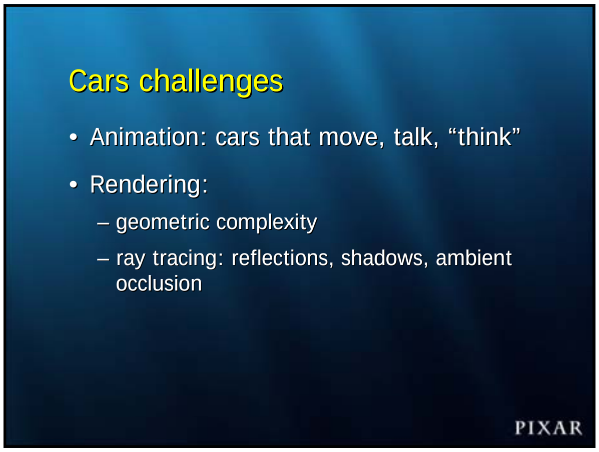
# Ray tracing for the movie ‘Cars’

Per Christensen

Pixar Animation Studios

Ayia Napa Seminar, June 2006

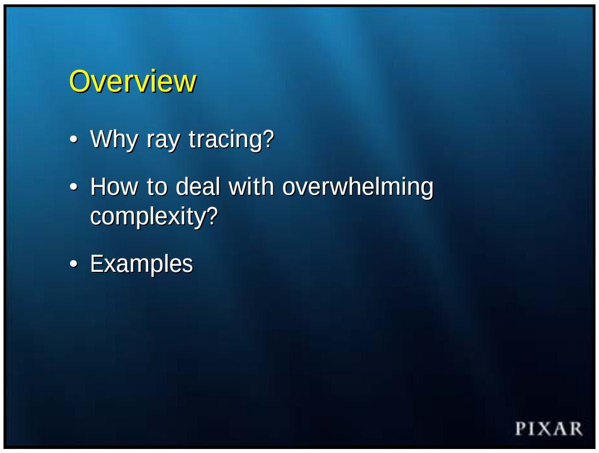
PIXAR



## Cars challenges

- Animation: cars that move, talk, “think”
- Rendering:
  - geometric complexity
  - ray tracing: reflections, shadows, ambient occlusion

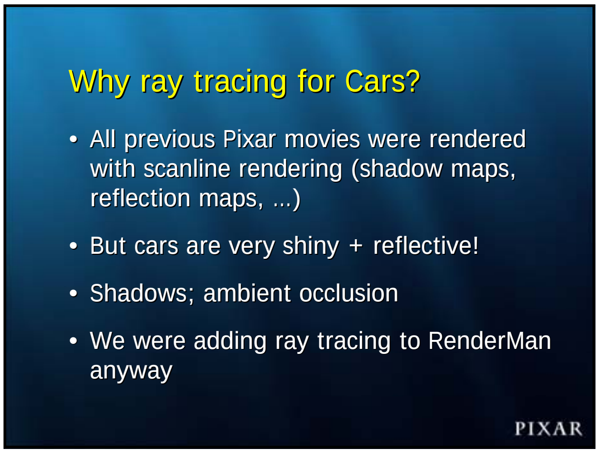
PIXAR



## Overview

- Why ray tracing?
- How to deal with overwhelming complexity?
- Examples

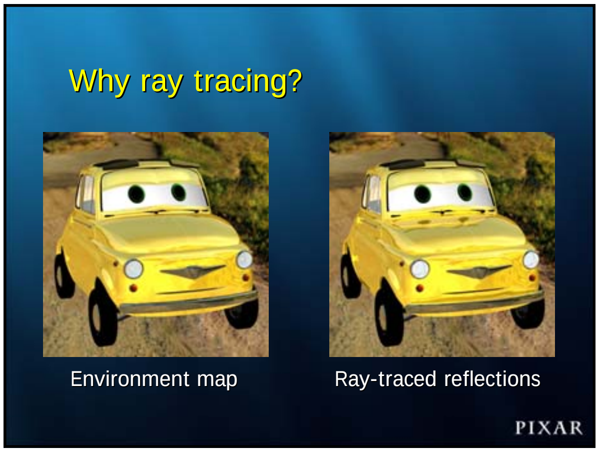
PIXAR



## Why ray tracing for Cars?

- All previous Pixar movies were rendered with scanline rendering (shadow maps, reflection maps, …)
- But cars are very shiny + reflective!
- Shadows; ambient occlusion
- We were adding ray tracing to RenderMan anyway

PIXAR

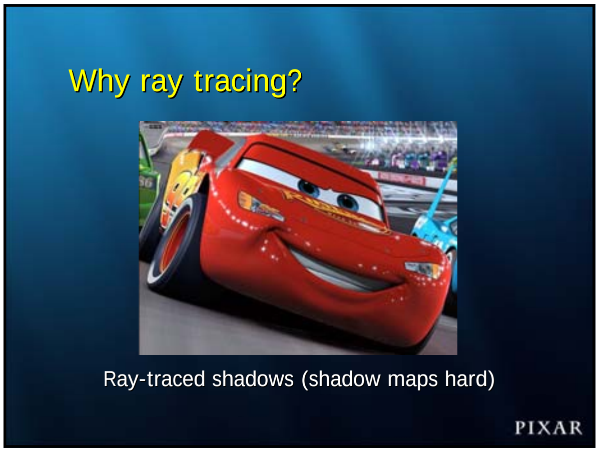


## Why ray tracing?

Environment map

Ray-traced reflections

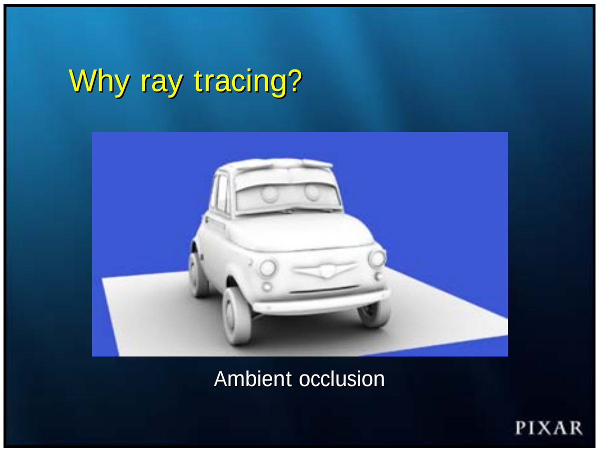
PIXAR



## Why ray tracing?

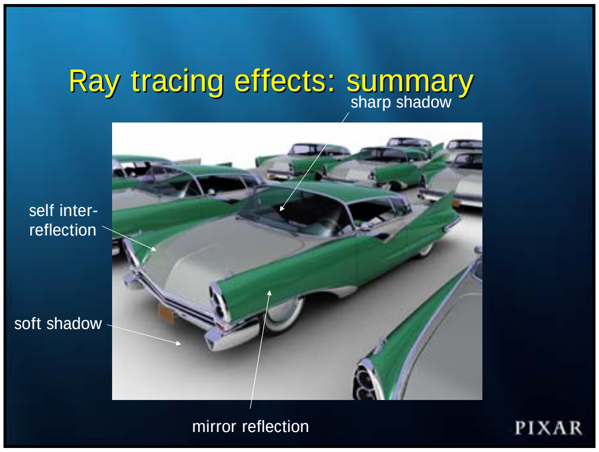
Ray-traced shadows (shadow maps hard)

PIXAR



## Why ray tracing?

Ambient occlusion



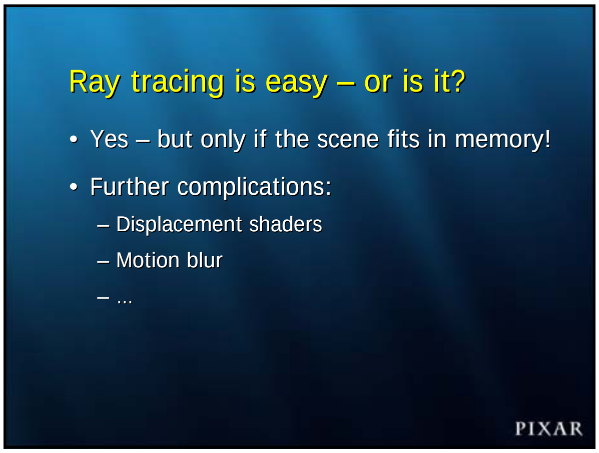
## Ray tracing effects: summary

sharp shadow

self inter-reflection

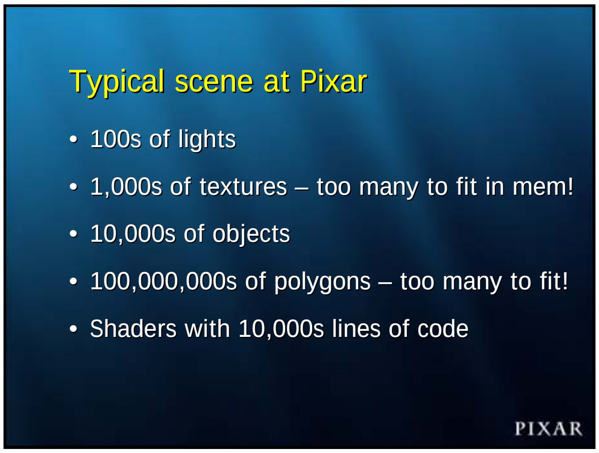
soft shadow

mirror reflection



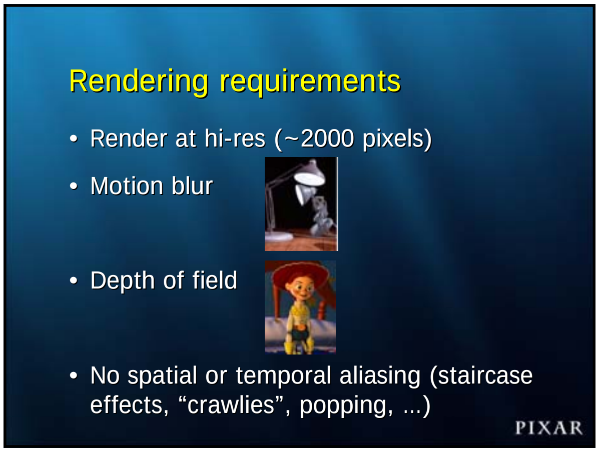
## Ray tracing is easy – or is it?

- Yes – but only if the scene fits in memory!
- Further complications:
  - Displacement shaders
  - Motion blur
  - …



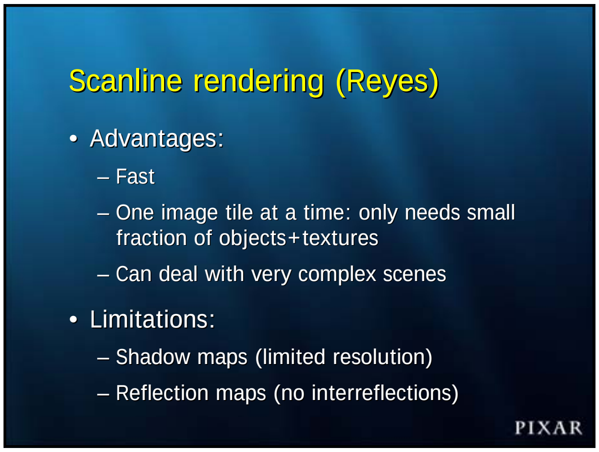
## Typical scene at Pixar

- 100s of lights
- 1,000s of textures – too many to fit in mem!
- 10,000s of objects
- 100,000,000s of polygons – too many to fit!
- Shaders with 10,000s lines of code



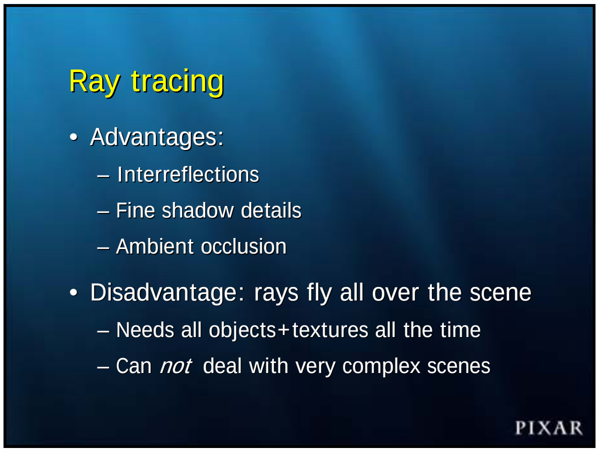
## Rendering requirements

- Render at hi-res (~2000 pixels)
- Motion blur
- Depth of field
- No spatial or temporal aliasing (staircase effects, “crawlies”, popping, …)



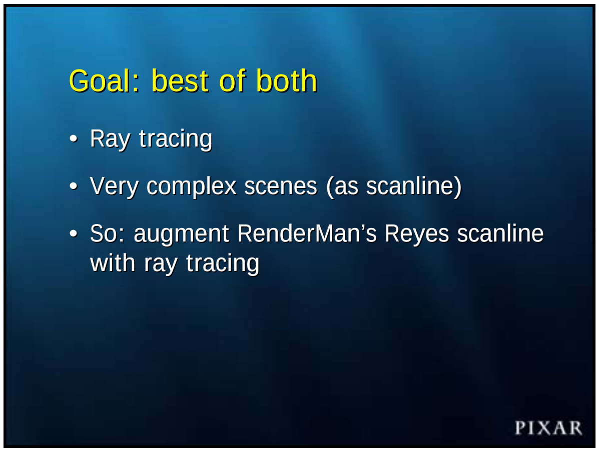
## Scanline rendering (Reyes)

- Advantages:
  - Fast
  - One image tile at a time: only needs small fraction of objects+textures
  - Can deal with very complex scenes
- Limitations:
  - Shadow maps (limited resolution)
  - Reflection maps (no interreflections)



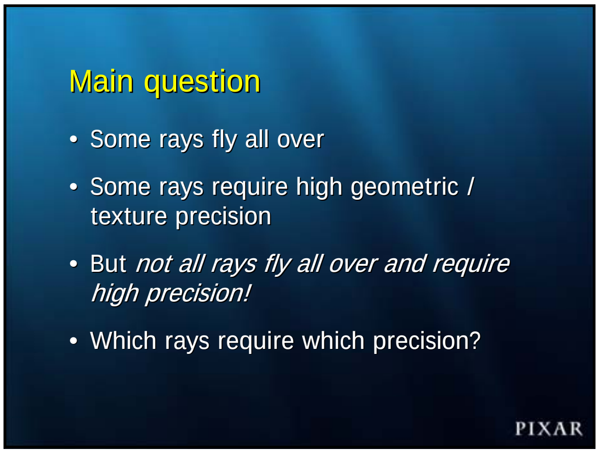
## Ray tracing

- Advantages:
  - Interreflections
  - Fine shadow details
  - Ambient occlusion
- Disadvantage: rays fly all over the scene
  - Needs all objects+textures all the time
  - Can *not* deal with very complex scenes



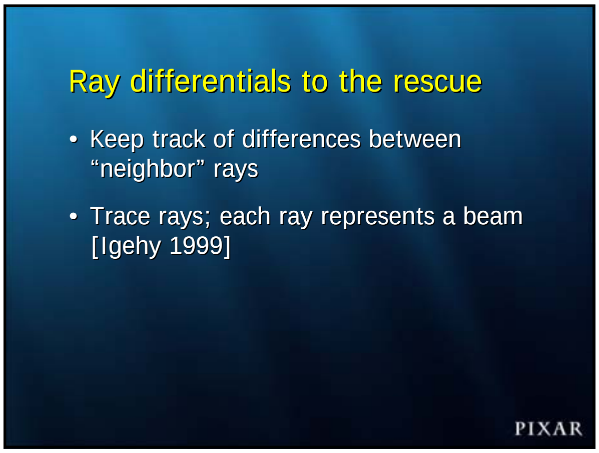
## Goal: best of both

- Ray tracing
- Very complex scenes (as scanline)
- So: augment RenderMan’s Reyes scanline with ray tracing



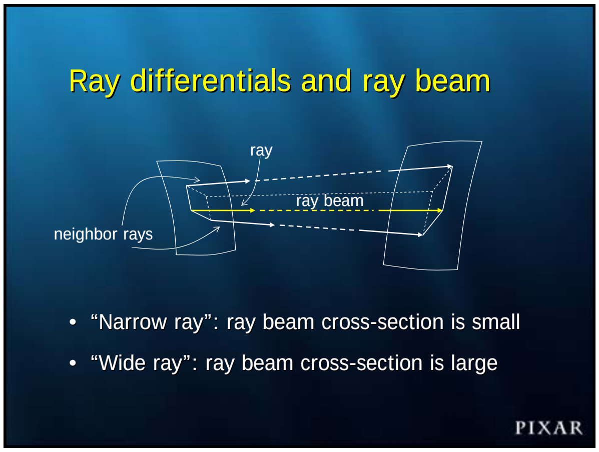
## Main question

- Some rays fly all over
- Some rays require high geometric / texture precision
- But *not all rays fly all over and require high precision!*
- Which rays require which precision?



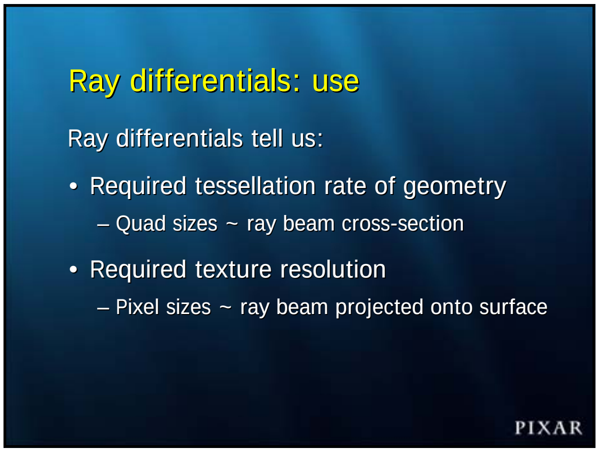
## Ray differentials to the rescue

- Keep track of differences between “neighbor” rays
- Trace rays; each ray represents a beam [Igehy 1999]



## Ray differentials and ray beam

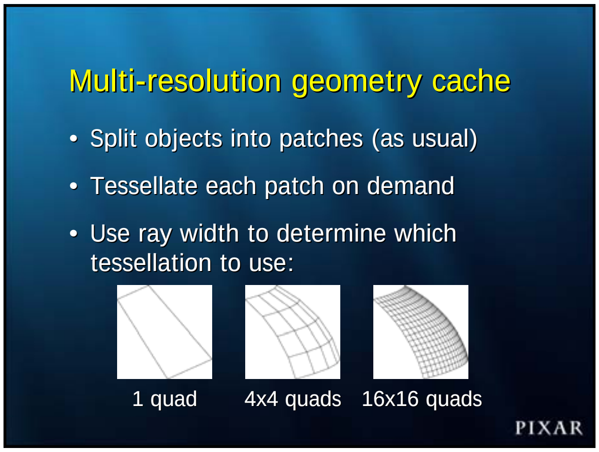
- “Narrow ray”: ray beam cross-section is small
- “Wide ray”: ray beam cross-section is large



## Ray differentials: use

Ray differentials tell us:

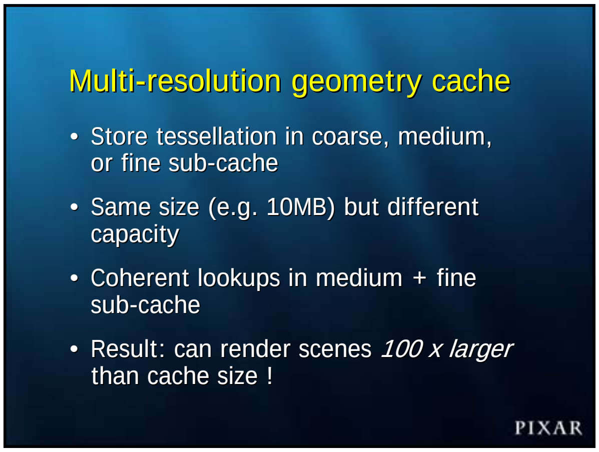
- Required tessellation rate of geometry
  - Quad sizes ~ ray beam cross-section
- Required texture resolution
  - Pixel sizes ~ ray beam projected onto surface



## Multi-resolution geometry cache

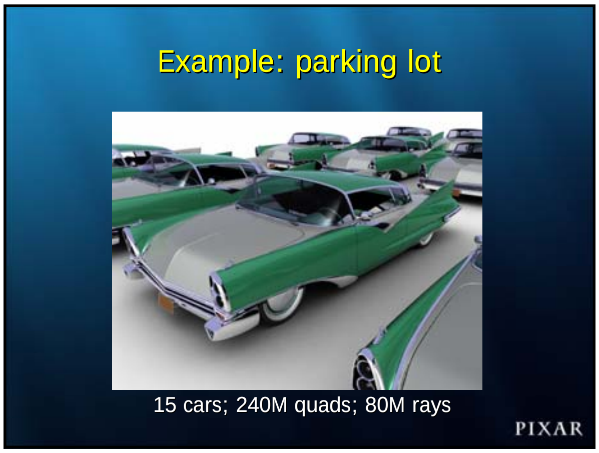
- Split objects into patches (as usual)
- Tessellate each patch on demand
- Use ray width to determine which tessellation to use:

1 quad
4x4 quads
16x16 quads
PIXAR



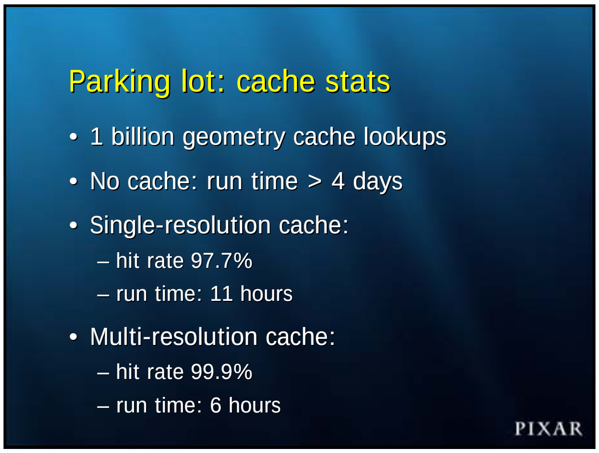
## Multi-resolution geometry cache

- Store tessellation in coarse, medium, or fine sub-cache
- Same size (e.g. 10MB) but different capacity
- Coherent lookups in medium + fine sub-cache
- Result: can render scenes *100 x larger* than cache size !



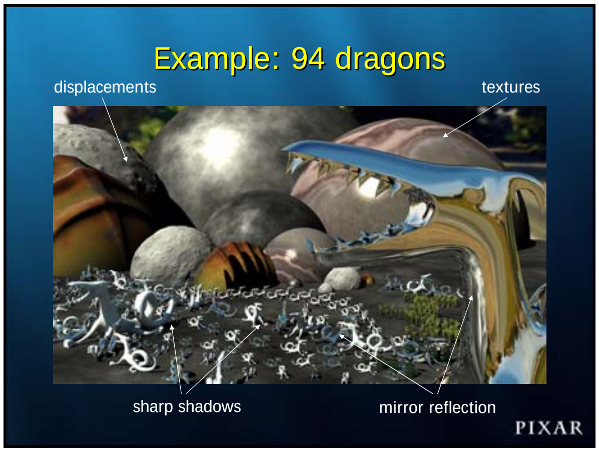
## Example: parking lot

15 cars; 240M quads; 80M rays

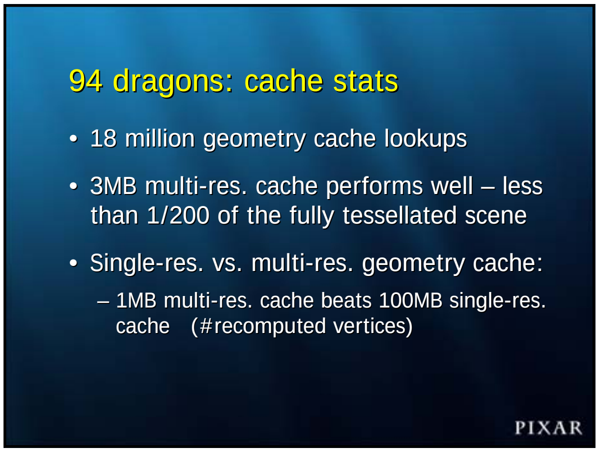


## Parking lot: cache stats

- 1 billion geometry cache lookups
- No cache: run time > 4 days
- Single-resolution cache:
  - hit rate 97.7%
  - run time: 11 hours
- Multi-resolution cache:
  - hit rate 99.9%
  - run time: 6 hours



## Example: 94 dragons



## 94 dragons: cache stats

- 18 million geometry cache lookups
- 3MB multi-res. cache performs well – less than 1/200 of the fully tessellated scene
- Single-res. vs. multi-res. geometry cache:
  - 1MB multi-res. cache beats 100MB single-res. cache (#recomputed vertices)

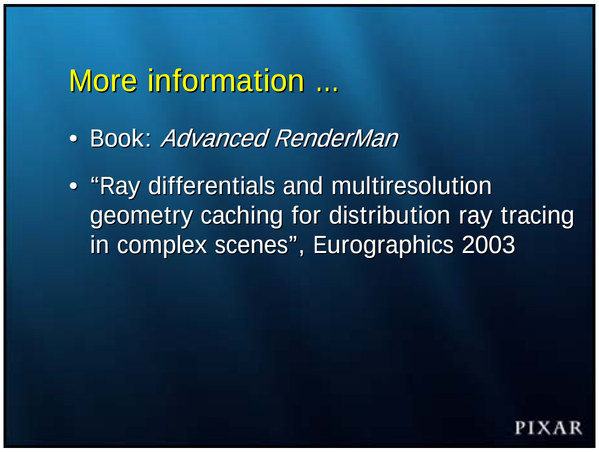
### Final car images ...

### Final car images ...

### Final car images ...

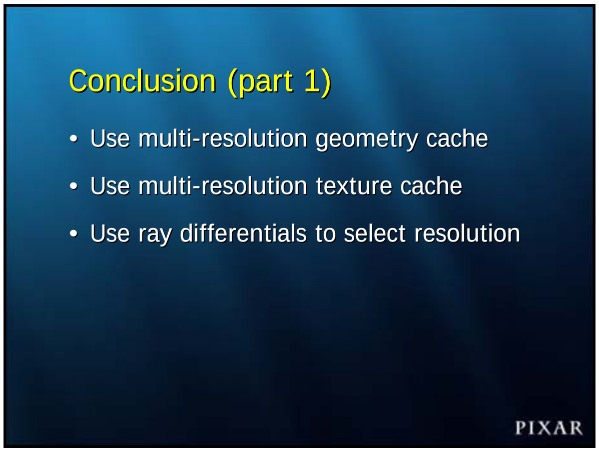
### Final car images ...

### Movie time



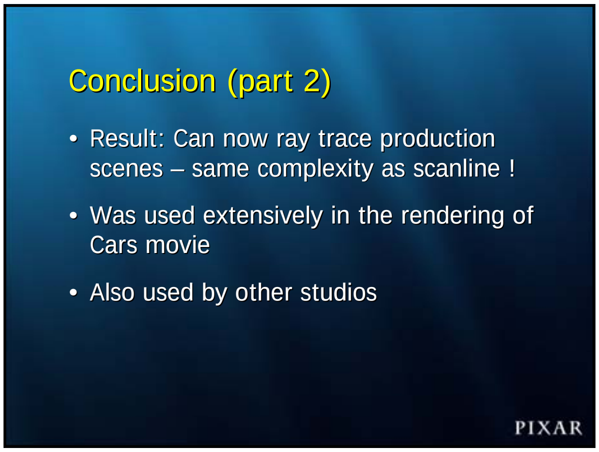
### More information ...

- Book: *Advanced RenderMan*
- "Ray differentials and multiresolution geometry caching for distribution ray tracing in complex scenes", Eurographics 2003



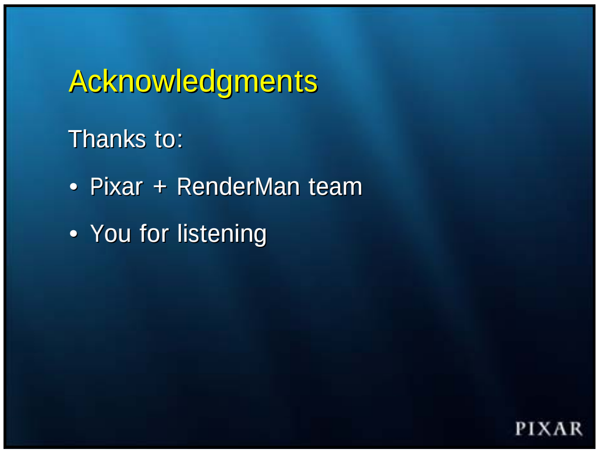
## Conclusion (part 1)

- Use multi-resolution geometry cache
- Use multi-resolution texture cache
- Use ray differentials to select resolution



## Conclusion (part 2)

- Result: Can now ray trace production scenes – same complexity as scanline !
- Was used extensively in the rendering of Cars movie
- Also used by other studios



## Acknowledgments

Thanks to:

- Pixar + RenderMan team
- You for listening

*Questions?*