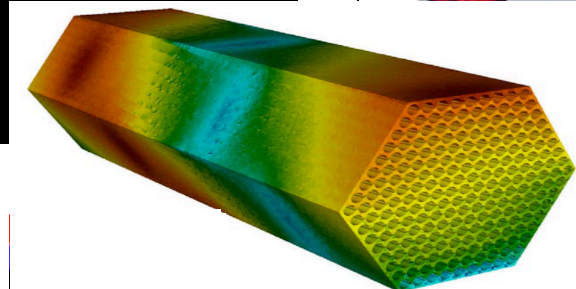
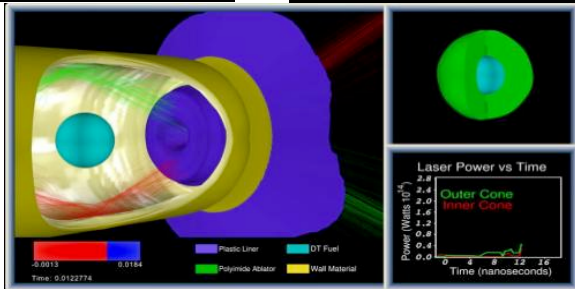
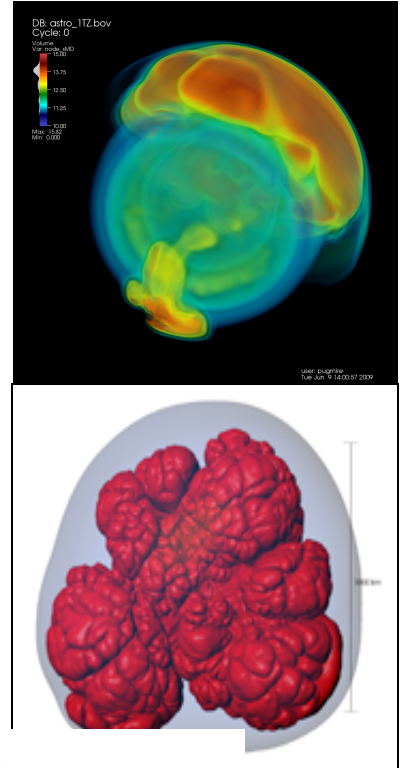
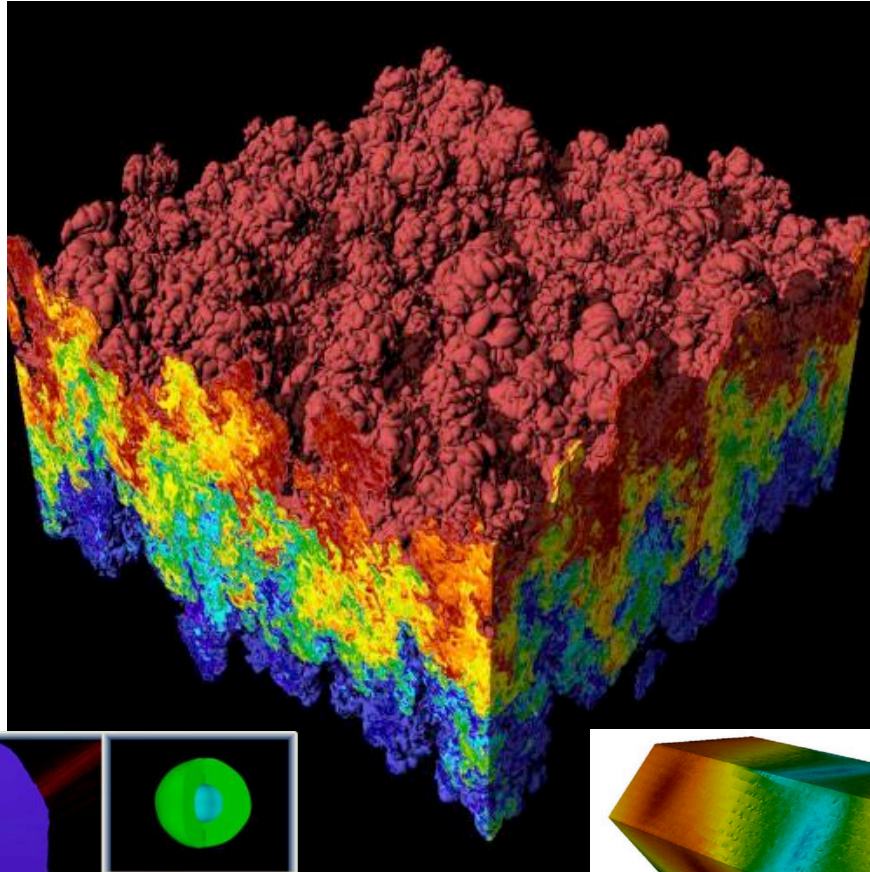
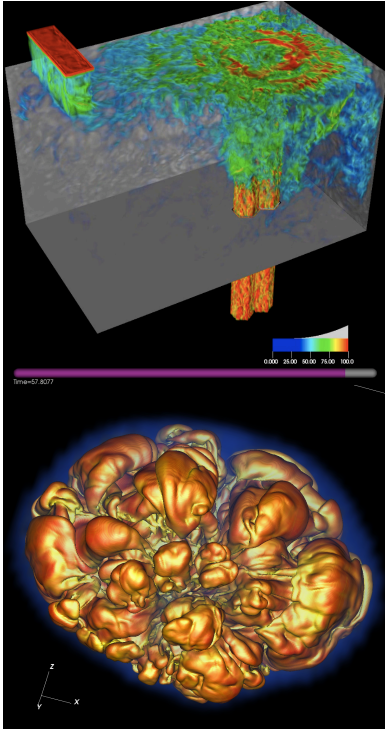


CIS 441/541: Intro to Computer Graphics

Lecture 14: Collisions and Level-of-Detail





Office Hours

- Now only 1 OH per week
 - Hank, Weds 230-330
- Abhishek still doing a lot...

3A

- If you make mistakes uploading textures, OpenGL often just defaults to the first texture

CIS 441

- □ ×



Wanted: colored by texture1 and tiger striped by texture 2

Got: colored by texture1 and tiger striped by texture 1

Project 3.X



- 3B (collisions) and 3C (level-of-detail) will be released today
- Other projects are in progress

Plan – Parentheticals Are Likely to Change



- This went well before, let's do it again

Week	Sun	Mon	Tues	Weds	Thurs	Fri	Sat
8		2B due	Lec13 (mouse+camera) (textures) 3A avail Proposals due		Lec14 (ray tracing) Quiz 4 (GL)		
9			Lec15 (textures) Live code 3B, 3C, ... avail		Quiz 5 (project 1D)		
10			More lecture		Quiz makeup		
Finals Week			Final Projects due All other work due: 1A-1F, 2A-2B not accepted after this point				

Plan for Thursday



- Start at 9am
- 9am-915am: Q&A on miscellaneous topics
- 915am-945am: Quiz 4
- Arrive no later than 910am



Quiz 4

- This quiz will test what you learned in Project 1D.
- Sorry to be reaching back to a project from a month ago, but there is a concept there I want to do the quiz on



3.X

- 3B: collisions
- 3C: level of detail



Collision Detection



Collision Detection

- Collision detection: as objects in the scene move, figure out when they collide and perform appropriate action (typically bouncing)
- Game setting: 30 FPS, meaning 0.033s to figure out what to render and render it.
 - Need to do this quickly!



Collision Detection

- Two flavors:
 - A priori
 - before the collision occurs
 - calculate the trajectory of each object and put in collision events right before they occur
 - A posteriori
 - after the collision occurs
 - with each advance, see if anything has hit or gotten close



How to Do Collision Detection: Brute Force

- For each object X
 - For each other object Y
 - Check if X and Y collide
- $\rightarrow O(n^2)$

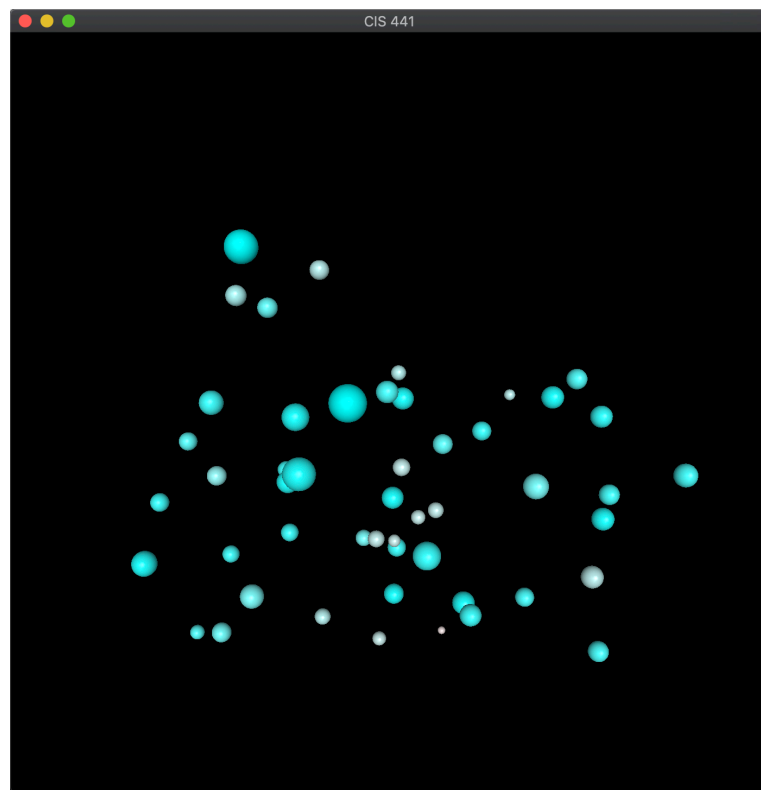


How to Do Collision Detection: Spatial Search Structures

- Divide volume into many cubes
- Place each object into its cube
- For each cube
 - Check to see if objects in cube have collided
- $\rightarrow O(n)$
 - ... Sort of / kind of / not really
 - How many cubes?
 - What is they all end up in the same cube
 - So maybe expected run time is $O(n)$.

3B

- Brute force on collision detection
 - Few enough objects that spatial search structures are not needed
- But nice effects for colliding balls



Level of detail (LOD) techniques



- level of detail: decreasing the complexity of some 3D object representations, because they
 - are far away
 - are moving fast
 - are not important
- increases the efficiency of rendering by decreasing the workload on graphics pipeline stages
 - reduced visual quality of the model is often unnoticed because of the small effect on object appearance when distant or moving fast

Types of LOD



- Two types:
 - Discrete LoD (DLoD)
 - Continuous LoD (CLoD)






Discrete LoD (DL0D)

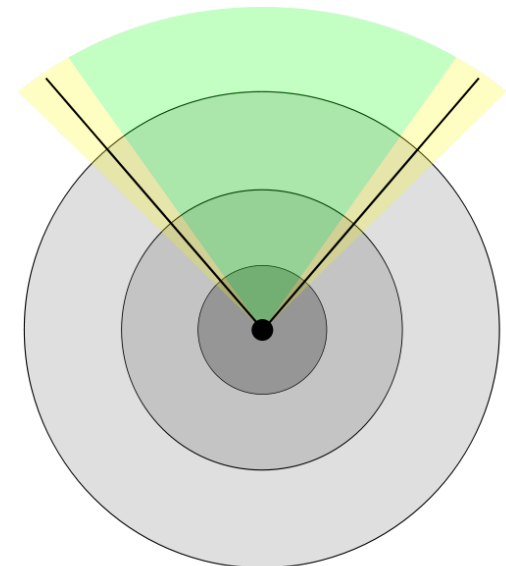


□ Discrete LoD (DLoD)

- Make a fixed amount of models, ranging from highest quality to coarse approximation & render appropriate one based on importance factor
- Fastest in practice, but leads to “popping”



Visual impact comparisons and measurements

Image					
Vertices	~5500	~2880	~1580	~670	140
Notes	Maximum detail, for closeups.				Minimum detail, very far objects.



Discrete LoD example



	Brute	DL0D
Rendered images	 A rendered scene with several spheres of varying sizes and a field of small black dots. The spheres are rendered with high detail, showing many small faces and sharp edges.	 The same rendered scene as the Brute image, but with significantly lower detail. The spheres are rendered with fewer faces, appearing smoother and less defined.
Render time	27.27 ms	1.29 ms
Scene vertices (thousands)	2328.48	109.44

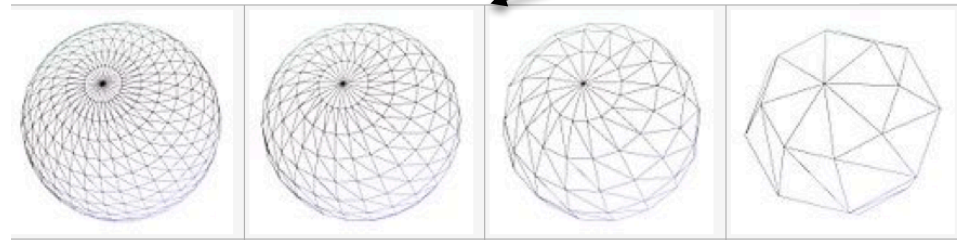
OK, how do we create coarse versions?



- How do we take



and make these?



- Answer: surface decimation (can lecture on this later)

- Have 3 levels of details for spheres
- Render closer spheres at high LOD, further away at low LOD

