keep the $VPLs_e$ and its importance for the computation in the next frame (line 10-11). In the following subsections, we will introduce these steps in detail.

3.1 VPL Candidates Generation

Once we have the voxelized representation of the scene (Step 1), $VPLs_c$ are then generated based on the location of the foveated region. Similar to bidirectional instant radiosity [38] and sequential Monte Carlo instant radiosity [14], we take each VPL as an end point of a light path, which is a path walking from light source before reaching camera, and place it on the surfaces of the scene that are indirectly visible (i.e. where a ray starting from the camera may land after two bounces) to the viewpoint. In order to do this, we first render the scene and sample the foveated region on the depth buffer uniformly using traditional rendering. We use uniform sampling in the foveated region under polar coordinates.

$$\begin{aligned} \mathrm{R} &= \sqrt{\mathrm{u}} \\ \theta &= 2\pi \mathrm{v} \end{aligned} \tag{1}$$

Where u and v are two uniformly distributed 1D variables from 0 to 1. Figure 2 shows the sampled result in the foveated region. Then a batch of rays are cast in the random directions from the hemisphere of these randomly chosen points and intersect with the voxels of the scene. The intersections are used to place the VPL candidates. In other words, these points are bounced once into the scene to generate the locations where the VPL candidates are put.

Then we use weighted resampling to get a uniformly distributed subset of VPL candidates approximately. We compute the foveated importance for these candidates with the method discussed in Section 3.2. A cumulative distribution function(CDF) is constructed based on the foveated importance. The resampling is implemented by selecting n random candidates from the CDF. Potential duplicate candidates selected by this method are eliminated by a separate check. As Hedman et al. proved in [14], the resampled candidates is helpful to keep temporal stability of illumination.

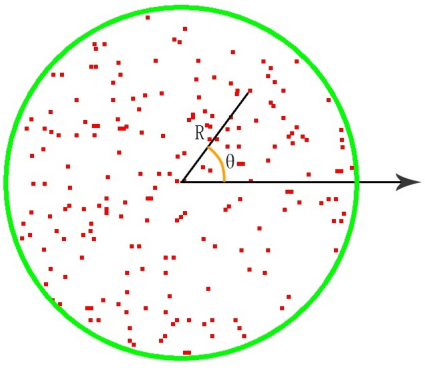

Fig. 2: The distribution of sampled points in the foveated region.

3.2 Foveated Importance Computation for VPLs

After selecting potentially important VPLs, we compute the foveated importance for VPLs, taking it as their illumination contribution for the current view, and use it in VPLs management.

Nevertheless, evaluating VPLs' contribution with the traditional ray tracing based methods is time-consuming. Because for each VPL, a certain amount of rays need to be emitted from sampled points on directly visible surfaces in the viewpoint to the VPL, and the visibility of these rays need to be determined and used to compute the VPL's contribution [38]. Hedman et al. proposed a mail-boxing scheme to determine indirect visibility for static scenes and used it into the instant radiosity method [14]. These methods transform the importance computation into indirect visibilities. Inspired from these ideas, we propose an efficient method to compute the importance of the VPLs, which is defined as foveated importance in our method, and the importance indicates the VPLs' illumination contribution for the foveated region. Our method can be used for dynamic scenes.

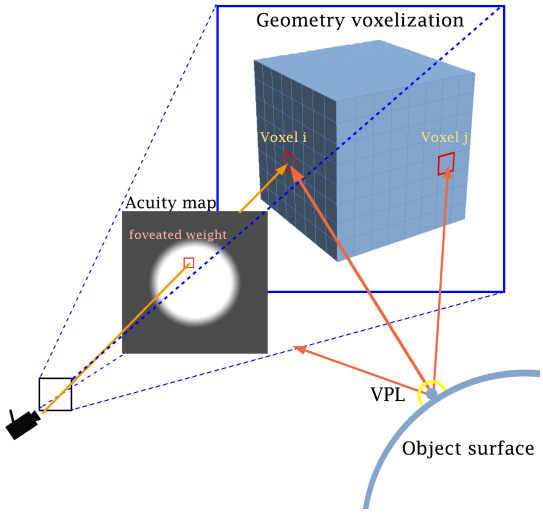

Fig. 3: Sketch map of foveated weight and foveated importance.

The foveated importance of VPL can be computed within two steps. The process is shown as a sketch map in Figure 3. First, we estimate a weight for each visible voxel in both static and dynamic voxelizations using Algorithm 2, which means the importance in our visual system and we call it foveated weight. The foveated weight represents its impact on our eyes, and voxels within the foveated region are assigned with larger weights. In Algorithm 2, Function FoveatedAcuity(v,foveatedC,foveatedR) is used to compute the visual acuity of v, which is set as the foveated weight of the corresponding voxel. A foveated acuity sampling map with a foveated center and radius is generated [42], and v is projected onto this map from the output viewpoint to get the value of the foveated weight.

Algorithm 2 Compute Foveated Weight

Input: static voxelization V_s, dynamic voxelization V_d, $foveatedC$, $foveatedR$, vp
Output: foveated weight for voxels $FoveatedWeight$

```
for each v ∈ V_s ∪ V_d do
    visibility ← VisibilityDetermine(v, vp)
    F ← FoveatedAcuity(v, foveatedC, foveatedR)
    FoveatedWeight[v] ← visibility * F
end for
return FoveatedWeight
```

Second, we calculate each VPL's foveated importance according to the weights of the visible voxels computed in the first step. The foveated importance of a VPL is an approximation of contribution of the VPL to output images especially in foveated regions. We use the idea of the diffuse cone model [2] to compute this value. As shown in Figure 3, for each VPL, we shoot rays in a fixed number of directions and

compute importance from intersected voxels of these rays. In essence, we convert the computation of the foveated region importance for VPL to the computation of the area on the hemisphere of VPL, which is generated by projecting the visible surface inside the foveated region. We tend to keep VPLs with large projection area longer but ignore the VPLs far away from the visible surface for the former contribute more to illumination. Changing these VPLs with large importance may break the temporal stability and cause flickers between the adjacent frames.

Then, we intersect the rays with the voxels, get the foveated weights of the first intersected voxels, and compute the foveated importance of the VPL by a directionally weighted accumulation of these voxel weights. The foveated importance computation for VPLs is shown in Algorithm 3.

Algorithm 3 Compute Foveated Importance

Input: VPL candidates $VPLs_c$, $FoveatedWeight$
Output: foveated importance for VPL candidates Imp_c

1: $Imp_c \leftarrow \varnothing$
2: **for** *each* $vpl \in VPLs_c$ **do**
3: $\quad Imp_{tmp} \leftarrow 0$
4: $\quad$ **for** *each* $d \in DIRECTIONS$ **do**
5: $\quad\quad v \leftarrow raycast(vpl, d)$
6: $\quad\quad Imp_{tmp} \leftarrow Imp_{tmp} + d.weight * FoveatedWeight[v]$
7: $\quad$ **end for**
8: $\quad Imp_c \leftarrow Imp_c \cup \{Imp_{tmp}\}$
9: **end for**
10: **return** Imp_c

As a result, VPLs with high foveated importance means they are important and contribute more illumination for the visible surface of the scene projected inside the foveated region.

Discussion (ray casting vs. cone tracing) In our implementation, we use ray casting instead of cone tracing (line 4). The importance obtained with cone tracing is smaller, especially when the cone intersects the voxels of the scene is that are far away from the VPL. This is because they use mipmapping to accelerate cone-voxel intersection, and mipmapping averages the importance of many empty voxels. The result is not acceptable for the objects with the shape of a narrow stripe. In ray casting, we intersect the ray with the voxel, and get its foveated weight directly.

3.3 VPL Management

Now that we are able to define the importance of each VPL. A foveated importance based VPL management method is introduced to determine $VPLs_e$. And we would like to introduce VPL management to obtain the $VPLs_e$. In order to render high-quality image of indirect illumination, several rules for VPL distribution may help.

First, the VPLs with large contribution should be added into the valid VPL set. Second, the migration of VPLs from the previous frame to the current frame should be smooth. Higher temporal coherence means higher reuse rate of the old VPLs. Third, the distribution of VPLs should be as uniform as possible for stable illumination in each frame. The process of VPL management is given in Algorithm 4 with consideration of these rules.

The first step of VPL management is to update $VPLs_p$ according to the information of the current frame and eliminate the invalid VPLs (line 3). The updated information includes geometry attributes, such as 3D position and normal, and foveated importance of VPL. Updating geometry attributes is mainly for VPLs on the dynamic objects, because these VPLs have to follow the movement of the objects. We record the object ID in each voxel and keep the object ID in each VPL. During the updating, the position of each VPL is transformed with the transformation matrix of the corresponding object. When the VPL is transformed to a new voxel of the current frame, the normal is found in the voxel and compared with its normal of the last frame. If the directions of these two normal change greatly, this VPL is eliminated. The VPLs with low updated foveated importance (below a predefined threshold) are also eliminated.

Algorithm 4 Manage VPL

Input: $VPLs_c, Imp_c$, valid VPL of the previous frame $VPLs_p$, importance of the previous valid VPLs Imp_p, $FoveatedWeight$
Output: valid VPLs for the current frame $VPLs_e$, importance of the current VPLs Imp_e

1: $VPLs_e \leftarrow \varnothing$
2: $Imp_e \leftarrow \varnothing$
3: $VPLs_p, Imp_{pn} \leftarrow UpdateVPLs(VPLs_p, FoveatedWeight)$
4: $VPLs_e, Imp_e \leftarrow VPLSelectionOld(VPLs_p, Imp_{pn}/Imp_p)$
5: $VPLs_{left} \leftarrow VPLs_p \setminus VPLs_e$
6: $Imp_{left} \leftarrow Imp_p \setminus Imp_e$
7: $VPLs_e{}^{tmp}, Imp_e{}^{tmp} \leftarrow VPLselectionCandidates(VPLs_{left}, Imp_{left}, VPLs_c, Imp_c)$
8: $VPLs_e \leftarrow VPLs_e \cup VPLs_e{}^{tmp}$
9: $Imp_e \leftarrow Imp_e \cup Imp_e{}^{tmp}$
10: **return** $VPLs_e$, Imp_e

According to rule 1 and 2, the VPLs with large importance need to be prioritized, and the valid VPLs of the last frame need to be reused, which can increase the rendering quality, reduce the flickers between frames and save time. Therefore, we further select the reusable VPLs for the current frame from these updated VPLs from the previous frame based on the foveated importance (line 4). The change rate of the current and the previous foveated importance is computed. Then, the probability that the VPL can be selected is calculated using Equation 2. The acceptance possibility of VPL with increased foveated importance is 1, which means the VPL is added into the reusable VPL list. The possibility of VPL with decreased importance is between 0 and 1, and the value represents the possibility is determined by:

$$\text{AcceptancePossibility} = \begin{cases} 1 - \cos\left(\frac{\pi}{2} \times \text{C}\right) & C \leq 1 \\ 1 & \text{otherwise} \end{cases} \quad \text{Where} \quad \text{C} = \frac{\text{Imp}_{\text{new}}}{\text{Imp}_{\text{old}}} \tag{2}$$

Because the viewpoint may change, the old VPL of the previous frame may not illuminate the new region that is newly visible from the current viewpoint. Here, we define $VPLs_c$ as the new VPL candidates generated from the current viewpoint. Therefore, we select the VPLs from $VPLs_c$, and add the result of the selection ($VPLs_e{}^{tmp}$) into $VPLs_e$ (Algorithm 4, line 7).

The detailed process of the VPL selection is written as Algorithm 5, in which all three rules are considered in the process. According to rule 1, the first part of $VPLs_e$ are selected from the new candidates with high importance. For rule 2, the second part of $VPLs_e$ can be selected from the new candidates and the left valid VPLs of the last frame by comparing their importance and density. With rule 3, density control is introduced to unify the distribution of $VPLs_e$. In order to obtain the density of each VPL, the 3D space is divided into uniform grids, a threshold $DENSITYLIMIT$ is predefined to indicate the maximum number of VPLs the cell can contain, and the density of the VPL is the number of valid VPL located in the cell over $DENSITYLIMIT$.

To add new VPLs from $VPLs_c$ to $VPLs_e$, we need to remove part of the VPLs from $VPLs_p$ and replace them with new VPL candidates. We design a VPL selection process to perform the replacement. First, we sort the new candidates in $VPLs_c$ with their importance and density in descending order and sort the left valid VPLs of the last frame ($VPLs_{left}$) in ascending order (line 2-6). Then, we compare the VPLs from the sorted candidates $VPLs_c$ with the sorted left valid VPLs $VPLs_{left}$ according to the value of $Imp + 1/(vd + C)$, where vd represent the density, C is a constant (line 7-15), select the VPLs that meets the importance and density requirements and add them into $VPLs_e{}^{tmp}$ (line 16-19). At last, if the number of the $VPLs_e{}^{tmp}$ is insufficient, the rest candidates in $VPLs_c$ are added in the sorted order to $VPLs_e{}^{tmp}$ until the number limitation is met (line 21-28).

Algorithm 5 VPL Selection

Input: left valid VPLs after selection from the previoud VPLs $VPLs_{left}$, importance of left previous VPLs Imp_{left}, $VPLs_c$, Imp_c
Output: ${VPLs_e}^{tmp}$, ${Imp_e}^{tmp}$

```
VPLs_e^tmp ← ∅
for each vpl ∈ VPLs_left ∪ VPLs_c do
    gDensity[vpl] ← GridDensity(vpl)
end for
SortAscending(VPLs_left, Imp_left, gDensity)
SortDescending(VPLs_c, Imp_c, gDensity)
N ← min(|VPLs_left|, |VPLs_c|)
for i = 1; i < N; i++ do
    vpl_l ← FirstElement(VPLs_left)
    vpl_c ← FirstElement(VPLs_c)
    VPLs_left ← VPLs_left \ vpl_l
    VPLs_c ← VPLs_c \ vpl_c
    vd_l ← gDensity(vpl_l)
    vd_c ← gDensity(vpl_c)
    if Imp[vpl_l] + 1/(vd_l + C) < Imp[vpl_c] 1/(vd_c + C) and vd_c < DENSITYLIMIT then
        VPLs_e^tmp ← VPLs_e^tmp ∪ {vpl_c}
    else
        VPLs_e^tmp ← VPLs_e^tmp ∪ {vpl}
    end if
end for
while |VPLs_e^tmp| < TOTALNUMBER and VPLs_c ≠ ∅ do
    vpl_c ← FirstElement(VPLs_c)
    VPLs_c ← VPLs_c \ vpl_c
    vd_c ← gDensity[vpl_c]
    if vd_c < DENSITYLIMIT then
        VPLs_e^tmp ← VPLs_e^tmp ∪ {vpl_c}
    end if
end while
Imp_e^tmp = getImportance(VPLs_e^tmp)
return VPLs_e^tmp, Imp_e^tmp
```

3.4 Radiosity Rendering

After $VPLs_e$ is obtained, we evaluate the radiosity of each VPL, and use it for rendering. A similar idea of importance computation is used to compute radiosity, but here cone tracing and mipmapping rather than ray casting on voxelization are used. The cone tracing based radiosity computation is used due to its simplicity and low time cost. The mipmap of voxelization is generated for each frame using the method used in [2]. The accumulation equation is also similar to the equation of the importance computation in Algorithm 3.

$$radiosity = \sum conetracing(D_i).radiosity \times D_i.weight \qquad (3)$$

The intensity of VPL is computed with its radiosity and probability density. Because the VPLs in our method are distributed uniformly, we simply use uniform probability density to compute the intensities. Our VPLs represent the end points of light paths, so most radiosity estimation methods can be integrated easily. Note that our method also can support multi-bounce indirect illumination by integrating the multi-bounce radiosity estimation like path tracing.

With each VPLs radiosity estimated, it is still time-consuming if we use all VPLs to render every pixel of the output image. To alleviate the computational cost, we refer to deferred shading and the interleaved sampling method [21] to accelerate rendering. The geometry in the output image is split into n×m tiles, and an approximately same small number of VPLs are assigned to each tile. To keep temporal coherence, old VPLs are assigned the same tile as the last frame. Then, the indirect illuminations of VPLs is accumulated for each tile and all tiles are combined into a whole image. However, there is some noise in the combined image, so we add a 5x5 box kernel filter to alleviate these artifacts.

4 Results and Discussion

We tested our method on five scenes with indirect illumination: Sponza has 279k triangles (Figure 1, Figure 4 row 3), Cornellbox has 5.8k triangles (Figure 4, row 1), Room has 476k triangles (row 2), Yard has 516k triangle (row 4) and Balcony has 192k triangles (row 5). We compared our method with path tracing and instant radiosity methods in quality and performance. Images of path tracing and instance radiosity are rendered with Mitsuba path tracer and virtual point light render [16]. In experiments, 2000 rays were sampled for each pixel in path tracing, and 2000 VPLs were used to illuminate the scene in instant radiosity method. We set four bounces indirect illumination in Mitsuba to achieve better visual effects. The resolution of the output images was 1024×1024. A gamma correction was applied to the output images of all three methods.

4.1 Quality

Our method uses about 1000 VPLs to render unless specified otherwise. In each frame, about 1000 candidates are generated initially, then only 1/10 of them are resampled per frame. These resampled VPLs and the valid VPLs from last frame are selected and added into the valid VPL set for the current frame. 20% of the valid VPLs' shadow maps are updated for each frame. Our method uses shadow maps of resolution 1024×1024 for direct lighting, and shadow maps of resolution 128×128 for VPLs. In Mitsuba, only one resolution can be set for both direct lighting and VPLs, so we render the instant radiosity results with two resolutions, one with 1024×1024 for both direct lighting and VPLs (IR), the other with 128×128 (IR'). Our method and the instant radiosity method use the same clamped geometry term in VPLs illumination, which is set manually as 0.03. The foveated radius in our method is assumed to be 25% of the width of the output image. Note that, however, our method is not limited to any fixed size and location of the foveated region.

Figure 3 shows the five images rendered with ray tracing (column 1), with our method (column 2), and with instant radiosity (IR in column 3, IR' in column 4). The green circles on the images of our method indicate the foveated regions of users. We also magnify the details inside the foveated region for comparison (column 5,top-left Path tracing, top-right ours, bottom-left IR, bottom-right IR'). The approximation errors produced by instant radiosity are salient: yellow color of the cube does not bleed on the white wall (row 1), color of the train bleeds excessively on the closet (row 2), color bleeding on the wall should be red that comes from the red plane, not green (row 3), the red on the barrier is too bright (row 4), and there should be green color combined with the shadow (row 5).

We quantify the quality produced by our rendering technique using two metrics. The first one is the average error E, which is defined as the average RGB error of each pixel compared to the ground truth image. The second one is the average temporal instability E', which is defined as the average RGB changes of each pixel between the adjacent frames of n frames with Equation 4.

$$E' = \frac{\sum_{i=2}^{n} E_{i-1,i}}{n-1} \qquad (4)$$

E is used to describe the pixel intensity errors observed in the final image, and E' indicates the temporal changes in the sequences. We quantify the rendered quality with E for the foveated region and periphery region respectively.

Table 1 shows the average pixel intensity errors E of the foveated region and the periphery region for our scenes, for our method and instant radiosity method. As can be seen in Table 1, with our method, the approximation errors of the foveated region are consistently smaller than the errors of the periphery region. While with instant radiosity method, the results are not consistent. Our method processes the foveated region and the periphery region differently, and gives more benefits to the foveated region since the user's visual system is more sensitive to the details of the foveated region. Instant radiosity method considers all regions of the output image as same, so the errors of different regions does not follow the obvious rules.

Fig. 4: Comparison between path tracing (column 1), our method (column 2) and instant radiosity method (IR 1024×1024, column 3; IR' 128×128, column 4). The details in the red square are magnified in column 5 (top-left: path tracing; top-right: ours, bottom-left: IR; bottom-right: IR').

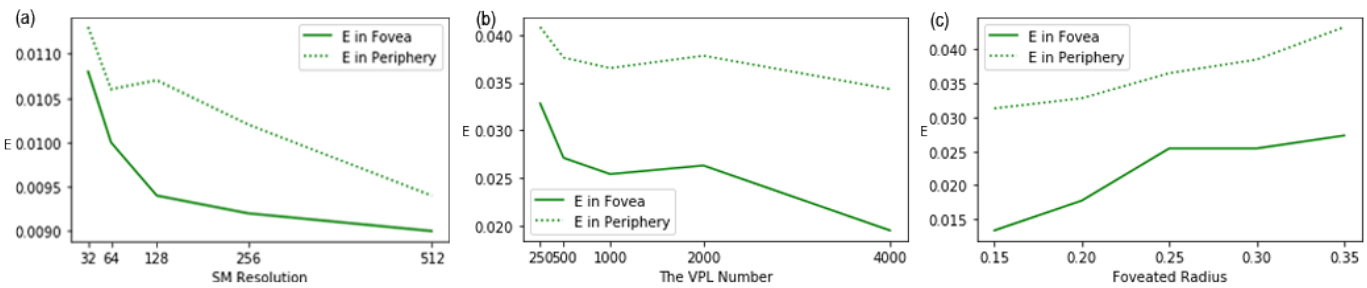

Fig. 5: Errors as a function of (a) the resolution of shadow map, (b) the VPL number and (c) the foveated radius.

The average pixel errors of the foveated region for Cornellbox and Room scenes produced by our method and instant radiosity are small. Ours are slightly larger than those of instant radiosity. One reason is the sizes of Cornellbox and Room are small and 2000 VPLs are used to render the output image with instant radiosity. Therefore, even without the optimization of VPL distribution, the number of VPLs that happen to illuminate the foveated area is sufficient for achieving good quality of rendering. Another reason is that there are small errors for direct lighting between the path tracer of Mitsuba and our method, 0.012 for Cornellbox and 0.011 for Room. This accounts for a large percentage of the total errors when the error values are small. The approximation errors of the foveated region for other scenes produced by our method are smaller than the errors produced by instant radiosity method. The errors of the periphery region of our method are not always smaller than the errors of instant radiosity. This is because our method optimizes the VPL distribution based on the location of the foveated region and generate high-quality illumination effect in the foveated region. Our method can not guarantee smaller errors of the periphery regions compared with instant radiosity method.

Table 1: Average pixel Errors in the Foveated and Periphery Regions for Our Method and for Instant Radiosity.

Scene	E in Fovea			E in Periphery		
	IR	IR'	Ours	IR	IR'	Ours
Cornellbox	0.011	0.008	0.020	0.012	0.011	0.028
Room	0.014	0.019	0.026	0.013	0.015	0.029
Sponza	0.036	0.043	0.036	0.052	0.058	0.058
Yard	0.020	0.028	0.018	0.017	0.027	0.024
Balcony	0.045	0.061	0.045	0.055	0.040	0.067

Table 2 shows the average temporal errors E' of the foveated region and the periphery region for the scenes, for our method and instant radiosity method. The length of the test sequences is about 40s. There are several cases of the adjacent frames. One is to keep all the factors static, such as the scene, light sources and the viewpoint. Others are to change one or more factors, such as changing the position of the light sources, changing the viewpoint and animating the objects in the scene. The temporal errors of the foveated region with our method are much smaller than the errors of instant radiosity method, and the reduction is from 0.032 to 0.002. Even the temporal errors of the periphery region with our method are smaller than the errors of instant radiosity method for all five scenes, because our method reuses the old valid VPLs of the last frame and only introduce a small number of the new VPLs for a new frame, which helps us to avoid large temporal errors of the adjacent frames. Instant radiosity method does not consider the temporal coherence of VPLs, and regenerates VPLs from the scratch for each frame.

It is also can be seen from Table 2, the temporal errors of the foveated regions with our method are smaller than the errors of the periphery regions. While with instant radiosity method, larger temporal errors may appear in the different regions for the different scenes. The reason is that we consider the foveated importance when selecting the reusable VPLs from the old VPLs.

The images on row 3 in Figure 1 are the visualization of the temporal error for each pixel for Sponze scene when the light source is moving. The visualization of our method (column 2) is much similar to the result of path tracing (column 1). The temporal error visualization generated with instant radiosity method has more gray pixels, which indicates more errors between the adjacent frames and lower temporal stability of the method.

Several parameters of our method affect the image quality in the foveated and periphery regions, such as the resolution of the shadow maps, the number of VPLs, and the radius of the foveated region. Figure 5 shows the average errors of our method as functions of the resolution of shadow maps, the VPL number and the foveated radius. The corresponding rendered images and pixel error visualization are shown in Figure 6.

Figure 5 (a) shows the approximation errors in the foveated region and in the periphery region of our method as a function of the shadow map resolution. For both regions, the errors vary little with the resolution of the shadow map. Although high-resolution shadow maps provide more accurate visibility of the VPLs, there are about one thousand VPLs in the scene, and the intensity of a pixel is blended multiple times due to multiple shadow maps casting on it.

Table 2: Average Temporal Errors in the Foveated and Periphery Regions for Our Method and for Instant Radiosity.

Scene	E' in Fovea			E' in Periphery		
	IR	IR'	Ours	IR	IR'	Ours
Cornellbox	0.006	0.011	0.004	0.013	0.016	0.007
Room	0.011	0.009	0.005	0.010	0.012	0.006
Sponza	0.046	0.054	0.014	0.024	0.047	0.014
Yard	0.013	0.019	0.007	0.015	0.018	0.011
Balcony	0.027	0.032	0.013	0.021	0.033	0.005

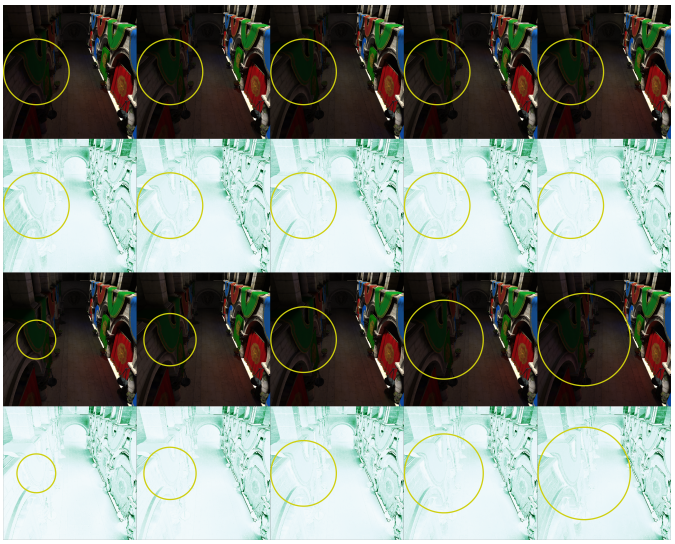

Fig. 6: Rendered effects and pixel error visualization of our method with the different VPL numbers and the different foveated radius.

For all the experiments described so far, we used 1000 VPLs to approximate indirect illumination, the number that is sufcient for small errors. (b) shows the approximation errors in the foveated region and in the periphery region of our method as a function of the number of VPLs. In these two regions, errors decrease with the number of VPL. That is, more VPLs, and better indirect lighting. It is also can be seen from (b), the errors of the foveated region decrease faster than those of the periphery region. This is because our method optimizes the VPL distribution according to the location of the foveated region, so the increased VPLs benefit more to the foveated region than the periphery region. Figure 6 row 1 shows the images rendered with different VPL numbers, 250, 500, 1000, 2000 and 4000. The images inside the foveated region become clearer and brighter when the VPL number is increasing. The images in row 2 visualize the pixel error of results. Brighter region means the region with smaller errors, vice versa. The pixel error changes in the images of row 2 indicate the results inside and around the foveated region become more similar to the results of path tracing.

For all the experiments described so far, the foveated radius is set to 25% of the width of the output image. Figure 5 (c) shows the approximation errors in the foveated region and in the periphery region of our method as a function of the foveated radius. The errors are increased when the foveated radius is increasing. Because the density of VPLs is proportional to the quality of the indirect illumination produced by the VPLs, smaller foveated region has greater VPL density casting on it.

Figure 6 row 3 shows the images rendered with different foveated

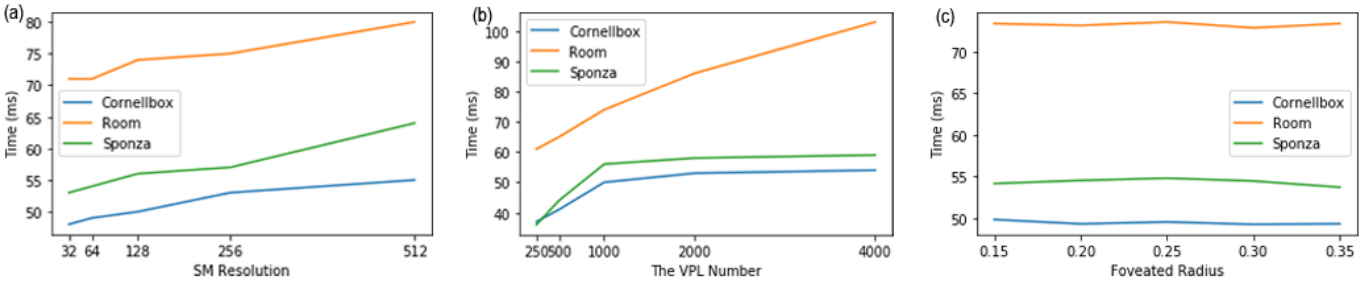

Fig. 7: Time cost as a function of (a) the resolution of shadow map, (b) the VPL number and (c) the foveated radius.

radius, 0.15, 0.20, 0.25, 0.30 and 0.35. The images inside the foveated region and the nearby region become blurrier when increasing the foveated radius. In the pixel error visualization in row 4, the pixels inside or around the foveated region are brighter with smaller foveated radius, which also indicates the quality of this region is close to the quality of path tracing when the foveated radius is small.

4.2 Performance

All performance numbers reported in this paper were recorded on a PC workstation with a 3.7 GHz Intel(R) Core(TM) i7-8700k CPU, with 16 GB of memory, and with an NVIDIA GeForce GTX 1070 graphics card. The time cost of path tracing and instant radiosity is calculated by Mitsuba.

Table 3 shows the frame rendering times for our method and the speedup versus path tracing and versus instant radiosity. Mitsuba uses BVH (bounding volume hierarchy) scene partitioning for acceleration. Our method is substantially faster than path tracing. The speedup is from 13,916× to 38,923× for the scenes tested. The time cost of instant radiosity in this section uses 1024×1024 (IR) or 128×128 (IR') shadow maps for both direct lighting and VPLs. Compared to instant radiosity method, our method also achieves a substantial speedup. The acceleration is from 118× to 184× for IR and from 50× to 166× for IR'. Since our method optimizes the distribution of the valid VPLs according to the foveated region, the images in the foveated regions with comparable quality can be generated with only half the number of VPLs and a small number of the updated VPL shadow maps.

Table 3: Performance of Our Method Compared to Path Tracing and Instant Radiosity(ms)

Scene	Cornellbox	Room	Sponza	Yard	Balcony
Ours	51	74	56	65	61
Speedup vs. PT	13,916×	29,829×	38,923×	18,219×	26,833×
Speedup vs. IR	182×	176×	184×	118×	125×
Speedup vs. IR'	50×	142×	166×	105×	97×

As shown in Table 4, we calculate the time cost of each step independently. The voxelization column shows the time cost of the voxelization for the dynamic objects in each frame, such as the cow in Cornellbox, the train in Room, and the plane in Sponza. Yard and Balcony scene has not any dynamic object, so they don't have time cost for the voxelization. The summation of the time cost for VPL generation, foveated importance computation and VPL management is shown in column valid VPLs generation. According to the data Table 4, the voxelization for dynamic objects and Valid VPLs generation are very fast, and our method spends most of our time on radiosity rendering. The time cost for radiosity rendering can be split into two parts: shadow map updating and shading. The time cost of these two parts is about half and half.

Figure 5 (a), (b) and (c) show the frame rendering times for our method as a function of the resolution of shadow maps, the VPL number and the foveated radius. As expected, the frame time increases when rendering the large shadow maps for the valid VPLs or using more VPLs to illuminate the scene. The performance data in (c) indicates the time cost vary little with the foveated radius, since all processes are kept the same with different foveated radius.

Table 4: Time Cost for the Steps of Our Method (ms)

Scene	Voxelization	Valid VPLs Generation	Rendering	
			Shadow maps	Shading
Cornellbox	12.1	3.8	9.0	25.6
Room	5.2	4.0	38.8	25.6
Sponza	5.7	4.0	20.4	25.6
Yard	7.9	2.0	20.3	30.9
Balcony	5.1	3.9	30.2	26.2

5 Conclusion, Limitations and Future Work

We have presented a foveated instant radiosity method for rendering high-quality global illumination in the foveated region based on optimizing the distribution of VPLs. We render 3D scenes with multiple resolutions of illumination. Our method has a signicant performance advantage over ray tracing while retaining acceptable quality. Compared to instant radiosity, our method produces not only better results but also higher frame rates. Our method also supports scenes with dynamic objects thanks to our reusing scheme of VPLs. Our method reuses valid VPLs from the previous frame, based on a novel importance metric of VPLs across adjacent frames. Our method produces smooth illumination, suppressing the temporal artifacts caused by sudden changes in the valid VPLs set, and achieves better temporal stability than the instant radiosity method.

One limitation is that, our method doesn't work well for rendering scenes with high-speed moving objects. Our method reuses VPLs temporally. If the dynamic objects move fast, the flickers may appear and the pixel errors may increase. This is a limitation inherent in the method based on temporal coherence. To alleviate artifacts, one possible future direction is to select some VPLs shot from both light sources and the viewpoint. In this way, the overall illumination will be more stable.

Another limitation of our method is missing the salient color bleeding in the periphery region that may be noticed by users. We calculate foveated importance to estimate VPLs contribution to images. However, there are some scenarios that the foveated region is very dark and do not have any obvious indirect illumination, while the periphery regions have strong color bleeding. Our method may ignore such indirect illumination since they are not in the foveated region. So another possible future direction is to combine the foveated importance with the extent of color bleeding. For example, more VPLs can be placed in regions with significant indirect illumination.

REFERENCES

[1] T. Barák, J. Bittner, and V. Havran. Temporally coherent adaptive sampling for imperfect shadow maps. In *Computer Graphics Forum*, vol. 32, pp. 87–96. Wiley Online Library, 2013.

[2] C. Crassin, F. Neyret, M. Sainz, S. Green, and E. Eisemann. Interactive indirect illumination using voxel cone tracing. In *Computer Graphics Forum*, vol. 30, pp. 1921–1930. Wiley Online Library, 2011.

[3] T. Davidovič, J. Křivánek, M. Hašan, P. Slusallek, and K. Bala. Combining global and local virtual lights for detailed glossy illumination. In *ACM Transactions on Graphics (TOG)*, vol. 29, p. 143. ACM, 2010.

[4] A. T. Duchowski, D. Bate, P. Stringfellow, K. Thakur, B. J. Melloy, and A. K. Gramopadhye. On spatiochromatic visual sensitivity and peripheral color lod management. *ACM Transactions on Applied Perception (TAP)*, 6(2):9, 2009.

[5] A. T. Duchowski and A. Çöltekin. Foveated gaze-contingent displays for peripheral lod management, 3d visualization, and stereo imaging. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 3(4):6, 2007.

[6] A. T. Duchowski, N. Cournia, and H. Murphy. Gaze-contingent displays: A review. *CyberPsychology & Behavior*, 7(6):621–634, 2004.

[7] A. T. Duchowski, D. H. House, J. Gestring, R. I. Wang, K. Krejtz, I. Krejtz, R. Mantiuk, and B. Bazyluk. Reducing visual discomfort of 3d stereoscopic displays with gaze-contingent depth-of-field. In *Proceedings of the acm symposium on applied perception*, pp. 39–46. ACM, 2014.

[8] M. Fujita and T. Harada. Foveated real-time ray tracing for virtual reality headset. *Light Transport Entertainment Research*, 2014.

[9] I. Georgiev and P. Slusallek. Simple and robust iterative importance sampling of virtual point lights. In *Eurographics (Short Papers)*, pp. 57–60, 2010.

[10] B. Guenter, M. Finch, S. Drucker, D. Tan, and J. Snyder. Foveated 3d graphics. *ACM Transactions on Graphics (TOG)*, 31(6):164, 2012.

[11] K. Gupta and S. Kazi. Gaze contingent depth of field display.

[12] M. Hašan, E. Velázquez-Armendáriz, F. Pellacini, and K. Bala. Tensor clustering for rendering many-light animations. In *Computer Graphics Forum*, vol. 27, pp. 1105–1114. Wiley Online Library, 2008.

[13] Y. He, Y. Gu, and K. Fatahalian. Extending the graphics pipeline with adaptive, multi-rate shading. *ACM Transactions on Graphics (TOG)*, 33(4):142, 2014.

[14] P. Hedman, T. Karras, and J. Lehtinen. Sequential monte carlo instant radiosity. *IEEE transactions on visualization and computer graphics*, 23(5):1442–1453, 2017.

[15] S. Hillaire, A. Lécuyer, R. Cozot, and G. Casiez. Using an eye-tracking system to improve camera motions and depth-of-field blur effects in virtual environments. In *2008 IEEE virtual reality conference*, pp. 47–50. IEEE, 2008.

[16] W. Jakob. Mitsuba renderer, 2010. http://www.mitsuba-renderer.org.

[17] H. W. Jensen. *Realistic image synthesis using photon mapping*. AK Peters/CRC Press, 2001.

[18] A. Keller. Instant radiosity. 1997.

[19] H. Ki and K. Oh. A gpu-based light hierarchy for real-time approximate illumination. *The Visual Computer*, 24(7-9):649–658, 2008.

[20] M. Knecht, C. Traxler, O. Mattausch, W. Purgathofer, and M. Wimmer. Differential instant radiosity for mixed reality. In *2010 IEEE International Symposium on Mixed and Augmented Reality*, pp. 99–107. IEEE, 2010.

[21] S. Laine, H. Saransaari, J. Kontkanen, J. Lehtinen, and T. Aila. Incremental instant radiosity for real-time indirect illumination. In *Proceedings of the 18th Eurographics conference on Rendering Techniques*, pp. 277–286. Eurographics Association, 2007.

[22] M. Levoy and R. Whitaker. Gaze-directed volume rendering. In *ACM SIGGRAPH Computer Graphics*, vol. 24, pp. 217–223. ACM, 1990.

[23] T. Lindeberg. Concealing rendering simplifications using gazecontingent depth of field, 2016.

[24] D. Luebke and B. Hallen. Perceptually driven simplification for interactive rendering. In *Rendering Techniques 2001*, pp. 223–234. Springer, 2001.

[25] R. Mantiuk, B. Bazyluk, and A. Tomaszewska. Gaze-dependent depth-of-field effect rendering in virtual environments. In *International Conference on Serious Games Development and Applications*, pp. 1–12. Springer, 2011.

[26] M. Mauderer, S. Conte, M. A. Nacenta, and D. Vishwanath. Depth perception with gaze-contingent depth of field. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 217–226. ACM, 2014.

[27] X. Meng, R. Du, M. Zwicker, and A. Varshney. Kernel foveated rendering. *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, 1(1):5, 2018.

[28] E. N. Molenaar. Towards real-time ray tracing through foveated rendering. Master's thesis, 2018.

[29] H. Murphy and A. T. Duchowski. Gaze-contingent level of detail rendering. *EuroGraphics 2001*, 2001.

[30] G. Nichols and C. Wyman. Multiresolution splatting for indirect illumination. In *Proceedings of the 2009 symposium on Interactive 3D graphics and games*, pp. 83–90. ACM, 2009.

[31] T. Ohshima, H. Yamamoto, and H. Tamura. Gaze-directed adaptive rendering for interacting with virtual space. In *Proceedings of the IEEE 1996 Virtual Reality Annual International Symposium*, pp. 103–110. IEEE, 1996.

[32] D. Parkhurst, I. Law, and E. Niebur. Evaluating gaze-contingent level of detail rendering of virtual environments using visual search. In *Symposium on Eye Tracking Research and Applications (ETRA)*, pp. 105–109, 2000.

[33] A. Patney, M. Salvi, J. Kim, A. Kaplanyan, C. Wyman, N. Benty, D. Luebke, and A. Lefohn. Towards foveated rendering for gaze-tracked virtual reality. *ACM Transactions on Graphics (TOG)*, 35(6):179, 2016.

[34] R. Prutkin, A. Kaplanyan, and C. Dachsbacher. Reflective shadow map clustering for real-time global illumination. In *Eurographics (Short Papers)*, pp. 9–12, 2012.

[35] E. M. Reingold, L. C. Loschky, G. W. McConkie, and D. M. Stampe. Gaze-contingent multiresolutional displays: An integrative review. *Human factors*, 45(2):307–328, 2003.

[36] T. Ritschel, E. Eisemann, I. Ha, J. D. Kim, and H.-P. Seidel. Making imperfect shadow maps view-adaptive: High-quality global illumination in large dynamic scenes. In *Computer Graphics Forum*, vol. 30, pp. 2258–2269. Wiley Online Library, 2011.

[37] M. Schwarz and H.-P. Seidel. Fast parallel surface and solid voxelization on gpus. *ACM transactions on graphics (TOG)*, 29(6):179, 2010.

[38] B. Segovia, J. C. Iehl, R. Mitanchey, and B. Péroche. Bidirectional instant radiosity. In *Rendering Techniques*, pp. 389–397, 2006.

[39] B. Segovia, J. C. Iehl, and B. Péroche. Metropolis instant radiosity. In *Computer Graphics Forum*, vol. 26, pp. 425–434. Wiley Online Library, 2007.

[40] F. X. Sillion, C. Puech, et al. *Radiosity and global illumination*, vol. 1. Springer, 1994.

[41] F. Simon, J. Hanika, and C. Dachsbacher. Rich-vpls for improving the versatility of many-light methods. In *Computer Graphics Forum*, vol. 34, pp. 575–584. Wiley Online Library, 2015.

[42] M. Stengel, S. Grogorick, M. Eisemann, and M. Magnor. Adaptive image-space sampling for gaze-contingent real-time rendering. In *Computer Graphics Forum*, vol. 35, pp. 129–139. Wiley Online Library, 2016.

[43] N. T. Swafford, J. A. Iglesias-Guitian, C. Koniaris, B. Moon, D. Cosker, and K. Mitchell. User, metric, and computational evaluation of foveated rendering methods. In *Proceedings of the ACM Symposium on Applied Perception*, pp. 7–14. ACM, 2016.

[44] M. Vinnikov and R. S. Allison. Gaze-contingent depth of field in realistic scenes: The user experience. In *Proceedings of the symposium on eye tracking research and applications*, pp. 119–126. ACM, 2014.

[45] I. Wald, C. Benthin, and P. Slusallek. Interactive global illumination in complex and highly occluded environments. In *Rendering Techniques*, pp. 74–81, 2003.

[46] H. Yee, S. Pattanaik, and D. P. Greenberg. Spatiotemporal sensitivity and visual attention for efficient rendering of dynamic environments. *ACM Transactions on Graphics (TOG)*, 20(1):39–65, 2001.