Deformable Model Collision Detection using A-Buffer

Hanyoung Jang[*] JungHyun Han[†]

College of Information and Communications
Korea University, Seoul, Korea

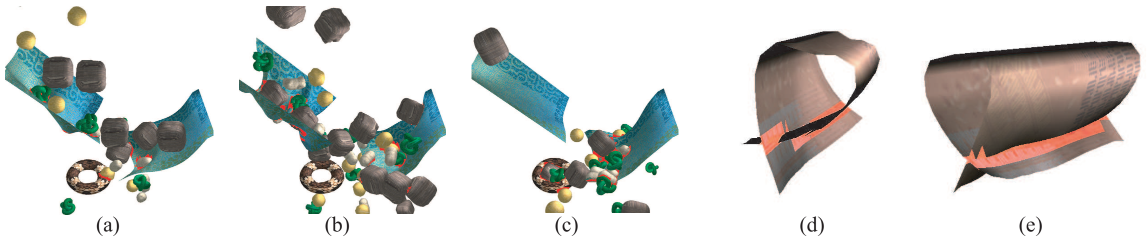

Figure 1: *Collision detection for deformable object : (a)(b)(c) collision of multiple objects, (d)(e) self-collision of cloth.*

Abstract

This paper presents a collision detection algorithm with A-Buffer for handling deformable objects in real-time. The proposed algorithm overcomes a variety of weaknesses which are revealed in the existing collision detection techniques: it does not require any preprocessing, it can handle dynamic models including deformable and fracturing meshes, it can compute self-collisions, etc. Its performance proves useful in real-time applications such as 3D games.

1 Introduction

The image-space collision detection algorithms do not require additional data structures like BVHs for collision detection. Therefore they can effectively handle deformable objects and dynamic environments. However they also have disadvantages, such as the overhead of rendering and readback, and inaccuracy of collision results. Our method improves such weaknesses of the image-space algorithm. The proposed method adopts stencil routing A-buffer [Myers and Bavoil 2007], which can capture layered depth images(LDIs) 8 times faster than ordinary methods on current shipping hardware. Moreover using a stream reduction technique, readback cost is reduced significantly. Finally our method computes potentially colliding sets(PCSs) in GPU, and CPU performs triangle intersection test using the PCSs. It guarantees the triangle level accuracy of collision response.

2 Collision Detection in GPU

In the proposed algorithm, the GPU finds PCSs. The GPU work consists of three parts - surface accumulation, PCSs generation, and stream reduction.

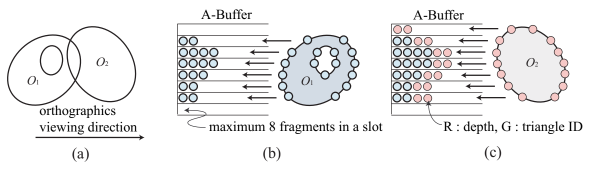
O_1
O_2
orthographics viewing direction
(a)
A-Buffer
O_1
maximum 8 fragments in a slot
(b)
A-Buffer
O_2
R : depth, G : triangle ID
(c)

Figure 2: *System architecture.*

[*]e-mail: jhymail@gmail.com
[†]e-mail:jhan@korea.ac.kr

Fig. 2 illustrates the mechanism of surface accumulation. The scene is shown in Fig. 2-(a). When O_1 and O_2 are rendered in sequence, all the fragments of O_1 and O_2, up to 8 fragments, are accumulated at A-Buffer by a single rendering. Fig. 2-(b) and Fig. 2-(c) show the result of the rendering. In A-Buffer, a fragment has two values into two color channels, Red and Green, to save surface information. The Red channel stores the *depth* value of the fragment, and the Green channel stores the *triangle ID* of the owner triangle of the fragment. Thanks to DirectX 10, triangle IDs are allocated by using *SV_PrimitiveID* semantic in shader without additional effort.

In order to compute PCSs, a simple test is invoked for the space between fragments of O_1 and O_2. If a fragment of O_1 and a fragment of O_2 are close enough, the triangle IDs are retrieved as a PCS. Given such PCSs, CPU computes the intersection points.

The result of PCS generation is recorded in render target texture. However most of texels in render target texture have no PCSs. To reduce the readback overhead, redundant data must be eliminated by stream reduction. Fortunately, geometry shader can trivially be used to remove unwanted elements from a stream of input, and the reduced elements are written to memory resource which can be copied to a staging resource for readback to the CPU.

3 Conclusion

Thanks to tremendous functionalities of GPU, the algorithm can handle dynamic objects and perform fast self-collision detection. The experimental result which tested with GeForce 8800GTS is described in Table 1.

Table 1: *Performance evaluation for Fig. 1 (times in ms).*

Scene	# of PCSs	# of Collision	CD time
(a)	12248	806	5.11
(b)	14196	1043	7.26
(c)	14000	1641	7.09
(d,e)	125	70	0.37

References

MYERS, K., AND BAVOIL, L. 2007. Stencil routed a-buffer. In *ACM SIGGRAPH 2007 sketches*, ACM, NY, USA, 21.