Temas Avanzados

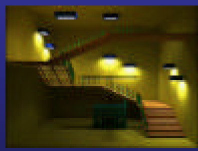
- Entornos dinámicos
- Iluminación natural
- Image Based Rendering (IBR)

IIiA - GGG

Introduction

Radiosity sequences are too expensive if each frame is treated independently

example:



– 4200 polygons
– For a half-minute animation:
10(min/fr).30(fr/sec).30(sec) = 150 hours!

IIiA - GGG

Animated Environments

- Definition:
  – Environments where the number of objects, their shape, location or surface attributes, may change in time
- Changing surface attributes
  – Only optical properties as emittance or reflectance changes ➔ simplest case, no visibility changes
- Changing geometry
  – Objects can change position and shape

IIiA - GGG

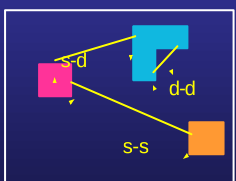
Animated Environments

- Changing geometry
  – Visibility changes must be detected and recomputed efficiently

IIiA - GGG

Animated Environments

- Changing geometry
  – Visibility changes must be detected and recomputed efficiently



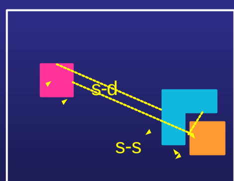
s: static

d: dynamic

IIiA - GGG

Animated Environments

- Changing geometry
  – Visibility changes must be detected and recomputed efficiently



s: static

d: dynamic

IIiA - GGG

Animated Environments

- Classification by Usability
  - Interactive algorithms:
    - For a required change the illumination must be updated interactively.
    - Examples: interactive design, virtual reality
  - High-quality rendering animation:
    - Many frames must be computed with a high level of accuracy.
    - Examples: high-quality realistic animation

IIiA - GGG

Animated Environments

- Interactive techniques
  - Incremental radiosity:
    - [Chen 90, George et al. 90, Muller 94]
    - Formulate a radiosity expression in time
    - Use progressive refinement methods to update incrementally the radiosity after a modification
    - Problem: Consider static meshing
  - Hierarchical radiosity:
    - [Shaw 97, Drettakis and Sillion 97]
    - Update links interaction in time
    - Use dynamic meshing according to lighting changes

IIiA - GGG

Animated Environments

- Animation
  - Back-buffer algorithm
    - [Baum 86]
  - Image-based framework
    - [Nimmeroff et al. 96]
  - Hierarchical radiosity
    - [Forsyth et al. 94, Dámez and Sillion 98]
  - Frame-to-frame coherence
    - [Martín et al. 99]

IIiA - GGG

Animated Environments

- Important aspects to consider:
  - Visibility changes management
  - Memory storage
  - View dependency
  - Speed-Up
  - Theoretical Foundation

IIiA - GGG

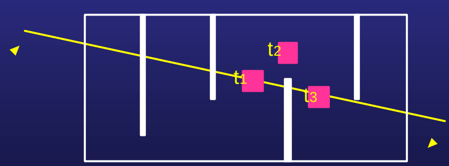
The Multi-Frame Lighting Method

- Outline:
  - Computes a previously-known animated sequence
  - All frames of the sequence are processed in a single simulation
  - Use global lines to manage visibility changes
  - Based on Monte Carlo radiosity
  - Resulted frames have the same accuracy as if they were treated independently
  - Completely view independent

IIiA - GGG

The Multi-Frame Lighting Method

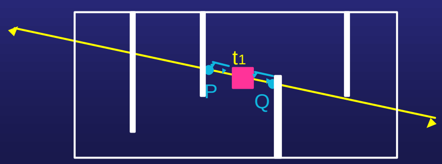
- Global methods and visibility changes
  - Global lines can encode visibility information for different times



IIiA - GGG

The Multi-Frame Lighting Method

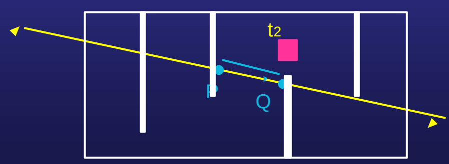
- Global methods and visibility changes
  - Changes in visibility are detected processing the visibility list



IIiA - GGG

The Multi-Frame Lighting Method

- Global methods and visibility changes
  - Changes in visibility are detected processing the visibility list



IIiA - GGG

The Multi-Frame Lighting Method

- Description of the method
  - Pre-processing step
  - Lighting computation step
  - Visualization step

IIiA - GGG

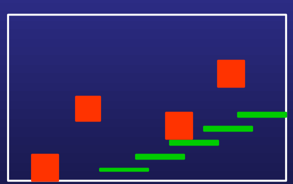
The Multi-Frame Lighting Method

- Pre-processing step
  - Prepare the animated model for the lighting computation
  - Compute and store all matrix transformations for all frames
  - The aim of this step is to have quickly access to any dynamic information

IIiA - GGG

The Multi-Frame Lighting Method

- Lighting computation step

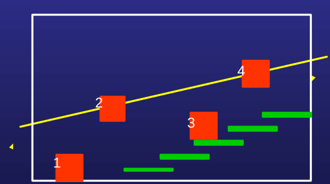


IIiA - GGG

The Multi-Frame Lighting Method

- Lighting computation step
  - Intersection

Cast global lines and intersect all object at all frames



IIiA - GGG

The Multi-Frame Lighting Method

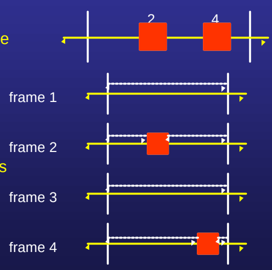
- Lighting computation step

Visibility Lists

Sort and split for each frame

Light energy transfer

Compute power exchanges for each visibility list



IIiA - GGG

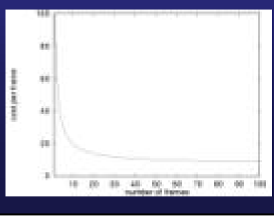
The Multi-Frame Lighting Method

- Analysis
  - Cost
    - Depends on parameters of the animation:
      - the number of static surfaces
      - the number of dynamic surfaces
      - the size of the motion volume where the motion takes place
      - the distribution of the objects is space

IIiA - GGG

The Multi-Frame Lighting Method

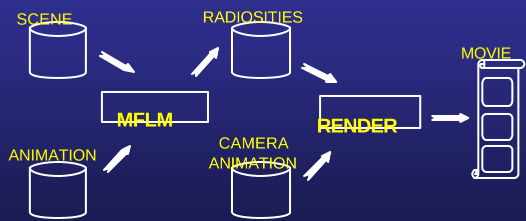
- Analysis
  - Cost
    - From a general analysis:
      - Cost per frame decreases as the number of frames grows
      - The efficiency is low bounded



IIiA - GGG

The Multi-Frame Lighting Method

Implementation



IIiA - GGG

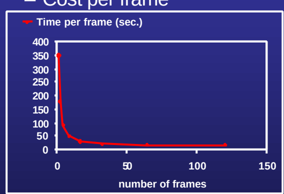
The Multi-Frame Lighting Method

- Usability
  - Splitting the animation
    - Estimate the number of frames for the maximum efficiency
    - Split the sequence in sub-animations
    - Process each sub-sequence independently
    - Distribution among different processors can be used

IIiA - GGG

The Multi-Frame Lighting Method

- Result tests
  - Cost per frame



- Bound frame → 60
- Speed-up → 17
- Time per frame → 25 seconds

IIiA - GGG

Interpolated-Based Acceleration

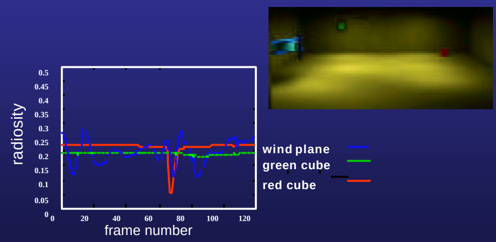
Use temporal coherence to accelerate the Multi-Frame Lighting Method (MFLM)

- Outline
  - Estimate the radiosity function over time in the sequence
  - Examine the gradient and determine important time points
  - Compute the sequence at this points using the MFLM
  - Obtain in-between frames by interpolation

IIiA - GGG

Interpolated-Based Acceleration

- Temporal Coherence in Radiosity
  - Sample sequence:



IIiA - GGG

Interpolated-Based Acceleration

- Radiosity estimation over time

Set a high minimum area for patches discretization

Compute the sequence using the MFLM

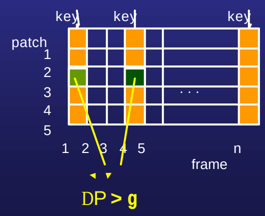
Obtain, in a short time, a coarse radiosity solution over time

IIiA - GGG

Interpolated-Based Acceleration

- Setting radiosity key-frames
  - Analyze the radiosity gradient over time
  - Use a conservative algorithm to set key-frames:

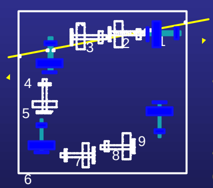
A power variation over a defined threshold sets a key



IIiA - GGG

Interpolated-Based Acceleration

- Computing the sequence
  - Compute with MFLM only for the key-frames



- Obtain in-between frames by interpolation
- Interpolation are performed at object space for each patch

IIiA - GGG

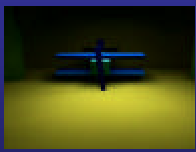
Interpolated-Based Acceleration

- Analysis
  - Improve both cost and storage
  - Cost:
    - Less objects are intersected
    - The cost of the estimation is about one-hundred of the full computation
  - Storage:
    - Less memory ➔ only full computed frames are stored

IIiA - GGG

Interpolated-Based Acceleration

- Results



Full sequence

Interpolated sequence

Relative speed-up: 1.4

Absolute speed-up: 36

Otros resultados

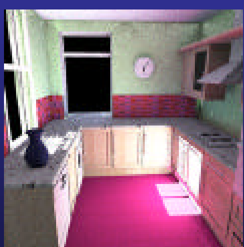
Iluminación natural

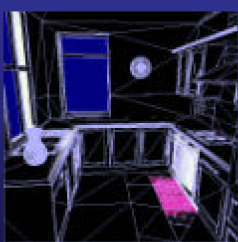
- Modelado de sol y cielo
  - CIE overcast

$$L(q) = L_z \frac{1+2\cos q}{3}$$

    - L_z Luminancia en el zenit
    - modelo empírico y poco preciso
  - CIE clear sky
    - modelo más complejo
- Complejidad de cálculo

Mesh produced by sun





With anisotropic diffusion

With anisotropic diffusion

IliA - GGG

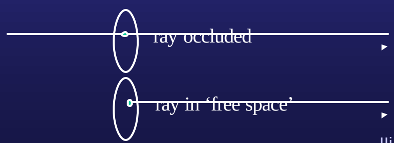
Image Based Rendering

- Introduction
- Rendering
- Representing Light Fields
- Practical Issues
- Conclusions

IliA - GGG

Introduction

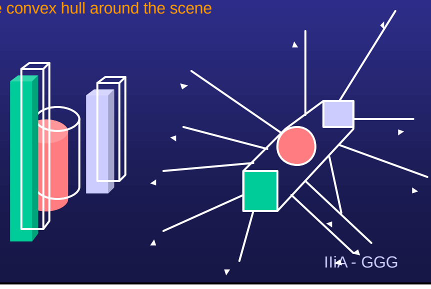
- L(p, ω) is defined over the set of all all rays (p, ω) with origins at points on surfaces
  - Domain of L is 5D ray space
- L is constant in 'free space', space free of occluders:



IliA - GGG

Introduction: Free Space

- Radiance is constant along each ray in free space emanating from the convex hull around the scene

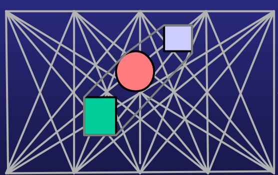


IliA - GGG

Introduction: Restrict L to 4D

- In free space the domain of L can be restricted to 4D
- L(s,t,u,v) can represent all rays between two parallel planes – a 'light slab'.

uv plane



Here radiance is flowing in direction uv to st.

st plane

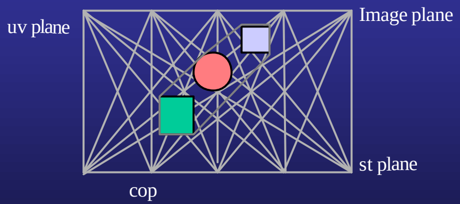
IliA - GGG

Introduction: Light Field = 6 Light Slabs

- 6 light slabs can represent rays in all directions:
  - Near-far, Far-near
  - Left-right, Right-left
  - Bottom-top, Top-bottom
- Discretisation provides approximation to 'all rays'.

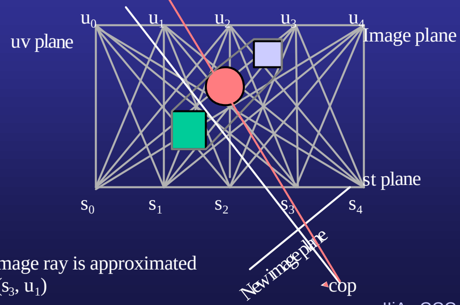
IliA - GGG

Introduction:Assigning Radiance to Each Ray



- Place the cop at each grid point on st plane
- Use uv plane as an image plane.
- Each ray is assigned a radiance.

Introduction: Synthesising a New Image



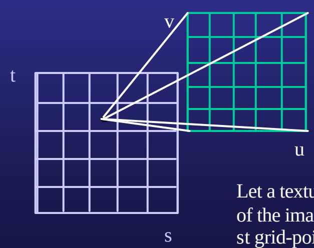
The image ray is approximated By $L(s_3, u_1)$

Summary

- Light field is a discrete representation of L restricted to the 4D free space domain
- It relies on production of a large set of images under strict viewing conditions
- Most often used for virtual images of real scenes – based on digital photographs
- Can be used for synthetic scenes – not much point unless globally illuminated
- Supports real-time walkthrough of scenes

Rendering

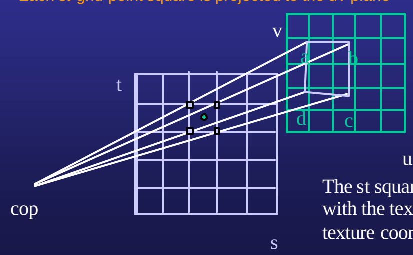
- Each st grid-point is associated with a uv image



Let a texture map consisting of the image be associated with st grid-point.

Rendering: Texture Mapping

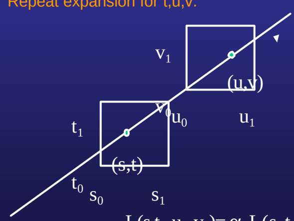
- Each st grid-point square is projected to the uv plane



The st square is rendered with the texture map and texture coords abcd

Rendering: Quadrilinear Interpolation

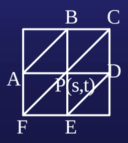
- Perform linear interpolation on s
- Repeat expansion for t,u,v.



$$L(s,t_0,u_0,v_0)= \alpha_0 L(s_0,t_0,u_0,v_0)+ \alpha_1 L(s_1,t_0,u_0,v_0)$$

Efficient Rendering: Linear-Bilinear Interpolation

- Gortler shows that texture mapping cannot be used to reproduce quadrilinear interpolation, but linear (on st), bilinear (on uv).



For the grid-point P render each of the triangles ABP, BCP, CDP, DEP, EFP, FAP with alpha-blending and texture mapping:
α=1 at P and α=0 at each vertex.

IIiA - GGG

Practical Issues: Resolution

- What resolution should be used for st, uv?
  - st ($M \times M$) can afford to be sampled at lower rate, since the accuracy depends on uv ($N \times N$), close to scene surfaces.
  - Typical values are M=32, N=256
    - for synthesis of 256×256 resolution images

IIiA - GGG

Practical Issues: Memory

- Suppose N=256, M=32
  - Each light slab requires $2^8 \times 2^8 \times 2^5 \times 2^5 = 2^{26}$ rays
  - Each ray carries 3Bytes (RGB)
  - There are 6 light slabs
- Full representation therefore requires:
  - $6 \times 3 \times 2^{26} >$ 1Gb memory
- Compression scheme needed (vector quantization used in practice).

IIiA - GGG

Conclusions

- Light Fields approach offers a different paradigm with many disadvantages, but interesting possibilities:
  - Exploits current hardware for a brute-force ‘solution’ of radiance equation.
  - Image of real scenes or synthetic scenes (and combinations).
  - Rendering time independent of scene complexity.
  - Promise of real-time walkthrough for global illumination

IIiA - GGG