A million pixels, a million polygons.
Which is heavier?

François X. Sillion

iMAGIS*

Grenoble, France

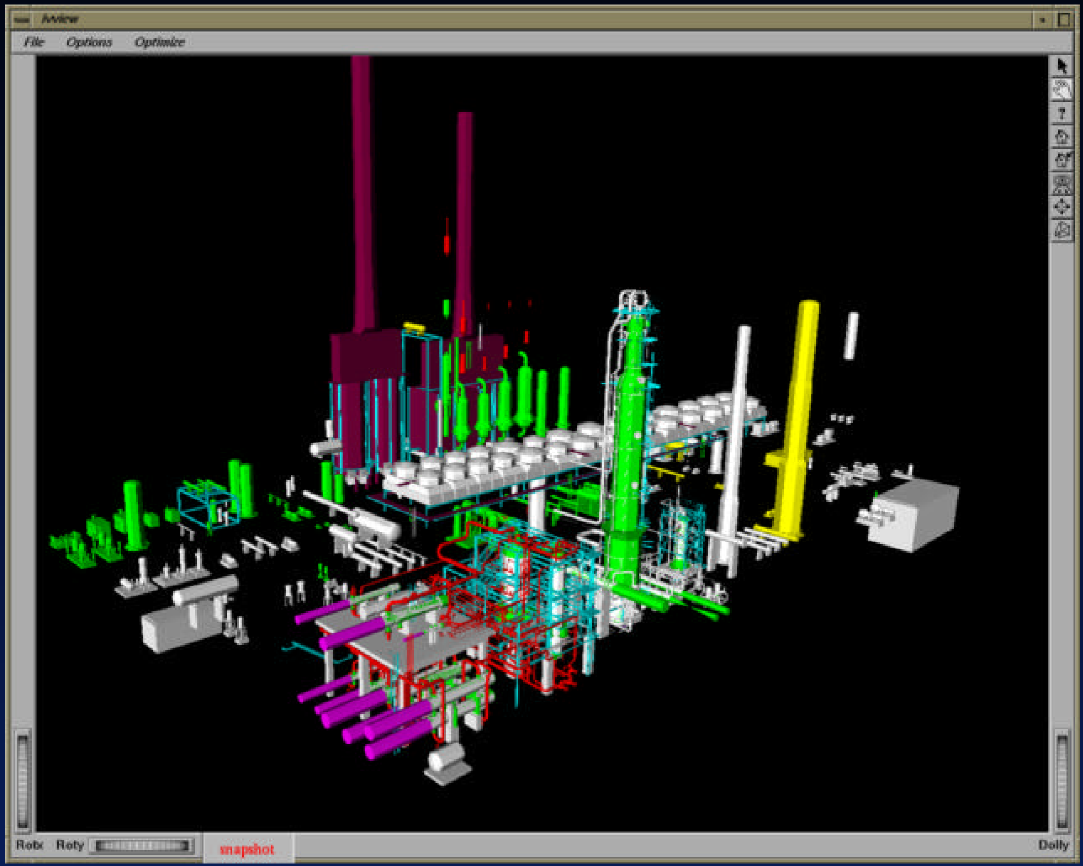
*A joint research project of CNRS, INRIA, INPG and UJF

i MAGIS

Why this question?

- ✓ Evolution of processing power and architectures
- ✓ New applications, demands and markets
  - Giant databases (digital mock up)
  - virtual reality, games...
- ✓ Image-based graphics:
  - current state and trends
  - potentialities

A million polygons



Who needs a million polygons?

- ✓ Assemblies of CAD models
- ✓ Integrated design/manufacturing
- ✓ Digital mock-ups

A million pixels

Rendering in Computer Graphics

- ✓ Models for 3D geometry, light reflection
- ✓ Global illumination simulation
- ✓ Real-time rendering

All of these requirements
present difficult challenges !

Subtle illumination effects

Real-time rendering for dynamic scenes

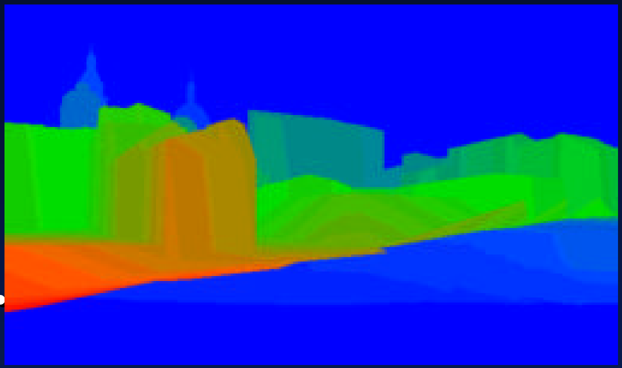
Image-based rendering (IBR)

- Avoid expensive/difficult 3D model
- Start from a set of images
- Manipulate pixels to create new image
- With real images, elaborate lighting effects are “free”
- QuicktimeVR [Chen95], [Laveau], [McMillan95,97],...

What's an image?

- ✓ array of RGB (α) samples

- ✓ add depth sample



- ✓ add multiple depths, normals… (Layered Depth Image, LDI)

Tour into the picture [Horry 97]

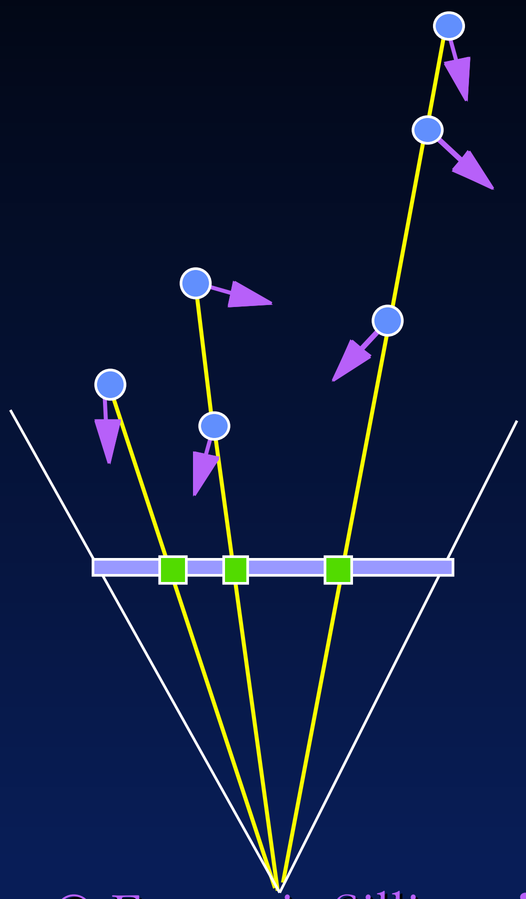
- ✓ Use a single image
- ✓ Manually define simple perspective
- ✓ Manually create layers with selected portions of the image

See http://www-syntim.inria.fr/~horry/images/s97slide.html

Layered depth images [Gortler97]

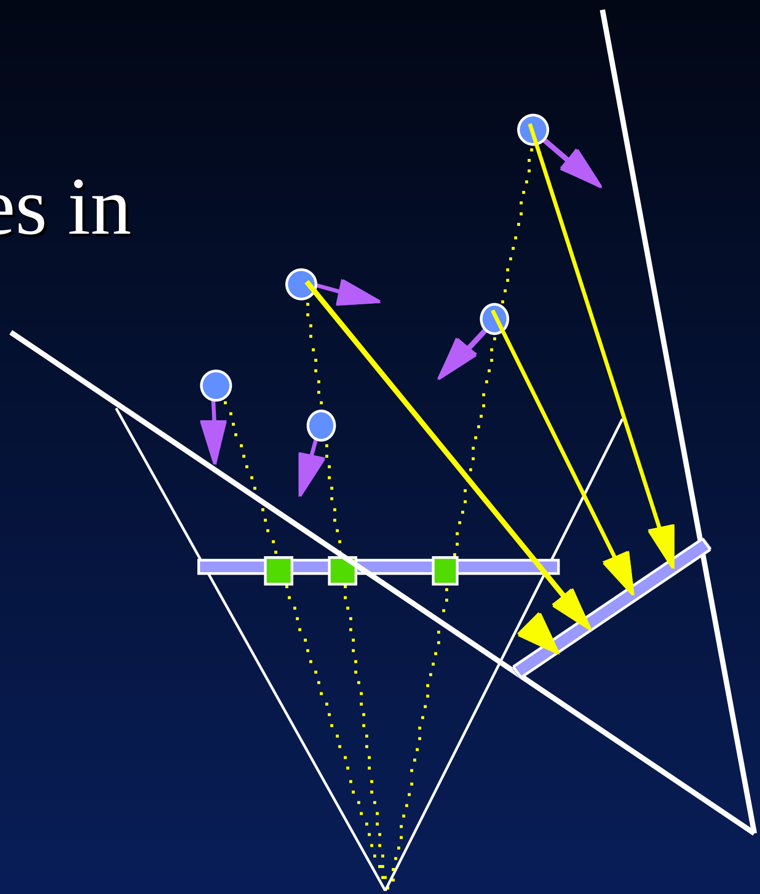
See http://www.research.microsoft.com/research/graphics/cohen/SIG_97_IBR/index.htm

- Gather multiple depth samples for each pixel



Layered depth images

- Reproject all samples in new image
  - no need for depth comparisons
  - splatting technique



Rendering from a million polygons?

✓ Transform 1-3M vertices	20 M flop
✓ Lighting	10 M flop
✓ Texturing	15 M flop
✓ Memory bandwidth	100 Mb
✓ Raster engine, z-buffering	?

Rendering from a million pixels?

- Transform 1M points (coherence) — 6 M flop
- No lighting
- No z-buffering
- Memory bandwith (coherent access) — 8 Mb

Rendering performance considerations

- ✓ 3D rendering reaches the consumer market
  - –thousands of lit,textured polygons / second.
  - –specialized boards require careful design for efficient integration.
- ✓ Image processing subsystems
  - –video (analog/digital),
  - –texture (games),
  - – multimedia extensions

Generating and obtaining IBR models

- ✓ From synthetic images
  - –Ray tracing
  - –Range images, LDIs, Lumigraphs
- ✓ From real images
  - –use panoramic views, vision techniques
  - –feature matching (difficult)
  - –Lumigraphs (no depth)

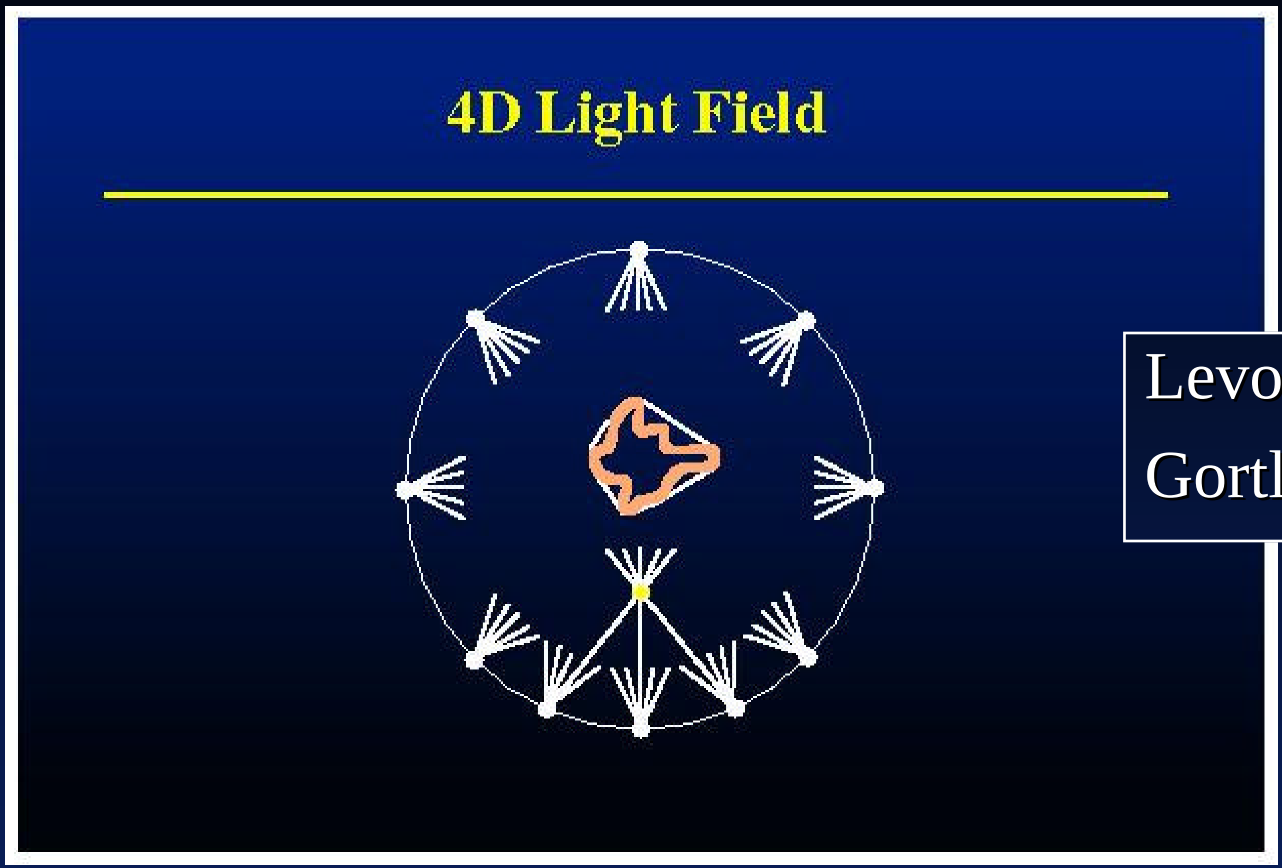
Link with vision

- ✓ Image based modeling (IBM...)
- ✓ Use images + parameters
  - –avoid WYSIAYG
  - –object class information
  - –interactive modeling (facade)

IBR = sampling + reconstruction

- ✓ Operate without geometry
- ✓ More complete representations (higher dimensionality)
- ✓ Simplified representations (adding simplified 3D model)

Light field - Lumigraph



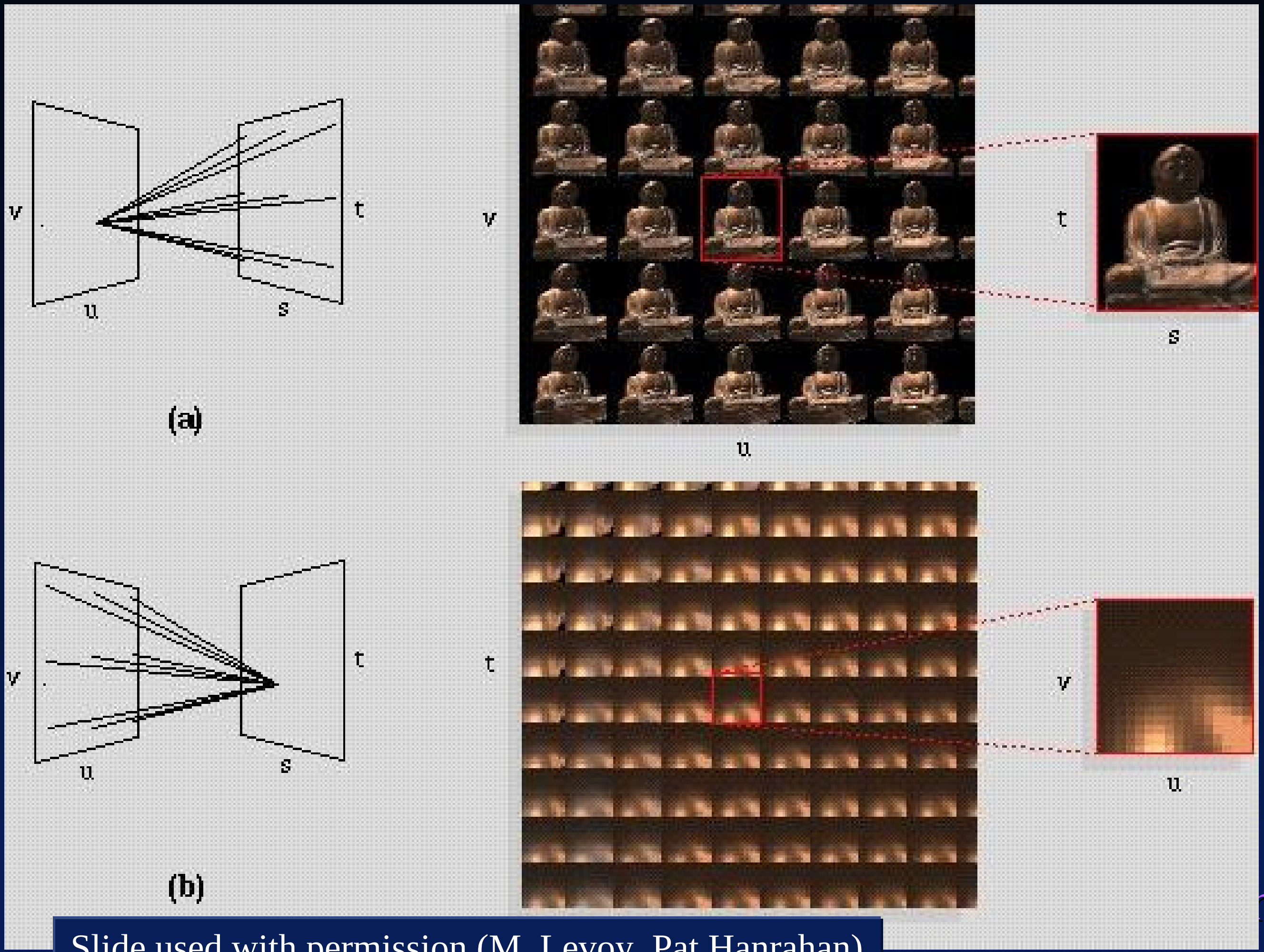
Levoy, Hanrahan 96

Gortler et al 96

Slide used with permission (M. Levoy, Pat Hanrahan)

See http://www-graphics.stanford.edu/projects/lightfield

Light field - Lumigraph sampling



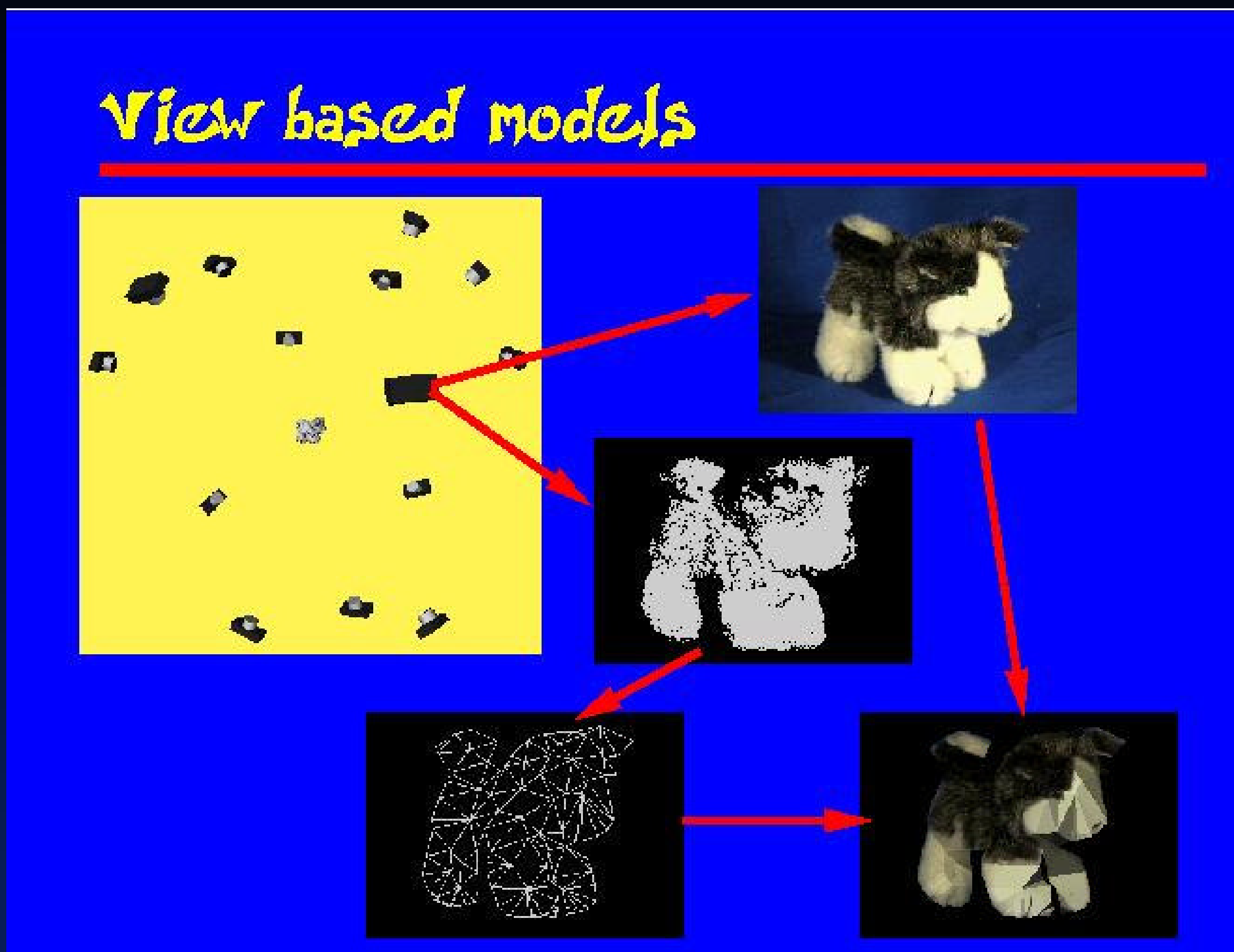
n, iMAGIS 1997

Slide used with permission (M. Levoy, Pat Hanrahan)

Impostors

- ✓ Create textured 3D model from images
  - –*simplified* representation
  - –rendered as 3D geometry
- ✓ Planar polygons [Maciel95, Schaufler96, Shade96]
- ✓ 3D meshes from range images
  [Pulli 97, Darsa 97, Sillion 97]

Textured 3D mesh from a range image



Pulli 97

ion, iMAGIS 1997

Slide used with permission (K. Pulli et al.)

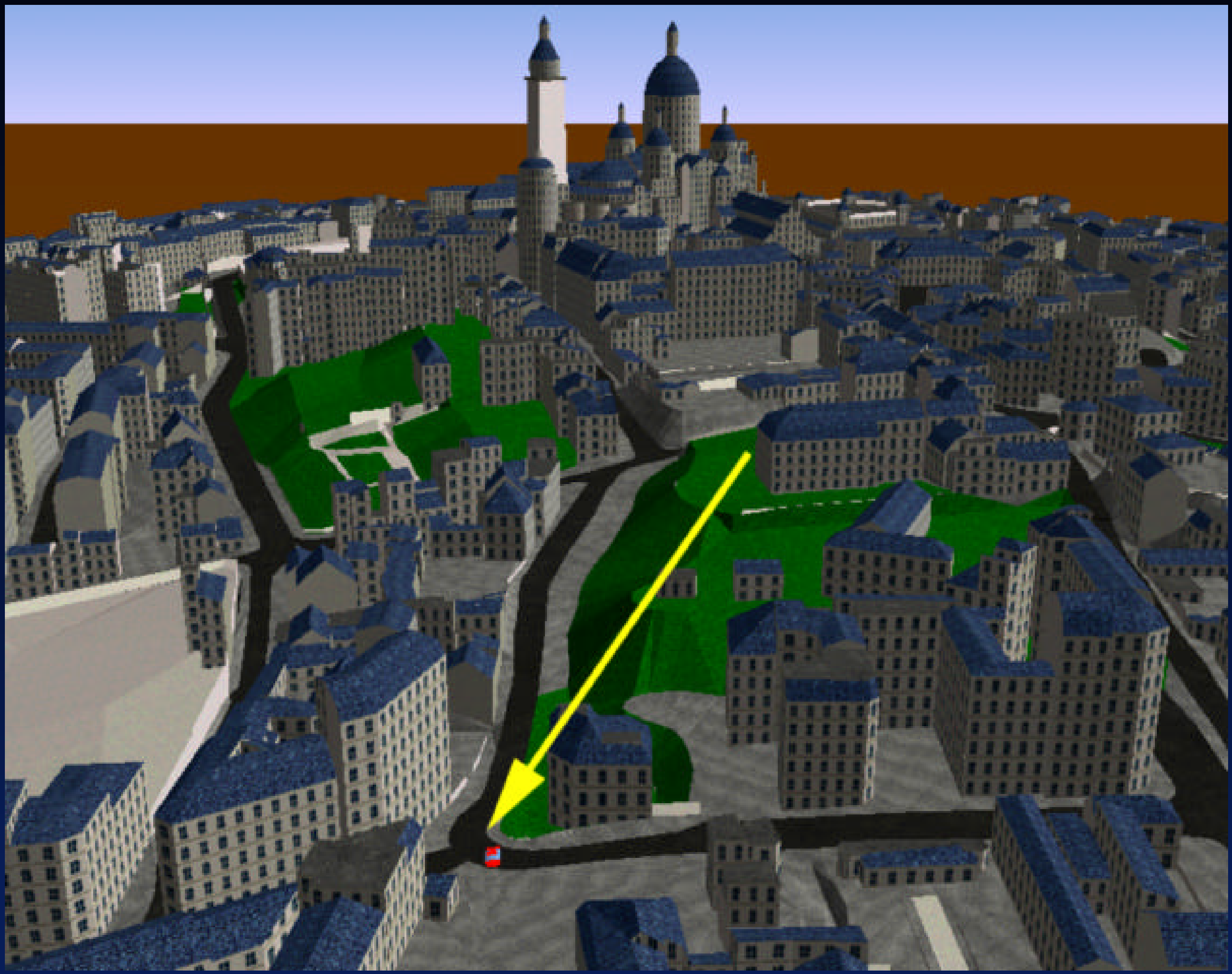
Blending required to combine views

without blending
(z-buffer)

with blending
[Pulli 97]

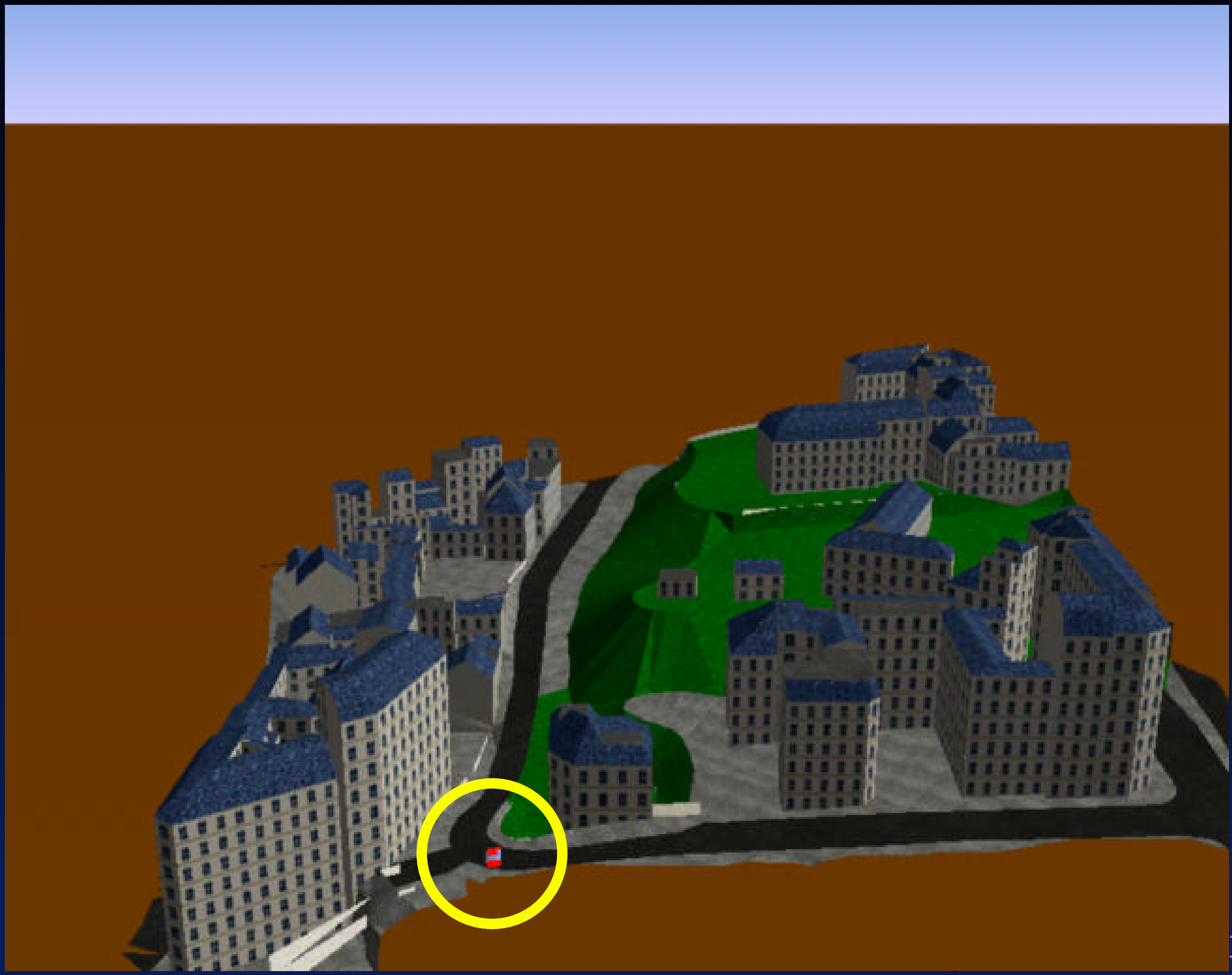
Images used with permission (K. Pulli et al.) See http://www.cs.washington.edu/homes/kapu

Principles of our approach: example



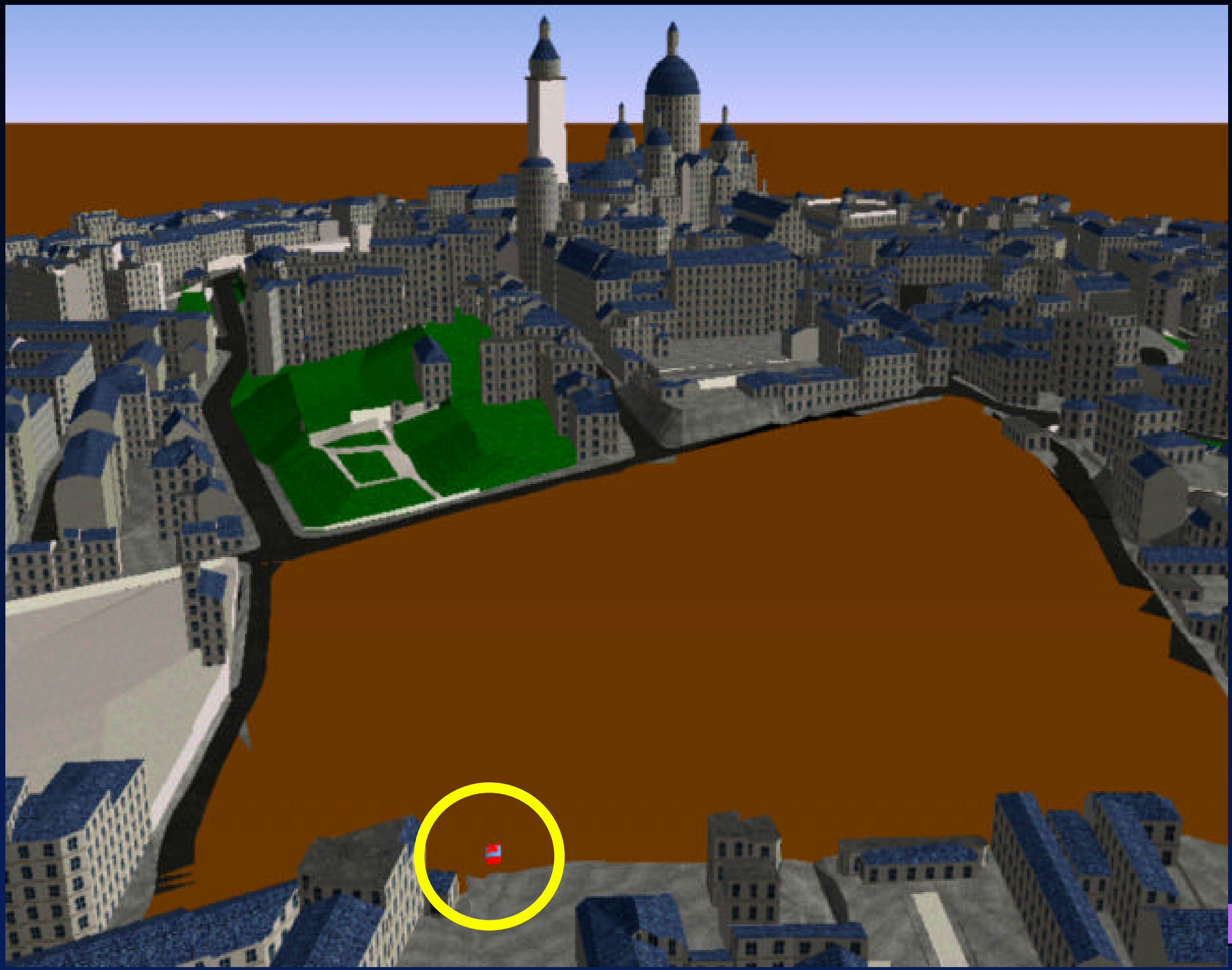
iMAGIS 1997

Local model (3D objects)



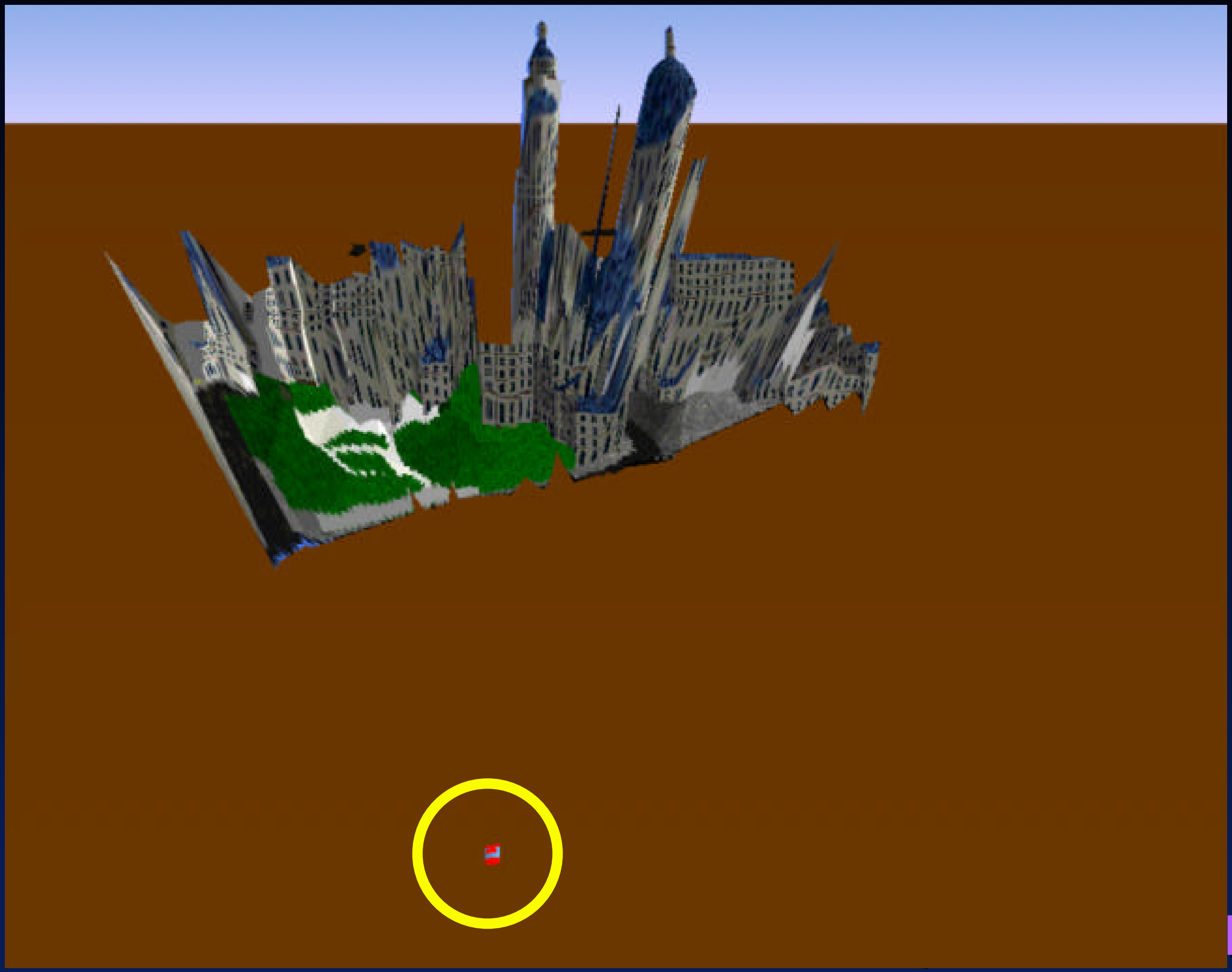
iMAGIS 1997

Distant model (3D objects)



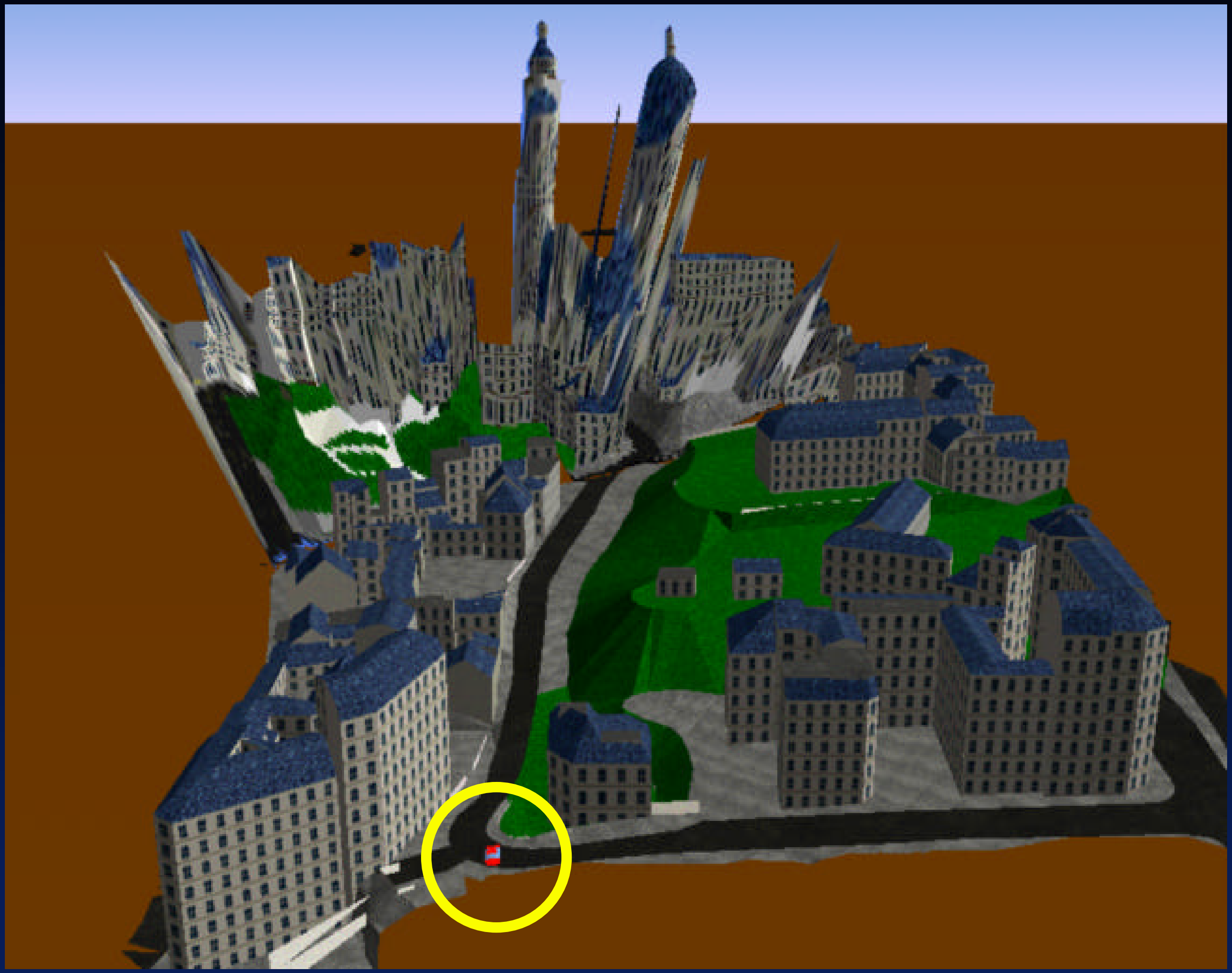
MAGIS 1997

Impostor (Textured 3D mesh)



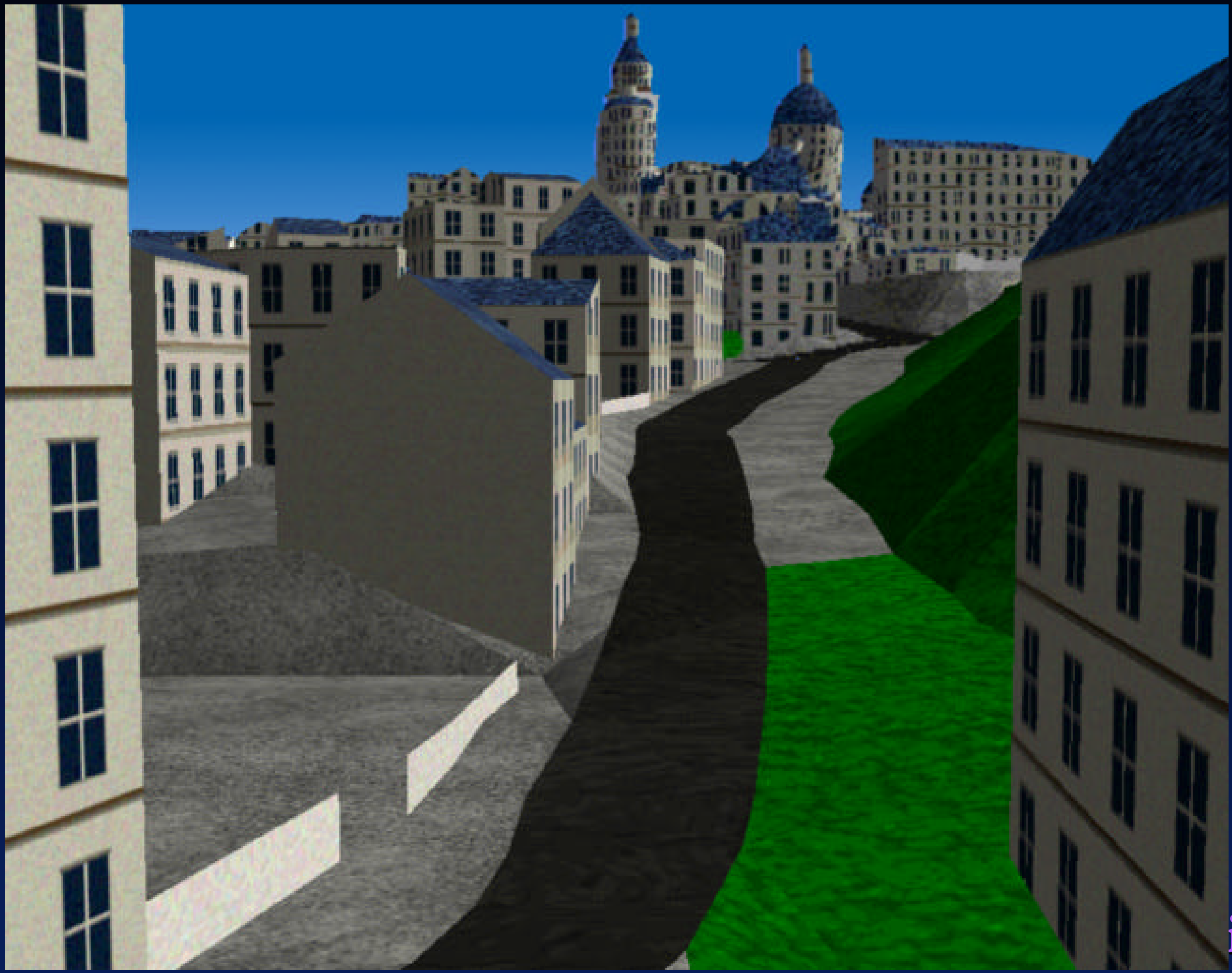
MAGIS 1997

Combined model (local+impostor)



MAGIS 1997

Combined model (local+impostor)



iMAGIS 1997

Deforming impostors

- ✓ Talisman [Torborg 96]
  - –Render sprites
  - –Layered model
  - –Affine transforms [Lengyel97]
- ✓ Impostor transition

Slide used with permission (J. Lengyel et al, Microsoft research.)

See http://research.microsoft.com/~jedl

Applications for IBR

- ✓ Walkthrough / view synthesis
- ✓ Stereo synthesis
- ✓ Interpolation/extrapolation
  - –Latency compensation
  - –Frame rate equalization
  - –Network transmission
  - –Leverage expensive rendering

Polygons	Pixels
✓ Continuous	✓ Discrete
✓ Modeling	✓ Capture
✓ Animation	✓ Video streams
✓ Level of detail	✓ Filtering

Pixels

- ✓ Discrete, regular nature
  - –easy to filter: adaptation to user perceptual limitations
- ✓ Work with real images
  - –Easy to capture
  - –Let nature do the modeling/lighting
  - –Work from existing images (historical, legal, forensic applications...)
- ✓ WYSIAYG

Polygons

- ✓ Complete 3D model
  - –solid modeling
  - –global illumination
  - –path planning, assembly checking, collision detection
- ✓ Common denominator for many modeling systems
- ✓ Can be simplified but it's hard to keep the model consistent

Extended notion of image-based models

- Use *both* images and 3D data
- Combine a simplified model with images
- model can be extracted from images or other information

IBR and availability of 3D models

- Complete 3D model
  - –IBR as graphics subsystem
- No 3D model
  - –QTVR, plenoptic rendering
  - –The model *is* the image(s)
- Range data available
  - –Scanned data is huge: need to simplify

Problems with current algorithms

- ✓ Holes in reconstructed images
- ✓ Image deformation (impostors)
- ✓ Volume of data
- ✓ Sampling/filtering artifacts

Can we expect hardware advances?

- ✓ view interpolator
- ✓ soft z-buffering and blending
- ✓ multiple or view-dependent textures
- ✓ decompression
- ✓ memory bandwith

Limitations of IBR

- ✓ Specularities
- ✓ Lighting/geometry/reflectance changes are hard
- ✓ Computer Vision issues: model building
- ✓ Images may not be available!

Marketability

- ✓ QTVR, panoramic images
- ✓ Image-based modeling
- ✓ Image-based rendering architectures
- ✓ Image caching, impostors
- ✓ Network applications (QoS)
- ✓ Light field

...and now?

- ✓ Simulation of global illumination
- ✓ Visibility calculations
- ✓ View-dependent texture mapping
  - –disparity/depth
  - –specularity/shading
  - –re-lighting
- ✓ Compression of depth values

Computer-augmented reality

input panorama

computed solution

Drettakis 97

iMAGIS 1997

Conclusions

- ✓ IBR offers useful advances
  - –leverage cost of high-quality rendering
  - –fast extension via specialized subsystem
- ✓ Vision issues limit applicability of “pure” IBR for real images
- ✓ Use combined 3D models and images
- ✓ Polygons are still useful!

Acknowledgements

- Yann Argotti, George Drettakis, Frédo Durand, Peter Kipfer, Céline Loscos, Stéphane Moreau, Cyril Soler
- Michael F. Cohen, Steven Gortler
- Marc Levoy, Pat Hanrahan, Kari Pulli, Jed Lengyel, Youichi Horry, Ken-ichi Anjyo, Kiyoshi Arai