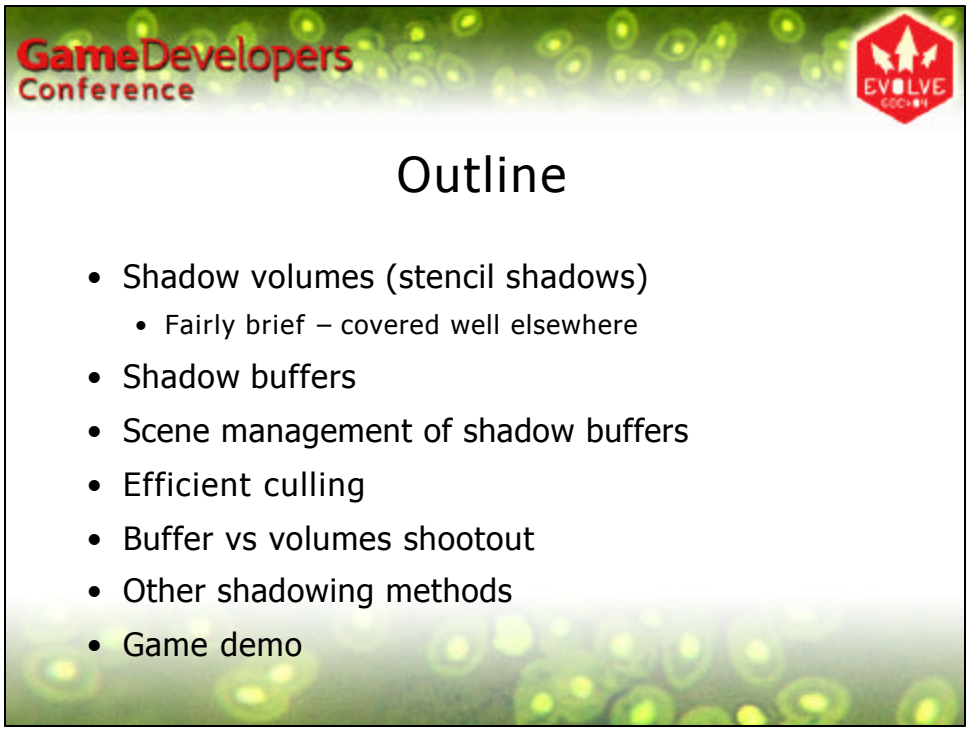

GameDevelopers Conference
EVOLVE
Practical Shadows
Out of the demo, into the engine
Tom Forsyth
RAD Game Tools


GameDevelopers Conference
EVOLVE
Outline
Shadow volumes (stencil shadows)
Fairly brief – covered well elsewhere
Shadow buffers
Scene management of shadow buffers
Efficient culling
Buffer vs volumes shootout
Other shadowing methods
Game demo

GameDevelopers Conference

EVOLVE

Shadow Volume Principles

- Extrude object shape away from light
- Cap ends to make a closed volume
- Count volume crossings to receiver
  - Front faces increment
  - Back faces decrement
  - Count=0 means surface is lit
  - Doesn't matter where you start counting!

GameDevelopers Conference

EVOLVE

How to Extrude: Software

- Brute force
  - Test every face, then every edge
- Follow silhouette edges around mesh
  - Have to work to spot extra loops
- External BSPs
  - Find which node light is in
- All require relatively low poly counts

GameDevelopers Conference

How to Extrude: Hardware

- Send vertex to infinity based on N.L>0
- Degenerate quads at every edge
  - ~6x vertex count, three unique per triangle
- No degenerates, use average normal
  - Needs high tessellation to avoid glitches
- Hybrid
  - Tessellate where mesh is smooth
  - Add degenerates at sharp edges

GameDevelopers Conference

Robustness Problems

- Volume clipped by near/far planes
  - Causes incorrect counting through holes
- Send far plane to infinity
  - Tiny loss in precision
- Switch to ZFAIL instead of ZPASS
  - Start counting from the back, not the front
- Lots of details in Everitt and Kilgard

GameDevelopers Conference

Problems with Volumes

- Gouraud shading chopped off
  - Test N.L>-epsilon instead
  - Epsilon can be computed per-vertex for static meshes, or just hand-tweaked
- Surface Acne
  - Z-fighting between volume and mesh
  - Move N.L<-epsilon vertices away instead
  - Moves Z-fighting to unlit side of mesh

GameDevelopers Conference

Problems with Volumes

- Large and unpredictable fillrate
  - Long volumes even from very small objects
  - Lots of overdraw from complex outlines
- Requires closed meshes
  - Some automated fixing possible
  - But always need some sort of artist help
- Needs hardware with stencil buffers
  - Dest. alpha can do it sometimes (PS2)

GameDevelopers Conference

EVOLVE

Problems with Volumes

- Animated meshes have artefacts
  - Where bones meet and/or blend
  - Skinned normals not normal to skinned faces, even with degenerates
  - Tessellate higher
  - Or do all/more work on CPU
- Cannot do shadows for alpha cutouts
  - No foliage, grills or chain-link fences

GameDevelopers Conference

EVOLVE

Shadow Buffers

- Render view from light to shadow buffer
- Store depth (or something like it)
  - Value is depth of frontmost (i.e. lit) object
- Render main view
- Project shadow buffer over scene
- Test pixel depth vs shadow buffer depth
- If equal, object is frontmost, and lit

GameDevelopers Conference

Depth Resolution

- Original method is store 24-bit Z
  - Sampling errors lead to "surface acne"
  - So use a bias to push Z value back a bit
  - But then causes "Peter Pan" problem – shadows become detached from casters
  - Limited hardware support
- Render back faces not front faces
  - "Second Depth" Maps - Wang & Molnar
  - Thin objects & silhouette edges still need bias
  - Makes Peter Panning much worse!

GameDevelopers Conference

ID Buffers

- Each object has an ID (unique integer)
- Write and compare IDs, not depths
- If IDs match – surface is lit
  - No maths, so no bias problems
  - 8 bit integers (255 IDs) often enough
  - Good hardware support (alpha test)

GameDevelopers Conference

ID Buffers

- Need multiple IDs per object to self-shadow
  - Convex sub-objects need unique IDs
  - Pre-processing work
  - Some objects (e.g. torus) can't be split perfectly
  - Or different ID per triangle – needs >8bits
- Neighbouring surfaces self-shadow
  - Sampling effect - "edge acne"
  - Sample surrounding texels as well
  - If *any* are equal, pixel is lit
  - Shrinks shadows by a texel

GameDevelopers Conference

Priority Buffers

- Same as IDs, but sorted by depth
- Sorting is not large CPU effort
  - PS1 did it (no Z buffer!)
  - Can be tricky in places e.g. hand holding mug
  - Circular sorts annoying, but very rare
- Can be filtered & mipmapped
  - Result of filtering is in sensible range
  - Reduces sparkling effect at minification
- Postprocess filter to solve edge acne
  - No fancy shader hardware needed

GameDevelopers Conference

Hybrid Priority & Depth

- Complex objects hard to assign IDs to
  - Especially animated ones
- Assign a range of IDs to the object
- Distribute IDs within object by depth
- Still needs bias
  - But per-object not per-scene, so easier
- Still no Peter Panning between objects

GameDevelopers Conference

Spatial Resolution

- Limited by memory and fill rate
- Ideally, texel size is proportional to distance from *camera*
- But shadow buffer projected from *light*
- Simple projection:
  - Objects further from light have larger texels
  - Distance from camera has no influence
  - "Duelling frustums" – camera faces light!

GameDevelopers Conference

EVOLVE

Perspective Shadow Maps

- Apply camera projection matrix to world
  - Squash view into (1,1,1)-(-1,-1,-1) clip cube
  - Things near camera grow in size
  - Straight lines still straight
- Render shadow buffer of distorted world
  - Things near to camera have more texels
- All uses existing 4x4 matrix pipeline
  - No hardware compatibility problems

GameDevelopers Conference

EVOLVE

Perspective Shadow Maps

- BUT!!! - special cases drive you nuts
- Directional lights become point lights
- Some lights become *sinks* not *sources*
- Objects outside frustum cast shadows
  - But they can be behind or intersect camera.z=0!
- Shadows shimmer and pop as camera moves
- Duelling frustums case still just as bad

"Friends don't let friends attempt to implement PSM"
- Matthew Rusch, EA

GameDevelopers Conference

EVOLVE

Trapezoid Shadow Maps

- Lots of different kinds
- Generalisation of PSM
- Arbitrary transform of world
  - But still must preserve straight lines
- All have singularities and special cases
- There is no single solution!
  - But some look better in some cases

GameDevelopers Conference

EVOLVE

Scene Management of Buffers

- Lights cannot share buffers
- Many lights cannot use just one buffer
  - Cases where PSM/trapezoid doesn't work
  - Point lights (need cubemaps or better)
  - Single buffer too large/not fine enough
- So need multiple buffers and frustums!
- Each buffer can be any type
  - Linear, PSM, floating-point, ID, etc

GameDevelopers Conference

EVOLVE

Multiple Frustums

- Each frustum has single shadow buffer
  - Might not render to all of it every frame
- Each object only uses one frustum/SB
  - (except for large/close objects)
  - But can be rendered *into* multiple frustums
    - Can cast shadows on objects in other frustums
- Each object knows size on screen
  - So knows how many SB texels per meter

GameDevelopers Conference

EVOLVE

Multiple Frustums

- Frustums chosen that:
  - Minimise number of frustums (speed!)
  - Provide SB as many objects as possible
  - No more than certain base angle, e.g. 120°
    - Otherwise texels too distorted
  - Smallest SB for required texels per meter
    - Highest of all the frustum's objects

GameDevelopers Conference

EVOLVE

Dynamic Frustum Choice

- Frustums recalculated every frame
- Each visible object looks for frustum
- Tries to enlarge existing frustums
  - Base angle constraint
  - SB size constraint
- If no suitable one exists, create new one
- Optimal, but shadows shimmer and pop

GameDevelopers Conference

EVOLVE

Persistent Frustum Choice

- Frustums persist from frame to frame
- Once created, frustums are static
  - May still render to subsections of SB
  - Checks bounding box of *visible* objects
- Each object checks if frustum still good
  - Camera may have come closer
  - Object may have moved

GameDevelopers Conference

EVOLVE

Persistent Frustum Choice

- If frustum no good, make a new one
  - Frustum tries to fit all unassigned objects, not just visible ones – prevents lots of frustums being created as camera pans
- Frustums with no objects deleted
- Each frame redo a few random objects
  - Slowly cleans up sub-optimal frustums, e.g. when the camera moves backwards

GameDevelopers Conference

EVOLVE

Frustum Types

- Moving lights use fully dynamic
  - Frustums *must* move with light anyway
  - So make it optimal speed, lowest cost
- Static lights use persistent
  - Static camera, light, object = no shimmers
  - Animated objects just pop on change
  - Rigid objects cross-fade on change
    - If possible – old frustum may not be big enough

GameDevelopers Conference

EVOLVE

Special Cases

- Large/close-to-light objects are special
  - Cover more than 120°
  - No single frustum will fit it
  - Draw with multiple frustums
  - Slower multi-pass rendering
  - Rare – usually only one per light or less
  - Still work normally as shadow *casters*
  - Can just draw them not shadowed at all

GameDevelopers Conference

EVOLVE

Multiple Frustums: Cool Things

- Duelling frustums works
- Point lights work
- Orthogonal to type of SB
- Fully adaptive
  - Use PSM/trapezoidal if it works *that frame*
  - Use 8/16/24/32 bit depth as needed
  - Custom frustums/SBs (e.g. hero) easy

Culling

- Vital early out for many scenes
- Helps both buffer & volume methods
- Indoor: lots of lights, but also lots of occluders
- Outdoor day: one light – the sun
- Outdoor night:
  - Large spotlights – small frustum angle
  - Small omnis – limited range
- Culling allows range cap on volumes
  - Reduces heaviest fillrate demand

GameDevelopers Conference
EVOLVE

Culling

- Similar problem to visibility culling
  - So use same portals, PVS, occluders, etc
- Remove caster objects that:
  - are not visible from the light
  - don't occlude any receivers in view
  - occluded from all receivers in view
  - unless they are also receivers in view!

GameDevelopers Conference

Shadow Shootout

	Buffers	Volumes
Fillrate	Win (usually)	Lose
Geometry	Lose	Win (usually)
Complexity	Draw!	Draw!
Aliasing	Lose	Win
Art flexibility	Win	Lose
Hardware	Win	Lose
Artist load	Win	Lose

GameDevelopers Conference

Other Shadow Methods

- Shadow maps (no depth info)
  - One per person
  - Cheat and use orthographic projection
- Blob shadows
  - Better looking in very ambient scenes
- Static light maps
- Static "property maps"
  - Light direction and colour per vertex
  - Monochrome shadow/light map
  - Does correct specular, normal maps, etc.

GameDevelopers Conference

Self Shadowing

- Difficult for both buffers and volumes
  - Do with more local methods instead
- Diffuse occlusion – "grubbiness map"
- Cone/ellipsoid of visibility
- SH self-occlusion
  - Generalisation of both cone & grubbiness
- Horizon maps
- All can be per-vertex or per-pixel

GameDevelopers Conference

Game Demo

- "StarTopia" by Muckyfoot Productions
- Released in 2001
- Not designed for shadows at all
- No artwork changes
- DX7 technology – no shaders
- Dynamic environment
  - Player builds most structures
  - Lots of AIs running around
- Lots of lights

GameDevelopers Conference

Useful References

- "Practical & Robust Stenciled Shadow Volumes for Hardware-Accellerated Rendering" - C. Everitt & M. Kilgard (GDC 2002)
- Priority buffers: "Algorithms for Antialiased Cast Shadows" – J. Hourcade & A. Nicolas
- "Second-Depth Shadow Mapping" – Y. Wang & S. Molnar
- "Practical Priority Buffer Shadows" – S. Dietrich (Games Programming Gems 2)
- "Perspective Shadow Maps" – M. Stamminger & G. Drettakis

GameDevelopers Conference

Questions?

Latest errata, hindsights, etc:

www.eelpi.gotdns.org

tomf@radgametools.com