GDC 12

GAME DEVELOPERS CONFERENCE 2012

SAN FRANCISCO, CA
MARCH 5-9, 2012
EXPO DATES: MARCH 7-9

Lighting & Simplifying *Saints Row: The Third*

Scott Kircher
Volition, Inc.

Saints Row: The Third

Saints Row 2 vs. "*The Third*"

- Nighttime flight in *Saints Row 2*

Saints Row 2 vs. "*The Third*"

Main Topics

- Latest iteration of inferred lighting
  - *Saints Row: The Third* vs. *Red Faction: Armageddon*
  - New optimizations and features

- Automated LOD Pipeline
  - Mesh simplification
  - Practical implementation issues

Main Topics

- Latest iteration of inferred lighting
  - *Saints Row: The Third* vs. *Red Faction: Armageddon*
  - New optimizations and features

- Automated LOD Pipeline
  - Mesh simplification
  - Practical implementation issues

The light! It blinds me!

INFERRED LIGHTING, THE NEXT ITERATION

Inferred Lighting, Related Work

- Developed at Volition, Inc.
- Originally published in SIGGRAPH 2009
  - [Kircher, Lawrance 2009]
- Version used in *Red Faction: Armageddon*
  - Presented at GDC last year [Flavin 2011]
- Variation of Deferred Lighting/Light-prepass
  - [Engel 2008]
  - [Lee 2008]
  - And many others

Inferred Lighting Refresher

Inferred Lighting Refresher

Low-res MRT Geometry Pass
(800x450 on consoles)

Inferred Lighting Refresher

Low-res Lighting Pass
(800x450)

Inferred Lighting Refresher

Low-res Lighting Pass (800x450)

Full-res Material Pass (1280x720, 2x MSAA)

Saints Row: The Third vs. *Red Faction: Armageddon*

- RF:A
  - Single resolution
    - 960x540
  - Discontinuity “patching”
- SR:TT
  - Multi-resolution
    - 800x450 for lighting
    - 1280x720 for main scene
  - Bilinear Discontinuity Sensitive Filter

Saints Row: The Third vs. *Red Faction: Armageddon*

- RF:A
  - Single resolution
    - 960x540
  - Discontinuity “patching”
- SR:TT
  - Multi-resolution
    - 800x450 for lighting
    - 1280x720 for main scene
  - Bilinear Discontinuity Sensitive Filter

Inferred Lighting Features

- Existing (SIGGRAPH 2009 & GDC 2011)
  - Lots of fully dynamic lights
  - Integrated alpha lighting (no forward rendering)
  - Hardware MSAA support (even on DX9)
- New
  - Lit rain (*IL required*)
  - Better foliage support (*applies only to IL)*
  - Screen-space decals (*enhanced by IL*)
  - Radial Ambient Occlusion (RAO) (*optimized by IL*)

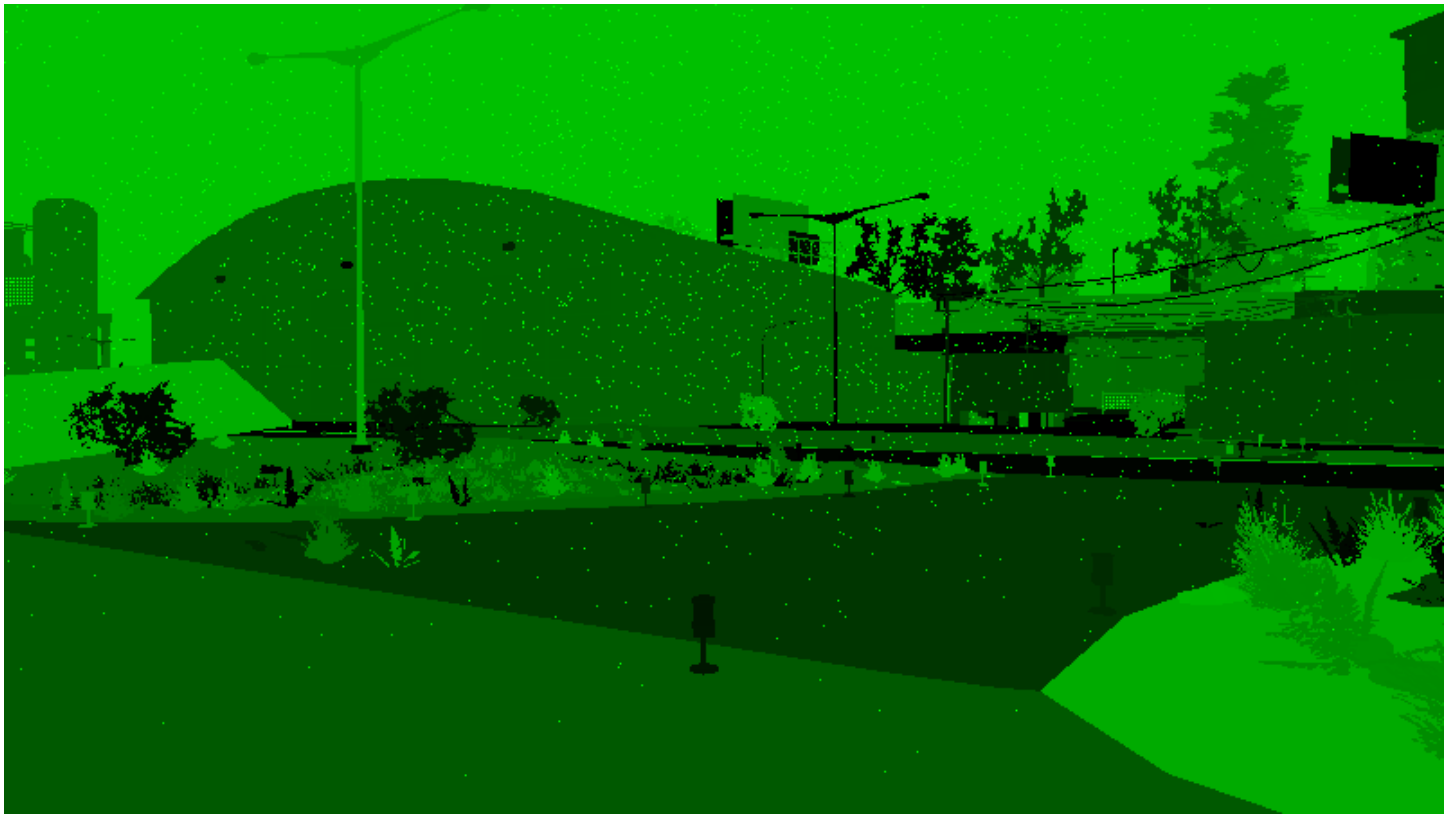
Lit Rain

94 visible lights. 10,000 rain drops.

Xbox 360: 34fps Dedicated rain lighting time: 0.6 ms

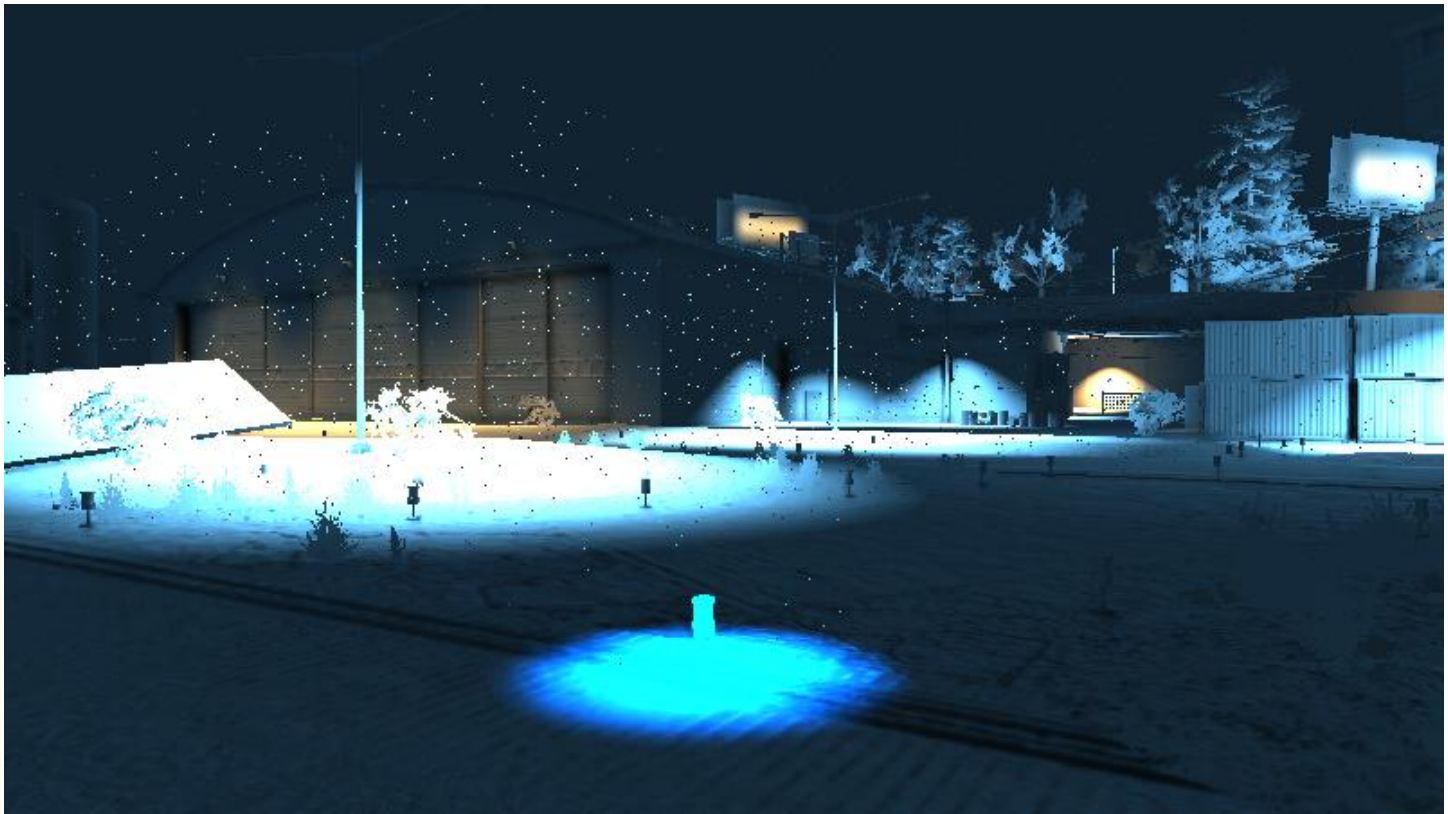
Lit Rain: Step 1

- Render single pixel per rain drop into G-buffers



Lit Rain: Step 2

- Lighting pass lights rain “for free”



Lit Rain: Step 3

- DSF automatically ignores rain samples

Lit Rain: Step 4

- Rain drops look up their lighting sample

Lit Rain: Normals

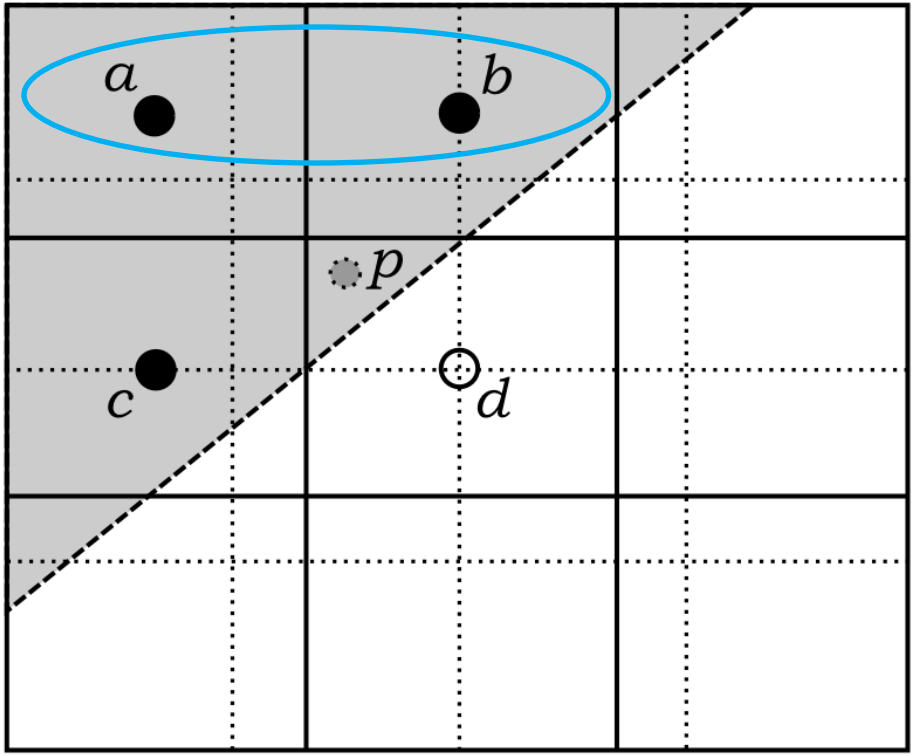
- Choosing a good normal for rain is difficult
  - Only one per rain drop
  - Water is translucent
- Decided to just use the world “up” vector
  - Most city lights are up high pointing down
  - Other lights still work because our lighting model is “half-Lambert” [Mitchell *et al.* 2006]
  - Could use special code in light shaders to remove normal influence altogether

Lit Rain: Car Headlights

Foliage

- Inferred lighting assumes low scene depth complexity to keep DSF cost bounded
- Foliage breaks that assumption

Faster DSF for Foliage



Foliage DSF Results

- Full DSF. PS3 Scene GPU time: 35.7ms

Foliage DSF Results

- 2-sample DSF. 33.7ms on PS3 (2ms saved)

Foliage DSF Artifacts

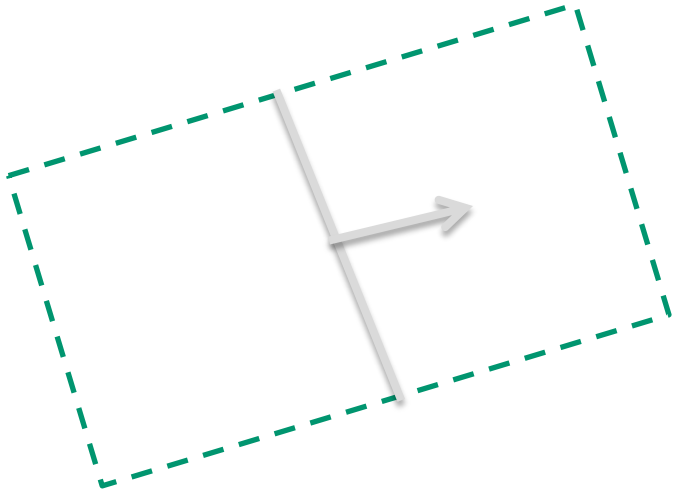
Dynamic Decals

Recent History of Decals at Volition

- *Saints Row 1* & *2*
  - Collect decals geometry from mesh at collision
  - Slow at creation, fast to render
  - Problematic on PS3 due to VRAM CPU restrictions
- *Red Faction: Guerrilla* & *Armageddon*
  - Re-render (sub)mesh for each decal
  - Fast creation, but potentially slow to render
    - Worked well with small mesh chunks created by destruction system

Screen-space Decals

- *Saints Row: The Third*
  - Volumetric decals applied in screen-space
  - Use DSF ID to restrict decals to specific objects



Importance of DSF ID for Decals

Importance of DSF ID for Decals

Importance of DSF ID for Decals

Radial Ambient Occlusion (RAO)

Without Radial Ambient Occlusion

RAO Details

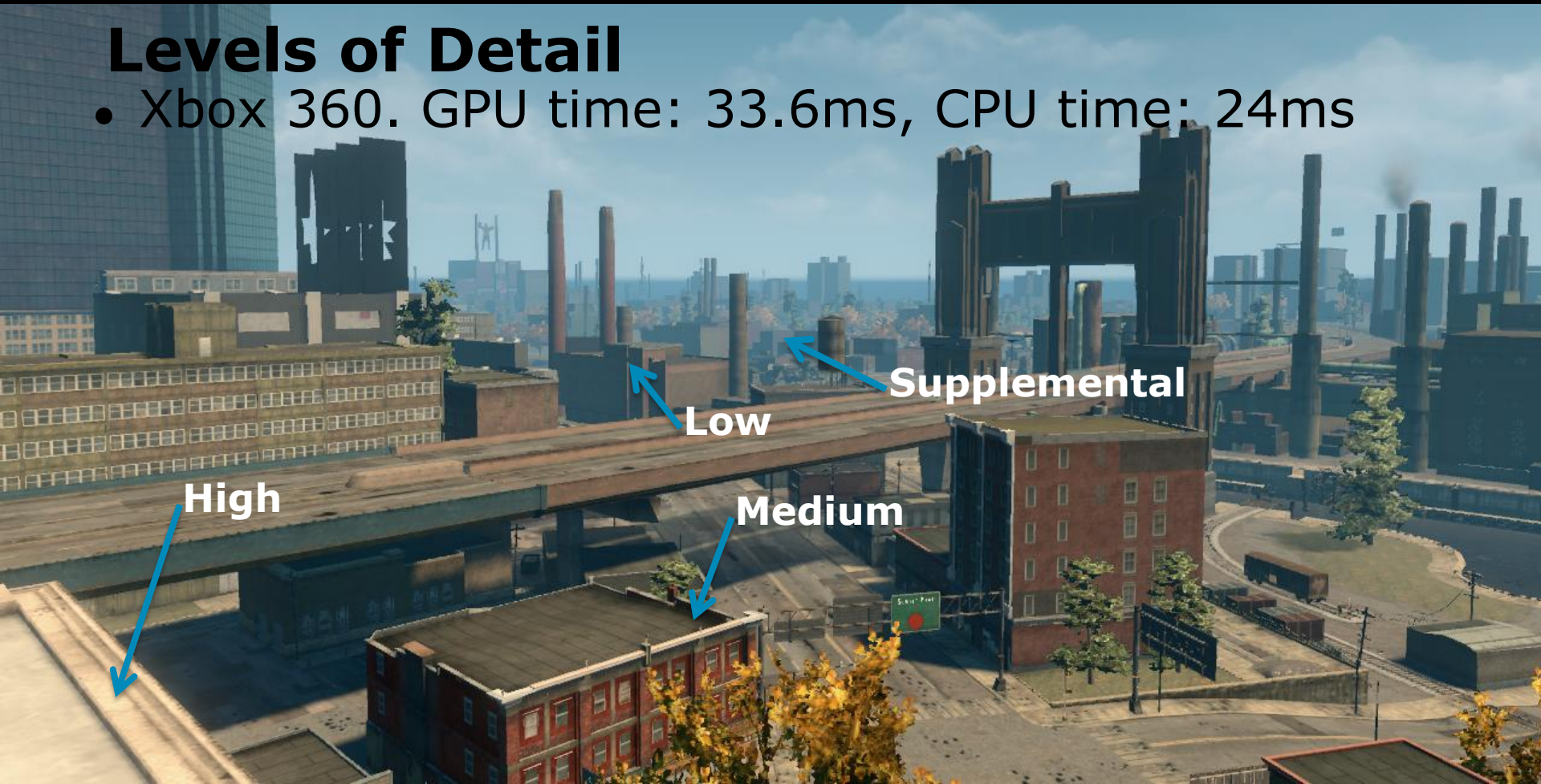
- Based loosely on [Shanmugam & Arikan 2007]
- Occlusion factor is based on normal and distance to box or ellipsoid
  - Very much like a regular light
  - Occlusion factor used to modulate lighting
- For vehicles, artist places box approximating vehicle body
- For humans, ellipsoids placed automatically at feet

And now for something (almost) completely different…

MESH SIMPLIFICATION

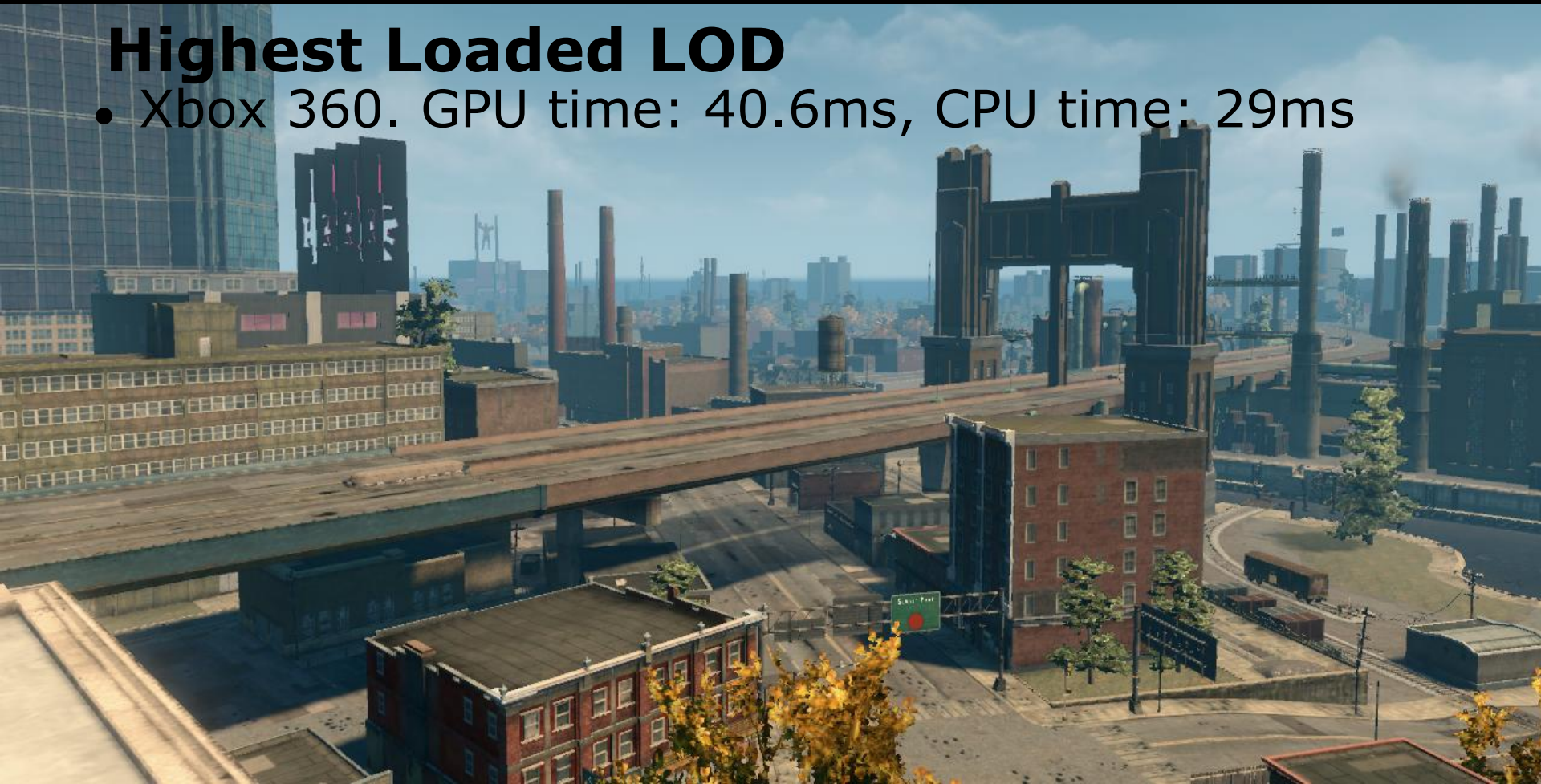
Levels of Detail

- Xbox 360. GPU time: 33.6ms, CPU time: 24ms



Highest Loaded LOD

- Xbox 360. GPU time: 40.6ms, CPU time: 29ms



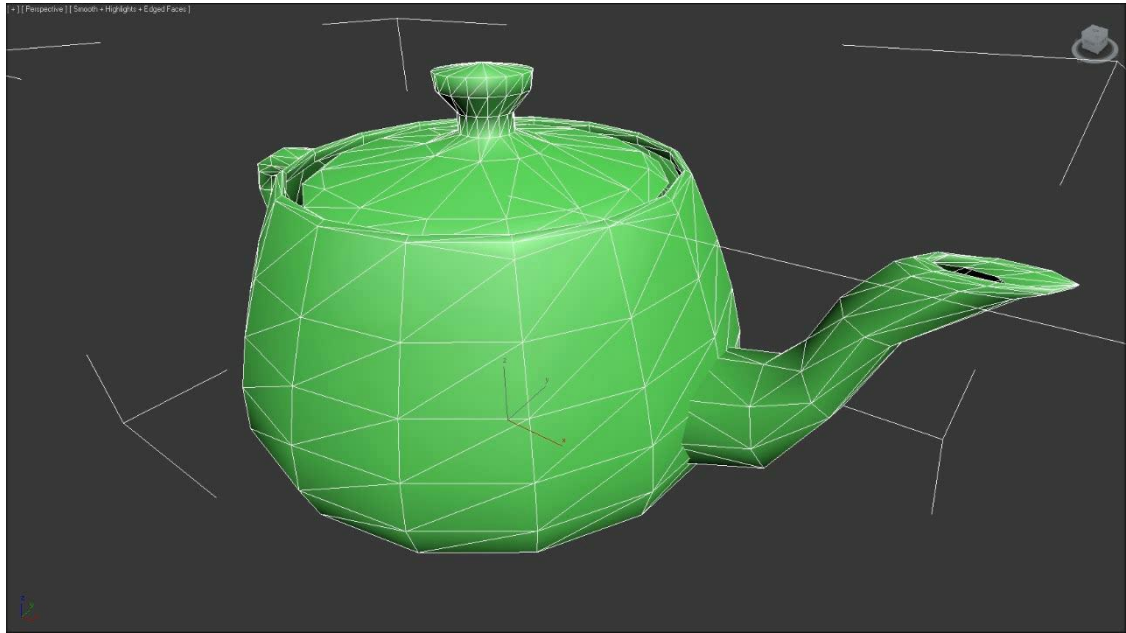
LOD Generation, the Old Way

- *Saints Row 2* LOD generation
  - Mostly artist authored
  - Time consuming for artists
    - Not many LODs actually created
    - Mostly opted for fading in “detail sets”

LOD Generation, the New Way

- *Saints Row: The Third* style:
  - Implemented our own full featured mesh simplifier
  - Runs in *crunchers,* **not** in DCC application

(Results can be previewed in 3D Studio Max)

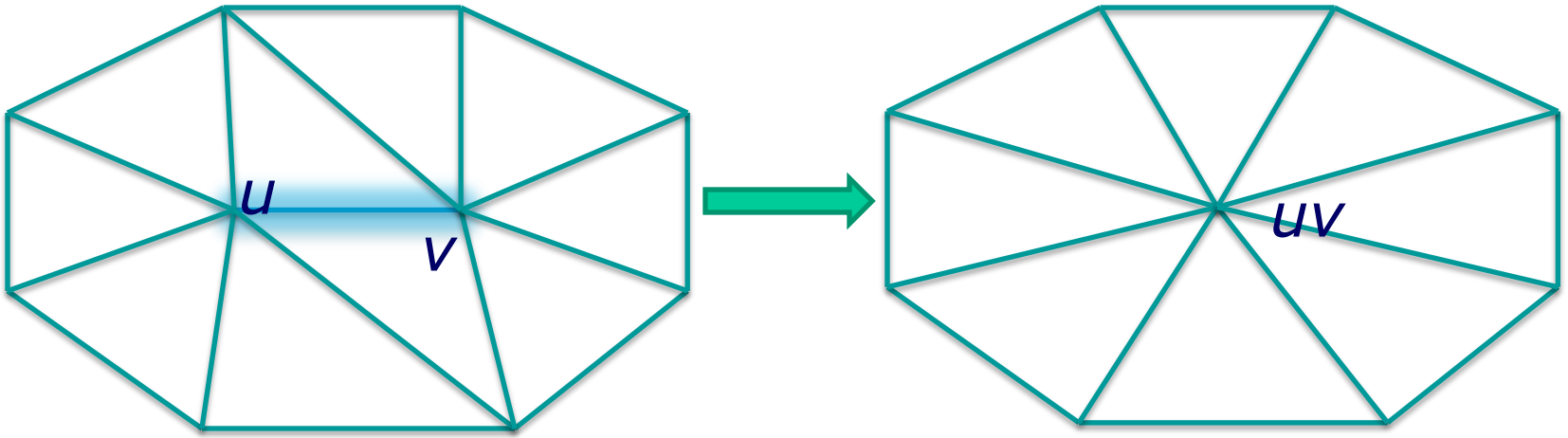


What We Used It For

- Mostly auto generated LODs, but artist tweakable:
  - Buildings
  - Characters
  - Vehicles
- Completely automated (no artist intervention):
  - Terrain
- Also used simplifier for generating:
  - Terrain collision hull
  - Building shadow proxies

Mesh Simplification

- Iterative Edge Contraction
  - Garland's Quadric Error Metric (QEM)
    - [Garland & Heckbert 1997]
  - Attribute Preservation
    - [Hoppe 1999]

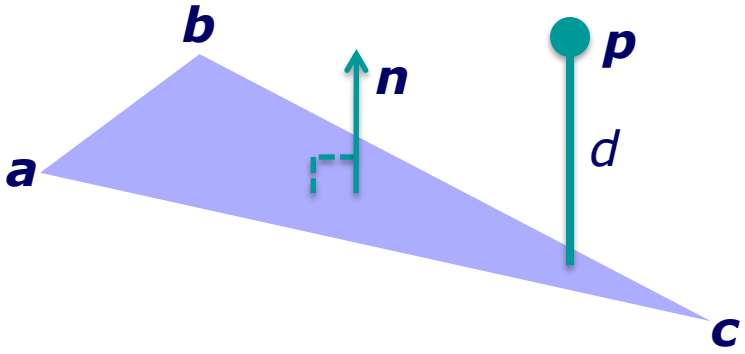


Error Metric

- An *error metric* measures how "bad" the mesh approximation is.
  - Used to compute the *contraction error*

- Determines
  - Which edge to contract first
  - Where to place resultant vertex

Quadric Error Metric Overview



Homogeneous coordinates

$$\boldsymbol{P} = (p_x, p_y, p_z, 1)$$

$$\Pi = (n_x, n_y, n_z, -\boldsymbol{a}\cdot\boldsymbol{n})$$

$$d^2 = (\boldsymbol{P}\cdot\Pi)^2$$

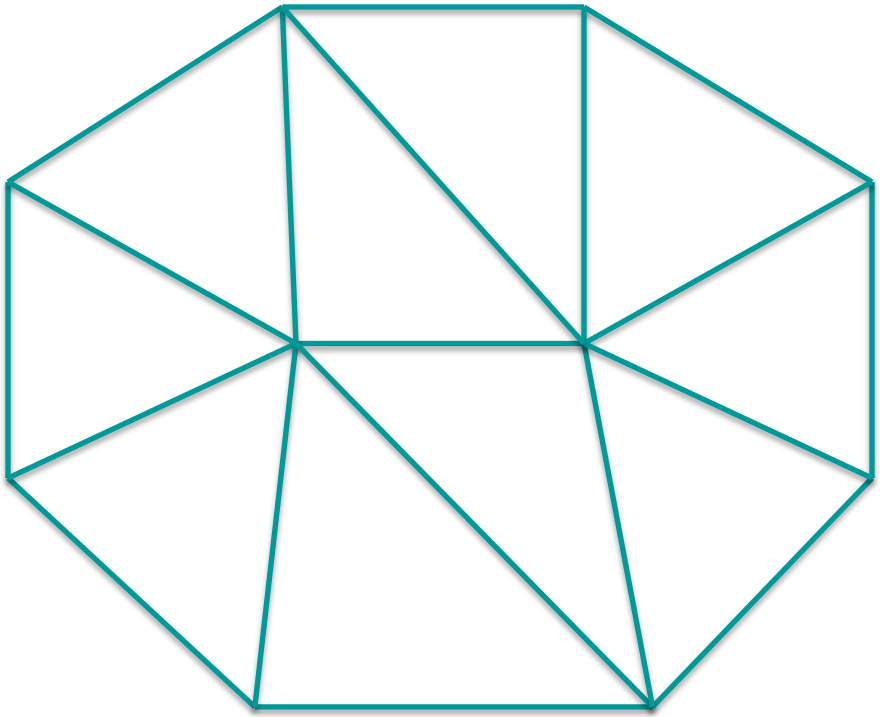
$$= \boldsymbol{P}(\Pi^{\mathrm{T}}\Pi)\boldsymbol{P}^{\mathrm{T}}$$

$$= \boldsymbol{P}\mathbf{Q}\boldsymbol{P}^{\mathrm{T}}$$

"Quadric" matrix

Mommy, Where Do Quadrics Come From?

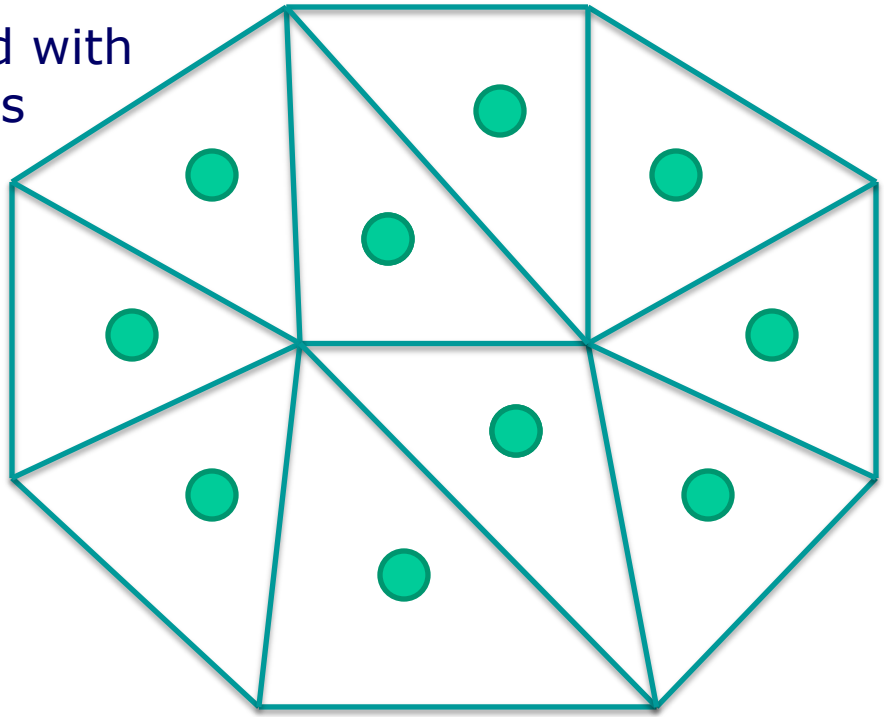
- Each original triangle defines a plane



Mommy, Where Do Quadrics Come From?

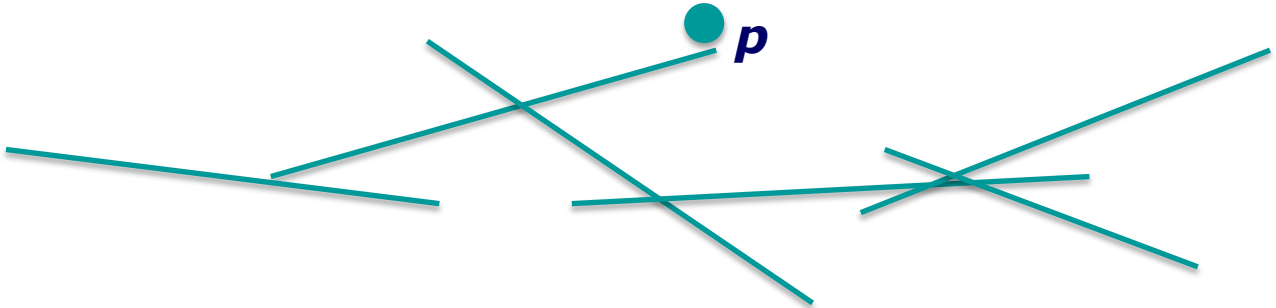
- Each plane defines a quadric

Quadrics associated with neighboring vertices



Using the Quadric Matrix

- Matrix **Q** can represent an entire set of planes

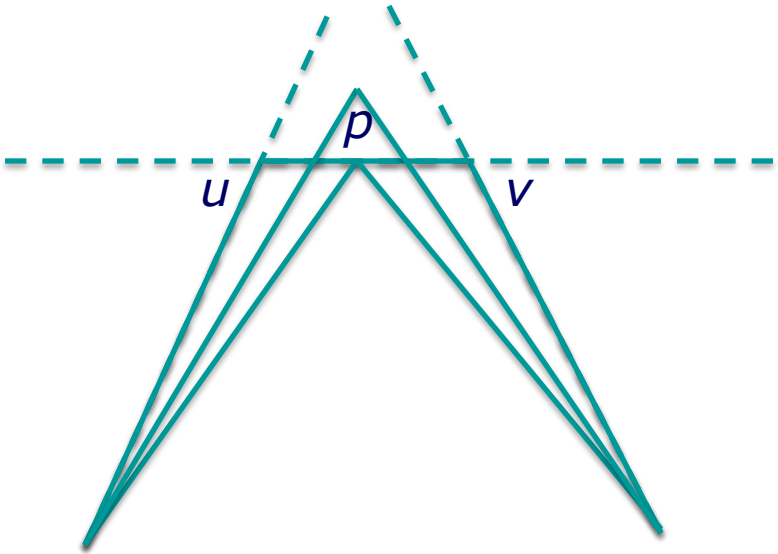


$$E = \boldsymbol{P}(\sum \mathbf{Q}_i)\boldsymbol{P}^{\mathrm{T}}$$

- At each edge contraction, quadrics are summed to get new quadric

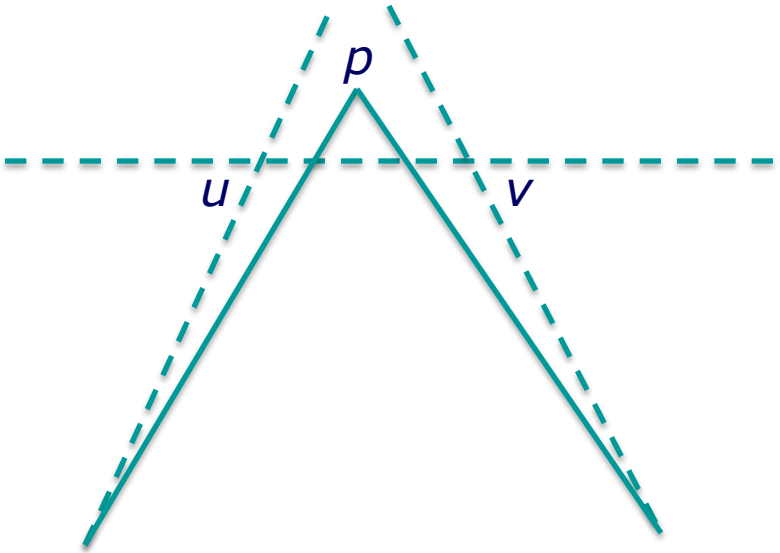
Vertex placement

- Consider contracting edge (*u*,*v*)
  - "Edge-on" view of triangle planes



Optimal Vertex Position

- After contracting edge (*u*,*v*):
  - Want to place vertex *p* to minimize error



$$E = \mathbf{P}\mathbf{Q}\mathbf{P}^{\mathrm{T}}$$

Minimize *E*

$$\mathbf{P} = \mathbf{o}\mathbf{Q}^{-1}$$

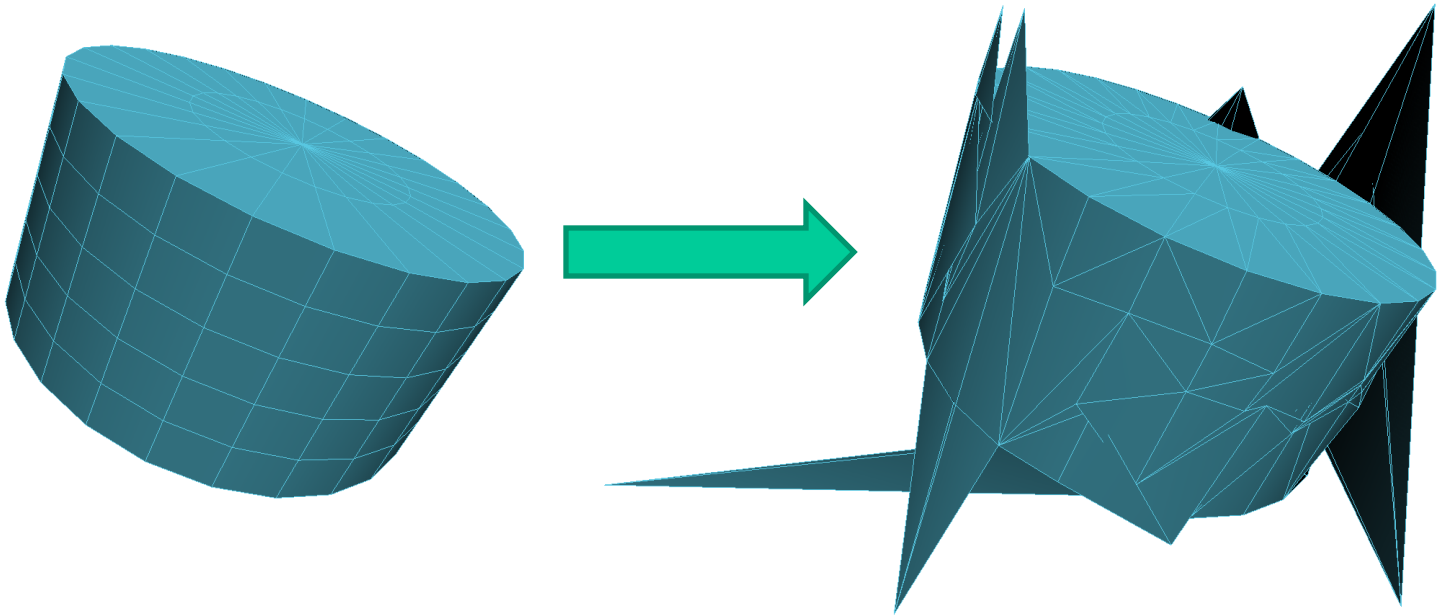
Where $\mathbf{o} = (0,0,0,1)$

Practical Implementation Issue #1

- Numerical precision & stability
  - *Use double precision floats*

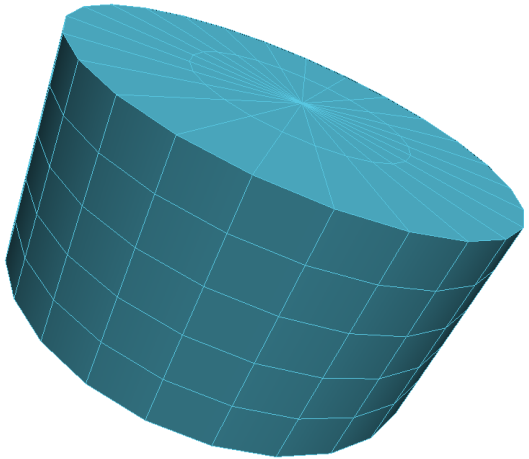
Practical Implementation Issue #1

- Numerical precision & stability
  - *Use double precision floats*
  - But double precision doesn't help with this:



Singular Quadrics

- Cannot always invert **Q**
  - Such *singular* matrices are obvious
    - Inversion algorithm will fail or produce NANs
  - Caused by flat or cylindrical areas

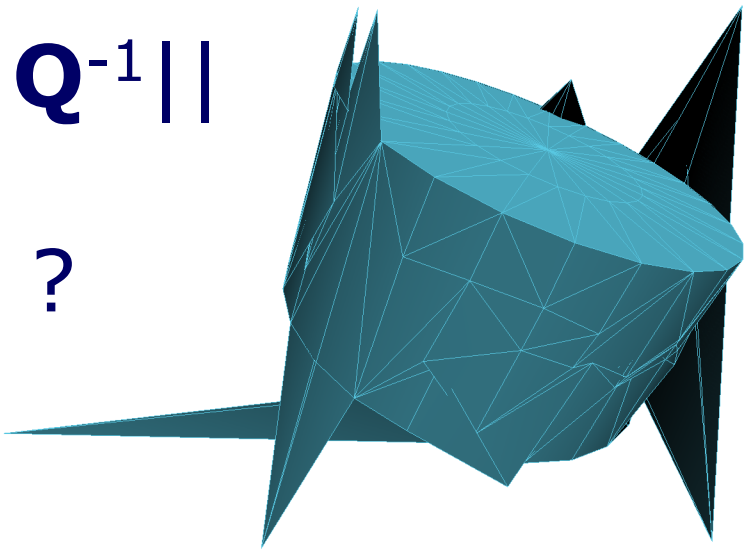


Ill-Conditioned Quadrics

- Even if $\mathbf{Q}^{-1}$ exists, result might be bad
  - if **Q** is "nearly-singular"
- Can detect by checking *condition number* of **Q**

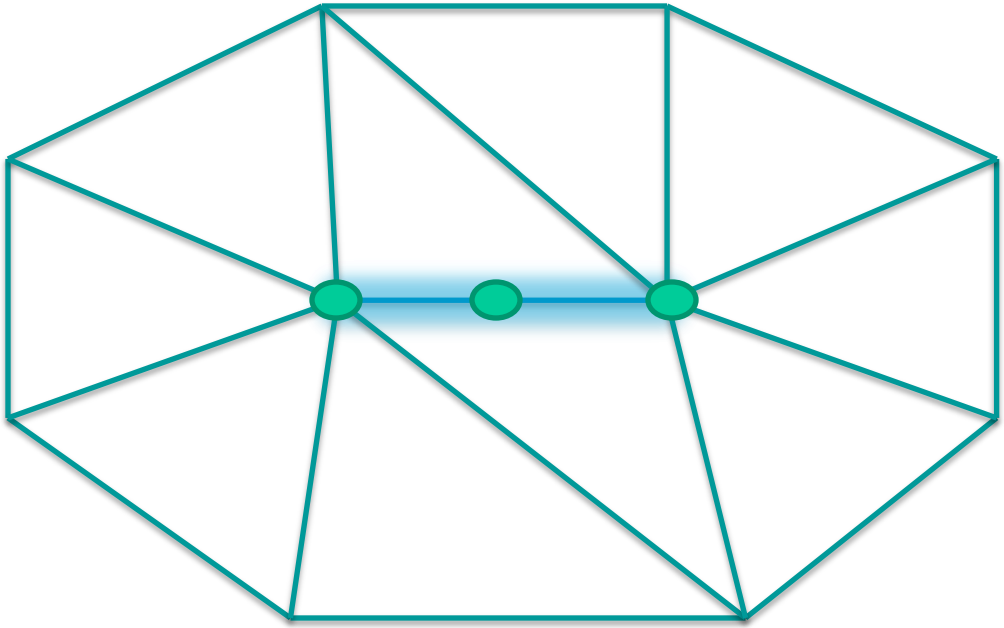
condition number $= ||\mathbf{Q}|| * ||\mathbf{Q}^{-1}||$

condition number > threshold ?



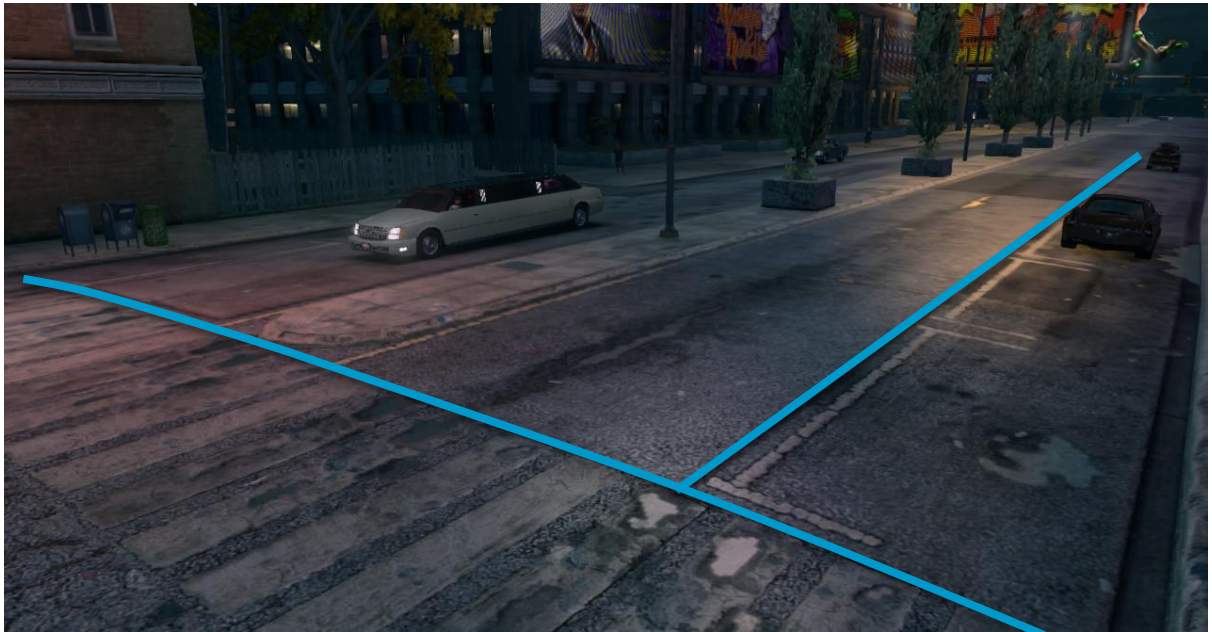
Handling Bad Conditioning

- Select best position from “candidates”
  - Lowest quadric error

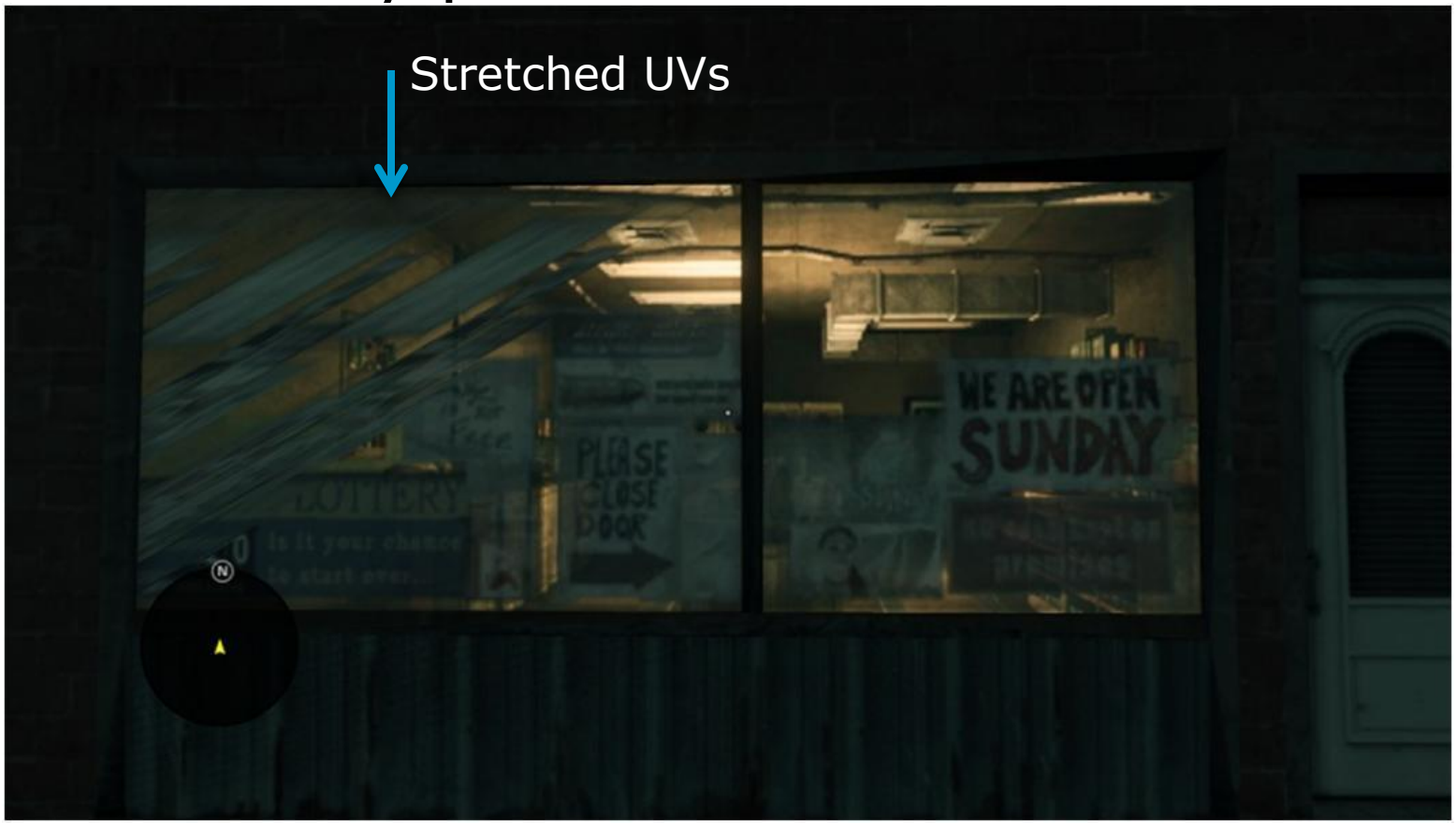


Practical Implementation Issue #2

- Texture coordinate (UV) preservation
  - See [Hoppe 1999]
  - Practical issues have to do with boundaries

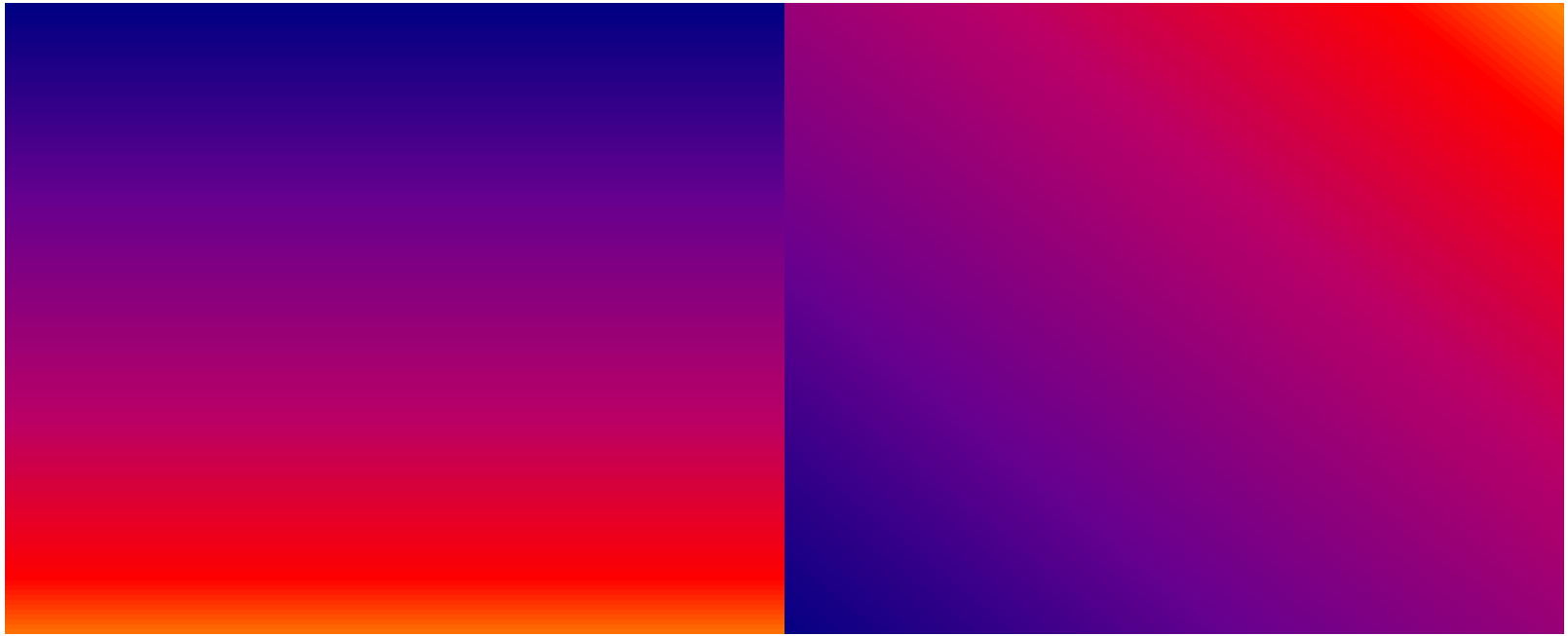


UV boundary problems



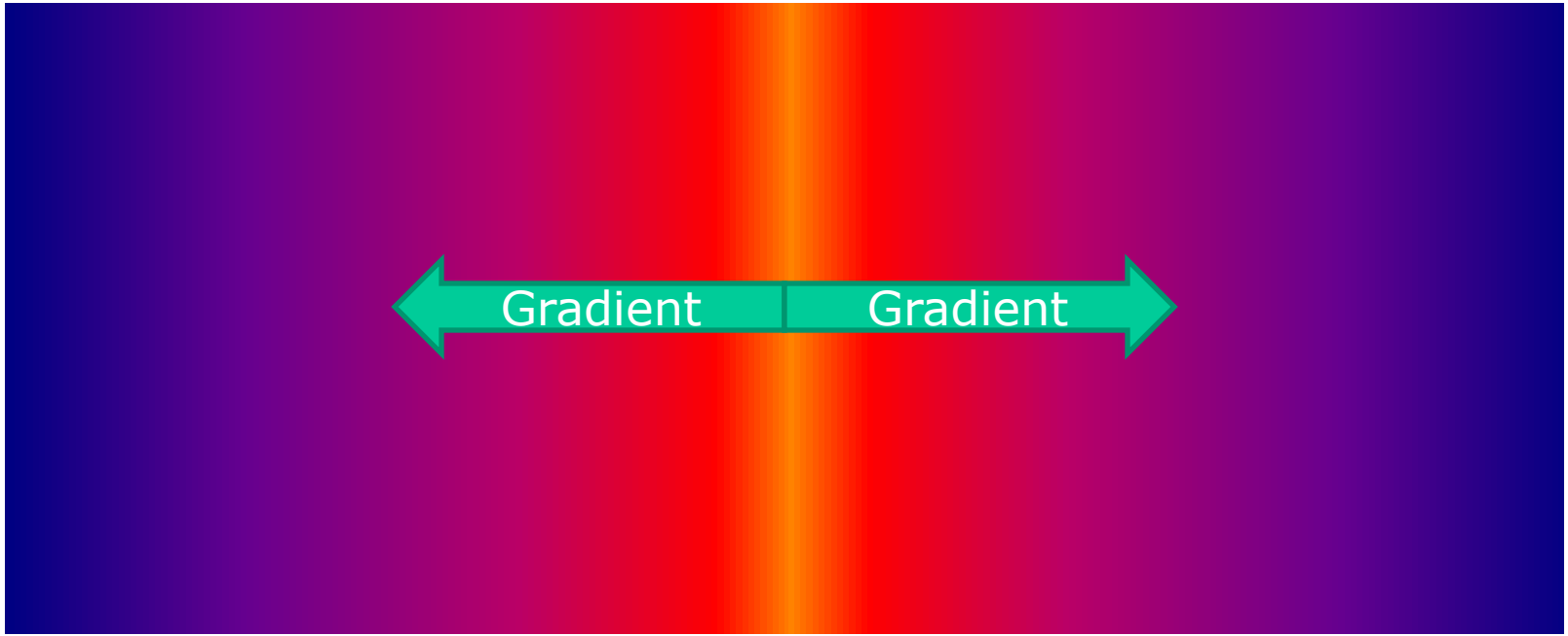
UV Boundary Types

- Obvious: UV Discontinuities



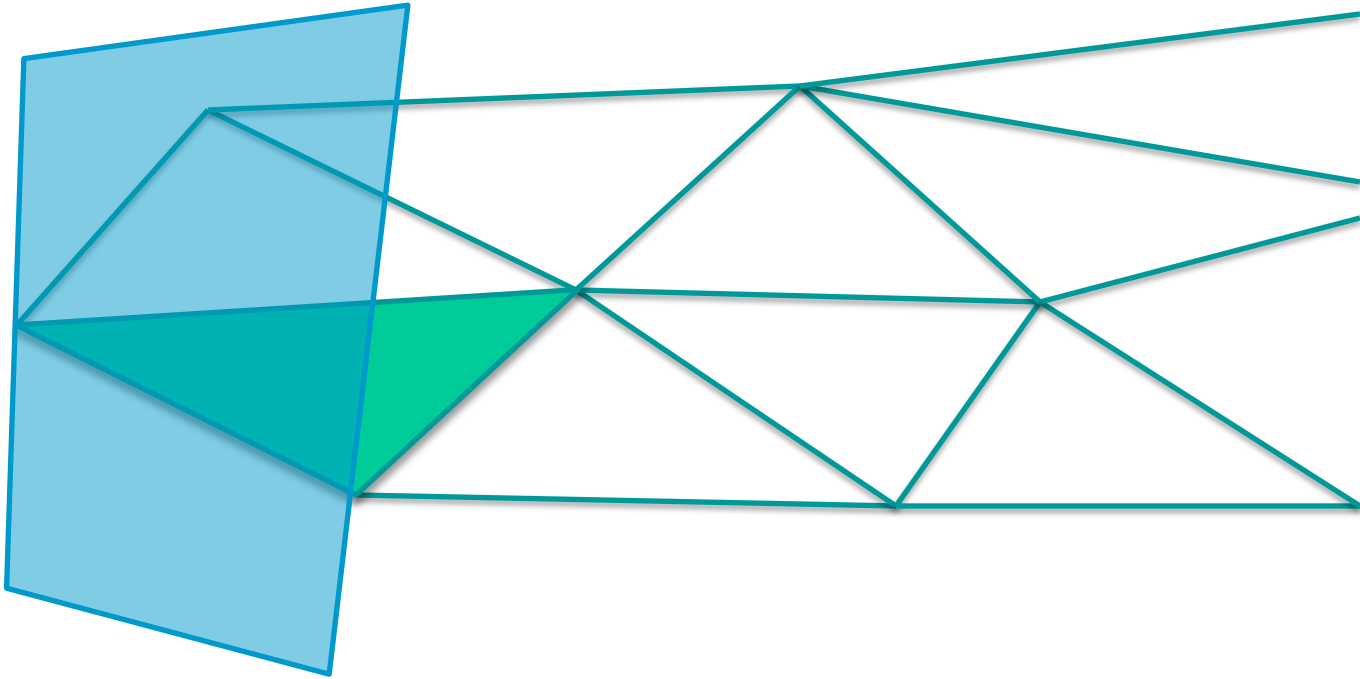
UV Boundary Types

- Not-so-obvious: UV Mirroring



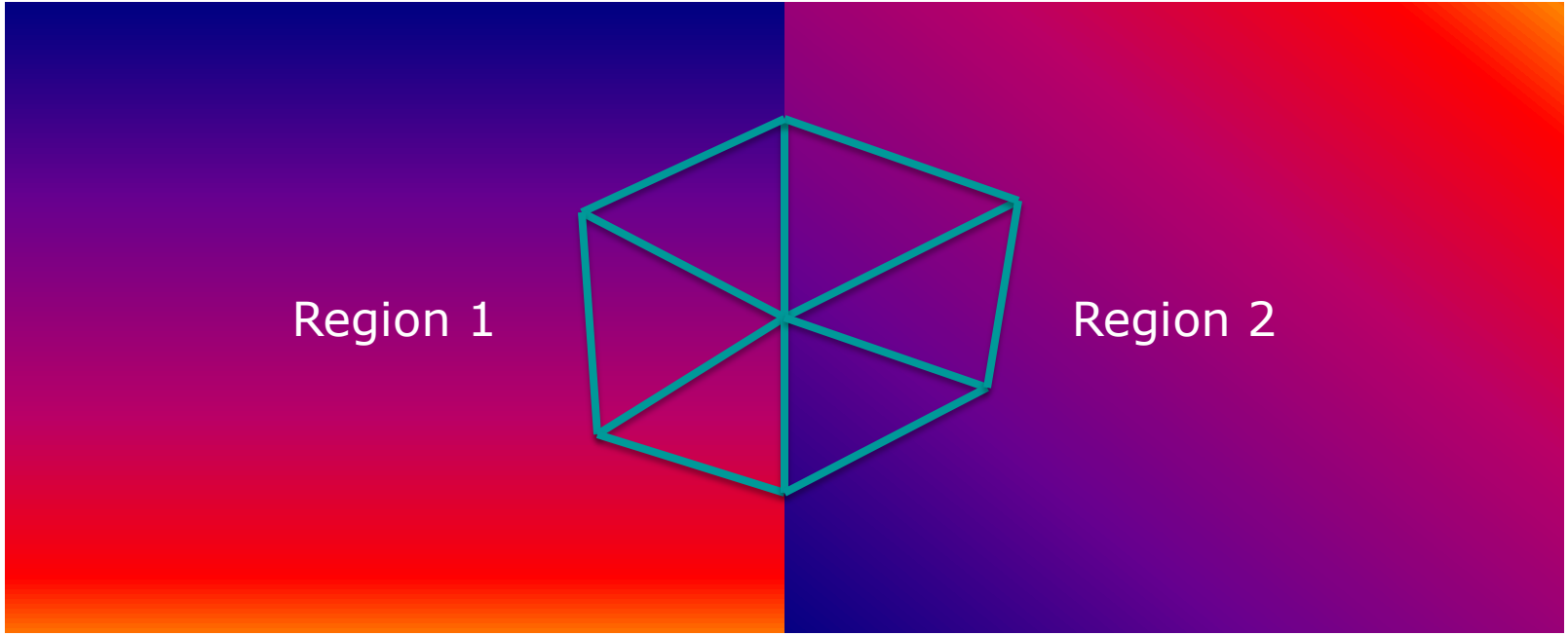
Preserving Boundaries

- Boundaries of any type preserved same way
  - Add “virtual” plane through boundary edge



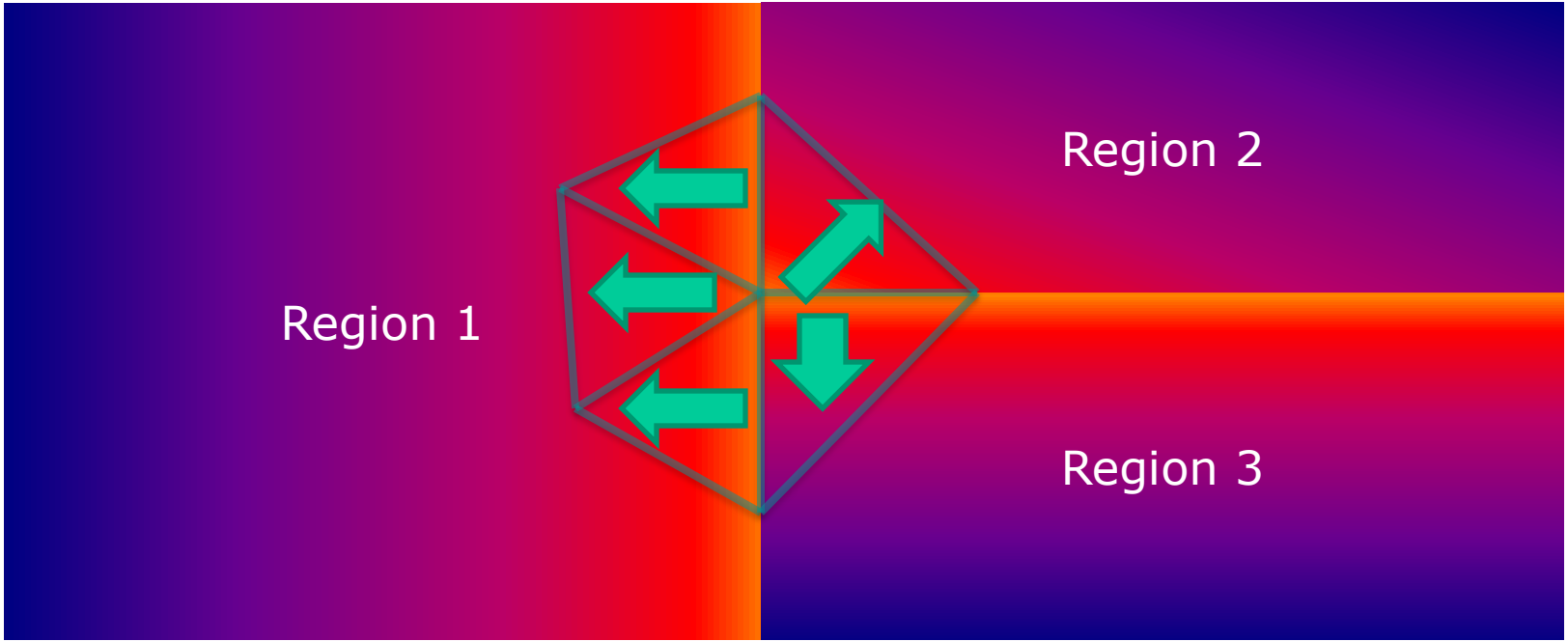
Continuous Regions

- At each vertex
  - Track regions that have continuous UVs



Continuous Regions Gotcha

- UVs may be continuous at a vertex…
  - Even though the regions are separate



Practical Implementation Issue #3

- Material counts
  - As LODs get simpler, material costs dominate

Reducing Material Counts

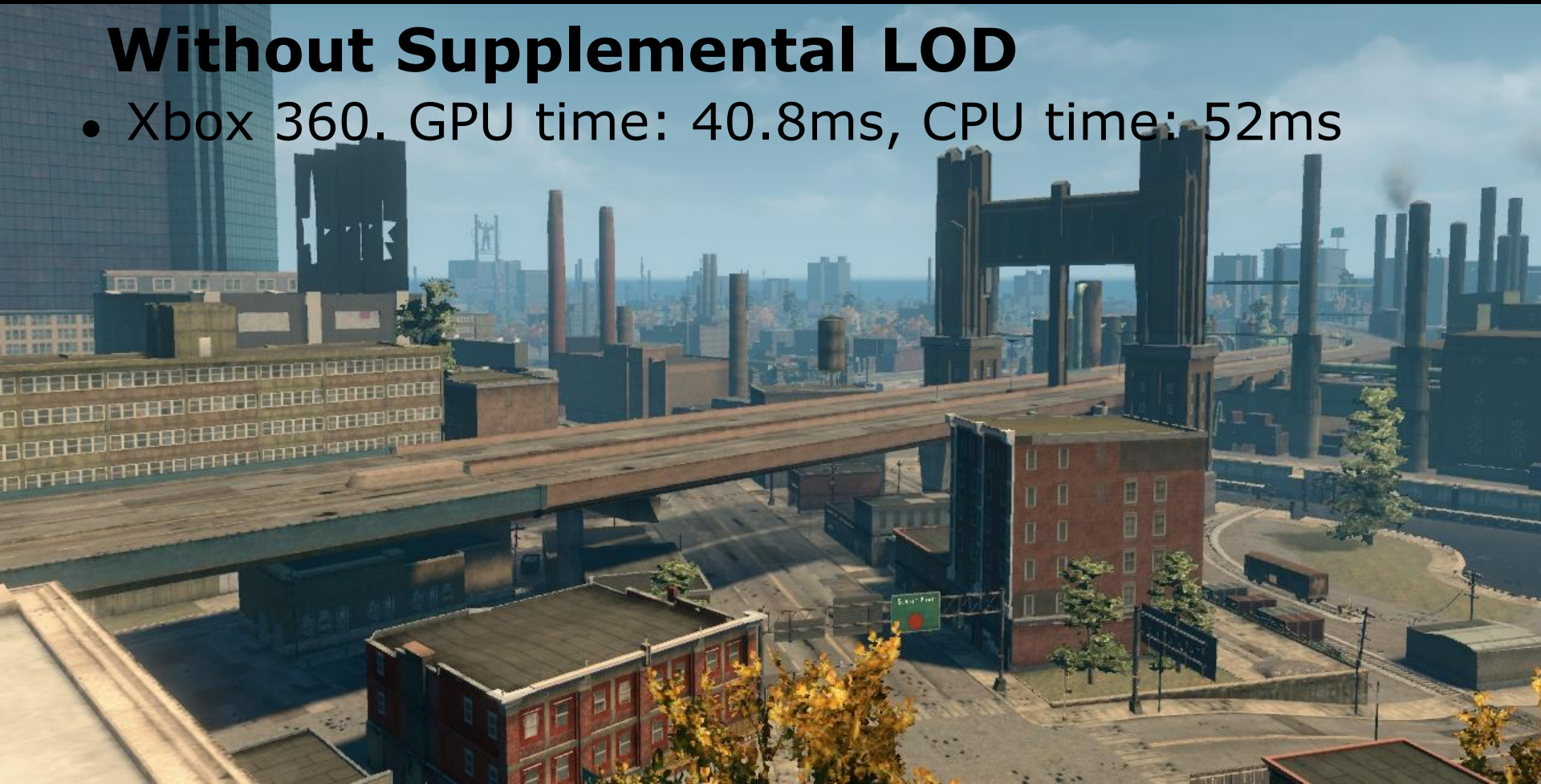
- Actively look for "small" area materials
  - Replace with larger material used on same mesh
  - Reduces count a bit, but not huge savings

Supplemental LOD

- Bake each streamable zone into single mesh
  - Simplify even more (around 5% of original verts)
  - Replace almost all materials with vertex coloring

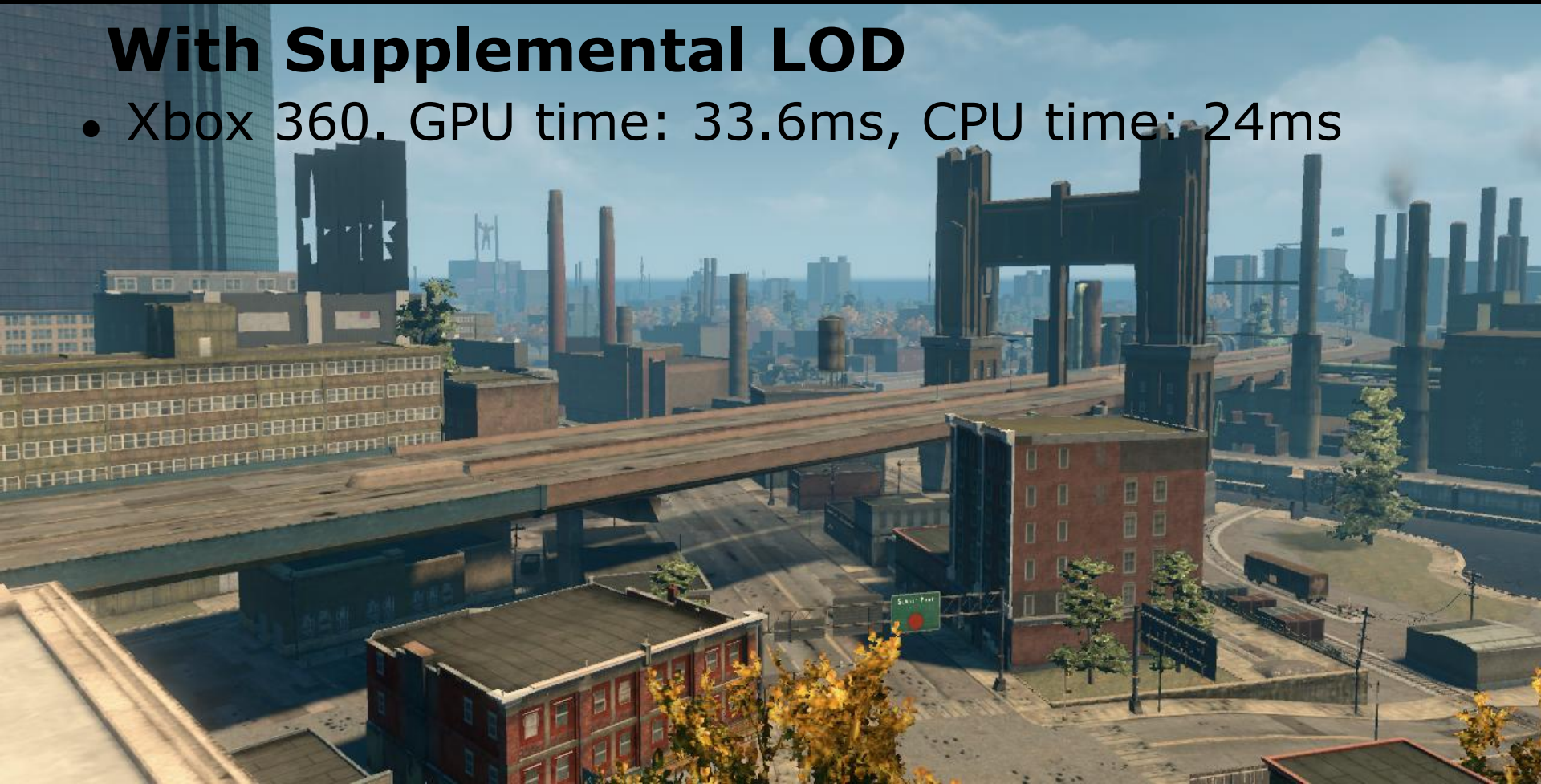
Without Supplemental LOD

- Xbox 360. GPU time: 40.8ms, CPU time: 52ms



With Supplemental LOD

- Xbox 360. GPU time: 33.6ms, CPU time: 24ms



Practical Implementation Issue #4: Artists

- LODs for SR:TT are almost entirely automatic
  - Some intervention by artists may be necessary

Artist Intervention

- Adjust simplifier settings (target count, etc…)

Parameters

CTG Mesh Simplifier

Defaults/Mode

- CTG Defaults
- SR3 Building Medium
- SR3 Building Low
- SR3 Vehicle Medium
- SR3 Vehicle Low
- SR3 Char. Medium
- SR3 Char. Low
- SR3 Char. Extra

Discard Custom Settings

Refresh

Percent Detail: 30.0

Pre-simp weld (cm): 1.0

Importance Weights

NOTE: Weights are relative to importance of maintaining geometric shape (which always has a weight of 1.0)

Boundary 8.0

UV Discontinuity 1.0

Material Discontinuity 2.0

Sharp Edges 2.0

Options

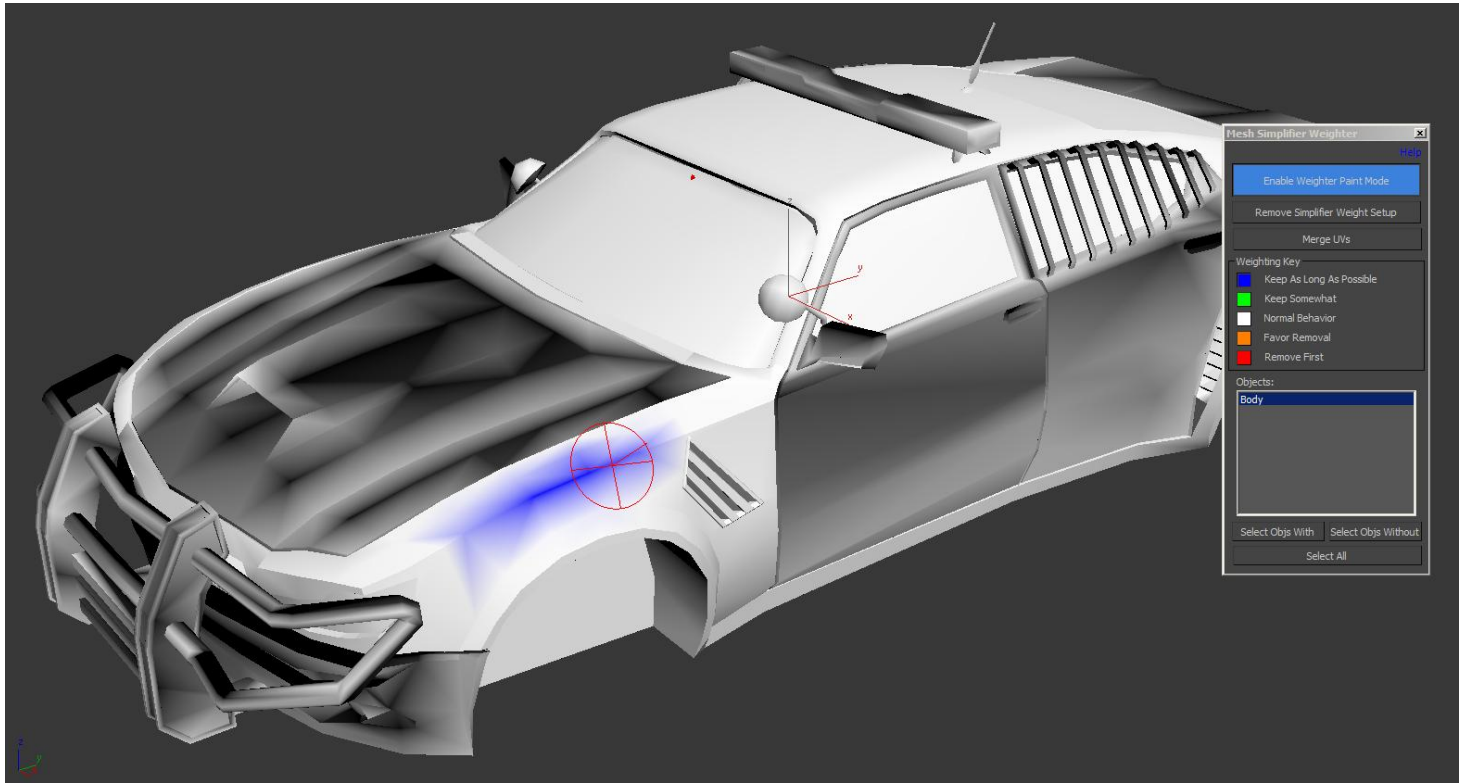
- Preserve Topology
- Prevent Triangle Flipping
- Completely Lock Boundary
- Weight Boundary Faces
- Use Only Original Vertices
- Area-Relative Weighting

CTG Mesh Simplifier Plugin

Scott Kircher

Artist Intervention

- Paint weights or “hints” to help simplifier

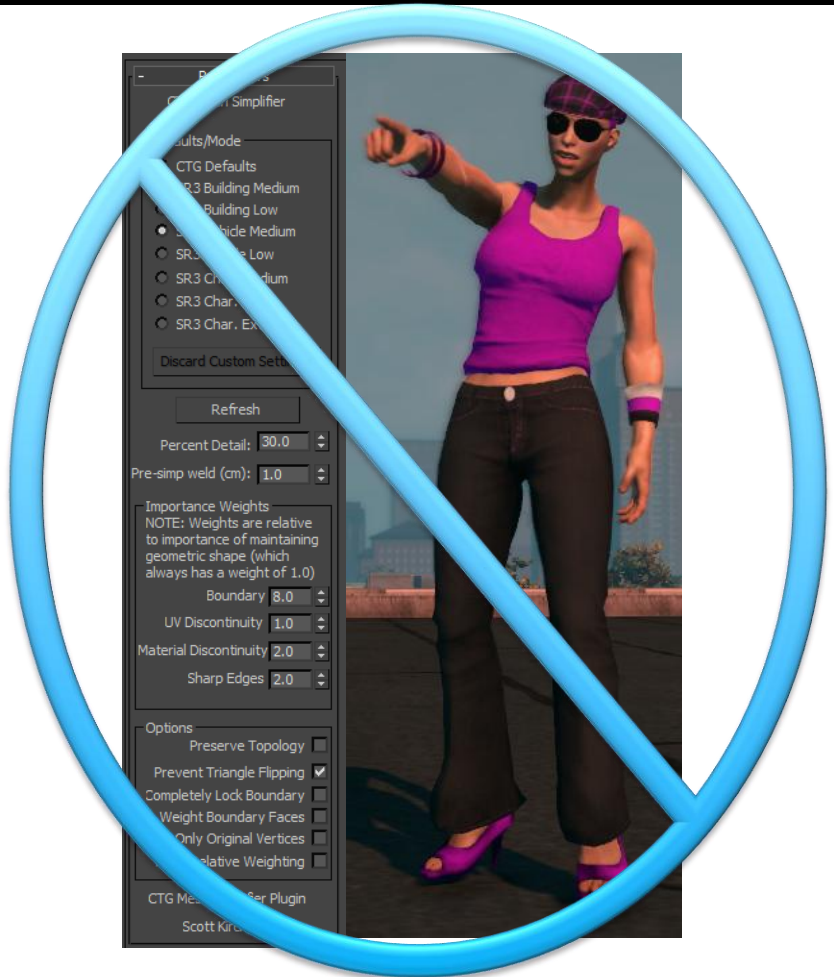


Artist Intervention

- Replace a particular LOD wholesale
  - Used sparingly, only on the “low” LOD for buildings

Artist Intervention

- No direct artist control over:
  - Supplemental LOD
  - Terrain LODs
  - Terrain collision proxy



Summary

- New inferred lighting features
  - Lit rain
  - Faster Discontinuity Sensitive Filter for foliage
  - Object-specific screen-space decals
  - Radial Ambient Occlusion
- Automatic LOD generation practical issues
  - Ill-conditioned quadric matrices
  - UV boundaries
  - Material count reduction
  - Artist intervention

Questions?

References

- Scott Kircher, Alan Lawrance, *Inferred lighting: Fast dynamic lighting and shadows for opaque and translucent objects,* Proceedings of the 2009 ACM SIGGRAPH Symposium on Video Games
  - http://dl.acm.org/citation.cfm?id=1581080
- Mike Flavin, *Lighting the Apocalypse: Rendering Techniques in Red Faction: Armageddon*, GDC 2011
  - http://www.gdcvault.com/play/1014525/Lighting-the-Apocalypse-Rendering-Techniques
- Wolfgang Engel, *Light Pre-Pass Renderer*, Blog post in 2008
  - http://diaryofagraphicsprogrammer.blogspot.com/2008/03/light-pre-pass-renderer.html
- Mark Lee, *Prelighting*, R&D post in 2008
  - http://www.insomniacgames.com/prelighting/
- Jason Mitchell, Gary McTaggart, Chris Green, *Shading in Valve's Source Engine*, Advanced Real-Time Rendering in 3D Graphics and Games Source – SIGGRAPH 2006
  - http://www.valvesoftware.com/publications/2006/SIGGRAPH06_Course_ShadingInValvesSourceEngine.pdf
- Perumaal Shanmugam, Okan Arikan, *Hardware accelerated ambient occlusion techniques on GPUs*, Proceedings of the 2007 symposium on Interactive 3D graphics and games
  - http://www.okanarikan.com/Papers/Entries/2007/5/23_Hardware_Accelerated_Ambient_Occlusion_Techniques_on_GPUs.html
- Michael Garland, Paul Heckbert, *Surface Simplification Using Quadric Error Metrics*, Proceedings of SIGGRAPH 1997
  - http://www.mgarland.org/papers/quadrics.pdf
- Hugues Hoppe, *New Quadric Metric for Simplifying Meshes with Appearance Attributes*, IEEE Visualization 1999 Conference
  - http://research.microsoft.com/en-us/um/people/hoppe/proj/newqem/