



## 基于凝视点的多分辨率光照渲染方法的探究

汇报人：

石雪怀

单位：

北京航空航天大学  
虚拟现实技术与系统国家重点实验室

指导老师：

王莉莉教授

# 目 录

## ■ 凝视点渲染概述

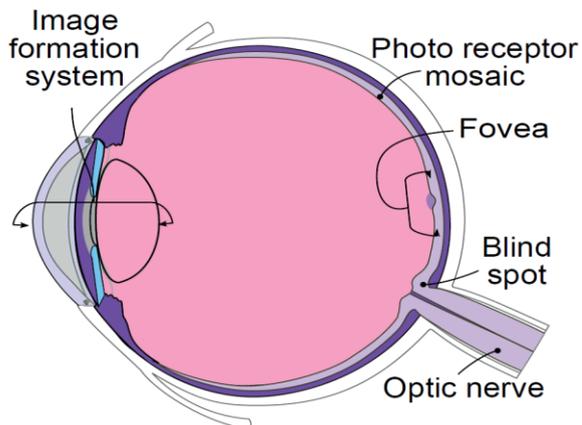
## ■ 基于凝视点的立即辐射度方法

Wang L, Li R, Shi X, et al. Foveated instant radiosity[C]//2020 IEEE International Symposium on Mixed and Augmented Reality (ISMAR). IEEE, 2020: 1-11. (CCF B类会议)

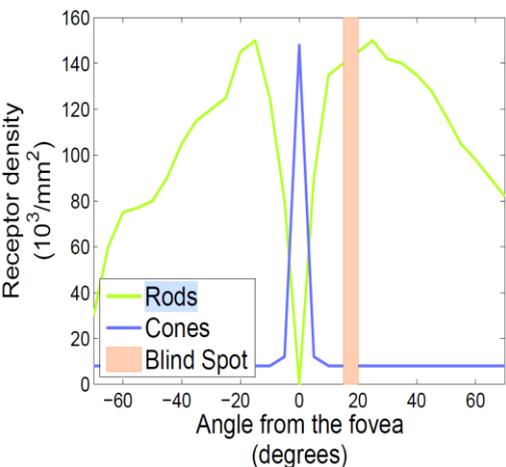
## ■ 基于凝视点的光子映射方法

Shi X, Wang L, Wei X, et al. Foveated Photon Mapping[J]. IEEE Transactions on Visualization and Computer Graphics, 2021, 27(11): 4183-4193. (CCF A类期刊)

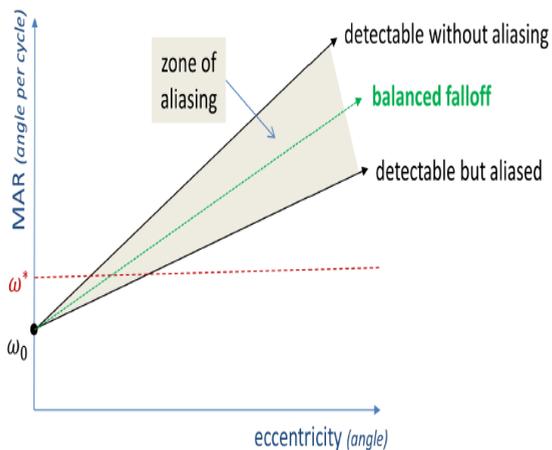
# 凝视点渲染



Wandell et al. 1995



Wandell et al. 1995



Guenter et al. 2012

- 研究方向：多分辨率光照渲染
- 研究目标：在用户无感知的基础上，对光照渲染质量进行降低，从而降低对计算机的性能要求；
- 算法输入：场景数据，实时眼动数据；
- 算法输出：用户无感知的低质量光照渲染结果。

## 技术难点

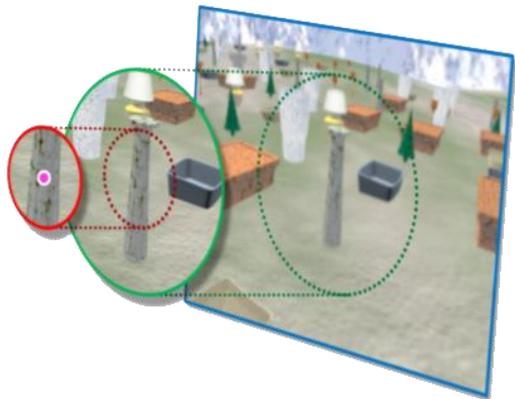
1. 如何与当前实时渲染方法（IR、PM、VCM等）结合来提升渲染方法性能？
2. 如何在视点、注视点动态情况下维持帧间稳定性？
3. 如何无感降低周边区域渲染质量？

# Foveated Instant Radiosity

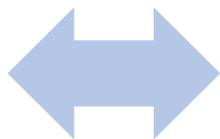
基于凝视点的立即辐射度方法

# 动机

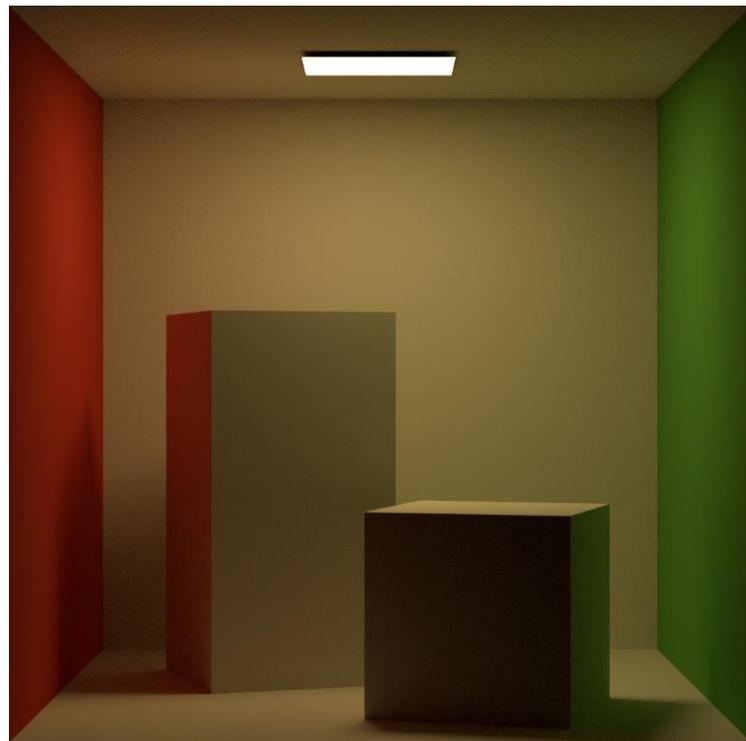
- 凝视点渲染(foveated rendering)与全局光照



[Guenter et al. TOG 2012]

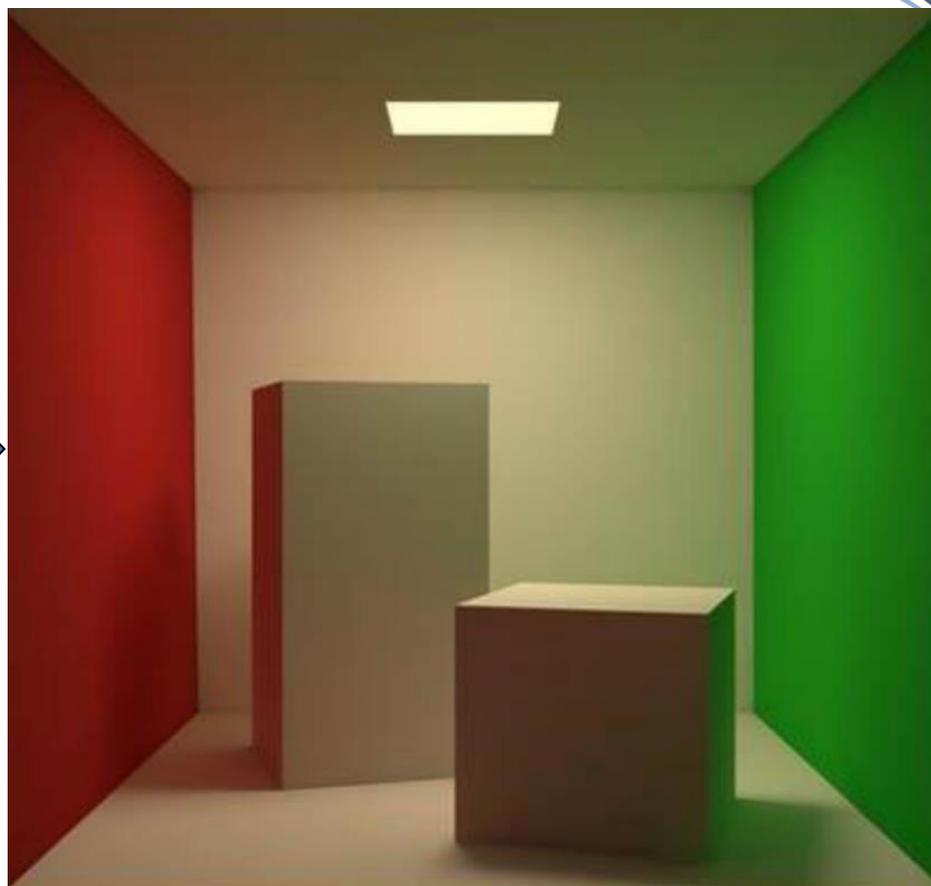
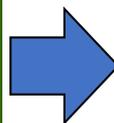
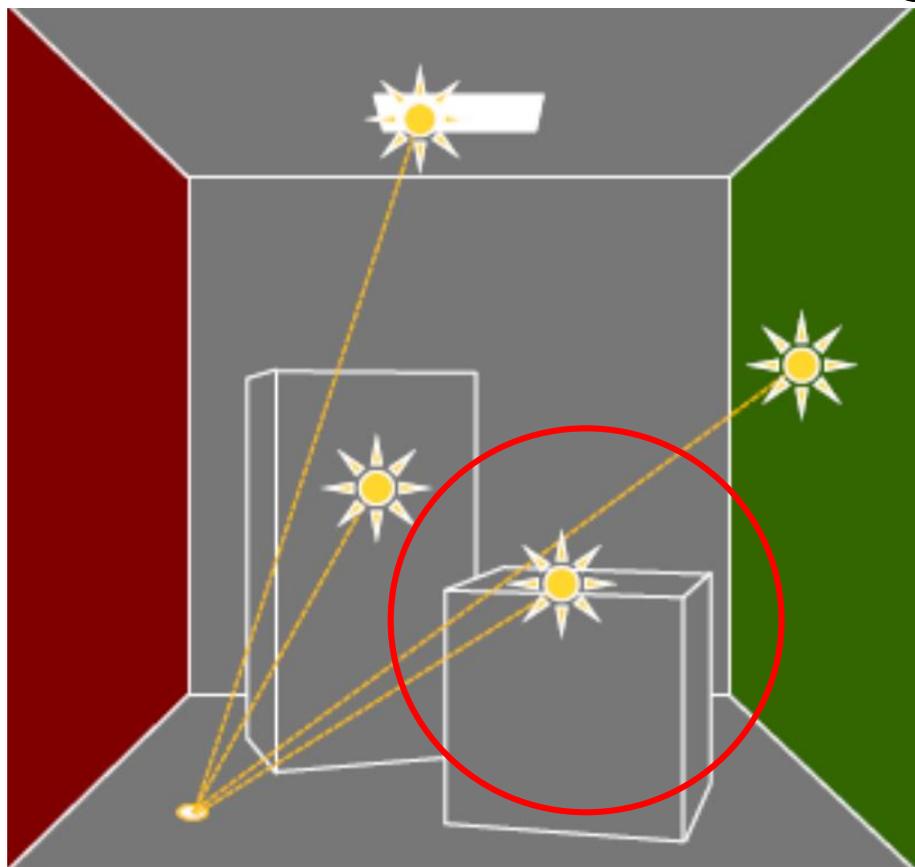


[Weier et al. CGF 2016]



# 动机

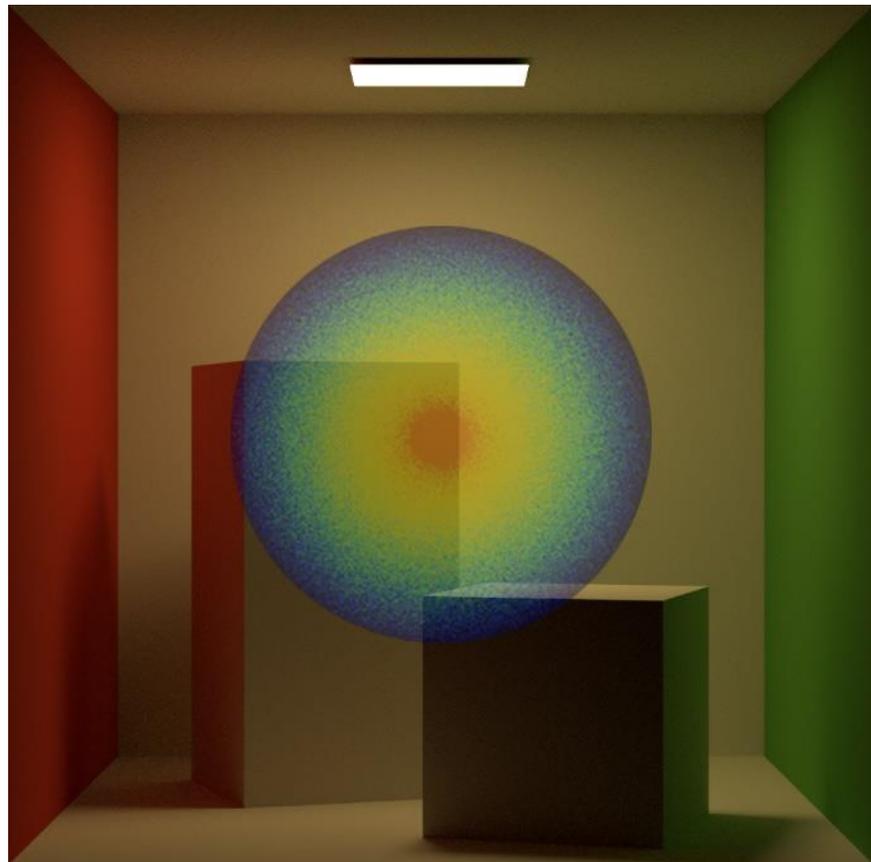
- 凝视点渲染(foveated rendering)与全局光照



立即辐射度(instant radiosity)

# 难点

- 使立即辐射度适用凝视点渲染



- 如何根据每帧的凝视区域生成VPL?
- 在视点、凝视区域或场景发生变化时, 如何保持连续帧之间的VPL分布与光照的稳定性?

# 整体流程



# 场景体素化

1. 场景体素化

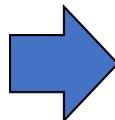
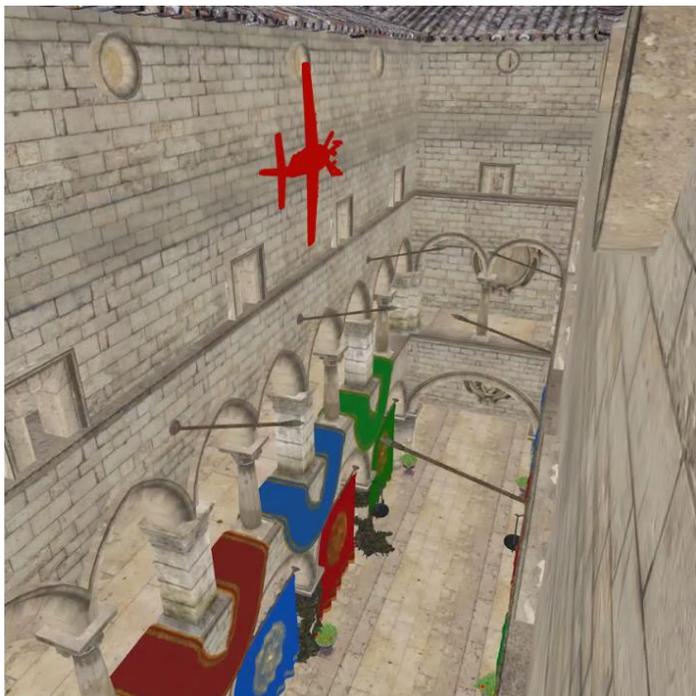
2. 生成VPL  
候选点

3. 凝视重要度  
计算

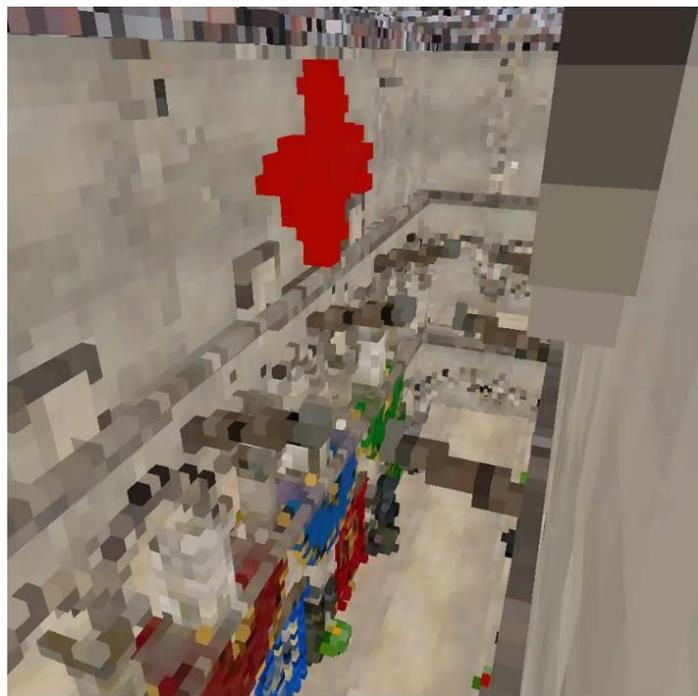
4. VPL管理

5. 最终渲染

## 原场景



## 体素化场景



# 生成VPL候选点

## 2.1 凝视区域均匀采样

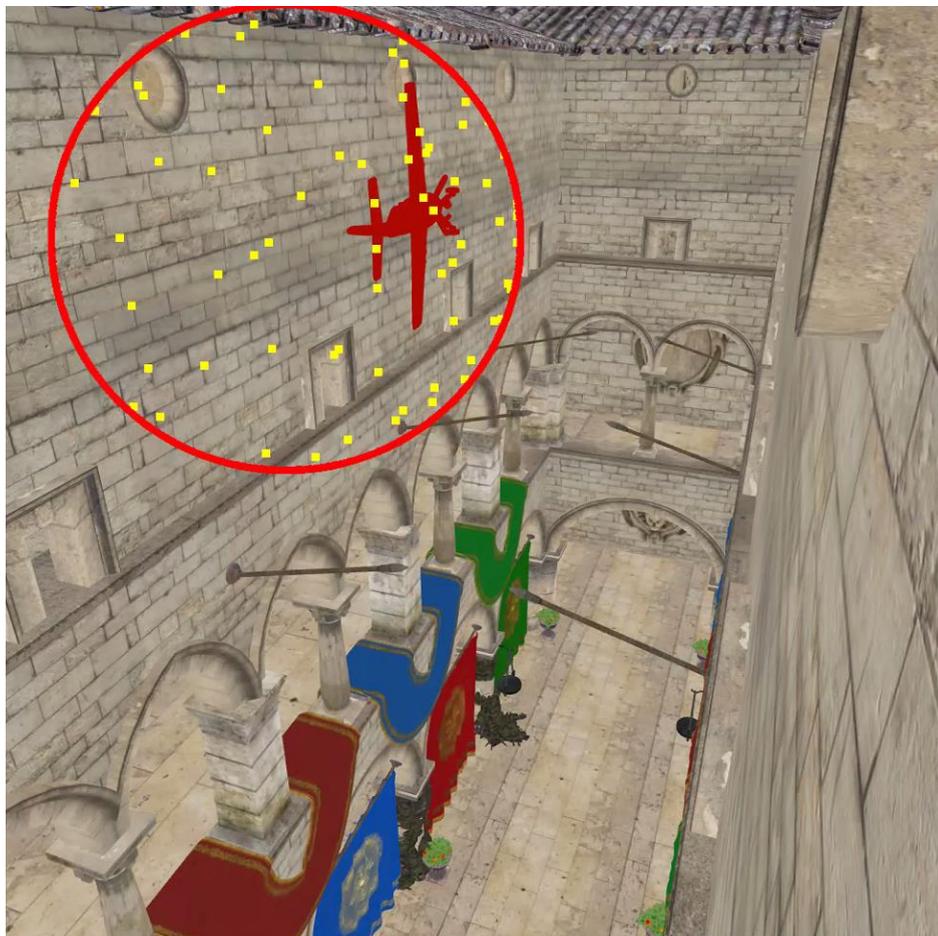
1. 场景体素化

2. 生成VPL  
候选点

3. 凝视重要度  
计算

4. VPL管理

5. 最终渲染



# 生成VPL候选点

1. 场景体素化

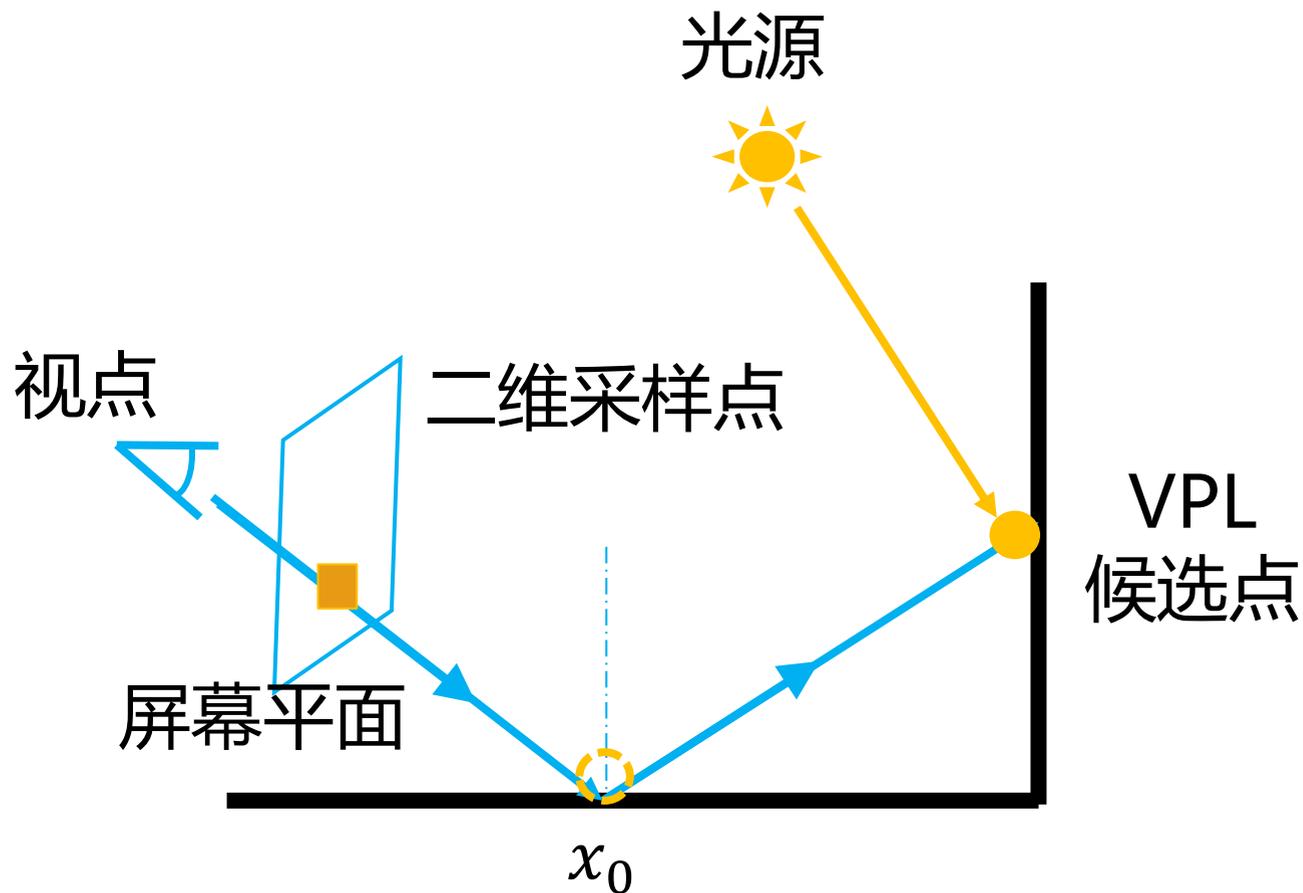
2. 生成VPL  
候选点

3. 凝视重要度  
计算

4. VPL管理

5. 最终渲染

## 2.2 每个采样弹射一次生成VPL候选点



# 凝视重要度计算

## 计算流程

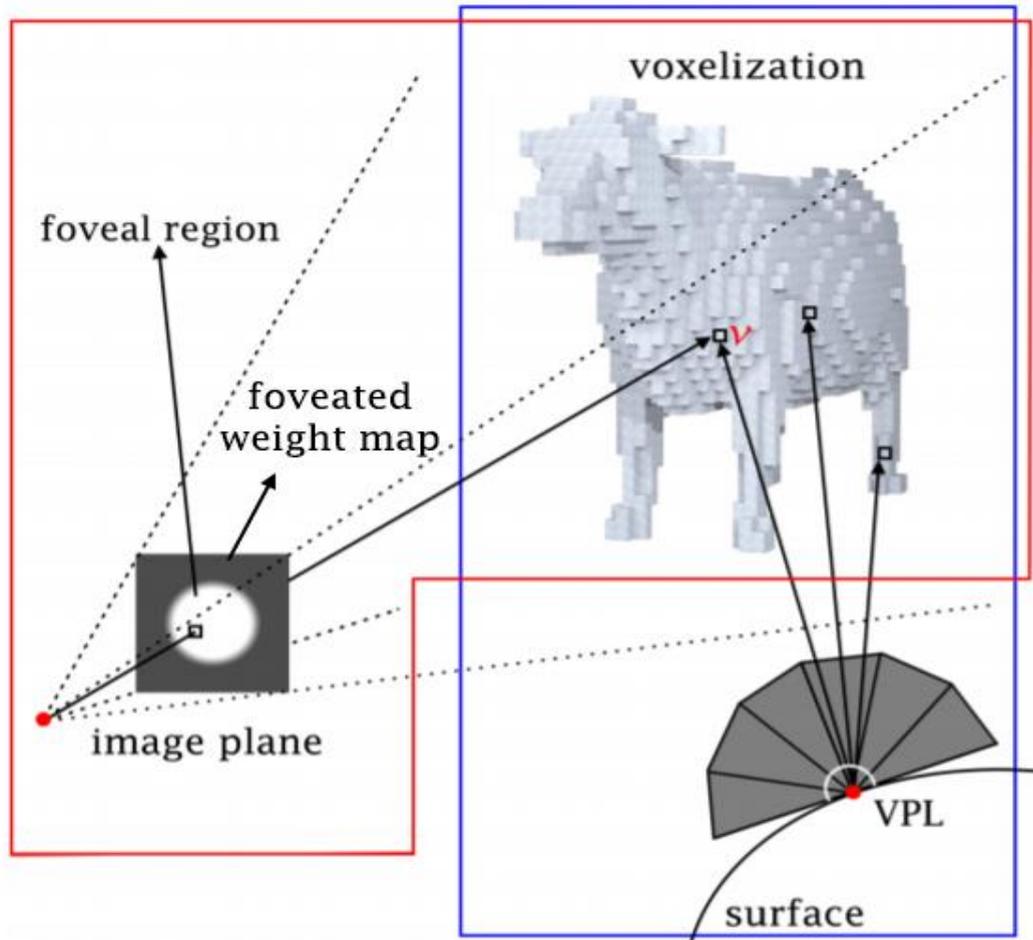
1. 场景体素化

2. 生成VPL  
候选点

3. 凝视重要度  
计算

4. VPL管理

5. 最终渲染



# 凝视重要度计算

1. 场景体素化

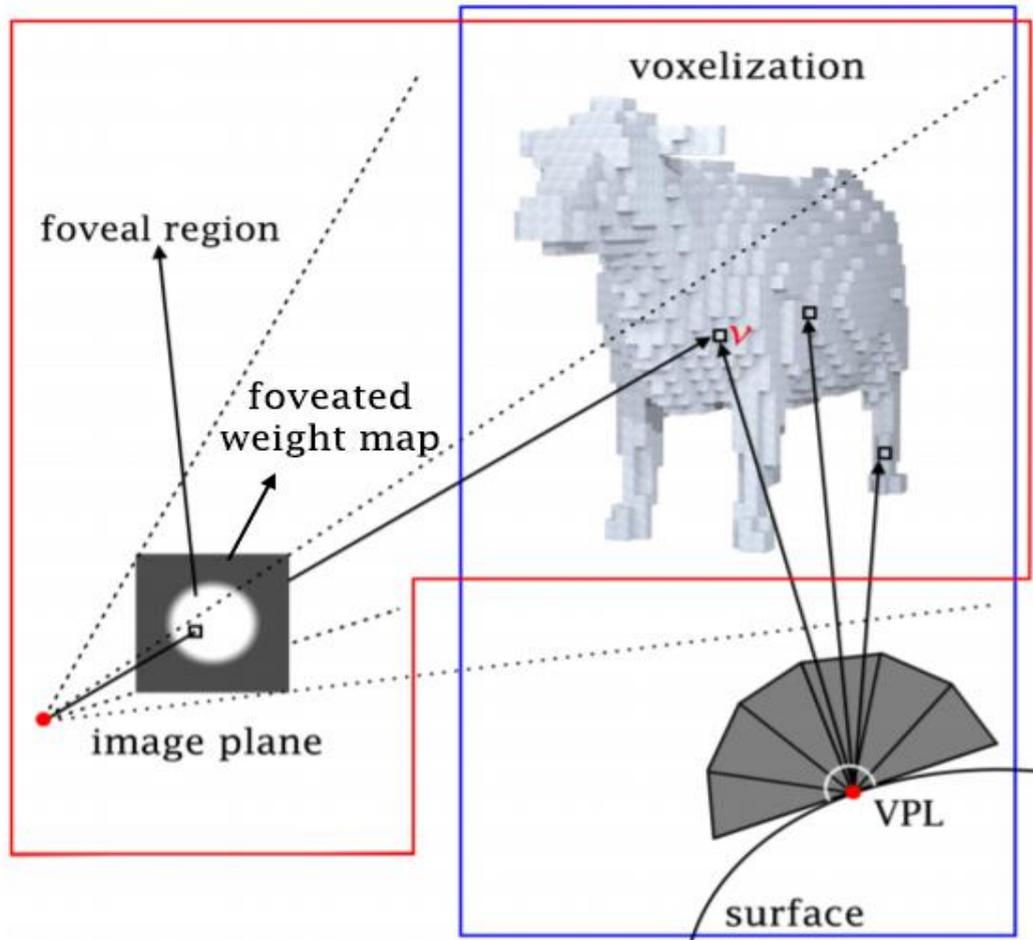
2. 生成VPL  
候选点

3. 凝视重要度  
计算

4. VPL管理

5. 最终渲染

## 3.1 体素视觉敏感度估计



# 凝视重要度计算

1. 场景体素化

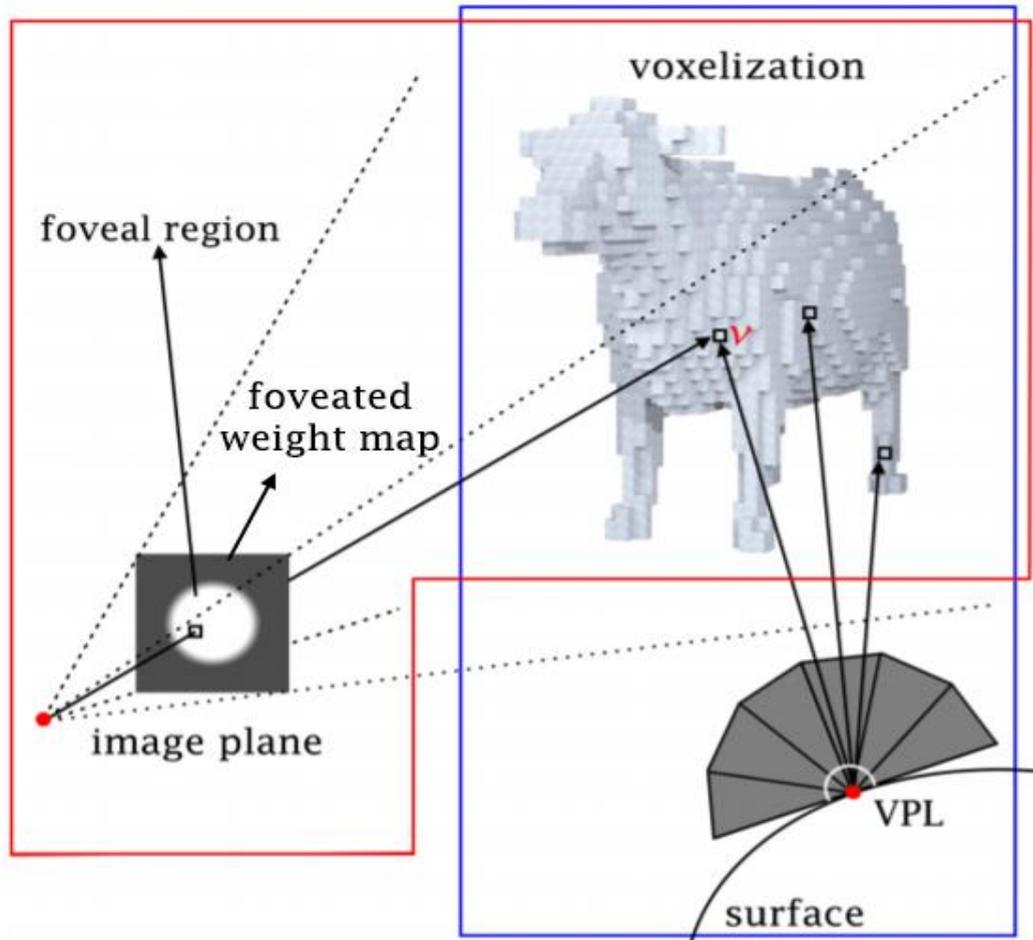
2. 生成VPL  
候选点

3. 凝视重要度  
计算

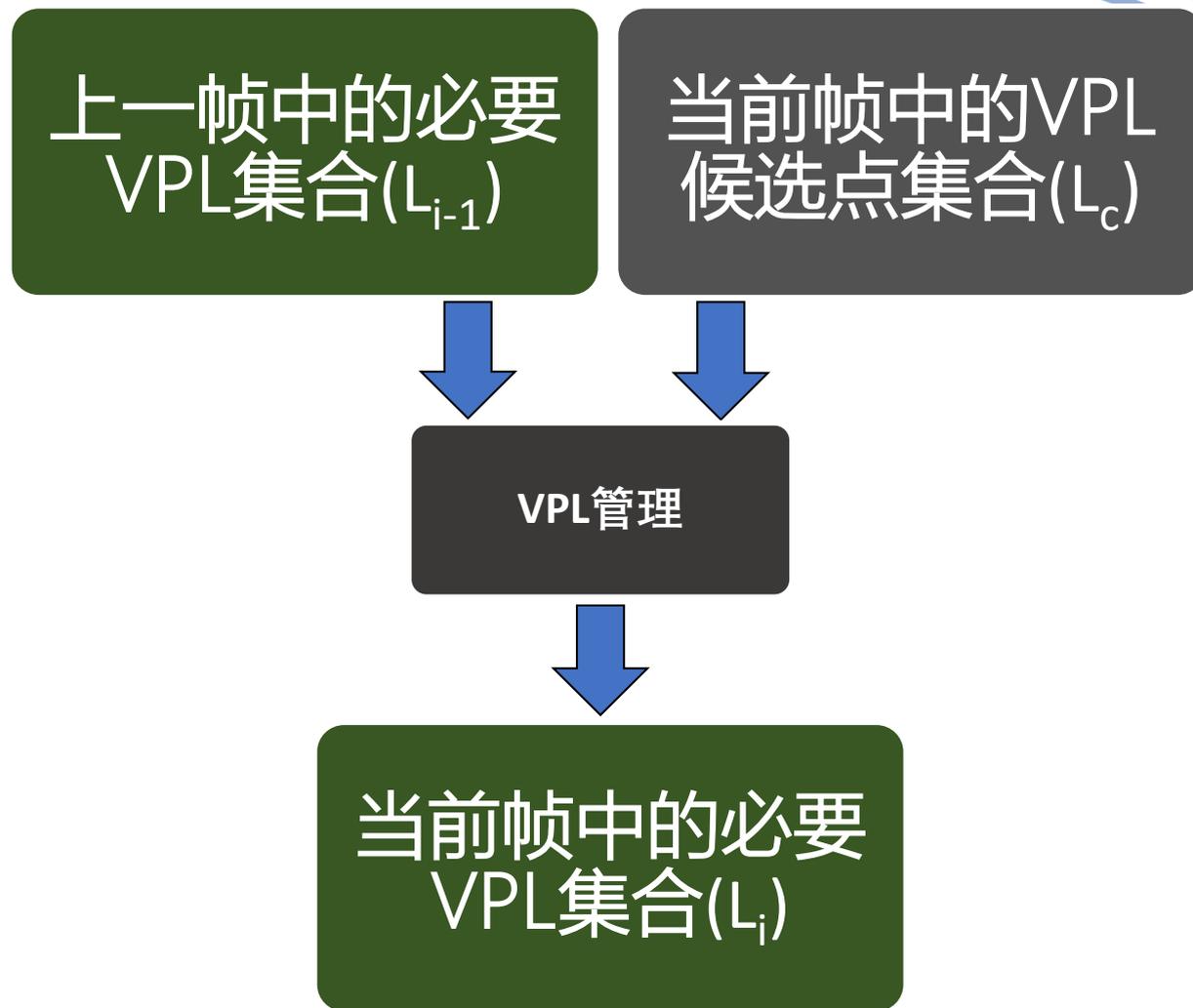
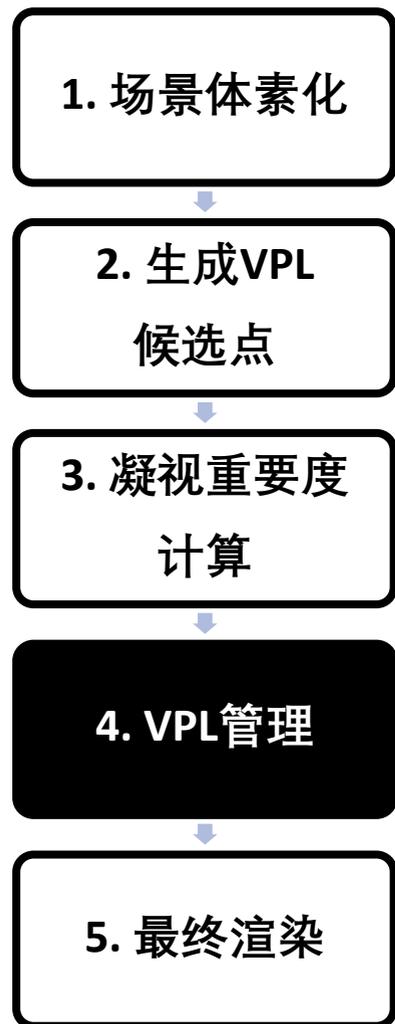
4. VPL管理

5. 最终渲染

## 3.2 凝视重要度计算



# VPL管理



# VPL管理

## 第 $i-1$ 帧中的必要VPL

1. 场景体素化

2. 生成VPL  
候选点

3. 凝视重要度  
计算

4. VPL管理

5. 最终渲染



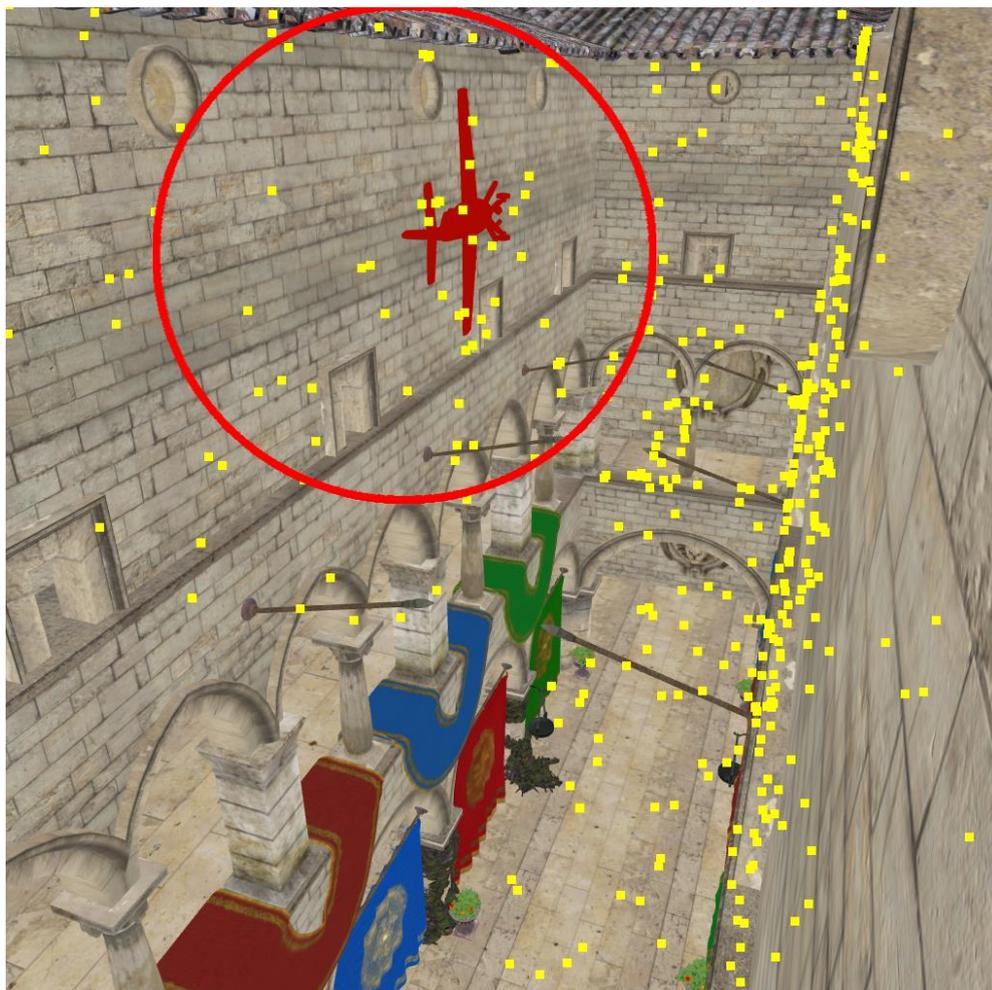
1. 场景体素化

2. 生成VPL  
候选点

3. 凝视重要度  
计算

4. VPL管理

5. 最终渲染



1. 场景体素化

2. 生成VPL  
候选点

3. 凝视重要度  
计算

4. VPL管理

5. 最终渲染



# VPL管理

1. 场景体素化

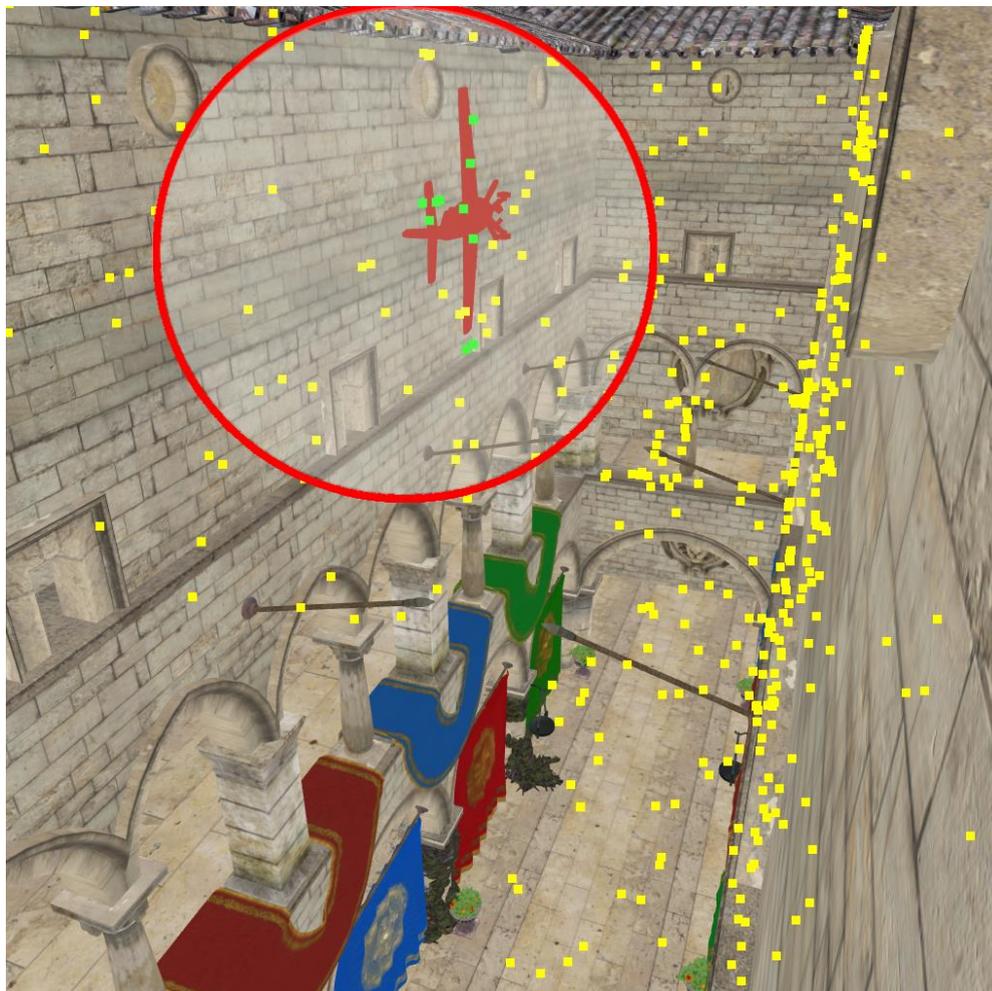
2. 生成VPL  
候选点

3. 凝视重要度  
计算

4. VPL管理

5. 最终渲染

## 4.2 重新计算第 $i-1$ 帧中的必要VPL的 凝视重要度



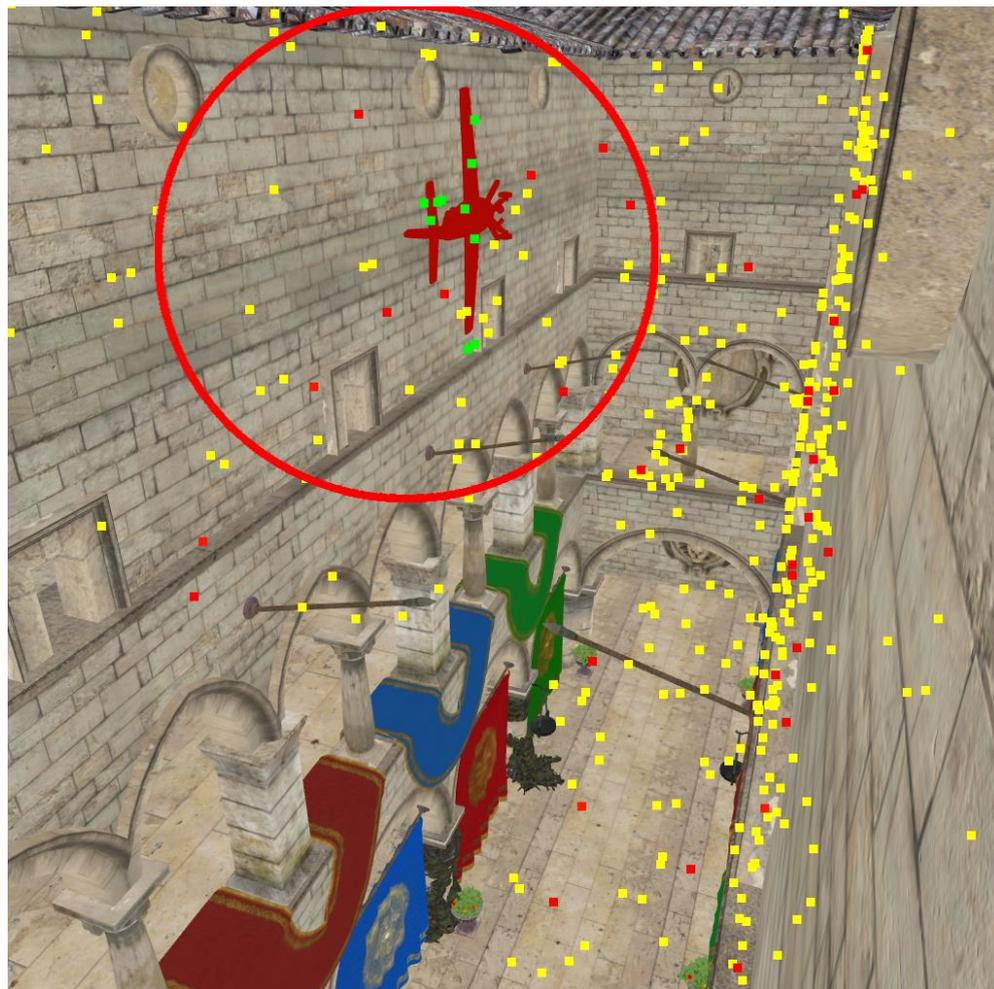
1. 场景体素化

2. 生成VPL  
候选点

3. 凝视重要度  
计算

4. VPL管理

5. 最终渲染



# VPL管理

## 第 $i$ 帧中的必要VPL

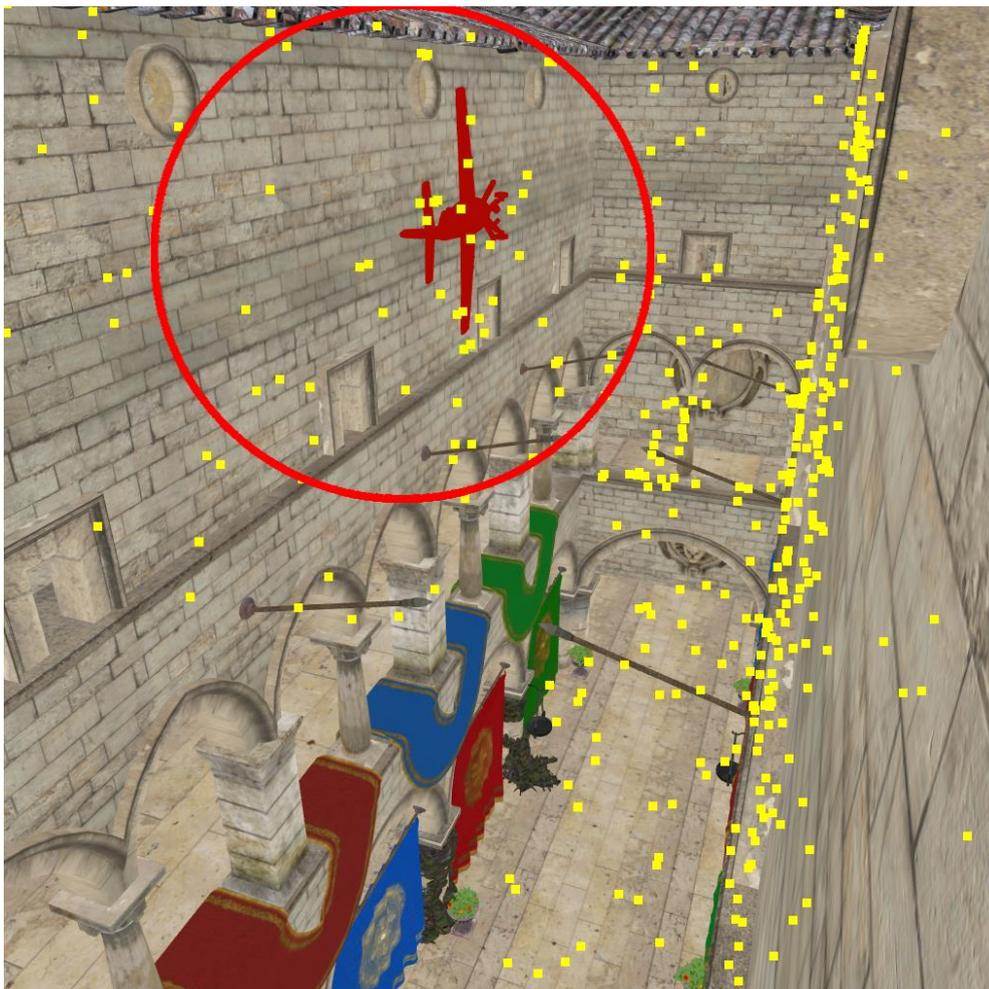
1. 场景体素化

2. 生成VPL  
候选点

3. 凝视重要度  
计算

4. VPL管理

5. 最终渲染



# 最终渲染

## 5.1 使用抛物面阴影图判断VPL可见性

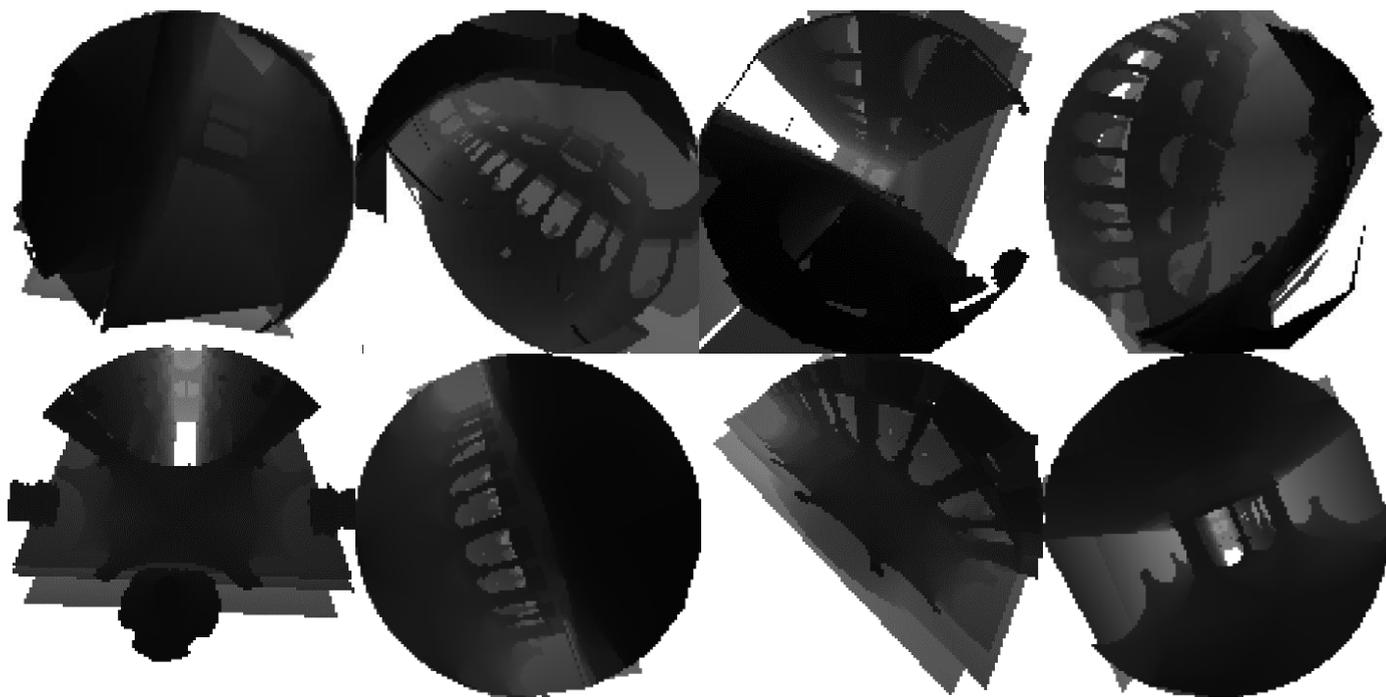
1. 场景体素化

2. 生成VPL  
候选点

3. 凝视重要度  
计算

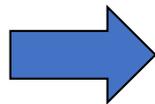
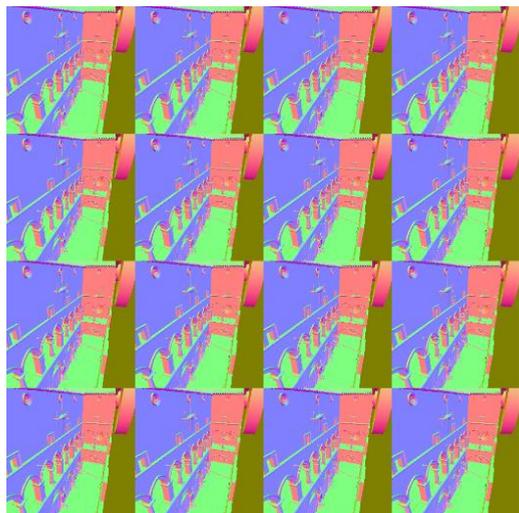
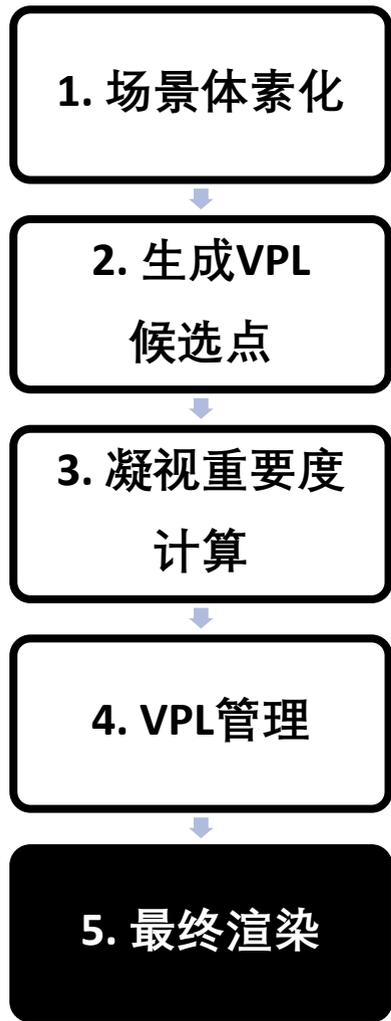
4. VPL管理

5. 最终渲染



# 最终渲染

## 5.2 基于间隔采样的VPL间接光照渲染



# 最终渲染

1. 场景体素化



2. 生成VPL  
候选点



3. 凝视重要度  
计算

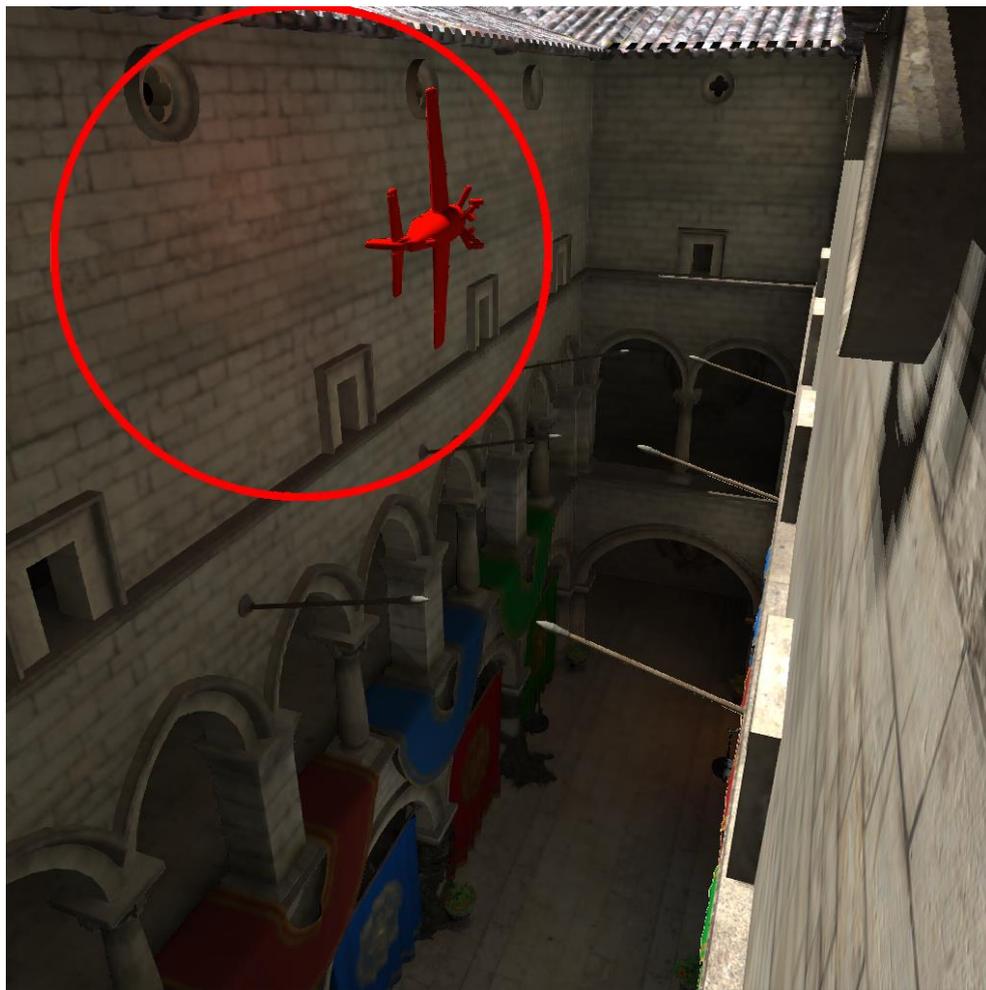


4. VPL管理

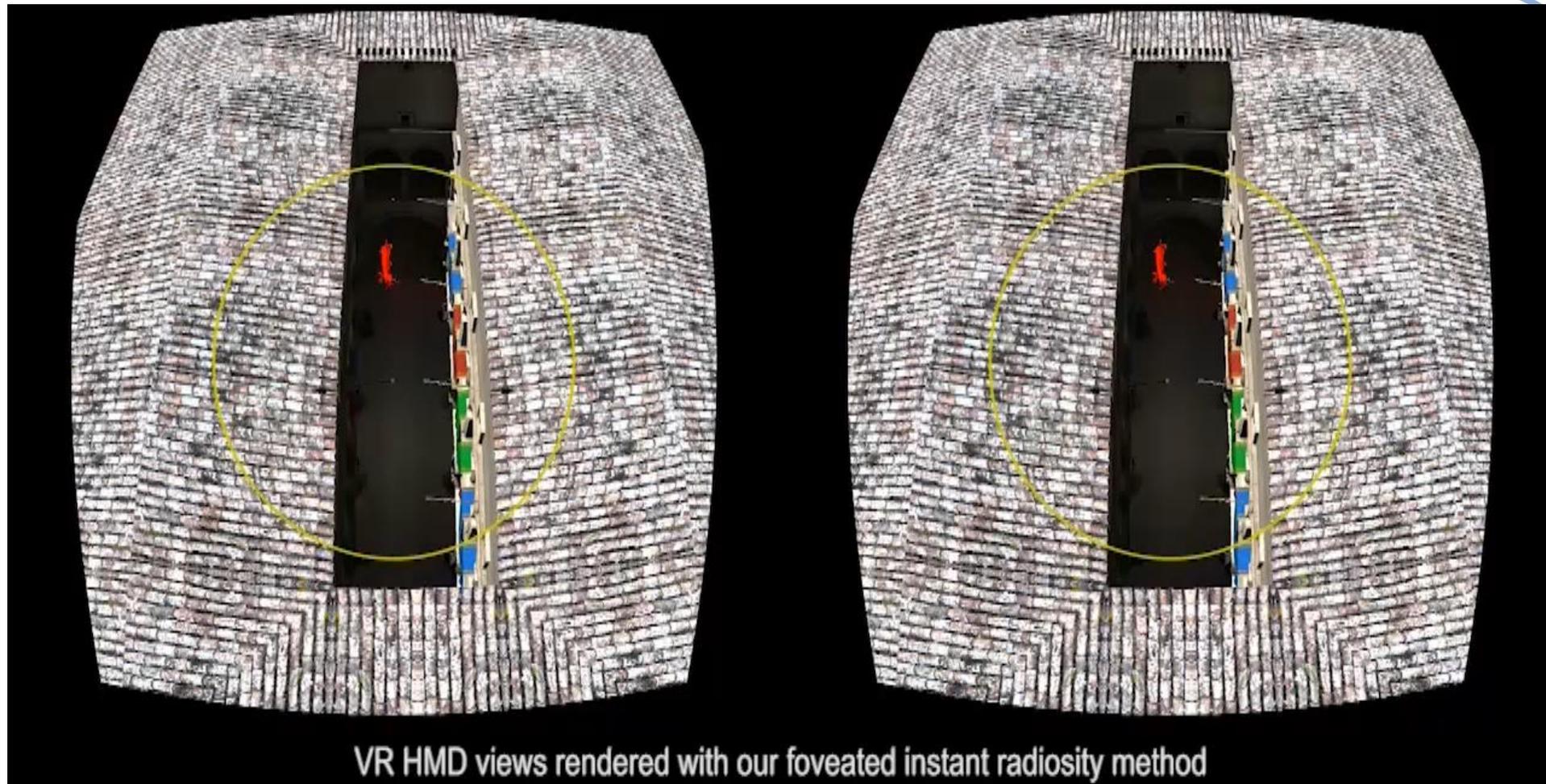


5. 最终渲染

## 5.3 结合间接光照与直接光照效果



# 结果与讨论



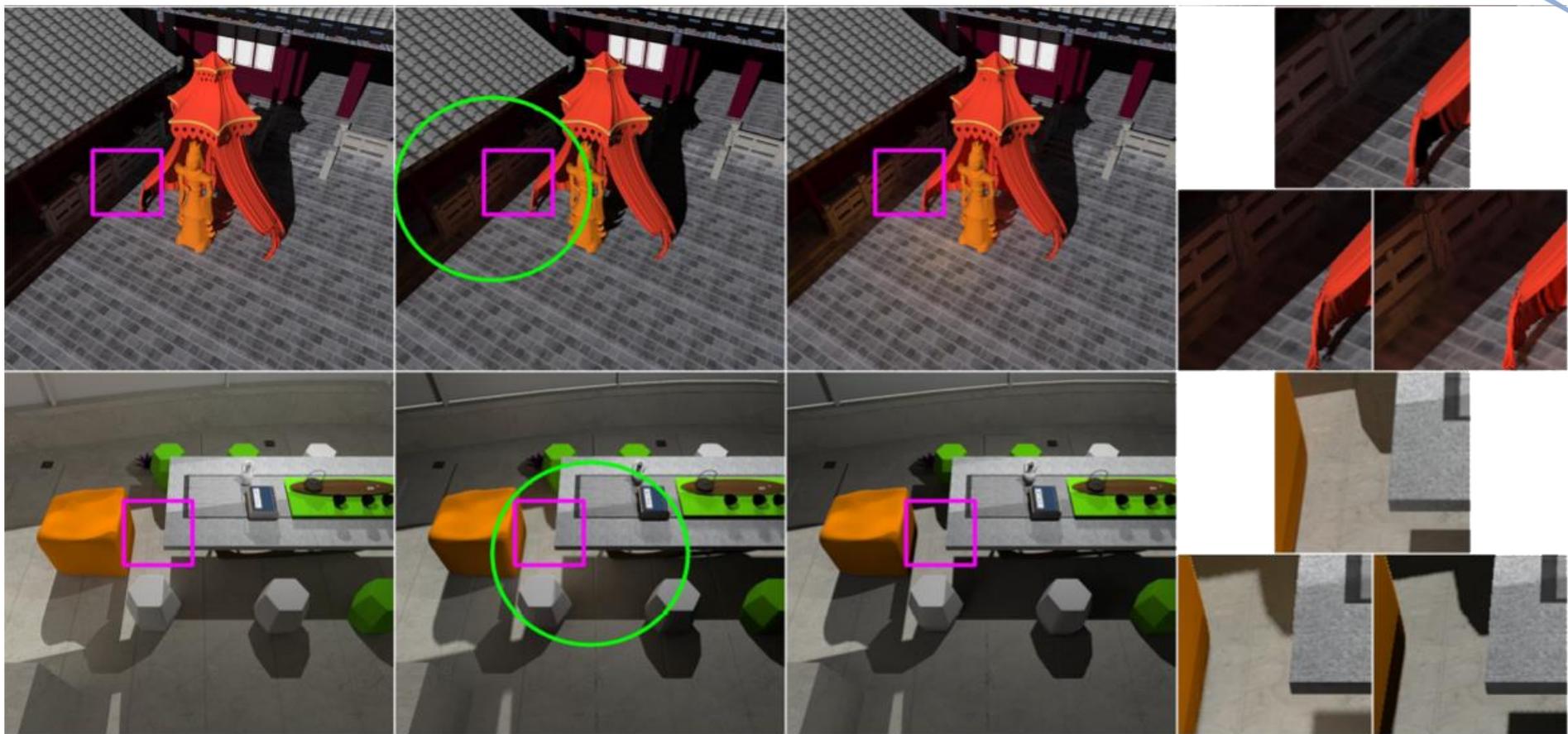
# 结果与讨论

我们的方法与立即辐射度方法均使用1000VPL



- 我们方法的渗色效果比立即辐射度更接近作为参考的路径追踪效果

# 结果与讨论



Path tracing

Our method

IR<sub>1000</sub>

Details comparison

# 结果与讨论

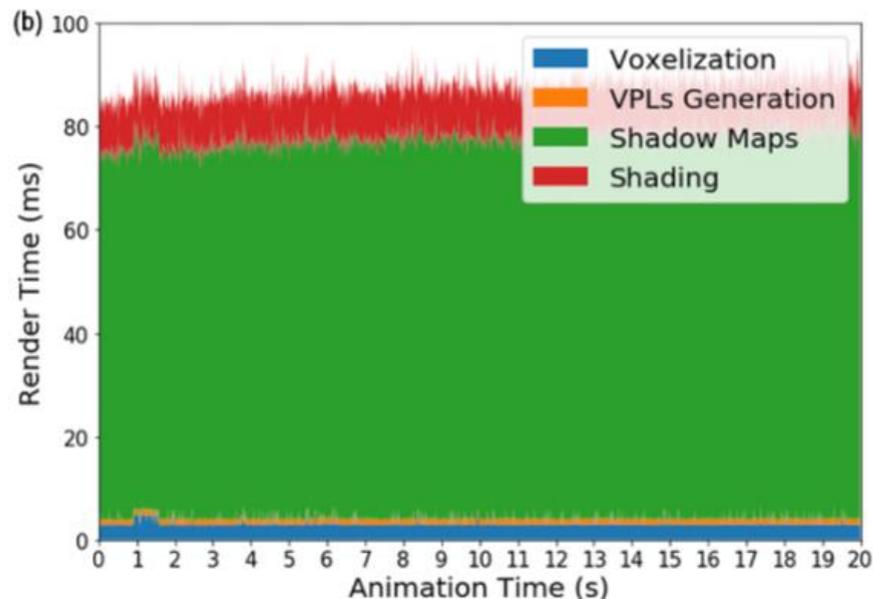
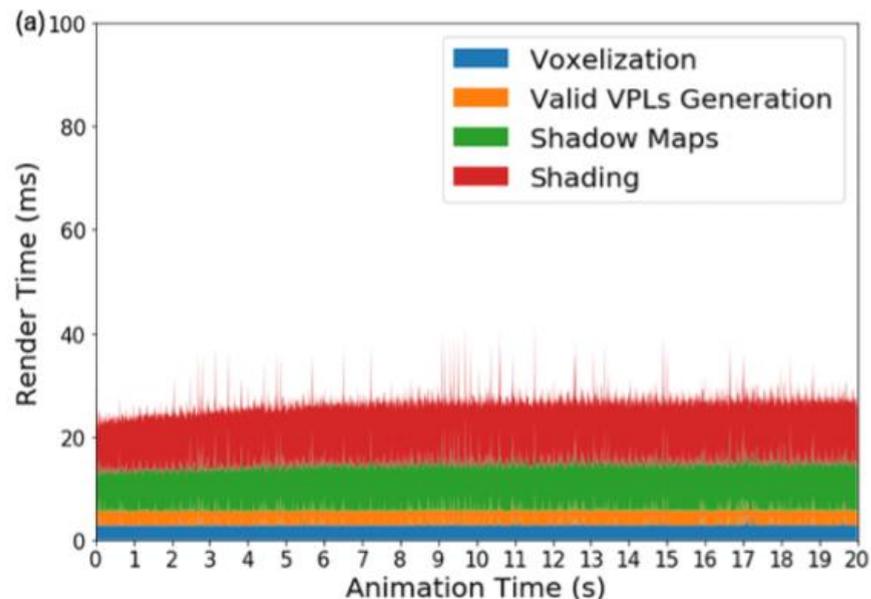
- Mean squared pixel error(MSE  $\times 10^{-3}$ )

Scene	MSE in Fovea		MSE in Periphery	
	IR <sub>1000</sub>	Ours	IR <sub>1000</sub>	Ours
Sponza	3.16	1.09	4.84	3.89
Cornellbox	1.16	0.74	1.59	2.07
Room	2.19	1.08	3.18	1.17
Yard	1.75	1.01	1.93	2.46
Balcony	8.61	4.30	7.43	5.90

- 与立即辐射度方法相比，我们的方法在凝视区域内取得了较小的误差
- 我们的方法中凝视区域内的误差总小于凝视区域外的误差

# 结果与讨论

- 性能



## 我们的方法

IR<sub>1000</sub>

- 我们的方法与立即辐射度方法渲染sponza场景的逐帧时间开销对比
- 可以节省80%左右的阴影图渲染开销。

# 结果与讨论

## • 性能

Scene	Sponza	Cornellbox	Room	Yard	Balcony
OurS	24ms	13ms	26ms	28ms	27ms
vs. IR <sub>1000</sub>	3×	3×	3×	3×	4×
vs. IR <sub>eq</sub>	12×	11×	13×	13×	10×
VPL Num. in IR <sub>eq</sub>	2700	2700	3000	2400	2400

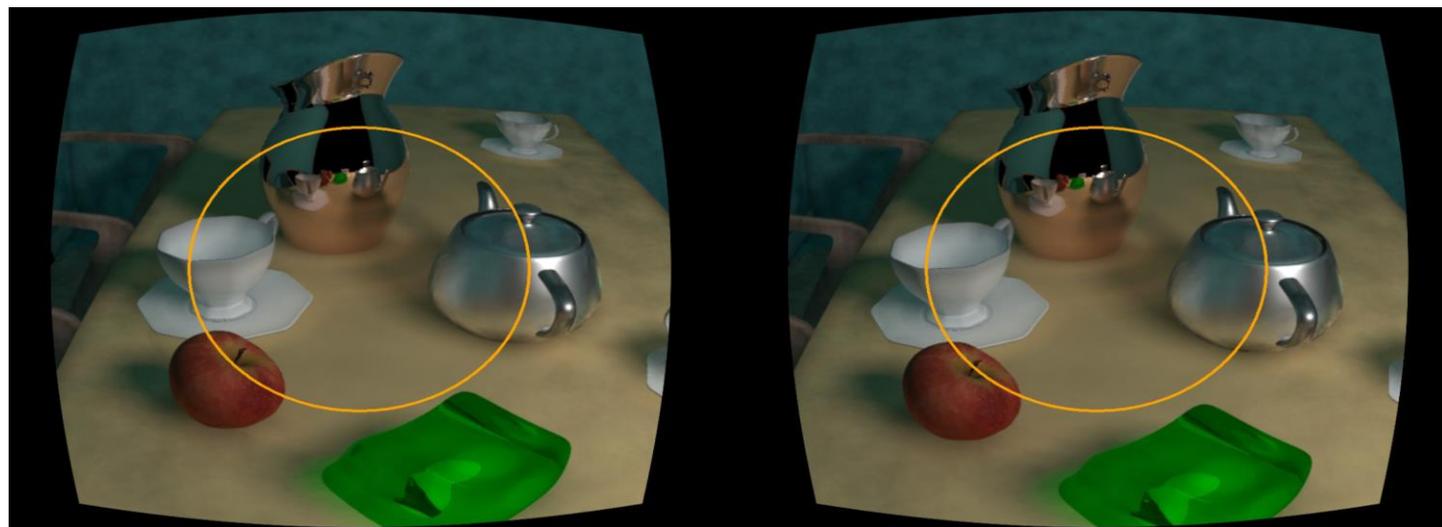
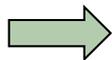
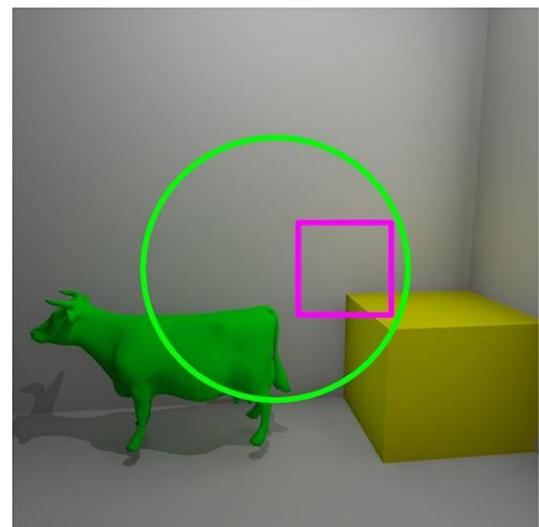
- 相比于1000VPL的立即辐射度方法，我们方法已经获得了3-4倍的提速。
- 相比于同质量的立即辐射度方法，我们方法达到了10-13倍的提速。

# Foveated Photon Mapping

基于凝视点的光子映射方法

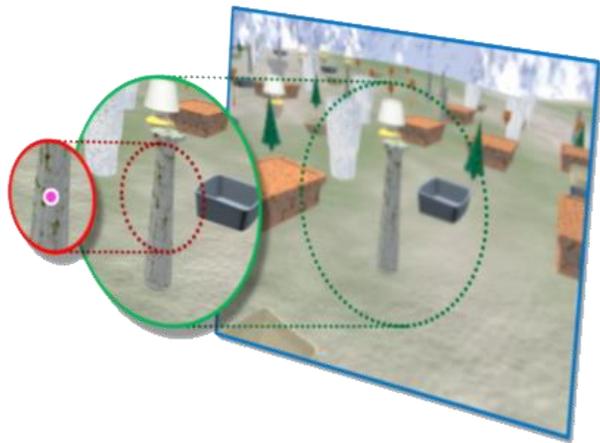
# 动机

- 为HMD实时渲染具有各种材质（镜面、金属、透明材质等）的复杂场景



# 相关工作

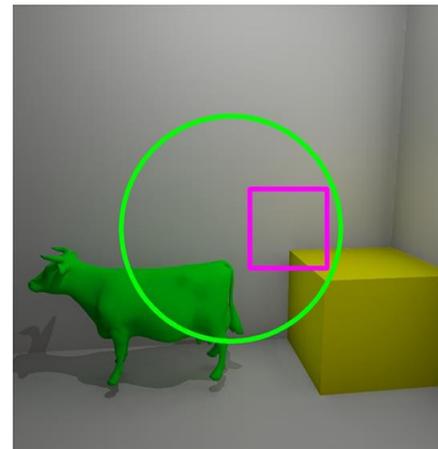
- 凝视点渲染



[Guenter 2012 TOG]



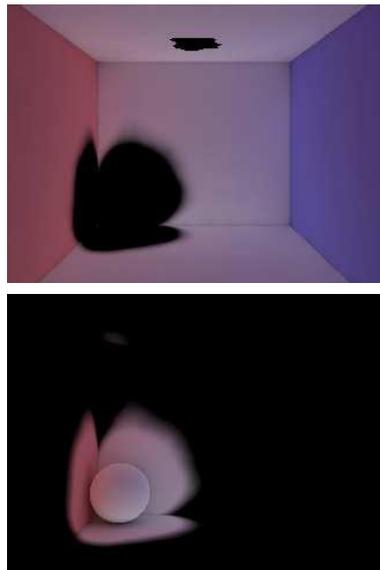
[Weier 2016 CGF]



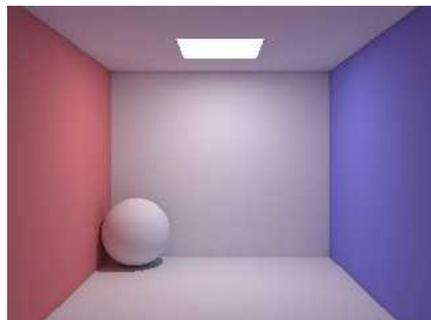
[Wang 2020 ISMAR]

# 相关工作

- 光子映射



[Tawara 2002 VMV]



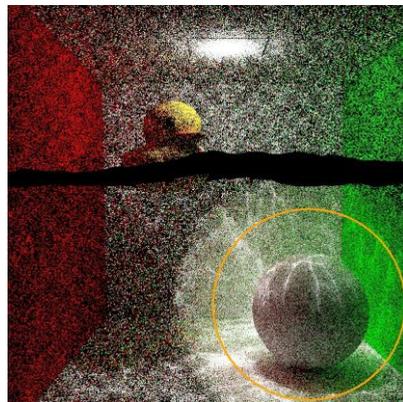
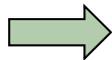
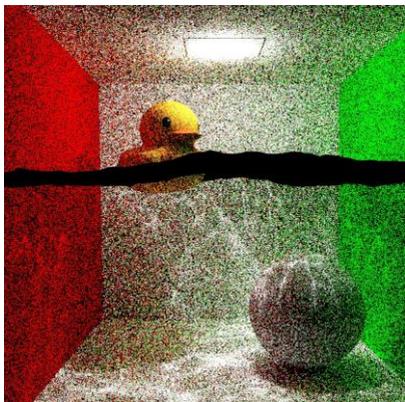
[McGuire 2009 HPG]



[Smal 2019 RayTracingGems]

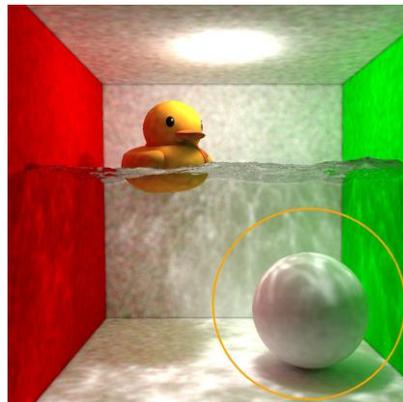
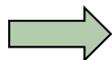
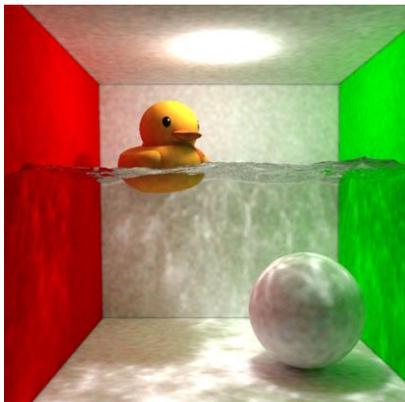
# 挑战

- 在凝视点区域生成高密度光子以渲染高质量图像



- 提升光子追踪的性能以满足实时性需求

10.0 FPS



35.1 FPS

# 整体流程

1. 光源光子追踪



2. 帧间光子管理



3. 凝视点光子追踪



4. Radiance计算

# 光源光子追踪

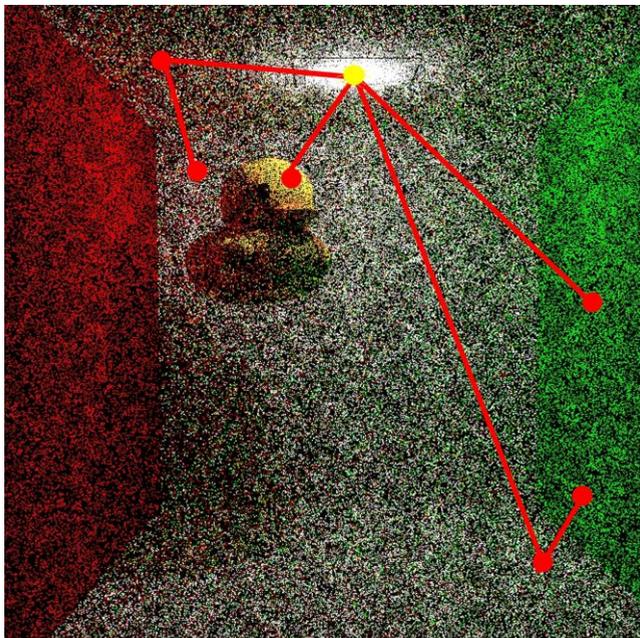
- 从光源发射并追踪光子

1. 光源光子追踪

2. 帧间光子管理

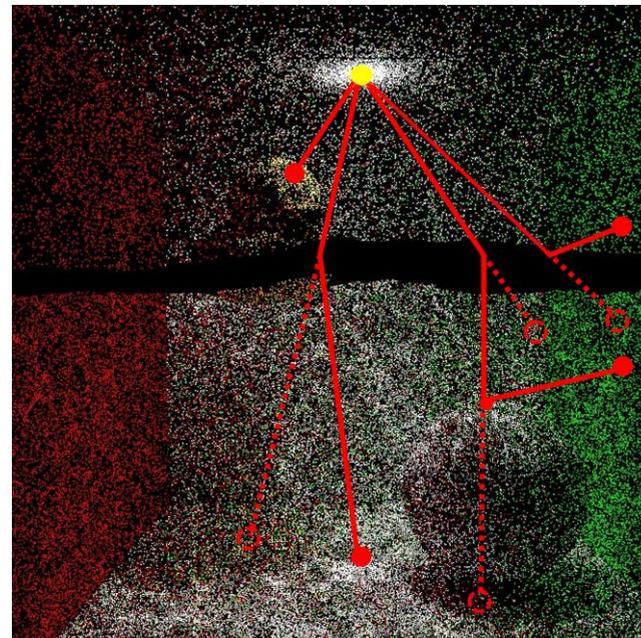
3. 凝视点光子追踪

4. Radiance计算



● Light Source ● Positive Light Photon

a) 静态光源光子追踪



● Light Source ● Positive Light Photon ○ Negative Light Photon

b) 动态光源光子追踪

光源光子追踪 [Tawara et al. 2002]

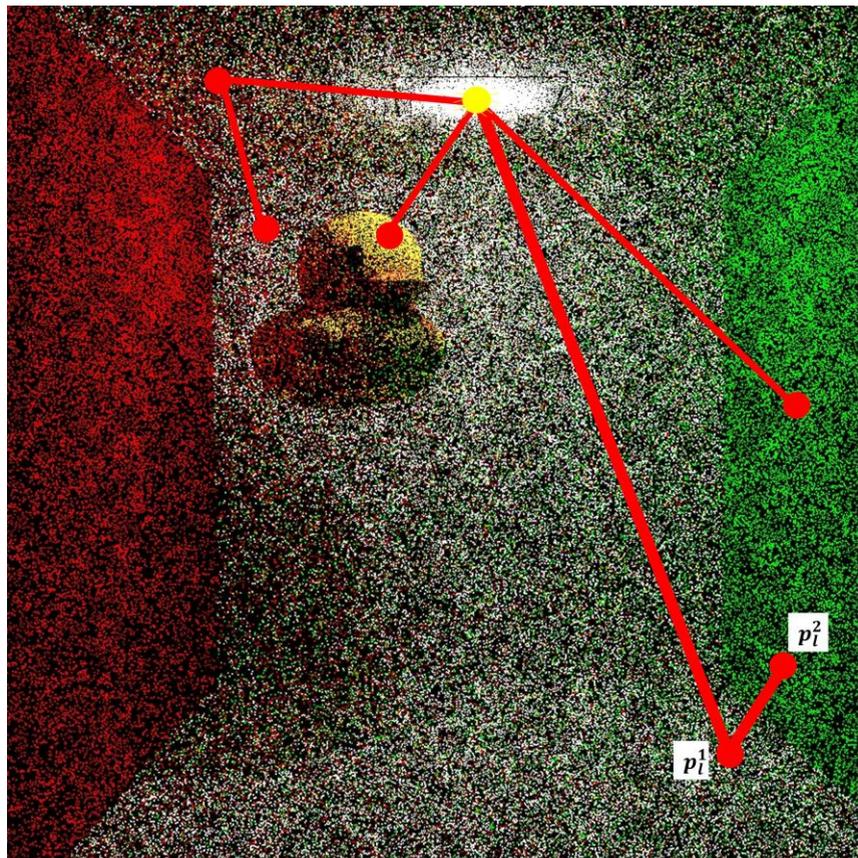
# 光源光子追踪

1. 光源光子追踪

2. 帧间光子管理

3. 凝视点光子追踪

4. Radiance计算



☀ Light Source      ● Positive Light Photon

a) 静态光源光子追踪

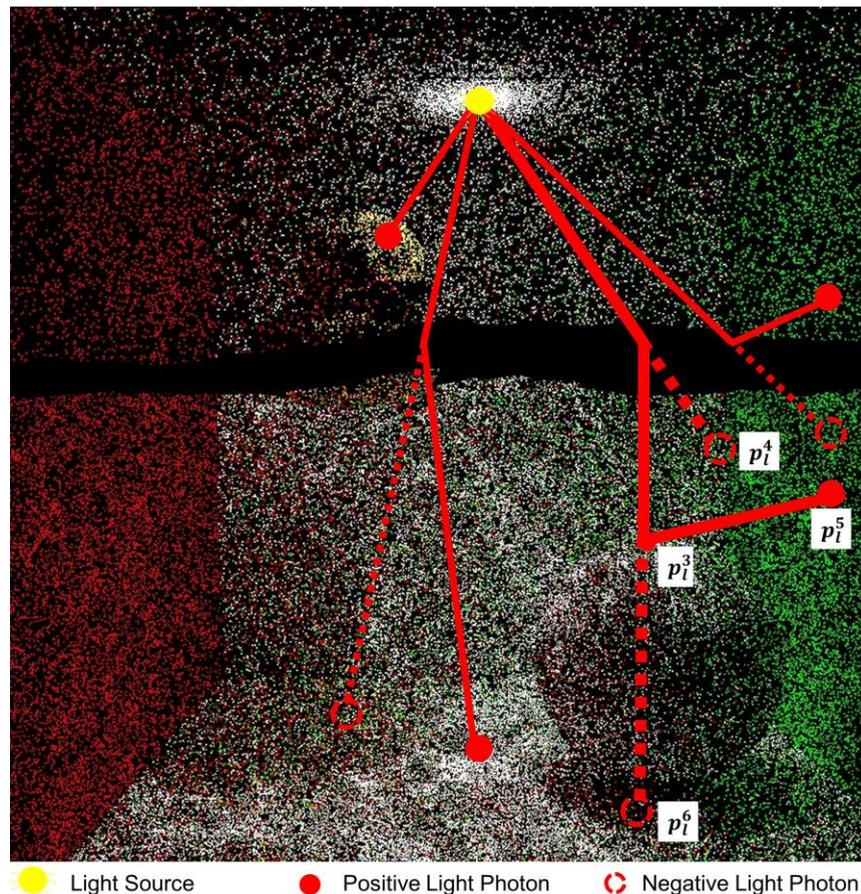
# 光源光子追踪

1. 光源光子追踪

2. 帧间光子管理

3. 凝视点光子追踪

4. Radiance计算



b) 动态光源光子追踪

# 帧间光子管理

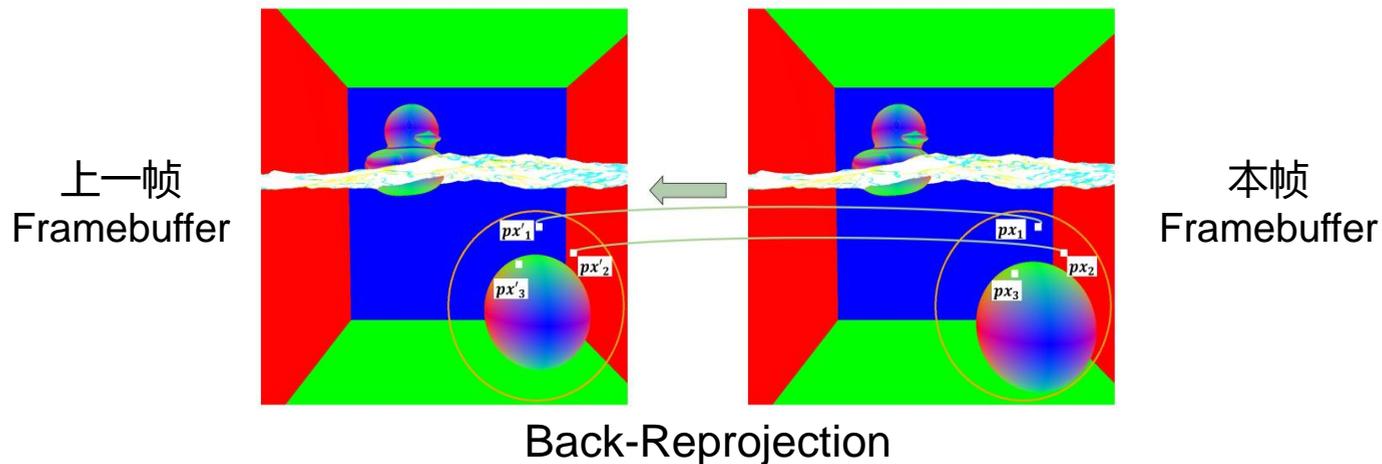
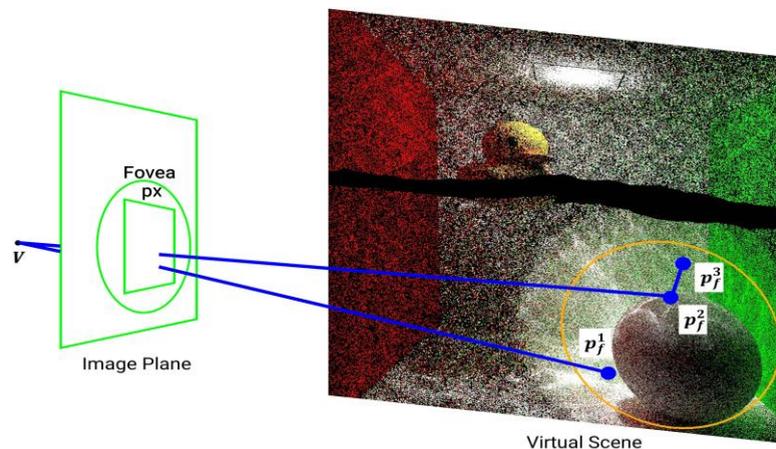
- 保留上一帧中有效的凝视点光子

1. 光源光子追踪

2. 帧间光子管理

3. 凝视点光子追踪

4. Radiance计算



# 帧间光子管理

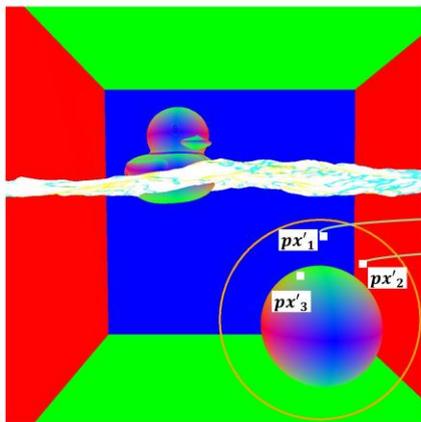
1. 光源光子追踪

2. 帧间光子管理

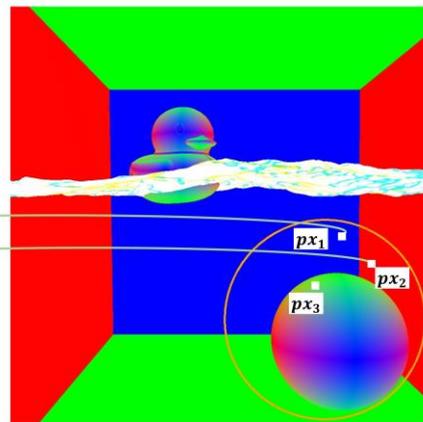
3. 凝视点光子追踪

4. Radiance计算

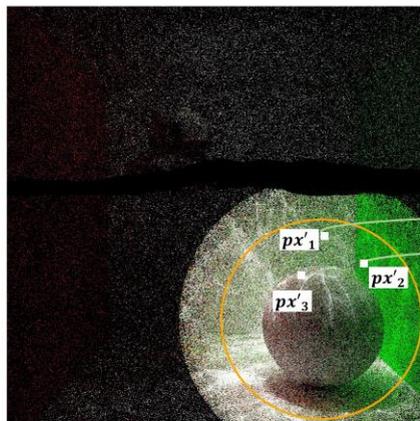
上一帧  
Framebuffer



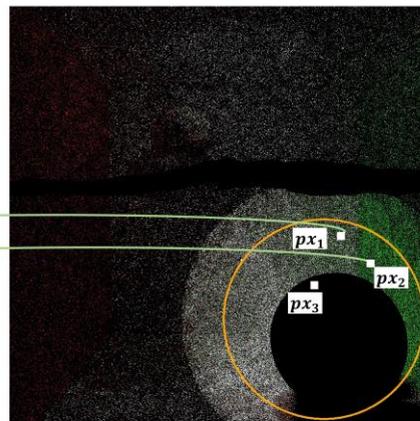
本帧  
Framebuffer



上一帧的凝  
视点光子



保留下来供本  
帧继续使用的  
凝视点光子



# 凝视点光子追踪

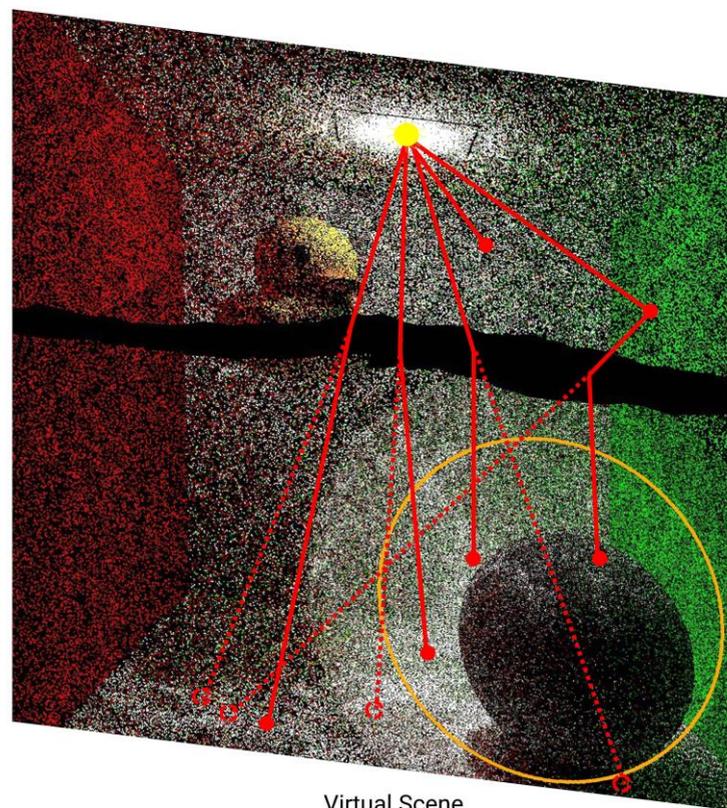
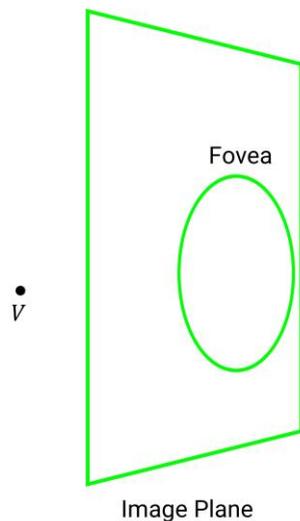
- 在凝视点区域产生高密度光子

1. 光源光子追踪

2. 帧间光子管理

3. 凝视点光子追踪

4. Radiance计算



● Light Source   ● Positive Light Photon   ● Negative Light Photon   ● Valid Foveated Photon   ● Invalid Foveated Photon

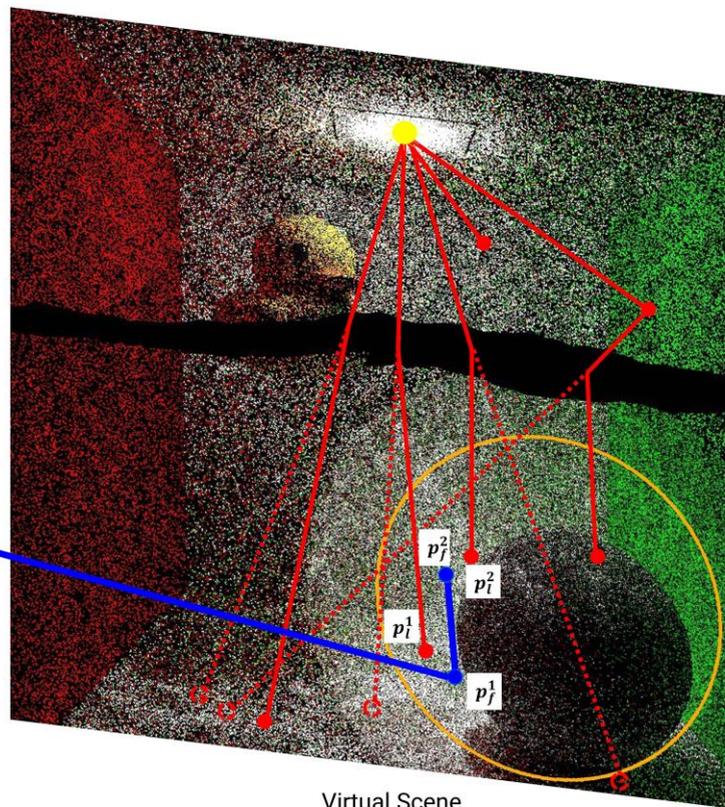
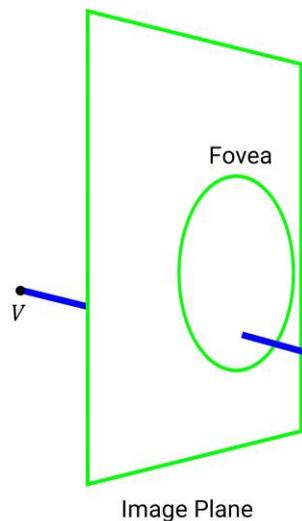
# 凝视点光子追踪

1. 光源光子追踪

2. 帧间光子管理

3. 凝视点光子追踪

4. Radiance计算



● Light Source    ● Positive Light Photon    ○ Negative Light Photon    ● Valid Foveated Photon    ○ Invalid Foveated Photon

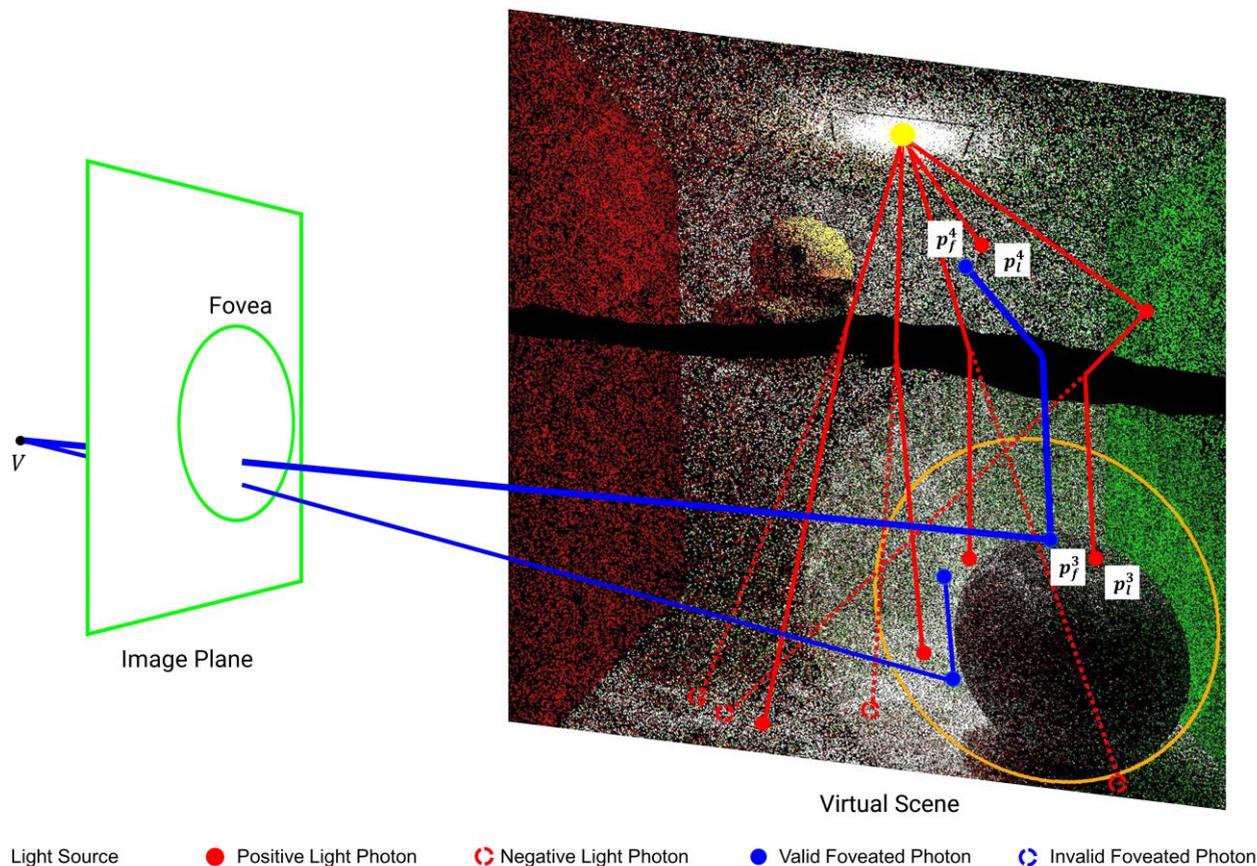
# 凝视点光子追踪

1. 光源光子追踪

2. 帧间光子管理

3. 凝视点光子追踪

4. Radiance计算



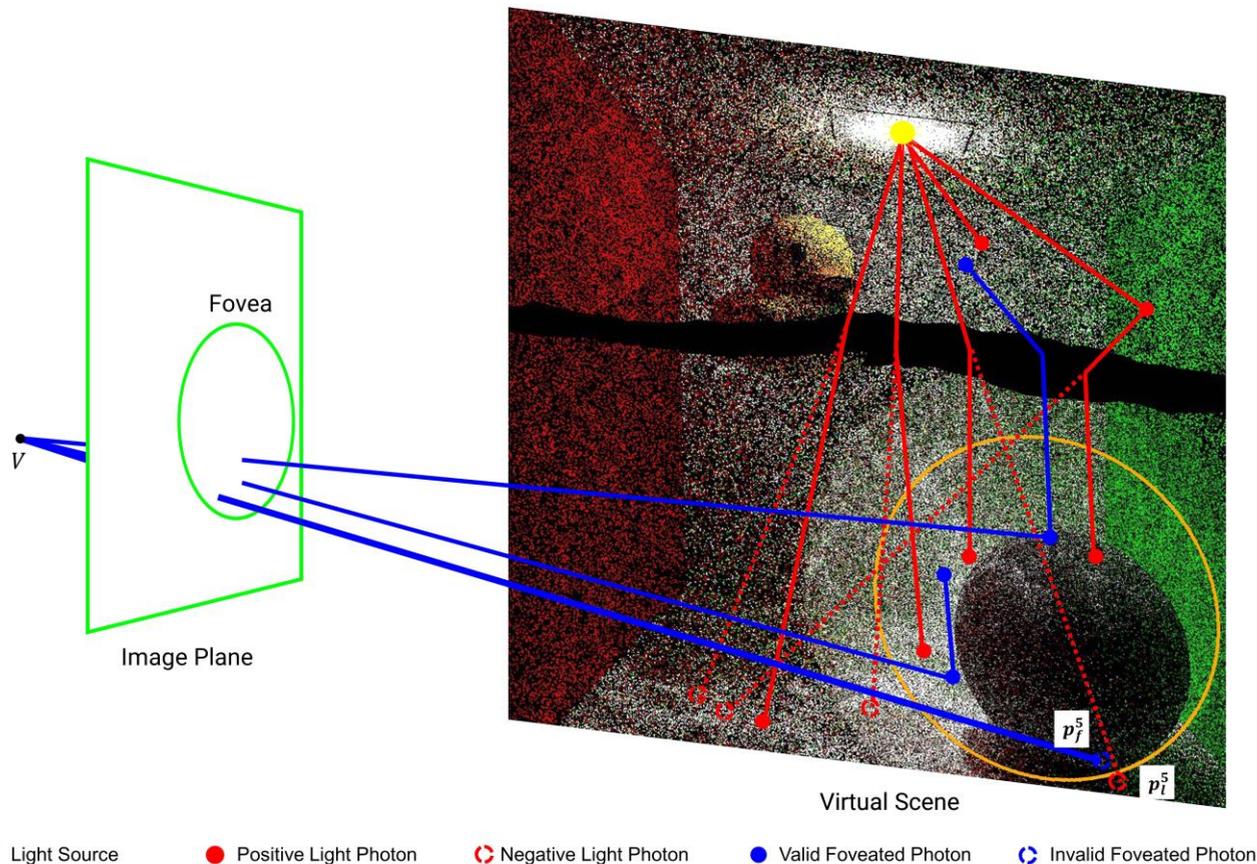
# 凝视点光子追踪

1. 光源光子追踪

2. 帧间光子管理

3. 凝视点光子追踪

4. Radiance计算



# Radiance计算

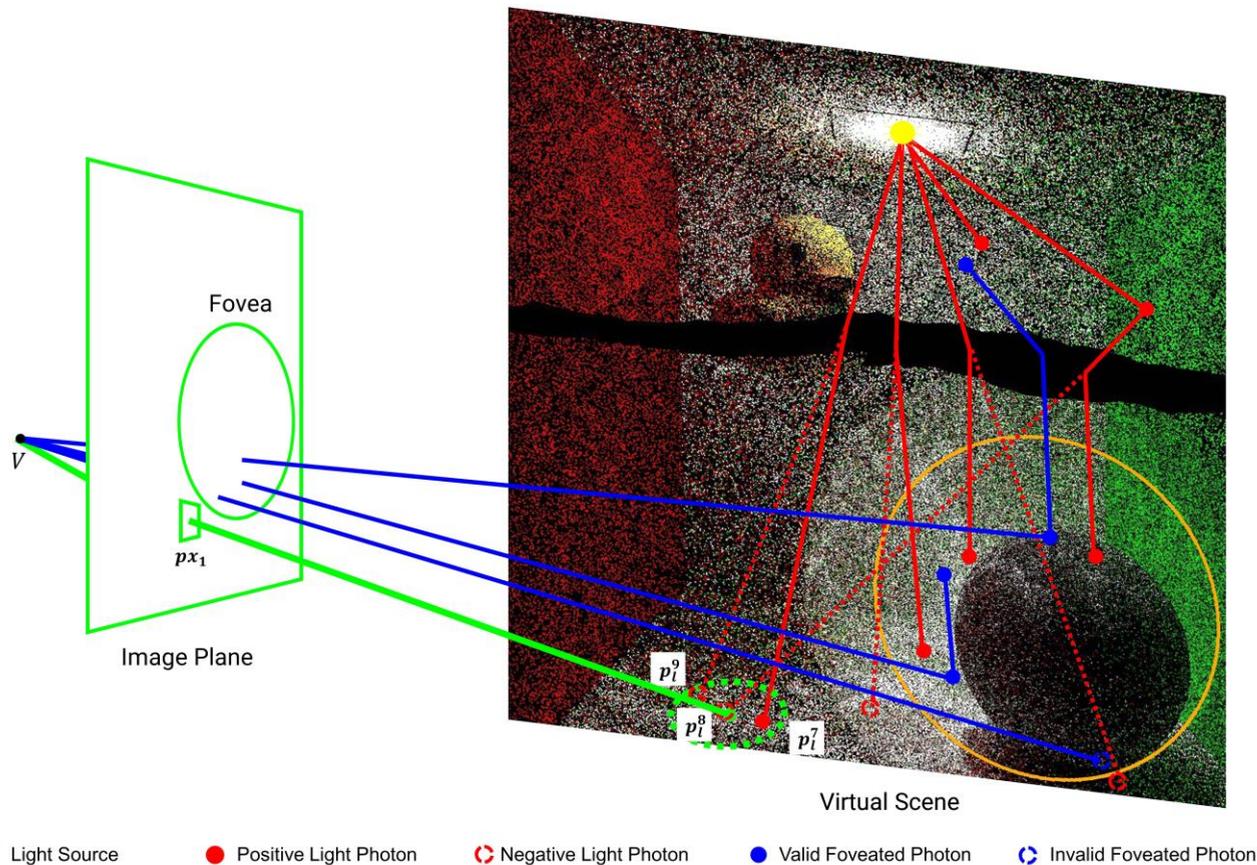
- 估计当前帧中每个像素的Radiance

1. 光源光子追踪

2. 帧间光子管理

3. 凝视点光子追踪

4. Radiance计算



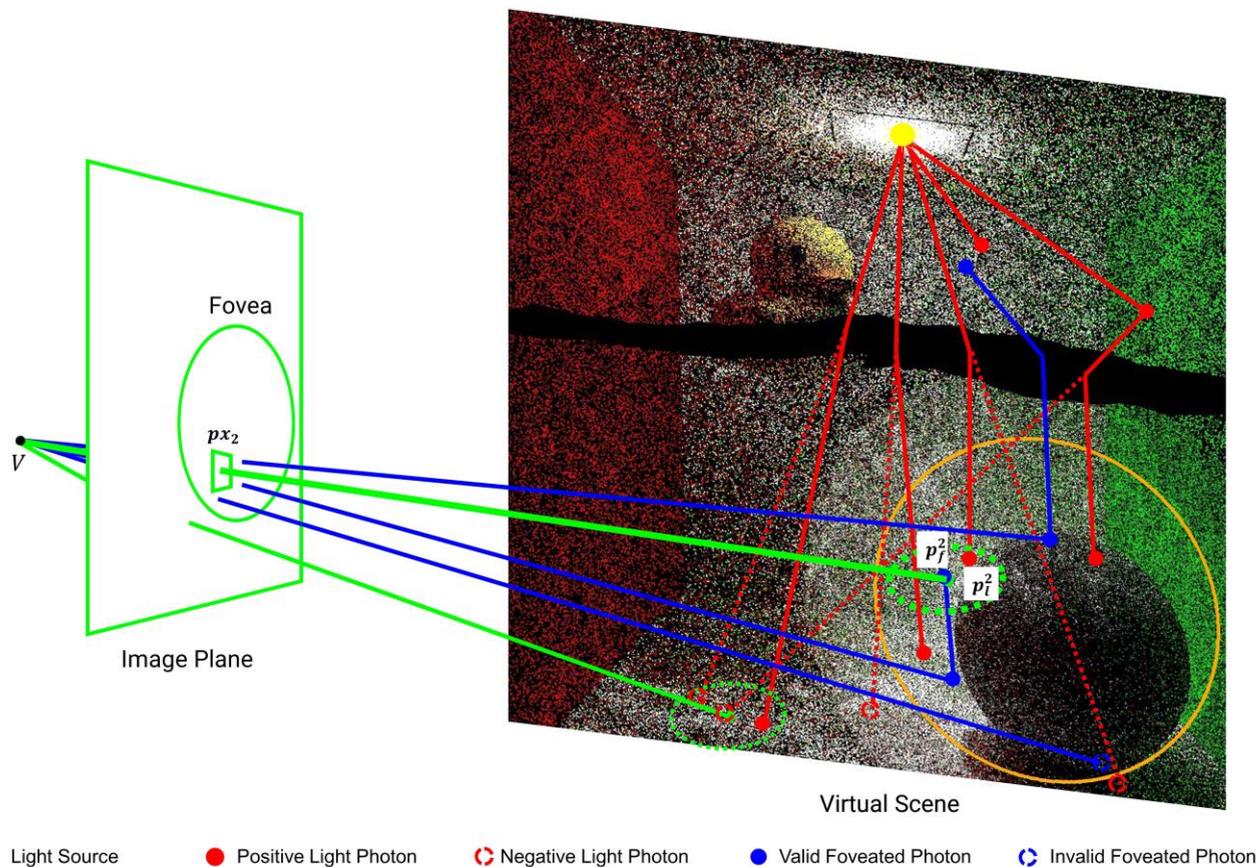
# Radiance计算

1. 光源光子追踪

2. 帧间光子管理

3. 凝视点光子追踪

4. Radiance计算



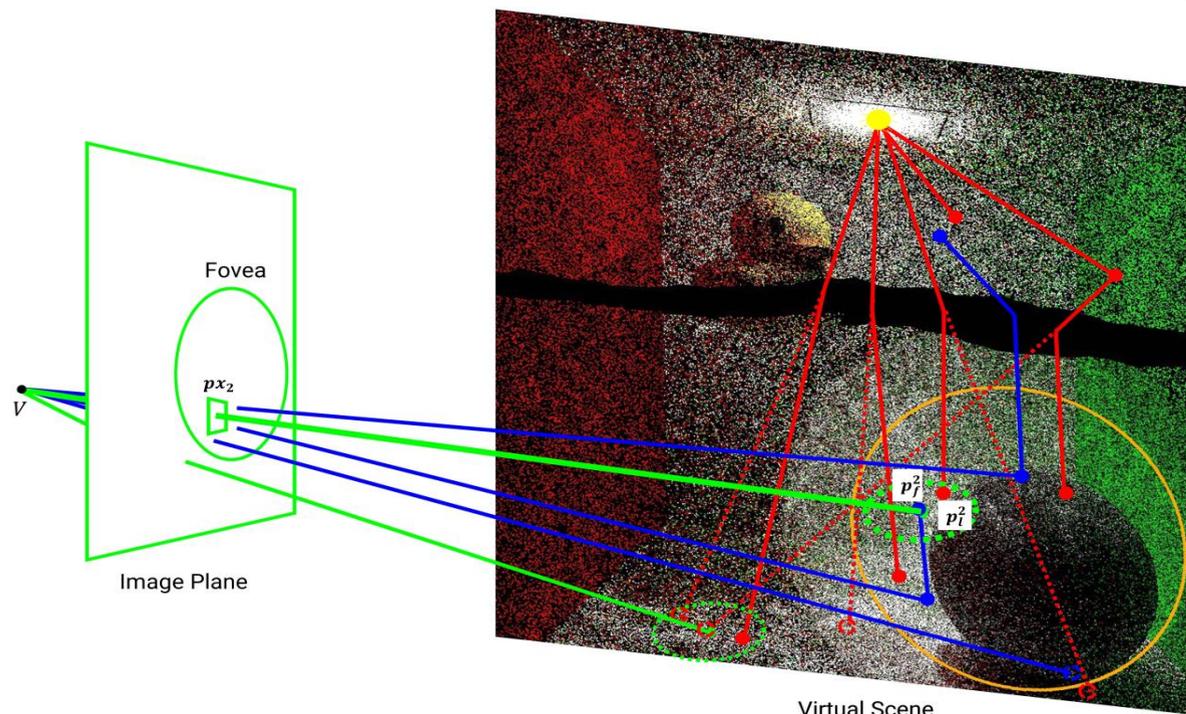
# Radiance计算

1. 光源光子追踪

2. 帧间光子管理

3. 凝视点光子追踪

4. Radiance计算



● Light Source

● Positive Light Photon

⊗ Negative Light Photon

● Valid Foveated Photon

⊗ Invalid Foveated Photon

Radiance Estimation:

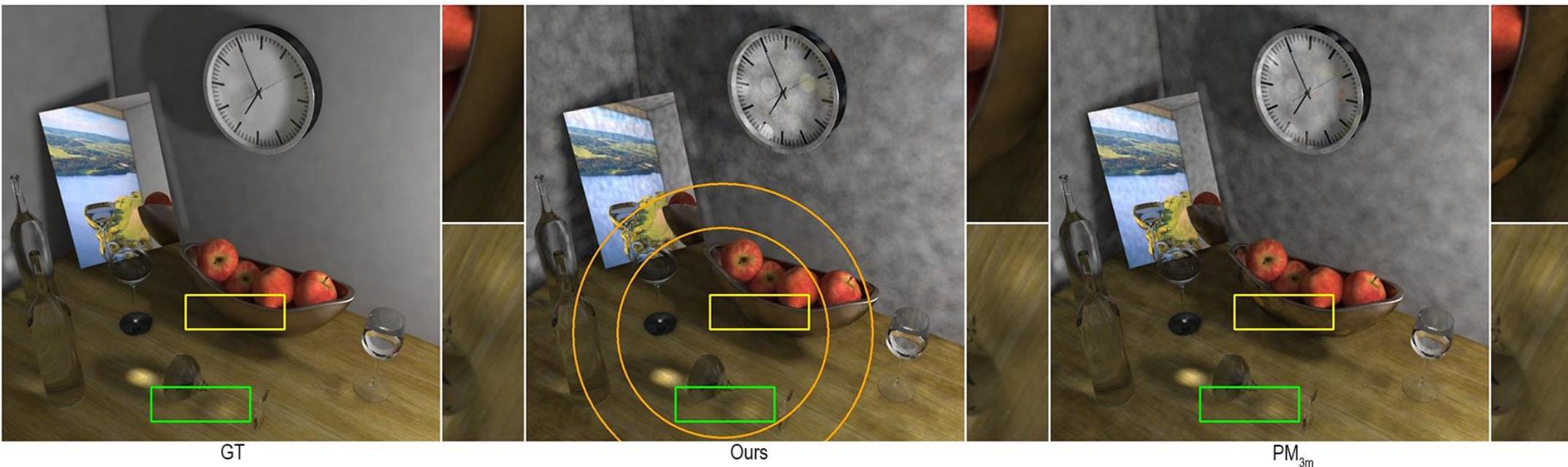
$$g(x, M, \eta) = \frac{\eta}{\pi\gamma^2} \sum_{p \in M} f_r(x, \omega_o, \omega_p) \varphi_p(x, \omega_p)$$

Energy-normalized Parameter:

$$\eta = \frac{\text{sumFlux}(M_{ls,ld}, r_t)}{\text{sumFlux}(M_{ls,ld,fs,fd}, r_t)}$$

# 结果 & 讨论

GT: 使用1亿光子的光子映射方法;  $PM_{3m}$ : 使用300万光子的光子映射方法; Ours: 使用300万光子的我们方法。



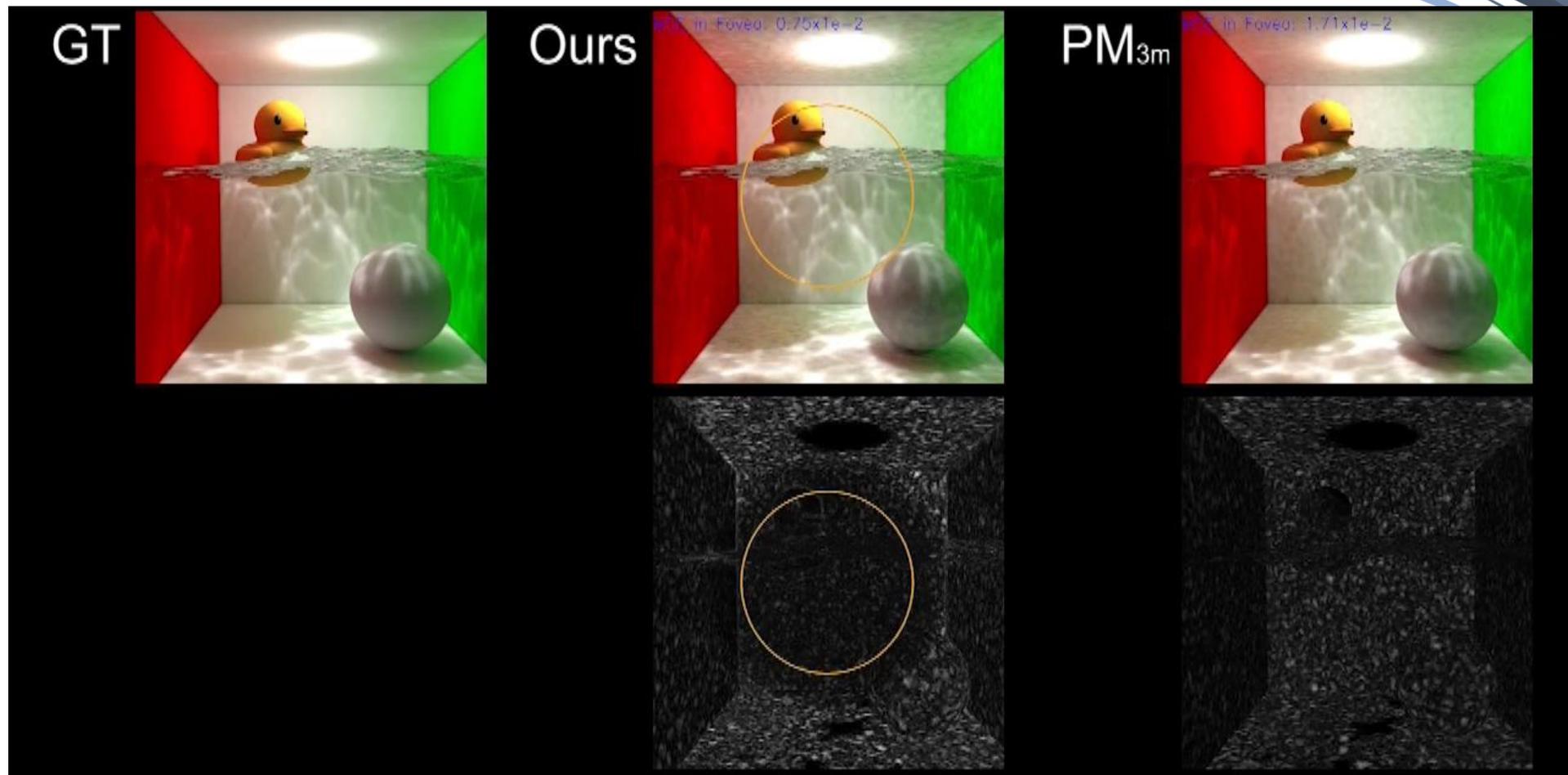
- 相比于 $PM_{3m}$  , Ours更接近于GT。

# 结果 & 讨论



在Room场景中， $PM_{3m}$  的MSE是Ours的2.95倍， $PM_{3m}$ 的帧间MSE是Ours的1.43倍。

# 结果 & 讨论



在Cornellbox场景中， $PM_{3m}$ 的MSE是Ours的3.70倍， $PM_{3m}$ 的帧间MSE是Ours的1.41倍。

# 结果 & 讨论

## • 性能

Ours与光子映射方法的性能比较,  $PM_{3m}$  为使用300万光子的光子映射方法,  $PM_{eq}$  为凝视点区域渲染质量与Ours相当的光子映射方法。

Scene	Sponza	Study	Cornellbox	Room	Kitchen
Ours	33.1ms	25.1ms	28.5ms	29.6ms	23.6ms
<i>Avg. Reu.</i>	83%	78%	71%	76%	78%
vs. $PM_{3m}$	4.5×	4.3×	3.5×	4.1×	4.1×
vs. $PM_{eq}$	9.1×	10.1×	9.0×	10.0×	12.0×
#Photon	8.1m	9.6m	9.9m	9.5m	11.5m

# 结果 & 讨论

## • 性能

Ours与光子映射方法的性能比较,  $PM_{3m}$  为使用300万光子的光子映射方法,  $PM_{eq}$  为凝视点区域渲染质量与Ours相当的光子映射方法。

Scene	Sponza	Study	Cornellbox	Room	Kitchen
Ours	33.1ms	25.1ms	28.5ms	29.6ms	23.6ms
<i>Avg. Reu.</i>	83%	78%	71%	76%	78%
vs. $PM_{3m}$	4.5×	4.3×	3.5×	4.1×	4.1×
vs. $PM_{eq}$	9.1×	10.1×	9.0×	10.0×	12.0×
#Photon	8.1m	9.6m	9.9m	9.5m	11.5m

# 结果 & 讨论

## • 性能

Ours与光子映射方法的性能比较,  $PM_{3m}$  为使用300万光子的光子映射方法,  $PM_{eq}$  为凝视点区域渲染质量与Ours相当的光子映射方法。

Scene	Sponza	Study	Cornellbox	Room	Kitchen
Ours	33.1ms	25.1ms	28.5ms	29.6ms	23.6ms
<i>Avg. Reu.</i>	83%	78%	71%	76%	78%
vs. $PM_{3m}$	4.5×	4.3×	3.5×	4.1×	4.1×
vs. $PM_{eq}$	9.1×	10.1×	9.0×	10.0×	12.0×
#Photon	8.1m	9.6m	9.9m	9.5m	11.5m

# 结果 & 讨论

## • 性能

Ours与光子映射方法的性能比较,  $PM_{3m}$  为使用300万光子的光子映射方法,  $PM_{eq}$  为凝视点区域渲染质量与Ours相当的光子映射方法。

Scene	Sponza	Study	Cornellbox	Room	Kitchen
Ours	33.1ms	25.1ms	28.5ms	29.6ms	23.6ms
<i>Avg. Reu.</i>	83%	78%	71%	76%	78%
vs. $PM_{3m}$	4.5×	4.3×	3.5×	4.1×	4.1×
vs. $PM_{eq}$	9.1×	10.1×	9.0×	10.0×	12.0×
#Photon	8.1m	9.6m	9.9m	9.5m	11.5m

# 结论

- 提出了一种基于凝视点的光子映射方法，可以在具有多材质的动态场景中实时渲染凝视点区域的高质量全局光照；
- 提出了一种凝视点区域光子追踪方法，用于在凝视点区域分布高密度光子，且不会造成能量的失衡；
- 提出了一种帧间光子管理算法，用于选择上一帧中有效的凝视点光子并为当前帧追踪新的光子。

# 限制

- 无法使用动态光源有效渲染虚拟场景；
- 当场景中有太多动态对象时，帧间光子管理算法的性能将下降；
- 我们的方法只针对凝视点区域渲染高质量图像，可能忽略了外围区域的一些强的视觉特征。

## 后续工作

- 基于Motion Vector的思想重用动态光子；
- 将感知模型集成到基于凝视点的光子映射方法中，并在更广的视力范围内生成高质量图像；
- 扩展基于凝视点的光子映射方法以支持大规模体数据或半透明材质。



感谢聆听！