Illuminating with YafRay

Alejandro Conty Estévez
conty@yafray.org
The purpose

- We will study the behaviour of yafray
- Controlling it from blender
  - Always under method = “full”
  - What is the meaning of some elements under this mode
- Full method
  - Yafray takes into account direct and indirect light
  - What is cache and how it works
  - What are photons and how they work
The origin of light

- Light from punctual lights
  - Calculated directly using a local model
  - Fast and predictable
- Non-punctual sources
  - Calculated by Monte-carlo method
  - Taking samples from environment shooting rays
  - Slow
  - Powerful and geometry independent
Montecarlo
Montecarlo
Non-punctual lights

- Illumination calculated by Monte-carlo techniques
- It “randomly” samples the environment
  - Similar to Ambient Occlusion
  - It also takes bounces into account
- Possible light sources
  - Background, simple image or HDR
  - Emitting objects
  - Objects that reflect light from other sources
Light from Background
Light from an Illuminated area
The Noise

- Appears when light sources are small and powerful
- The smaller the source, the smaller the probability of finding it by Monte-carlo
- The smaller the probability, the higher the noise
- Typical cases
  - A small emitting object
  - A small window
  - A HDR background with light concentrated in small areas
VISIBILITY: The real problem with noise
Big visible sky: almost no noise

- It is very easy to find the sky with a random ray
Big visible sky render
Less visible sky: some noise

- Now we see only half sky
- We have to double the GI power to get the same result
Less visible sky render
Small visible sky: a lot of noise

- Sky seen by a small aperture
- 8 times the original power
Small visible sky render
Reducing the noise

- Quality settings in yafray panel affect the number of samples
- When light sources are small, more samples help MC to find them
- Render time increases linearly with number of samples
- Noise decreases as $\frac{1}{\sqrt{\text{samples}}}$
- Use it carefully
  - Help with photons
  - Help with cache refinement
What to do when light sources are small

• This usually means indoor scenes
• Use photons
  • Not caustic photons as in explicit photonlight
• Photons give extra information to Montecarlo method
  • Where does the light come from?
• We skip the bounce stage
  • Photons give us an estimation
We shoot photons from light sources

- Photons bounce and leave “marks”
- Those marks build a photonmap
The photonmap, an approximation
Setting up photons

- Enable “photons” in yafray GI panel
- Choose a **number of photons** and a **radius**
  - Radius means blurring the photonmap
  - Radius depends on the scene size and the precision you want
  - Increasing the number of photons means more precision and less “blotching” (noise)
- What is a good photonmap?
  - It must not have noticeable light leaks
  - As smoothed as in the previous image
Radius $= 1.0$
Radius = 0.7
Radius = 0.4
How montecarlo uses the photonmap?
Common cases where photons can help

- Small illuminated area in the scene
  - We have spot and omni as photon emitters
- Emitting object
  - We have arealight as emitter
- Diffuse light coming in from a window
  - We also use arealights as emitters
Sun getting in from a window

Sun

Small illuminated area

SUN

Small illuminated area
Interior lamp

Small Illuminated area
To use photons, fake the sun

As far as possible
When faking the sun with spots:

• Make sure the cone covers all the window
• Use several spots for several windows
• Make sure cones do not overlap when using several spots
  • It is not mandatory, just minimize overlapping
• Make cones as small as possible
  • Avoids lost useless photons
For interior lamp: spot or omni
• Has a material with $emit \neq 0$
• We place one or more arealights
• They must cover the emitting surface as good as possible
  • For a square plane we can use just one arealight
  • For more complex objects, we can use several arealights along its surface
• Its color must be the same as the surface or a good average
• Power 1.0 always. Tweak GI power if needed
Diffuse light from sky from a window

- Cover the entrance of the window with an arealight
- Direction pointing to inside
- We also can use several arealight for complex windows
  - A simple approximation of the shape is enough
- Its color must be the same as the background
- Power $1.0$ too
Diffuse light through a window
Covering the window with an arealight as close as possible.
So what?

- You have an indoor scene with small light sources
- You choose to use photons
- Make sure every light source can act as emitter
  - Spot and omni do
  - For objects and windows use arealights
- Setup number of photons and radius \((500000, 0.5)\)?
- Adjust with “tune photons” enabled
And …

- Compare the intensity of the photonmap with the low quality non-photon render
- If they do not match:
  - Maybe you forgot to tune the color of an arealight
  - Maybe it is a bug (most likely)
- Sometimes light leaks are due to non-solid modelling
The irradiance Cache. What?

- Avoids calculating GI at every pixel
- We select some pixels to calculate and then interpolate
  - Interpolation dissolves the noise a bit
  - We have low frequency noise (clouds)
- Quite annoying for animations yet (flickering)
- Works fine and saves time for stills
Cache settings: Where?

- Enable “cache” check button to see all settings
- Needs “full” method

![Cache settings interface](image)
Stages of the render

1. Photon shooting stage
   - From each potential emitter it shoots photons
   - Photons are distributed between all of them

2. First render pass (fake pass)
   - Takes samples each “prec” pixels
     - Taking more samples at the corners (concave)
Stages of the render

1. Refinement stage (2 more fake passes)
   - Identify changes in light greater than “refinement”
   - Anti-alias them taking more samples

2. Final render stage
   - Interpolate using the resulting cache
Shadow quality

- Is a value between 0 and 1
- Controls how much does the density of samples grow near corners
  - The higher, the better shadow definition near corners
- Default value 0.9 is good for almost any scene
- Leave improvement to refinement stage
Refinement

- Is a value between 0 and 1
- Any “fast” change in light greater than this value will be “antialiased”
  - Shadow edges
  - Unexpected artifacts due to low sample count
- A value of 1.0 means no refinement, 0.03 means heavy refinement
- This pass is repeated two times
Original render, no refinement
Refinement = 0.05
Can I improve it more?

- Shadows are antialiased now
- But still some blotching around
  - Not located in shadow edges
- What to do?
  - Increase samples (quality)
  - Try to smooth the photonmap a bit more
- Smoothing the photonmap is useless beyond a certain point
  - Excessive smoothing yields light leaks
After increasing quality to high
Smooth enough?
Caustics

- There are two ways of getting caustics in YafRay:
  - From a punctual light source
  - From an object, sky or any non-punctual element
- When it is a punctual light we should use an explicit “photonlight”
- Otherwise:
  - Full method can produce them by itself
  - Watch “CDepth” parameter, increase if needed
  - The source should not be too small
Caustics from a big window
Known problems

- Cache works fine for stills
- In animations you will notice flickering
  - Can be solved by increasing quality
  - Too much render time
- Still no way of shooting photons from an HDR background
- HDR’s with concentrated light will produce lot of noise