國立交通大學

資訊學院資訊科技（IT）產業研發碩士班

碩 士 論 文

在螢幕空間中做環境遮擋與間接照明

Ambient Occlusion and Indirect Illumination in Screen Space

研 究 生：林鼎盛

指導教授：施仁忠　教授

中 華 民 國 九 十 九 年 一 月

在螢幕空間中做環境遮擋與間接照明
Ambient Occlusion and Indirect Illumination in Screen Space

研 究 生：林鼎盛　　Student：Tin-Sheng Lin

指導教授：施仁忠　　Advisor：Zen-Chung Shih

國 立 交 通 大 學

資訊學院資訊科技（IT）產業研發碩士班

碩 士 論 文

A Thesis

Submitted to College of Computer Science

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master

in

Industrial Technology R & D Master Program on

Computer Science and Engineering

January 2010

Hsinchu, Taiwan, Republic of China

中華民國九十九年一月

在螢幕空間中做環境遮擋與間接照明

學生：林鼎盛　　指導教授：施仁忠 教授

國立交通大學資訊學院產業研發碩士班

摘　要

即時的全域照明在電腦圖學一直是很重要的研究主題，過去最普遍的方法為環境遮擋(Ambient Occlusion)，之後 CryTek 在他的遊戲引擎更加以改進提出一個能在螢幕空間(Screen Space)中近似環境遮擋效果的方法。本篇論文提出一個結合螢幕空間的環境遮擋(SSAO)及反射陰影圖(Reflective Shadow Map)在延遲著色中近似全域照明的方法。以螢幕空間的環境遮擋為基礎加入區域直接照明(Local direct illumination)及全域間接照明(Global indirect illumination) ，並在螢幕空間中進行內插以減少 GPU 的負擔，由於我們主要在螢幕空間上做計算所以不受場景複雜程度的影響。我們經由實作證明本演算法所產生的結果可以顯示於較大且動態的場景，並可達到即時描繪的效能。

Ambient Occlusion and Indirect Illumination in Screen Space

Student：Tin-Sheng Lin　　Advisors：Dr. Zen-Chung Shih

Industrial Technology R & D Master Program of
Computer Science College
National Chiao Tung University

ABSTRACT

Real-Time Global illumination in computer graphics is a very important topic. In the past, a popular approach was ambient occlusion. Then CryTek Company improves this algorithm. They propose an approach to approximate ambient occlusion in the screen space. In this thesis, we introduce a further enhancement to approximate global illumination by combining screen space ambient occlusion and reflective shadow map in a deferred shading context. We include local direct illumination and global indirect illumination by extending screen space ambient occlusion. On the other hand, we reduce GPU overhead by using screen-space interpolation. Since the algorithm works in screen space, it does not depend on the geometric complexity of 3D scenes. We demonstrate the result which can be displayed for large and fully dynamic scenes at real-time frame rates.

Acknowledgement

First of all, I would like to thank to my advisor, Dr. Zen-Chung Shih, for his help and supervision in this work. Also, I thank for all the members of Computer Graphics & Virtual Reality Lab for their comments and instructions. Especially, I gratefully acknowledge the helpful discussions with Research and Development department of Realluison Company on several points in the thesis. Finally, special thanks go to my family and my girlfriend, and the achievement of this work dedicated to them.

Contents

List of Figures

List of Tables

Chapter 1
Introduction

1.1 Motivation

Real-Time global illumination is still an unsolved problem for large and dynamic scenes. State of the art, computer animation and computer games must achieve sufficient frame rates are only through approximations. On the other hand, the human visual system is not sensitive to small errors in low frequency light variation. As a consequence, one such approximation is *ambient occlusion* (AO) [10]. It is often used in feature films and computer animations nowadays, because it has rather aggressive simplification of the light transport and simple implementation. However, AO still takes too much time, since it shoots rays and computes accessibility value in 3D scenes.

In order to make computer games extensively and improve efficiency of AO, the CryTek Company presents new techniques called *screen space ambient occlusion* (SSAO) [13] in his game — Crysis. The main idea behind SSAO is to approximate the occlusion function at points on visible surfaces by sampling the depth of neighboring pixel in screen space. The result will miss occlusion cues from objects that are currently hidden on screen, but the approximation is generally quite convincing and high speed simultaneously.

Although the research of AO has been already realistic result in the past, it decouples visibility and illumination, allowing only for a coarse approximation of the actual illumination.

AO typically displays darkening of cavities as Figure 1.1, but ignores directional information of incoming light even indirect lighting. We extend recent developments in SSAO such as Mittring et al.[13] and Filion et al.[5], towards more realistic illumination which include local direct lighting and one bounce indirect lighting.

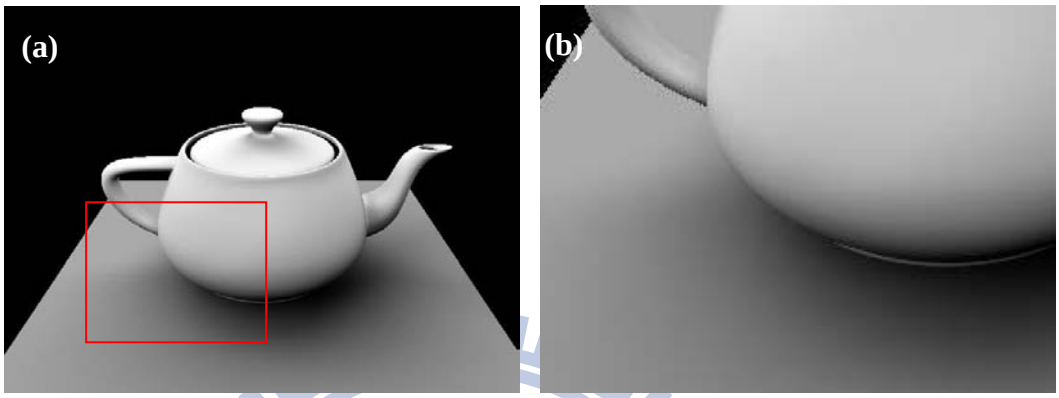

Figure 1.1 (a) AO of teapot and zoom in (b) only displayed darken of cavities

Similarly, we consider one bounce indirect light in screen space to enhance SSAO realistically. We use the reflective shadow map (RSM) [3] which is an algorithm of plausible indirect illumination. It is extension to standard shadow map, where every pixel is considered as an indirect light source. Use adaptive sampling and screen-space interpolation on-the-fly to achieve real-time frame rates. To be concise, our algorithm has the following main contribution:

1. Our research proposes a new idea to consider the screen-space visibility and one bounce of indirect illumination simultaneously within a deferred shading rendering.
2. For fully dynamic scenes, instead of restrictions on the animation and deformation undergone by the geometry and without CPU pre-computation.
3. We use image space computation on GPU and independent on complex scene to achieve real-time performance.

1.2 System Overview

Both ambient occlusion and indirect illumination are low frequency. They are assumed that imprecise shapes of shadow areas and lighting attenuation are visually insignificant. We exploit this property to simplify global illumination. Figure 1.2 shows an overview of our proposed system. The system is implemented via multi-pass algorithm running on the GPU. First, we render the scene from the view of which light source to generate RSM. The resulting depth buffer is called shadow map which can be used to generate direct shadows. Second, we render the scene from camera view and the resulting depth map and normal map are called **ND-buffer**. It produces an AO map in screen space. Third, we use adaptive sampling to evaluate the low resolution of indirect illumination which the pixel in RSM as a small area light source that illuminates the scene. Finally, we use screen space interpolation and further combined and modulated within a deferred shading rendering engine.

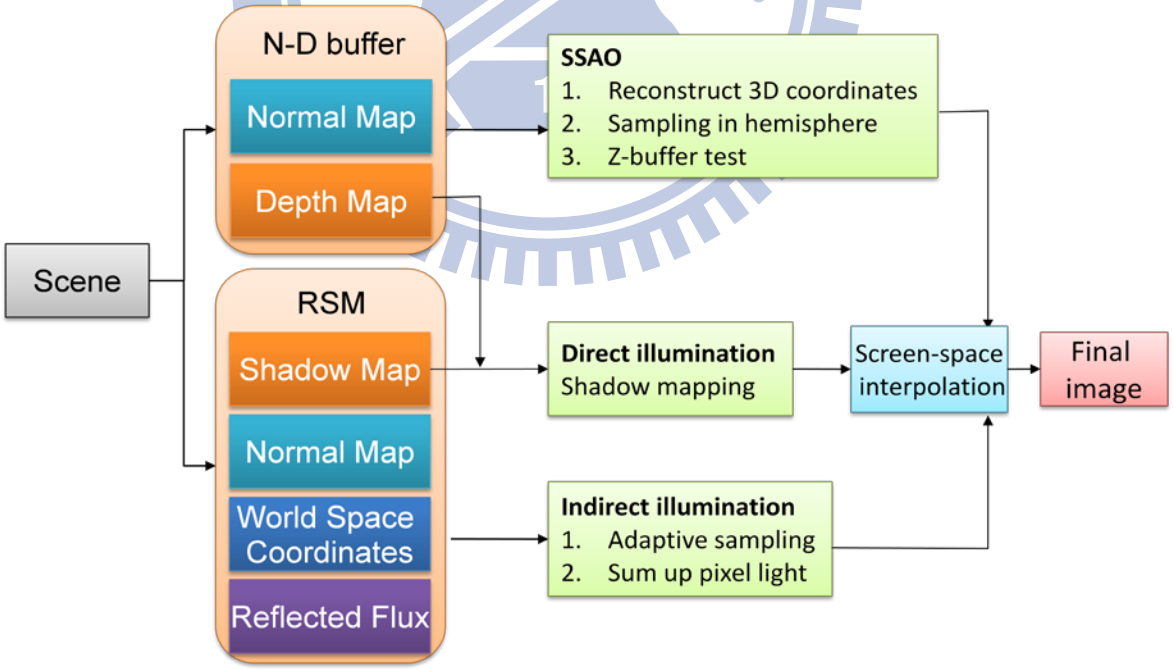

Figure 1.2 Overview of workflow

1.3 Thesis Organization

The rest of this thesis is organized as follows: First, we review some of the related works in real-time ambient occlusion and indirect illumination in Chapter 2. In Chapter 3 we describe our generalizations of ambient occlusion for the illumination and how to generate reflective shadow maps and include these effects that improves the visual quality. In Chapter 4 provides some implementation details. We present our results in Chapter 5. We also give some discussion in this chapter. Finally, we conclude our work and propose some future works in the Chapter 6.

Chapter 2
Related Works

In this chapter, we briefly review related works, with a focus on real-time ambient occlusion and indirect illumination.

2.1 Real-Time Ambient Occlusion

Several techniques have been introduced over the last five years to approximate AO in real-time. These methods range from pre-computation to without pre-computation. Mendez et al.[11] pre-calculate the ambient occlusion value on a per-patch basis and perform updates only for regions with moving objects as shown in Figrue 2.1a. Kirk and Arikan[8] takes a data-driven approach to approximate ambient occlusion in real-time on dynamic character skin as shown in Figure 2.1b. They use k-means clustering to group the degrees of freedom into a small number of pose clusters and moving least squares to blend the reconstructed ambient occlusion values from small number of pose clusters. However, these methods spend more memory space to store AO values or limit specific type for dynamic scene.

(a). **(b).**

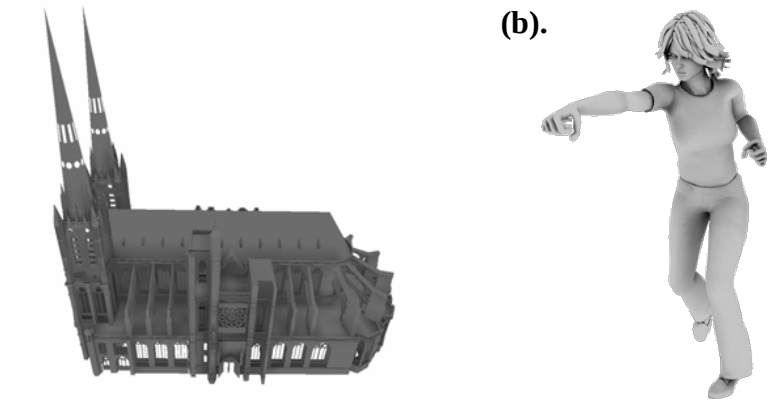

Figure 2.1 Result of pre-computation ambient occlusion

In order to avoid the pre-computation of ambient occlusion values, Bunnell et al. [2] proposed to transform meshes into surface discs of different sizes, covering the original surfaces as shown in Figrue 2.2a. They approximate the shadowing between these discs to determine ambient occlusion. However, highly tessellated objects are needed to get high quality shadows as the visibility is estimated per-vertex only. Hoberock and Jia [6] extended this algorithm to work on a per-fragment basis. Kontkanen and Laine [9] presented a technique for computing inter-object ambient occlusion. They define an ambient occlusion field in the surrounding space which encodes an approximation of the occlusion caused by the object as shown in Figure 2.2b. This information is then used for defining shadow casting between objects in real time, but it only works well for the rigid transformation of objects.

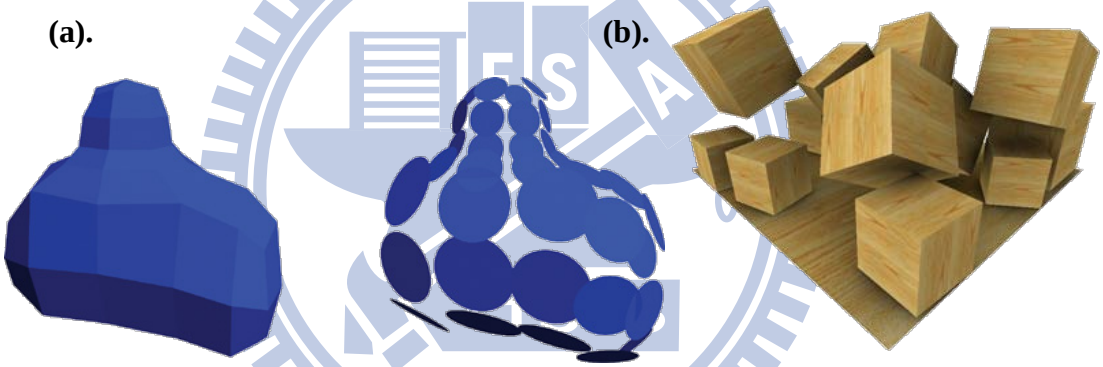

Figure 2.2(a). Surface discs representation and (b). AO field

2.2 Screen Space Method

As mentioned in the real-time ambient occlusion, these methods either use a discretization of the surface or rely on ray tracing, which both do not scale well to the amount of dynamic scenes used in current interactive applications like computer games. Therefore, instead of computing occlusion over surfaces, recent methods approximate AO in screen space (SSAO) [1, 5, 13, 16]

At first, Crytek has also implemented a new way of ambient occlusion in their recent game-Crysis [13], which mainly reduces heavy pre-process in AO effect. It has no data

preprocessing need, no loading time, no memory allocation in system memory, independent on scene complexity and simpler implementation than AO. Even, image-space methods can also be used to efficiently simulate subsurface scattering by Mertens et al. [12]. At the same time, Shanmugam and Arikan [17] split up the computation of AO into two parts: a high-frequency part, evaluated on near objects using an image-space approach, and a low-frequency part, approximated on distant objects with spherical occluders.

2.3 Indirect Illumination

Many standard global illumination methods usually intersect rays with the scene geometry, such as path tracing, photo mapping and ray tracing. They use accurate visibility for direct as well as indirect illumination. Robert et al. [16] presented Distributed ray-tracing that also called Stochastic ray-tracing. As shown in Figure 2.3, it send many secondary rays in randomized directions and could handle and kind of reflection. Keller [7] proposed Instant Radiosity. His idea is tracing paths from primary light source and creates new virtual point light (VPLs). These light sources are placed at each reflection point and used to illuminate scene. However, both methods are too slow for complex scene in global illumination.

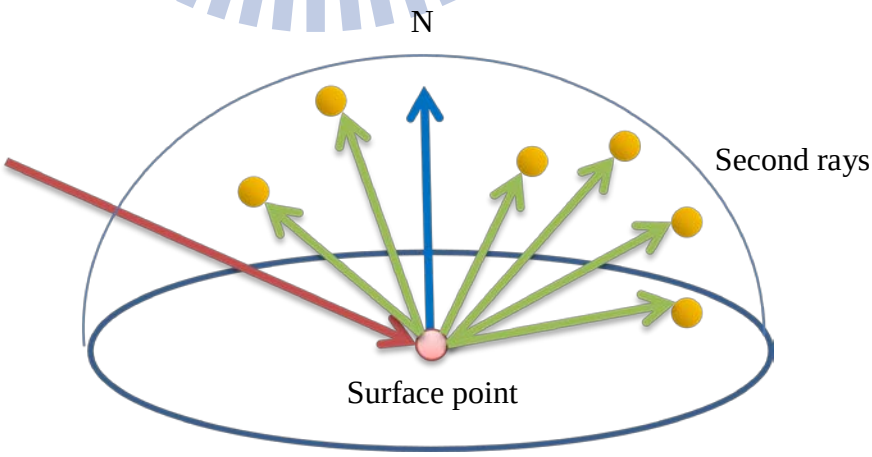

Figure 2.3 Distributed ray-tracing

Ritschel et al. [15] presented a method for interactive computation of indirect illumination in large and fully dynamic scenes based on approximate visibility queries. They named this approach "Imperfect Shadow Maps" and rendered from a crude point-based representation of the scene, but ultimately failed to handle non-hierarchical point representation scenes. Further, indirect shadows are generally smoothed out considerably. Dachsbacher and Stamminger [3] presented "Reflective Shadow Maps", an algorithm for interactive rendering of plausible indirect illumination. They compute a rough approximation on screen space for the one-bounce indirect light in a scene. To exploit indirect illumination is low frequency property, ignores indirect occlusion to improve rendering time, instead of shooting many rays repeatedly in a scene. Our algorithm also uses this concept in indirect lighting computation to draw indirect illumination.

In summary, our work takes advantage of information that is already computed during the SSAO process to approximate indirect illumination in the realism of results. We will describe how the illumination can be computed efficiently and coherently to produce plausible indirect light.

Chapter 3
ALGORITHM

Basically, in our system, we enhance the approach proposed by Mitting et al.[13] ,Dachsbacher and Stamminger[5]. The enhancement considers indirect illumination into the original screen space ambient occlusion algorithm.

First, we compute accessibility value from the scene geometry in image space, which considers ND-buffer from the view of camera, and sampling at each pixel in the run time process. Then we consider the view of light to generate the reflective shadow map in deferred buffer and use adaptive sampling on RSMs to reduce evaluation of the sum pixel light sources. Finally, we speed up the rendering performance that use screen space interpolation according to the previous generated SSAO and indirect illumination results.

The following sections are organized as follows. In Section 3.1, we review the work of screen space ambient occlusion method related to our approach. In Section 3.2, we describe reflective shadow maps data and how to generate. In Section 3.3, we describe how to evaluate indirect irradiance to improve our scene realism. Finally, in Section 3.4, we use screen space interpolation of SSAO and indirect lighting and describe how to speed up the performance.

3.1 Screen Space Ambient Occlusion

Our SSAO is inspired by the Crysis SSAO algorithm [13] but also the Filion et al. [5] approach that requires depth and normal information to be stored in a prior step. This is the same depth buffer that we use for deferred lighting and then we consider the normal information improve original result. Figure 3.1 illustrates the idea of our work. We use the value of z-buffer to reconstruct pixel world-space position from screen space. At any visible point of a surface on the screen, we take multiple samples from neighboring points in the scene. The number of samples may be 8 to 32 (in our experiment, we use 16 is good). We take the surrounding of a pixel and compare their depth comparison. Then we are able to compute a darkening factor to obtain silhouettes around the objects.

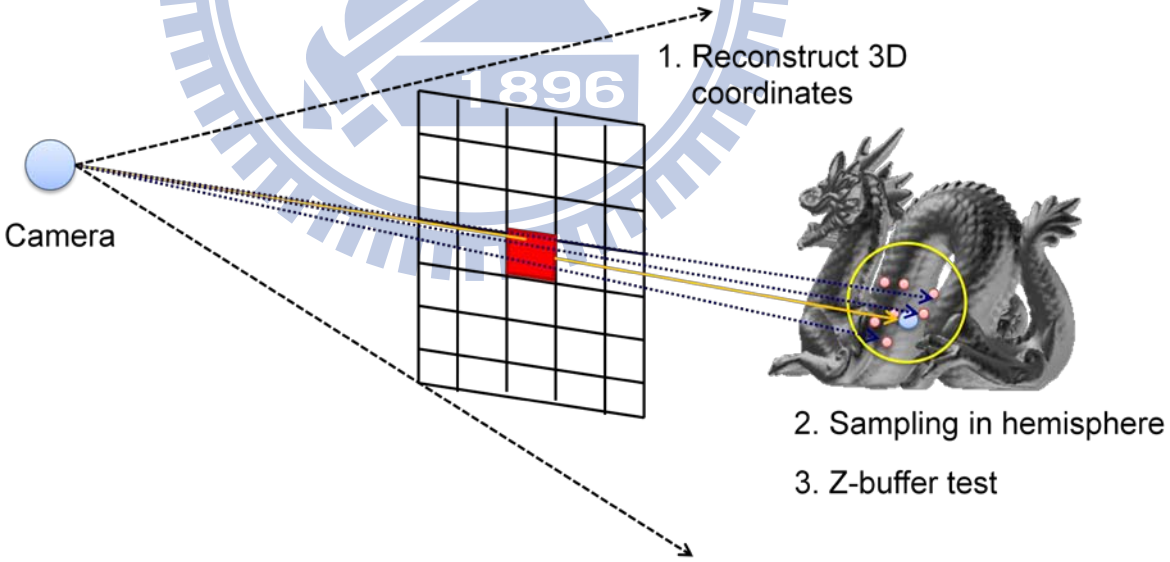

Figure 3.1 SSAO process

According to Filion et al. [5], the depth test of sample points is not simply a Boolean

operation. It determines the distance between current point and sample point as shown in Figure 3.2a. If the occluding surface is close to the pixel being occluded, it will occlude a lot more than it is further away. Thus we need to compute occlusion factor and how much occlusion occurs. On the other hand, consider dot product of pixel normal between current point and sample point that handle self occlusion to produces convincing results. Because the polygon of geometry faces to same direction more absorb light source than face to itself. For example, it is obtained in the Figure 3.2b.

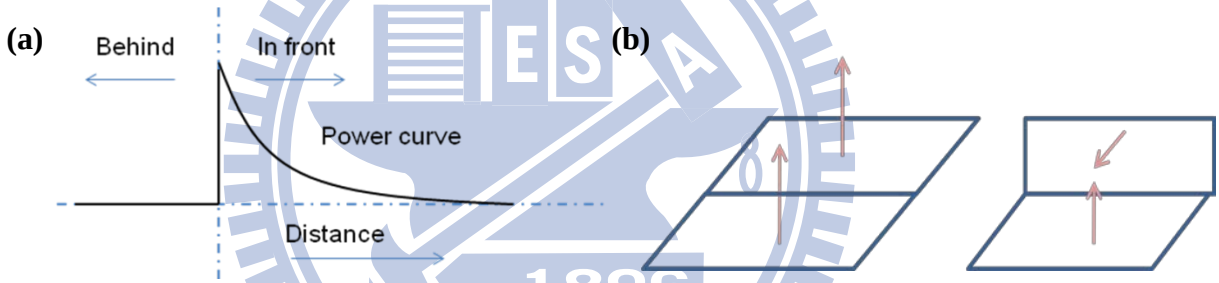

Figure 3.2 ND-buffer means (a) depth buffer and (b) normal buffer

In order to make the result physically correct, the sample vectors should be randomized thoroughly as showed in Figure 3.3. In our experiment, we use random noise texture to trace 16 rays for every fragment on the screen. The rays are shot in a random direction in a hemisphere around the normal of the current fragment. By texture lookups, the depth and normals are compared between current fragments and traced ones.

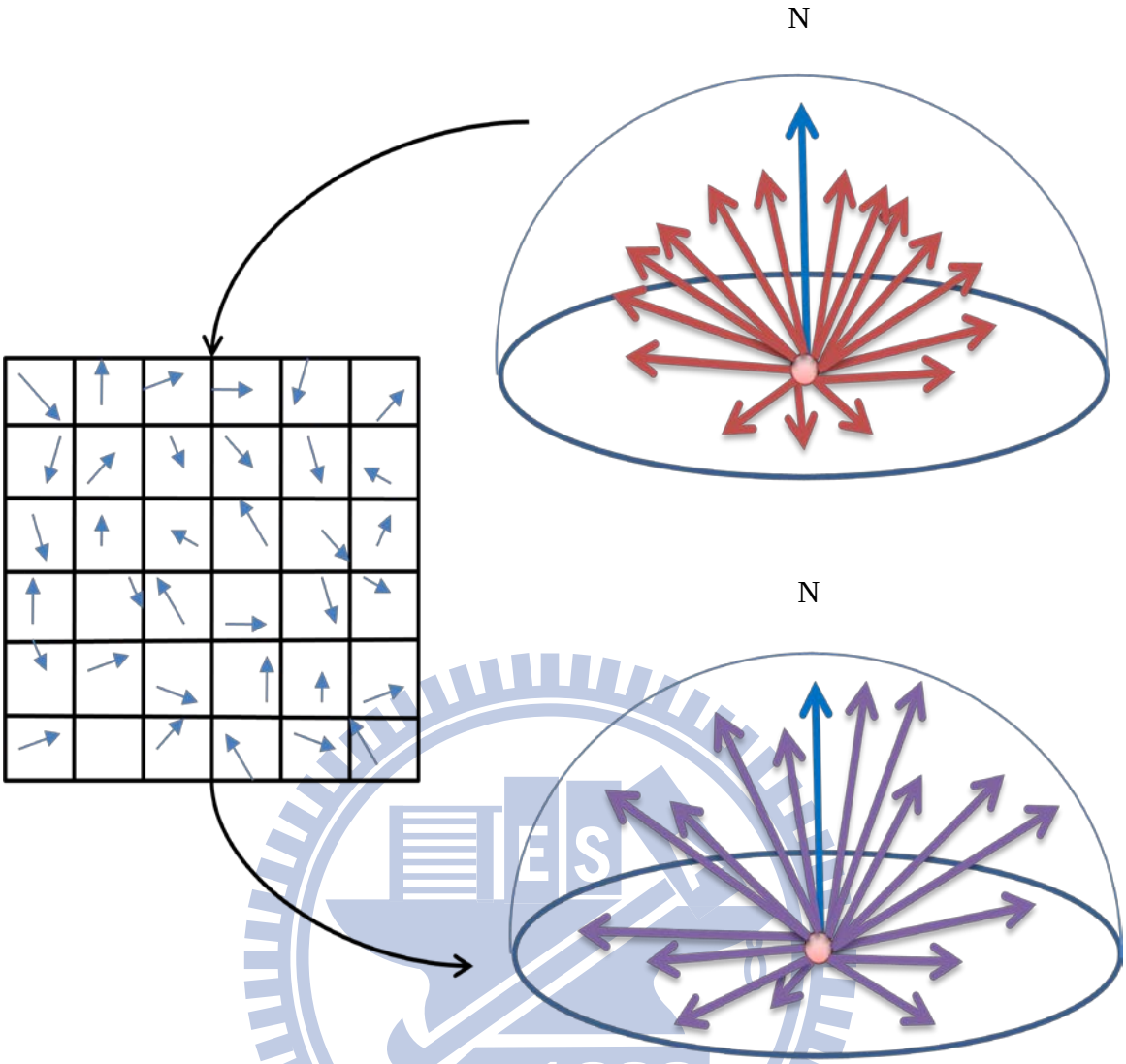

Figure 3.3 Randomized sampling process

Figure 3.4a illuminates SSAO result of without random noise of the sampling. It has artifact that quality-wise is clearly not good enough to be acceptable. As shown in the Figure 3.4b, we use randomly rotated texture of sample vectors to improve quality. Unfortunately, we also introduce a lot of noise in final picture. A common method to reduce the noise is to blur the image but also introduce a lot of noise in the final picture. So we pick a 4x4 pixel blurring area size to remove high frequency components of the noise as shown in Figure3.4c.

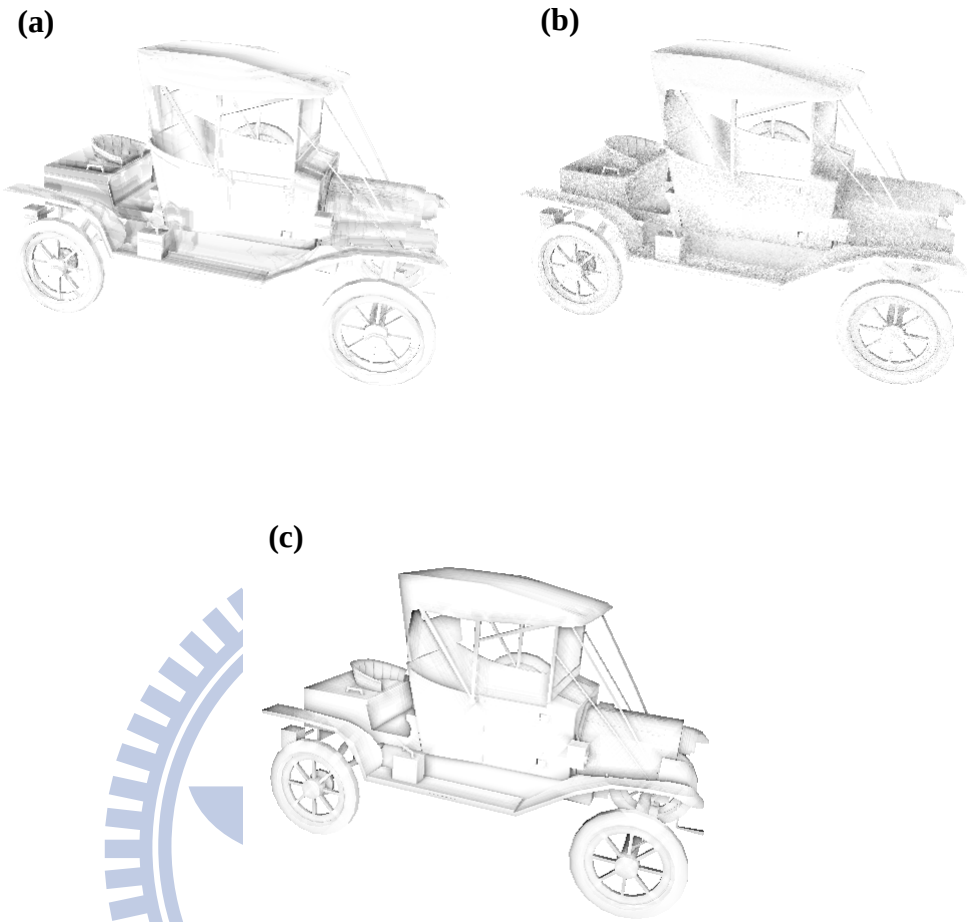

Figure 3.4 (a) SSAO Map with ND-buffer (b) after random sampling applied and (c). blur result image

3.2 Reflective Shadow Maps

Reflective Shadow Maps (RSMs) have been developed by Carsten Dachsbacher and Marc Stamminger in 2005[5]. The idea is to consider all pixels of shadow map as indirect light sources. We use this concept to generate the one-bounce indirect illumination in a scene. By sampling all points of the shadow maps generated in a scene, it is easy to capture all

secondary light sources. The result gives moderate color bleeding based on the locality of objects in the scene.

However, in order to obtain this shadows map, we still need more attributes: Position, depth from light view, normal, and flux, as shown in the Figure 3.5. We then render the whole scene from light view to generate RSMs in the first pass.

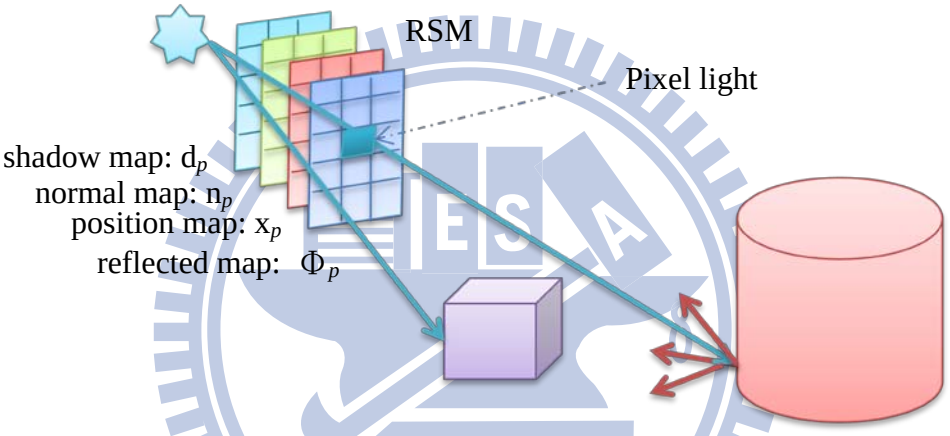

Figure 3.5 RSMs organization

Flux Φ_p is the light emitted by a given point in the shadow map. The calculation of the light is straight forward. The flux is a constant value for a uniform parallel light and for a spotlight the flux decreases with the cosine to the spot direction due to decreasing angle. The reflected flux is then the flux through the pixel times the reflection coefficient of the surface. We required flux is obtained by combining the reflective component of the emitting surface to get the final value the shadow map. As shown in the Figure 3.6, the flux buffer looks like an unshaded image.

Figure 3.6 Flux Map

The world space position x_p can be computed from the pixel coordinates and the depth value. The depth value d_p from the view of light, we can generate direct shadow that use approach of shadow mapping [14,18]. As show in Figure 3.7, we compare pixel value d_c of depth map of camera view and pixel value d_l of shadow map of light view. If d_c larger than d_l, it is the shadow point. The normal n_p, it is spatial emission characteristics. This combined information generates the irradiance of a given pixel in the shadow map.

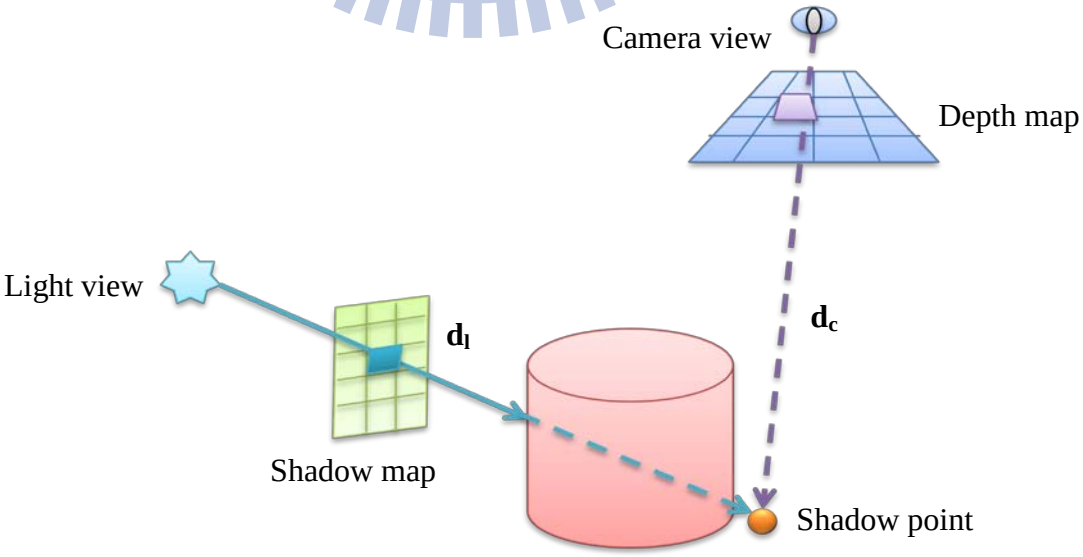

Figure 3.7 Shadow Mapping

3.3 Indirect Irradiance Evaluation

As mentioned in Sections 3.2 and 3.3, we have already direct lighting information and accessibility value from screen space. The next step is combining indirect light to improve the realism. The irradiance at a surface point ***x*** with normal ***n*** due to pixel light ***p*** as follow:

$$E_p(x,n) = \Phi_p \frac{\max(0, n_p \cdot (x - x_p))\max(0, n \cdot (x_p - x))}{\|x - x_p\|^4} \quad (1)$$

The summed irradiance of all pixels on the shadow map represents the total contribution from indirect lighting in the scene.

$$E(x,n) = \sum_{pixels\ p} E_p(x,n) \quad (2)$$

Figure 3.7 shows an example of point light illuminates a room with a box from upper left. A pixel light source at position x_p illuminates the scene along its normal n_p. According Equ.(2), the point x on the floor receives light from x_p. Because an RSM cannot capture occlusion for indirect lighting, the floor is also lit on the opposite side at location y of the box. But in most cases, it is better to generate some indirect lighting effects and acceptable imprecise result. Furthermore, we introduce SSAO effect in the first step and light attenuation with distance can alleviate these artifacts.

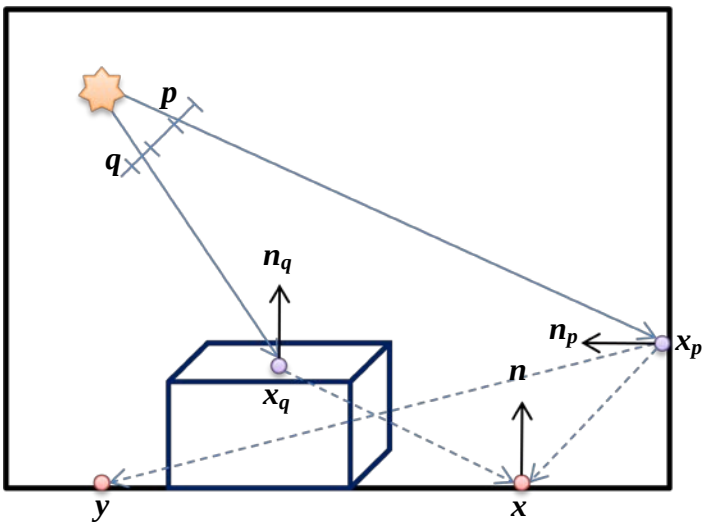

Figure 3.8 Example of a room with box

However, if we consider every pixel light on RSM for indirect lighting of ***x***, the evaluation of the summation is very expensive and not practical in a real-time context. Thus, in the summation, we only consider smaller number of light sources to approximate indirect illumination and then speed up the computation. By using important-driven approach, we can reduce the number of pixel lights and try to concentrate the sampling to the relevant pixel lights.

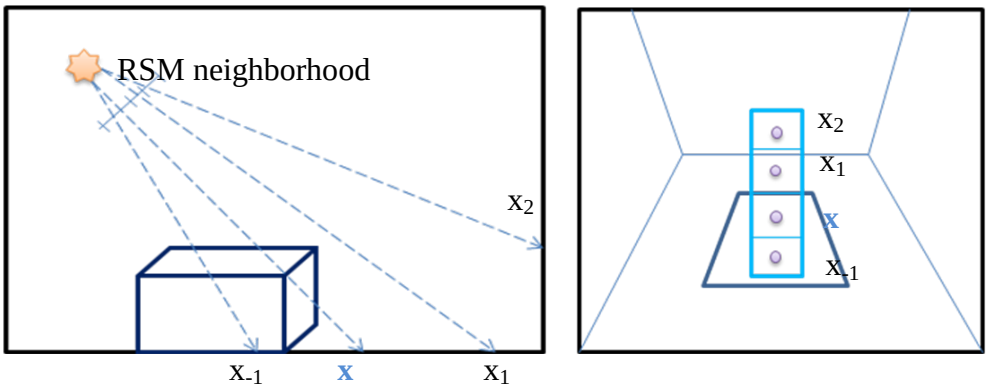

Figure 3.9 Indirect lights will always be close

Consider in Figure 3.8. The surface point x is not illuminated directly, so it is not contained in the RSM. If we project x into the shadow map, the pixel lights that are closest in world space are also close in the shadow map. Although x_{-1}, x_1 and x_2 are close to x, only emission of the pixel light x_2 is directed accordingly to contribute to the lighting of x. So, we can say the distance in the shadow map between x and a pixel light x_p is a reasonable approximation for their distance in world space.

Then, we project the surface point x to be lit into the shadow map and select pixel lights around its shadow map coordinate (s,t). The sample density decreases with the squared distance to (s,t). The sampling pattern can be computed using polar coordinates and two uniformly distributed random numbers ξ_1 and ξ_2 as follow:

$$(s + r_{\max}\zeta_1 \sin(2\pi\zeta_2), t + r_{\max}\zeta_1 \cos(2\pi\zeta_2)) \quad (3)$$

In order to compensate the varying sampling density, we multiply the achieved samples with ξ_1^2 and perform a final normalization by different disc radii.

3.4 Screen Space Interpolation

We have already computed indirect illumination and ambient occlusion using per-pixel lighting in previous steps. Although we reduce the number of samples to a few hundred, the indirect lighting computation is still very expensive. Fortunately, the indirect lighting as

described above is low frequency. We can evaluate indirect lighting only at fewer locations in screen space and use interpolation for the remaining pixels.

First, we compute the indirect illumination for a low-resolution camera view, as shown in Figure 3.9a. Then we check for every pixel if the indirect lighting can be interpolated from its four surrounding low-resolution image. If their world-space position is nearby the pixel's position and if their normals are similar (we compare the custom threshold), the sample is suitable for interpolation. The contributions of these samples are weighted by the same factors used for bi-linear interpolation and normalization if not all four samples are used. If no interpolation is possible, we discard the pixel in the rendering pass and its indirect lighting is computed later with a complete light gathering step.

Figure 3.9b shows the result after screen space interpolation. For planar surfaces, the interpolation works well and only a few samples are required. However, for complex or unconnected geometry, the interpolation scheme will not be applicable and falls back to full evaluation.

Figure 3.10(a) low-resolution indirect illumination and (b) result after screen space interpolation image

Chapter 4
IMPLEMENTATION DETAILS

Since our approach works on screen space, the goal is to reduce the dependency of scene. The deferred shading algorithm suits this scenario very well, because it avoids rasterizing the geometry multiple times, which would be mandatory in traditional multi-pass approach. Therefore, it is necessary to use multiple render targets (MRTs) that store intermediate values.

We implemented our approach using DirectX 9.0c with High Level Shader Language. Further, we use Pixel Shader 3.0 that supporting more shader length, dynamic branching and multiple render targets. We apply the direct and indirect illumination as a deferred shading process to accelerate rendering performance as far as possible to prevent the execution of complex shaders for hidden surfaces.

In our approach, any rendered opaque model will store the following to MRTs bound in its main rendering pass as shown in Figure 4.1~4.2: depth, normal, ambient occlusion term, material parameters, world space position and its corresponding coordinates in light space. These textures are view-dependent and need to be updated when the camera moves. However, much of the mainstream hardware is limited to four rendering targets bounds at once, so we make sense to pick deferred component that will fit within these target.

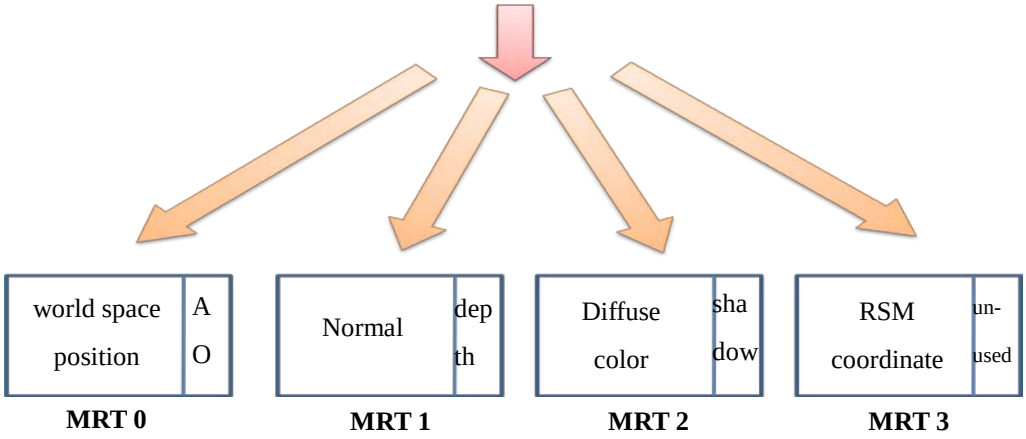

Figure 4.1 MRT setup

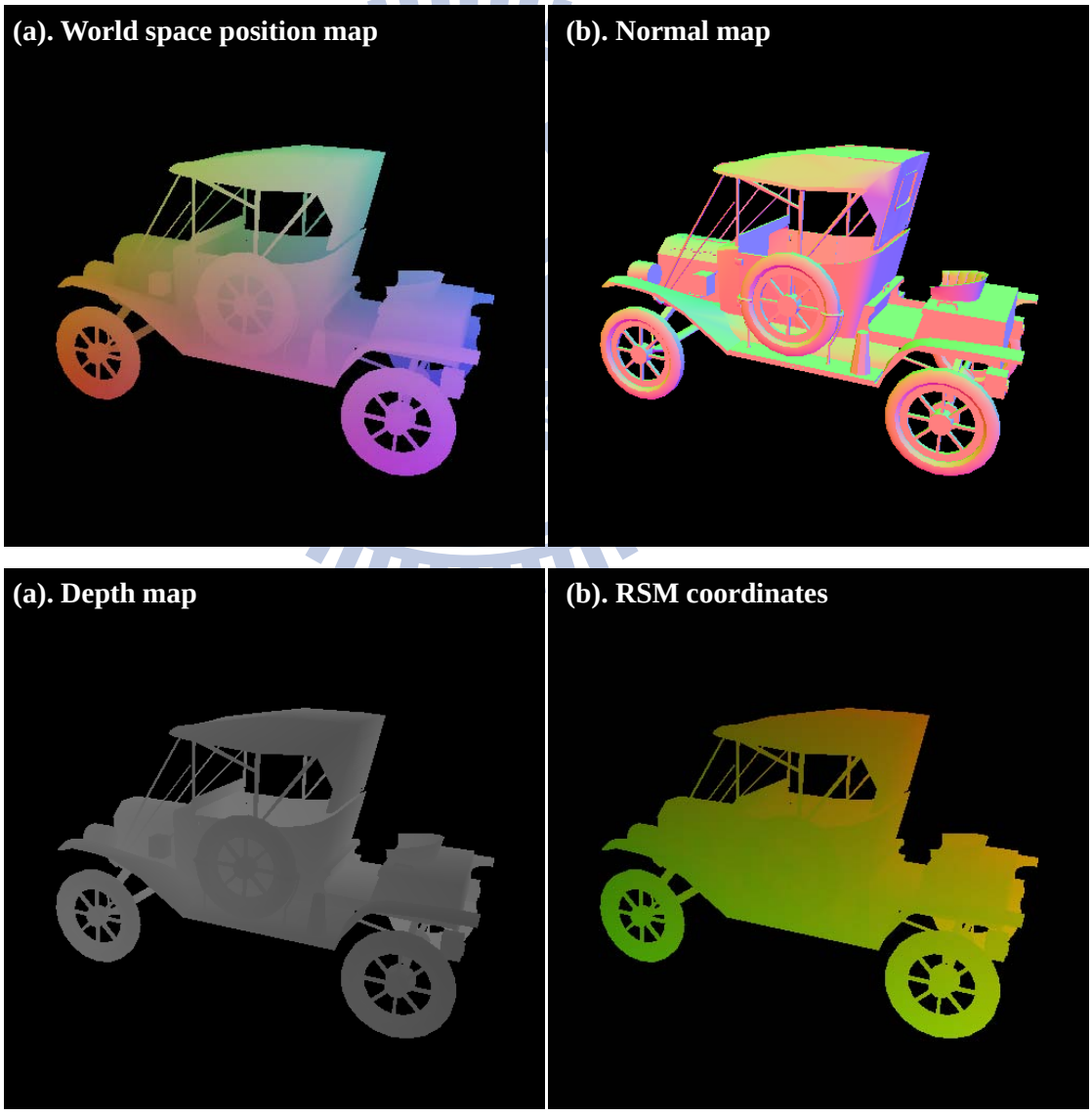

Figure 4.2 Deferred buffers

Chapter 5
RESULTS AND DISCUSSION

We implement our algorithm on a desktop PC with Intel Core i7 975 CPU 3GHz, 8G RAM, and NVIDIA GeForce GTX 295 video card. As mentioned before, all results are rendered with single-bounce of indirect illumination at 512 x 512 and 1024 x 768 pixels.

Figures 5.1~5.4 show four scenes render by our technique. We vary both the RSM sampling as well as the number of samples used for each indirect illumination to demonstrate their influence on the result. Furthermore, we compare our results to a difference effect. Consider the first row. Two scenes are rendered with direct lighting by shadow mapping, whereas the second row are rendered with add indirect lighting by 200 samples of RSM sampling and the last scenes are rendered with 400 samples. On the other hand, in the right column, three scenes are rendered with SSAO by 16 samples and the left column without.

The next step that we closer views of four scenes to find out what difference and describe our result is plausible global illumination and achieves real-time frame rates.

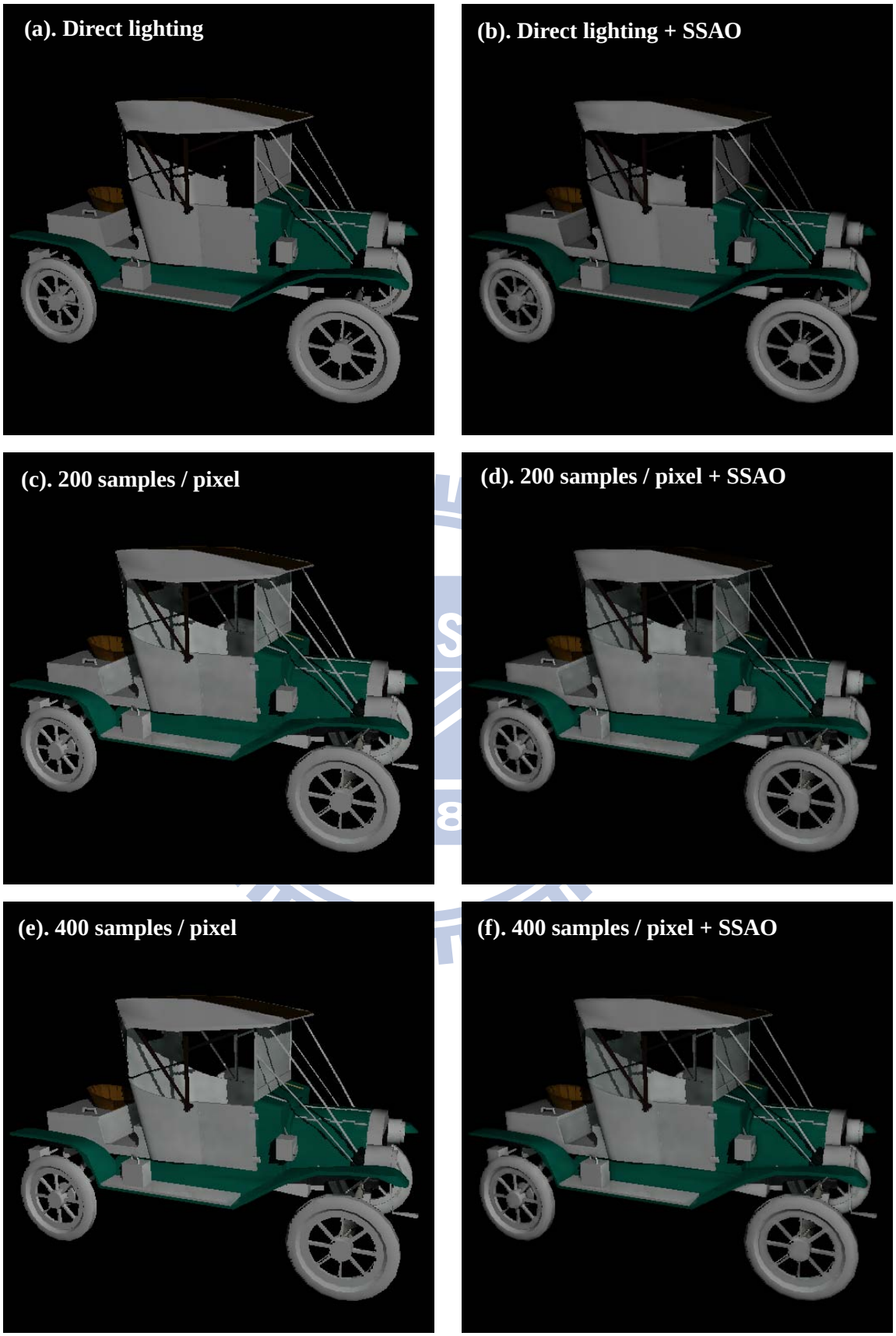

Figure 5.1 Ford Scenes

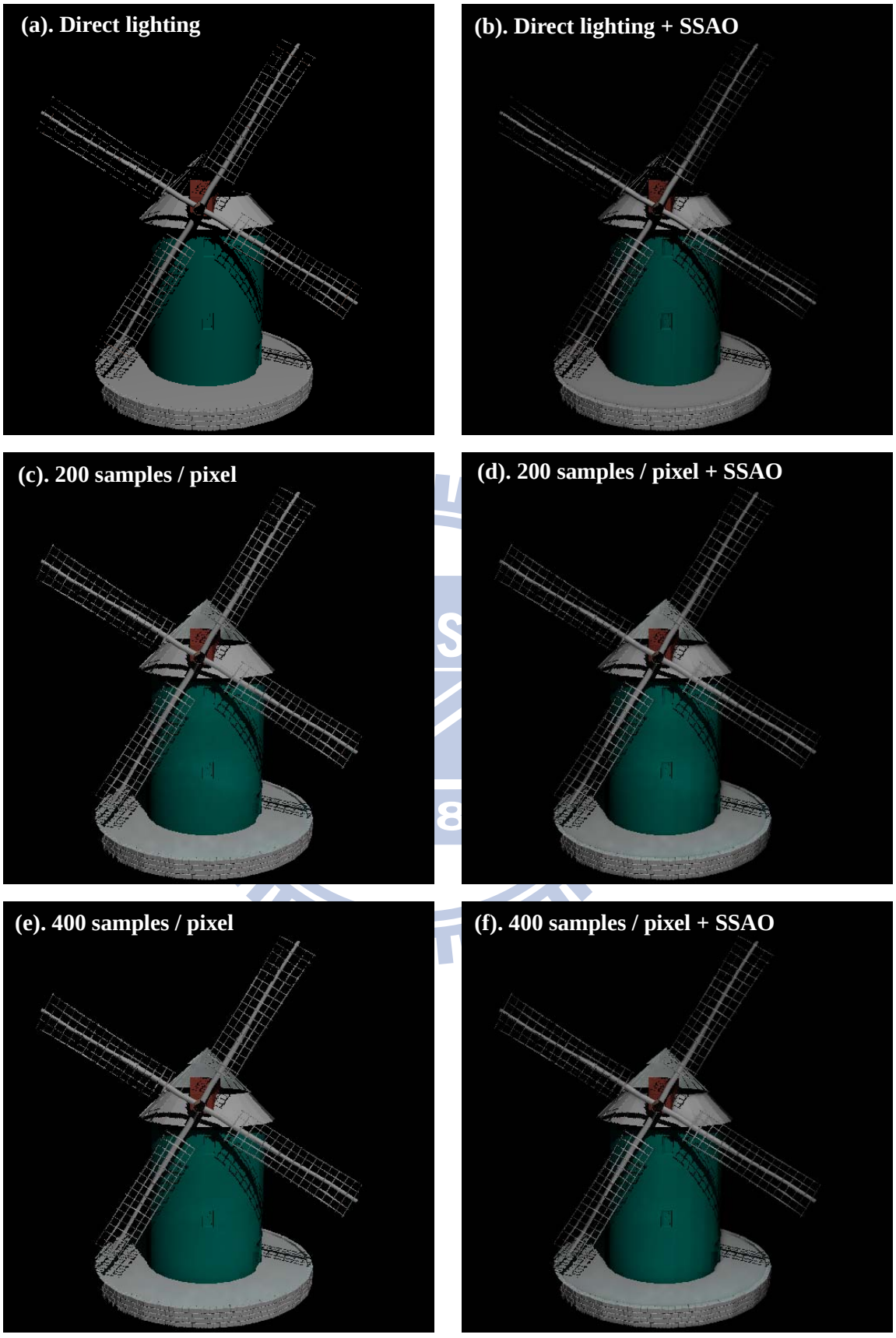

Figure 5.2 Windmill Scenes

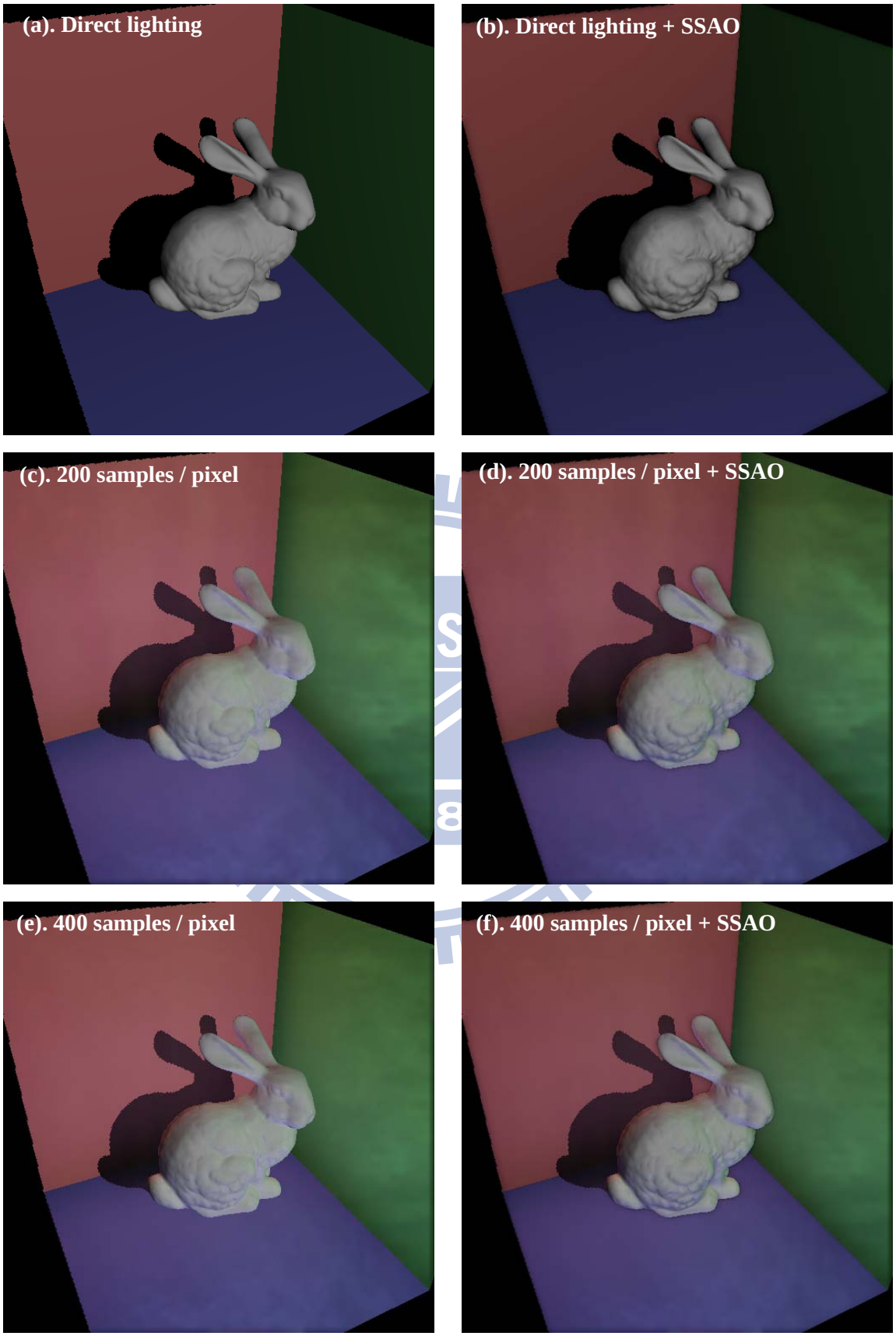

Figure 5.3 Bunny with three color walls

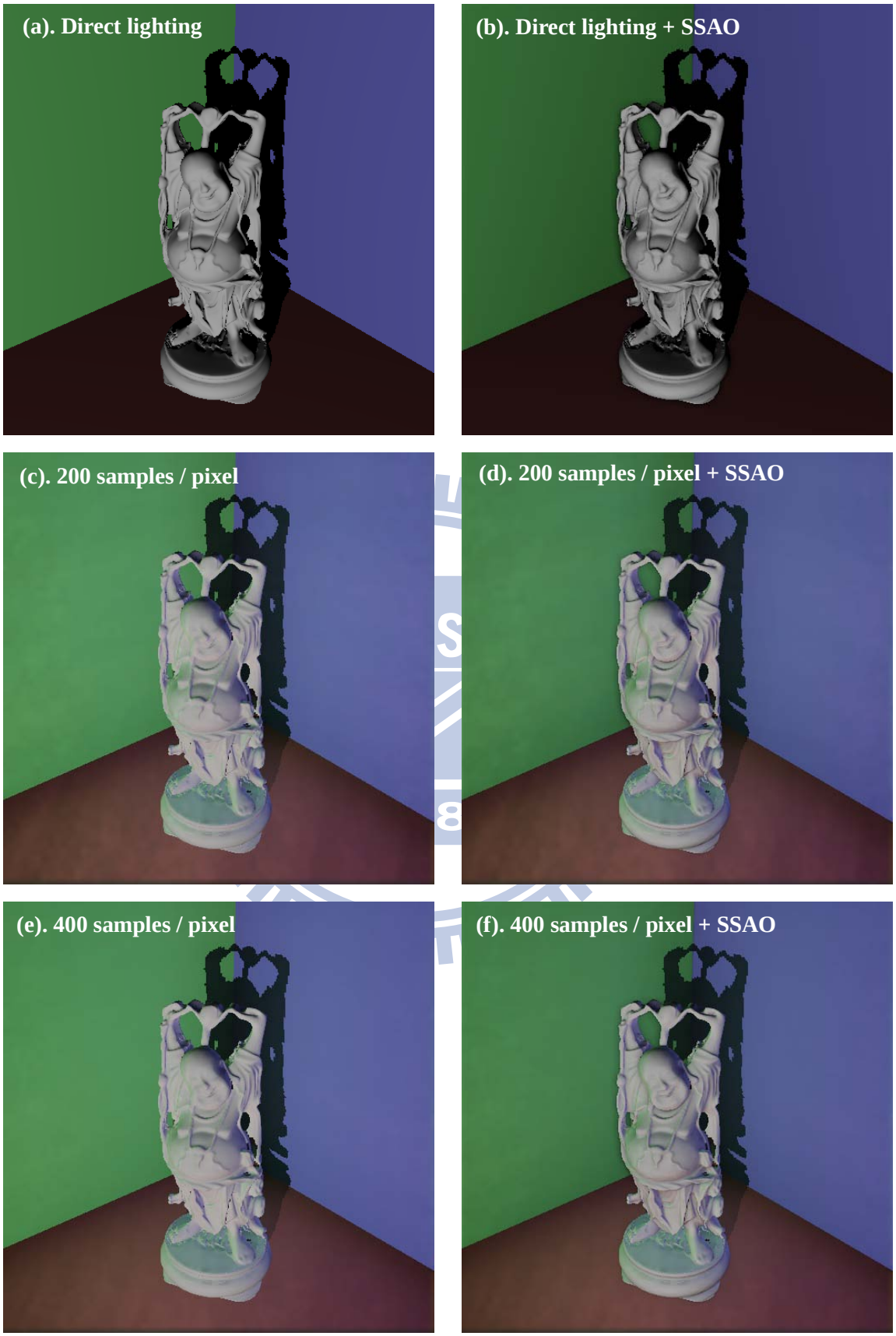

Figure 5.4 Buddha with three color walls

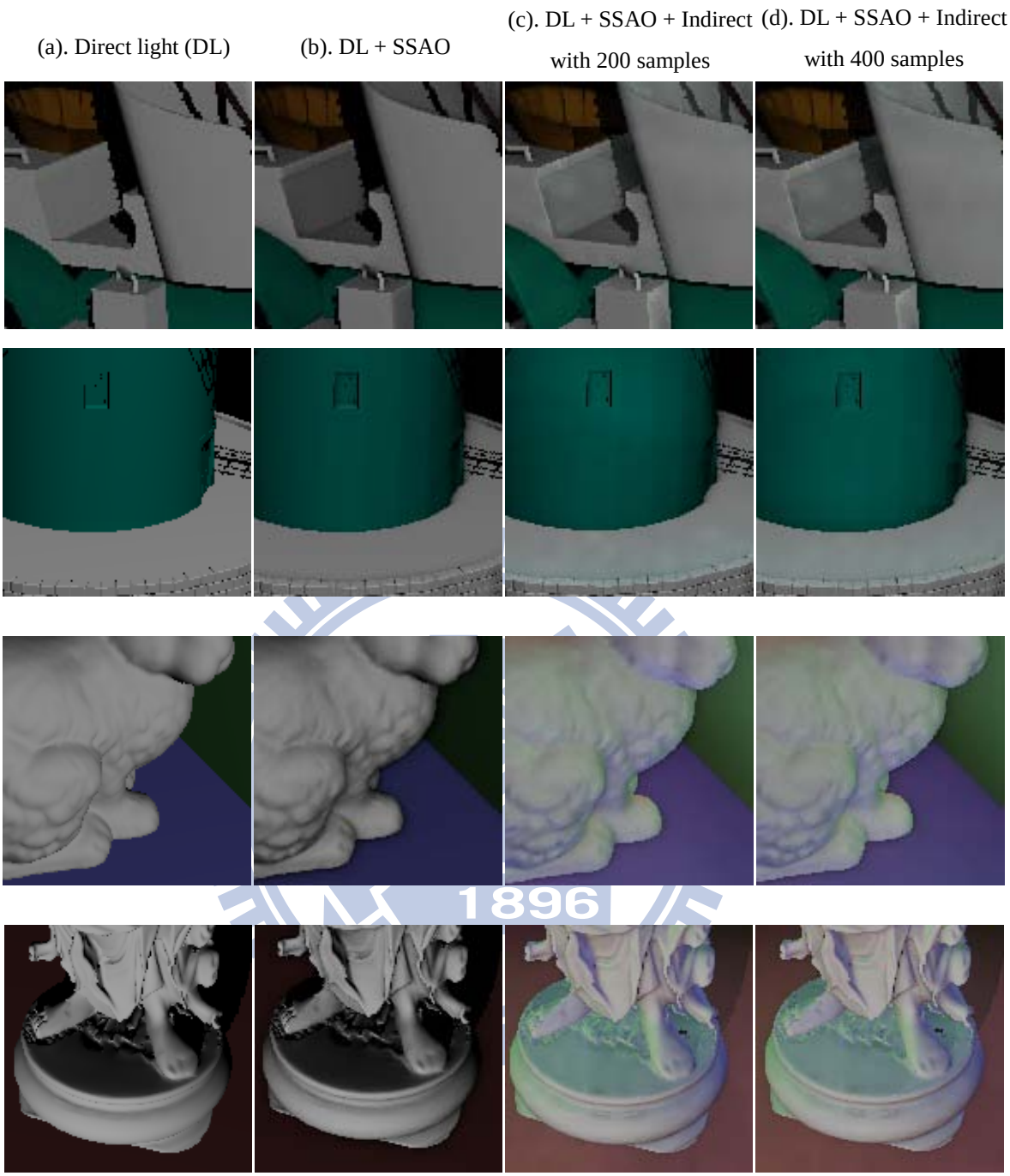

Figure 5.5 Closer views of four scenes

As shown in Figure 5.5a~d, we zoom in these four scenes to close view of difference region by different effect and vary samples of RSM sampling for indirect illumination. As shown the first column results are rendered with direct lighting that only represents direct shadow, diffuse color and silhouette of object. The second column results are

rendered with direct lighting and SSAO are improved darkening of cavities. The last two scenes are rendered with single-bounce indirect illumination. The Figure 5.5d uses 400 sample of RSM samples the result smoother than 200 samples.

Table 5.1 shows the result of our implementation using resolution of 512 x 512 and 1024 x 768 pixels for the camera view. We measured the difference in performance for various RSM samplings and screen sub samplings of indirect lighting. As shown in the table, the performance does not depend on the geometry complexity. Note that the user can move light source and camera view freely and interactively, we recomputed the light and camera view for each frame. Even though, we can still achieve real-time frame rate for large scene.

Table 5.1

Scene	Resolution	DL+ SSAO		One Bounce Indirect lighting			
		samples	fps	RSM Sampling (samples)	Low-res	fps	+%
Ford (11,878 faces)	512 x 512	16	724	200	64 x 64	288	60
				400		146	79
	1024 x 768		433	200	128 x 96	66	84
				400		38	91
Windmill (50,176 faces)	512 x 512	16	592	200	64 x 64	366	38
				400		190	68
	1024 x 768		383	200	128 x 96	120	68
				400		67	82
Bunny (130,196 faces)	512 x 512	16	411	200	64 x 64	349	15
				400		181	53
	1024 x 768		265	200	128 x 96	76	71
				400		47	82
Buddha (86,138 faces)	512 x 512	16	452	200	64 x 64	347	23
				400		181	60
	1024 x 768		310	200	128 x 96	74	77
				400		50	83

Table 5.1 Typical frame rates with various resolutions of figure 5.1~5.4 for our method

Chapter 6
CONCLUSION AND FUTURE WORK

We propose a new algorithm by using directional information and indirect bounces of screen space ambient occlusion. Both extensions considerably improved realism to offer the possibility of approximating indirect illumination in dynamic scenes. However, while our methods succeed to generate approximate global illumination, there are some limitations. In our current implementation, we use reflective shadow maps (RSMs) to generate indirect light source, restricting us to point and spotlight illumination. Additionally, we only can handle opaque model of scene.

In the future we would like to apply our method to other types of light source. Other possible extensions would be multiple indirect bounces, secular materials, subsurface scattering and important sampling. One could think about multiple indirect bounces that use RSMs of multiple resolutions or similar imperfect shadow map (ISM) [15] to generate imperfect reflective shadow map (IRSM). Instead of consider the single resolution for RSM to store flux buffer.

Since shadow mapping only generate hard shadow and anti-aliasing problem. We would combine any soft-shadow algorithm for direct illumination. We believe that the optimal combination of improved approaches is an interesting avenue of further research bringing

together physically plausible illumination of dynamic complex scenes and read-time rates.

References

[1] Bavoil. L., Sainz, M., and Dimitrov, R. 2008. Image-space horizon-based ambient occlusion. In SIGGRAPH '08: ACM SIGGRAPH talks, ACM, New York, NY, USA, 1-1.

[2] Bunnell, M. 2005 Dynamic Ambient Occlusion and Indirect Lighting. In GPU Gems2, M. Pharr, Ed. Addison Wesley, Mar., ch. 2, 223-233.

[3] Dachsbacher, C., and Stamminger, M. 2005. Reflective Shadow Maps. In Proceedings of the ACM SIGGRAPH 2005 symposium on Interactive 3D Graphics and Games, 203-213.

[4] Dachsbacher, C., and Stamminger, M. 2006. Splatting Indirect Illumination. In Proceedings of the ACM SIGGRAPH 2006 symposium on Interactive 3D Graphics and Games,93-100.

[5] Filion, D., and Mcnaughton, R. 2008 Effects & techniques. In SIGGRAPH '08: ACM SIGGRAPH 2008 classes, ACM, New York, NY, USA, 133-164

[6] Hoberock J., and Jia, Y. In GPU Gems3 High Quality Ambient Occlusion. Addison-Wesley, Reading, MA, ch. 12, 239-274

[7] Keller, A. 1997. Instant Radiostiy. In SIGGRAPH '97, 49-56.

[8] Kirk, A.G., Arikan, O.: Real-time ambient occlusion for dynamic character skins. In Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games, pp. 47-52. ACM, Seattle, WA (2007).

[9] Kontkanen, J., and Laine, S. 2005. Ambient Occlusion Fields, In Proceedings of ACM SIGGRAPH 2005 Symposium on Interactive 3D Graphics and Games, 41-48.

[10] Mendez-Feliu A., Sbert M.: From obscurances to ambient occlusion: A survey. Vis. Comput. 25, 2 (2009), 181-19.

[11] Mendez-Feliu A., Sbert M., Cata J.: Real-time obscurances with color bleeding. In SCCG '03: Proceedings of the 19th spring conference on Computer graphics (New York, NY, USA, 2003), ACM, pp. 171-176.

[12] Mertens, T., Kautz, J., Bekaert, P., Reeth, F. V., and Seidel, H.-P. 2005. Efficient Rendering of Local Subsurface Scattering. Comput. Graph. Forum 24, 1, 41-49

[13] Mittring M.: Finding next gen: Cryengine 2. In SIGGRAPH 07: ACM SIGGRAPH 2007 courses, ACM, pp. 97-121.

[14] Reeves, W. T., Salesin, D. H., and Cook, R. L. 1987. Rendering antialiased shadows with depth maps. In Computer Graphics (Proceedings of SIGGRAPH '87), vol. 21, 283-291.

[15] Ritschel, T., Grosch, T., Kim, M. H., Seidel, H.-P., Dachsbacher, C., and Kautz, J. 2008. Imperfect shadow maps for efficient computation of indirect illumination. ACM Trans. Graph. (Proc. SIGGRAPH Asia) 27, 5, 12:1-129:8.

[16] Robert L. Cook , Thomas Porter , Loren Carpenter, Distributed ray tracing, ACM SIGGRAPH Computer Graphics, v.18 n.3, p.137-145, July 1984

[17] Shanmugam, P., and Arikan, O. 2007. Hardware Accelerated Ambient Occlusion Techniques on GPUs. In Proceedings of ACM Symposium in Interactive 3D Graphics and Games. ACM, B. Gooch and P.-P.J. Sloan, Eds., 73-80.

[18] Williams, L. 1978. Casting curved shadows on curved surfaces. In Computer Graphics (Proceedings of SIGGRAPH '78), vol. 12 270-274.